HAL
open science

Proportional-Fair Scheduling of Mobile Users based on a Partial View of Future Channel Conditions

Thi Thuy Nga Nguyen

To cite this version:

Thi Thuy Nga Nguyen. Proportional-Fair Scheduling of Mobile Users based on a Partial View of Future Channel Conditions. Automatic Control Engineering. INSA de Toulouse, 2020. English. NNT : 2020ISAT0023 . tel-03351962v2

HAL Id: tel-03351962

https://theses.hal.science/tel-03351962v2

Submitted on 22 Sep 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par l'Institut National des Sciences Appliquées de Toulouse

Présentée et soutenue par

Thi Thuy Nga NGUYEN

Le 30 novembre 2020

Ordonnancement garantissant l'équité proportionnelle des utilisateurs mobiles basé sur une connaissance partielle des conditions futures des canaux

Ecole doctorale : **SYSTEMES**

Spécialité : **Automatique et Informatique**

Unité de recherche :
LAAS - Laboratoire d'Analyse et d'Architecture des Systèmes

Thèse dirigée par
Olivier BRUN et Balakrishna PRABHU

Jury

M. Rachid ELAZOUZI, Rapporteur
Mme Hind CASTEL, Rapporteure
Mme Isabel AMIGO, Examinatrice
M.Alain PIROVANO, Examinateur
M. Olivier BRUN, Directeur de Thèse
M. Balakrishna PRABHU, Co-directeur de thèse
M. Bruno COUDOIN, invité

Acknowledgments

I would like to take this opportunity to say thank so much to my two supervisors, Bala and Olivier, for their guidance during my thesis. Thank so much Mr. Bruno Coudoin for his help during the past three years. Thank very much the members of my thesis committee for spending their time reading the manuscript and giving me many useful comments. Thank a lot and again Bala, for many things he helped me, I am grateful to him.

I would like to say sincerely thank to my friends and my colleges at LAAS, at Torus Actions and from many place in France and in Vietnam for their support, not only in studying but also in many other aspects of life.

Thank my family, my parents and my two sisters, who constantly support me and always believe in me for whatever I choose.

Thank the professors at Institute of Mathematics, Vietnam academy of science and technology for their support and giving me the chance to start my journey in France.

Thank Urtzi for his guidance when I started working on applied mathematics domain and partially helping me do the first paper I have ever done.

Also, I would like to thank Continental Digital service in France and LAAS-CNRS for giving me this great opportunity.

Toulouse, 30 November 2020

0.1 Abstract - French Version

Dans les réseaux de communication, un ordonnanceur décide quelles ressources doit être attribuée à quel utilisateur. Les ressources disponibles étant limitées et les besoins des utilisateurs étant hétérogènes, le choix de l'ordonnanceur joue un rôle important dans la conception du réseau. Avec l'augmentation de la demande en ressources réseaux, due à l'utilisation croissante d'appareils mobiles et notamment à l'émergence des véhicules connectés, ce problème d'ordonnancement devient à la fois plus critique et plus complexe. Les ordonnanceurs utilisés actuellement allouent le canal en considérant son état actuel, et éventuellement ses états passés, mais sans tenir compte de ses états futurs. Ceci conduit à une allocation sous-optimale des ressources, ce qui peux avoir un effet néfaste sur les performances du réseau dans les périodes de congestion. Dans cette thèse, nous proposons un ensemble d'algorithmes d'ordonnancement qui exploitent l'information sur les états futurs du canal pour améliorer l'utilité totale du réseau. Le premier ensemble d'algorithmes est conçu comme une amélioration de l'ordonnanceur à équité proportionnelle dont l'objectif est de maintenir un certain équilibre entre d'une part un débit total élevé et d'autre part une certaine équité entre utilisateurs garantissant à chacun un niveau proportionnelle de service. Le deuxième ensemble d'algorithmes effectuent conjointement contrôle de puissance et allocation du canal, toujours dans le but de maximiser une fonction d'utilité basée sur le concept d'équité proportionnelle. Les expériences numériques réalisées avec des modèles simples de mobilité ainsi qu'avec des traces générées en utilisant l'environnement SUMO montrent que les algorithmes proposés améliorent l'utilité, à la fois lorsque le réseau comporte une seule station de base et lorsqu'il en comporte plusieurs. Un des inconvénients des algorithmes proposés est qu'à chaque instant de décision il est nécessaire de résoudre un problème d'optimisation convexe de grande dimension, ce qui peut être rédhibitoire pour certains scénarios. C'est pourquoi, dans la dernière partie de la thèse, nous explorons une méthode basée sur un réseau de neurones profond pour apprendre les décisions des algorithmes proposés. Cette méthode permet de générer des décisions beaucoup plus rapidement tout en ayant une faible erreur d'approximation.

0.2 Abstract - English Version

In communication networks, a scheduler decides which network resources are allocated to which user. Due to limited available resources and heterogeneous user requirements, the choice of the scheduler plays an important role in network design. The increasing use of mobile devices, and in particular connected vehicles, is expected to drive the demand for network resources even higher making the scheduling problem more critical and complex. The current generation of schedulers base their decisions mainly on the past and the current channel state information but do not take into account the future channel state information. This leads to a sub-optimal allocation of resources which can then have a significant and adverse impact on network performance during periods of saturation. In this thesis, we propose a set of scheduling algorithms based on future channel state information with the objective of improving the total network utility. The first set of algorithms are designed as an improvement to the proportional fair scheduler whose objective is to maintain the balance between getting high total throughput and guarantee everyone getting a proportionally level of service. The second set of algorithms perform joint power control and channel allocation again with the objective of maximizing the proportional fair utility. Numerical experiments conducted with simple mobility models as well as traces generated using the SUMO mobility environment show that the proposed algorithms improve the utility in both single and multi-base stations networks. One of the downside is that, at each decision instant, the proposed algorithms need to solve a high dimensional convex optimization problem that may be computationally prohibitive in some real-time scenarios. In the final part of the thesis, we explore a deep neural network based method to learn the decisions of the proposed algorithms. This method is able to generate decisions much faster while maintaining a low approximation error.

Contents

0.1 Abstract - French Version 3
0.2 Abstract - English Version 4

1 Introduction **9**
1.1 Motivation of the thesis 9
1.2 Problems addressed in the thesis 10
1.3 Thesis contributions 10
1.4 Thesis organization 11

2 Background **13**
2.1 Connected Vehicle Technology 13
2.1.1 eHorizon project by Continental Digital Service in France 14
2.2 Wireless connectivity solutions 15
2.3 Basics concepts 16
2.3.1 Channel Capacity and Signal-to-noise ratio (SNR) . . . 16
2.3.2 Fading effects 16
2.3.3 Data rate, Throughput 17
2.3.4 Maximum Utility and Generally Fair Scheduling Problem 17
2.4 Channel Allocation and Power Control 21
2.4.1 Some proposed problems in scheduling 21
2.4.2 Models and Methods proposed in Scheduling 23
2.5 Technical parts 25
2.5.1 KKT conditions for solution of Convex Optimization Problem 25
2.5.2 Supervised learning with Deep Feedforward Neural Networks 26
2.5.3 Deep Reinforcement Learning by Modeling as a Markov Decision Process (MDP) 28

3 Channel allocation **33**
3.1 Introduction 34
3.1.1 Contributions 36

3.1.2 Organisation . . . 36
3.2 Background . . . 37
3.2.1 Projection on simplex . . . 37
3.2.2 Projection on feasible set D . . . 37
3.3 Problem formulation . . . 38
3.4 Existing Algorithms . . . 40
3.4.1 Greedy allocation . . . 40
3.4.2 Proportional Fair (PF) allocation . . . 40
3.4.3 Predictive Finite-horizon PF Scheduling ((PF)^{2}S) . . . 40
3.5 Projected gradient approach . . . 42
3.5.1 Projected gradient short term objective algorithm (STO1) 45
3.5.2 Projected gradient short term objective algorithm 2 (STO2) . . . 46
3.6 Numerical results . . . 47
3.6.1 One road network . . . 47
3.6.2 Network simulation with SUMO . . . 50
3.7 Summary . . . 52

4 Joint power control and channel allocation 57
4.1 Introduction . . . 58
4.1.1 Contributions . . . 59
4.1.2 Related work . . . 59
4.2 Problem formulation . . . 60
4.2.1 Channel gains . . . 62
4.3 Algorithms . . . 62
4.3.1 Locally optimal algorithm . . . 63
4.3.2 Short-term Objective 1 (STO1) . . . 63
4.3.3 Short-term Objective 2 (STO2) . . . 65
4.4 Numerical experiments . . . 66
4.4.1 Stationary channel . . . 66
4.4.2 Slowly varying channel . . . 67
4.4.3 Mobility Model . . . 68
4.5 Summary . . . 71

5 Learning the channel allocation algorithm 75
5.1 Abstract . . . 75
5.2 Introduction . . . 75
5.2.1 Related works . . . 76
5.2.2 Contributions . . . 77
5.2.3 Organization . . . 77
5.3 Problem Formulation . . . 78

5.3.1 Optimization problem and the STO1 algorithm 78
5.3.2 Learning STO1 with DFNN 81
5.4 System Setup for learning 83
5.4.1 State 84
5.4.2 Target 84
5.5 Discussion about the continuity of the STO1 function 85
5.5.1 Learning with the dual value 91
5.5.2 Example in a small dimension 91
5.5.3 Proposed method to obtain a continuous function . . . 98
5.5.4 Loss, DFNN architecture, initial parameters and optimizer 98
5.6 Numerical Comparisons 99
5.6.1 An Unified Data Generator for Comparison 99
5.6.2 Comparison of different DFNN architectures 99
5.6.3 Comparisons of different loss functions 105
5.6.4 Comparisons of different ordering schemes 105
5.6.5 Comparisons of learning the channel allocation against learning dual values 105
5.7 Computing times 108
5.8 Discussion on a reinforcement learning based approach for learning Optimal Policy 109
5.9 Summary and Discussion 112

6 Discussion and Future Works 115

Chapter 1

Introduction

1.1 Motivation of the thesis

When a connected vehicle moves on the road, it needs internet connectivity for exchanging a high volume of information for services such as security, driving conditions, local information, infotainment, etc. Due to mobility, wireless connectivity is the only communication solution that is possible. Compared to wireline networks, wireless resources are much more scarce and expensive which makes resource allocation decisions more important in wireless networks, especially for vehicular networks.

In cellular networks, proportional fairness is the de-facto standard criterion that guarantees each user a proportional level of service while maintaining a high system throughput. Recently in 3G and 4G networks, opportunistic schedulers have been incorporated in order to maximize proportional fairness [12]. Proportional fair (PF) scheduling algorithms [37] are typical in those opportunistic schedulers. The PF algorithm allocates resources to users based on current and past information and has been proven asymptotically optimal when the wireless channel follows a stationary process and the sojourn time of users are long. However, those assumptions do not necessarily hold for vehicular traffic where the channel state is not stationary due to mobility and the sojourn time is not enough long to get asymptotically property. Therefore, PF algorithms are not necessary optimized for vehicular systems.

With the connected vehicle technology [25, 30, 71], future information can be accessed to improve efficiency of the schedulers, since the more information we have, the better the scheduling decisions that can be made. One of the main difficulty for wireless scheduling is that the exact data rate of a wireless link can be very difficult to predict even on short-time scales due to various

phenomenon (fading, shadowing, e.g.) that are specific to wireless networks. However, it is possible to obtain a partial information on the future channel conditions in the form of estimations. For example, the future positions of a user can be predicted over a short time period if the user accepts to share information about his travel itinerary. Another possibility for estimating the future positions of a user is to compute his most probable path by combining up-to-date information on road traffic conditions with historical data on past travel itineraries of the user [36, 31]. Position of the users may not give the exact data rate, but still it can be useful in the sense that if a user is far from the Base Station (BS), he should get a lower data rate and conversely if he is close to the BS, he should get a higher rate with a high probability. In this thesis, we shall assume that the mean of channel gains, or the mean data rate can be obtained thanks to users' predicted future paths.

1.2 Problems addressed in the thesis

In this thesis, we consider two scheduling problems for vehicular systems. The first problem is channel allocation, and the second one is a joint problem of channel allocation and power control. Both problems share the same objective which is to maximize the proportional fairness between users. We shall assume that the future positions of the users are predicted over a short time period (some next seconds) and from that we assume the means of future data rates (or means of future channel gains) are known for that period. We take that partial information on the future channel condition into account to improve allocation decisions.

1.3 Thesis contributions

The contributions of this thesis can be divided into three main parts which are presented in three chapters: Chapter 3 which is based on [51], Chapter 4 which is based on [52] and Chapter 5.

The first part of the contributions, which is presented in Chapter 3, deals with channel allocation for improving the proportional fairness between users using partial knowledge on future channel condition. In more details, we assume that the future positions of moving users are predictable over the next few seconds. Combining the positions with the Signal-to-Noise-Ratio (SNR) maps, the mean of future data rates can be computed. We present two heuristics using the short-time future information to improve fairness utility. We also propose an idea that mixes two types of time scaling to reduce the

dimension of the original optimization problme. We then compare the performance of the two heuristics against other existing algorithms using SUMO which is an open-source road traffic simulator. The simulation results show that the heuristics outperform other algorithms. However, the computation times of the heuristics are quite heavy comparing to the other algorithms since they require solving a convex optimization problem frequently.

The second contribution, which is discussed in Chapter 4, concerns a joint problem of channel allocation and power control. Again based on a partial view of future channel conditions, we propose two heuristics which are based on those proposed in 3. We then evaluate the heuristics on three types of model: stationary channel gains with fixed mean, slowly-varying channel gains and mobility model where the channel gains vary in every channel allocation slot. For the above three models, the heuristics are shown to outperform the other existing algorithms in which future information is not available. However, the same problem of high computational times for one of the heuristics forces us to omit it out from numerical evaluations for mobility model.

The last part of the contributions is presented in Chapter 5 in which we propose using a machine learning based method, and more specifically a Deep Feedforward Neural Network (DFNN), to learn one of the allocation heuristics (the STO1 heuristic, the one that requires more computational time) in order to have an approximate algorithm which is many times faster. A Deep Feedforward Neural Network (DFNN) can approximate a continuous function with low error, but we provide a counter-example proving that unfortunately STO1 is not continuous. We then characterize the set of all discontinuities, and propose several ordering schemes to reduce the impact of these discontinuities on the learning time. Numerical results show that the proposed ordering schemes enable to converge faster. Although STO1 is not continuous, we show that the dual values of the problem are continuous with the input features. We thus propose learning the dual values instead of the primal ones. Numerical results shows learning with dual values is faster.

1.4 Thesis organization

In Chapter 2, we provide background material on resource allocation in wireless network. That includes connected vehicle technology, basics concepts for scheduling and resource allocation in wireless networks (SNR, Shannon channel capacity, fading effects, data rate, throughput, fairness, utility maximization, etc.). In addition to a short reminder about the mathematical tools used in this thesis (Deep Feed-forward neural networks, KKT condition, etc),

we also present a short review of previous works on channel allocation and power control, including potential methods and explanation for why we use or not use them.

In Chapter 3, we consider channel allocation problem for vehicular networks in a multiple BS setting. In that chapter, we first state the assumptions and define the objective function. We then discuss some existing algorithms that we shall use for comparison. After that we present our heuristic algorithms that use a partial view of future channel information for improved proportional-fair utility. Finally, we employ SUMO for evaluation of the heuristics and the existing algorithms.

In Chapter 4, we consider the joint channel allocation and power control problem for a single BS in three settings: stationary channel, slower time scale varying channel and mobility model where the channel gains vary at a faster time scale. We first state the problem and assumptions, and then present an existing algorithm that does not take future information into account. We then describe our heuristic algorithms that use future information to improve proportional-fair utility. The numerical comparisons for the three settings for both existing and heuristics are shown at the end.

In Chapter 5, we present a supervised learning based method for learning one of our heuristic algorithms. We first recall the resource allocation problem that we consider in Chapter 3 in a single BS case, then we recall STO1 algorithm and state the learning problem that we want to do in this chapter. We define input-output for the DFNN model, then discuss about the continuity of that setting and how to reduce discontinuity. Numerical results are shown after that. At the end we discuss several research directions that can be done in the future including a reinforcement learning based approach, that we did not succeed in but is nevertheless interesting to discuss.

Finally, some conclusions are drawn in Chapter 6, where we also discuss possible extensions of this work and future research directions.

Chapter 2

Background material

2.1 Connected Vehicle Technology

Connected vehicle technology enables vehicles and infrastructures to communicate and share transportation information using wireless technology. Thanks to a high-speed wireless connection, a connected car can carry onboard many applications to improve traffic safety, security and comfort (such as infotainment, parking assistance).

The connected vehicle concept refers to not only the vehicles and the infrastructure but also to applications, services, and technologies that enable a vehicle to interact with its environment.

There are five ways a vehicle can connect to the surroundings and communicate with them:

- connecting to the infrastructure (V2I), which lets the vehicle be aware of the infrastructure surrounding it and also lets the infrastructure obtain information from the vehicle.

- communicating between vehicles (V2V), the purpose of this is to share information about speed, position, etc. between neighbouring vehicles in order to reduce congestion, avoid accidents and increase positive impact on the surroundings.

- exchanging information with the Cloud (V2C), this allows the vehicle to exchange information about and for applications (such as driving assistance, infotainment, and vehicle maintenance).

- communicating with personal mobile devices with the Pedestrian (V2P) to perceive the environment in order to improve safety and mobility.

- interconnecting with Everything (V2X) including other connected vehicles, infrastructure and personal mobile devices.

Such types of connectivity help a connected vehicle and the system be better aware of each other and of the surroundings in order to have more intelligent decision during its mobility.

2.1.1 eHorizon project by Continental Digital Service in France

Electronic Horizon is an embedded software for cloud-based virtual sensor network for exchanging detailed road network information (map data, vehicle's mobility model and the road ahead) between the cloud and the vehicles to help the vehicle make intelligent decisions. It has been a subject of several academic papers [25], [30],[71] and practical tests [1].

With the Electronic Horizon project (eHorizon), Continental wants to make mobility safer, proper and smarter. Continental perceives digital transformation as a tremendous lever to strengthen its contribution to these three objectives. This relates to the transformation of the manufacturing and use of automotive systems, but beyond the whole mobility experience.

Continental Digital Services France[1] (CDSF) is a new subsidiary of Continental Automotive France which was created to address these opportunities around connected vehicles, autonomous vehicles and mobility services. The goal is to merge on-board intelligence with that of the in-house platform "in-the-cloud". For each connected vehicle, a cloud assistant can access information in real time far beyond the horizon of its on-board sensors. On a larger scale, it makes possible global analysis on the history of movement flows of all vehicles, while preserving user confidentiality.

The new services considered in the eHorizon project of the automotive supplier Continental thus integrate ever more numerous and important interactions between vehicles that have become connected and their environment. These new interactions will link various pieces of equipment in the vehicle, and in particular important components for passenger safety such as brakes, steering or obstacle detection radars, with signaling elements (traffic lights, solid lines non-crossing, level crossings, etc.) or other users of the traffic network (other vehicles, cyclist, pedestrians, speed bump, etc.).

Among the new services, some, such as engine monitoring or increased visibility, will be critical. However, they will have to share the available

[1]This thesis is supported by a contract with Continental Digital Services in France. It is in the eHorizon project which is about connected vehicles, autonomous vehicles and mobility services.

bandwidth with non-critical applications such as multimedia and infotainment. By definition, critical services require a higher quality of service than non-critical services for whom the quality can be adjusted according to the circumstances. A vehicle will be driven in very varied environments - urban environment, traffic jams, highways, tunnels, remote areas. Each of these environments has a different effect on the quality of communication between the vehicle and the infrastructure, which can lead to large variations in throughput. Despite these variations, critical services and applications should be as accessible as possible in all of these environments. However, since the communication channels will be shared between the different services and applications, if no particular protection measure is taken, critical services, for which it is important to ensure a minimum quality, will suffer from these variations as much as the non-critical services.

The eHorizon as described above is a complex system that creates numerous potential challenges. In [25] the author presents nine big challenges that need to be solved before eHorizon can be released. In this thesis, we address two of these nine problems: how to distribute information efficiently and how to establish fair access to resources. The first objective of this thesis is to design an efficient resource allocation algorithms (with and without power control) by using predicted future path provided by the eHorizon infrastructure. The information on the future path will be used to derive a partial view of future channel conditions and improve scheduling decisions. The second objective is to make the algorithms simpler by using machine learning based methods that are fast and that can be potentially implemented in real vehicles.

2.2 Wireless connectivity solutions for connected cars

Connected vehicles are expected to interact closely with their environment by exchanging a large volume[2] of information related to security, driving conditions, local information or infotainment. Due to vehicular mobility, wireless connectivity is the only solution for the electronic horizon described above.

Wireless channels have several characteristics that distinguish them from wired channels. First, the bandwidth provided by current wireless technologies is much smaller and much more expensive than wired technologies, and second, the data rate of a wireless link can be unpredictable even on short-

[2] 4TB per day per autonomous car [2].

time scales due to various phenomenon (fading, shadowing, e.g.) that are specific to wireless communication. A consequence of the unpredictability is that the channel allocation decisions can be much inefficient in wireless networks.

One novel feature of connected vehicles is that the itinerary of a vehicle will be known in advance to the scheduler based on a program allowing future path prediction. From that itinerary, we can compute the distance of the vehicle to the BS and obtain a partial view of future channel conditions. With this additional information on the channel conditions, improvements can be expected in the scheduling efficiency. In chapters 3 and 4, we shall present algorithms which profit from the future channel information and improve.

In the following sections, we present an overview of resource allocation in wireless networks as well as some concepts of convex optimization, Markov decision processes and machine learning which will prove to be useful is the next chapters.

2.3 Basics concepts for resource allocation in wireless networks

2.3.1 Channel Capacity and Signal-to-noise ratio (SNR)

Channel capacity is the maximum achievable rate at which information can be transmitted over a communications channel. It is the highest information rate that can be attained with arbitrary small error.

Following the model in [70], the Shannon channel capacity and Signal-to-noise ratio (SNR) (in additive white Gaussian noise case) of a moving user can be computed as:

$$C = B \log_2 (1 + \mathrm{SNR}) = B \log_2 \left(1 + \frac{P \cdot d(t)^{-\gamma}}{N \cdot B} \right),$$

where C is the Shannon capacity of the user, B is the bandwidth of the channel, P is the transmit power of the BS, d is the distance of the user to the BS, N is power of the noise and interference over the bandwidth, and γ is the path loss exponent.

2.3.2 Fading effects

Fading is the variation of the channel strength in wireless communications which can be divided into two types: large-scale fading and small-scale fading.

Large-scale fading is caused by path loss of signal due to the distance to the BS and shadowing by large objects such as building, hills, etc. Large-scale fading is typically independent of the frequency. Small scale is the deviation of the signal strength considered over very small distances due to multi-path effects. When the signal is sent, it can reach the user via many different paths due to reflection, diffraction and scattering. Small scale fading is frequency dependent. For more details about fading and the many models proposed to capture fading we refer to [69].

Fading makes the SNR (and Channel Capacity) become random process. In our work, we will assume SNR is a random process by multiplying with a random number as follow:

$$\gamma(t) = \eta \cdot f(d(t))$$

where $\gamma(t)$ is SNR at time t, d is distance to the BS and η is a random number between $(1-\epsilon, 1+\epsilon)$ where ϵ stands for noise level and is a number in $(0, 1)$. The noise in the SNR needs not be necessarily multiplicative as above, we use it for our numerical results and believe it works for other types of noise as well as the mean of SNR is known to the scheduler.

2.3.3 Data rate, Throughput

Data rate is the potential rate of data (bits per second) that can be transmitted. It is not the actual amount of data that is transmitted every time but the theoretical amount that can be sent or received on a link. Throughput is the practical amount of data (bits per second) that the link can achieve. For example, the throughput is often less than data rate since the user may not be served in some time-slots due to the presence of other users. In our problem, we shall use the term 'maximizing throughput', it actually means 'maximizing total throughput of all users over a long time horizon'.

2.3.4 Maximum Utility and Generally Fair Scheduling Problem

Maximum Utility Scheduling Problem

In this thesis, we shall focus on the network utility maximization problem for connected vehicles. For wired networks, this problem was formulated in [45] and has been applied in various wireless settings [10], [66]. In brief, each user stays in the system for a certain duration during which the scheduler assigns it some bandwidth. This allocation can either be continuous in time or in blocks. The utility of the user is defined as some function of the total

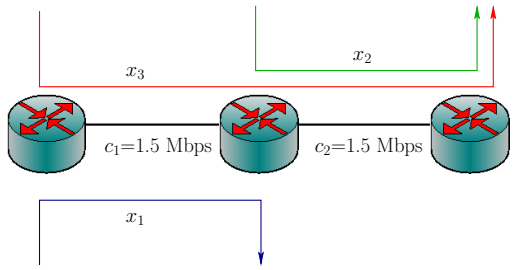


Figure 2.1: Simple scenario illustrating the trade-off between network throughput and user fairness.

bandwidth it receives. The network utility maximization problem is to assign bandwidth or to schedule users in such a way so as to maximize the sum of the utilities of all the users. In [45], a large class of polynomial utility functions parameterized by the degree of the polynomial were introduced. They generalized the log utility function [32], [33].

Formally, a *network utility maximization problem* amounts to finding a vector $x^* = (x_s^*)_s$ which is an optimal solution of the optimization problem

$$(U) \begin{cases} \max \sum_s U_s(x_s) \\ \text{subject to } x \in \mathcal{X}, \end{cases}$$

where x_s is the rate allocated to user s and U_s is the utility function of that user. As the objective function represents the total utility of all users, problem (U) is often referred to as the *maximum utility problem.*

In general, the feasible set $\mathcal{X}$ is defined as the set of all non-negative vectors $x \geq 0$ satisfying a capacity constraint of the form $Ax \leq C$. In this constraint, C is a vector specifying the capacity of each resource j, and A is an incidence matrix such that $a_{j,s} = 1$ if user s uses resource j, and 0 otherwise. To illustrate these notations, let us consider the simple example depicted in Figure 2.1. In this example, there are 2 links of capacities $C_1 = C_2 = 1.5$ Mbps and 3 users (or flows). User 1 uses only link 1, user 2 uses only link 2, while user 3 uses both links. As a consequence, an allocation is a vector $(x_1, x_2, x_3) \geq 0$ satisfying the capacity constraints $x_1 + x_3 \leq 1.5$ and $x_2 + x_3 \leq 1.5$.

Of course, the optimal rate allocation x^* depends on the utility functions U_s. What is an appropriate choice for these functions? As we shall see below, the answer depends on the properties that we expect for the rate allocation. To illustrate this, let us consider different choices of the utility functions.

Allocation maximizing the total network throughput

A somehow natural choice for the utility functions is $U_s(x_s) = x_s$, in which case the utility of a user s is the rate x_s allocated to it, and the objective function of problem (U) represents the total network throughput $\sum_s x_s$. The optimal allocation can then be computed as the solution of the following linear program

$$\begin{cases} \max \sum_s x_s \\ \text{subject to } Ax \leq C, \ x \geq 0. \end{cases}$$

This choice of the utility functions however often raises fairness issues between users, as can be illustrated with the example of Figure 2.1. Indeed, for this example, we look for a vector $(x_1, x_2, x_3) \geq 0$ maximizing $x_1 + x_2 + x_3$ subject to the constraints $x_1 + x_3 \leq 1.5$ and $x_2 + x_3 \leq 1.5$. It turns out that the optimal solution of this simple linear program is $x_1 = x_2 = \frac{3}{2}$ and $x_3 = 0$. In other words, the optimal allocation gives the maximum possible rate to users 1 and 2, and nothing to user 3. This is of course not something acceptable in a real setting, where a certain fairness between users is required.

Max-min fair allocation

The concept of max-min fairness was introduced in [57]. A max-min fair allocation is defined as follows.

Definition 2.3.1 (Max-min fairness)**.** *A feasible allocation x of resource to the users is max-min fair if for each user s, x_s cannot be increased without decreasing the allocation $x_{s'}$ of another user s' such that $x_{s'} \leq x_s$.*

The max-min fair allocation can be easily computed using a simple water-filling algorithm. Initially, the algorithm assigns to all users the same rate r, that is, $x_s = r$ for all s. Starting from the value $r = 0$, the algorithm increases the value of r until the capacity constraint $\sum_s a_{j,s} x_s = r \sum_s a_{j,s} \leq C_j$ is satisfied as an equality for one of the resource j. The water-filling algorithm then set the rates of all users using this resource j to $C_j / \sum_s a_{j,s}$, and starts a new iteration with the other users, until the rates of all users are set.

For the example in Figure 2.1, the water filling algorithm starts from $x_1 = x_2 = x_3 = r$, and then increases r until either $x_1 + x_3 = 2r = 1.5$ or $x_2 + x_3 = 2r = 1.5$. In this simple case[3], the solution is of course $x_1 = x_2 = x_3 = \frac{3}{4}$. This solution is particularly fair to the users since they all get the

[3]If the capacity of the second link was $C_2 = 2$ Mbps instead of $C_2 = 1.5$ Mbps, the water-filling algorithm would need another iteration, and the solution would be $x_1 = x_3 = \frac{3}{4}$ and $x_2 = \frac{5}{4}$.

same rate, but this greater fairness is obtained at the price of a lower network throughput. Indeed, the total network throughput is only $3 \times \frac{3}{4} = 2.25$ with the max-min fair allocation, instead of $2 \times \frac{3}{2} + 0 = 3$ with the allocation maximizing the total network throughput.

Proportional Fair (PF) allocation

Between the two extreme allocations discussed above, there are in fact many other allocations making a trade-off between network throughput and user fairness. A particularly appealing allocation is the Proportional Fair (PF) allocation, which was defined by F. Kelly in [32] as follows.

Definition 2.3.2. *A feasible allocation x of resource to the users is said* proportionally fair *if for any other feasible allocation x' we have:*

$$\sum_s \frac{x'_s - x_s}{x_s} \leq 0.$$

Equivalently, the PF allocation can be defined as the solution of the following optimization problem:

$$(PF) \begin{cases} \max_s \sum_s \log(x_s) \\ \text{subject to } Ax \leq C, x \geq 0 \end{cases}$$

As a consequence, when $U_s(x_s) = \log(x_s)$, the maximum utility problem (U) is called the PF scheduling problem. We note that the logarithm forbids to allocate nothing to a user, and at the same time it makes it non profitable to allocate too much capacity to a single user (concavity).

For the example in Figure 2.1, the PF allocation is a vector $(x_1, x_2, x_3) \geq 0$ maximizing $log(x_1)+log(x_2)+log(x_3)$ subject to the constraints $x_1+x_3 \leq 1.5$ and $x_2 + x_3 \leq 1.5$. The optimal solution of this simple non-linear program is $x_1 = x_2 = 1$ and $x_3 = \frac{1}{2}$, yielding a network throughput equals to $2 \times 1 + \frac{1}{2} = 2.5$. This allocation is less fair than the max-min fair allocation (the user using two resources has half the rate of the other users), but it yields a greater network throughput.

α-Fair allocation

The concept of α-fairness was proposed in [45] as an attempt to unify all previously proposed fairness concepts. Given $\alpha \geq 0$, an α-fair allocation is the solution of the maximum utility problem (U) in which the utility $U_s(x)$ is defined for $x \geq 0$ as follows

$$U_s(x) = \begin{cases} \frac{x^{1-\alpha}}{(1-\alpha)} & \text{if } \alpha \neq 1, \alpha \geq 0, \\ \log(\alpha) & \text{if } \alpha = 1. \end{cases}$$

The formula of $U_s(x)$ is continuous both in x and α. Obviously, the PF allocation is a special case of the α-fair allocation obtained when $\alpha = 1$. When α tends to 0^+, the α-fair allocation tends to maximize the total network throughput. When α tends to 2, some delay can be reduced. Finally, when α goes to $+\infty$, the α-fair allocation tends to the max-min fair allocation, as proven in [45].

For a detailed discussion about network fairness, utility and resource allocation, we refer to [60]. In this thesis, we shall concentrate only on the PF scheduling problem.

2.4 Channel Allocation and Power Control

2.4.1 Some proposed problems in scheduling

The first works on dynamic scheduling for wireless channels appeared in early 90s [67, 68]. By dynamic, we mean that the scheduler uses the information on the channel conditions and the queue-length of the nodes. In [67], the scheduling problem was to determine which wireless nodes to activate based on the queue-length information of the nodes. The wireless channel induced the constraint that certain nodes could not be simultaneously activated due to the interference generated by the transmissions. They proposed an algorithm of type *MaxWeight* that activates the set of nodes that have the highest weight, where the weight of subset of nodes is a linear combination of the queue-length and the data rates of the nodes in the subset. The advantage of this algorithm is that it has maximal stability, that is, as long as the arrival rates to the nodes are within the capacity region of the system, the queues will be stable. In [68] similar results for parallel link and binary channels that could be active or inactive were shown for the policy Longest Queue First (LQF). In general, such optimality results do not carry over to more general topologies as was shown in [46]. However, they show that *MaxWeight* does better than *backpressure* in terms of message delays in networks of large size. Due to difficulty in the exact analysis of such algorithms, some works turned towards asymptotic analysis. In [61], it was shown that a generalized version of *MaxWeight* minimized corresponding polynomial cost on the queue-length in the heavy-traffic regime. A bound, independent of the number of nodes, on the message delay as a function of the load was obtained in [50]. The

above mentioned studies were concerned primarily with stability or with bounds on performance. Besides, there are many other different objectives and algorithms have been proposed [49], [32], [45], [53], [29]. In [49], the author consider the problem that makes a trade-off between minimal energy and queue length of network delay. In [53] the maximizing total probability of departure of users is considered. In [29], the authors consider a joint optimization of channel allocation and transmit power control problem for multiplexing schemes OFDM.

Another line of research has investigated *fairness* which can be understood as follows: when the resource is not sufficient for all demand, it should be allocated *fairly* to all users and this discipline should be the fundamental principle in resource allocation [59]. However, there are many ways to define *fairness* for resource allocation [57], [32], [45]. The definition of proportional fair scheduling was first proposed in [32] and then was generalized to α-fair in [45] as mentioned above. The concept of max-min fairness, which was also discussed in Section 2.3.4, has been proposed in [57].

Regarding the time horizon, there are two categories that are *short-term* objective [39], [14] and *long-term* objective [29], [62]. Short-term fairness tries to optimize over a short time period, while long-term tries to optimize over longer horizon or until the end of the users' sojourn. Short-term objective can be good for QoS guarantee since it concentrates on solving the current QoS and also good for real time processing since it solves a smaller dimension problem comparing to long-term objective. But when the resource is insufficient for the demand, short term objective is difficult to be satisfied. Another disadvantage of short term fairness is total throughput (over whole horizon) may not be high. Beside resource allocation, the transmission power levels can also be assigned differently for users to increase the network efficiency [29, 52, 39]. In [29], a joint optimization of channel allocation and transmit power control in long term horizon problem for multiplexing schemes OFDM is considered, in which the power is the same for every time-slot but can be divided for many users and many sub-channels. In [52], the power can be varied and so can be different between time-slots but have to satisfy an average constraint. For detailed discussions about fair scheduling, we refer to these surveys [60], [48]. In this thesis, we shall restrict ourselves to *proportional fair scheduling* for a long-term horizon, with two problems: the first one is for only resource allocation and the second one is for joint power control and resource allocation.

Proportionally fair scheduling has been studied extensively in the past both for stationary [33], [37] and non stationary channel system [43], [3]. Many results have been proposed for both types of system. In [37] the authors presented an algorithm named PF and then proved asymptotic optimality of

the algorithm in stationary environment. Many authors have studied for PF [37, 26, 41, 4, 22]. In [37] (2004) H. Kushner and P. Whiting give an analysis for PF algorithm behavior. It is proved that the PF algorithm is asymptotically optimal in stationary environment. Before [26]. J. Holtzman proved a similar result but restricting for two classes of users with different fading characteristics. In [41] (2008) the authors give a raw analysis (there are some reasonable guesses without proof) for PF scheduling gain comparing to RR (round robin scheduler) in stationary environment. They assume that the instantaneous data rate follows Gaussian distribution with the mean and deviation depending on SINR (signal to interference-plus-noise ratio). In [4] (2012) the authors also considered PF scheduler in stationary environment but more general situation with in OFDMA systems for bursty traffic conditions. They used the method that presented in [41] to infer the analytical closed-form expressions for the average throughput, Jain's fairness index, and packet delay. In [22] (2018) the authors compute analytically a lower bound for competitive ratio of two primal-dual algorithms (that covers PF scheduling) for a class of online convex optimization problems. The classes of α-fair objective functions satisfy their sufficient condition and they show that the lower bound for worst case competitive ratio of PF to the optimal is equal to 1/2. However, it is also shown in many papers that PF may not be optimal work for mobile systems [43], [51]. In [43], a real-time implementation of the *proportional fair* based on predicted future data rates of the users was proposed. In [3], the rate is assumed to be accurately predicted in every time-slots, then an optimization problem with shorter horizon over that the rate can be predicted is solved.

2.4.2 Models and Methods proposed in Scheduling

Many models and methods has been proposed for scheduling in the past. In this section, we list the methods in our knowledge and explain why we use or not use them.

Whittle index based method for multi-armed bandits has been proposed for scheduling in Wireless Network [5], [53]. In [53], the authors model the scheduling problem as a restless multi-armed bandits where each user inside the system can be considered as a bandit, then employ Whittle index method to have an heuristic index based algorithm. The main assumptions are straight road (which is simple topology map) and same velocity for every user. These assumptions are strict, but they help to simplify the model, to show indexability and to formulate Whittle index. Although the objective considered in the paper is not proportional fair, the method can applied for the proportional-fair scheduling problem. However in general, the Whittle

index formula is not easy to compute [5], [9]. The complexity of finding Whittle index makes it become hardly possible for complex system with heterogeneous users in much more complex map rather than straight road.

Gradient based method has been proposed in several papers, both not using future information [33] and using future information [43]. In [43] the authors consider the PF scheduling, and propose an algorithm using a method looks like gradient based, it is an index policy with index of an user equals to the ratio of its current rate over the sum of past rate and current rate and estimated future rate. The estimated future rate is computed based on estimated future allocations. The authors propose three way to estimate it: (i) *round-robin*, (ii) *blind search* and (iii) *local search*. We shall discuss below only *round-robin* because *blind search* is in fact the same as the PF scheduler which chooses the one that maximizes the ratio of its current rate over the sum of past rate and current rate. Finally, in each time slot, *local search* iteratively computes until T and then allocates according to the index policy making it a computationally expensive method. Also in the paper, the authors indicate that for local search to be effective, the prediction error has to be low for the whole horizon. If the prediction error is high then local search can be worse than round robin. The method based on round robin estimation is simpler and is showed to work well compared to the current PF scheduler. The formula, is in fact, one step gradient descent with starting point equal to round robin. It could be better if we do more iteration rather than stop at one step to come closer to the optimal. Also, gradient based can be applied for simple constraints such as choosing one user at one time slot, but difficult for more constraints such as average power control as the one we consider in [52].

In [3], the authors consider a long-term fairness which is the same objective function with us. But they assume the rate is accurately predicted in every time-slots. An optimization problem with shorter horizon over that the rate can be predicted is solved. There are two restrictions for using this method: firstly, exactly prediction in every small time slot is hard especially for fast fading; secondly, solving the optimization with original size (one slot equals to 2 ms in 3G networks, 500 slots in every seconds, thus of the order of thousands in dimension for each user if considered in some next seconds) makes a large dimension problem and requires exhaustive calculation, especially if we have to repeat it in every small time-slot.

Deep Reinforcement Learning is also proposed for scheduling for mobile users [78], [6]. In [6], the authors consider a RSU scheduling problem in V2I communication with the objective having an efficient scheduler that preserves the battery power and extends the network's lifetime while promoting the safety of the driving environment and satisfying acceptable QoS levels. Then,

deep reinforcement learning is applied to produce a well-performing policy. In our first problem with resource allocation (without power control), we also model our system as a Markov decision process and apply reinforcement learning to propose heuristic algorithms. The heuristic works well when the dimension is small but not so well when the dimension is large. Two main disadvantages of our problem if using reinforcement learning is that, firstly, we do not have a terminal state; secondly, the state space is extremely large. Detailed discussion about this is placed in the chapter 5.

Scheduling problems are often convex optimization problems and therefore decomposition methods [54] can be applied. There, the constraints can be added into the objective function with a multiplier to reduce the number of constraints. The dual multiplier can be considered as a penalty. We have tried but observed that it works for stationary channel but not for non-stationary channel. It is worth investigating how to design a good penalty (multiplier) for mobile system.

2.5 Technical parts

2.5.1 KKT conditions for solution of Convex Optimization Problem

In this section, we consider a property of a solution of a convex optimization problem, that is Karush-Kuhn-Tucker conditions recalled in [21]. We will use it to discuss about the continuity of the STO1 function which is one of our heuristic algorithms. Consider this following convex optimization:

$$\begin{cases} \max O(\alpha) \\ \text{subject to } g_m(\alpha) \geq 0,\ m = 1, \dots, M \\ \text{and } h_n(\alpha) = 0,\ n = 1, \dots, N \end{cases} \tag{CO}$$

where O is the concave function and the feasible set defined by $\{g_m\}_m$ are differential concave functions and $\{h_n\}_n$ are affine functions.
For our problem later, the objective function is $O(\alpha) = \sum_{i=1}^{K} \log\big(\sum_{j=1}^{T} \alpha_{ij} r_{ij}\big)$, the inequality constraints are $\alpha_{ij} \in [0, 1]$ and the equality constraints are $\sum_{i=1}^{K} \alpha_{ij} = 1$, $j = 1, \dots, T$, which satisfy the strong duality via Slater's condition, since there exist α_0 in the feasible set such that $g_m(\alpha_0) > 0$ for any m and $h_n(\alpha_0) = 0$ for any n. Now we can characterize the solution of the above optimization problem, through Karush-Kuhn-Tucker (KKT) conditions in the following theorem:

Theorem 2.5.1. *Suppose that $\alpha^* \in \mathbb{R}^T, \lambda^* \in \mathbb{R}^M$ and $\mu^* \in \mathbb{R}^N$ satisfy the following conditions:*

1. *(Primal feasibility)* $g_m(\alpha^*) \geq 0, m = 1, ..., M$ *and* $h_n(\alpha^*) = 0, n = 1, ..., N$,

2. *(Dual feasibility)* $\lambda_m^* \geq 0, m = 1, ..., M$,

3. *(Complementary slackness)* $\lambda_m^* g_m(\alpha^*) = 0, m = 1, ..., M$, *and*

4. *(Lagrangian stationary)* $\nabla_\alpha \big(O(\alpha^*) + \sum_{m=1}^M \lambda_m^* g_m(\alpha^*) + \sum_{n=1}^N \mu^* h_n(\alpha^*)\big) = 0$.

Then α^* *is an optimal solution of* (CO). *Furthermore, if strong duality holds, then any optimal solution of* (CO) α^* *must satisfy the conditions 1 - 4 with some constants* $\lambda^* \in \mathbb{R}_+^M$ *and* $\mu^* \in \mathbb{R}^N$.

2.5.2 Supervised learning with Deep Feedforward Neural Networks

In this section, we briefly present the background of supervised learning with DFNN which we will use in Chapter 5, for more detail about this method we refer to [23], [75].

According to Wikipedia [75], supervised learning is a learning task that is to learn the relation of output as a function of input based on a bunch of examples of pairs input-and-output which is called training data. The relation can be derived from those examples by analyzing the training data and an inferred function will be then produced. A learning algorithm is considered as good if it can generalize the relation of input and output, i.e, it is able to determine outputs of unseen inputs with small error. So the objective here is to find the function representing as well as possible the relation of input-output.

Mathematically, denote by $\mathcal{X}, \mathcal{Y}$ the space of input and output respectively. Let f^* be the true function that represents the relation between input and output. However, if the true function is unknown, our task is to find an approximate function f that is inferred from examples taken from training data $(x_n, y_n)_{n=1}^N$. Our objective is thus to find f such that it minimizes empirical risk:

$$R_{\text{erm}}(f) = \frac{1}{N} \sum_{n=1}^{N} l(y_n, f(x_n)),$$

where l is a loss function which measures how far y_n is from $f(x_n)$, l is not necessarily a distance but it looks like a distance in the meaning that it is small if the difference between y_n and $f(x_n)$ is small. *Deep Feedforward*

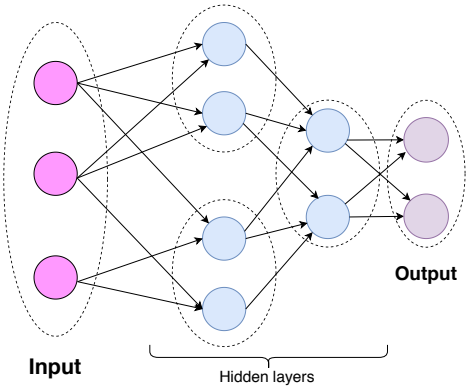


Figure 2.2: Graph of a DFNN.

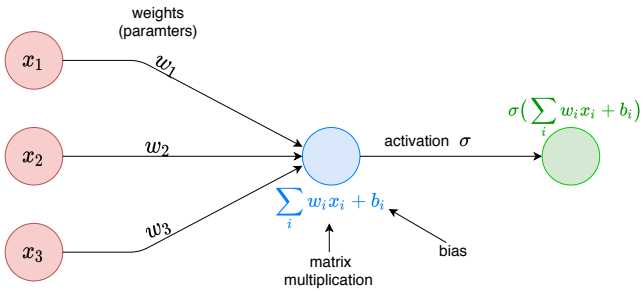


Figure 2.3: Computational Graph of a neural (unit) in a single layer perceptron.

Neural Networks (DFNN) or *multi-layer perceptrons* are the typical models[4] in deep learning. DFNN refers to a class of function which is of this form $f(x) = f^{(n)}(f^{(n-1)}(\cdots f^{(2)}(f^{(1)}(x)))\cdots)$ where each $f^{(k)}, k = \overline{(1,n)}$ is a *singer feedforward neural networks* or *single-layer perceptron*, which is a combination of a linear and non-linear function (often called activation as well). Each $f^{(k)}$ will be called *kth-layer*, and n will be called number layers or the *depth* of the model. The layer of $f^{(k)}$ for $k = \overline{(1, n-1)}$ is called *hidden layer* since it is not shown in the output. The last layer $f^{(n)}$ is called the *output layer*. The graph of a DFNN is shown in figure 2.2. Each single layer contains many *neurals* (units), the computational graph of one such neural is shown in figure 2.3.

Finding a good DFNN function means finding a good architecture (how many layers, which linear and non linear function in each layer, and the

[4]we shall use model and function to mean the approximate function interchangeably.

width of each layer) and then their weights (parameters). Finding a good architecture is not an easy job [64]. In our result we shall compare some architectures by doing experiments. After fixing the architecture, we have to find good parameters, by minimizing $R_{\text{erm}}(f)$ defined above.

2.5.3 Deep Reinforcement Learning by Modeling as a Markov Decision Process (MDP)

We shall also model the problem as an MDP and use Deep Q Networks (DQN) approach. So the following part is dedicated to defining a MDP (see [55], [8] for more details). Even though we were not successful with this model to produce a good heuristic algorithm, it is nevertheless interesting to discuss about it. The main discussion will be placed in section 5.8.

Markov Decision Problem Formulation

We are concerned with a control model which can be defined by the five-tuple $(X, \mathcal{A}, A(x)_{x\in X}, P^a_{x,y}, r(x,a))$ as follows:

1. X is a state space, which is the set of all states of the system under observation;

2. A Borel space $\mathcal{A}$, called the action space;

3. A family $(A(x), x \in X)$ of non-empty measurable subsets $A(x)$ of $\mathcal{A}$, where $A(x)$ denotes the set of actions or decisions available to the controller when the state of the system is $x \in X$.
   Let $K := \{(x;a)|x \in X, a \in A(x)\}$ be the set of all feasible state-action pairs.

4. A measurable real-valued function $r(x,a)$ on K, called the one step reward function, which is assumed to be measurable in $a \in A(x)$ for each fixed $x \in X$.

5. $P^a_{x,y}$ is the transition probability from state x to state y if action a is chosen for state x at the beginning of period;

At time $n = 1, 2, 3, ...$, the history $h_n = x_1, a_1, ..., x_{n-1}, a_{n-1}, x_n$ is observed and the action a_n is selected.
A policy π is a sequence $\pi_1, \pi_2, ...$ of regular transition probabilities π_n from $H_n = (X \times A)^{n-1} \times X$ to A such that

$$\pi_n(A(x_n)|x_1, a_1, ..., x_{n-1}, a_{n-1}, x_n) = 1.$$

For any initial state x and any policy π define a probability measure P_x^π on $((X \times A)^\infty, \mathcal{B}((X \times A))^\infty)$ and the expectation with respect to P_x^π is denoted by $\mathbb{E}_x^\pi$.

Definition 2.5.2. *A policy φ is called Markov, if decisions are non-randomized and depend only on the current state and time, $a_n = \varphi_n(x_n)$. A policy φ is called stationary if decisions are non-randomized and depend only on the current state (not depend on time), $a_n = \varphi(x_n)$.*

In our work, we consider only stationary policies. Below we present the discount-factor reward and value iteration which is a method to find the optimal policy by iterating the value function. In fact, in our formulation we consider an average reward instead of discounted total reward. But discounted reward makes the formula easier to update, and we shall see that when the discount factor is large enough, the optimal policy does not change compared with that of the average reward although it takes long time to converges to the optimal.

Definition 2.5.3. *(Reward criterion) With initial state x and policy π we define the finite-horizon expected total reward as the following:*

$$V_{\gamma,n}^\pi(x) = \sum_{t=1}^{n} \gamma^t \mathbb{E}_x^\pi(r(x_t, a_t))$$

where x_t, a_t are state and decision at time t, and $\gamma \geq 0$ is a discount factor. Infinite-horizon expected total reward is defined by:

$$V_\gamma^\pi(x) = \sum_{t=1}^{\infty} \gamma^t \mathbb{E}_x^\pi(r(x_t, a_t))$$

where $0 \leq \gamma < 1$.
Average expected reward per time is defined by:

$$V^\pi(x) = \limsup_{n\to\infty} \frac{1}{n} \sum_{t=1}^{n} \mathbb{E}_x^\pi(r(x_t, a_t)).$$

For any objective function $g^\pi(x)$, define $g(x) = \sup_{\pi\in\Pi} g^\pi(x)$ be the *value function.* A policy π is called *optimal*, if $g^\pi(x) = g(x)$ for any $x \in X$. Value function for infinite-horizon expected total rewards $V_\alpha(x)$ satisfies the Bellman equation:

$$V_\gamma(x) = \max_{a\in A(x)} \{r(x, a) + \gamma \sum_{y\in X} P_{x,y}^a V_\gamma(y)\}, \text{ for any } x \in X.$$

And vice-versa, if the policy π is such that $V_\gamma^\pi(x)$ satisfies the above equation then π is the optimal policy.

We define the action-state function of policy π- Q_π as follows: Given a policy π we define:

$$\begin{aligned} Q_\pi(x,a) &= \mathbb{E}\big(r(x,a) + \gamma r_1 + \gamma r_2 + \cdots\big) \\ &= r(x,a) + \gamma \sum_{x'} P_a(x,x') Q_\pi(x', \pi(x')) \\ &= r(x,a) + \gamma \sum_{x'} P_a(x,x') V_\pi(x') \end{aligned}$$

And define the Q-Action and state function (depends on the state and action) as follows:

$$\begin{aligned} Q(x,a) &= \max_\pi Q_\pi(x,a) \\ &= r(x,a) + \gamma \max_\pi \sum_{x'} P_a(x,x') Q(x', \pi(x')) \\ &= r(x,a) + \gamma \max_\pi \sum_{x'} P_a(x,x') V_\pi(x') \\ &= r(x,a) + \gamma \sum_{x'} P_a(x,x') V(x') \end{aligned}$$

Intuitively, Q function give us: ***how good the action*** a ***is in state*** x**.** Because $V(x) = \max_{a'} Q(x,a')$ so we have Bellman's equation for Q:

$$Q(x,a) = r(x,a) + \gamma \sum_{x'} P_a(x,x') \max_{a'} Q(x,a')$$

The iterations of value function and Q function cost too much memory when the state space is large. Therefore, we will not apply the traditional methods to update Q. Instead, we learn an approximate function $Q(x,a,\theta)$ of Q by find the good parameters for θ. The algorithm to find this approximate function in our work is Deep Q-learning which is presented later.

Some algorithms that we will apply

The update rule that we tried for Q-learning is based on temporal difference (TD) learning, which is used for both V-learning and Q-learning.

Update rule for value learning:

$$V(s) \longleftarrow V(s) + \alpha[r + \gamma V(s') - V(s)]$$

Update rule for action value learning:

$$Q(a,s) \longleftarrow Q(a,s) + \alpha[r + \gamma Q(a',s') - Q(a,s)]$$

where α is learning rate, γ is discount factor.
The reason we use that rule is that:

- From Monte Carlo methods which try to estimate the average returns

$$V_{n+1} := \frac{\sum_{k=1}^{n} w_k G_k}{C_n} = V_n + \frac{w_n}{C_n}(G_n - V_n) \tag{2.1}$$

  where $C_n = \sum_{k=1}^{n} w_k$, (G_i) is a sequence of returns starting at the same state (i.e many trials)

- V_n is average return after n trials.

- Replace G_n by $r+\gamma V(s')$ i.e current reward and next state reward with discount factor.

Now we explain the algorithms in detail.

1. **Traditional Q-learning, [65] (which is Off-policy learning)**

ALGORITHM 1: Traditional Q-learning

Result: action value function Q.
Initialize Q.;
for *each episode* **do**
  initialize state x;
  for *each step, x is not terminal state* **do**
    play and observe r, x';
      $Q(a,x) \longleftarrow Q(a,x) + \alpha[r + \gamma \max_{a'} Q(a',x') - Q(a,x)]$;
  end
end

Difficulty of this algorithm: Because of large state space, there is not enough memory to store the table of $Q(x,a)$ for all (x,a). Therefore we will try another way to find an approximate function for Q in the below algorithm.

2. **Deep Q-Network (DQN)**

ALGORITHM 2: Deep Q-Network (DQN)

Result: Approximate function for Q (denote by $Q(x, a, \theta)$)
Initialize memory D;
Initialize action-value function;
for *each episode* **do**
  Initialize $s_1 = \{x_1\}$ and $\phi_1 = \phi(s_1)$;
  for $t = 1, T$ **do**
    choose action a-action follows $\epsilon-$greedy;
      $a_t = \max_a Q(\phi(s_t), a; \theta)$;
      play a_t and observe r_t, x_{t+1};
      $s_{t+1} = s_t, a_t, x_{t+1}$ then $\phi_{t+1} = \phi(s_{t+1})$;
      Store $(\phi_t, a_t, r_t, \phi_{t+1})$ in D;
      Sample random minibatch from $D, (\phi_j, a_j, r_j, \phi_{j+1})$;
      $y_j = \begin{cases} r_j & \text{for terminal} \\ r_j + \gamma \max_{a'} Q(\phi_{j+1}, a'; \theta^-) & \text{otherwise} \end{cases}$;
      Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$;
  end
end

Chapter 3

CHANNEL ALLOCATION

In this chapter, we shall assume the transmit power of the BS is fixed and shall consider the channel allocation problem for improved proportional-fairness for vehicular users. As the channel conditions experienced by vehicular users in cellular networks vary as they move due to distance and fading effects as described in 2.3.1, we investigate how proportional fairness could be improved by knowing a partial information of future data rates in mobile system which is a non-stationary environment. Based on electronic horizon (described in 2.1.1), we shall assume the future positions of vehicular users are known in some next seconds, since either the users agree to share their itineraries or it can be predicted based on their moving history. Combining that information with Signal-to-Noise Ratio (SNR) maps, we assume the mean future rates are known. Using the information, we propose two algorithms which predict future allocation over a short-term horizon at regular time intervals, and then uses this extra-knowledge for improved online channel allocation. The prediction of future allocation is obtained by solving a relaxed version of the shorter horizon problem based on a projected gradient method. Using event-driven simulations, we compare the performance of the proposed algorithm against those of other channel allocation algorithms, including the Proportional Fair (PF) scheduler, which is known to be optimal in stationary environments, and the $(PF)^2S$ scheduler, which was devised for mobiles nodes in non-stationary environments. The simulated scenarios include scenarios with multiple base stations and are based on realistic mobility traces generated using the road traffic simulator SUMO. Simulation results show that the proposed algorithms outperform the other algorithms and that exploiting the knowledge of future radio conditions allows a significantly better channel allocation.

3.1 Introduction

A central and challenging problem in cellular networks is channel allocation, that is, to decide which mobile user the base station (BS) should serve in each time slot. To this end, the BS gathers the channel state information (CSI) from users in order to know their radio conditions, which are mainly determined by their distances to the BS and by fading effects. As maximizing the overall throughput would lead to the starvation of distant users (those with the worst potential data rates), today cellular networks allocates the channel to the user with the highest potential rate proportionally to its time-average throughput[1]. With this strategy, users with comparatively low allocated throughput are assigned a higher priority even when they are in worse channel conditions. This scheduling algorithm, which is known as the Proportional Fair (PF) scheduler, provides a fair and efficient sharing of bandwidth between users in the sense that it maximizes the aggregate logarithmic utility of obtained throughput in a fixed population of permanent users [32].

A number of studies have been devoted to the analysis of the performance of PF scheduling in wireless networks [10, 66, 11, 79, 80, 13], assuming either a static population of permanent users, or a dynamic setting in which random finite-size data transfers come and go over time. In both cases, it was shown that PF scheduling strikes a good balance between the overall network throughput and the degree of fairness among users. However, most of the literature is based on the assumption that users experience stationary channel conditions. This was partly motivated by the fact that a simple index-based allocation algorithm had been shown to be optimal for stationary channels [37]. Thus, even if they take into account the fast channel variations due to multi-path propagation, most studies ignore the variations of the channel conditions on slower time scale dues to user mobility. Taking into account such slow fading effects is particularly important for vehicular users as the mean of the Signal-to-noise ratio (SNR) improves as a vehicle comes closer to a BS and then worsens as it moves away. Another usual assumption which is not realistic for vehicular users is the assumption of long sojourn times. Indeed, a vehicle typically stays in the coverage range of a BS for only a few minutes.

In this article, we investigate to which extent the quality of channel allocation could be improved by exploiting information on future radio conditions in non-stationary environments. Our main motivation comes from connected

[1]The throughput is different from the data rate. While the latter is potential rate at which a user can be served, the former can be smaller since in some slots a user may not be served due to the presence of other users.

vehicles which will use cellular networks to exchange informations related to security and driving conditions with their environment. If the trajectory of a car is known or can be estimated from historical travel data and/or observations of the surrounding environment, then one can obtain good statistical predictions of the SNR that will be experienced by the car in the near future. In turn, these predictions could be used by the BS to achieve a channel allocation with a higher utility than that of the PF algorithm. In this chapter, we propose a channel allocation policy exploiting this extra knowledge and evaluate the improvement in utility that it yields in non-stationary environments. Note that such an improvement in utility is not possible under the assumption of a stationary channel as knowing the car trajectory does not bring any new information on the future data rates.

The idea of using information on future radio conditions for channel allocation was already explored in [7]. It uses future information by looking at channel state of users in a few small time-slots. Different from their approach, we do not look at the predicted channel state in few time slots which may be different between users and difficult to predict correctly due to fast fading. Instead, we base our allocation on average rate the user will experience during the time interval this user stays inside the coverage range of the BS.

Another closely related work is [43] in which, using SNR maps obtained by measurements, the authors first show that PF scheduling may perform poorly in the presence of slow fading. They then propose a scheduling algorithm which is similar to PF in that the channel is allocated to the user with the highest potential rate proportionally to its total throughput. This new algorithm, which is called $(PF)^2S$ differs however from PF in that the total throughput includes an estimation of the future throughput whereas PF considers only the already allocated throughput. In order to estimate the future throughput, the authors proposed three methods: round-robin, blind estimation, and a local search heuristic. it was shown that even with this rough estimation of the future throughput, this new index leads to an improved utility compared to the PF algorithm in non-stationary environments. The channel allocation policy proposed in this chapter is similar to the $(PF)^2S$ scheduling policy except that we use a different method for estimating future throughputs of vehicles. For the purposes of numerical comparisons, we shall assume in this chapter that $(PS)^2S$ uses the round-robin policy. It was stated in [43] that, out of the three estimation methods, round-robin is the most robust to prediction errors.

3.1.1 Contributions

We present a heuristic algorithm for non stationary channels that improves the total utility of users compared to the PF and the $(PF)^2S$ algorithms. Our heuristic is similar to the $(PF)^2S$ algorithm, except that instead of computing an estimation of future throughput from a round-robin allocation, we compute it as the solution of a utility maximization problem over a short-term horizon assuming that the means of the future data rates are known over this short horizon.

The original utility maximization problem being computationally complex, we employ three techniques to obtain a lower complexity heuristic: *(i)* we relax the integer constraints of the original problem; *(ii)*, we shorten the time horizon over which the problem is solved; and *(iii)* we compute the solution over macroscopic time slots instead of microscopic ones that helps the algorithm run in real time. The relaxation turns the problem into a convex one and allows for its efficient resolution. Shortening of the time horizon and solving over macroscopic slots reduces the number of variables in the problem and decreases the computation time.

We compare the performance of the proposed algorithm against those of other channel allocation algorithms using event driven simulations. The simulated scenarios include scenarios with multiple base stations and are based on realistic mobility traces generated using the open-source road traffic simulator SUMO with vehicles moving at either equal or different speeds. Simulation results show that the proposed algorithm outperform other algorithms and that exploiting the knowledge of future radio conditions allows a significantly better channel allocation.

A preliminary version of the chapter limited to scheduling in a single base station setting and not including experiments with SUMO appeared in ASMTA2019 [51].

3.1.2 Organisation

In Section 3.3, we state the assumptions and define the objective function. In Section 3.4, we give some background on PF and $(PF)^2S$ algorithms. In Section 3.5, we present our heuristic for improving the utility based on estimations of future average data rate. Section 3.6 contains the numerical results for scenarios with homogeneous as well as heterogeneous vehicles. Finally, we end the chapter in Section 3.7 with a few open problems.

3.2 Background

3.2.1 Projection on simplex

Let $S \subset \mathbb{R}^K$ be a a-simplex, i.e,

$$S := \{\mathbf{x} = (x_1, x_2, ..., x_K) \big| \sum_{i=1}^{K} x_i = a, x_i \geq 0 \quad \forall i = 1, 2, ..., K\}.$$

It is obvious that S is a convex set, therefore the projection on S is unique. In [17], the authors present several ways to compute the projection, we shall use one of them as the following. Let $\mathbf{y} = (y_1, y_2, ..., y_n)$ be a point in $\mathbb{R}^K$, then its projection on simplex $\mathbf{x}$ can be computed following the below algorithm.

ALGORITHM 3: The Projection on Simplex Algorithm

1. Sort $\mathbf{y}$ into $\mathbf{u}$: $u_1 \geq 1 \geq u_2 \cdots \geq u_K$.
2. Set $H := \max\{h | 1 \leq h \leq K, \big(\sum_{r=1}^{h} u_r - a\big)/h < u_h\}$.
3. Set $\tau := \big(\sum_{h=1}^{H} u_h - a\big)/H$.
4. For $i = 1, 2, ..., K$, $x_i = \max\{y_i - \tau, 0\}$.

3.2.2 Projection on feasible set D

Denote by $D = \Big\{\alpha \in [0,1]^{K \times T} \ : \ \sum_{i=1}^{K} \alpha_{ij} = 1, \ j = 1, 2, ..., T\Big\}$ the feasible set of the relaxed problem that we shall present in (I) in section 3.5. The set D is not a simplex, therefore we cannot apply directly the algorithm in 3.2.1. However, for every j the feasible set of allocations is indeed a simplex. We can therefore obtain a projection on D by projecting independently on simplexes corresponding to each of the time-steps. Indeed, the set D is a Cartesian product of J simplexes: $D = D_1 \times D_2 \cdots \times D_J$ where

$$D_j = \{a_j = (\alpha_{ij})_{i=\overline{1,K}} \in [0,1]^K, \sum_{i=1}^{K} \alpha_{ij} = 1\}$$

for all $j = 1, 2, ..., J$.
Since $(D_j)_j$ are simplexes, we can compute the projection on D_j following 3.2.1. The projection on D can thus be computed by the simple following lemma:

Lemma 3.2.1. *If* $Y = (y_{ij})_{i=\overline{1,K},j=\overline{1,J}} \in \mathbb{R}^{K\times J}$, *then*

$$\Pi_D(Y) = \Pi_{D_1}(Y_1) \times \Pi_{D_2}(Y_2) \times \cdots \times \Pi_{D_J}(Y_J),$$

where $Y_j = (y_{ij})_{i=\overline{1,K}}$.

Proof. (of the lemma 3.2.1) Denote by $Z = \Pi_{D_1}(Y_1)\times\Pi_{D_2}(Y_2)\times\cdots\times\Pi_{D_J}(Y_J)$. It is easy to check that for any $X \in D^{K\times J}$ then $\langle Y - Z, X - Z\rangle \leq 0$. □

As described in [17], the complexity of finding Π_{D_j} is equal to $K\log(K)$ by observation in practice, and equal to $O(K^2)$ in the worst case. Therefore the complexity of finding projection on $D = D_1 \times D_2 \cdots \times D_J$ is equal to $JK\log(K)$ in practice.

3.3 Problem formulation

We consider a geographical region with a network of roads that is served by a set of M base stations $\{B_1, B_2, ..., B_M\}$. The region is partitioned into M non-overlapping sub-regions each of which represents the coverage area of a base station. Users (vehicles, bicycles, pedestrians, etc) enter the network, move along different routes, and leave the network. Figure 3.1 shows an area within the city of Toulouse which will be later used in the numerical experiments. In the figure, the width of the box is approximately 1 km, and the height is around 0.65 km. The data for BS location can be found on the website[2] of the French Frequency Agency (ANFR), which manages all radio frequencies in France.

Every $\delta = 2$ ms each BS has to decide which user to serve in a decentralized fashion. We shall assume that the data rate received by a user depends on the distance between the BS and that user. The data rate depends upon the SNR which itself can vary along the road. In our numerical experiments, we assume that the data rate decays exponentially as in formula (3.1) below

$$r^m(x) = \begin{cases} 0 & \text{if } d(x, B_m) > d_m, \\ 1 + \kappa\, e^{-d(x,B_m)/\sigma} & \text{otherwise,} \end{cases} \tag{3.1}$$

where x is the position of the user, B_m is the position of BS m, $d(x, B_m)$ is the Euclidean distance between positions B_m and x, and κ and σ are adjustable parameters. The scheduling algorithm we propose does not however require this assumption to work.

Denote by T the time horizon over which the scheduling decisions are made, and let K be total number of users who pass through the considered

[2] `https://data.anfr.fr/anfr/portail`

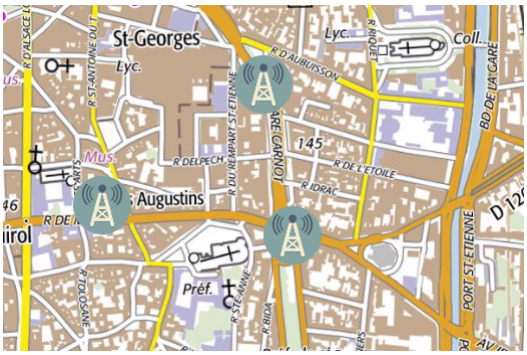


Figure 3.1: A selected area of Toulouse which is covered by three BSs (LTE1800) of the French mobile network operator Free Mobile.

region during that time. For simplicity, we assume that T is a multiple of δ. Our objective is to achieve the proportional-fairness between users, which is described by the following optimization problem (see, e.g., [10, 43, 3]):

$$\begin{cases} \text{maximize} & O(\alpha) = \sum_{i=1}^{K} \log\left(\sum_{m=1}^{M}\sum_{j=1}^{T} \alpha_{ij}^{m} r_{ij}^{m}\right) \\ \text{subject to} & \\ & \sum_{i=1}^{K} \alpha_{ij}^{m} = 1, \quad j = 1,\dots,T,\ m = 1,\dots,M, \\ & \sum_{m=1}^{M} \alpha_{ij}^{m} \leq 1, \quad j = 1,\dots,T,\ i = 1,\dots,K, \\ & \alpha_{ij}^{m} \in \{0,1\}, \quad j = 1,\dots,T,,\ i = 1,\dots,K,\ m = 1,\dots,M. \end{cases} \tag{I}$$

where:

- α_{ij}^{m} is a binary decision variable which is equal to 1 if the channel of BS m is allocated to user i at time j, and 0 otherwise.
- r_{ij}^{m} is the potential data rate of user i at time j if it served by BS m. This potential data rate is given by $r_{ij}^{m} = r^{m}(x_{ij})$, where x_{ij} is the position of the user at time j and the rate function $r^{m}(x)$ is defined in formula (3.1).

Constraints $\sum_{i=1}^{K} \alpha_{ij}^{m} = 1$ imply that each BS serves exactly one user at each time j. Constraints $\sum_{m=1}^{M} \alpha_{ij}^{m} \leq 1$ imply that each user i is served by at most one base station at each time j. Finally, the last constraints $\alpha_{ij}^{m} \in \{0,1\}$ imply that a feasible solution is a binary vector α. To make the problem easier to solve, we will remove the constraints $\sum_{m=1}^{M} \alpha_{ij}^{m} \leq 1$ by assuming that a user can only be served by the closest BS.

3.4 Existing Algorithms

In this section, we present some of the existing heuristics for channel allocation. These heuristics will be later compared with the heuristics we propose in this chapter.

3.4.1 Greedy allocation

In the greedy scheme, the channel is always allocated to the vehicle with the maximum rate, that is, at each time-slot j the channel of BS m is allocated to a vehicle $i_m^* \in \text{argmax}_i(r_{i,j}^m)$.

3.4.2 Proportional Fair (PF) allocation

Remark that problem (I) is a discrete problem. Even though the number of options is finite, it is NP-hard to find the optimal solution (see, e.g., [43]). Nevertheless, a simple heuristic, called PF-EXP [37], is known to be optimal when the number of users is fixed and that the data rates $r_{i,j}^m$ are time stationary and ergodic, that is, there is no correlation between $r_{i,j}^m$ and $r_{i,j+1}^m$.

The PF algorithm chooses the user with the highest ratio of the current rate to the observed throughput, that is, it chooses the user i who maximizes the ratio $r_{i,j}^m / A_i(j-1)$, where

$$A_i(j) = A_i(0) + \sum_{t=1}^{j} \sum_{m=1}^{M} r_{i,j}^m \alpha_{i,j}^m,$$

is the total allocated rate to user i up to time j ($A_i(0)$ is the initial value for each user). In the long-run when T goes to ∞, this algorithm was shown to be optimal for a stationary and ergodic channel and for a fixed number of users [37].

As already mentioned, the stationarity assumption is not necessarily true for road traffic when all users always move instead of resting in the same place. As can be seen in Fig. 3.1, when users move on a given path, their rate can vary with the distance to the BSs. Thus, the rate process observed by vehicles need not be stationary, and the PF-EXP algorithm need not be optimal for vehicles moving in a network.

3.4.3 Predictive Finite-horizon PF Scheduling ((PF)^{2}S)

In [43], a modified PF algorithm based on predicted future rate was proposed. This algorithm works as follows:

- It predicts future data rates $\hat{r}^m_{i,j}$ of cars in every future slot,
- it estimates future channel allocations $\hat{\alpha}^m$ based on the data rate predictions. As mentioned in Sec. 3.1, the estimations can be computed using either a round-robin policy, a blind estimation, or a local-search method. It is stated in [43] that, out of these three, round-robin is more robust to prediction errors. Given this, we shall use *Round Robin Estimation* (RRE) as the estimation policy for (PS)^{2}S in the numerical comparisons. As a reminder, RRE assumes that future time slots are allocated in a round-robin manner and each user receives an equal number of slots.
- for each time slot j, the BS m chooses the user who maximizes $M^m_{i,j}$, where

$$M^m_{i,j} = \frac{r^m_{i,j}}{\sum_{t=1}^{j-1} \alpha^m_{i,t} r_{i,t} + \hat{\alpha}^m_{i,j} r_{i,j} + \sum_{t=j+1}^{T} \hat{\alpha}^m_{i,t} \hat{r}^m_{i,t}}. \tag{3.2}$$

The index $M^m_{i,j}$ looks similar to that of the PF-EXP algorithm but includes the future allocation. It is related to the gradient of the utility function in (I). In the case of one BS (so that we can omit the index m), provided the future channel allocations $\hat{\alpha}$ can be predicted correctly, an optimal solution to problem (I) can be obtained, as stated in Proposition 3.4.1.

Proposition 3.4.1. *If there exist α^* satisfying $\alpha^*_{i^*,j} = 1$ and $\alpha^*_{i,j} = 0, \forall i \neq i^*_j$, where*

$$i^*_j \in \arg \max_{i \in \{1,2,\ldots K\}} \frac{r_{i,j}}{\sum_{t=1}^{j-1} \alpha^*_{i,t} r_{i,t} + \alpha^*_{i,j} r_{i,j} + \sum_{t=j+1} \alpha^*_{i,t} r_{i,t}}, \tag{3.3}$$

then α^* is the optimal solution of problem (I).

The proof of proposition 3.4.1 is placed after the proof of proposition 3.5.1, since we shall use proposition 3.4.1 as a corollary of 3.5.1.

Note that Condition (3.3) is a sufficient condition for α^* to be an optimal solution of problem (I), but not a necessary condition.

In the next section, we present our heuristic. The motivation for the heuristic comes from the observation that the formula of (PF)^{2}S looks like one-step of the gradient descent with starting point chosen according to the round robin policy when the Round Robin Estimation is used. We may expect to get a better allocation if we do more iterations instead of only one, ensuring that in every iteration the allocation is in the feasible set. To do this we employ a projected gradient algorithm, as described in the next section.

3.5 Projected gradient approach

We shall assume that each BS allocates the channel independently, that is, in a decentralized manner and without coordination with the other BSs. The channel allocation is done by the BS by taking into account the future data rates of the users currently attached to this BS. Since each BS decides independently, we omit the index m of the BS for simplicity.

We propose two heuristic algorithms, Short Term Objective 1 (STO1) and Short Term Objective 2 (STO2), which are presented in the following. The two heuristics use a different method (to be explained below) for estimating the future throughput than the round-robin used in the $(\mathrm{PF})^2$S algorithm. This estimate is based on optimizing the objective with the future mean channel gains as an estimate for the actual realizations. This is similar in spirit to Stochastic Model Predictive Control [44]. The two heuristics differ in the time-scale on which updated future information is used as well as in the dimension of the optimization problem solved at each decision epoch.

Before describing the two heuristics, we explain the ideas common to them. The first step is to relax the integer constraints on the allocation variables in optimization problem (I), so as to obtain the following convex optimization problem

$$\begin{cases} \text{maximize} & O(\alpha) = \sum_{i=1}^{K} \log\left(\sum_{j=1}^{T} \alpha_{ij} r_{ij}\right) \\ \text{subject to} & \\ & \sum_{i=1}^{K} \alpha_{ij} = 1, \quad j = 1, \ldots, T, \\ & \alpha_{ij} \in [0,1], \quad j = 1, \ldots, T, \ i = 1, \ldots, K, \end{cases} \tag{II}$$

which is very similar to the original problem, except that α_{ij} can now be non-integer in $[0,1]$. The relaxed problem (I) can be solved efficiently using the projected-gradient algorithm based on the formula for the projection on a simplex as described below.

Denote by $D = \left\{\alpha \in [0,1]^{K\times T} \ : \ \sum_{i=1}^{K} \alpha_{ij} = 1, \ j = 1, 2, ..., T\right\}$ the feasible set of the relaxed problem. We can obtain a projection on D by projecting independently on simplexes corresponding to each of the time-steps. The procedure for computing the projection Π_D on the set D is formalized in 3.2.2.

The projected gradient algorithm then works as follows. Starting from an arbitrary initial solution $\alpha^0 \in D$, the algorithm computes at each iteration $n = 1, 2, \ldots$ a new feasible solution using the formula

$$\alpha^{n+1} = \Pi_D(\alpha^n + \epsilon_n \nabla O(\alpha^n)), \tag{3.4}$$

where $\nabla O(\alpha^n)$ is the gradient of the objective function at iteration n and $\epsilon_n \in (0,1)$ is the step size at that iteration. A new feasible solution is computed until convergence is reached. In our numerical examples, we have however limited the number of iterations to 20.

Denote by $\tilde{\nabla} O(\alpha) = \Pi_D(\alpha + \epsilon \nabla O(\alpha)) - \alpha$ with the step size $\epsilon \in (0,1)$ small enough. Proposition 3.5.1 below says that if the iterations (3.4) converge, then the resulting allocation is optimal.

Proposition 3.5.1. *If $\alpha^* \in D$ and $\tilde{\nabla} O(\alpha^*) = 0$ then α^* is the optimal value of the relaxed problem* (I).

Proof. (proof of Proposition 3.5.1) The optimal is obtained by proving that for any $\alpha \in D$,

$$\nabla O(\alpha^*)(\alpha^* - \alpha) \leq 0.$$

From lemma 3.2.1, it follows that it is sufficient to prove the above property on D_1. Assuming O is convex function on D_1, we shall prove that if $\alpha^* = (\alpha_i^*)_{i=1,\dots,K} \in D_1$ satisifes

$$\Pi_{D_1}(\alpha^* + \epsilon \nabla(\alpha^*)) = \alpha^* \tag{3.5}$$

where ϵ positive, then

$$\nabla O(\alpha^*)(\alpha^* - \alpha) \geq 0, \text{ for any } \alpha \in D_1$$

i.e, α^* is global optimal of O. Indeed, without loss of generality, we assume that

$$\alpha_1^* + \epsilon \frac{\partial O}{\partial \alpha_1^*} \geq \alpha_2^* + \epsilon \frac{\partial O}{\partial \alpha_2^*} \geq \dots \geq +\alpha_M^* + \epsilon \frac{\partial O}{\partial \alpha_M^*} \geq \dots \geq \alpha_K^* + \epsilon \frac{\partial O}{\partial \alpha_K^*}$$

where M is the largest index such that

$$\frac{1}{M} \sum_{i=1}^{M} \big(\alpha_i^* + \epsilon \frac{\partial O}{\partial \alpha_i^*} - 1\big) \leq \alpha_M^* + \epsilon \frac{\partial O}{\partial \alpha_M^*}.$$

Denote by $\tau = \frac{1}{M} \sum_{i=1}^{M} \big(\alpha_i^* + \epsilon \frac{\partial O}{\partial \alpha_i^*} - 1\big)$, by proposition 10 in [17] we have: $\Pi_{D_1}(\alpha^* + \epsilon \cdot \nabla(\alpha^*)) = \big(\alpha_1^* + \epsilon \frac{\partial O}{\partial \alpha_1^*} - \tau, \alpha_2^* + \epsilon \frac{\partial O}{\partial \alpha_2^*} - \tau, \dots, \alpha_M^* + \epsilon \frac{\partial O}{\partial \alpha_M^*} - \tau, 0, \dots, 0\big)$. Using (3.5) to compare term by term we get:

1. $\alpha_{M+1}^* = \cdots = \alpha_K^* = 0$,
2. $\alpha_{M+1}^* + \epsilon \frac{\partial O}{\partial \alpha_{M+1}^*} \leq \tau, \cdots, \alpha_K^* + \epsilon \frac{\partial O}{\partial \alpha_K^*} \leq \tau$. Now, from the first item we have $\alpha_{M+1}^* = \cdots = \alpha_K^* = 0$. It implies $\epsilon \frac{\partial O}{\partial \alpha_{M+1}^*} \leq \tau, \cdots, \epsilon \frac{\partial O}{\partial \alpha_K^*} \leq \tau$,

3. $\epsilon\frac{\partial O}{\partial \alpha_1^*} = \cdots = \epsilon\frac{\partial O}{\partial \alpha_M^*} = \tau$.

Thus,

$$\begin{aligned}
\epsilon\nabla O(\alpha^*)(\alpha^* - \alpha) &= \sum_{i=1}^{K} \epsilon\frac{\partial O}{\partial \alpha_i^*}(\alpha_i^* - \alpha_i) \\
&= \sum_{i=1}^{M} \epsilon\frac{\partial O}{\partial \alpha_i^*}(\alpha_i^* - \alpha_i) + \sum_{i=M+1}^{K} \epsilon\frac{\partial O}{\partial \alpha_i^*}(\alpha_i^* - \alpha_i), \\
&= \sum_{i=1}^{M} \tau(\alpha_i^* - \alpha_i) + \sum_{i=M+1}^{K} \epsilon\frac{\partial O}{\partial \alpha_i^*}(\alpha_i^* - \alpha_i), \\
&= \sum_{i=1}^{K} \tau\alpha_i^* - \sum_{i=1}^{K} \tau\alpha_i + \sum_{i=M+1}^{K} \big(\epsilon\frac{\partial O}{\partial \alpha_i^*} - \tau\big)(\alpha_i^* - \alpha_i), \\
&= \tau - \tau + \sum_{i=M+1}^{K} \big(\epsilon\frac{\partial O}{\partial \alpha_i^*} - \tau\big)(0 - \alpha_i) \\
&\geq 0.
\end{aligned}$$

The last sum is less than 0 since all its terms are greater than or equal to 0. □

Proof. (proof of Proposition 3.4.1) In fact the condition (3.3) implies that $\tilde{\nabla} O(\alpha^*) = 0$ and from Proposition 3.5.1 we can conclude. □

Solving (I) using the projected gradient algorithm (3.4) requires the knowledge of all the future arrivals which may not be available. Further, the horizon T could be potentially large (tens of minutes giving roughly of the order of 300000 small slots). This means the BS will have to solve a very high dimensional problem every 2 ms.

For the heuristics, we circumvent these two issues as follows. First, we solve (I) for only cars that are actually present in the coverage range and ignore the future arrivals. Second, we reduce the computational complexity in two ways: *(i)* we solve the problem over a shorter horizon; and *(ii)* we compute the future allocations on a larger time-scale rather than the short time-scale of channel allocation slots δ, which is usually in the order of a few milliseconds. The distance travelled in δ ms by a vehicle is typically too small to observe large changes in the mean channel conditions. Therefore, we define the notion of a big-slot over which there is noticeable change in the mean channel conditions. For example, a big-slot can be $500 \times \delta$, giving a value of 1 second for the big-slot when $\delta = 2$ ms. The exact value of a big-slot is an adjustable parameter that can be set by the system designer.

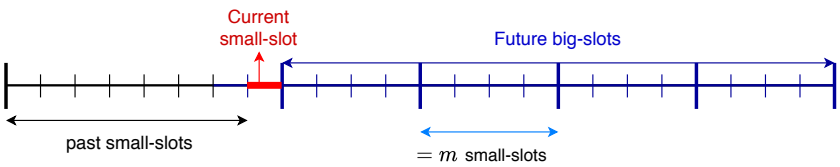


Figure 3.2: Small-slot and Big-slot in STO1.

Next, we describe the two heuristics.

3.5.1 Projected gradient short term objective algorithm (STO1)

Let Δ be the size of the big-slot in absolute time units and let $m = \Delta/\delta$ be the number of small slots in a big-slot. If $\bar{r}_{ij}$ is the average rate in slot j for user i, then $\bar{\rho}_{i\tau} = \sum_{j=(\tau-1)m+t+1}^{\tau m+t} \bar{r}_{ij}$, is the total average data rate that user i will get in big-slot τ. Define $\bar{\alpha}_{i\tau}$ to be the allocation in future big slot τ. These allocations can be interpreted as the fraction of small slots that user i will be allocated in the big-slot τ. We illustrate these two types of time slot in figure 3.2

In small time slot t, let $a_i(t) = \sum_{j=1}^{t} \alpha_{ij} r_{ij}$ be the cumulative allocated rate of user i until time slot t, and $K(t)$ be the number of users inside the coverage range.

The STO1 heuristic works in two steps. At each small slot t, it first solves the allocation for the current small slot and the future big-slots. In the second step, it allocates the channel to the user with the largest fractional allocation for the current slot. These steps are described below:

- **Step 1**– solve the following optimization problem over a short-term horizon of J big-slots using the projected gradient algorithm:

$$\begin{cases} \text{maximize} & \displaystyle\sum_{i=1}^{K(t)} U_i \\ \text{subject to} & \\ & \sum_{i=1}^{K(t)} \alpha_{it} = 1, \\ & \sum_{i=1}^{K(t)} \bar{\alpha}_{i\tau} = 1, \quad \tau = 1, \ldots, J \\ & \alpha_{it}, \bar{\alpha}_{i\tau} \in [0,1], , \quad \tau = 1, \ldots, J, \ i = 1, \ldots, K, \end{cases} \tag{III}$$

where

$$U_i = \log\left(a_i(t-1) + \alpha_{it} r_{it} + \sum_{\tau=1}^{J} \bar{\alpha}_{i\tau} \bar{\rho}_{i\tau} \right).$$

The decision variables in Problem (III) are the allocations in the current small slot, α_{it}, and the allocations in the future big-slots, $\bar{\alpha}_{i\tau}$. Since the future allocations are only computed on the time-scale of big-slots, there is reduction of factor m in the number of variables in (III).

- **Step 2** – allocate the channel to the user who has the *largest allocation computed by* $(\alpha_{it})_{i=\overline{1,K(t)}}$ that is one user which is $\arg\max_i \alpha_{it}$.

The complexity of numerically optimal α computed in step 2 is equal to $20(J+1)\bar{K}\log(\bar{K})$ where 20 is the number of iteration steps of projected gradient in Step 1, $\bar{K}$ is average number of users inside the coverage range, J is the number of big slots.

3.5.2 Projected gradient short term objective algorithm 2 (STO2)

In STO2, we further reduce the complexity by recomputing the future allocations only at the beginning of a big-slot. The future allocation thus computed is then used until the end of this big-slot. If one new user arrives to the system in the middle of big-slot, we just ignore it for this big-slot and wait until the beginning of next big-slot to update the state . Once the allocations for future big-slots are computed, then in every small slot of this big-slot, we apply an index-based policy as in (3.2).

The steps for STO2 are:

- **Step 1** – In each big slot τ, solve the following problem using projected gradient:

$$\begin{cases} \text{maximize} & \displaystyle\sum_{i=1}^{K(\tau)} U_i \\ \text{subject to} & \\ & \sum_{i=1}^{K(\tau)} \bar{\alpha}_{i\tau} = 1, \quad \tau = 1,\dots,J, \\ & \bar{\alpha}_{i\tau} \in [0,1], \quad \tau = 1,\dots,J,\ i = 1,\dots,K, \end{cases} \tag{IV}$$

where

$$U_i = \log\left(a_i\left((\tau-1)m\right) + \sum_{\tau=1}^{J} \bar{\alpha}_{i\tau}\bar{\rho}_{i\tau}\right).$$

Here $a_i((\tau-1)m)$ is the total data rate received by user i up to big slot τ. The other quantities are the same as for algorithm STO1.

- **Step 2** – Inside a big-slot, in each small slot j, compute M_{ij} as in (3.2) where the future allocation $\hat{\alpha}$ is the solution $\bar{\alpha}$ of (IV).

Note that Step 1 in STO2 is computed only once every big-slot unlike m times in every big-slot as in STO1. By doing this, we further reduce the number of computations almost by a factor of m times since we calculate $\bar{\alpha}$ in each big-slot only.

3.6 Numerical results

We now compare the utility of the proposed heuristics with the PF-EXP, $(\text{PF})^2$S and a greedy algorithm. For the $(\text{PF})^2$S the future allocation was done using the round robin algorithm.

Denote by

$$O^A = \sum_{i=1}^{K} \log\left(\sum_{j=1}^{T} \alpha_{ij}^A r_{ij}\right),$$

the total reward of algorithm A and by $\bar{O}^A = \frac{1}{K} O^A$ its average reward over K users. Given two algorithms A and B, the ratio between A and B equals $\exp(\bar{O}^A - \bar{O}^B)$. The percentage of improvement of algorithm A over B is computed as $(\exp(\bar{O}^A - \bar{O}^B) - 1) \cdot 100\%$.

Due to the logarithm in the objective function, taking a different unit of measure for the rate will give a different percentage of improvement between algorithms. Although logarithm is an increasing function, we can know which algorithm is better than the other, but we will not get a consistent percentage of improvement across different units of measure. Therefore, by taking the difference as above we construct a consistent criterion for comparison.

3.6.1 One road network

In the first set of simulations, there is only one base station and one straight road. The road length is taken to be $L = 1000$ m with 0 at the leftmost edge. The closest point on the road to the BS is at $x = 500$ m. The data rate at position x along the road is given by:

$$r(x) = \eta \cdot (1 + \kappa \exp(|x - 500|/\sigma)), \tag{3.6}$$

where $\kappa \geq 0$ is a real number and η is uniform random variable whose range will be in $[0.7, 1.3]$ unless stated otherwise.

Remark 3.6.1. *The method for generating the noise η need not be necessarily multiplicative as in* (3.6)*. Our heuristics can be used as long as the means*

of the future data rate are known. In this chapter, we limit the numerical evaluation to the form in (3.6).

A sample path of $r(x)$ is shown in Fig. 3.3,3.4. This function has the highest mean at the mid-point of the segment and the lowest mean at the two end points. We emphasize the algorithm itself is independent of the rate function. We chose the above rate function for convenience.

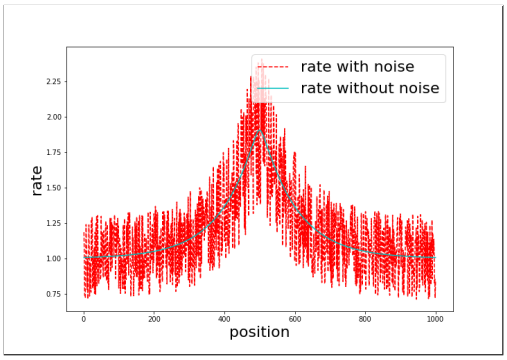


Figure 3.3: $\kappa = 1$.

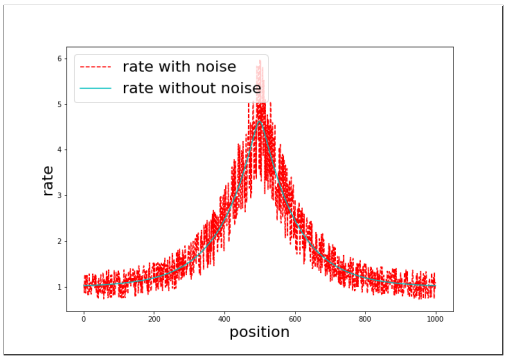


Figure 3.4: $\kappa = 4$.

Figure 3.5: Sample path of data rate at various positions along the road. $\sigma = 100$, and $\eta \in [0.7, 1.3]$.

The time horizon T was $4,000,000$ small time slots which corresponds to 8000 seconds (slightly more than two hours). The big slot length Δ for our projected gradient short term objective algorithm was taken as 1 second or equivalently 500 small time slots.

Homogeneous vehicle velocities

First, we show the results when all vehicles move with the same velocity which is taken to be $v = 25$ m/s. That is, there are $N = 20,000$ spatial small slots in the coverage range and $J = 40$ seconds. A new car enters through the left edge in every second with probability p.

Figure 3.6 shows the average utility obtained by a vehicle for each of the four algorithms as a function of the arrival probability p. Figure 3.7 shows the percentage of improvement of the three other algorithms compared to PF-EXP. The proposed algorithm does better than PF-EXP and more importantly better than $(\text{PF})^2$S. Although, we have shown the greedy algorithm for comparison, we emphasize that greedy is not practically implemented because it can be very unfair to users that have heterogeneous rates. In the

simulated scenario, all vehicles move along the same road and observe statistically identical but position-dependent radio conditions during their stay. These conditions are rather favorable for the greedy algorithm.

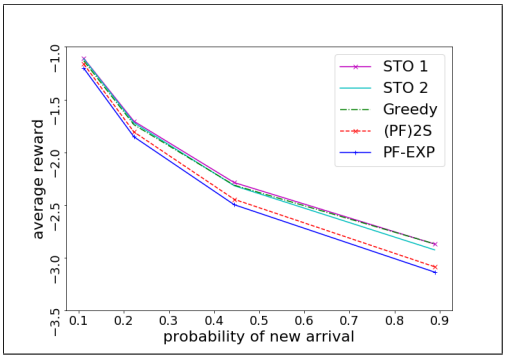


Figure 3.6: Average reward per car. Homogeneous velocities.

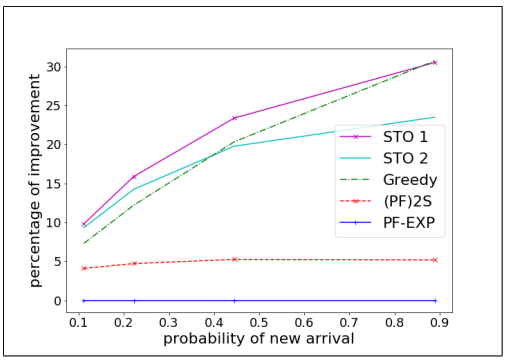


Figure 3.7: Percentage of improvement over PF-EXP. Homogeneous velocities.

Comparison with the upper bound

Next, again for homogeneous velocities, we also include the solution of the relaxed problem (I) but for a smaller road length and shorter horizon because it is computationally expensive. The parameters for this setting are: $L = 100$ m, $J = 40$ s, $T = 500$ s, and the other parameters are the same as in the homogeneous case. We assume that the relaxed algorithm knows the future arrivals as well as the future data rates exactly whereas the other algorithms do not know this information. The solution to the relaxed problem gives an upper bound on the optimal solution of the original problem (I).

Figures 3.8 and 3.9 plot the average reward per car and percentage improvement for the five algorithms with respect to PF-EXP. It can be observed that the proposed algorithm is quite close to the upper bound in this scenario.

In the following, unless otherwise specified, the parameters are chosen as follows: $\kappa = 4, \sigma = 100$, big slot $\Delta = 1s$ and the short-term horizon J is the maximal remaining staying time of the users that are currently inside the system. We calculate the allocation plan every one second. From now on, we do not compare STO 1 because STO 1 takes much longer to run and may not be computationally interesting on small time-scales. Also, we also do not show the performance of greedy here since some users may starve in a greedy allocation leading to a value of $-\infty$.

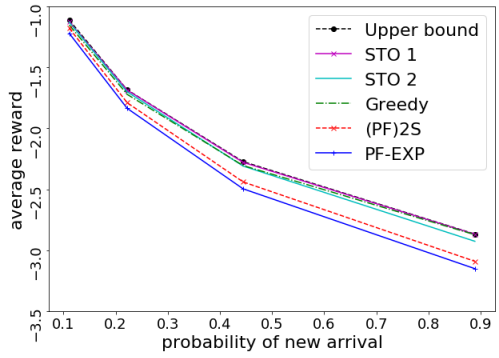


Figure 3.8: Average reward per car. Includes the upper bound from the solution of (II). Small setting of homogeneous velocities.

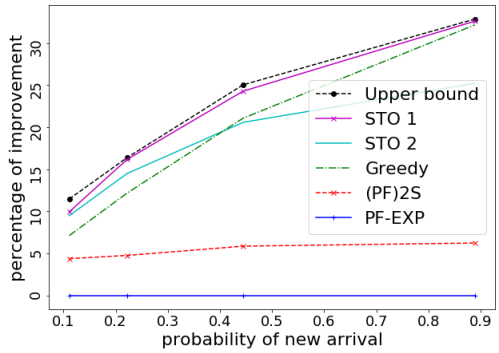


Figure 3.9: Percentage of improvement over PF-EXP. Small setting of homogeneous velocities.

3.6.2 Network simulation with SUMO

Simulation of Urban MObility application (SUMO) [42] is an open source software designed for simulating mobility of vehicles in large traffic networks. One of the features of this simulator is that we can import maps of different cities and simulate realistic mobility traces. We use this application to simulate the complex driving dynamic systems in a specific region of Toulouse city to have an objective comparison of our heuristics against existing algorithms in realistic scenarios.

The performance evaluation of heuristics in done in two steps: in the first step SUMO is used for generating the mobility traces of vehicles. These traces are then fed to a Python script which implements the different heuristics and computes the value of the objective function.

A simple network with 1 BS

Let us consider the network presented in in Figure 3.10. There are two classes of users: one that arrives from A then moves along the long road to B and D (the blue one), and another one that arrives from A then moves to B and then to C (the red one). If we apply the greedy heuristic in this situation, then many users of the second class are never allocated the channel. This is the reason we do not show performance of the greedy algorithm for this scenario. Figure 3.11 shows the numerical results for this case. In this scenario, it was observed that PF-EXP always gives priority to the new arrivals no matter what the initial value is. This leads to a higher sub-optimality of PF-EXP since the other heuristics focus on users that are closer to the base station

and have a higher quality channel.

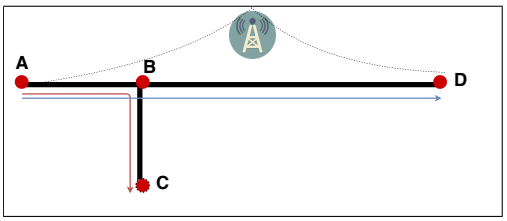


Figure 3.10: Utility Comparison: STO, (PS)^{2}S and PF-EXP.

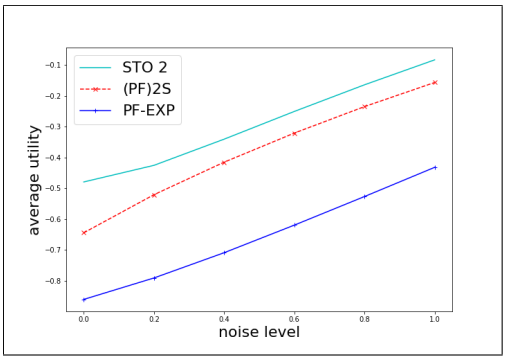


Figure 3.11: Utility Comparison: STO, (PS)^{2}S and PF-EXP.

Place Wilson (Toulouse) scenario with 2 BSs

In this scenario, we evaluate the algorithms on users moving in the Place Wilson area of Toulouse with two BSs as shown in Fig. 3.12. The average utilities of the different heuristics are shown in Fig. 3.13. The various parameters for the rate function are the same as those indicated at the beginning of this section. It took 219 seconds, 229 seconds, 433 seconds and 833 seconds respectively to run greedy, PF-EXP, (PS)^{2}S and STO2 for simulating 1.07 hours of traffic with 483 users (including cars, buses, and bicycles). The staying times of the users varied from $2s$ to $361s$. We do not show greedy in the utility comparison since there were several starving users in this case. As expected, there is a trade-off between the quality of the solution and the computation time. STO2 takes longer to solve but gives a better allocation. Now, we change some of the parameters to see how the performance of the heuristic is influenced by these parameters.

Figure 3.14a, 3.14b, 3.14c plot the average utilities for different values of κ with the same H and Δ. The gap between STO2 and (PS)^{2}S become larger when κ increases. Figure 3.15a, 3.15b, 3.15c and 3.15d illustrate the average utilities for different values of J with same κ, Δ. Remark that we assume (PS)^{2}S and STO2 use the same information, so in (PS)^{2}S the future information is estimated until J as well. It is seen that the more information we have, the better (PS)^{2}S and STO2 perform.

Figure 3.16a, 3.16b, 3.16c illustrate for different values of big-slot Δ with the same values of J and κ. The performance of STO2 is almost the same for these different values of Δ but the running time is significant faster.

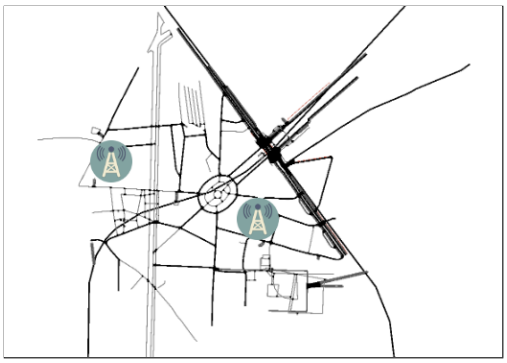


Figure 3.12: Place Wilson, Toulouse with 2 Free Mobile BSs 4G.

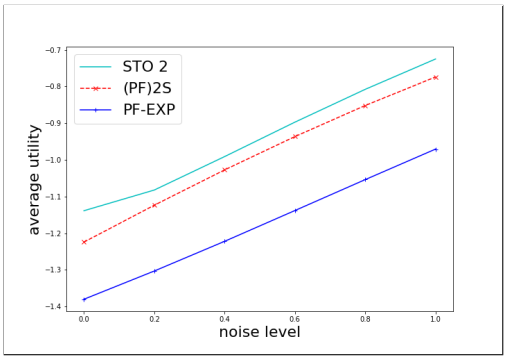


Figure 3.13: Utility Comparison: STO2, $(PS)^2S$ and PF-EXP.

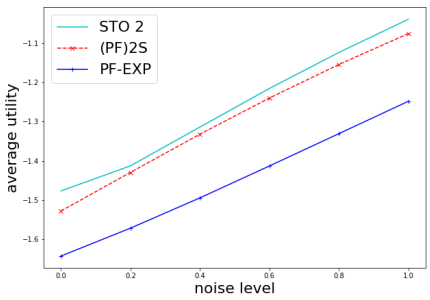
Jardin de Plantes (Toulouse) scenario with 4 BSs

In the final set of simulations, we take another area of Toulouse called Jardin des Plantes with four BSs as shown in Fig. 3.17. The average utilities for this scenario are plotted in Fig. 3.18. It took 976 seconds, 1009 seconds, 1502 seconds and 3403 seconds to run greedy, PF-EXP, $(PS)^2S$ and STO2 for a 1.05 hours of traffic with 740 users (including cars, buses, motorbikes, bicycles and pedestrian). Again, we do not show greedy in the utility comparison since there are several starving users in this case.

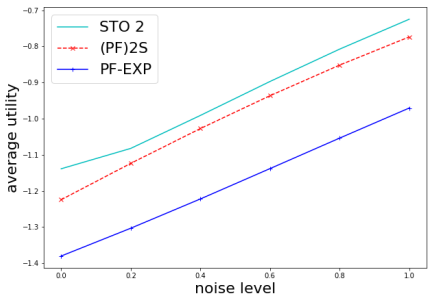
3.7 Summary

We proposed two heuristics that use future mean channel gain information to improve the utility of users in cellular networks. In order to reduce the computational complexity, they solve the problem over a shorter time horizon as well as on a larger time-scale. It was shown on numerical experiments carried out on traces generated from realistic mobility patterns that these heuristics give better utility compared to PF as well as $(PS)^2S$ algorithms. However, the power control has been missing in the problem in this chapter, therefore in chapter 4, we shall consider the joint of channel allocation and power control problem.

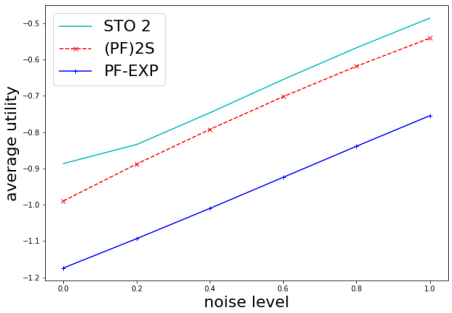
Regarding the computation time, as we mentioned above, STO1 requires a large computation time and may not be possible to run in real-time. And STO2 is also heavy when the system is large. It opens another direction of research that is how to reduce the computation time of the two algorithms. In chapter 5 we shall learn STO1 algorithm using DFNN based method to have a faster algorithm which performs close to these proposed algorithms.



(a) $\kappa = 2$.

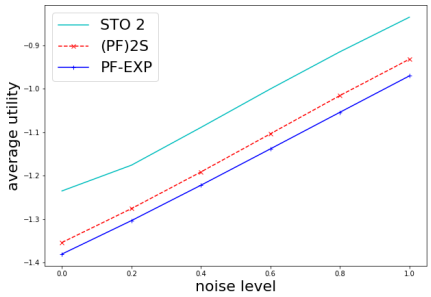


(b) $\kappa = 4$.

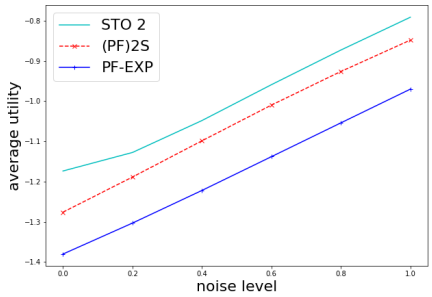


(c) $\kappa = 6$.

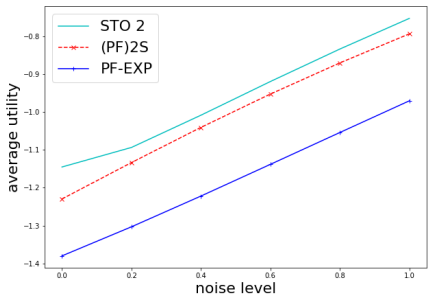
Figure 3.14: Place Wilson, utility comparison for different κ.



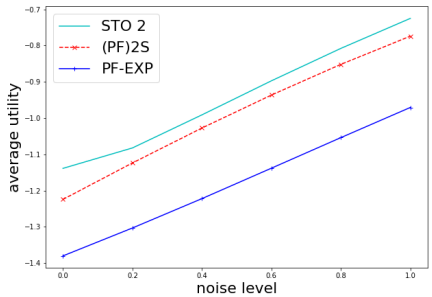
(a) $J = 20s$.



(b) $J = 60s$.

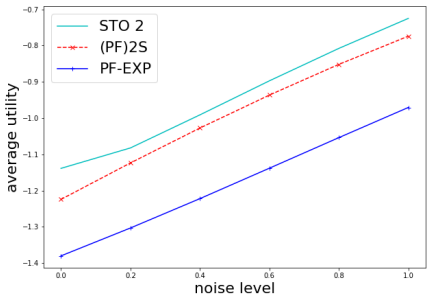


(c) $J = 120s$.

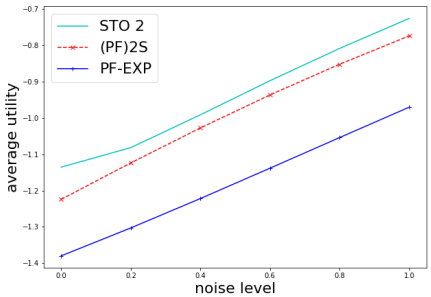


(d) $J =$ maximum remain time in term of big-slot of all users inside the system.

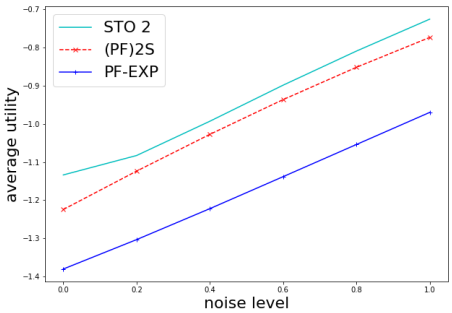
Figure 3.15: Place Wilson, utility comparison for different short time horizon J. Here we assume that (PS)^{2}S and STO2 use the same information, so in (PS)^{2}S the future information is estimated until J as well.



(a) $\Delta = 2s$. ($833s$ to run STO2).



(b) $\Delta = 4s$ ($568s$ to run STO2).



(c) $\Delta = 6s$ ($438s$ to run STO2).

Figure 3.16: Place Wilson, utility comparison for different Δ. The performance are almost the same, but the running time is much faster when increasing big-slot.

Figure 3.17: Jardin de Plantes, Toulouse with 4 BSs (Free and SFR) type 4G.

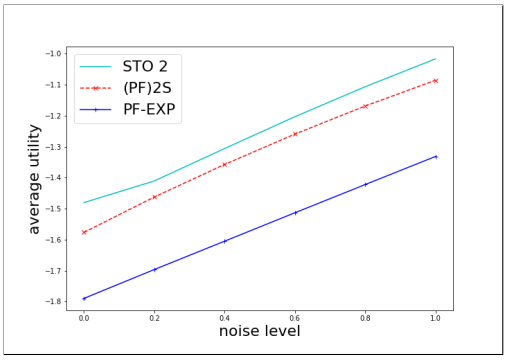


Figure 3.18: Utility Comparison: STO2, $(PS)^2S$ and PF-EXP.

Chapter 4

JOINT POWER CONTROL AND CHANNEL ALLOCATION

In this chapter, we consider the joint channel allocation and power control problem on the downlink, that is the transmit power of the BS can be varied subject to an average power constraint that does not exceed a certain available power budget to the BS. The objective is to design algorithms that work well for system with mobile users in order to improve proportional fairness. We propose two downlink scheduling algorithms that take advantage of partial information on future channel conditions, that is the mean of the channel gains based on predicted future positions of the users, for improving the sum utility. The scheduling model allows for both power control and channel allocation. The objective of the scheduler is the long-term utility under an average power constraint. The two algorithms incorporate the channel predictions in their decisions. The STO1 algorithm computes the decision in each slot based on the means of future channel gains. Depending on the horizon considered, this can require solving a large-dimensional problem in each slot. The STO2 algorithm reduces the dimensionality by operating on two time-scales. On the slower scale it computes an estimation over a larger horizon, and in the faster scale of a slot, it computes the decision based on a shorter horizon. Numerical experiments with both fixed number of users as well as a dynamic number of users show that the two algorithms provide gains in utility compared to agnostic ones.

4.1 Introduction

Downlink scheduling and power control has been widely investigated for wireless networks [39, 14, 29, 40]. In each time-slot, one or multiple base-stations have to decide the users to serve and the transmit powers with the objective of maximizing the sum of the utilities of the users. Two different types of utilities can be defined: (i) opportunistic [39, 14]; and *(ii)* long-term[29, 62]. In the opportunistic model, the utility function operates on the rate obtained in a time-slot whereas in the long-term model, the utility operates on the average rate obtained over an horizon.

The focus of this chapter is on the long-term utility model. When transmit powers cannot be varied, the celebrated Proportional-Fair algorithm (see [37] and references therein) is known to work well when the utilities are logarithmic functions of the total rate. This algorithm belongs to the class of gradient-based algorithms that choose the user that maximizes the marginal utility or the gradient of the utility function. In [29] the gradient-based solution was then extended to the setting in which joint power control and user scheduling is possible.

We revisit this problem in the context of vehicles which share their itineraries with the decision maker. With the availability of SINR maps in urban zones, the decisions can now be based also upon the future channel conditions of the users (or vehicles) [43]. The future channel conditions are, however, not known exactly as the they vary randomly in time. The SINR maps are assumed to give the expected values of the channel gains on the routes takes by the users. With this additional information on the future expected channel gains, performance improvements can be expected compared to the setting when this information is not available.

We consider a downlink scheduling and power control problem for one base station with an average power constraint. The objective is the sum utility of the users, and the dependence of the utility function on the channel allocation and transmit power is similar to that in [29] with only one sub-channel and we do not have queues. In that respect, our model is a special case of that in [29]. However, the scheduling algorithms that we propose can be extended to the multiple sub-channel case though for this chapter we restrict ourselves to the simpler case of one sub-channel. There are two difference with the models studied previously. The first, and the main difference is that the base station is also aware of the mean channel gains in the future slots, and the second one is that we include an average power constraint over time in the optimization problem.

4.1.1 Contributions

We propose two heuristics for that use information on future channel conditions in order to improve the total utility. Both the heuristics were first proposed in [51] in a setting without power control. The first one optimizes over future time-slots in every current slot whereas the second one reduces the complexity by optimizing over the future time-slots only once in a certain number of slots. The performance of these heuristics will be evaluated on three types of stochastic models for the channel gain processes. In the first, the channel gains are a stationary process with fixed mean channel gains (which can be different for different users) whereas in the second model, the mean channel gains are themselves varying on a slower time-scale to that of the channel gains themselves. Finally in the third model, the mean channel gains vary in every slot. For three models, these heuristics are shown to perform better than the other setting in which future information is not available.

4.1.2 Related work

Scheduling on the downlink but with fixed power examined in numerous papers [11, 66, 13]. For a logarithmic utility function, the Proportional-Fair (PF) scheduler has been known to be optimal for stationary channel conditions [37] and its performance has been analyzed in both static and dynamic user scenarios [10]. A restless bandit framework for network utility maximization for channel states modeled as partially observable Markov chains is proposed in [40].

In [3], future rates are assumed to be known accurately over the optimization horizon and improvement in data rates and fairness is compared with algorithms such as greedy (or max-rate) and equal share. Proportional-Fair algorithms with partial information on future channel conditions have been proposed in [7] (on short-time scales) and [43] (on longer-time scales). The algorithm in [43] is based on SINR maps that can be obtained from vehicles or users that were present earlier. They proposed a PF-like index algorithm that takes into account the future rate allocations using round-robin or a few other heuristics. In [51], we proposed two heuristics that solve in each time-slot the utility maximization problem over a short-term horizon assuming that the means of the future rates are known over this short horizon.

The joint optimization of channel allocation and transmit power control has been investigated for different multiplexing schemes such as like CDMA [62] and OFDM [29]. The proposed algorithms based their decisions on the current channel conditions and previous decisions. In the context of

high-speed trains, [77] solves the opportunistic utility maximization problem assuming all the future rates are known and with average power constraints.

4.2 Problem formulation

Consider a base station with K mobile users in its coverage range. In time-slot t, the base station transmits to user i with power $p_i(t)$. Assume that user i has a channel gain of $\gamma_i(t)$ in slot t. These gains will be assumed to stochastic and independent between vehicles but not necessarily stationary for each user.

The received data rate for user i is computed according to the Shannon formula:

$$r_i(t) = x_i(t) \log \left(1 + \frac{\gamma_i(t) p_i(t)}{x_i(t)}\right), \tag{4.1}$$

where $x_i(t)$ is the fraction of the channel assigned to user i is slot t. User i gets a utility of $U_i(z)$ when it obtains an average rate of z. In order to keep the notation light, we will write $r_i(t)$ as a shorthand for $r_i(x, p, \gamma)$. Let $\mathcal{S}$ be the appropriate dimensional simplex. We shall the use the notation

$$[x_i(t)] \in \mathcal{S} \tag{4.2}$$

to mean that the vector $[x_1(t), \ldots, x_K(t)] \in \mathcal{S}$, where $\mathcal{S}$ is the K-dimensional simplex.

The utility function will be assumed to concave and differentiable. A widely-used class of utility functions is that of the α-fair functions [45] that are parameterized by $\alpha \geq 0$:

$$U_i(z) = \begin{cases} \frac{z^{1-\alpha}}{1-\alpha}, & \alpha \neq 1; \\ \log(z), & \alpha = 1. \end{cases}$$

The special case of $\alpha = 1$ is also known as the proportional-fair utility function.

The objective of the base station is to choose the power and the channel allocation so as to maximize the total utility of these K users over a horizon

of T time slots. That is, the base station solves the optimization problem:

$$\text{maximize} \sum_{i=1}^{K} \left[U_i \left(\frac{1}{T} \sum_{t=1}^{T} r_i(t) \right) \right] \tag{OPT}$$

$$\text{subject to } [x_i(t)] \in \mathcal{S}, \forall t; \tag{4.3}$$

$$\frac{1}{T} \sum_t \sum_i p_i(t) \leq \bar{P}; \tag{4.4}$$

$$\sum_i p_i(t) \leq P_{max} \, \forall t. \tag{4.5}$$

Here $\bar{P}$ is the average transmit power budget available to the base station, and (4.4) is the average power constraint. This constraint also makes the problem different from that in [29] where there was no constraint on the average power.

Remark 4.2.1 (Short-term fairness)**.** *A drawback of the utility function defined on the average rate is that if the scheduler knows that, for a particular user, the channel gain may be very high some time in the future then it might wait until this time to serve this user. This user may be starved of allocations in the short-term and the solution may be unfair to this user on short-time scales. One way to resolve the short-term unfairness is to introduce additional quality of service constraints such as requiring each user be scheduled at least once every given number of slots. This constraint was imposed in, for example, [43]. We can also include this constraint in the optimization problem. For simplicity, we do not impose it. We believe the results will be similar as long as this constraint is not very restrictive.*

Remark 4.2.2 (Fractional channel allocation)**.** *In* (OPT) *we have allowed for fractional channel allocations. If the system imposes a binary constraint, that is only one user on one channel in any given slot, then these constraints can be imposed in OPT as well as in the algorithms we propose. In the experiments with a logarithmic utility function, we observed that allocations were mostly binary. So, we expect the qualitative conclusions will be valid whether allocations are binary or not.*

Remark 4.2.3 (Maximum power constraint)**.** *For conciseness, we shall not write the maximum power constraint explicitly in the optimization problems that we will define from now on. This constraint will be implicit and assumed to be applicable in all slots.*

The current literature mostly solves (OPT) when the base station is aware of only the channel gains in the current slot[1]. For vehicles sharing their

[1]We shall use slot and time-slot interchangeably to mean the decision making instants.

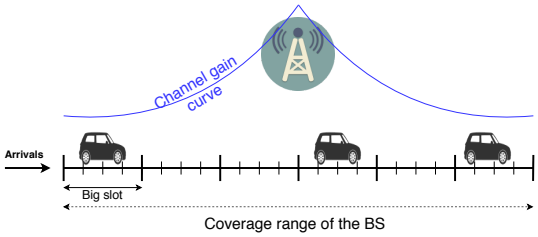


Figure 4.1: Mobility model.

itineraries, partial information on the future channel conditions could also be available to the decision maker in the current time-slot. We assume that this partial information is in the form of the mean of the channel gains in the future slots. One would expect to improve the value of the objective when this information is incorporated in the decision-making.

4.2.1 Channel gains

The heuristics will be evaluated on three different stochastic models for channel gains: *(i)* the stationary model in which, for each vehicle i, $\gamma_i(t)$ are independent and identically distributed across t with mean $\bar{\gamma}_i$; *(ii)* non-stationary and slowly varying model in which, for each vehicle i, the mean of the channel gains varies on a slow time-scale. For example, for a time-slot of 2 ms, the means may vary every 100 slot or 200 ms; and *(iii)* a mobility model on a stretch of road, where cars can come and leave as shown in Fig. 4.1. For this model $\gamma_i(t)$ are non-stationary, and also the mean of $\gamma_i(t)$ changes every slot.

Partial information on future channel conditions will be shown to be helpful for the second model and the third model. Nevertheless, the first (or the stationary) model will still be useful to illustrate the benefits of being able to vary the allocated power under an average power constraint.

4.3 Algorithms

In this section, we present the three algorithms that will be evaluated in the numerical experiments. The first one is the standard gradient-based scheduling which will be taken as the baseline. The other two are the ones we propose in this chapter.

4.3.1 Locally optimal algorithm

The locally optimal algorithm is a special case of the one in [29] for the scenario in which there is only one sub-channel and the power budget is $\bar{P}$ in each slot. We remark that the algorithm can be handle the multiple sub-channel case but that in this chapter our focus is on the single sub-channel case. When there is no average-power constraint as in [29], this algorithm chooses maximizes the objective function computed without the knowledge of the future scheduling decisions. That is, in slot t, the scheduler solves

$$\text{maximize} \sum_{i=1}^{K} \left[U_i \left(\frac{1}{t} \sum_{t=1}^{t} r_i(t) \right) \right] \tag{LA}$$

$$\text{subject to } [x_i(t)] \in \mathcal{S}, \forall t; \tag{4.6}$$

$$\sum_i p_i(t) \leq \bar{P}, \forall t. \tag{4.7}$$

Here, the past rate $r_i(s)$ for $s = 1, 2, ..., t-1$ are known to the scheduler, so the decision variables are the channel allocations $x_i(t)$ and the transmit powers $p_i(t)$ in slot t. The maximum power constraint is set to $\bar{P}$ in order to make the comparison fair with algorithms which are subject to the average power constraint of $\bar{P}$. For the logarithmic utility function, we shall call this the Proportion Fair (PF) solution.

4.3.2 Short-term Objective 1 (STO1)

We begin by defining a big-slot to be B consecutive time-slots. The first heuristic we propose uses future mean channel gains to improve the objective. It includes in its decisions the future channel allocations and transmit powers but only on the scale of big-slots. In each time-slot, the allocations are recomputed for the current time-slot as well as the future big-slots. The intuition behind this is to maximize in each slot the long-term objective based on the best information available. We explain its workings in the context of the non-stationary slowly varying channel model and the mobility model.

For the slowly varying model, B will be the number of time-slots during which mean channel gain remains constant whereas in for the mobility model B can be set by the system designer depending on how fast the mean channel gains vary. Let $\hat{T} = T/B$ be the number of big-slots in the horizon, and let $\hat{\tau}_t \in \{1, \ldots, \hat{T}\}$ be the big-slot to which time-slot t belongs to, and let θ_t be the number of slots remaining in big-slot τ_t not including t, that is

$$\theta_t = (\tau_t + 1)B - t.$$

We shall use the notation $\hat{x}$ for a quantity that is computed over a big-slot. For example $\hat{p}_i(\tau)$ will denote the power used in all the slots inside big-slot τ. Similarly,

$$\hat{r}_i(\tau) = B\hat{x}_i(\tau)\log\left(1 + \frac{\bar{\gamma}_i(\tau)\hat{p}_i(\tau)}{\hat{x}_i(\tau)}\right) \tag{4.8}$$

is the total rate obtained by vehicle i in big-slot τ when it is served $\hat{x}_i(\tau)$ fraction of time at a transmit power of $\hat{p}_i(\tau)$. Note that here the rate is computed assuming that the channel gain is its mean value in big-slot τ. With slight abuse of notation,

$$\hat{r}_i(\tau_t) = \theta_t\hat{x}_i(\tau_t)\log\left(1 + \frac{\bar{\gamma}_i(\tau_t)\hat{p}_i(\tau_t)}{\hat{x}_i(\tau_t)}\right), \tag{4.9}$$

shall denote the total rate in the remaining slots in current big-slot τ_t. Also, define

$$P_t = T\bar{P} - \sum_{i=1}^{K}\sum_{s=1}^{t-1} p_i(s)$$

to be the total remaining power available to the scheduler in slot t.

In each slot, STO1 maximizes

$$\sum_{i=1}^{K} U_i\left(\frac{1}{T}\left(\sum_{s=1}^{t-1} r_i(s) + r_i(t) + \sum_{\tau=\tau_t}^{\hat{T}} \hat{r}_i(\tau)\right)\right) \tag{STO1}$$

subject to

$$[x_i(t)] \in \mathcal{S};\ [\hat{x}_i(\tau)] \in \mathcal{S},\ \tau = \hat{\tau}_t, \ldots, \hat{T}; \tag{4.10}$$

$$\sum_{i=1}^{K}\left(p_i(t) + \theta_t\hat{p}_i(\tau_t) + \sum_{\tau=\tau_t+1}^{\hat{T}} B\hat{p}_i(\tau)\right) \leq P_t. \tag{4.11}$$

The variables in this problem are $[p_i(t)]$ and $[\hat{p}_i(\tau)], \tau = \tau_t \ldots \hat{T}$, and the corresponding channel allocations. In (4.11), the LHS is the total transmit power starting from the current slot which has to be less than the remaining power P_t.

For the mobile model, instead of solving (STO1-Opt) over the whole horizon T, we solve it on a shorter horizon, which is equal to the maximum staying time of the users currently inside the system. This shorter time horizon can vary from one slot to another, and it explains the words 'short-term objective' in the name of the algorithm. The advantages of this is to reduce the computation time which can be helpful when the algorithm has to be executed every 1ms.

4.3.3 Short-term Objective 2 (STO2)

The STO1 algorithm recomputes in each time slot the optimal solution of the future big-slots. In STO2, we recompute the solution of the future big-slots only at the beginning of each big-slot. Inside a big-slot, we compute only the solution for the current slot assuming the solution for the future big-slots to be the same as that computed at the the start of the current big-slot. That is if $t \equiv 1 \pmod{B}$, STO2 first maximizes

$$\sum_{i=1}^{K} U_i \left(\frac{1}{T} \left(\sum_{s=1}^{t-1} r_i(s) + \sum_{\tau=\tau_t}^{\hat{T}} \hat{r}_i(\tau) \right) \right) \tag{STO2-Big}$$

subject to

$$[\hat{x}_i(\tau)] \in \mathcal{S},\ \tau = \hat{\tau}_t, \ldots, \hat{T} \tag{4.12}$$

$$B \sum_{i=1}^{K} \sum_{\tau=\tau_t}^{\hat{T}} \hat{p}_i(\tau) \leq P_t. \tag{4.13}$$

The variables in this problem are $[\hat{x}_i(\tau)]$ and $[\hat{p}_i(\tau)], \tau = \tau_t \ldots \hat{T}$.

Next, in each slot t, we compute the optimal allocation and transmit power assuming that the allocations and transmit powers in the future big-slots are those computed from solving (STO2-Big). In slot t, STO2 maximizes

$$\sum_{i=1}^{K} U_i \left(\frac{1}{T} \left(\sum_{s=1}^{t-1} r_i(s) + r_i(t) + \sum_{\tau=\tau_t}^{\hat{T}} \hat{r}_i(\tau) \right) \right) \tag{STO2-Small}$$

subject to

$$[x_i(t)] \in \mathcal{S},\ [\hat{x}_i(\tau_t)] \in \mathcal{S}; \tag{4.14}$$

$$\sum_{i=1}^{K} (p_i(t) + \theta_t \hat{p}_i(\tau_t)) \leq P_t - \sum_{i=1}^{K} \sum_{\tau=\tau_t+1}^{\hat{T}} B\hat{p}_i(\tau). \tag{4.15}$$

The variables in this problem are $[x_i(t)]$, $[p_i(t)]$, $[\hat{x}_i(\tau_t)]$ and $[\hat{p}_i(\tau_t)]$. As can be seen, in a slot the dimension of the problem is no bigger than the one for STO1. As in STO1, for the mobile model, STO2 solves (STO2-Big) and (STO2-Small) over a shorter horizion which is the maximum staying time of the users currently in the coverage range. The pseudo code for STO2 is shown in Algorithm 4.

In terms of computational effort, compared to STO1, STO2 solves a lower dimensional problem in each slot except at the starting of every big-slot where the dimension is same as for STO1. Thus, one can except it to be faster but further from the optimal solution.

ALGORITHM 4: The STO2 algorithm

```
t ← 1
while t ≤ T do
    if t ≡ 0 mod B then
        Solve (STO2-Big) and obtain x̂(τ) and p̂(τ).
        Solve (STO2-Small)
    else
        Solve (STO2-Small)
    end
end
```

4.4 Numerical experiments

The numerical experiments were run in Python, and all optimization problems were solved using the python package CVXPY [20] and the solver MOSEK. The results will be presented according to the channel-gain models. For the stationary and slowly varying channel models, we assume that the number of users is fixed. The third model will be evaluated in a dynamic setting in which users will arrive and leave the network.

In all the experiments, $\bar{P}$ is set to 15 and P_{max} is set to 30. Whenever we show any performance measure of the optimal solution, it will be assumed to mean that the optimal is computed assuming all the future channel gains are known exactly.

4.4.1 Stationary channel

For the first experiment, we take $K = 4$, that is four users, and a logarithmic utility function for every user. The vector of means $[\bar{\gamma}_i] = [6.76, 5.45, 4.35, 1.31]$. The channel gain in slot t for user i is generated as follows:

$$\gamma_i(t) = \bar{\gamma}_i A_i(t; \eta) \tag{4.16}$$

Here $A_i(t; \eta)$ is a sequence of i.i.d. uniform random variable in the range $[1 - \eta, 1 + \eta]$. We shall refer to η as the *noise level*. It is assumed that $A_i(t)$ and $A_j(t)$ are assumed to be independent for $i \neq j$. Varying η from 0 to 1 changes the variance of $A_i(t)$ from low to high. For $\eta = 0$, the channel gains become deterministic and known to the decision maker.

Remark 4.4.1. *The method for generating $\gamma_i(t)$ need not be necessarily multiplicative as in* (4.16)*. Our heuristics can be used as long as the means of the future channel gains are known. In this chapter, we limit the numerical evaluation to the form in* (4.16)*.*

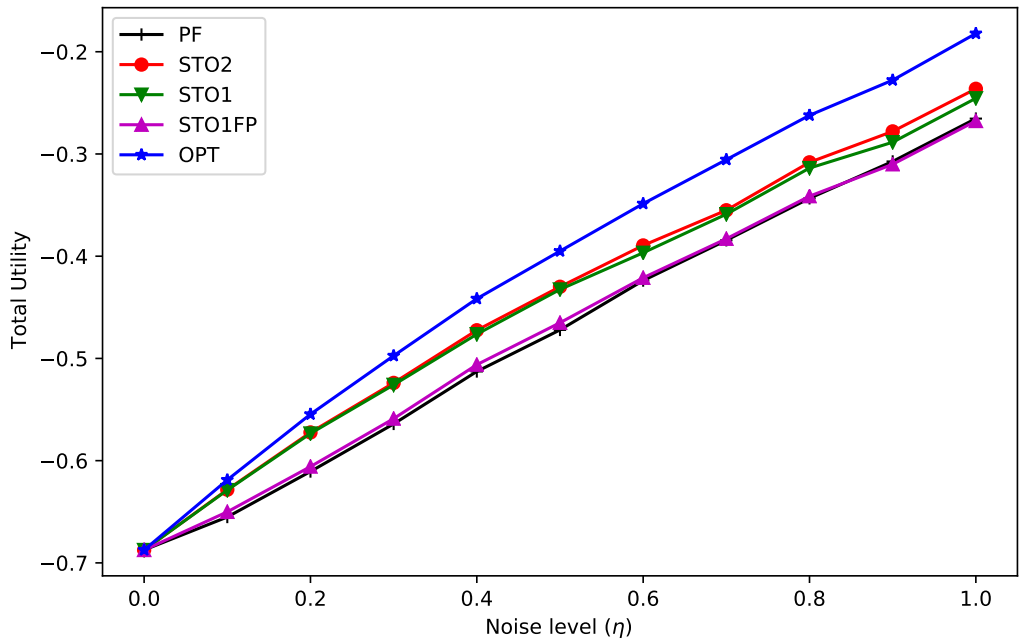


Figure 4.2: Total utility as a function of noise level. Channel is stationary. Log utility function.

Figure 4.2 shows the total utility as a function of the noise level η. The time horizon T was taken to be 500 with 5 big-slots, that is, one big-slot has $B = 100$. If the scheduling slot is 1 ms as in 4G [19], then the scheduling horizon is of 500 ms. Five sample paths for channel gains were generated, and the plot shows the average of these 5 samples. The label PF is for the local optimization algorithm. The suffix FP attached to STO1 means that STO1 was run wilth a fixed power budget of $\bar{P}$ in each slot.

As expected, allowing for an average transmit power constraint and using future information (even if it is just the mean channel gains) improves the utility. When $\eta = 0$, all the algorithms are equivalent and give the same utility since the channel gain is the same is every time slot and is known. Further, performance improvement is more when the noise variance is higher which is again to be expected.

4.4.2 Slowly varying channel

Next, we conduct experiments with the slowly varying channel model. It is assumed that the channel means are constant during $B = 100$ slots. The optimization horizon is $T = 2000$, that is there are 20 big-slots in each run.

The mean channel gains were first determined for each big-slot. Within a big-slot, the channel gains were then generated using the same method as for the stationary case and given in (4.16). The number of users was again set to 4.

In the first experiment, the mean channel gains are relatively homogeneous with an empirical average of the mean channel gains being $[5.98, 5.55, 4.69, 5.30]$. Figure 4.3a, plots the total utility as a function of the noise level while Fig. 4.3b shows the total transmit power in a slot as a function of time-slot. The data in the latter plot was obtained on a separate run with $B = 50$, $\eta = 1$, and only STO2 and OPT are shown so as to have a more readable figure.

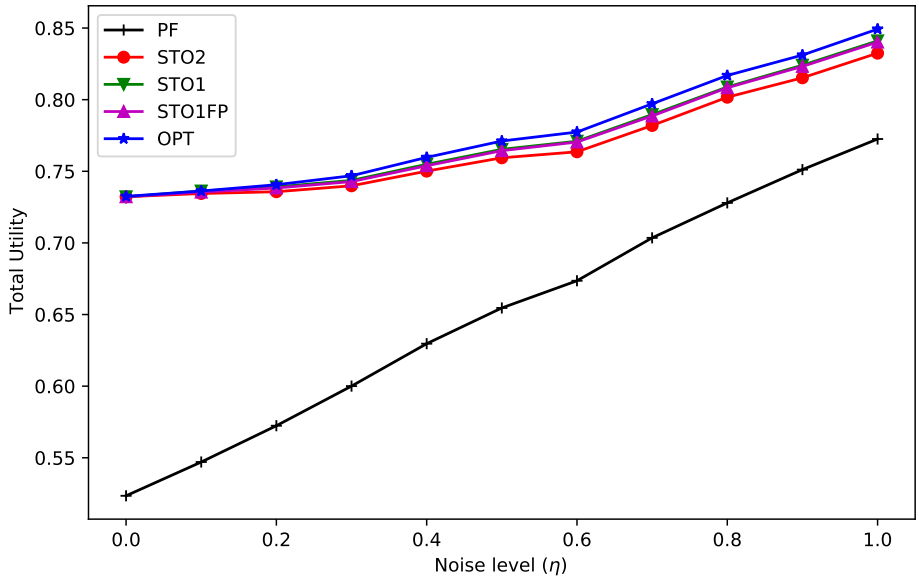
We observe that all algorithms except PF are close to optimal almost throughout. Since the total transmit power in a slot is not far from $\bar{P}$, STO1FP is almost as good as STO1. However, in this scenario prediction is still useful as PF is away from OPT even for $\eta = 0$, that is when there is no noise.

The mean channel gains in the second experiment are widely varying with the empirical average of the mean channel gains being $[2.63, 10.3, 3.76, 0.009]$. One user is in a very bad channel state, whereas another one has much better mean channel gains than the others. The plots for the total utility and total transmit power in a slot are shown in Fig. 4.4. Again, the total power was computed from a separate run.

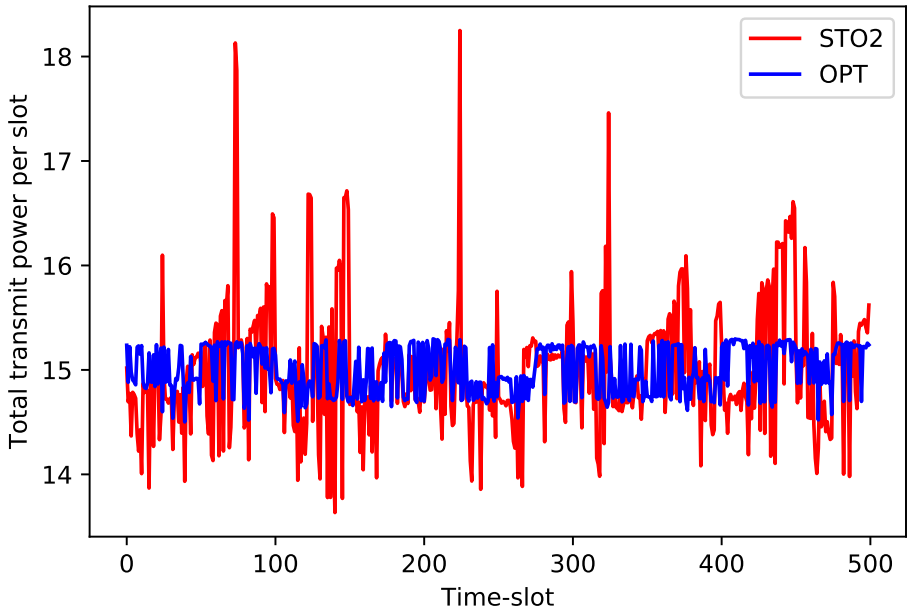
This time we observe that both power control and channel prediction result in improvements. Due to one user being much worse than the others, the optimal transmit power varies much more than in the previous experiment and hits the maximum constraint quite often. Since the fixed-power version is inflexible in this respect, it performs worse for all noise levels.

4.4.3 Mobility Model

Consider a stretch of road of length 1 km covered by one base station, in which vehicles enter from the left and leave on the right (Fig. 4.1). To simplify for illustration, we assume they move with same velocity $v = 25$ m/s, but the algorithms presented in Sec. 4.3 do not depend on this assumption. So they stay in the coverage range of the base station in 40 seconds. New vehicles can enter the network only at the start of the big-slots. The length of a big-slot is set to 1 sec. The probability of a new arrival in a big-slot is set to $p = 0.3$. For each value of noise level (η) we simulate this network for 800 seconds. In this model, the channel gain of mobile user is non-stationary and varies every time slot (1 ms), STO1 and STO2 solves over an horizon of 40 seconds which is the staying time of users in the network. These two aspects

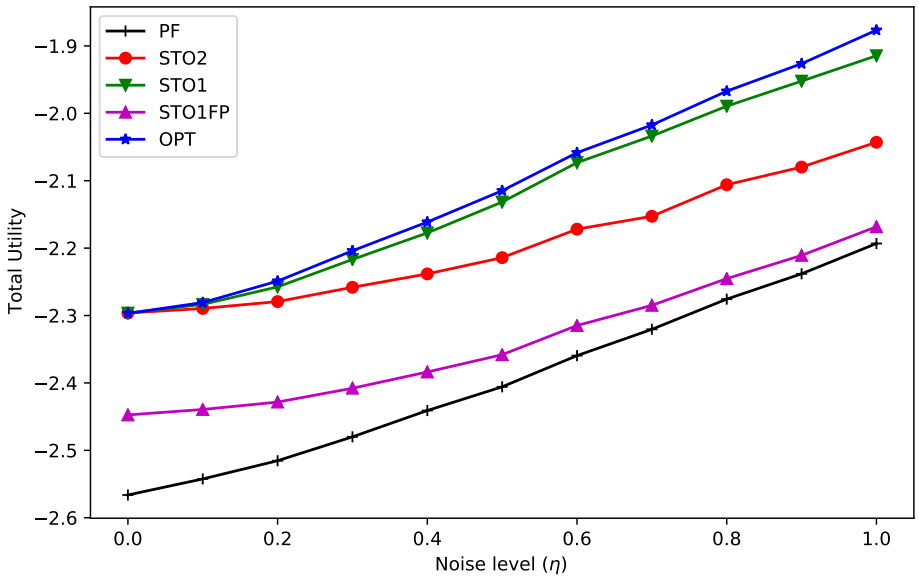


(a) Total utility vs. noise level

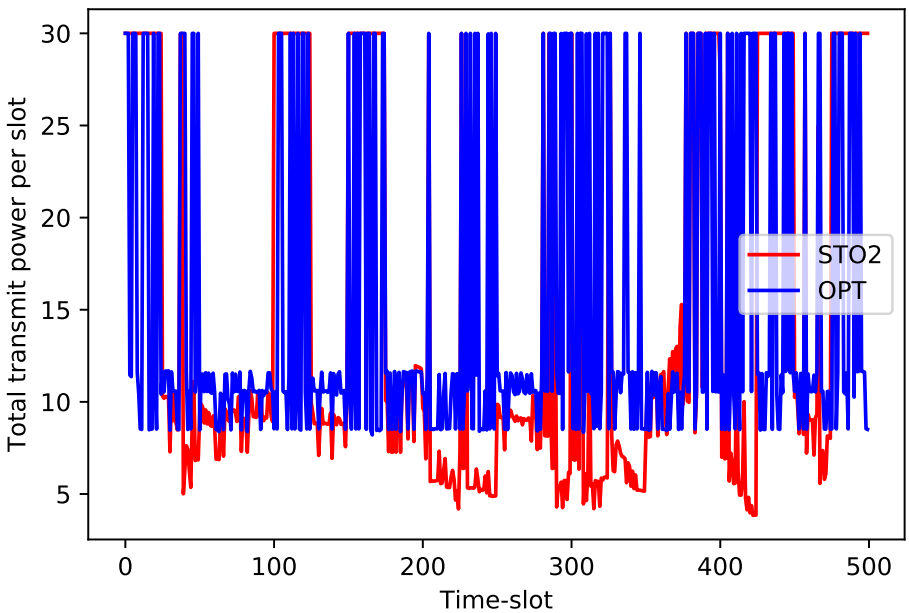


(b) Total allocated power per slot vs. time-slot

Figure 4.3: Slowly varying channel. Log utility function. Experiment 1. $\eta = 1$.



(a) Total utility vs. noise level



(b) Total allocated power per slot vs. time-slot

Figure 4.4: Slowly varying channel. Log utility function. Experiment 2. $\eta = 1$.

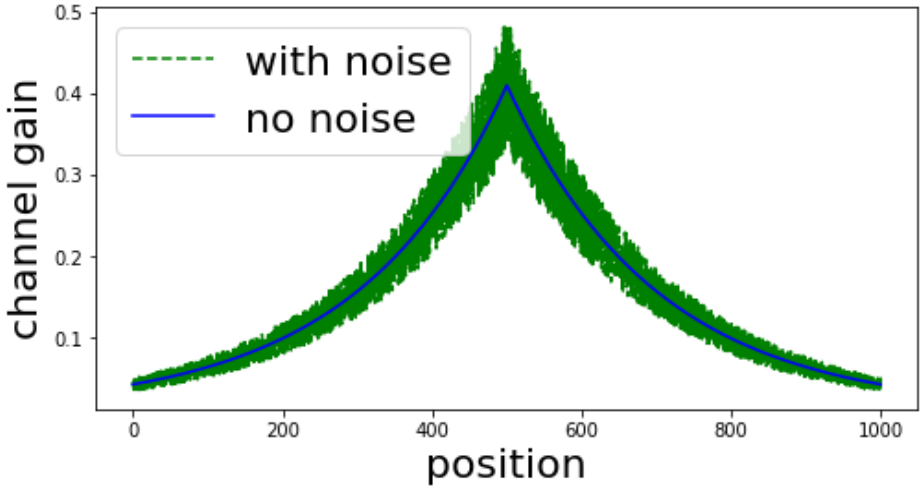


Figure 4.5: Channel gain for two cases: with noise ($\eta = 0.2$) and without noise.

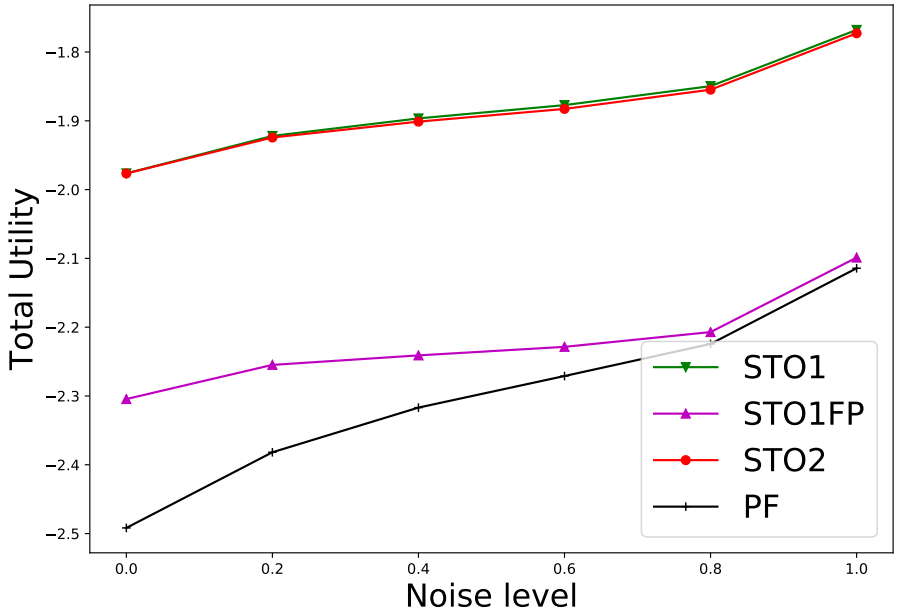
make this model different from the two first models. Since the dimension of (OPT) is very large in this case, we do not compute the optimal solution in the experiments. Also, since STO1 solves a high dimensional problem in every slot compared to STO2, it is much slower when the horizon is large. So, we show only the performance of STO2 and PF.

Fig. 4.5 illustrates the channel gain curve in the noise and no noise cases. Here, the rate is a function of users' position which is in fact a function of the distance to the base station. Let us take the left margin of the road be 0, and the right margin be 1000, then for position $x \in [0, 1000]$ inside the coverage range, the channel gain is equal to $f(x) = \beta(1 + \kappa \exp(|500 - x|/\sigma)$. Here, β, κ, σ are adjustable parameters; in our experiment $\beta = 0.01, \kappa = 40, \sigma = 200$.

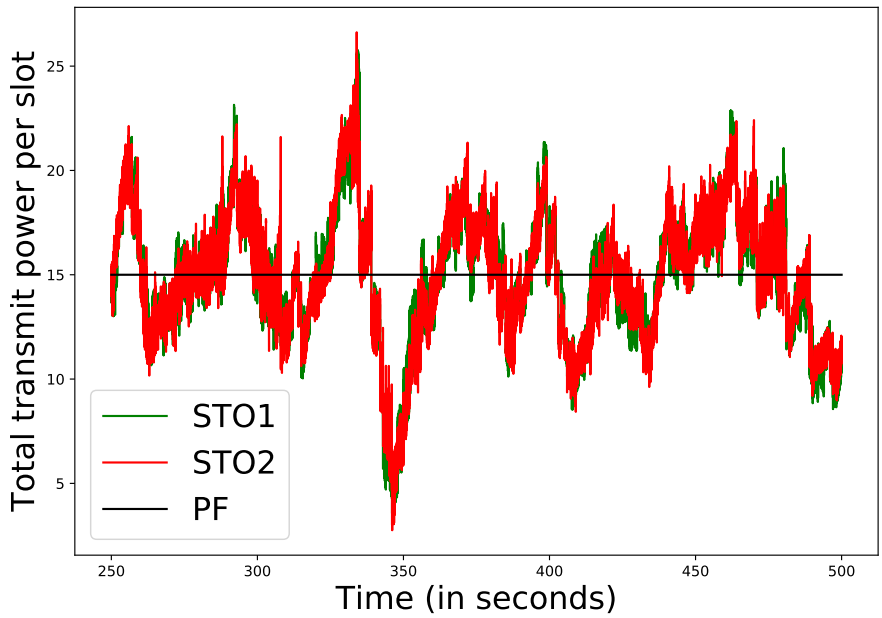
Fig. 4.6 illustrates the numerical results for the above mobile system. The total allocated power is shown only for a small interval of time. Again, we observe that channel prediction leads to a better total utility. Further, by observation, STO2 allocates more power when there are many users (than usual) close to the peak where channel gains are good with high probability.

4.5 Summary

We proposed two heuristics for joint power control and channel allocation that exploit partial information future channel conditions to improve the utility. Even little information such as mean channel gains is sufficient to



(a) Total utility vs. noise level



(b) Total allocated power vs. time (in second). $\eta = 0.2$.

Figure 4.6: Mobility model. Log utility function.

observe improvement compared to when no information is used.

The heuristics are well performed but they are quite heavy since they have to solve an optimization problem frequently (every time-slot for STO1 and every big slot for STO2). In chapter 5 we shall use machine learning based method to learn STO1 to produce an approximate algorithm with less computing time.

Chapter 5

LEARNING THE CHANNEL ALLOCATION ALGORITHM

5.1 Abstract

In Chapter 3 and Chapter 4, we have introduced two heuristic scheduling algorithms that can be applied to general networks. However, in those algorithms, an optimization problem needs to be solved frequently which requires a costly computation and thus may impair real-time processing for large networks. In this chapter, we propose to use Deep Feedforward Neural Networks (DFNN) for learning the relation between inputs and outputs of one of those algorithms. It yields an approximate solution with much less computation time.

5.2 Introduction

Due to the limited availability of resources and to the heterogeneity of user requirements, a proper scheduler plays an important role for an effective sharing of network resources between users and for improving the quality of service. Solving the scheduling problem often amounts to solving a large-scale optimization problem [62], [43]. As has been shown numerically in Chapter 3 and Chapter 4, we can improve the user allocation by taking into account all the available information (that is, past and current information as well as a partial future information) and then solving an optimization problem based on this information. This is what is done by the STO1 algorithm. However, solving large-scale optimization problems so frequently (i.e. every few milliseconds) poses a new challenge: real time processing.

In this chapter, we propose a machine learning-based approach to obtain

an approximate solution to the scheduling problem with much less computation time as compared to the STO1 algorithm. The key idea is to use a function of DFNN form to express approximately the relation between the inputs and the outputs of STO1.

5.2.1 Related works

The theory of approximation for DFNN has been studied in many papers. Motivated by Komogorov's superposition theorem [35] in 1957, many approximation results have proven the approximation capabilities of feed-forward neural networks for the class of continuous functions such as [18],[28],[47]. In his theorem, Komogorov proved that any continuous function can be represented as a superposition of continuous functions of one variable. In [18] (1989), Cybenko proved that any multivariate continuous function with support in a hypercube can be uniformly approximated by a linear finite combinations of compositions of a sigmoidal functions and a set of affine functions. This representation is in fact a feed-forward neural networks with sigmoidal activation function. Independently with the work of Cybenko, Hornik [28] (1989) also proved a similar result. Two years later, Hornik [27] showed that multi-layer feed-forward neural networks with arbitrary bounded and non-constant activation function can approximate arbitrary well real-valued continuous functions on compact subsets of $\mathbf{R}^n$ as long as sufficiently many hidden layers are available. The word "deep" in "deep learning" thus simply means many layers.

Learning an algorithm to produce an approximate algorithm in order to reduce computation time has been proposed in several recent research papers [24], [63]. In [24], the authors consider a Sparse Coding problem which is used for extracting features from raw data. The problem is that Sparse Coding is often too slow for real-time processing in several applications such as pattern recognition. The authors propose a method using a non linear, feed-forward function to learn Sparse Coding to produce an approximate algorithm with 10 times less computation.

Learning an algorithm for wireless resource management by DFNN has been proposed in [63]. In that work, the authors used DFNN to learn an algorithm for the interference channel power control problem. They obtain an almost real time algorithm, since passing the input through a DFNN to get the output only requires a small number of simple operations as compared to an iterative optimization algorithm. They show that, by choosing an appropriate initialization, the initial power control algorithm performs a continuous mapping which can be efficiently learnt.

In this chapter, we use DFNNs for learning a channel allocation algorithm

maximizing the proportional fairness between vehicular users. The proposed method is however potentially applicable to other convex optimization problems.

5.2.2 Contributions

We first propose a machine learning based method for learning the STO1 algorithm introduced in Chapter 3 and Chapter 4. We compare different DFNN architectures and different loss functions to find the most appropriate ones for our problem.

As discussed in Section 5.2.1, any continuous function can be approximated arbitrary well by a DFNN. However, discontinuous functions are much more difficult to learn. Unfortunately, as we show with a simple example, the input-output mapping realized by the STO1 algorithm is discontinuous. This makes the model slower to learn due to oscillations at discontinuity points.

We then characterize these discontinuity points of STO1 by explicitly indicating the set where they reside. In a small dimension, we use this characterization to propose a simple user ordering scheme and a method for choosing the output when the solution is not unique which make the STO1 mapping continuous, and thus hopefully easier to learn. Unfortunately, in larger dimensions, finding an appropriate order requires to solve the original optimization problem which we are trying to avoid solving. We thus propose several heuristic ordering schemes. Although they do no necessarily result in a continuous function, numerical results show that these heuristic orders allow to reduce the learning time. However, these heuristic orders do not always increase accuracy.

Finally, we propose another approach which amounts to learning the dual values, which are proven mathematically to be continuous. Numerical results in section 5.6.5 show that learning dual values is faster and has better performance in comparison with learning the primal values.

5.2.3 Organization

In Section 5.3, we recall the resource allocation problem introduced in Chapter 3 in the case of a single Base Station (BS). We also remind the reader of the STO1 algorithm and state the learning problem we address in this chapter. In Section 5.4, we formally define the input-output relationship for the DFNN model. We discuss about the continuity of this input-output mapping in Section 5.5 and then propose to learn dual values as well as several user ordering schemes for reducing the discontinuity of the original mapping. Numerical results are presented in Section 5.6. We compare the computing

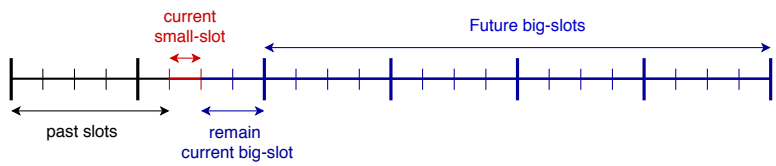


Figure 5.1: The different types of time slots in the STO1 algorithm.

times of the DFNN-based prediction algorithm against those of the original algorithm in Section 5.7 to evaluate the reduction in computing times. In Section 5.8, we discuss another approach for learning the optimal channel allocation which is based on reinforcement learning. Although we did not obtain good results with the latter approach, we believe it is worth presenting it. Finally, in Section 5.9 we discuss several research directions that can be followed in future work.

5.3 Problem Formulation

5.3.1 Optimization problem and the STO1 algorithm

We recall in the following the optimization problem for a single Base Station. The methods proposed in this chapter can be applied for scenarios with multiple BSs, but we shall focus on a single BS since it is faster to produce numerical results. We consider the relaxed form of the channel allocation problem [51], which we recall below:

$$\begin{cases} \text{maximize} & O(\alpha) = \sum_{i=1}^{K} \log\left(\sum_{j=1}^{T} \alpha_{ij} r_{ij}\right) \\ \text{subject to} & \\ & \sum_{i=1}^{K} \alpha_{ij} = 1, \quad j = 1, \ldots, T, \\ & \alpha_{ij} \geq 0, \quad j = 1, \ldots, T, \ i = 1, \ldots, K. \end{cases} \tag{I}$$

Here α can be non integer.

Remark 5.3.1 (Joint power control and channel allocation)**.** *For brevity, we give the problem formulation only for the channel allocation problem treated in Chapter 3. A similar formulation can be done for the joint power control and channel allocation problem studied in Chapter 4. In the rest of the chapter, a remark shall be made wherever the treatment of the two problems differs in order to highlight their differences.*

Let Δ be the size of the big-slot in absolute time units and let $m = \Delta/\delta$ be the number of small slots in a big-slot (see Figure 5.1). Denote by $\bar{r}_{ij}$ the mean rate in slot j for user i (we shall assume it is a function of distance from the user to the BS). At each small-slot t, with a slight abuse of notation, we shall denote by $\bar{\rho}_{i,0} = \sum_{j=t+1}^{(m-(t \mod m))+t} \bar{r}_{ij}$ the total rate for user i in the remaining channel allocation slots of the current big-slot $\tau = 0$, where t mod m denotes the remainder when dividing t by m. We also define $\bar{\alpha}_{i,0}$ as the corresponding allocation for the current big-slot $\tau = 0$.

Denote by $\bar{\rho}_{i\tau} = \sum_{j=(\tau-1)m+A+1}^{\tau m+A} \bar{r}_{ij}$ where $A = \left(\left\lfloor \frac{t}{m} \right\rfloor + 1\right) m$, is the total average data rate that user i will get in the future big-slot τ ($\tau = 1, 2, ..., J-1$), where big slot τ starts after the current big-slot, and J is the short time horizon in term of big slots over which we can estimate the mean future rate. We also define $\bar{\alpha}_{i\tau}$ as be the corresponding allocation for user i in future big slot τ. These allocations $\bar{\alpha}_{i\tau}$ can be interpreted as the fraction of small slots that user i will be allocated in the big-slot τ.

Note that this definition is slightly different from definition in chapter 3. The differences are as follows: in chapter 3, there is no current big slot and the future big-slot starts just after the current small slots, but it does not change much. The above definition of two time slot types corresponds in fact to the ones introduced in Chapter 4.

Denote by $a_i(t) = \sum_{j=1}^{t} \alpha_{ij} r_{ij}$ the total throughput allocated to user i up to time slot t, and let $K(t)$ be the number of users inside the coverage range of the BS at time t.

The algorithm STO1 contains two steps which are as follows:

- **Step 1**– solve the following optimization problem over a short-term horizon of J big-slots:

$$\begin{cases} \text{maximize} & \displaystyle\sum_{i=1}^{K(t)} \log\left(a_i(t-1) + \alpha_{it} r_{it} + \sum_{\tau=1}^{J} \bar{\alpha}_{i\tau}\bar{\rho}_{i\tau}\right) \\ \text{subject to} & \\ & \sum_{i=1}^{K(t)} \alpha_{it} = 1, \\ & \sum_{i=1}^{K(t)} \bar{\alpha}_{i\tau} = 1, \quad \tau = 0, \ldots, J-1, \\ & \alpha_{it}, \bar{\alpha}_{i\tau} \in [0,1],, \quad \tau = 0, \ldots, J-1, \ i = 1, \ldots, K(t). \end{cases} \tag{STO1-Opt}$$

  The decision variables in Problem (STO1-Opt) are the channel allocations in the current small slot, α_{it}, and the channel allocations in the current and future big-slots, $\bar{\alpha}_{i\tau}$. Since the future allocations are only computed on the time-scale of big-slots, there is a reduction by factor m in the number of variables in (STO1-Opt).

Figure 5.2: The Carmes borough in Toulouse, with one BS (Free Mobile type LTE1800). The actual size is $200m \times 400m$.

- **Step 2** – allocate the channel to the users with the fraction α_{it} of computed allocation in the current small-slot t.

Here in step 2, we consider the relaxed solution which is different from STO1 in Chapter 3 (which rounds the relaxed solution to obtain an integer solution) since we want to use the properties of convex optimization problems to investigate the continuity of the solution. After learning the optimal (fractional) allocation, we will come back to an integer solution by allocating the channel to the user with the largest fraction α_{it}. Actually, we observed that the solutions of (I) and (STO1-Opt) are integer with high probability. Figure 5.3 shows STO1 with integer version and STO1 with relaxed version for a scenario generated with SUMO using the map shown in Figure 5.2 with 244 users in around 61.7 minutes (the other parameters are identical to those used in Section 5.6.1). We can notice that the behaviors are not much different.

Remark 5.3.2. (STO1-Opt) *is solved thanks to the python package CVXPY [20] and the solver MOSEK. In Chapter 3, we solve this optimization problem using a projected gradient algorithm, since this method allows to approximate solution of a convex optimization problem by iterations when the feasible set is of simplex form (i.e, a simplex or a Cartesian product of simplex), no matter how complex the objective function is as long as it is smooth and convex. But here we employ CVXPY [20] since it can be used to generalize this idea to other convex optimization problems, with more complex constraints such as power control constraints or others QoS constraints (e.g. delay constraints).*

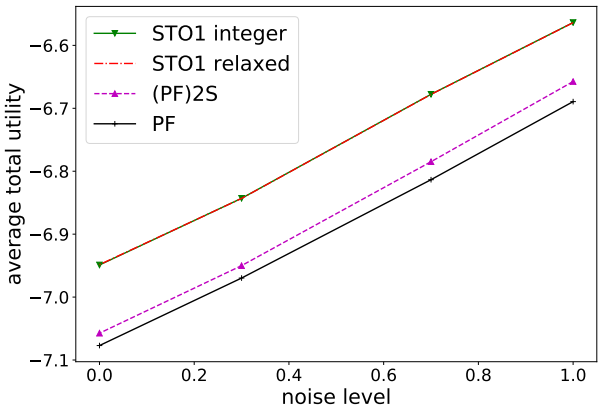


Figure 5.3: Comparison of integer and relaxed versions of STO1.

5.3.2 Learning STO1 with DFNN

As presented in Chapter 3 and Chapter 4, STO1 and STO2 are better than the other existing algorithms. However they have to solve an optimization problem with a large number of variables and constraints frequently (every small slot for STO1 or every big slot for STO2), so even if their performance is good, their computations are heavy. When the system is large and requires many more QoS contraints, they may not be able to run in real time. In addition, even when they are able to run in real time, it is good to reduce the computation time without reducing too much the quality of the allocation.

Therefore in this chapter, we want to learn the STO1 algorithm using Deep Feedforward Neural Networks to obtain a new algorithm that behaves like STO1 but with a significantly reduced computation time. In other words, we want to learn the input-output relationship of STO1, by approximating the input-output mapping of STO1 with a DFNN. Here we focus only on STO1, but a similar approach is expected to work for STO2. After getting the approximation function (that is, the DFNN), the output can be computed by feeding the DFNN with the input value, instead of solving an optimization problem. This simpler method is expected to work faster than the original algorithm.

Obviously, the same idea could be used for other problems in order to obtain an approximate method which performs almost as well as the original algorithm but requires much less computing time. In short, the approach is as follows:

1. Design a well-performing algorithm based on the best available infor-

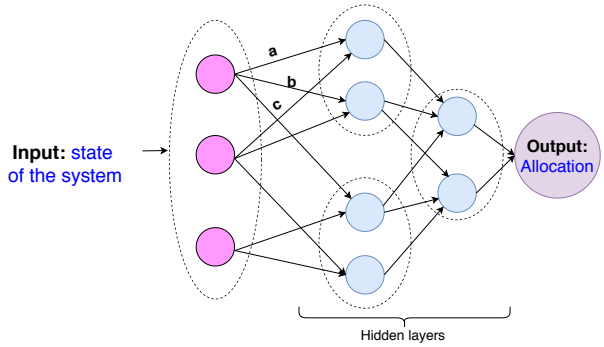


Figure 5.4: An example of Deep Feed-forward Neural Network.

mation;

2. Learn it by an approximate algorithm which behaves closely to the original one and has less computation.

The background on supervised learning with DFNN was presented in Section 2.5.2. The basic idea can be summarized as follows. The STO1 algorithm can be seen as a function F that maps an input $X \in \mathcal{X}$ (a problem instance) to an output $Y \in \mathcal{Y}$ (a channel allocation), where $\mathcal{X}$ and $\mathcal{Y}$ denote the input and output spaces of STO1, respectively. Unfortunately, STO1 is too complicated to get the exact formula of F. Therefore here we want to approximate it by another function $\hat{F} : \mathcal{X} \to \mathcal{Y}$ which is in the form of a DFNN (an example of DFNN is illustrated in Figure. 5.4).

A DFNN is composed of many linear functions (sum of matrix multiplications and bias vectors) and non-linear functions (relu, sigmoid, softmax, etc [76]), and inside linear functions there are many parameters. So finding a good DFNN function means finding a good architecture, that is: the way the linear and non-linear functions are combined, the linear and non-linear functions in each layer, the size (number of units in each hidden layer) and then their parameters. Finding a good architecture is in general not an easy task [64]. In this chapter, we shall empirically compare some architectures through experiments presented in Section 5.6.2. After fixing the architecture, we have to find appropriate parameters by minimizing the empirical risk as described in Section 2.5.2.

5.4 System Setup for learning

Recall that, in STO1, we have two types of time scales: one is the big slot Δ and the other one is the small slot δ. In STO1, we solve problem (STO1-Opt) with variables of size $(1+J)*K$ every small slot, where J is the time horizon in terms of big slots and K is the number of users in the system. The size of the allocation vector for each user is equal to $1+J$ since it contains the allocation for the current small slot, α_{it}, and the average allocation $\bar{\alpha}_{i\tau}$ for the subsequent J big slots (including the current big slot). The input of STO1 is a data rate vector of size $(1+J)*K$ and the total allocated throughput for the K users. The output of STO1 is the current allocation vector $(\alpha_{it})_{i=1,\dots,K}$ which is of size K, since we shall only use current allocation for making decision.

As it is defined at the moment, STO1 is not well suited to be modelled as a learning problem for the two reasons stated below.

Firstly, since K can vary over time, the dimension of the input vector will also vary. To circumvent this problem and to properly define STO1 as a function, we have to fix the size of the state. To do that, we extend the real state of the system by adding some pseudo users. Let us assume that there are at most K_M users inside the system. We will then add $K_M - K$ pseudo users, where K is the number of real users in the system at time t. We will actually learn an extended version of STO1 which is STO1 when we restrict it to K users. There are many ways to extend STO1, but here we try to define an extended version that preserves as much as possible the continuity of STO1. When we mention "learning STO1", it means "learning the extended function" of STO1.

Secondly, the output of STO1 as defined above is the solution of an optimization problem. So in fact STO1 is a set-valued mapping since the solution need not be unique. But by using the CVXPY package to solve the convex optimization problem (STO1-Opt), we agree with the way it determines one of the solutions. This makes STO1 becomes a function (instead of set-valued mapping). However, the way the solution is chosen when it is not unique can make the function become continuous or discontinuous. In [63], the authors show that a mapping can be made continuous by choosing an appropriate initial point for the algorithm. We shall discuss the continuity of the STO1 function and how to choose the solution when it is not unique in order to obtain a continuous mapping in a small dimension in Section 5.5.

Remark 5.4.1 (Joint power control and channel allocation)**.** *For the joint power control and channel allocation problem, the state needs to be augmented by the remaining total power. The output of the DFNN will now give the*

transmit power to each user as well as the fraction of the channel it gets allocated.

5.4.1 State

We define a state as a matrix of size $(2+J)\times K_M$, where K_M is the maximum number of users in the system. There are thus $2+J$ rows, and each row has K_M elements. The interpretation is as follows:

- The first row gives the current rates of the K users. We fill in the K positions on the left hand side with these current rates, and the remaining $K_M - K$ positions are filled with -1.
- The next J rows (from $2, .., J+1$) give the average rates of the users in the next J big slots. For pseudo users (the $K_M - K$ columns on the right hand side), we use the value $(-1)\cdot(\Delta/\delta - (t \mod \Delta/\delta))$ for the current big slot and the value $(-1)\cdot\Delta/\delta$ for the other big slots.
- The last row gives the total allocated throughput of the K users. For pseudo users, we use a large enough value which is significantly greater than the total allocated throughput of real users.

By observing how STO1 works, we remark, as expected, that STO1 gives priority to users with a low allocated throughput and a high current rate. Therefore, the way we define the state (that is, by using by negative numbers for the current and future rates of pseudo users, and extremely large values for their allocated throughput) is intended to help the model ignore quickly the pseudo users.

Remark 5.4.2. *Remark that the real K users in the system at present time. Therefore, the K places of the real users in the first row which give the current rates of those users have to be strictly positive. The future rates (from the second row to the $(J+1)$-th row) can be zero.*

5.4.2 Target

We remind the reader that we want to learn only the current allocation, not the future allocation. Therefore, the target will be a vector of size K_M, where the first K positions represent the fractional allocation α_{it} of the K users as computed by STO1, and the last positions are filled with zero. Since in the optimization problem, the sum of allocation should be equal to 1, when there is no user in the system (all positions correspond to pseudo users), the allocation vector will be set to $(1/K_M, ..., 1/K_M)$ by convention.

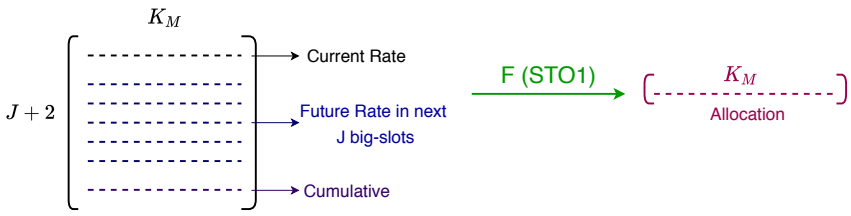


Figure 5.5: Input and output of the DFNN model.

Figure 5.6: Two connected components of $E_{i,1} \forall i$.

Figure 5.5 illustrates the input and output of the DFNN model as described above.

Remark 5.4.3. *From now on, we shall denote by F the input-output mapping realized by STO1 when its output is restricted to the current allocation vector $(\alpha_{it})_{i=1,\ldots,K}$. One could of course learn the full allocation matrix (that is, including future allocations $\bar{\alpha}_{i\tau}$ over the big slots), in which case the output would be of size $(J+1) * K_M$ instead of size K_M. By doing that, the output provides more information than the way it was defined above. However, as STO1 is not a continuous function as we shall shortly discuss, the definition proposed here eases the characterization of discontinuity points.*

5.5 Discussion about the continuity of the STO1 function

As defined above, F is a mapping from $\prod_{i=1,j=1}^{i=K_M,j=J+2} E_{i,j}$ to $\mathcal{S}_{K_M}$, where $\mathcal{S}_{K_M}$ is the simplex of size K_M and

- For $j = 1$ and $i = 1, \ldots, K_M$, $E_{i,1} = (0, +\infty] \cup \{-1\}$,
- For $j = 2, \ldots, J+1$ and $i = 1, \ldots, K_M$, $E_{i,j} = [0, +\infty] \cup \{-1, -2, ..., -\Delta/\delta\}$,
- For $j = J+2$ and $i = 1, \ldots, K_M$, $E_{i,J+2} = [0, +\infty]$.

We can see that the pseudo users and real users lie in different connected components (see Figure 5.6), from which it follows that the continuity of its extension depends only on STO1. We shall thus concentrate on the continuity

of the STO1 function on the connected component corresponding to the K real users. Unfortunately, STO1 itself is not a continuous function. Let us take a simple example illustrating the discontinuity. Assume that $K = 2$, $J = 1$, $\Delta = \delta$ and consider the following sequence of states:

$$R_n = \begin{pmatrix} r + \left(\frac{1}{n}\right) \cdot (-1)^n & r - \left(\frac{1}{n}\right) \cdot (-1)^n \\ r - \left(\frac{1}{n}\right) \cdot (-1)^n & r + \left(\frac{1}{n}\right) \cdot (-1)^n \\ 0 & 0 \end{pmatrix}.$$

Then it is easy to show that $R_n \to \bar{R}$, where

$$\bar{R} = \begin{pmatrix} r & r \\ r & r \\ 0 & 0 \end{pmatrix}.$$

However

$$F(R_n) = \begin{cases} [1, 0] \text{ if } n = 2k, \\ [0, 1] \text{ if } n = 2k + 1. \end{cases}$$

This simple example shows that the STO1 function is not continuous. The discontinuities make the learning problem much more difficult since near discontinuity points, the DFNN model does not know which direction of the output its parameters should follow. For example, as illustrated in the above 2-dimension example, near $\bar{R}$, the target (output) oscillates between $[0, 1]$ and $[1, 0]$. Since a DFNN-based function is a composition of many continuous functions, it is also a continuous function. Trying to fit every target near $\bar{R}$ into the DFNN model can make its parameters conflict and result in a slower convergence. Therefore in the remainder this section we shall characterize the discontinuity points of the STO1 function and discuss how to derive a continuous function from STO1.

Above we showed one discontinuity point ($\bar{R}$) when $K = 2, J = 1$. Now, our objective is to characterize the set containing all discontinuity points. Before doing that, let us analyze the property of an optimal solution resulting from the KKT conditions (which are necessary and sufficient conditions in our case as presented in section 2.5.1). To simplify the notation, let us assume that a big slot is equal to a small slot, that is, $\Delta = \delta$ (the proof is however still valid if a big slot is equal to multiple small slots). With $\Delta = \delta$, we can write (STO1-Opt) as follows:

$$\begin{cases} \text{maximize} \sum_{i=1}^{K(t)} \log \left(\sum_{j=t}^{t+J} \alpha_{ij} r_{ij} + c_i \right) \\ \sum_{i=1}^{K(t)} \alpha_{ij} = 1, \quad j = t, t+1, ..., t+J, \\ \alpha_{ij} \in [0, 1], \quad j = t, t+1, ..., t+J, \ i = 1, \ldots, K(t). \end{cases} \tag{I$_R$}$$

Let us denote by r_{ij} the rate of user i at time j, and let c_i be the total throughput allocated to user i up to present time. Consider the state R defined by

$$R = \begin{pmatrix} r_{1t} & r_{2t} & \cdots & r_{K_M t} \\ \cdots & \cdots & \cdots & \cdots \\ r_{1,t+J} & r_{2,t+J} & \cdots & r_{K_M,t+J} \\ c_1 & c_2 & \cdots & c_{K_M} \end{pmatrix},$$

in which the ith column provides all the information available for user i. We denote the above optimization problem by (I_R) to emphasize that it depends on R.

The Lagrange function of this convex optimization problem is $L = O(\alpha) + \sum_j \lambda_j (1 - \sum_i \alpha_{ij}) + \sum_{i,j} \rho_{i,j}\alpha_{ij}$ where $O(\alpha) = \sum_{i=1}^{K(t)} \log\left(\sum_{j=t}^{t+J} \alpha_{ij} r_{ij} + c_i\right)$. The KKT conditions for problem (I_R) are as follows:

$$\begin{cases} 1, \text{ Primal feasibility: } \sum_i \alpha^*_{ij} = 1 \quad \forall j, \text{ and } \alpha^*_{ij} \geq 0 \quad \forall i, j, \\ 2, \text{ Dual feasibility: } \rho^*_{ij} \geq 0 \quad \forall i, j, \\ 3, \text{ Complementary slackness: } \alpha^*_{ij}\rho^*_{ij} = 0 \quad \forall i, j, \\ 4, \text{ Lagrange stationary: } \frac{r_{i,j}}{c_i + \sum_k \alpha^*_{ik} r_{ik}} = \lambda^*_j - \rho^*_{ij} \quad \forall i, j. \end{cases} \tag{KKT}$$

The last condition is implied by $\nabla_\alpha L(\alpha^*) = 0$.

From (KKT), it follows that if $\rho^*_{ij} > 0$, then $\alpha^*_{ij} = 0$ and $\frac{r_{i,j}}{c_i + \sum_k \alpha^*_{ik} r_{ik}} = \lambda^*_j - \rho^*_{ij} < \lambda^*_j$. It implies that α^*_{ij} is positive only when $\rho^*_{ij} = 0$ and in this case the ratio $\frac{r_{i,j}}{c_i + \sum_k \alpha^*_{ik} r_{ik}} = \lambda^*_j$ represents the maximum ratio one user i can get in slot j. Since $\sum_i \alpha^*_{ij} = 1$, there exists at least one user who gets a strictly positive allocation in time slot j, i.e, there exists at least one user i for which $\rho^*_{ij} = 0$. The output that we consider is only the current allocation in time slot t, therefore we shall discuss only the continuity of the allocation at time t. Let denote by $I_t(R)$ the set of users who are able to have a strictly positive allocation in current slot, i.e, $I_t(R) = \{i \mid \rho^*_{it}(R) = 0\}$. By defining $I_t(R)$, we implicitly claim that ρ^*_{it} is defined uniquely by R and we shall prove later this uniqueness property. Consider the following sets

$$\mathcal{A} = \{R | \# I_t(R) = 1\}, \mathcal{B} = \{R | \# I_t(R) \geq 2\}.$$

Then $\mathcal{A}$ and $\mathcal{B}$ represent a partition of the whole input space since $\# I_t(R) \geq 1$ for all states R as explained above. The set $\mathcal{A}$ is the set of problem instances for which there is exactly only one user having a strictly positive allocation at time t, that user thus get full allocation and others get nothing. The set $\mathcal{B}$ is the complement. Proposition 5.5.1 below states two fundamental properties of the set $\mathcal{A}$.

Proposition 5.5.1.

- *On $\mathcal{A}$, $\alpha^*_{\mathbf{t}} = (\alpha^*_{it})_{i=1}^K$ is uniquely defined by R.*
- *F: $R \mapsto \alpha^*_{\mathbf{t}} = (\alpha^*_{it})_{i=1}^K$ is continuous on $\mathcal{A}$, i.e, all potential discontinuities lie in $\mathcal{B}$.*

Remark 5.5.2. *Note that we do not claim that all points in the set $\mathcal{B}$ are points of dicontinuity. We just claim that any discontinuity point necessarily belongs to the set $\mathcal{B}$.*

Before proving Proposition 5.5.1, we need a lemma, in which we shall consider the following equivalent problem:

$$\begin{cases} \text{maximize} \displaystyle\sum_{i=1}^{K(t)} \log(C_i) \\ \mathbf{C} \in \mathcal{C} \end{cases}, \tag{II$_R$}$$

where

$$\mathcal{C} = \{\mathbf{C} = (C_1, \ldots, C_K) \mid C_i = \sum_{j=t}^{t+J} \alpha_{ij} r_{ij} + c_i \,\forall i, \sum_i \alpha_{ij} = 1 \,\forall j, \alpha_{ij} \geq 0 \,\forall i, j\}.$$

Then (II$_R$) is equivalent to (I$_R$), and $\mathcal{C}$ is a convex set. Let $\mathbf{C}^*$ be the optimal solution of (II$_R$).

Lemma 5.5.3. *The optimal solution $\mathbf{C}^*$ is uniquely defined by R and continuous in R.*

Proof. Indeed since (II$_R$) is a convex optimization problem, we have for any $\mathbf{C} \in \mathcal{C}$:

$$\nabla O_{\mathbf{C}}(\mathbf{C}^*)(\mathbf{C} - \mathbf{C}^*)^T \leq 0. \tag{5.1}$$

where $O(\mathbf{C}) = \sum_i \log(C_i)$. Let us take R' close to R, with $|R' - R|_\infty = \epsilon$ and assume that $\mathbf{C}'^*$ is solution corresponding to the matrix R'. By definition of the set $\mathcal{C}$, there exists α^* and α'^* such that:

$$\begin{aligned} C_i^* &= \sum_{j=t}^{t+J} \alpha_{ij}^* r_{ij} + c_i, \forall i, \\ C_i'^* &= \sum_{j=t}^{t+J} \alpha_{ij}'^* r_{ij}' + c_i', \forall i. \end{aligned} \tag{5.2}$$

It is obvious that $C_i^*, C_i'^*$ are strictly positive $\forall i$. From (5.1) we obtain

$$\sum_{i=1}^{K} \frac{C_i}{C_i^*} \leq K \quad \forall \mathbf{C} = (C_1, C_2, ...C_K) \in \mathcal{C}. \tag{5.3}$$

It follows that

$$\begin{aligned}
\left|\sum_{i=1}^{K} \frac{C_i'^*}{C_i^*}\right| &= \left|\sum_{i=1}^{K} \frac{\sum_{j=t}^{t+J} \alpha_{ij}'^* r_{ij}' + c_i'}{\sum_{j=t}^{t+J} \alpha_{ij}^* r_{ij} + c_i}\right| \\
&= \left|\sum_{i=1}^{K} \frac{\left(\sum_j \alpha_{ij}'^* r_{ij} + c_i\right) + \left(\sum_{j=t}^{t+J} \alpha_{ij}'^* (r_{ij}' - r_{ij}) + (c_i' - c_i)\right)}{\sum_{j=t}^{t+J} \alpha_{ij}^* r_{ij} + c_i}\right| \\
&\leq \left|\sum_{i=1}^{K} \frac{C_i}{C_i^*}\right| + \left|\sum_{i=1}^{K} \frac{\left(\sum_{j=t}^{t+J} \alpha_{ij}'^* (r_{ij}' - r_{ij}) + (c_i' - c_i)\right)}{\sum_{j=t}^{t+J} \alpha_{ij}^* r_{ij} + c_i}\right| \\
&\leq \left|\sum_{i=1}^{K} \frac{C_i}{C_i^*}\right| + A_1\epsilon \\
&\leq K + A_1\epsilon,
\end{aligned} \tag{5.4}$$

where $C_i = \sum_j \alpha_{ij}'^* r_{ij} + c_i$ for all i and $A_1 = \sum_{i=1}^{K} \frac{(J+1)+1}{C_i^*}$ is a positive number which does not depend on ϵ.

The fourth implication follows from $|R' - R| = \epsilon$ and $0 \leq \alpha_{ij}'^* \leq 1$, whereas the last implication follows from (5.3) and the fact that $\mathbf{C} = (C_i)_i \in \mathcal{C}$. So from (5.4) we get $\sum_{i=1}^{K} \frac{C_i'^*}{C_i^*} \leq K + A_1\epsilon$. Similarly, we have $\sum_{i=1}^{K} \frac{C_i^*}{C_i'^*} \leq K + A_2\epsilon$, where A_2 is also a positive number not depending on ϵ. It implies that $\sum_{i=1}^{K} \left(\frac{C_i^*}{C_i'^*} + \frac{C_i'^*}{C_i^*}\right) \leq 2K + (A_1 + A_2)\epsilon$. Combining this with the fact that for each i we have $\left(\frac{C_i^*}{C_i'^*} + \frac{C_i'^*}{C_i^*}\right) \geq 2$ (this is implied by the AM–GM inequality which claims that $a + b \geq \sqrt{ab}$ for all $a \geq 0$ and $b \geq 0$), we can derive $\left(\frac{C_i^*}{C_i'^*} + \frac{C_i'^*}{C_i^*}\right) \leq 2 + (A_1 + A_2)\epsilon$ for all i. Defining by $t_i = \frac{C_i^*}{C_i'^*}$ we have $t_i > 0$ and $t_i + 1/t_i \leq 2 + (A_1 + A_2)\epsilon$. This is a quadratic inequality, and solving it we get

$$1 + D\epsilon - \sqrt{(D\epsilon)^2 + 2D\epsilon} \leq t_i \leq 1 + D\epsilon + \sqrt{(D\epsilon)^2 + 2D\epsilon}, \tag{5.5}$$

for all i, where $D = \frac{A_1 + A_2}{2} > 0$ does not depend on ϵ.

From (5.4) and (5.5) we have that:

- From (5.4), if we take $R' = R$, the last inequality becomes $\sum_{i=1}^{K} \frac{C_i'^*}{C_i^*} \leq K$, and also $\sum_{i=1}^{K} \frac{C_i^*}{C_i'^*} \leq K$. Similarly to above implications, this implies

$t_i + 1/t_i = 2$ for all i. It implies that $t_i = 1$ for all i, i.e, $C_i^* = C_i'^*$ for all i, which shows the uniqueness of the solution of (II_R).

- From (5.5), we obtain that

$$\lim_{\epsilon \to 0} \frac{C_i'^*}{C_i^*} = 1 \quad \forall i, \tag{5.6}$$

i.e $\lim_{\epsilon \to 0} C_i'^* = C_i$ for all i, which proves the continuity of $\mathbf{C}^*$ in R.

□

Proof. (proof of proposition 5.5.1) Recall that $\mathcal{A} = \{R | \# I_t(R) = 1\}$, i.e, given $R \in \mathcal{A}$, there exists an unique i_0 such that $\rho_{i_0 t} = 0$ and for every $i \neq i_0$ we have $\rho_{it} > 0$. It yields

$$\frac{r_{i_0 t}}{C_{i_0}^*} = \lambda_t^* > \lambda_t^* - \rho_{it}^* = \frac{r_{i_t}}{C_i^*}, \quad \forall i \neq i_0, \tag{5.7}$$

and $\alpha_{i_0 t}^* = 1$, whereas $\alpha_{it}^* = 0$ for all $i \neq i_0$. This means that the allocation $\alpha_{\mathbf{t}}^*$ is unique for every input state R in $\mathcal{A}$.

Consider $R \in \mathcal{A}$ and R' close to R: $|R' - R|_\infty = \epsilon$. Since we consider the continuity of the current allocation only, our objective is to prove that $\alpha_{\mathbf{t}}'^* = (\alpha_{it}'^*)_i$ is close to $\alpha_{\mathbf{t}}^* = (\alpha_{it}^*)_i$. In fact we have that $\alpha_{\mathbf{t}}'^* = \alpha_{\mathbf{t}}^*$ if ϵ is small enough. Indeed, from claim (5.5) in Lemma 5.5.3, we have $\frac{C_i'^*}{C_i^*} = 1 + \mathcal{O}(\sqrt{\epsilon})$ $(0 < \lim_{\epsilon \to 0} \frac{\mathcal{O}(\sqrt{\epsilon})}{\epsilon} < +\infty)$, where $(C_i^*)_i$ and $(C_i'^*)_i$ are the solutions of (II_R) corresponding to inputs R and R', respectively. Since $R \in \mathcal{A}$, there exists i_0 such that $\frac{r_{i_0 t}}{C_{i_0}^*} > \frac{r_{i_t}}{C_i^*}$ for all $i \neq i_0$. It implies that $\frac{r'_{i_0 t}}{C_{i_0}'^*} > \frac{r'_{i_t}}{C_i'^*}$ for all $i \neq i_0$ if ϵ is small enough. In turn, this implies that $\alpha_{i_0 t}'^* = 1$ and $\alpha_{it}'^* = 0$ for all $i \neq i_0$, that is, $\alpha_{\mathbf{t}}^* = \alpha_{\mathbf{t}}'^*$. □

Proposition 5.5.1 shows that for all problem instances $R \in \mathcal{A}$, there exists a unique allocation α_t, and moreover that this allocation is continuous in R. However, the allocation is not necessarily unique for problem instances $R \in \mathcal{B}$, and it may even be discontinuous in this set, as shown with our simple example. Interestingly, it turns out that the dual variables of problem of (I_R) are continuous in R, as proved in Corollary 5.5.4, which directly follows from Lemma 5.5.3.

Corollary 5.5.4. *The optimal solution of the dual problem is such that:*

1. $\lambda^* = (\lambda_t, \lambda_{t+1}, ..., \lambda_{t+J})$ *is uniquely defined by* R *and continuous in* R.
2. $\rho^* = (\rho_{ij}^*)_{i=1,j=t}^{i=K,j=t+J}$ *is uniquely defined by* R *and continuous in* R.

Proof. We first prove the first assertion. From the third condition of (KKT), for each time slot j, there exists at least one i_0 such that $\rho^*_{i_0 j} = 0$. Combining with the fourth condition of (KKT) we obtain that $\lambda^*_j = \max_i \frac{r_{ij}}{C^*_i}$, that is, the maximum is attained because of the existence of i_0. Since $\mathbf{C}^*$ is unique and continuous in R, it implies the uniqueness and continuity of λ^*.

Regarding the second assertion, it follows from the fourth condition of (KKT) that

$$\rho^*_{ij} = \lambda^*_j - \frac{r_{ij}}{C^*_i}. \tag{5.8}$$

Since $\mathbf{C}^*$ and λ^* are unique and continuous in R, it implies the uniqueness and continuity of ρ^*. □

5.5.1 Learning with the dual value

As shown above, the solution $\alpha^*_{\mathbf{t}}$ of (I_R) is not continuous in the input R, but the dual variables λ^* and ρ^* are. We thus propose the following idea. Instead of learning the allocation vector $\alpha^*_{\mathbf{t}}$, we could learn the dual variable ρ^*. As mentioned above, the users who get a positive fraction of allocation $\alpha_{i,t} > 0$ all belong to the set $I_t(R)$. Therefore, after learning the dual values (ρ^*_{ij}), we propose to choose the current allocation as follows: $\alpha^*_{i^*(t),t} = 1$ and $\alpha^*_{i,t} = 0$ for all $i \neq i^*(t)$, where $i^*(t)$ is an index in $I_t(R)$. If there are more than one index in $I_t(R)$ we choose arbitrarily an index in that set. This is equivalent to first learn $\mathbf{C}^* = (C^*_1, C^*_2, ..., C^*_{K_M})$ and then choose arbitrarily an index in the set argmax $\frac{r_{it}}{C^*_i}$.

5.5.2 Example in a small dimension

As stated in Proposition 5.5.1, F is continuous on $\mathcal{A}$, so the set of all discontinuities is a subset of $\mathcal{B}$. It is difficult to provide further information on this set for the general case. Therefore, we shall consider in this section a small 2-dimension example for which the analysis is easier, in order to understand better the behaviour of the output of F and how to make it continuous.

We restrict ourselves to the case $K = 2$, $J = 1$ and $t = 1$ and we further assume that the total allocated throughput of each one of the two users is equal to 0. It follows that the input R is of the following form:

$$R = \begin{pmatrix} r_{11} & r_{21} \\ r_{12} & r_{22} \\ 0 & 0 \end{pmatrix}$$

Figure 5.7 shows the structure of the sets $\mathcal{A}$ and $\mathcal{B}$, which is proven in Proposition 5.5.5.

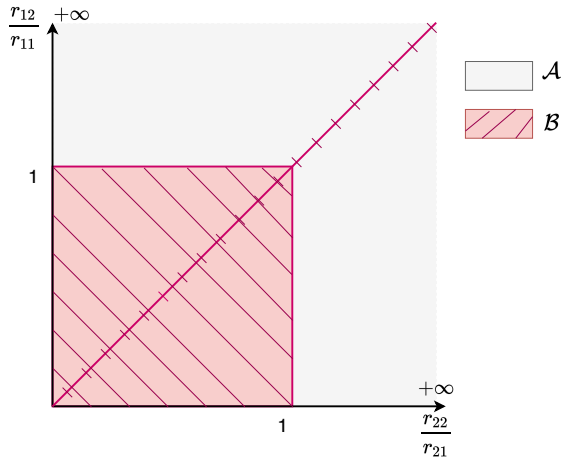


Figure 5.7: 2-dimension illustration for the sets $\mathcal{A}$ and $\mathcal{B}$.

Proposition 5.5.5. *With the axes shown in figure 5.7, we have the following claims:*

- *$\mathcal{B}$ contains the lines $\frac{r_{22}}{r_{21}} = \frac{r_{12}}{r_{11}}$ and the box defined by $0 \leq \frac{r_{22}}{r_{21}} \leq 1$ and $0 \leq \frac{r_{12}}{r_{11}} \leq 1$.*
- *On $\mathcal{A}$, the optimal current allocation is unique, continuous and integer.*
- *On the open box defined by $0 \leq \frac{r_{22}}{r_{21}} < 1$ and $0 \leq \frac{r_{12}}{r_{11}} < 1$, the optimal current allocation is unique, continuous but not integer, except for the inputs R on the line $\frac{r_{22}}{r_{21}} = \frac{r_{12}}{r_{11}}$. On the segment defined by $\frac{r_{22}}{r_{21}} = 1$ and $\frac{r_{12}}{r_{11}} < 1$, and on the segment defined by $\frac{r_{12}}{r_{11}} = 1$ and $\frac{r_{22}}{r_{21}} < 1$, the optimal current allocation is unique, continuous and integer.*
- *On the line $\frac{r_{22}}{r_{21}} = \frac{r_{12}}{r_{11}}$, the optimal current allocation is not unique and not continuous, no matter how we choose the current allocation among the set of optimal allocations.*

Proof. Let us characterize the set $\mathcal{B} = \{R | \#I_1(R) \geq 2\}$. The set $\mathcal{A}$ will be the complement. Since we have 2 users,

$$\rho_{11}^* = \rho_{21}^* = 0. \tag{5.9}$$

Combining with the fourth condition in (KKT) we have:

$$\frac{r_{11}}{C_1^*} = \frac{r_{21}}{C_2^*}. \tag{5.10}$$

There are three possible cases:

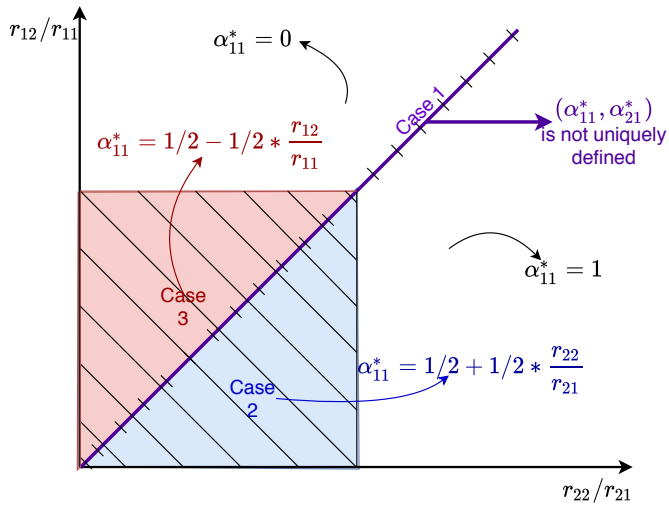


Figure 5.8: The output of STO1 in 2-dimension.

- Case 1: $\alpha^*_{12} \in (0,1)$, which implies that $\alpha^*_{22} \in (0,1)$.
- Case 2: $\alpha^*_{12} = 0$, which implies that $\alpha^*_{22} = 1$.
- Case 3: $\alpha^*_{12} = 1$, which implies that $\alpha^*_{22} = 0$.

Case 1: $\alpha^*_{12} \in (0,1)$ and therefore $\alpha^*_{22} \in (0,1)$. From the second condition in (KKT), it implies $\rho^*_{12} = \rho^*_{22} = 0$. Combining with the fourth condition in (KKT) we have:

$$\frac{r_{12}}{C^*_1} = \frac{r_{22}}{C^*_2}. \tag{5.11}$$

Dividing (5.11) by (5.10) side by side we get:

$$\frac{r_{12}}{r_{11}} = \frac{r_{22}}{r_{21}}. \tag{5.12}$$

Conversely, if $\frac{r_{12}}{r_{11}} = \frac{r_{22}}{r_{21}}$, we shall prove that the solution of (I_R) is of the form

$$\begin{bmatrix} \alpha^*_{11} & \alpha^*_{21} \\ \alpha^*_{12} & \alpha^*_{22} \end{bmatrix} = \begin{bmatrix} a & 1-a \\ \frac{1+t}{2t} - \frac{a}{t} & \frac{t-1}{2t} + \frac{a}{t} \end{bmatrix},$$

for any $a \in [\max(0, \frac{1-t}{2}), \min(1, \frac{1+t}{2})]$, where $t = \frac{r_{12}}{r_{11}} = \frac{r_{22}}{r_{21}}$. Indeed, problem

(I_R) is equivalent to the following problem:

$$\begin{aligned}
&\max\left[\log(\alpha_{11}+t\alpha_{12})+\log(\alpha_{21}+t\alpha_{22})\right]\\
\Leftrightarrow &\max\left[\log(\alpha_{11}+t\alpha_{12})+\log((1-\alpha_{11})+t(1-\alpha_{12}))\right]\\
\Leftrightarrow &\max\left[\log(X)+\log\left((1+t)-X\right)\right]\text{ (where } X=\alpha_{11}+t\alpha_{12})\\
\Leftrightarrow &\max\left[X\left((1+t)-X\right)\right].
\end{aligned} \tag{5.13}$$

We have

$$X\left((1+t)-X\right)\leq\frac{(X+((1+t)-X))^2}{4}=\frac{(1+t)^2}{4},$$

and the equality occurs if only if $X=\frac{1+t}{2}$, i.e, if $\alpha_{12}=\frac{1+t}{2t}-\frac{\alpha_{11}}{t}$. So, we get

$$\alpha^*=\begin{bmatrix} a & 1-a\\ \frac{1+t}{2t}-\frac{a}{t} & \frac{t-1}{2t}+\frac{a}{t}\end{bmatrix},$$

with $a\in[\max(0,\frac{1-t}{2}),\min(1,\frac{1+t}{2})]$ to guarantee that every value in the matrix is in $[0,1]$.

Conclusion for case 1: on the line $\frac{r_{12}}{r_{11}}=\frac{r_{22}}{r_{21}}$, the solution is not unique, the solutions are of the above form.

Case 2: $\alpha^*_{12}=0$ and therefore $\alpha^*_{22}=1$. This implies

$$\rho^*_{22}=0.$$

Therefore:

$$\frac{r_{22}}{C^*_2}=\lambda^*_2\geq\lambda^*_2-\rho^*_{12}=\frac{r_{12}}{C^*_1}. \tag{5.14}$$

Combining with (5.10), we get

$$\frac{r_{22}}{r_{21}}\geq\frac{r_{12}}{r_{11}}. \tag{5.15}$$

On the other hand, in this case Problem (I_R) amounts to finding the maximum of

$$[\log(r_{11}\alpha_{11})+\log(r_{21}(1-\alpha_{11})+r_{22})]:=f(\alpha_{11}).$$

$$f'(\alpha_{11})=0\Leftrightarrow\alpha_{11}=\frac{1}{2}\left(1+\frac{r_{22}}{r_{21}}\right). \tag{5.16}$$

Let us check whether the optimal α^*_{11} is on the boundary or is the above stationary point. On the boundary we have:

$$f(0)=-\infty,$$

$$f(1)=\log(r_{11})+\log(r_{22})>f(0).$$

For the stationary, we have two cases:

- If $\alpha_{11} = \frac{1}{2}\left(1 + \frac{r_{22}}{r_{21}}\right) \in [0,1]$, i.e $r_{22} \leq r_{21}$, then we have:

$$f\left(\frac{1}{2}(1+\frac{r_{22}}{r_{21}})\right) = \log\left(r_{11}\frac{r_{21}+r_{22}}{2r_{21}}\right) + \log\left(\frac{r_{21}+r_{22}}{2}\right) \tag{5.17}$$
$$\geq \log(r_{11}) + \log(r_{22})(\text{since } r_{22} \leq r_{21}) \tag{5.18}$$
$$= f(1) \tag{5.19}$$

  So in this case the optimal solution is given by $\alpha_{11} = \frac{1}{2}\left(1 + \frac{r_{22}}{r_{21}}\right)$ and the matrix input has to satisfy $\frac{r_{22}}{r_{21}} \leq 1$.

- If $\alpha_{11} = \frac{1}{2}\left(1 + \frac{r_{22}}{r_{21}}\right) > 1$ then we get $\frac{r_{22}}{r_{21}} > 1$ and f attains its maximum at $\alpha_{11} = 1$ (and therefore $\alpha_{21} = 0$). But it can not happens in the set $\mathcal{B}$ with $\rho^*_{11} = \rho^*_{21} = 0$. Indeed, in this case, $C^*_1 = r_{11}, C^*_2 = r_{22}$, and

$$\lambda^*_1 = \max\left(\frac{r_{11}}{C^*_1}, \frac{r_{21}}{C^*_2}\right) = \max\left(1, \frac{r_{21}}{r_{22}}\right) = 1. \tag{5.20}$$

  Combining with the fourth condition of (KKT), we get

$$\rho^*_{21} = \lambda^*_1 - \frac{r_{21}}{C^*_2} = 1 - \frac{r_{21}}{r_{22}} > 0. \tag{5.21}$$

  That is a contradiction.

Conversely, if $\frac{r_{22}}{r_{21}} > \frac{r_{12}}{r_{11}}$ (here we consider only the strict inequality since the equality has been already considered in case 1) and $0 \leq \frac{r_{22}}{r_{21}} \leq 1$. Then the solution of (I_R) is this form:

$$\begin{bmatrix} \alpha^*_{11} & \alpha^*_{21} \\ \alpha^*_{12} & \alpha^*_{22} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left(1 + \frac{r_{22}}{r_{21}}\right) & \frac{1}{2}\left(1 - \frac{r_{22}}{r_{21}}\right) \\ 0 & 1 \end{bmatrix}.$$

Indeed, from $\frac{r_{22}}{r_{21}} > \frac{r_{12}}{r_{11}}$ and (5.10) we obtain that

$$\frac{r_{22}}{C^*_2} > \frac{r_{12}}{C^*_1} \implies \rho^*_{12} = \lambda^*_2 - \frac{r_{12}}{C^*_1} > \lambda^*_2 - \frac{r_{22}}{C^*_2} = \rho^*_{22} \geq 0. \tag{5.22}$$

So $\rho^*_{12} > 0 \overset{KKT}{\implies} \alpha^*_{12} = 0$ and therefore $\alpha^*_{22} = 1$. Therefore, (I_R) becomes an optimization problem of one variable α_{11}, and by solving it we get $\alpha^*_{11} = \frac{1}{2}\left(1 + \frac{r_{22}}{r_{21}}\right) \in (\frac{1}{2}, 1]$ and therefore $\alpha^*_{21} = \frac{1}{2}\left(1 - \frac{r_{22}}{r_{21}}\right) \in [0, \frac{1}{2})$. From these values we can compute the dual value:

$$\frac{r_{11}}{C^*_1} = \frac{r_{21}}{C^*_2} = \frac{2r_{21}}{r_{21}+r_{22}}(\text{ therefore } = \lambda^*_1) \implies \rho^*_{11} = \rho^*_{21} = 0. \tag{5.23}$$

Conclusion for case 2: The region in this case is equal to the region defined by $\frac{r_{22}}{r_{21}} \geq \frac{r_{12}}{r_{11}}$ *and* $0 \leq \frac{r_{22}}{r_{21}} \leq 1$*. On the triangle defined by* $\frac{r_{22}}{r_{21}} > \frac{r_{12}}{r_{11}}$ *(strictly inequality here) and* $0 \leq \frac{r_{22}}{r_{21}} \leq 1$*, the solution is unique, and is given by*

$$\alpha_{11} = \frac{1}{2}\left(1 + \frac{r_{22}}{r_{21}}\right).$$

The formula shows the continuity on the interior of the triangle. In this region, except on the segment defined by $\frac{r_{22}}{r_{21}} = 1 \geq \frac{r_{12}}{r_{11}}$*, the solution is not integer.*

Case 3: Similar to case 2, if $\alpha_{12}^* = 1$, then R has to satisfy: $0 \leq \frac{r_{12}}{r_{11}} \leq 1$ and $\frac{r_{12}}{r_{11}} \geq \frac{r_{22}}{r_{21}}$. When $0 \leq \frac{r_{12}}{r_{11}} \leq 1$ and $\frac{r_{12}}{r_{11}} > \frac{r_{22}}{r_{21}}$, the solution is unique and given by

$$\begin{bmatrix} \alpha_{11}^* & \alpha_{21}^* \\ \alpha_{12}^* & \alpha_{22}^* \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left(1 - \frac{r_{12}}{r_{11}}\right) & \frac{1}{2}\left(1 + \frac{r_{12}}{r_{11}}\right) \\ 1 & 0 \end{bmatrix}.$$

Conclusion for case 3: The region in this case is equal to the region defined by $\frac{r_{22}}{r_{21}} \leq \frac{r_{12}}{r_{11}}$ *and* $0 \leq \frac{r_{12}}{r_{11}} \leq 1$*. On the triangle defined by* $\frac{r_{22}}{r_{21}} < \frac{r_{12}}{r_{11}}$ *and* $0 \leq \frac{r_{12}}{r_{11}} \leq 1$*, the solution is unique, and is given by*

$$\alpha_{11} = \frac{1}{2}\left(1 - \frac{r_{12}}{r_{11}}\right). \tag{5.24}$$

The formula shows the continuity on the interior of the triangle. In this region, except on the segment $\frac{r_{12}}{r_{11}} = 1 \geq \frac{r_{22}}{r_{21}}$*, the solution is not integer.*

The above analysis characterizes the structure of the set $\mathcal{B}$. Since $\mathcal{A}$ is the complement of $\mathcal{B}$, it contains two connected components: one defined by $\frac{r_{12}}{r_{11}} > 1$ and $\frac{r_{12}}{r_{11}} > \frac{r_{22}}{r_{21}}$; and the other one defined by $\frac{r_{22}}{r_{21}} > 1$ and $\frac{r_{12}}{r_{11}} < \frac{r_{22}}{r_{21}}$.

Recall that on $\mathcal{A}$ we have: $\#I_1(R) = 1$, i.e at time slot 1 only one user gets a full allocation while the other one gets nothing. There are thus only two options: either $(\alpha_{11}^*, \alpha_{21}^*) = (0, 1)$ or $(\alpha_{11}^*, \alpha_{21}^*) = (1, 0)$. As stated in Proposition 5.5.1, $(\alpha_{11}^*, \alpha_{21}^*)$ is continuous on $\mathcal{A}$. Together with the fact that there are only two possible options for the output which is of discrete type, this implies that F must be equal to a constant (either $(0, 1)$ or $(1, 0)$) in each connected component. Therefore, to know which value of the output corresponds to each connected component, we just need to choose one point in that connected component of $\mathcal{A}$ and solve the optimization problem for that point.

Let use consider the first connected component of $\mathcal{A}$ which is defined by $\frac{r_{12}}{r_{11}} > 1$ and $\frac{r_{12}}{r_{11}} > \frac{r_{22}}{r_{21}}$. In this region, $(\alpha_{11}^*, \alpha_{21}^*) = (0, 1)$. Indeed, let us choose one input point in this connected component such that it satisfies $r_{11} > 1$.

Since either $\alpha_{11}^* = 0$ or $\alpha_{11}^* = 1$, we just need to compare $\max O_{|\alpha_{11}=1}(\alpha_{12})$ and $\max O_{|\alpha_{11}=0}(\alpha_{12})$. We have

$$O_{|\alpha_{11}=1} = \log(r_{11} + \alpha_{12} r_{12}) + \log((1-\alpha_{12}) r_{22}) := g(\alpha_{12}). \tag{5.25}$$

This is a function of one variable α_{12}, we have:

$$g'(\alpha_{12}) = 0 \Longleftrightarrow \alpha_{12} = \frac{1 - r_{11}}{1 + r_{12}}. \tag{5.26}$$

We have $\alpha_{12} = \frac{1-r_{11}}{1+r_{12}} < 0$ since $r_{11} > 1$. Therefore the optimal α_{12}^* is stay in the boundary, i.e,

$$\begin{aligned} \max O_{|\alpha_{11}=1} &= \max\left(O_{|\alpha_{11}=1,\alpha_{12}=1}, O_{|\alpha_{11}=1,\alpha_{12}=0}\right) && (5.27)\\ &= \max\left(\log(r_{11}) + \log(r_{22}), -\infty\right) && (5.28)\\ &= \log(r_{11}) + \log(r_{22}). && (5.29) \end{aligned}$$

On the other hand,

$$O_{|\alpha_{11}=0} = \log(\alpha_{12} r_{12}) + \log(r_{21} + (1-\alpha_{12}) r_{22}). \tag{5.30}$$

Combining with the condition $\frac{r_{12}}{r_{11}} > \frac{r_{22}}{r_{21}}$ we obtain

$$\begin{aligned} O_{|\alpha_{11}=0,\alpha_{12}=1} &= \log(r_{12}) + \log(r_{21}) && (5.31)\\ &> \log(r_{11}) + \log(r_{22}) && (5.32)\\ &= \max O_{|\alpha_{11}=1}. && (5.33) \end{aligned}$$

This implies $\alpha_{11}^* = 0$ in this case.
Similarly, for the connected component defined by $\frac{r_{22}}{r_{21}} > 1$ and $\frac{r_{12}}{r_{11}} < \frac{r_{22}}{r_{21}}$ we have $\alpha_{11}^* = 1$.

Conclusion:

- $\mathcal{B}$ contains the box restricted by $0 \le \frac{r_{12}}{r_{11}} \le 1$ and $0 \le \frac{r_{22}}{r_{21}} \le 1$ and the line $\frac{r_{22}}{r_{21}} = \frac{r_{12}}{r_{11}}$. $\mathcal{A}$ is the remaining.
- As proven in Proposition 5.5.1, F is continuous that on $\mathcal{A}$, and moreover
  - on the region defined by $\frac{r_{12}}{r_{11}} > 1$ and $\frac{r_{12}}{r_{11}} > \frac{r_{22}}{r_{21}}$, $\alpha_{11}^* = 0$,
  - on the region defined by $\frac{r_{22}}{r_{21}} > 1$ and $\frac{r_{12}}{r_{11}} < \frac{r_{22}}{r_{21}}$, $\alpha_{11}^* = 1$.

  Combining with the formulas obtained in the three above cases for the set $\mathcal{B}$, we can see that the set of all discontinuities is the line $\frac{r_{12}}{r_{11}} = \frac{r_{22}}{r_{21}}$.
- $(\alpha_{11}^*, \alpha_{21}^*)$ is uniquely defined except on the line $\frac{r_{22}}{r_{21}} = \frac{r_{12}}{r_{11}}$.

□

Figure 5.8 illustrates Proposition 5.5.5 by showing the different regions and the optimal solution in each region.

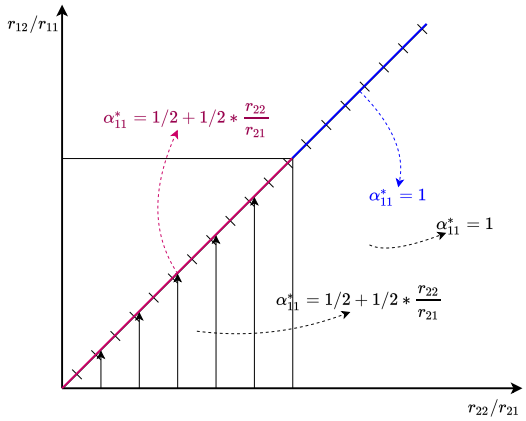


Figure 5.9: Choosing solution in the line to obtain a continuous function in 2-dimension.

5.5.3 Proposed method to obtain a continuous function

In the above 2-dimension case, the set of discontinuities (the line) is a small set in the sense that it has Lebesgue measure equal to 0. However it makes the model difficult to learn when the matrix input is near the line, since it oscillates between $\alpha^*_{\mathbf{1}} = (0, 1)$ and $\alpha^*_{\mathbf{1}} = (1, 0)$ out side the box and oscillates between $\alpha^*_{\mathbf{1}} = (\frac{1}{2} - \frac{1}{2}x, \frac{1}{2} + \frac{1}{2}x)$ and $\alpha^*_{\mathbf{1}} = (\frac{1}{2} + \frac{1}{2}y, \frac{1}{2} - \frac{1}{2}y)$ ($x \approx y$ and $x, y > 0$) inside the box. In order to make a continuous function, we propose to choose the optimal allocation when it is not unique as illustrated in Figure 5.9. Define $H(a_1, a_2) = (\max(a_1, a_2), \min(a_1, a_2))$. If we choose the solution as described in Figure 5.9, $H \circ F$ is a continuous function. It is reasonable to hope that learning $H \circ F$ is easier than learning the non-continuous function F directly.

In general, choosing an order in order to have a continuous function is not an easy task, since to choose an order as above we actually have to solve an optimization problem. We instead propose several heuristic orders, that are either based on current rate, based on the ratio between current rate and cumulative (the PF index), or based on $(\mathrm{PS})^2$S index. The numerical results of these heuristic orders are shown in Section 5.6.4.

5.5.4 Loss, DFNN architecture, initial parameters and optimizer

We will try several different loss functions and architectures and compare them in the numerical section. The initial paramters (weights) of the DFNN

will be chosen as proposed in [38], which allows the initial parameters to be not too big and not too small. The optimizer is Adam first introduced in [34] which is a stochastic first-order gradient-based optimization. The convergence of Adam is proven in [58].

5.6 Numerical Comparisons

In this section, we do simulations to evaluate the influence of many factors on the behavior of the DFNN model (loss functions, architecture of the DFNN, ordering scheme and learning with dual value). We use the keras library [16] to implement our code and follow the instructions in [15].

There are actually a lot of factors that can have an impact on the behavior of the learning procedure such as: initial learning rate, learning rate decay, optimizer, initial weight, number of parameters, activation functions in layers... Here we are not able to justify all our choices, but we focus on the factors which have the most significant impact on the learning algorithm in our opinion. The initial learning rate is chosen equal to 0.0015 and after each epoch, this learning rate decays by a factor 0.998.

5.6.1 An Unified Data Generator for Comparison

To support the comparisons in this section, the data (both for training and validation) is generated as follows. The number of users is generated randomly from 0 to $K_M = 10$. The sojourn time of each user is generated in $(0, 400)$ seconds. This value could of course be increased, but here in order to reduce the learning time and be able to make many comparisons, we consider only small scenarios. The transmission rate in each small slot is generated randomly between 0 and $5 * \delta/\Delta$. The rate we use for evaluating in SUMO scenarios are given by

$$r(x) = \eta \left(1 + \kappa\, e^{-d(x,BS)/\sigma}\right), \tag{5.34}$$

where $d(x, BS)$ is the distance from position x to the BS, and η represents the noise level. For the SUMO scenarios in Section 5.6.5, we use $\kappa = 3$, $\sigma = 100$ and $\eta \sim \text{Uniform}(0.7, 1.3)$. The others parameters are equal to $J = 10, \Delta = 1$ s, $\delta = 2$ ms.

5.6.2 Comparison of different DFNN architectures

In this part, we will consider 4 different architectures of the DFNN model and compare their performances. For the 4 models, the activation function

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 120)               0
_________________________________________________________________
dense_1 (Dense)              (None, 500)               60500
_________________________________________________________________
activation_1 (Activation)    (None, 500)               0
_________________________________________________________________
batch_normalization_1 (Batch (None, 500)               2000
_________________________________________________________________
dense_2 (Dense)              (None, 10)                5010
_________________________________________________________________
activation_2 (Activation)    (None, 10)                0
=================================================================
Total params: 67,510
Trainable params: 66,510
Non-trainable params: 1,000
```

Figure 5.10: Model 1 architecture.

used in hidden layers is the relu function, whereas the output layer uses the softmax function since we want the sum of the allocations to be equal to 1. In this comparison, we use the same loss function for all models, the huber loss [74].

Model 1

The first model used in this section contains 2 layers which are 1 hidden layer and 1 output layer. The hidden layer contains 500 units, and in total the model has $67,510$ parameters. The architecture of this model is illustrated in Figure 5.10.

Model 2

As the first model, the second model contains 2 layers: 1 hidden layer and 1 output layer. However, the hidden layer contains 1000 units, and in total the model has $135,010$ parameters. We take the same number of layers as in model 1 (but more units in hidden layers) in order to compare whether it is better to have more parameters.

Model 3

As the two previous models, the third model contains 2 layers (1 hidden layer and 1 output layer). The hidden layer contains 100 units, and we have $13,510$ parameters in total. We take the same number of layers as in model

1 (but fewer units in hidden layers) to compare whether it is better to have fewer parameters.

Model 4

The last model contains 10 layers which are 9 hidden layers and 1 output layer. Each hidden layer contains 82 units, and in total the model contains $67,496$ parameters. We take a model has almost the same number of parameters with the model 1, to compare whether it is better to have more layers or fewer layers.

Remark 5.6.1 (Joint power control and channel allocation)**.** *For the joint power control and channel allocation problem, we still compare the four above models except that the output layer of each model will be modified since it includes not only the channel allocation but also the power.*

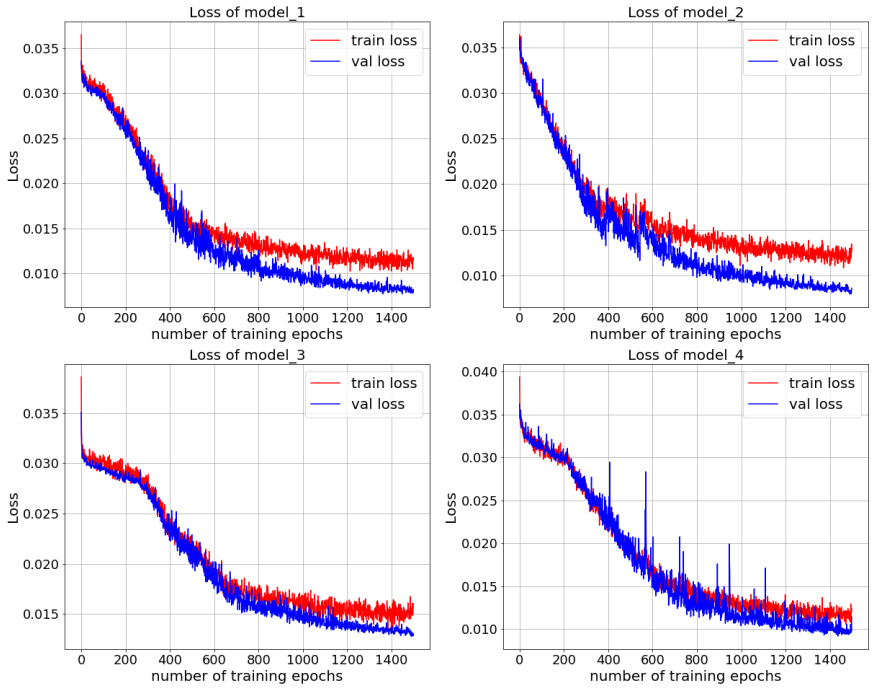
Figure 5.11a illustrates the loss of the 4 models on training and validation data. Figure 5.11b plots loss and absolute error of the 4 models on the same axis on validation set. The same quantities but for the problem of joint power control and channel allocation are shown in Figure 5.12a and Figure 5.12b respectively.

From these figures, we observe that for the model without power control:

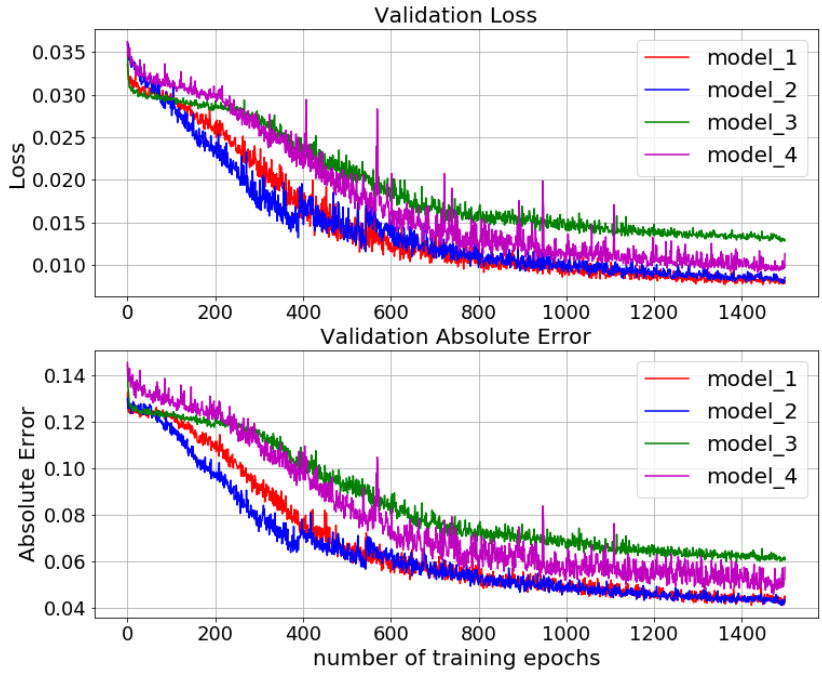
- Having almost the same number of parameters, model 1 with few layers is better than model 4.
- Having the same layers, model 1 and model 2 with more parameters are better than model 3.
- Model 1 and model 2 behave similarly and have the same number of layers. However model 1 has less parameters than model 2 so it is less costly from a computational point of view. Therefore from now on we shall use model 1 for other comparisons in the sequel.

For the model with power control:

- Having almost the same number of parameters, model 1 with few layers is slightly better than model 4, but the difference is quite small in this case.
- Having the same layers, model 1, 2 and 3 are almost the same but model 3 has fewer parameters.

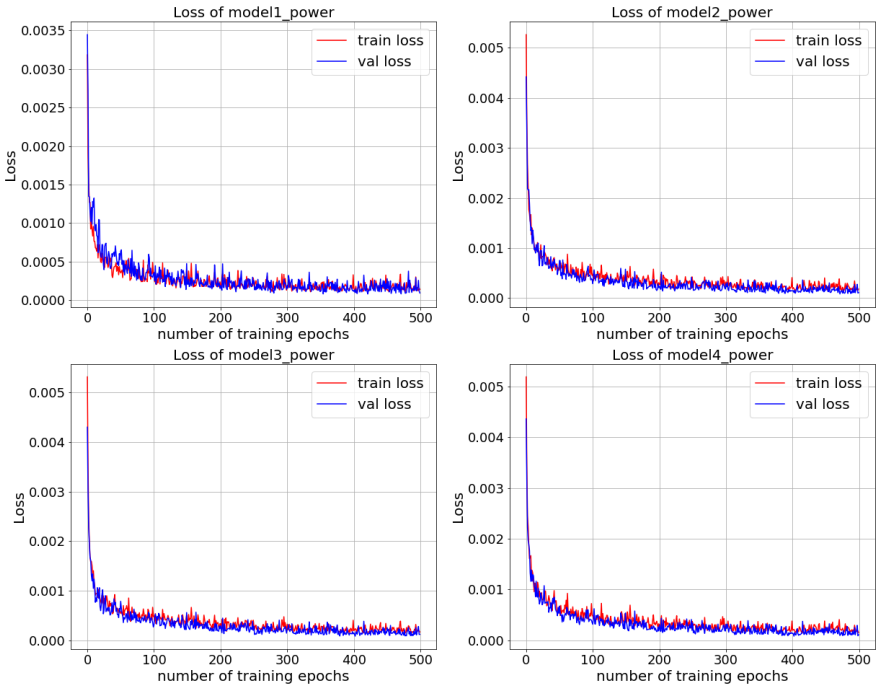


(a) Loss on training set and validation set of each model

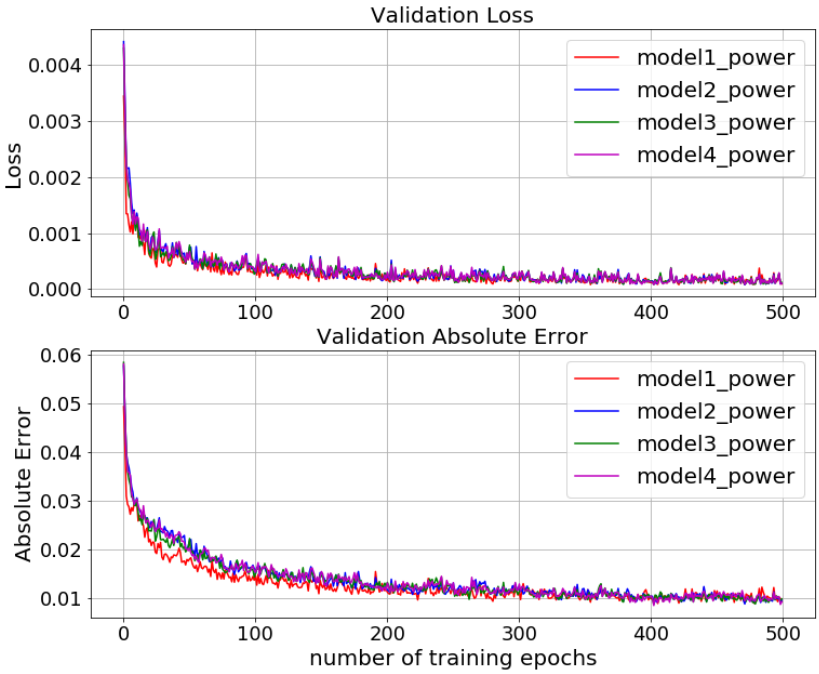


(b) Plot on same axis for loss and absolute error on validation set of all the four models

Figure 5.11: Comparison of the 4 DFNN models.

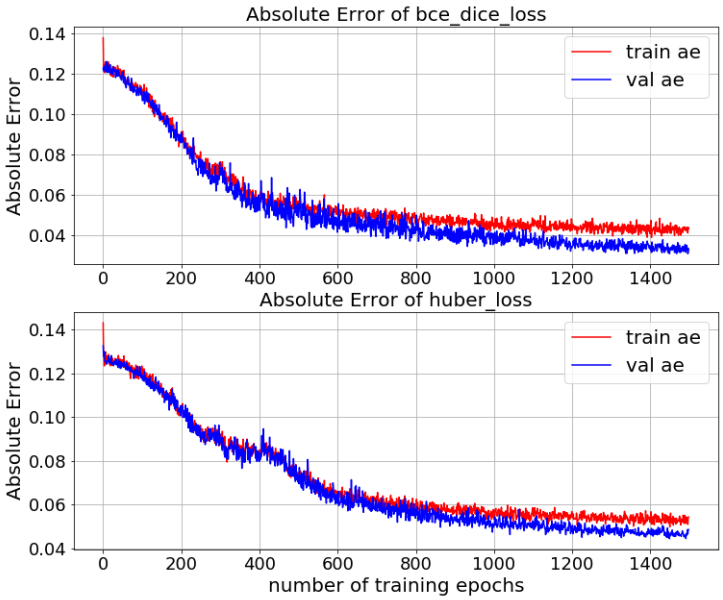


(a) Loss on training set and validation set of each model

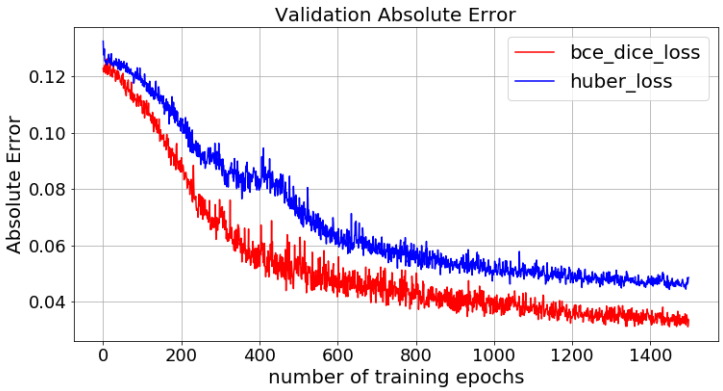


(b) Plot on same axis for loss and absolute error on validation set of all the four models

Figure 5.12: Comparison of the 4 DFNN models for the joint power control and channel allocation problem.



(a) Absolute error on training and validation set of the two losses



(b) Plot on same axis of the absolute error on validation of the two losses

Figure 5.13: Comparison of Loss functions.

5.6.3 Comparisons of different loss functions

To compare the quality of the learning model obtained using different loss functions, we use the same model, that is model 1. Figure 5.13 presents the results obtained with the huber loss [74], with the sum of binary cross-entropy [72] and dice loss (which equals to 1− dice coefficient [73]). The second loss function is denoted by bce_dice loss. From the figures, the bce_dice loss function is better in this case. So for the next comparisons, we shall use model 1 and bce_dice loss.

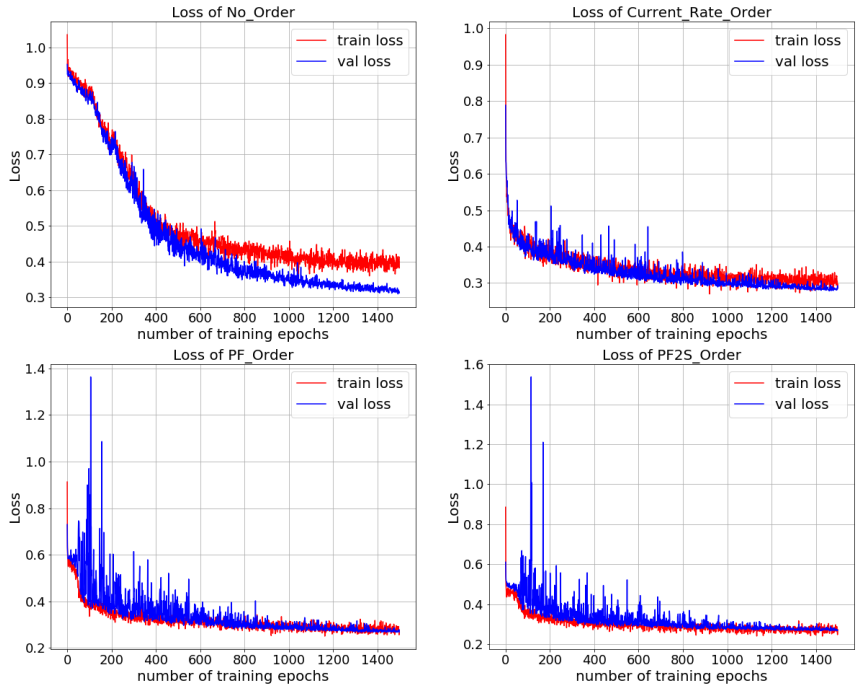
5.6.4 Comparisons of different ordering schemes

As mentioned above, we shall use model 1 and bce_dice loss for this comparison. We compare the results obtained with 4 different ordering schemes: no order, current rate order, PF order and $(PS)^2S$ order. Figure 5.14 presents our numerical results. $(PS)^2S$ order seems the best one among the 4 ordering schemes.

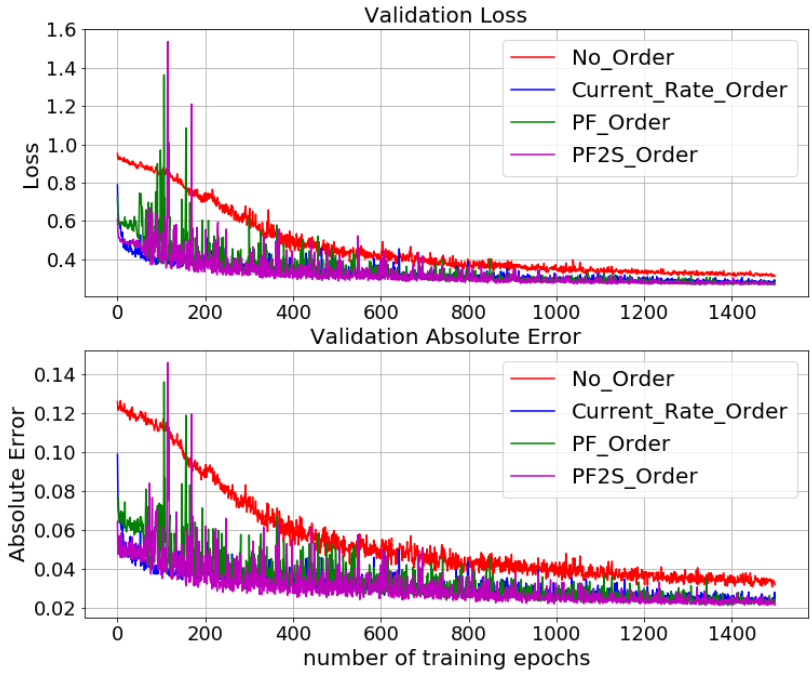
5.6.5 Comparisons of learning the channel allocation against learning dual values

In this section, we compare learning dual values against learning directly the allocation. As mentioned above, for learning the allocation, we use model 1 and bce_dice loss. Since the value of the dual need not lie in $[0, 1]$, we cannot use and a softmax activation function in the output layer and bce_dice loss directly. We shall normalize the dual values by dividing each element by the sum of all elements before fitting it into the DFNN model. This normalization is continuous, therefore we still have a continuous mapping. Finally, we can use sofmax and bce_dice loss. The absolute error of the different learning schemes are shown in Figure 5.15.

Since the values of the allocation and the dual are different, it is not possible to compare the losses or absolute errors directly as before. We shall instead compare three different learning schemes (learning allocation without order, learning allocation with $(PS)^2S$ order and learning the dual values) on two different scenarios created with SUMO. The first scenario contains 244 users and lasts in 61.7 minutes inside the coverage of one BS. The map of this scenario is shown in Figure 5.2. The results obtained with the different learning schemes on this scenario are shown in Figure 5.17. The second scenario contains 214 users and lasts 62.4 minutes. The map of this scenario is shown in Figure 5.16. The results obtained with the different learning schemes on this scenario are shown in Figure 5.18.



(a) Loss on training and validation set



(b) Plot on same axis of the loss and absolute error on validation set

Figure 5.14: Comparison of 4 different types of ordering schemes.

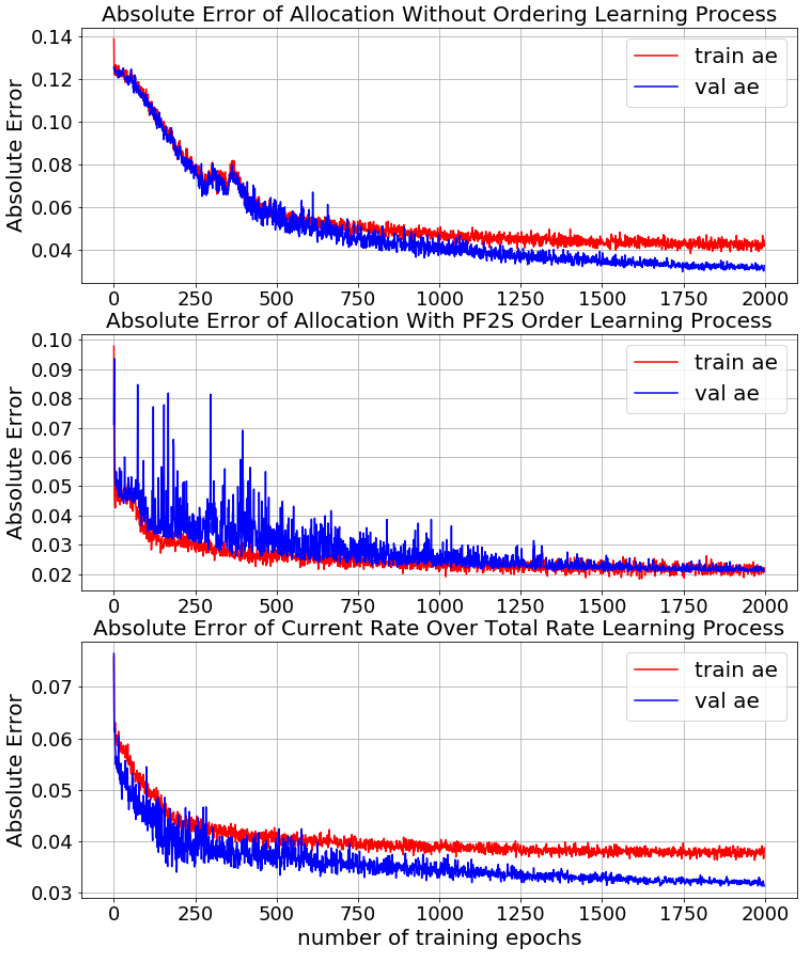


Figure 5.15: Comparisons of Absolute Error of the Learning Allocation and Learning Dual Value.

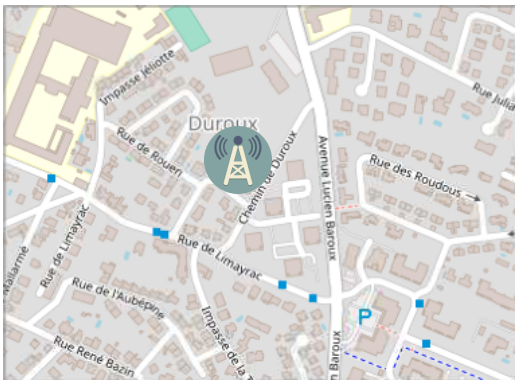


Figure 5.16: Duroux, one BS type LTE1800, operator SFR. The actual size is around 350*500m.

We see that learning dual values outperforms the two other learning schemes: it is more stable, provides a better allocation and it converges faster than learning directly the allocation without any ordering scheme. It costs the same time for *prediction* as the other two approaches since they are using the same architecture (i.e, model 1). However learning directly allocation with a user ordering scheme requires to compute in addition the order, which consumes more computing time than the other two approaches.

5.7 Computing times

The computing time of STO1 depends on the convex optimization solver used, whereas the learning algorithm has only to feed the DFNN model with the input matrix. We consider the same setting as in Section 5.6.2, that is $K = 10$ (there are 10 users in the system) and the short term horizon is $J = 10$ seconds. For these values, the average computing time of mosek is around 43.7 ms, whereas the prediction with the DFNN model (model 1) takes only 0.65 ms on average. These computing times are averages over 10000 samples, and both are measured on a machine using GPU (graphics processing unit) which allows computing many calculations in parallel.

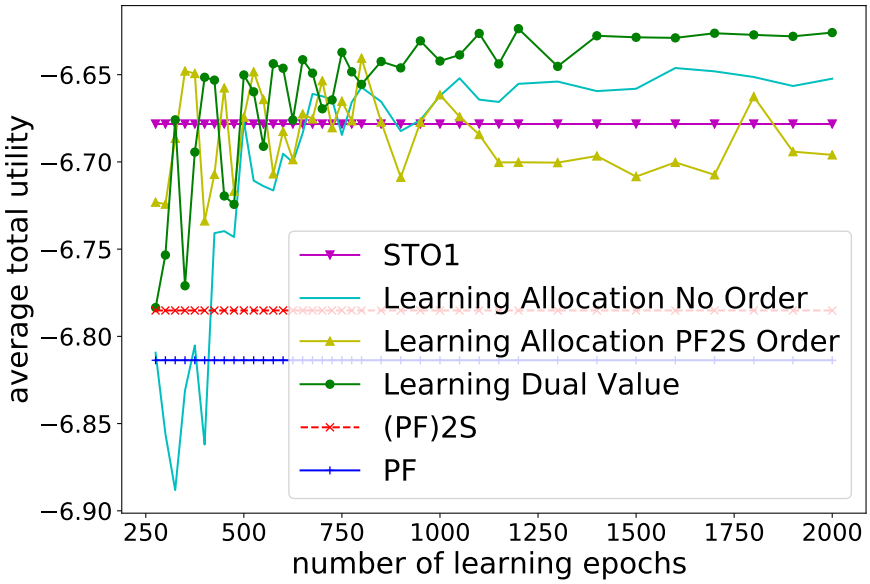


Figure 5.17: Comparisons of evaluated on Carmes scenario created by SUMO of Allocation and Dual Value.

5.8 Discussion on a reinforcement learning based approach for learning Optimal Policy

In this section, we shall discuss about learning the optimal policy by using Deep Q Network (DQN) learning. The background of DQN is recalled in 2.5.3. To do that, it requires the state of the system follows a Markov process. To simplify, we consider a straight road only in this discussion.

Assume that every time-slot, at most one arrival enters the system and the new arrival comes independently and with probability p. Now we define the tuple of state, action, reward one-step and transition matrix to get a Markov Decision Problem (MDP).

State and action: A state is a matrix 2-D size $N \times 2$ (N is number of small slots in coverage range), where:

- first row is vector of 0 and 1 to indicate whether there is a car in that place or not.
- second row is a vector of corresponding cumulative data.

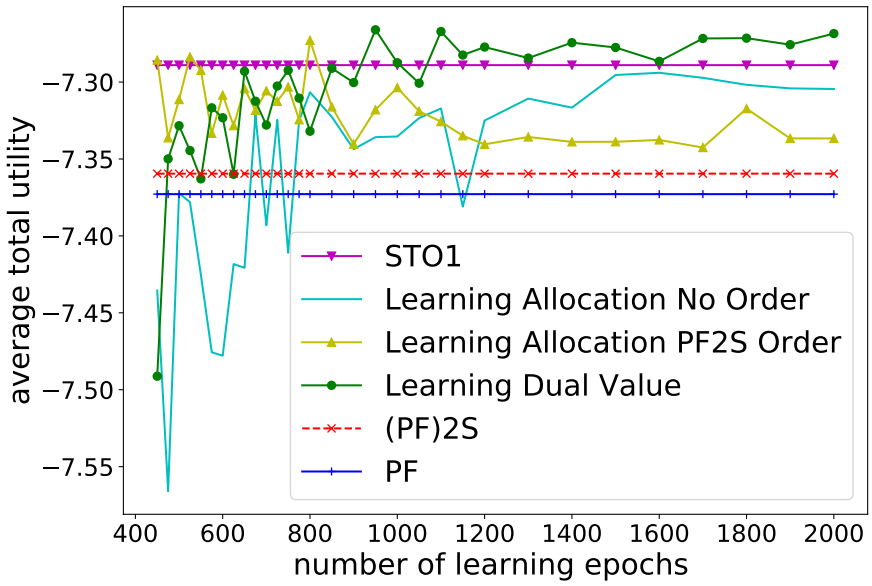


Figure 5.18: Comparisons of evaluated on Duroux scenario created by SUMO of Allocation and Dual Value.

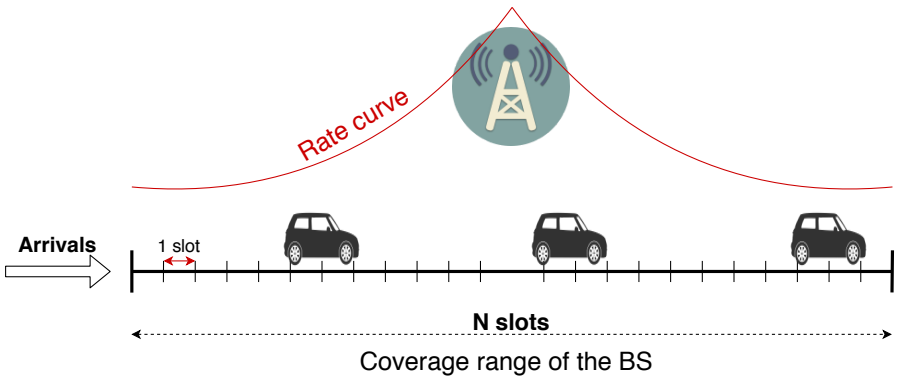


Figure 5.19: Straight road.

cumulative data of every user is initialized with d_0 where d_0 is a very small number to avoid log of zero. If there is no user in a place, the cumulative data for that place is filled by d_0 by convention. Here the future rate is not taken into account in the state, but the Q function is expected to learn it by learning the data rate map during iterations, since here we restrict the learning on a specific map that is a straight road. So, in this setting, after getting the Q function, the algorithm will work only for this specific map (this is different from the supervised learning that was done for STO1 where the model is trained on a general system and therefore will work on an arbitrary network).

Action space $\mathcal{A} = \{1, 2, ..., N\}$, if action $a = k$ it means the car in place k is chosen.

Reward one-step is defined:

$$R(X, a) = \log(x_{a,2} + r(a)) - \log(x_{a,2}),$$

where r is the rate function, $r(a)$ means rate at position a and $x_{a,2}$ is the element in position a of the second row of the matrix X.

Transition matrix: As a new arrival can enter with probability p and the vector of cumulative data can be changed, if we are currently in state $X = (x_{i,j})_{i=1,2,j=1,2,...N}$, it can be turned into new state as follows:

- with probability p it goes to state $Y = (y_{i,j})_{i=1,2,j=1,2,...N}$ where:
  - ⋆ $y_{1,1} = 1, y_{1,2} = d_0$,
  - ⋆ and $y_{i+1,1} = x_{i,1}$, $y_{i+1,2} = x_{i,2}$ if user in place i is not chosen and otherwise $y_{i+1,2} = x_{i,2} + r(i)$ where $r(i)$ stands for the rate at place i, for all $i = 2, 3, .., N - 1$.
- with probability $1 - p$ it goes to state $Y = (y_{i,j})_{i=1,2,j=1,2,...N}$ where:
  - ⋆ $y_{1,1} = 0, y_{1,2} = d_0$,
  - ⋆ and $y_{i+1,1} = x_{i,1}$, $y_{i+1,2} = x_{i,2}$ if user in place i is not chosen and otherwise $y_{i+1,2} = x_{i,2} + r(i)$ where $r(i)$ stands for the rate at place i, for all $i = 2, 3, .., N - 1$.

Objective: We want to build a potential function, which takes state X as an input and gives us suggestion for the good action a, that is Q-function $Q(X, a)$.

By value iteration, we get the optimal solution for this MDP when N small. In value iteration, we need to store the table of the value function of all states which is not possible when N large. One of the methods people use

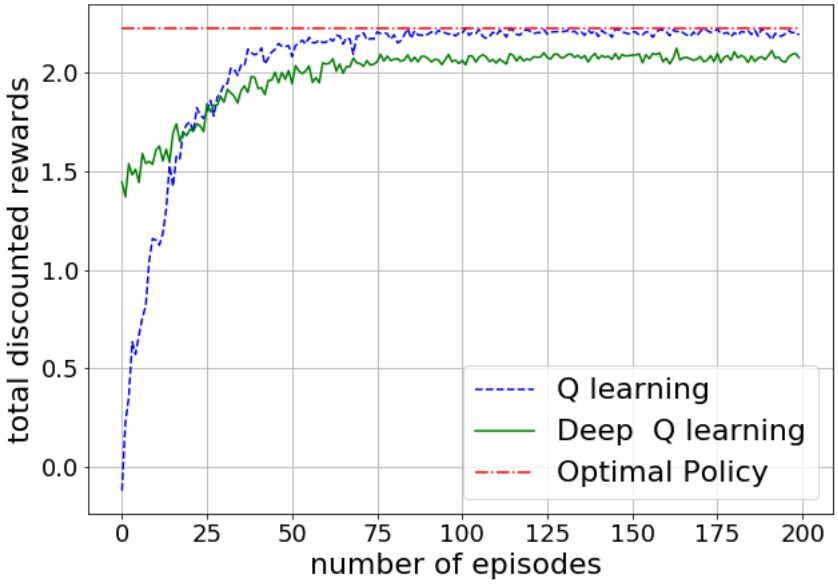


Figure 5.20: Deep Q learning comparing to Q learning and Optimal Policy, $N = 3, p = 0.5$, discounted factor $\gamma = 0.9$.

to deal with large state space is to learn Q function instead of V by using function approximation.

Fig. 5.20 illustrates the comparison of three algorithms: Optimal Policy got by value iteration, Q learning and Deep Q learning. In that numerical result, in each episode, we do the same 2000 iterations both for DQN and Q learning. Here we can see even with $N = 3$ it takes so long time (75 episodes) to see the convergence of the DQN and DQN even converges quite far from the true value got by Value Iteration. There are two disadvantages of this method. Firstly, when N increases, the size of the state-space increases exponentially. In fact, N should be of the order of thousands, which makes a very huge state space. Secondly, by the way we defined the model above, we do not have a terminal state. Those two things may make both of Q learning and Deep Q learning more difficult to converge to the true value.

5.9 Summary and Discussion

We have proposed machined learning based method for learning one of our heuristics (the STO 1) introduced in chapter 3 and chapter 4 in order to

produce an approximate solution which can process many times faster. We have proved that the primal values used in STO 1 are not continuous in the input. We then proposed some ordering methods to reduce the discontinuity of the primal values. We also proved that the dual values are continuous, we thus proposed to learn the duals instead of the primals. The ordering methods help the model converges faster but not always increases the performance, while the dual method helps the model learn faster and perform better. We also presented another approach using reinforcement learning. Different from Supervise Learning approach, Reinforcement Learning works for specific topology maps, needs long time to converge even for very small scenario.

The way we define the state for the DFNN here makes the model can learn on a general network, not only for specific topology map. There are several directions of research that can be investigated to improve the learning, such as better generator of data, better loss function, better architecture DFNN model, and other thing such as optimizer, learning rate, etc.

Chapter 6

Discussion and Future Works

In this thesis, we have investigated two different resource allocation problems. The first one deals with the channel allocation to vehicular users in cellular networks, whereas the second one is the joint channel allocation and power control problem in cellular networks. For each problem, we have proposed two heuristic algorithms which assume that the mean future rates of vehicular users can be predicted over a few seconds. The main originality of our algorithms is that they combine this information on future radio conditions with the information on present channel channel conditions to compute what should be the optimal allocation in the next few seconds. This is done by solving a relaxed version of a utility maximization problem over a short-term horizon. This extra-knowledge on future allocations can then be used for improved online channel allocation and/or power control. Our algorithms can be applied to general network topologies and are shown numerically to outperform other existing algorithms. In the last part of the thesis , we have also presented a machine learning-based method for obtaining approximate algorithms that mimic our resource allocation algorithms but can be run many times faster. Numerical results show that the approximate algorithms are well suited to real-time processing and that the degradation of the channel allocation quality is very small. Below, we discuss some open research directions that could be followed to extend our work.

There are several directions in which our work can be extended. One avenue is to implement a centralized and coordinated version for the multiple BSs case of our heuristics. On the analytical side, obtaining sub-optimality bounds for our heuristics would also be worth investigating. It would be also interesting to study the robustness of our heuristic algorithms with respect to errors in future information. The dual-decomposition methods presented in [54] could also be applied to get another heuristic algorithm but with a penalty term instead of the average power constraint.

For both problems, it would be interesting to extend our work to different utility functions and to problems in which users have QoS requirements (e.g., latency or jitter). We also note that the joint channel allocation and power control problem could be studied on the up-link, and that similar algorithms could be investigated for the opportunistic utility model.

Finally, another interesting extension would be to consider settings in which the information available on users is heterogeneous. We particularly have in mind the two following settings:

- a setting in which future information is not available for some users which are unpredictable. For instance, the future positions of pedestrians are probably more difficult to predict than those of cars moving along a highway. An idea that could be applied in this case is to apply PF scheduling for those "unpredictable" users, and to use our heuristics for the other users,

- a setting in which the positions of some users can be predicted with higher accuracy and over a longer time period than those of the other users. For example, compared to the individual users, the public transports (bus, tram, metro,...) may be easier to predict in longer time period and with higher accuracy. Another example is the case of vehicular users agreeing to share the information on their itinerary. An idea that could be applied in these situations is that the estimated future rate can be equal to expectation over all possibilities of the future path prediction.

Bibliography

[1] Press release: Local clouds for greater road safety - joint development project between bosch, nokia, and deutsche telekom. 2016. https://www.bosch-presse.de/pressportal/de/en/local-clouds-for-greater-road-safety-63296.html.

[2] Just one autonomous car will use 4,000 gb of data/day. December 2016. http://www.networkworld.com/article/3147892/internet/one-autonomous-car-will-use-4000-gb-of-dataday.html.

[3] H. Abou-zeid, H. S. Hassanein, and N. Zorba. Long-term fairness in multi-cell networks using rate predictions. In *2013 7th IEEE GCC Conference and Exhibition (GCC)*, pages 131–135, Nov 2013.

[4] M. H. Ahmed, O. Dobre, and R. Almatarneh. Analytical evaluation of the performance of proportional fair scheduling in ofdma-based wireless systems. *Journal of Electrical and Computer Engineering*, 2012, 08 2012.

[5] A. Anand and G. Veciana. A whittle's index based approach for qoe optimization in wireless networks. *ACM SIGMETRICS Performance Evaluation Review*, 46:39–39, 06 2018.

[6] R. Atallah, C. Assi, and M. Khabbaz. Deep reinforcement learning-based scheduling for roadside communication networks. pages 1–8, 05 2017.

[7] H. J. Bang, T. Ekman, and D. Gesbert. Channel predictive proportional fair scheduling. *IEEE Transactions on Wireless Communications*, 7(2):482–487, February 2008.

[8] D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2nd edition, 2000.

[9] V. Borkar and S. Pattathil. Whittle indexability in egalitarian processor sharing systems. *Annals of Operations Research*, 08 2017.

[10] S. Borst. User-level performance of channel-aware scheduling algorithms in wireless data networks. *IEEE/ACM Transactions on Networking*, 13(3):636–647, June 2005.

[11] S. Borst, N. Hegde, and A. Proutiere. Mobility-driven scheduling in wireless networks. In *IEEE INFOCOM 2009*, pages 1260–1268, April 2009.

[12] T. Bu, L. Li, and R. Ramjee. Generalized proportional fair scheduling in third generation wireless data networks. In *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, pages 1–12, 2006.

[13] P. Chandur, R. M. Karthik, and K. M. Sivalingam. Performance evaluation of scheduling algorithms for mobile wimax networks. In *2012 IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 764–769, March 2012.

[14] M. Chiang, P. Hande, T. Lan, and C. W. Tan. Power control in wireless cellular networks. *Foundations and Trends® in Networking*, 2(4):381–533, 2008.

[15] F. Chollet. *Deep Learning with Python*. Manning Publications Co., USA, 1st edition, 2017.

[16] F. Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.

[17] L. Condat. Fast projection onto the simplex and the l1 ball. *Math. Program.*, 158:575–585, 2016.

[18] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989.

[19] E. Dahlman, S. Parkvall, and J. Sköld. *4G LTE-Advanced Pro and The Road to 5G (Third Edition)*. Academic Press, third edition edition, 2016.

[20] S. Diamond and S. Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[21] C. B. Do. Convex optimization overview (cnt'd). *CS229 Lecture notes*, November 2009.

[22] R. Eghbali and M. Fazel. Worst case competitive analysis of online algorithms for conic optimization. *CoRR*, abs/1611.00507, 2016.

[23] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[24] K. Gregor and Y. LeCun. Learning fast approximations of sparse coding. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 399–406, Madison, WI, USA, 2010. Omnipress.

[25] D. Grewe, M. Wagner, M. Arumaithurai, I. Psaras, and D. Kutscher. Information-centric mobile edge computing for connected vehicle environments: Challenges and research directions. In *Proceedings of the Workshop on Mobile Edge Communications*, MECOMM '17, page 7–12, New York, NY, USA, 2017. Association for Computing Machinery.

[26] J. M. Holtzman. Asymptotic analysis of proportional fair algorithm. *12th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications. PIMRC 2001. Proceedings (Cat. No.01TH8598)*, 2:F–F, 2001.

[27] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251 – 257, 1991.

[28] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359 – 366, 1989.

[29] J. Huang, V. Subramanian, R. Agrawal, and R. Berry. Downlink scheduling and resource allocation for ofdm systems. *IEEE Transactions on Wireless Communications*, 8:288–296, 01 2009.

[30] R. Hussain, J. Son, H. Eun, S. Kim, and H. Oh. Rethinking vehicular communications: Merging vanet with cloud computing. In *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*, pages 606–609, 2012.

[31] A. Karbassi and M. Barth. Vehicle route prediction and time of arrival estimation techniques for improved transportation system management. In *IEEE IV2003 Intelligent Vehicles Symposium. Proceedings (Cat. No.03TH8683)*, pages 511–516, 2003.

[32] F. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, 8(1):33–37, 1997.

[33] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49(3):237–252, Mar 1998.

[34] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[35] A. Kolmogorov. On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. *Dokl. Akad. Nauk SSSR*, 114(5):953–956, 1957.

[36] J. Krumm. A markov model for driver turn prediction. In *Society of Automotive Engineers (SAE) 2008 World Congress, April 2008*. SAE 2008 World Congress, April 2008. Lloyd L. Withrow Distinguished Speaker Award.

[37] H. J. Kushner and P. A. Whiting. Convergence of proportional-fair sharing algorithms under general conditions. *IEEE Transactions on Wireless Communications*, 3(4):1250–1259, July 2004.

[38] Y. Lecun, L. Bottou, G. Orr, and K.-R. Müller. Efficient backprop. 08 2000.

[39] J. Lee, R. R. Mazumdar, and N. B. Shroff. Opportunistic power scheduling for dynamic multi-server wireless systems. *IEEE Trans. Wireless Communications*, 5(6):1506–1515, 2006.

[40] C. Li and M. J. Neely. Network utility maximization over partially observable markovian channels. *Perform. Evaluation*, 70(7-8):528–548, 2013.

[41] E. Liu and K. K. Leung. Proportional fair scheduling: Analytical insight under rayleigh fading environment. In *2008 IEEE Wireless Communications and Networking Conference*, pages 1883–1888, 2008.

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

[43] R. Margolies, A. Sridharan, V. Aggarwal, R. Jana, N. K. Shankaranarayanan, V. A. Vaishampayan, and G. Zussman. Exploiting mobility in proportional fair cellular scheduling: Measurements and algorithms. *IEEE/ACM Trans. Netw.*, 24(1):355–367, Feb. 2016.

[44] A. Mesbah. Stochastic model predictive control: An overview and perspectives for future research. *IEEE Control Systems Magazine*, 36(6):30–44, Dec 2016.

[45] J. Mo and J. Walrand. Fair end-to-end window-based congestion control. *IEEE/ACM Transactions on Networking*, 8(5):556–567, 2000.

[46] S. Moharir and S. Shakkottai. Maxweight versus backpressure: Routing and scheduling in multichannel relay networks. *IEEE/ACM Transactions on Networking*, 23(5):1584–1598, Oct. 2015.

[47] H. Montanelli and H. Yang. Error bounds for deep relu networks using the kolmogorov–arnold superposition theorem, 2019.

[48] K. Navaie and H. Yanikomeroglu. Fairness of resource allocation in cellular networks : a survey. 2005.

[49] M. J. Neely. Energy optimal control for time-varying wireless networks. *IEEE Transactions on Information Theory*, 52(7):2915–2934, 2006.

[50] M. J. Neely. Order optimal delay for opportunistic scheduling in multi-user wireless uplinks and downlinks. *IEEE/ACM Transactions on Networking*, 16(5):1188–1199, Oct 2008.

[51] N. Nguyen, O. Brun, and B. Prabhu. An algorithm for improved proportional-fair utility for vehicular users. *The 25th International Conference on Analytical and Stochastic Modelling Techniques and Applications ASMTA-2019*, May 2019.

[52] N. Nguyen, O. Brun, and B. Prabhu. Joint downlink power control and channel allocation based on a partial view of future channel conditions. *The 15th Workshop on Resource Allocation, Cooperation and Competition in Wireless Networks*, June 2020.

[53] T. T. N. Nguyen, U. Ayesta, and B. J. Prabhu. Scheduling users in drive-thru internet : a multi-armed bandit approach. 2019.

[54] D. P. Palomar and Mung Chiang. A tutorial on decomposition methods for network utility maximization. *IEEE Journal on Selected Areas in Communications*, 24(8):1439–1451, 2006.

[55] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

[56] B. Radunovic and J. Le Boudec. A unified framework for max-min and min-max fairness with applications. *IEEE/ACM Transactions on Networking*, 15(5):1073–1083, 2007.

[57] B. Radunovic and J. Le Boudec. A unified framework for max-min and min-max fairness with applications. *IEEE/ACM Transactions on Networking*, 15(5):1073–1083, 2007.

[58] S. J. Reddi, S. Kale, and S. Kumar. On the convergence of adam and beyond. *CoRR*, abs/1904.09237, 2019.

[59] S. Shenker. Fundamental design issues for the future internet. *IEEE Journal on Selected Areas in Communications*, 13(7):1176–1188, 1995.

[60] H. SHI, R. V. Prasad, E. Onur, and I. G. M. M. Niemegeers. Fairness in wireless networks:issues, measures and challenges. *IEEE Communications Surveys Tutorials*, 16(1):5–24, 2014.

[61] A. L. Stolyar. Maxweight scheduling in a generalized switch: State space collapse and workload minimization in heavy traffic. *Ann. Appl. Probab.*, 14(1):1–53, 02 2004.

[62] V. G. Subramanian, R. A. Berry, and R. Agrawal. Joint scheduling and resource allocation in CDMA systems. *IEEE Trans. Inf. Theory*, 56(5):2416–2432, 2010.

[63] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos. Learning to optimize: Training deep neural networks for wireless resource management. *CoRR*, abs/1705.09412, 2017.

[64] S. Sun, W. Chen, L. Wang, X. Liu, and T.-Y. Liu. On the depth of deep neural networks: A theoretical view. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 2066–2072. AAAI Press, 2016.

[65] R. S. Sutton and A. G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998.

[66] L. Tan, Z. Zhu, F. Ge, and N. Xiong. Utility maximization resource allocation in wireless networks: Methods and algorithms. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(7):1018–1034, July 2015.

[67] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12):1936–1948, Dec 1992.

[68] L. Tassiulas and A. Ephremides. Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, 39(2):466–478, March 1993.

[69] D. Tse and P. Viswanath. *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[70] Q. Wang, P. Fan, and K. B. Letaief. On the joint v2i and v2v scheduling for cooperative vanets with network coding. *IEEE Transactions on Vehicular Technology*, 61(1):62–73, 2012.

[71] C. Wiesner, M. Ruf, D. Sirim, and G. Klinker. Visualisation of the electronic horizon in head-up-displays. pages 87–89, 09 2016.

[72] Wikipedia. Cross entropy. 20 June 2020. `https://en.wikipedia.org/wiki/Cross_entropy`.

[73] Wikipedia. Sorensen dice coefficient. 28 July 2020. `https://en.wikipedia.org/wiki/S%C3%B8rensen%E2%80%93Dice_coefficient`.

[74] Wikipedia. Huber loss. 29 May 2020. `https://en.wikipedia.org/wiki/Huber_loss`.

[75] Wikipedia. Supervised learning. 5 July 2020. `https://en.wikipedia.org/wiki/Supervised_learning#How_supervised_learning_algorithms_work`.

[76] Wikipedia. Activation function. 7 July 2020. `https://en.wikipedia.org/wiki/Activation_function`.

[77] S. Xu, G. Zhu, C. Shen, Y. Lei, and Z. Zhong. Utility-based resource allocation in high-speed railway wireless networks. *EURASIP Journal on Wireless Communications and Networking*, 2014(1):68, Apr 2014.

[78] H. Ye, G. Y. Li, and B. F. Juang. Deep reinforcement learning based resource allocation for v2v communications. *IEEE Transactions on Vehicular Technology*, 68(4):3163–3173, 2019.

[79] Y. Yi and M. Chiang. Stochastic network utility maximisation—a tribute to kelly's paper published in this journal a decade ago. *European Transactions on Telecommunications*, 19(4):421–442, 2008.

[80] H. Zhou, P. Fan, and J. Li. Global proportional fair scheduling for networks with multiple base stations. *IEEE Transactions on Vehicular Technology*, 60(4):1867–1879, May 2011.