DISSERTATION

In Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
from Sorbonne Universités

Specialization

Communication Systems

Ècole Doctorale Informatique, Télécommunications et Èlectronique (Paris)

Presented by

Nikolaos LIAKOPOULOS

Machine Learning Techniques for Online Resource Allocation in Wireless Networks

Defense scheduled on July 8th 2019

before a committee composed of:

Dr. E. Veronica Belmega	*Associate Professor*	Examiner
Dr. Georgios Paschos	*Principal Researcher*	Industrial Supervisor
Dr. Leandros Tassiulas	*Professor*	Reviewer
Dr. Marceau Coupechoux	*Professor*	Examiner
Dr. Navid Nikaein	*Professor*	Thesis Supervisor
Dr. Thrasyvoulos Spyropoulos	*Associate Professor*	Academic co-Supervisor
Dr. Urtzi Ayesta	*Director of Research*	Reviewer

Abstract

Traditionally, network optimization is used to provide good configurations in real network system problems based on mathematical models and statistical assumptions. Recently, this paradigm is evolving, fueled by an explosion of availability of data. The modern trend in networking problems is to tap into the power of data to extract models and deal with uncertainty. This thesis is targeting on augmenting the arsenal of algorithms against specific network problems, by proposing algorithmic frameworks for wireless networks, based both on *classical* or *data-driven optimization* and *machine learning.*

In wireless networks, application of optimization is gaining momentum both in practice and research, following the wireless communication proliferation and the need for greatly enhanced resource utilization efficiency. Demand for wireless data is increasing exponentially, networks are becoming extremely dense in devices and cells, services come with extreme and diverse QoS requirements. Current network architecture which is based on one-type-fits-all services and spectrum re-use is already becoming unprofitable.

As a consequence, resource provisioning in wireless networks is evolving in a very challenging problem. This can be briefly accounted to the following: i) high spatiotemporal variation of demand, ii) high dimension of optimization and iii) coupling of decisions. The challenge is to come up with optimization methods that are fast, scalable and quickly adapting to input changes; while being robust against fluctuations to guarantee the service requirements. We target two use cases, *user association* and *cloud resource reservation.*

The baseline approach for *user association*, connecting wireless devices to the base station that provides the strongest signal, leads to very inefficient configurations in current and future wireless networks. We focus on tailoring user association based on resource efficiency and service requirement satisfaction (QoS guarantees), depending on the underlying network demand.

First, we study the user association problem for two network services (chapter 2); one requiring QoS guarantees to VIP flows, and one best effort service. The goal is to take advantage of *statistical multiplexing* in order to optimize the use of resources, while ensuring that the VIP flows enjoy active *performance guarantees.* We formulate this as an optimization problem, show that the problem is convex, and finally demonstrate that the optimum point can in fact be realized by distributed user association rules.

In chapter 3 we tackle *user association* focusing on developing a central scalable algorithm. We steer wireless traffic to Cloud-Radio Access Networks (C-RANs) by designing device association rules, in order to load balance the network. To address the challenge of massive connectivity and the resulting computational bottleneck, we propose an approach based on the theory of *optimal transport*, which studies the economical transfer of probability between two distributions.

Further, we propose a data-driven framework for *user association* leveraging the theory of *robust optimization*, see in chapter 4. The main idea is to predict future traffic fluctuations, and use the predictions to design association maps before the actual arrival of traffic. Although the actual playout of the map is random due to prediction error, the maps are robustly designed to handle uncertainty, preventing constraint violations, and maximizing the expectation of a convex utility function, which is used to accurately balance base station loads. The optimal maps have the intriguing property that they jointly optimize the predicted load and the variance of the prediction error. We validate our robust maps in Milano-area traces with dense coverage.

Moving to the topic of *cloud resource reservation*, we develop a novel framework to handle resource reservation in worst-case scenaria, where the demand is engineered by an adversary aiming to harm our performance. We provide policies that have “no regret” and guarantee *asymptotic feasibility in budget constraints*, under such workloads, complementing the results of recent literature in cloud computing and more importantly in Online Convex Optimization (OCO).

In chapter 5 we propose a policy for *cloud resource reservations* that eventually learns the minimum cost reservation, while satisfying a *time-average constraint for violations.* This uses a combination of the Lyapunov optimization theory and a linear prediction of the future based on the recent past. We validate our policy on real cloud system traces.

Next, in chapter 6 we generalize the results of chapter 5, creating a framework for *online convex optimization problems with long-term budget constraints.* Problems like this arise naturally in networks as reliability guarantees or total consumption constraints. Our proposition is *cautious online Lagrangian descent* (COLD) for which we derive explicit bounds, in terms of both the incurred regret and the residual budget violations.

Finally, chapter 7 contains the conclusions of our work and our future goals and some preliminary results on the comparison of the proposed methods.

Acknowledgements

In this small page I would like to thank all of the people involved into making these years more enjoyable and much more than a didactic process. They were supporting on the tough moments and enhancing the best moments, making this experience in many ways better.

First, I would like to thank, George and Akis. They both share a great passion for work and science, but ultimately their heart is in guiding and teaching, qualities that makes them both great supervisors.

My good old and new friends in Paris, Athens and beyond, are bound to be included! I will refrain from mentioning them all, having made unexpectedly many and great friendships during these years, both with colleagues from work and outside. Special thanks go to Apostolos, Gregory and Nicoleta, for wasting most of my time. To George, Praxitelis, Ioanna, Mike and Electra, good friends since the undergrad that we all shared time in Paris and we will still share time wherever we are. To Joao, Michele, Ana and Lucie, who were always next doors. Last mentions but definitely not least, are Maria, Theo, Revekka and the late addition, Marina.

Finally and most importantly, I would like to thank my mother, Maria and my brother Panayiotis, for the unconditional love and support over the years and my late father, Petros for being my inspiration for this journey.

Contents

List of Figures

List of Tables

Acronyms

Here is the list of acronyms used in the text.

UA	User Association
UDN	Ultra Dense Network
SINR	Signal-to-Interference-plus-Noise Ratio
BS	Base Station
QoS	Quality of Service
NR	New Radio
BE	Best Effort
LTE	Long Term Evolution
PS	Processor Sharing
DPS	Discriminatory Processor Sharing
URLLC	Ultra-Reliable Low-Lattency Communication
MAC	Medium Access Control
DCUAA	Distributed Constrained User Association Algorithm
OT	Optimal Transport
C-RAN	Cloud-Radio Access Network
RRH	Remote Radio Head
CoMP	Coordinated MultiPoint
RUAM	Robust User Association Map
GRMA	Generalized Robust Map Algorithm
SLA	Service Layer Agreement

SARIMA	Seasonal AutoRegressive Integrated Moving Average model
LSTM	Long Short-Term Memory
PGD	Projected Gradient Descent
MCEL	Minimize the Cost of the Expected Load
MWCC	Minimize the Worst Case Cost
ACF	AutoCorrelation Function
CPU	Central Processing Unit
THOR	Time Horizon Online Reservations
OCO	Online Convex Optimization
SDN	Software-Defined Networking
VNF	Virtual Network Function
FTL	Follow-The-Leader
DPS	Drift Plus Penalty
DPLPS	Drift Plus Loss Plus Smoothness
CCDF	Complementary Cumulative Distribution Function
CPC	Cost Per Click
COLD	Cautious Online Lagrangian Descent
OGD	Online Gradient Descent
ONO	Online Network Optimization

Chapter 1

Introduction

1.1 Optimization in Wireless Networks

Network optimization is the combined theory of mathematical modeling, statistical analysis and mathematical optimization used on networks to select the best possible configuration, based on a criterion, over a set of available configurations. Application of optimization in networks is often prototypical and many common problems of operation research appear, see in [3]. Most of the network problems encountered in a real system can be reduced to these general problems, allowing the use of advanced methods and algorithms that have been developed through many years of research.

On the other hand, recently we witnessed an explosion in the availability of data. Massive amounts of data are now routinely collected in all businesses, including wireless network operators, cloud computing platforms or retailers. This has triggered a modern trend in networking problems; to tap into the power of available data to extract models and deal with uncertainty. This thesis is targeting on augmenting the arsenal of algorithms against targeted network problems, by proposing algorithmic frameworks for wireless networks, based both on classical or data-driven optimization and machine learning.

In wireless networks, application of optimization is gaining momentum both in practice and research, as it is envisioned to greatly enhance resource utilization efficiency. A straighforward application of optimization techniques on the existing network resources can minimize the cost of operation, for example by reducing power, increasing performance (e.g. maximize throughput, balance load or minimize delay) and can robustify service providing performance guarantees (Quality of Service). Alternatively, different optimization tools can be used for future network planning, expansion, sharing of resources between operators and pro-active resource installation or reservation.

The requirement for optimization is driven from the proliferation of wireless communications. Demand for wireless data is increasing exponentially [4], networks are becoming extremely dense in devices and cells [5–7], services come with extreme and diverse QoS requirements [8–10]. Current network architecture which is based on one-type-fits-all services and spectrum re-use (cell densification) is already becoming unprofitable [10–12].

The integration of optimization in future systems is enabled by recent technological advancements, which can be used to renew network architecture. The main enablers we identify is, i) Softwarization of network services – formally mentioned as Network Function Virtualization (NFV) and Software Defined Networking (SDN) and centralization of computing resources for networks – the Cloud-Radio Access Network (C-RAN) architecture. The combination of the two allows centralized management of resources (orchestration) and provide collective knowledge and computation power to solve central optimization problems. ii) Massive collection of data and advancement of data mining techniques. The data collected by the network centrally can give intuition about the system and user behavior and can be utilized by machine learning techniques to improve network efficiency.

Despite the recent evolution of network design and orchestration, resource provisioning in wireless network remains a very challenging problem. This can be briefly accounted to the following factors: i) *high spatiotemporal variation of demand* – wireless environment is rapidly evolving following human activity [13–16], ii) *high dimension of optimization* – the problems encountered are large-scale with millions of variables [10,17] and (iii) *coupling of decisions* – the network needs to be reconfigured following the evolution of demand, to retain efficiency [10,16,18]. The challenge is to come up with optimization methods that are fast, scalable and quickly adapting to input changes; while being robust against fluctuations to guarantee the service requirements.

The contributions of this thesis are separated into two main parts. The first part corresponds to chapters 2 and 3, where we focus on optimizing resource utilization of networks in real time; while in chapters 5 and 6 we consider proactive resource reservations. Meanwhile, chapter 4, serves as a bridge between the two parts, as it tackles an online problem with an offline data-driven solution. In the thesis, we present two main use cases: *user association* and *cloud resource reservation*, but the solutions presented can be applied to a plethora of problems, sometimes out of the scope of networking (prominent examples are chapters 4 and 6).

1.1.1 Use Cases

User Association

An important problem in Ultra Dense Networks (UDNs) is assigning wireless devices to serving base stations, this problem is referred to in the literature as user association problem [9,16–24]. In existing systems a user will choose to associate with the base station that provides the maximum received signal to interference ratio (MaxSINR) [9]. This baseline user

association rule does not account for traffic hot spots, and leads to load imbalances and performance deterioration. We consider techniques that focus on two major metrics – instantaneous achievable rate at the physical layer and base station (cell) load and our goal is to derive algorithms that achieve QoS guarantees over the volatile wireless environment.
Similar Problems:*(Generalized) Assignment, Flow Routing, Matching.*

Cloud Resource Reservation

A fundamental challenge in cloud computing is to reserve just enough resources (e.g. memory, CPU, and bandwidth) to meet application runtime requirements [25]. We desire reservations that accurately meet the requirements: resource over provisioning causes excessive operation costs, while under provisioning may severely degrade service quality, causing interruptions and real-time deployment of extra resources, which costs heavily [26]. The problem resembles the well-known *newsvendor model* [27], where we seek an inventory level that maximizes the vendor revenue versus a forecast demand. In cloud computing, however, the common assumption of demand predictability does not always hold. The profile of cloud resources exhibits highly non-stationary behavior, and prediction is very difficult, if not impossible. Furthermore, in the increasingly relevant scenario of edge computing, the workload is expected to vary quickly with geography, mobility, and user application trends, and therefore its fluctuations will be even more unpredictable. We consider a model-free online reservation framework for cloud computing using ideas from machine learning.
Similar Problems:*Portfolio Optimization, Advertisement Placement, Volatile Resource Market, Inventory Optimization.*

1.2 Contributions and Outline of the Thesis

1.2.1 Contributions and Outline

The chapters of the thesis, and the main contributions in each of them, are organized as following:
Chapter 2 - Distributed User Association with Quality of Service Guarantees

We begin by studying the problem of *distributed user association* in Ultra Dense Networks (UDNs) for two network services; one requiring QoS guarantees to VIP flows, and one best effort service. The goal is to take advantage of *statistical multiplexing* in order to optimize the use of resources, while ensuring that the VIP flows enjoy active *performance guarantees.*

Distributed approaches are very common in the literature and they offer great scalability and low computation and communication overhead [18–23]. We base our model on the framework of [19], enriched by introducing a requirement-based class of flows (VIP). We prove that by constraining the amount of load of VIP flows on a base station, while giving strict priority to the privileged flows, the average number of VIP users in the selected base stations' queue will be bounded; this readily translates to a threshold on the average response time per flow.

We formulate the formerly described user association as an optimization problem, show that it is convex and use partial lagrangian relaxation to derive the new distributed association rules per class. From the partial relaxation a new metric appears, the lagrangian multiplier, which reflects the price of joining the base station due to excess VIP flow load. Using the derived association rules on a running average estimation of the actual base station load (both VIP and BE) and the lagrangian multiplier (VIP "price") the associations converge iteratively to the optimal assignment. In this way, this framework uses the fundamental problem of user association to achieve *isolation* and *statistical multiplexing* in the context of UDNs, when different services or slices must share BS resources, as envisioned in 5G networks.

In the numerical section we demonstrate no violations of the VIP flow constraint on real data traces for mobile network traffic, while a baseline best effort distributed policy applied to this setup inflicts up to 46.5% violations. Hence, we conclude that in a slowly changing environment, where the empirical load and the prices converge to the optimal, the derived distributed algorithm can provide QoS while efficiently utilizing the network resources. On the other hand, when traffic is fluctuating rapidly, it can lead to constantly changing association decisions and suboptimal configurations. In the following chapter, we will counter these drawbacks by designing a centralized scalable algorithm for association decisions.

The work on this chapter has been published in the following paper:

- *N. Liakopoulos, G. Paschos, Thr. Spyropoulos, "User Association for Wireless Network Slicing with Performance Guarantees", IEEE GLOBECOM, Abu Dhabi, UAE, December 2018*

Chapter 3 - Centralized Scalable User Association based on Computational Optimal Transport

In this chapter, we study the problem of connecting mobile traffic to Cloud-Radio Access Network (C-RAN) stations. The centralized approach of this chapter is motivated by the upcoming 5G wireless network architecture. The architecture of C-RAN economizes computation and signal processing by migrating the computing part of base stations to a central cloud location. Having the intelligence moved to the C-RAN controller enables, differently than the previous chapter, a stable and well informed association decision for the wireless devices.

However, the massive scale of future wireless networks causes a computational bottleneck in performance optimization. In practice, a general purpose linear programming solver will fail to solve an optimization problem of this scale, which will have millions of variables [10, 17]. To address the challenge of massive connectivity, we propose an approach based on the theory of *optimal transport* [28–30], which studies the economical transfer of probability between two distributions. The proposed methodology could further inspire scalable algorithms for large-scale optimization problems in wireless networks.

Although user association is not an optimal transport problem, we observe that by choosing the marginal distributions based on a target load solution, we can very quickly derive the association rules using reguralized OT. With this observation in mind, we propose an iterative algorithm that adjusts the target load and efficiently steers traffic away from overloaded base stations reducing the overall delay. In the numerical section we show the time efficiency of computational optimal transport against a standard linear solver in experiments with different scales and the delay reduction achieved in simulation scenaria by our iterative algorithm.

We note that, this chapter's method relies on a good estimate of the current traffic and into the ability of the network to compute and update the network configuration in real time. If this decision update process is slower than the change of status in the system (in the meantime new users appear, traffic volume changes, etc), the new configurations will be stale and suboptimal and will fail the QoS guarantees. In the following, we will present a pro-active optimization method, that will not require knowledge of the demand and real time configurations, but will extract the demand pattern and compute robust associations based on collected data.

The work on this chapter has been published in the following paper:

- *G. Paschos, N. Liakopoulos, Mérouane Debbah, Tong Wen, "Computational Optimal Transport for 5G Massive C-RAN Device Association", IEEE GLOBECOM, Abu Dhabi, UAE, December 2018*

Chapter 4 - Data-Driven User Association based on Robust Optimization

In chapter 4, we study the *user association* problem with QoS guarantees, in the context of UDNs, where traffic demand rapidly evolves in space and time following human activity. Here, we propose the association decisions to be computed at the central intelligence of the wireless network, the C-RAN, based on historical data about telecommunication activity that have been collected and processed by the network. The framework introduced here, instead of monitoring and trying to follow the rapidly-changing status of the network, as in chapter 3, it takes decisions based on the predicted traffic demand based on the data.

Including the distributed method presented in chapter 2, a number of recent works formalize the QoS user association problem and attempt to find an optimal solution, [9, 18–24]. Nevertheless, these frameworks are ineffective for this setup for two main reasons:

- *Spatio-temporal variability*: Due to smaller user/site ratios [5,7], the traffic demand will vary significantly more over time and space, giving rise to unpredictable traffic spikes [9].
- *Increased QoS requirements*: With the rise of vertical applications, 5G networks are expected to support slices that provide guaranteed Quality of Service (QoS) [9]. Unexpected traffic spikes combined with dynamic association decisions reacting to them, might lead the system to oscillations, instability, and violation of QoS requirements.

Instead, we propose a novel data-driven technique leveraging the theory of robust optimization [31]. The main idea is to predict future traffic fluctuations, and use the predictions to design association maps before the actual arrival of traffic. Although the actual playout of the map is random due to prediction error, the maps are robustly designed to handle uncertainty, preventing constraint violations, and minimizing the expectation of a convex cost function, which is used to accurately balance base station loads.

We propose a generalized iterative algorithm, referred to as Generalized Robust Map Algorithm (GRMA), which is shown to converge to the optimal robust map. The optimal maps have the intriguing property that they jointly optimize the predicted load and the variance of the prediction error. The algorithm admits the average load threshold cap as in chapter 2; while allowing a trade-off between being conservative and optimal (assuming expected traffic load), by having a configurable parameter tuning the probability of failing the constraint per base station. In addition the framework works with any convex separable cost function, which includes the commonly used α-optimal functions, seen in chapter 2 or [19,32].

Depending on the choice of the objective function, the derivative computation can be extremely complex, hence we propose methods to approximate the optimal map. In the numerical section, we validate our robust maps in Milano-area traces [33], with dense coverage and find that we can reduce violations from 25% (inflicted by a baseline adaptive algorithm) down to almost zero. Furthermore, we demonstrate the effect of α-objectives [19,32].

Finally, we discuss advanced time series forecasting methods, like Seasonal AutoRegression-Integration-Moving Average models (SARIMA [34]) and Long Short-Term Memory (LSTM [35]) neural networks. Using these methods on the data [33], we validate the gaussian estimator error model and improve the quality of the map produced by GRMA; decreasing the cost gap compared to an optimal map down to 5%.

The robust framework serves as an introduction to our resource reservation body of work later, as it could be used to economically reserve resources, while guaranteeing QoS. For example a routing map, computed by GRMA, could associate flows to a cloud, optimizing the expectation and variance of the cost of traffic load on the server. The map would give the routing decisions and indirectly the amount of resources (computation, memory, etc) required to be reserved. This framework though, depends on the predictability of future demand, which as described in the following chapters (5 and 6) is not always true for cloud workloads. For such applications, we will need a new framework that will adapt the reservations based on the changing distributions of the demand.

The work on this chapter has been published in the following articles:

- *N. Liakopoulos, G. Paschos, Thr. Spyropoulos, "Robust User Association for Ultra Dense Networks", IEEE INFOCOM, Honolulu, HI, April 2018*
- *N. Liakopoulos, G. Paschos, Thr. Spyropoulos, "Robust Optimization Framework for Proactive User Association in UDNs: A Data-Driven Approach", IEEE/ACM Transactions on Networking, August 2019*

Chapter 5 - Online Cloud Resource Reservation with Budget Constraints

Here, the *resource reservation* problem is introduced. Specifically, we study learning an economical yet *robust* resource reservation for cloud computing, i.e. reserve just enough resources meet application runtime requirements [25]. The goal is to have reservations that accurately meet the requirements: resource over-provisioning causes excessive operation costs, while under-provisioning may severely degrade service quality, causing interruptions and real-time deployment of extra resources, which costs heavily [26].

Assuming that demand is predictable the robust framework presented in chapter 4 could be applied, but often in cloud computing this common assumption does not hold. Our experimentation [36] in a Google cluster dataset, as well as other recent works [1], show that the profile of cloud resources exhibits highly non-stationary behavior, and prediction is very difficult, if not impossible. Instead, we propose a novel model-free approach that has its root in online learning. We allow the workload profile to be engineered by an adversary who aims to harm our decisions, and we investigate a class of policies that aim to minimize regret (minimize losses with respect to a static policy that knows the workload sample path).

Specifically, we formulate the problem of reserving resources for cloud computing as a constrained Online Convex Optimization (OCO) problem. At each slot, i) an online reservation policy decides a reservation vector, then ii) the adversary decides a demand vector, and last iii) a cost is paid for the reserved resources and a violation is noted if the demand was not covered by the reservation.

Regret minimization in the presence of adversarial constraints is a very difficult problem; in fact, a well-cited result in the literature is that of [37], which shows by a counter example that in certain cases it is impossible. Here, we slightly relax the benchmark of [37], and propose Time Horizon Online Reservations (THOR) a policy we prove to achieve asymptotic feasibility and "no regret". The performance guarantees of THOR are obtained by a novel combination of the Lyapunov K-slot drift technique [38] with the linearization idea of Zinkevich [39]. THOR inherits the simplicity of online gradient, and therefore is straightforward to implement in practical systems.

In the numerical section we validate THOR resource reservations using a public dataset provided by Google [2]. THOR vastly outperforms our implementation of the textbook Follow The Leader (FTL) policy in guaranteeing the violations constraint, while it achieves similar or sometimes better performance than the static oracle T-slot policy, in the challenging, non-stationary CPU workload.

In the next chapter, we will generalize THOR to a new framework for online learning and resource reservation with budget constraints. In this framework we strengthen the adversary to pick both the cost and constraint functions, we introduce the K-benchmark, a set of K-slot best cost K-slot feasible static policies, we generalize some of the base assumptions of prior work and we prove that our new policy, Cautious Online Lagrangian Descent (COLD), has "no regret" against K-benchmarks when $K = o(T)$.

The work on this chapter has been published in the following paper:

- *N. Liakopoulos, G. Paschos, Thr. Spyropoulos, "No Regret in Cloud Resources Reservation with Violation Guarantees", IEEE INFOCOM, Paris, FR, May 2019*

Chapter 6 - General Online Resource Reservation with Budget Constraints: A New Framework

In the final chapter of the main body of work, we study a class of *online convex optimization problems with long-term budget constraints* that arise naturally as reliability guarantees or total consumption constraints. This is a general setting, that subsumes the work presented in chapter 5.

In this general setting prior work by [37] has shown that achieving "no regret" is impossible if the functions defining the agent's budget are chosen by an adversary. To overcome this obstacle, we refine the agent's regret metric by introducing the notion of a "K-benchmark", i.e., a comparator which meets the problem's allotted budget over any window of length K.

The impossibility analysis of [37] is recovered when $K = T$; however, for $K = o(T)$, we show that it *is* possible to minimize regret while still meeting the problem's long-term budget constraints. We achieve this via an online learning algorithm based on *Cautious Online Lagrangian Descent* (COLD) for which we derive explicit bounds, in terms of both the incurred regret and the residual budget violations.

On the other end, when $K = 1$, recent work done in [40], has proven that a combination of OGD with a virtual queue can indeed provide "no regret", compared to a static action that is strictly feasible for all constraint functions g_t, i.e., a policy $x^\star$ must satisfy $g_t(x^\star) < 0$ for all $t = \{1, ..., T\}$.

They present improved bounds for this special case, however, especially in an adversarial setting, their strict feasibility assumption might not be easy to achieve. Even a small degree of residual constraint violation injected by the adversary could disqualify any comparator action. More importantly, since $x^\star$ is artificially constrained in this way, the obtained regret guarantee can be fairly loose.

Our theoretical findings are also validated by a series of numerical experiments which suggest that increasing K – that is, enlarging the window over which the budget must be balanced – makes the K-benchmark guarantee tighter. Hence, proving "no regret over K-benchmark" for large K results in tighter performance guarantees – an observation which is not a priori obvious in a bona fide adversarial setting.

The work on this chapter has been published in the following paper:

- *N. Liakopoulos, A. Destounis, G. Paschos, Thr. Spyropoulos, Panayotis Mertikopoulos, "Cautious Regret Minimization: Online Optimization with Long-Term Budget Constraints", to appear at ICML, Long Beach, CA, June 2019*

Chapter 7 - Conclusions and Future Research

We close the thesis with the conclusion and future research chapter. Here, we present in brief the main conclusions derived by the body of this work. We discuss the envisioned general optimization framework for networks and present some preliminary results derived in our study applying different optimization methods in network problems [41].

Chapter 2

Distributed User Association with Quality of Service Guarantees

2.1 Introduction

In order to cope with the rapid growth of data traffic demand, wireless operators move to ultra dense, heterogeneous deployments, also referred to as Ultra Dense Networks (UDNs) [5–7, 10]. These consist of many low-power small cells, to maximize spatial reuse of the available bandwidth, overlaid with macro-cells which ensure coverage. With ultra dense deployment, the problem of user association becomes increasingly important, but also highly complex. Naive SINR-based schemes [9, 42], like the ones commonly used in current wireless networks, can be highly suboptimal, failing to properly balance the load across the different base stations which is necessary to translate the denser Base Station (BS) deployment into actual Quality of Service (QoS) improvement.

Furthermore, new applications and service types are emerging that need to be carried over cellular networks with diverse QoS requirements. These, intensify the need for efficient resource utilization to cope with the conflicting demands of reduced latency or increased throughput [8]. These prerequisites can be effectively treated by breaking the one-type-fits-all-services scheme and considering application specific (flow) traffic steering and prioritizing, as envisioned in 5G New Radio (NR) [8]. However, it is far from clear how to optimally allocate the common resources between different services.

Consider for example a scenario with a “VIP” and a “Best Effort” (BE) service. The former requires low delay for its applications, while the latter could correspond to standard data and voice. Naively pre-allocating specific resources, e.g., part of the bandwidth or Resource Blocks (in LTE) to each service is suboptimal [43], as part of the resources might stay under-utilized, while others might become congested. On the other hand, while joint resource allocation leads to better statistical multiplexing gains, a surge of BE traffic may endanger the performance of the VIP service. *To address the challenge of isolation vs statistical multiplexing, this chapter considers the basic scenario with one VIP service and one Best Effort (BE). We propose a few modifications on a simple and well-established model for BS schedulers [44], and develop an analytical optimization framework with the following goals:*

1. Provide *service isolation*, i.e., Best Effort load should not have an impact on VIP performance.
2. Provide *performance guarantees* to the VIP flows, in terms of bounding the mean number of active VIP flows at specific base stations (this readily translates to per flow delay guarantees as well).
3. Provide *statistical multiplexing gains* to both services, by optimally allocating the resources between VIP and BE flows, subject to respecting the strict priority of the VIP.
4. Ensure that when all the above goals are satisfied, loads across BSs are configured to provide *optimal network-wide performance* (in terms of different tunable metrics).

As a final note, the proposed framework achieves the set goals, while being minimally invasive to existing schedulers. It introduces a second queue, but still applies the existing scheduling policy on each queue. For example, the framework can be extended to explicitly support delay-sensitive 5G service classes (e.g. URLLC [45]), as well as for resource allocation among multiple slices [11, 12, 43], corresponding to different operators or services. To extend our framework to multiple classes, a combination of priority queuing and discriminatory processor sharing [46] could be used. We defer this to future work.

2.1.1 Related Work and our Contribution

A number of more advanced schemes, beyond simple SINR-based, have been proposed to take advantage of the dense deployment of base stations and utilize all available resources [19, 22, 23]. These schemes aim to optimally balance different, often conflicting goals such as: giving each user a high enough rate, minimizing the average network-wide delay, and ensuring that no base station is congested. A seminal work in this direction is the framework of [19] that utilizes the α-optimal function to balance these goals, and derives optimal and distributed user association rules, assuming best effort flows. This work has since been extended to jointly optimize uplink and downlink traffic [20], consider backhaul constraints [21], energy efficiency [47], and

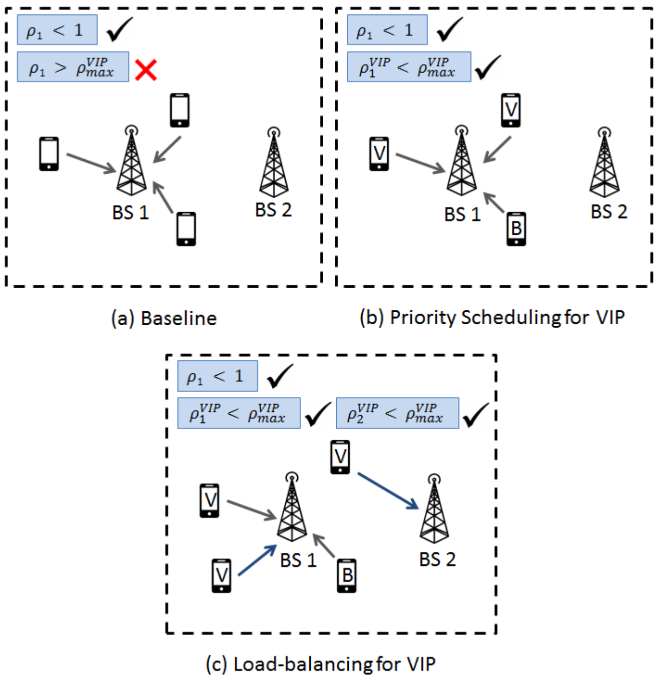

$\rho_1 < 1$ ✓
$\rho_1 > \rho_{max}^{VIP}$ ✗
BS 1
BS 2
(a) Baseline
$\rho_1 < 1$ ✓
$\rho_1^{VIP} < \rho_{max}^{VIP}$ ✓
V
B
BS 1
BS 2
(b) Priority Scheduling for VIP
$\rho_1 < 1$ ✓
$\rho_1^{VIP} < \rho_{max}^{VIP}$ ✓
$\rho_2^{VIP} < \rho_{max}^{VIP}$ ✓
V
B
BS 1
BS 2
(c) Load-balancing for VIP

Figure 2.1: Simplified example for different association policies

a number of other directions. In the context of slicing, the authors in [48] propose dynamic sharing of radio access network resources between (virtual) operators; while the aforementioned approach achieves capacity savings for the tenants, it cannot give performance guarantees for a class of applications (VIP).

Consider Fig.2.1 as a motivating example. A baseline association approach that does not considers flow differentiation (no priorities and no VIP constraints) will lead to high load for VIP flows in Fig.2.1a for BS 1 and low quality wireless service. Enhancing the network with a scheme that explicitly differentiates between VIP and BE flows and uses a scheduler that prioritizes VIP flows (as in Fig.2.1b), temporarily fixes the performance guarantee for VIP flows with no impact on total load. When, as shown in Fig.2.1c a 3rd VIP flow is added to BS 1 though, the QoS for BS 1 cannot be met due to the high concentration of VIP load. In this configuration, our proposed algorithm will switch one VIP flow to BS 2, achieving the VIP QoS for both base stations, with a trade-off cost of reduced SINR or instantaneous rate for the switched flow.

In our work we extend the user association framework of [19] and make the following contributions: (i) Different from [19], to ensure isolation we assume a priority-based MAC scheduler at each BS, where VIP flows are always served before BE flows. We show that such a BS scheduler can be modeled as a Processor Sharing (PS) queue with 2 priorities. (ii) We introduce a constraint on the VIP load at each base station, that is used to provide appropriate VIP performance guarantees. (iii) We derive novel distributed user association rules for both the VIP and BE services, that provably converge to optimize an α-optimal function of the *total* base station load, thus maximizing the statistical multiplexing gains within the feasible region. (iv) We show that our association policy outperforms both the original best effort policy of [19], and an improved version of the latter, adapted for the 2-class setup, using the original association rule but also giving priority to VIP flows, using telecom traces from the area of Milano [33].

2.2 Architecture

2.2.1 System Model

Spatial traffic. We consider a region $\mathcal{L} \in \mathbf{R}^2$ with coverage from $\mathcal{B}$ (heterogeneous) base stations. At each point $x \in \mathcal{L}$, users generate flow requests according to an inhomogeneous Poisson point process with spatial intensity $\lambda(x)$ and have independently distributed file sizes with mean $\frac{1}{\mu(x)}$.

Service Rate. If users at point $x \in \mathcal{L}$ are associated to base station $i \in \mathcal{B}$, then their flows are served with rate $C_i(x)$. In our model, $C_i(x)$ will be a location-dependent metric that depicts the wireless signal degradation due to distance. One might be tempted to include in $C_i(x)$ channel fading and dynamic interference, which depends on user association decisions and power control, however not considering these phenomena is common in user association literature [19, 20, 22, 49], next we give a short justification. Fast fading is averaged out due to the time scale of association being large compared to the time of channel coherence. Furthermore, since we are considering a low mobility environment the result of slow fading or shadowing can be captured from SINR. Finally, if SINR is measured based on reference pilot signals emitted by all base stations simultaneously the measurement produces approximately equivalent results with the assumption that nearby base stations are saturated. This assumption is common in most related work [19–22, 44].

Here we give a specific model for $C_i(x)$, with the understanding that other models that are location-dependent are admissible in our work:[1]

$$C_i(x) = W \log(1 + (\mathrm{SINR}_i(x))), \tag{2.1}$$

where W is the available frequency band, and $\mathrm{SINR}_i(x)$ is given by:

$$\mathrm{SINR}_i(x) = \frac{P_i G_i(x)}{\sum_{j \neq i} P_j G_j(x) + N_0}.$$

[1]We clarify that despite the particular modeling of $C_i(x)$, the association decisions remain coupled through the base station loads that affect the performance received at each point.

Above, P_i denotes the transmission power of base station i, N_0 denotes noise power and $G_i(x)$ is the path loss between the antenna and the UE. In our analysis, powers are static.

Flow Differentiation. Depending on service requirements the flows are classified into VIP (V) and Best Effort (B). The overarching goal is to provide VIP flows with better service quality, which we will achieve via a two step scheme: (i) the service of VIP flows is strictly prioritized over Best Effort (thus they only compete for service with other VIP flows), and (ii) the load of VIP flows is regulated at each base station.

Association Rules. Let $\pi_i^V(x) = \{0, 1\}$ and $\pi_i^B(x) = \{0, 1\}$ be the association rules, indicating if point x flow of type V, B is associated with base station i. At each time instance we enforce the constraint that points are uniquely associated to one base station, hence $\sum_{i \in \mathcal{B}} \pi_i^T(x) = 1$, where $T = V, B$. The association variables $\pi_i^B(x), \pi_i^V(x), \forall x \in \mathcal{L}$ will be the means to control the performance of the system.

Base Station Load. The fraction of time required to deliver the traffic load destined to location $x \in \mathcal{L}$ by base station i is defined as the load density for i in x:

$$\varrho_i^T(x) = \frac{\lambda^T(x)}{\mu^T(x) C_i(x)}, \qquad T = V, B.$$

The fraction of time a base station i is busy, called the load ρ_i, due to a specific type of traffic V or B is given by:

$$\rho_i^T = \int_{\mathcal{L}} \varrho_i^T(x) \pi_i^T(x) dx, \quad T = V, B. \tag{2.2}$$

The total load of base station i is then

$$\rho_i = \rho_i^V + \rho_i^B.$$

The base station load vector $\boldsymbol{\rho} = (\rho_i)$ is an important performance metric of the system. In subsection Sect.2.2.3 we will explain how we can choose association rules $\pi_i^B(x), \pi_i^V(x), \forall x \in \mathcal{L}$ to control $\boldsymbol{\rho}$ and ultimately provide differentiated levels of service to different user flow types.

2.2.2 Queue Delay Model

Even though the exact dynamics of the LTE schedulers are not standardized, it is generally accepted that in practice a proportional-fair scheduling policy ($\alpha \approx 1$) is used [22, 42, 50]. In this case and most general case of temporal fair schedulers, the dynamics of the base station queues are captured by a multi-class $M/G/1$ processor sharing system [44]. A well known result for Processor Sharing [51] is that the stationary distribution of the number of customers is insensitive to the distribution of service times, hence assuming

$\rho_i < 1$ we have the following: The stationary distribution for the total number of flows in base station i is:

$$q_i(n) = \rho_i^n(1-\rho_i).$$

By the above equation we can show that the mean number of active flows in BS i is:

$$E[N_i] = \frac{\rho_i}{1-\rho_i}.$$

By Little's Law the Expected Delay (Response time) in Queue i is:

$$E[D_i] = \frac{1}{\lambda_i}E[N_i] = \frac{1}{\lambda_i}\frac{\rho_i}{1-\rho_i}.$$

where the incoming arrivals at the queue are

$$\lambda_i = \int_{\mathcal{L}} (\lambda^V(x)\pi_i^V(x) + \lambda^B(x)\pi_i^B(x))dx.$$

Proposition 1 (Isolation and Performance Guarantee)**.** *By considering that the base stations also follow a preemptive priority scheduling policy in favor of the VIP flows, all of the above equations can be rewritten for ρ_i^V. By limiting the VIP load $\rho_i^V \le c_i$ we get the upper bounds:*

$$E[N_i^V] \le \frac{1}{1-c_i} - 1,$$
$$E[D_i^V] \le \frac{1}{\lambda_i}(\frac{1}{1-c_i} - 1).$$

Therefore, in order to guarantee a certain average delay performance, below we optimize $\boldsymbol{\rho}$ subject to ensuring the constraint $\rho_i^V \le c_i$ at each base station.

2.2.3 Dual Service Problem Formulation

First, we relax the integrality requirement of the association decisions. In practice, a fractional association could be interpreted as a time sharing of different integral associations, hence different jobs may be handled by different base stations. With this relaxation, a feasible association vector $\boldsymbol{\pi} = \{\boldsymbol{\pi}^V, \boldsymbol{\pi}^B\}$ assigns each layer of flows at location in $x \in \mathcal{L}$ with a probability $\pi_i(x) \in [0,1]$ at base station $i \in \mathcal{B}$ such that all the flow requests are served and ensures that all the base stations are stable ($\rho_i < 1$, $\forall i \in \mathcal{B}$). Since we are also targeting performance guarantees for the service of VIP flows, we impose a threshold on the load of VIP flows per base station $\rho_i^V \le c_i$ (Proposition 1).

Definition 1. *$\mathcal{F}$ is the convex set of flow differentiated feasible load vectors $\boldsymbol{\rho}^B, \boldsymbol{\rho}^V$:*

$$\begin{aligned}\mathcal{F} = \{\boldsymbol{\rho} \mid \rho_i = &\int_{\mathcal{L}} \left(\varrho_i^V(x)\pi_i^V(x) + \varrho_i^B(x)\pi_i^B(x)\right) dx \\ &0 \le \rho_i \le 1-\epsilon, \quad \forall i \in \mathcal{B} \\ &0 \le \rho_i^V \le c_i, \ \forall i \in \mathcal{B} \\ &\sum_{i\in\mathcal{B}} \pi_i^T(x) = 1, \ \forall x \in \mathcal{L}, \ T = V, B \\ &0 \le \pi_i^T(x) \le 1, \ \forall i \in \mathcal{B}, \ \forall x \in \mathcal{L}, \ T = V, B\},\end{aligned}$$

where c_i is the VIP load threshold for base station i.

Lemma 1. *The feasible set $\mathcal{F}$ is convex.*

Proof. Consider vectors $\boldsymbol{\rho}^1, \boldsymbol{\rho}^2 \in \mathcal{F}$ and $\boldsymbol{\rho}^1 \neq \boldsymbol{\rho}^2$. Let $\boldsymbol{\rho} = \theta\boldsymbol{\rho}^1 + (1-\theta)\boldsymbol{\rho}^2$ with $\theta = [0, 1]$. We will show that $\boldsymbol{\rho} \in \mathcal{F}$. The elements of vector $\boldsymbol{\rho}$ are $\rho_i = \theta\rho_i^1 + (1-\theta)\rho_i^2$ and thus

$$\begin{aligned}\rho_i = \theta &\int_{x\in\mathcal{L}} (\varrho_i^V(x)\pi_i^{1V}(x) + \varrho_i^B(x)\pi_i^{1N}(x))dx \\ &+ (1-\theta)\int_{x\in\mathcal{L}} (\varrho_i^V(x)\pi_i^{2V}(x) + \varrho_i^B(x)\pi_i^{2N}(x))dx \\ \rho_i = &\int_{x\in\mathcal{L}} \varrho_i^V(x)(\theta\pi_i^{1V}(x) + (1-\theta)\pi_i^{2V}(x))dx \\ &+ \int_{x\in\mathcal{L}} \varrho_i^B(x)(\theta\pi_i^{1N}(x) + (1-\theta)\pi_i^{2N}(x))dx.\end{aligned}$$

Considering $\pi_i^V(x) = \theta\pi_i^{1V}(x) + (1-\theta)\pi_i^{2V}(x)$ and $\pi_i^B(x) = \theta\pi_i^{1B}(x) + (1-\theta)\pi_i^{2B}(x)$, we may check that vector $\boldsymbol{\rho}$ satisfies all the equations in $\mathcal{F}$, thus $\boldsymbol{\rho}$ is feasible and $\mathcal{F}$ is convex. □

In the optimization scope of the user association problem, the objective is to select from the feasible vectors $\mathcal{F}$ the vector that optimizes a selected network performance. As we have defined in section 2.2.2, we would like to optimize for some metric defined by $\boldsymbol{\rho}$, the load vector of the base station queues. For this purpose we select the α-optimal objective functions.

Definition 2. *The α-optimal functions, for $\alpha \in [0, \infty)$ are:*

$$\phi_\alpha(\boldsymbol{\rho}) = \begin{cases} \sum_i \frac{(1-\rho_i)^{1-\alpha}}{\alpha-1} & \alpha \neq 1 \\ \sum_i \log(\frac{1}{1-\rho_i}) & \alpha = 1 \end{cases}$$

The optimization of $\phi_\alpha(\boldsymbol{\rho})$ shares some analogies with the commonly used in resource allocation optimization family of α-fair functions. It is shown in [19] that selecting: (i) $\alpha = 0$ maximizes the throughput of the system, (ii) $\alpha = 1$ maximizes the geometric mean of base station's idle time (1-ρ_i), (iii) $\alpha = 2$ minimizes the average number of flows in the base station queues, and (iv) $\alpha \to \infty$ leads to max-min load fairness described in [32]. We can now formulate our primal problem.

Problem 1 (P1: The Service Differentiation User Association Problem)**.**

$$\underset{\boldsymbol{\rho} \in \mathcal{F}}{\textit{minimize}}\ \phi_\alpha(\boldsymbol{\rho}) = \sum_{i \in \mathcal{B}} \frac{(1-\rho_i)^{1-\alpha}}{\alpha - 1}. \tag{2.3}$$

Since $\phi_\alpha(\boldsymbol{\rho})$ is a convex function and $\mathcal{F}$ is a convex set, P1 is a convex optimization problem. Below we exploit the convexity of P1 to design an algorithm that finds the optimal solution $\boldsymbol{\rho}^\star$ in a distributed manner. We will show that our algorithm yields integral association rules that converge to the optimal solution of P1, hence it also solves the integral counterpart of P1.

2.3 Distributed Constrained User Association

In this section we will solve P1 in a distributed fashion by using the theory of Lagrangian relaxation for constrained convex optimization. Initially, we relax the box VIP load constraint allowing its violation at a price γ_i. This will allow us to derive the optimal association rules for given user class, $x \in \mathcal{L}$, $\boldsymbol{\gamma}$ prices and $\boldsymbol{\rho}$ vector of loads. The derived rules can be used to iteratively solve the user association problem for fixed $\boldsymbol{\gamma}$. We will then show how to update price vector $\boldsymbol{\gamma}$ in order to converge to the optimal solution of P1. The full algorithm in steps is presented in subsection 2.3.4.

2.3.1 Partial Lagrangian Relaxation

After relaxing the load constraint the feasible load vectors are $\boldsymbol{\rho} \in \mathcal{F}'$, where $\mathcal{F}'$ allows $\boldsymbol{\rho}^V \in [0, 1)$ and the objective function is the partially relaxed Lagrangian:

$$\Phi_\alpha(\boldsymbol{\rho}, \boldsymbol{\gamma}) = \sum_{i \in B} \frac{(1-\rho_i)^{1-\alpha}}{\alpha - 1} + \sum_{i \in B} \gamma_i(\rho_i^V - c_i). \tag{2.4}$$

Problem 2 (P2: Relaxed User Association Problem)**.**

$$\underset{\boldsymbol{\gamma} \geq 0}{\textit{maximize}} \left\{ \underset{\boldsymbol{\rho} \in \mathcal{F}'}{\textit{minimize}} \{\Phi_\alpha(\boldsymbol{\rho}, \boldsymbol{\gamma})\} \right\}. \tag{2.5}$$

We will prove that the solutions of the relaxed problem (P2) will be primal optimal and primal feasible.

2.3.2 Optimal User Association Rules

Lemma 2. *If P2 is feasible, the optimal association decision for each user is given by the following rule, depending on user type:*

$$\pi_i^{\star V}(x) = 1\left\{ i^V(x) = \underset{j\in B}{argmax}\left\{ \frac{C_j(x)(1-\rho_j^\star)^\alpha}{1+\gamma_j^\star(1-\rho_j^\star)^\alpha} \right\}\right\}, \tag{2.6}$$

$$\pi_i^{\star B}(x) = 1\left\{ i^B(x) = \underset{j\in B}{argmax}\{C_j(x)(1-\rho_j^\star)^\alpha\}\right\}, \tag{2.7}$$

where $\rho_j^\star$ and $\gamma_j^\star$ are an optimal load and price vector of the problem above.

Proof. (Optimality of $\boldsymbol{\pi}^{\star V}, \boldsymbol{\pi}^{\star B}$ given $\boldsymbol{\rho}^\star, \boldsymbol{\gamma}^\star$). We have by using Eq.(2.2) and Eq.(2.4):

$$\begin{aligned}
\langle \nabla_{\boldsymbol{\rho}} \Phi_\alpha(\boldsymbol{\rho}^\star)\cdot \Delta\boldsymbol{\rho}^\star\rangle &= \sum_{i\in B}\left(\frac{\partial \Phi_\alpha}{\partial \rho_i^V}\Delta\rho_i^{\star V} + \frac{\partial \Phi_\alpha}{\partial \rho_i^B}\Delta\rho_i^{\star B}\right) \\
&= \sum_{i\in B}\left(\frac{1}{(1-\rho_i^\star)^\alpha} + \gamma_i^\star\right)(\rho_i^V - \rho_i^{\star V}) \\
&\quad + \sum_{i\in B}\left(\frac{1}{(1-\rho_i^\star)^\alpha}\right)(\rho_i^B - \rho_i^{\star B}) \\
&= \int_{\mathcal{L}} \varrho^V(x) \sum_{i\in B} \frac{1+\gamma_i^\star(1-\rho_i^\star)^\alpha}{C_i(x)(1-\rho_i^\star)^\alpha}\left(\pi_i^V(x) - \pi_i^{\star V}(x)\right)dx \\
&\quad + \int_{\mathcal{L}} \varrho^B(x) \sum_{i\in B} \frac{1}{C_i(x)(1-\rho_i^\star)^\alpha}\left(\pi_i^B(x) - \pi_i^{\star B}(x)\right)dx,
\end{aligned}$$

since $\pi_i^{\star T}(x)$ satisfy Eq.(2.6) and Eq.(2.7):

$$\sum_{i\in B}\frac{1+\gamma_i^\star(1-\rho_i^\star)^\alpha}{C_i(x)(1-\rho_i^\star)^\alpha}\pi_i^V(x) \geq \sum_{i\in B}\frac{1+\gamma_i^\star(1-\rho_i^\star)^\alpha}{C_i(x)(1-\rho_i^\star)^\alpha}\pi_i^{\star V}(x),$$

$$\sum_{i\in B}\frac{\pi_i^B(x)}{C_i(x)(1-\rho_i^\star)^\alpha} \geq \sum_{i\in B}\frac{\pi_i^{\star B}(x)}{C_i(x)(1-\rho_i^\star)^\alpha}.$$

Hence, the first order convex optimality criterion is met [52]:

$$\langle \nabla_{\boldsymbol{\rho}} \Phi_\alpha(\boldsymbol{\rho}^\star)\cdot \Delta\boldsymbol{\rho}^\star\rangle \geq 0,$$

and $\boldsymbol{\pi}^{\star V}, \boldsymbol{\pi}^{\star B}$ are optimal association vectors. □

The association rules produce two association maps[2], one for each service, where a user at location $x \in \mathcal{L}$ is associated to base station $i \in \mathcal{B}$ if this

[2]An optimal map at a time instance assigns the total traffic of a location of the grid $\mathcal{L}$ to the optimal serving base station i. Since, we differentiate how we assign locations based on flow class, we have two Association Maps, one for BE traffic and one for VIP. The maps are direct represenations of their respective association vectors $\boldsymbol{\pi}^{B\star}(x), \boldsymbol{\pi}^{V\star}(x)$.

is prescribed by the corresponding map. The aggregate of all the assigned load densities to the serving Base Station is equivalent to the optimal $\rho_i^\star$. Moreover, the optimal association rules Eq.(2.6) and (2.7) are deterministic, which proves that the optimal values for $\pi_i(x)$ are integral and that there is no loss of accuracy due to the continuous relaxation. More intuition about the proof is given in [19].

Starting from an initial load vector $\boldsymbol{\rho}^{(0)}$ we can iteratively find the $\boldsymbol{\rho}$ that minimizes Φ for fixed $\boldsymbol{\gamma}$, by using the association rules we derived. The process is described in steps in 2.3.4. In the following subsection we will show how to update the prices $\boldsymbol{\gamma}$ to reach the primal optimal $\boldsymbol{\rho}^\star$.

2.3.3 Maximization Method

The maximization step of P2 depends on the selection of the α-objective. By selecting $\alpha > 0$ the partially relaxed Lagrangian is differentiable and we can solve the master problem by gradient method. When $\alpha = 0$, then we use a subgradient method for the maximization.

2.3.3.1 Gradient Ascent for $\alpha > 0$

The Hessian matrix of our α-optimal cost function Eq.(4.26) for $\alpha > 0$ is positive definite, thus is strictly convex. By Proposition 6.1.1 and Appendix B of Nonlinear Programming [53], since the load constraint function is linear, the set $\mathcal{F}$ is convex and compact (closed, bounded and subset of R^n) and ϕ is strictly convex, the minimized partial Lagrangian function is differentiable and

$$\nabla_{\boldsymbol{\gamma}} \Phi_\alpha(\boldsymbol{\rho}^\star, \boldsymbol{\gamma}) = \boldsymbol{\rho}^{\star V} - c.$$

The gradient ascent algorithm will update the prices $\boldsymbol{\gamma}$ of the outer problem iteratively:

$$\boldsymbol{\gamma}^{(k+1)} = \boldsymbol{\gamma}^{(k)} + s^{(k)} \nabla_{\boldsymbol{\gamma}} \Phi_\alpha(\boldsymbol{\rho}^{(k)}, \boldsymbol{\gamma}),$$

with a constant step size $s^{(k)} = c$.

2.3.3.2 Subgradient Method for $\alpha = 0$

The α-optimal cost function Eq.(4.26) is affine (non-strictly convex):

$$\phi_0(\boldsymbol{\rho}) = \sum_{i \in \mathcal{B}} \rho_i.$$

The minimized partial Lagrangian over $\boldsymbol{\rho} \in \mathcal{F}'$ is non-differentiable. The subgradient iteration is similar to the gradient but since the gradient may not exist, a subgradient $g^{(k)}$ is used instead. One of the subgradients of the minimized over $\boldsymbol{\rho}$ Lagrangian function is the load constraint function at a minimizer of the Lagrangian $\boldsymbol{\rho}^\star$. By selecting a sufficiently small step size it is shown that the subgradient method minimizes the distance to the optimal solution and converges to the optimal. The subgradient method will update the prices $\boldsymbol{\gamma}$ of the outer problem according to the iteration:

$$\boldsymbol{\gamma}^{(k+1)} = \boldsymbol{\gamma}^{(k)} + s^{(k)} g^{(k)}$$

with a subgradient step $s^{(k)} = \frac{1}{\sqrt{k}}$.

2.3.4 Distributed Constrained User Association Algorithm

Below, we present the Distributed Constrained User Association Algorithm (DCUAA) that solves P2 by combining the optimal user association rules described in Sect.2.3.2 and the maximization methods described in Sect.2.3.3. The algorithm is trigger based and will iterate until convergence is reached. The output will be an optimal association map for both flow types connecting location $x \in \mathcal{L}$ to serving base station $i \in \mathcal{B}$.

Distributed Constrained User Association Algorithm (DCUAA)

```
Iterate over t until convergence
Base Stations calculate $\gamma_i^{(t+1)} \leftarrow \left[\gamma_i^{(t)} + s^{(t)} \nabla_g \Phi_\alpha\right]^+$
Broadcast $\gamma_i^{(t+1)}$
    Iterate over k until convergence
    User at location $x \in \mathcal{L}$ calculates $\pi_i(x)$:
        $\pi_i^V(x) = 1\left\{ i^V(x) = \underset{j \in \mathcal{B}}{\operatorname{argmax}} \left\{ \frac{C_j(x)(1-\rho_j^{(k)})^\alpha}{1+\gamma_j^{(t+1)}(1-\rho_j^{(k)})^\alpha} \right\} \right\}$
        $\pi_i^B(x) = 1\left\{ i^B(x) = \underset{j \in \mathcal{B}}{\operatorname{argmax}} \left\{ C_j(x)(1-\rho_j^{(k)})^\alpha \right\} \right\}$
    Base Station $i \in \mathcal{B}$ measures utilization:
        $U_i^{(k)} = \min\left[\int_{\mathcal{L}} (\varrho_i^V(x)\pi_i^V(x) + \varrho_i^B(x)\pi_i^B(x))dx, 1-\epsilon\right]$
        $\rho_i^{(k+1)} = \beta\rho_i^{(k)} + (1-\beta)U_i^{(k)}$
    Broadcast $\rho_i^{(k+1)}$
```

Lemma 3. *For $\alpha > 0$, the algorithm presented above will converge on the optimal association maps for P1 ($\boldsymbol{\pi}^{V\star}, \boldsymbol{\pi}^{N\star}$).*

Proof. For $\alpha > 0$, the objective in P2 is strictly convex over $\boldsymbol{\pi}$, the hessian matrix is positive definite ($\nabla^2_{\boldsymbol{\pi}} \Phi > 0$). Hence, there exist a unique solution (association maps) to the dual problem, which is primal feasible and cost equivalent to the primal [52]. This completes the proof. $\square$

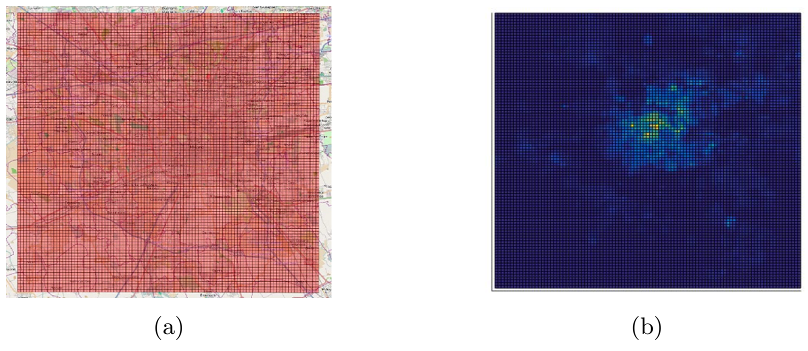


Figure 2.2: Milano Data Set. (a) Overview of Milano City Grid and (b) Color map of arrival intensity per square of the grid on a busy time instance

2.4 Numerical Evaluation

For the numerical evaluation of our algorithm we consider a simulation scenario that verifies the performance guarantees for the VIP flows achieved by the DCUAA for delay and also the VIP isolation from BE traffic. We use telecommunication traces from the Milano dataset [33], to confirm the results on real traffic input.

2.4.1 Simulation Setup

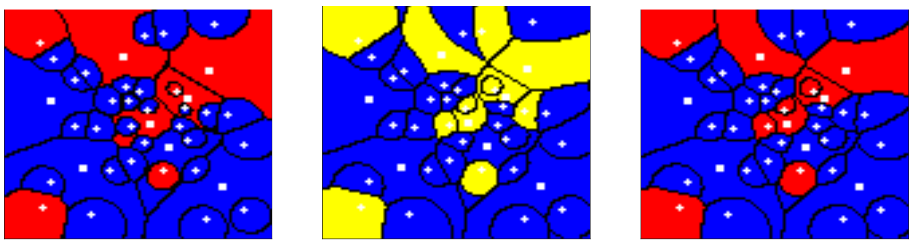
We assume that the user receives data at Shannon Capacity Eq.(2.1) and we model the propagation loss $G_i(x)$ with a path loss exponent 3:

$$G_i(x) = \left(\frac{1}{\text{dist}(\text{BS}_i \text{ to } x)} \right)^3 .$$

The LTE parameters for the transmission power of each base station tier and the transmission rate approximation are taken according to table 2.1, found also in [20].

Table 2.1: Simulation Parameters

Parameter	Variable	Value
Transmission Power Macro BS	P_M	43 dbm
Transmission Power Micro BS	P_m	29 dbm
System Bandwidth	W	10 MHz
Noise Density	No	-174dbm/Hz



(a) MaxSINR (b) DCUAA VIP (c) Kim et al VIP

Figure 2.3: Constraint violation comparison between the algorithms, red means violation, yellow is tight, blue means below threshold. (b) and (c) are calculated for $\alpha = 1$.

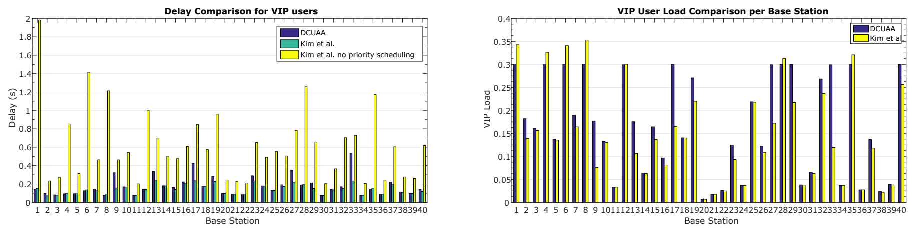
2.4.2 Validation of the results on the Milano dataset

To evaluate on algorithm on real telecommunication traffic traces, we will use the publicly available Milano dataset [33]. The Milano dataset provides spatially aggregated data about the telecommunication activity. The data are grouped on a regular grid overlaying the territory of Milano with 100×100 squares. Consequently, the grid designates the area $\mathcal{L}$ and every square is a location $x \in \mathcal{L}$ to be associated with base stations. For every square of this grid the data set contains the aggregate per ten minutes telecommunication events in the period of 01/11/13-01/01/14.

In the experiments we consider a weekday (Tuesday 3/12/2013) during peak traffic hour at midday. An overview of the Milano Grid and the midday arrival intensity per square of the grid can be see in Fig.2.2(a) and Fig.2.2(b), respectively. Since there is no priority differentiation for traffic in the dataset, we arbitrarily select 2, out of the 6, 10 min samples in an hour as VIP traffic. This is based on the fact that we expect VIP flow requests to manifest in (smaller) proportion to the expected total flow requests in an area.

We consider a two-tier Heterogeneous deployment of 40 Base Stations with fixed positions. The wireless network in the Milano area is simulated by 8 eNBs and 32 microcells, which are deployed with increased density in the area corresponding to the city center. We specifically design this subset of base stations, to accurately simulate a simplified environment of a dense deployment in a city, bringing in the front all the aspects of the user association problem.

We will show that our algorithm guarantees the performance of VIP flows and we will use as baseline, our implementation of the algorithm described in [19]. The association objective is set to maximize throughput ($\alpha = 1$) and the utilization cap for VIP flows is set to $c_{i\in\mathcal{B}} = 30\%$. Each of the plots in Fig.2.3 demonstrate location $\mathcal{L}$, the Base Stations are the white dots, squares are eNBs, circles are microcells, the black lines are the borders for the area



(a) VIP Delay per Base Station

(b) VIP Load per Base Station

Figure 2.4: Average Delay and Load metric comparison per base station

of service of each corresponding base station and colors indicate whether an area satisfies the performance constraint. Blue means below threshold, yellow means utilization is at the threshold, red means that utilization is above the requirement. In Fig.2.3(a), Fig.2.3(c) and Fig.2.3(b) we notice the areas of $\mathcal{L}$ that achieve the $\rho_i^V < c_i$ constraint, depending on the algorithm and user association policy selected.

A short summary of the most important results of this simulation can be found in table 2.2. In table 2.2 we can see that DCUAA achieves the set VIP performance guarantee, while not greatly degrading the overall performance in average delay of the total traffic. In contrast Kim et al. [19] best effort only algorithm does not achieve the VIP constraints for up to 46.5% of the incoming arrivals.

Finally, in Fig.2.4, we can see per base station, the average delay and load performance of the two algorithms in comparison. We can see that our algorithm effectively moves flows from the crowded base stations to less congested ones, with a trade off of reduced average delay on the congested base stations but slightly increased load in average.

2.4.3 Service Isolation

In order to demonstrate that our algorithm provides service isolation, which is an important aspect of future wireless network slicing architecture, we

Table 2.2: Simulation Results on the Milano dataset

Algorithm	Constr. Violation %	Av. Delay Tot (s)
DCUAA $\alpha = 1$	0	0.2575
Kim et al. $\alpha = 1$	46.5	0.2471
DCUAA $\alpha = 2$	0	0.2814
Kim et al. $\alpha = 2$	38.9	0.2874
DCUAA $\alpha = 5$	0	0.4404
Kim et al. $\alpha = 5$	9.5	0.4368

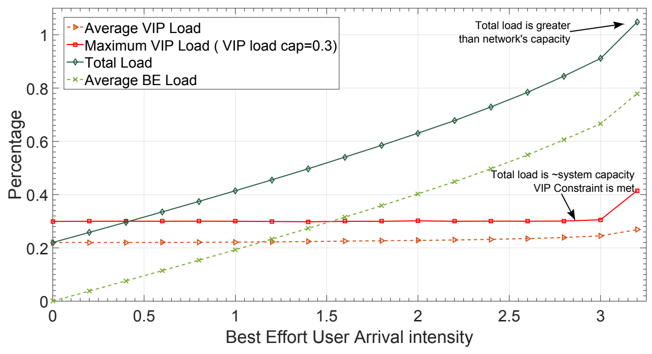


Figure 2.5: Service Isolation. Increasing BE arrival intensity, while not changing VIP arrival intensity, has no effect on VIP guaranteed performance inside the Feasibility Region.

successively scale up the arrival intensity of Best Effort users, from 0 up to the point that the average total incoming traffic exceeds the wireless network's capacity.

For each successive scaling of the BE input traffic we solve the optimal network maps (user association maps for VIP and BE flows) with the DCUAA. In Fig.2.5 we plot the best effort arrival intensity scaling factor and over all the base stations according to the optimal maps created by the DCUAA: (i) the average VIP load, (ii) the maximum VIP load (which has to be bellow the 0.3 load guarantee threshold), (iii) the total load of the system and (iv) the average BE effort load. From the Fig.2.5 it is clear that, the algorithm converges to configurations that guarantee the performance of the VIP flows, as long as the average total traffic is admissible.

2.5 Conclusion

In this chapter we have proposed a framework for user association based on distributed constrained optimization for 5G New Radio (NR) in the context of future Ultra Dense Networks (UDNs). We have derived distributed association rules, that provably converge to the optimum point of operation in a stationary environment. The resulting association guarantees performance to the VIP flows whilst balancing the load between both service types. The method is based on non-invasive extensions to current wireless networks, while it can be generalized in future work for multiple class priorities and applications in wireless network slicing. Our simulation results demonstrate the capabilities of the framework in bounding the mean number of VIP flows at the base stations and also the improved performance over BE only policies.

Hence, we conclude that in a slowly changing environment, where the empirical load and the base station admission prices (lagrangian multipliers) converge to the optimal, the derived distributed algorithm can provide QoS while efficiently utilizing the network resources. On the other hand, if traffic is evolving in a faster time scale than the distributed algorithm convergence, this can lead to instability and oscillations; constantly re-associating users and getting stuck to suboptimal configurations. In the following chapters, we will counter these drawbacks by designing a centralized algorithms for association decisions. In chapter 3 we will present a fast scalable centralized algorithm; while in chapter 4 we will show how to use data to create centrally Robust User Association Maps (RUAM). The RUAM presented in chapter 4 can be extended to multiple maps, one for each priority class of services, following the methodology of this chapter.

Chapter 3

Centralized Scalable User Association based on Computational Optimal Transport

3.1 Introduction

We revisit the problem of *device association* in the setting of *massive connectivity* and *ultra dense network deployment*, from the scope of online centralized decisions. In this problem, we seek to find a rule to associate mobile traffic to serving stations such that the incurred load is balanced across the available stations. Although a plurality of centralized association methodologies are available in the literature, here we focus on the underexplored aspect of scalability; we seek load-balancing associations for thousands of devices to hundreds of stations, a setting where even linear program solvers become cumbersome. To address the computational challenge, we propose to use *Optimal Transport* (OT) theory, which studies the transfer of masses over a metric space. Recently, an entropic regularization of OT was shown to provide super-fast algorithms for very large OT instances [54], making the framework applicable to a wide range of challenging applications, including image processing and machine learning.

Our centralized approach is motivated by the upcoming 5G wireless networks. According to recent reports, telecom operators are increasingly interested in deploying Cloud-Radio Access Network (Cloud-RAN, or C-RAN) systems, cf. [55]. The architecture of C-RAN economizes computation and signal processing by migrating the computing part of base stations to a central cloud location, and using simple *Remote Radio Heads* (RRH) to broadcast signals [56]. Since all the intelligence is now moved to the C-RAN controller, provisioning connectivity in a large geographical area is centrally

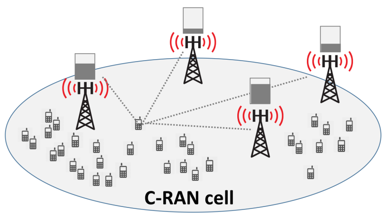


Figure 3.1: Devices associate to RRHs causing various load levels.

decided, motivating the centralized association of devices to RRHs.

A contemporary C-RAN controller manages a big number of RRHs and mobile devices, thus the centralized device association is inherently a large scale problem. Furthermore, the number of RRHs and of devices are both expected to increase in the near future. On one hand, the deployment of RRHs will become very dense to improve the effective capacity of the network [6, 7, 9, 10], while on the other hand, we expect a huge number of heterogenous smart IoT devices to connect to 5G mobile networks by 2020–estimated 20.4 billions in [57]. Since 5G applications can have radically diverse requirements, the traffic footprint of each connected device (IoT or regular) can vary significantly. *This motivates us to study the large-scale centralized device association with potentially different traffic requirements per device.*

3.2 The Device Association Problem

3.2.1 Downlink Model

We consider the downlink[1] transmissions of a large C-RAN cell, containing $\mathcal{I}$ devices and $\mathcal{J}$ RRHs. We call $\pi_{ij} \in [0, 1]$ the association variable, which is the fraction of time device $i \in \mathcal{I}$ connects to RRH $j \in \mathcal{J}$. Our objective is to decide the association rules $\boldsymbol{\pi} \equiv (\pi_{ij})$ in order to optimize a network performance metric, such as the sum of RRH load or the average job completion time (delay).

In order to evaluate the impact of an association rule we consider the *download rate* obtained when device i is connected to station j while receiving exclusive service; this is denoted by R_{ij}. This rate should be calculated at the granularity of association changes. Typically, we may assume the use of a temporally-fair scheduler which distributes the station resources to the different users and averages out fast fading effects. In this case, a reasonable model for the download rate is:

$$R_{ij} = W \log(1 + \mathrm{SINR}_{ji}),$$

[1]Our approach applies also to the uplink as long as the modeled upload rate R_{ij} can be considered independent of the association rule.

where, W is the bandwidth, $\text{SINR}_{ji} \triangleq \frac{P_j G_{ji}}{\sum_{k\neq j} P_k G_{ki}+N_0}$, and the index ji is reversed on purpose to denote directivity, P_j denotes the transmit power, G_{ji} the path loss, N_0 the strength of thermal noise. This model has been extensively used in the literature, cf. [19, 21]. Here we use it only as an example; our framework only requires that R_{ij} are independent of the association variables. At this point, it might seem reasonable to connect each device to the RRH that provides the largest R_{ij}–known as the *MaxSINR rule*–however, this results in poor performance as we explain next.

3.2.2 RRH Load

We suppose that a device i requires to download traffic λ_i, which is known[2] and device-dependent. Also, it downloads jobs with average size $1/\mu$; extension to device- or RRH-specific job sizes is trivial. Given a decision $\boldsymbol{\pi}$ we may determine *the load of RRH j* as:

$$\rho_j(\boldsymbol{\pi}) = \sum_i \frac{\lambda_i}{\mu R_{ij}} \pi_{ij}.$$

Connecting devices to RRHs with high R_{ij} has the beneficial effect that the load contribution of each device $\lambda_i/\mu R_{ij}$ is minimized (since the term R_{ij} is maximized). However, when devices are not uniformly distributed in the area and/or total traffic demand is imbalanced, some RRHs attract more connections and become overloaded. This will result into poor performance, because wireless service also depends on the competition between users at the associated RRH, and rapidly degrades as $\rho_j(\boldsymbol{\pi}) \uparrow 1$. For example, we may estimate the *average job completion time* by $\frac{E[N_j]+1}{\mu R_{ij}}$, where $E[N_j]$ is the average number of jobs running at RRH j. Assuming the jobs are served according to the *processor sharing discipline*, for $\rho_j(\boldsymbol{\pi}) < 1$, $E[N_j]$ is given by [51]

$$E[N_j] = \frac{\rho_j(\boldsymbol{\pi})}{1-\rho_j(\boldsymbol{\pi})}.$$

As $\rho_j(\boldsymbol{\pi}) \uparrow 1$, the average completion time of a job of the connected devices will become very high.

Summing up the associations of device i, we may quantify its average completion time by $\sum_j \frac{\pi_{ij}}{\mu R_{ij}(1-\rho_j(\boldsymbol{\pi}))}$. Ultimately, we average over all the devices to obtain the *device association problem*:

$$\begin{aligned} \min_{\pi_{ij}\geq 0} & \sum_{ij} \frac{\pi_{ij}}{\mu R_{ij}(1-\rho_j(\boldsymbol{\pi}))} \qquad (3.1) \\ \text{s.t.} & \sum_j \pi_{ij} = 1 \quad \forall i \in \{1,2,\ldots,n\}, \\ & \rho_j(\boldsymbol{\pi}) < 1 \quad \forall j \in \{1,2,\ldots,m\}. \end{aligned}$$

[2]This information can either be provided by the device, or forecasted by the system.

This is a non-convex continuous optimization problem, and we are interested to solve it for large dimensions.

3.2.3 Related Work

The baseline MaxSINR association rule reads "connect each device to the station with the strongest signal". This simple association rule works well when the mobile traffic load is low, or symmetrically scattered around stations, but otherwise it can lead to significant load imbalances. An improved rule that captures interference (but not traffic fluctuations) is the MaxSINR rule "connect each device to the station with the highest SINR", [58].

To improve the performance over the above rules, the problem of device association has been studied in its integral form, where each device can be associated to a single station, i.e. the association variable $\pi_{ij} \in \{0, 1\}$, cf. [22, 24, 59, 60]. However, the combinatorial nature of such formulations makes them inefficient for large-scale instances. Instead, [19] allowed $\pi_j(x) \in [0, 1]$ for all points x in the plane and proved that the optimal solution is integral almost everywhere (except boundary points). In this chapter, we directly define the association variables to take values $\pi_{ij} \in [0, 1]$ with the understanding that fractional solutions correspond to multi-station coverage like CoMP [61]. We observe that our optimal solutions are also "sparse", meaning that although variables are allowed to take values in $[0, 1]$, at optimality most of them will be integral (0 or 1).

In the context of continuous device association, past work has considered the optimization of α-optimal functions [18–21] or general convex functions [16]. None of the past approaches can be used to solve (3.1), which is non-convex. In fact, to the best of our knowledge, solving a large non-convex problem like (3.1) is generally intractable. Our goal in this chapter is to propose a useful methodological tool, which can be applied to very large device association problems.

From the perspective of scalability, most past approaches do not meet the requirements we consider here. An exception is perhaps the distributed algorithm of [19] and the works expanding that framework like chapter 2. A limitation of this prior work, however, is that the decisions depend on the convergence of the empirical load and the lagrange multipliers to the optimal; leading to oscillations and rapid association changes, when optimal point is changing faster than the algorithm's convergence.

3.3 Optimal Transport

3.3.1 Introduction to Optimal Transport

The concepts of OT date back to the French mathematician Gaspard Monge who studied in 1781 the transportation of sand masses [30], what seems to be one of the first linear programming problems studied.

Although the theory of OT has been generalized to optimization with infinite variables, here we restrict our discussion to the illustrative case of "discrete OT"[3], where probability mass must be transported between two discrete distributions $\boldsymbol{p} \equiv (p_1, \ldots, p_m)$ and $\boldsymbol{q} \equiv (q_1, \ldots, q_n)$, and the transportation cost from point i to point j is C_{ij}–quite often taken to be the Euclidean distance between the two points. Due to Kantorovich [29], we can describe the transportation with a coupling π_{ij}, essentially a joint probability distribution $\boldsymbol{\pi}$ with marginals $\boldsymbol{p}$ and $\boldsymbol{q}$. The discrete OT problem can be written as:

$$\min_{\pi_{ij} \geq 0} \sum_{ij} C_{ij}\pi_{ij} \tag{3.2}$$

$$\text{s.t.} \sum_{j} \pi_{ij} = p_i, \quad i = 1, \ldots, m, \tag{3.3}$$

$$\sum_{i} \pi_{ij} = q_j, \quad j = 1 \ldots, n. \tag{3.4}$$

This is a linear program, solvable in polynomial time w.r.t. its size mn. Further, consider a bipartite graph connecting the points with links of weight C_{ij}, and connect each point i of $\boldsymbol{p}$ to a virtual source with link capacity p_i, and each point j of $\boldsymbol{q}$ to a virtual destination with link capacity q_j. The discrete OT corresponds to finding a minimum cost s-t flow of one unit. Using network simplex [62], we can obtain the solution in $O(E^2 \log V)$,[4] which for $V = m + n$, $E = mn$, and $n = m$, becomes $O(n^4 \log n)$, essentially quadratic to the input size mn. Although such solution is polynomial to the input size, our problem is of enormous dimensions, and hence the degree of the polynomial is important as well. Below we describe the regularized OT, a method to approximate OT in $O(n^2 \log n)$.

3.3.2 Regularized OT

In 2013, Cuturi [54] proposed to approximate OT with a regularized version. In particular, he proposed to modify the objective of OT by subtracting the *entropy* $H(\boldsymbol{\pi}) = -\sum_{ij} \pi_{ij}(\log \pi_{ij} - 1)$, weighted with the *regularization*

[3]The notion of discreteness refers to discrete probability measures, hence we have a finite number of continuous transportation variables.

[4]Pseudo-polynomial algorithms are faster, but their runtime guarantee depends on the values of C_{ij} [63, 64].

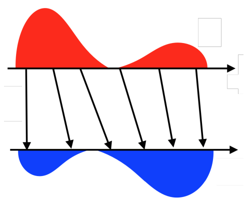


Figure 3.2: OT studies the mass transportation cost that satisfies initial (red) and final (blue) conditions.

strength coefficient $\epsilon > 0$. The regularized OT becomes:

$$\begin{aligned} \min_{\pi_{ij}} & \sum_{ij} C_{ij}\pi_{ij} + \epsilon \sum_{ij} \pi_{ij}(\log \pi_{ij} - 1) \\ \text{s.t.} & \sum_{j} \pi_{ij} = p_i, \quad i = 1, \dots, m, \\ & \sum_{i} \pi_{ij} = q_j, \quad j = 1 \dots, n. \end{aligned} \tag{3.5}$$

Adding $-\epsilon H(\boldsymbol{\pi})$ has a number of beneficial effects:

- The objective of the regularized OT is 1–strongly convex, hence (3.5) has a unique optimal solution.
- By Proposition 4.1 of [28], the sequence of unique optimal solutions of (3.5) for $\epsilon^{(k)}$ converges to an optimal solution to (3.2) as $\epsilon^{(k)} \to 0$.
- $H(\boldsymbol{\pi})$ forces π_{ij} to be non-negative, hence we can drop the constraint $\pi_{ij} \geq 0$.

More importantly, the new objective admits an intuitive reformulation, which leads to a faster algorithm. Consider the affine constraint sets $\mathcal{C}_p = \{\boldsymbol{\pi} \mid (3.3)\}$, and $\mathcal{C}_q = \{\boldsymbol{\pi} \mid (3.4)\}$, and let $\xi_{ij} = \exp(-C_{ij}/\epsilon)$; we can rewrite the regularized OT as:

$$\min_{\boldsymbol{\pi} \in \mathcal{C}_p \cup \mathcal{C}_q} KL(\boldsymbol{\pi}, \boldsymbol{\xi}),$$

where $KL(\boldsymbol{\pi}, \boldsymbol{\xi}) \triangleq \sum_{ij} \pi_{ij}(\log \frac{\pi_{ij}}{\xi_{ij}} - 1)$ is the Kullback-Leibler (KL) divergence. Hence, the regularized OT is a KL projection of $\boldsymbol{\xi}$ onto the intersection of $\mathcal{C}_p$ and $\mathcal{C}_q$, an interpretation pointed out in [65]. Since the KL divergence is the special case of the Bregman divergence for the entropy function, our KL projections benefit from the convergence property of iterative Bregman projections on intersections of affine constraint sets [66, 67]. To obtain an iterative projection algorithm it is convinient to operate on the dual domain.

Consider the Lagrangian function

$$L(\boldsymbol{\pi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = KL(\boldsymbol{\pi}, \boldsymbol{\xi}) + \sum_i \alpha_i \left(\sum_j \pi_{ij} - p_i \right) + \sum_j \beta_j \left(\sum_i \pi_{ij} - q_j \right).$$

The KKT stationarity condition requires that for each (i, j) it must hold:

$$\frac{\partial L}{\partial \pi_{ij}} = 0 \quad \Leftrightarrow \quad \pi_{ij} = \xi_{ij} e^{-\alpha_i} e^{-\beta_j}.$$

To obtain the projection with respect to $\mathcal{C}_p$ we follow the steps: (i) fix (β_j), (ii) apply the complementary slackness condition $\sum_j \pi_{ij} = p_i$, (iii) solve for α_i. Similarly for $\mathcal{C}_q$.

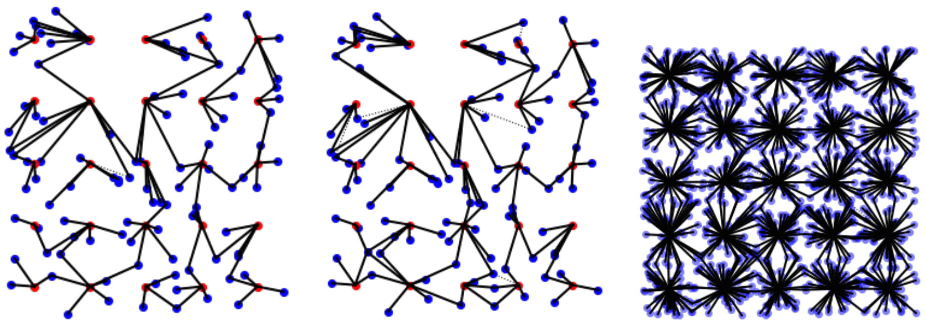
Setting $a_i \equiv e^{-\alpha_i}$ and $b_j \equiv e^{-\beta_j}$, we obtain the Sinkhorn algorithm [68]:

Sinkhorn Algorithm

```
Input  : $\boldsymbol{C}$, $\boldsymbol{p}$, $\boldsymbol{q}$, $\epsilon$
Output : $\boldsymbol{\pi}$
1 initialize  $\boldsymbol{b}^{(0)} = \mathbf{1}$, $\xi_{ij} = e^{-C_{ij}/\epsilon}$;
2 while accuracy do
3     $k \leftarrow k + 1$ ;
4     $a_i^{(k)} \leftarrow \frac{p_i}{\sum_j b_j^{(k-1)} \xi_{ij}}, \quad \forall i$ ;
5     $b_j^{(k)} \leftarrow \frac{q_j}{\sum_i a_i^{(k)} \xi_{ij}}, \quad \forall j$ ;
6 end
7 $\pi_{ij} \leftarrow \xi_{ij} a_i^{(k)} b_j^{(k)}, \quad \forall (i, j)$
```

Theorem 1 (From [69]). *Assume $m = n$, fix $\tau > 0$, and choose $\gamma = \frac{4 \log n}{\tau}$, Sinkhorn algorithm computes a $\tau-$approximate solution of* (3.2) *in* $O(n^2 \log n \tau^{-3})$ *operations.*

Since the problem size is n^2, Sinkhorn algorithm converges almost linearly for any fixed τ (less the logarithmic term). This is to be contrasted with the almost quadratic convergence of network simplex $O(n^4 \log n)$. Further, Sinkhorn requires only matrix-vector multiplications, hence it admits highly efficient GPU implementations. More information on computational transport can be found in [28].



(a) LP, 100 × 25 (b) Reg OT, 100 × 25 (c) Reg OT, 1k × 25

Figure 3.3: Device association to RRHs.

Devices	RRHs	*LP-glpk*	*Sinkhorn* $\tau = 1\%$	*Sinkhorn* $\tau = 0.1\%$
10	25	6	2.27	5.02
50	25	88	4.09	8.95
100	25	315	8.09	9.7
500	25	8130	10.1	31.6
1000	25	out of memory	27	37.9
5000	25	out of memory	135	204
10000	25	out of memory	434	568

Table 3.1: Runtime (msec) comparison Sinkhorn vs LP-glpk.

3.3.3 Sinkhorn vs LP

We implemented Sinkhorn in python and compared its performance to the embedded glpk solver [70], known to be one of the fastest LP solvers. Table 3.1 provides some indicative numerical results to highlight the advantageous performance of Sinkhorn over a standard LP solver, namely: (i) it is faster, (ii) scales better, and (iii) doesn't run out of memory.

In the experiments, we stopped Sinkhorn when the total absolute residual becomes less than 1% or 0.1%. The runtimes provided are averages over 7 runs, computed with the `timeit` package. Notably, we can compute a 1.001-optimal transport 25 × 10k in half a second.

Fig. 3.3(a)-3.3(b) showcase associations obtained by Sinkhorn and LP-glpk, where we can verify the fidelity of Sinkhorn to the optimal LP solution. Fig. 3.3(c) showcases a very large instance that the LP solver cannot address.

3.4 OT as a Heuristic Load Balancer

In this section we explain how one can use OT algorithms to obtain a feasible device association rule. An instance of the OT problem is defined by the triplet $(\boldsymbol{C}, \boldsymbol{p}, \boldsymbol{q})$, i.e., the costs and the left-right marginals. In order to produce a load balancer based on OT, we must provide appropriate values for these parameters. We propose the following choices:

- For $\boldsymbol{C}$: we propose to use (i) the Euclidean distance between the location x_i of device i and the location x_j of RRH j, i.e., $C_{ij} = \|x_i - x_j\|$, or (ii) the incurred load per unit traffic $C_{ij} = 1/\mu R_{ij}$.
- For $\boldsymbol{p}$: the input traffic, $p_i = \lambda_i$.
- For $\boldsymbol{q}$: we propose to use (i) equal RRH traffic $q_j = \sum_i \lambda_i / n$, (where n is the number of RRHs) or (ii) RRH traffic from the maxSINR rule, $\Lambda_j = \sum_i \pi_{ij}^{SINR} \lambda_i$.

We provide some explanations about the choices. First, the left marginal $\boldsymbol{p}$ ensures that the entire traffic of each devices is split among RRHs. To choose $\boldsymbol{q}$ wisely, we should know the RRH traffic at optimality, however this information is often not accessible. Instead, it is easy to obtain the maxSINR rule. Last, Euclidean cost favors nearby RRHs, while normalized load connects devices to the RRHs with minimum incurred load, taking into account interference and path loss. We have the following interesting result:

Lemma 1. *Consider the maxSINR rule denoted with π_{ij}^{SINR}, and the incurred traffic $\Lambda_j = \sum_i \pi_{ij}^{SINR} \lambda_i$, and suppose it is feasible, i.e., $\rho_j(\boldsymbol{\pi}^{SINR}) < 1$, $\forall j$. Also, consider a $(\boldsymbol{C}, \boldsymbol{p}, \boldsymbol{q})$ instance of the OT problem, such that (i) $C_{ij} = 1/\mu R_{ij}$, (ii) $p_i = \lambda_i$, (iii) $q_j = \Lambda_j$, with solution $\boldsymbol{x}^*$.*

Then, the association $\pi_{ij} \doteq x_{ij}^* / \lambda_i$ minimizes the total load $\sum_j \rho_j(\boldsymbol{\pi})$.

Proof. First, note that whenever the maxSINR rule π_{ij}^{SINR} is feasible, it minimizes the total load. This is easy to check by observing that the total load contribution of each device $\lambda_i \sum_j \pi_{ij}^{SINR} / \mu R_{ij}$ is minimized under the maxSINR rule. Then the total load minimization follows by summing up over devices.

The OT solution of the lemma is defined as follows:

$$\begin{aligned} \boldsymbol{x}^* \in \arg\min_{x_{ij} \geq 0} & \sum_{ij} \frac{x_{ij}}{\mu R_{ij}} \\ \text{s.t.} & \sum_j x_{ij} = \lambda_i, \quad i = 1, \ldots, m, \\ & \sum_i x_{ij} = \Lambda_j, \quad j = 1 \ldots, n. \end{aligned} \tag{3.6}$$

Note that $\lambda_i \pi_{ij}^{SINR}$ satisfies all the constraints of (3.6), and thus it is a feasible solution of (3.6). Further, since π_{ij}^{SINR} minimizes the total load, it

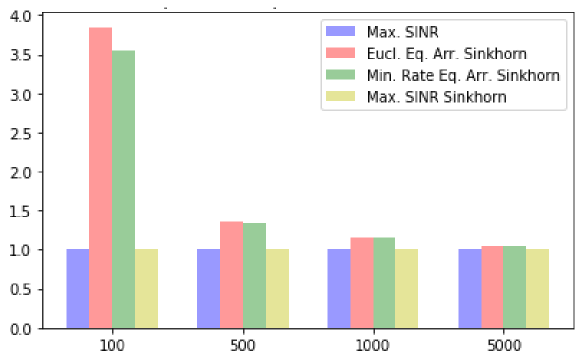


Figure 3.4: Relative completion time (uniform traffic)

must be an optimal solution of (3.6). Therefore:

$$\sum_{ij} \pi^*_{ij} \frac{\lambda_i}{\mu R_{ij}} = \sum_{ij} \pi^{SINR}_{ij} \frac{\lambda_i}{\mu R_{ij}} = \min_{\boldsymbol{\pi}} \sum_j \rho_j(\boldsymbol{\pi}),$$

and the lemma follows. □

The lemma states that given the RRH traffic under the maxSINR rule Λ_j, we may use OT to retrieve the association variables that minimize the total load, a very useful property. More broadly, OT can be used to find the associations that produce certain RRH traffic patterns.

Choosing the marginals like in Lemma 1 (or with a more elaborate scheme as we show below) will work better in practice, however, the choices given at the start of the section can also be useful: they are simpler to compute and can be sufficient when the spatial traffic is uniform. To showcase this property, we experiment with a scenario with random but uniform traffic. In a C-RAN cell with 25 RRHs we scale the number of devices from 100 to 5000. We compare the average completion time under 4 policies, (a) maxSINR, (b) OT with Euclidean costs and equal RRH traffic, (c) OT with load costs and equal RRH traffic, and (d) OT with load costs and RRH traffic equal to Λ_j (same as Lemma 1). The values shown in Fig. 3.4 are the ratios of average completion time between the algorithm and the maxSINR. We observe that algorithm (d) performs the same with maxSINR (a) as predicted by the Lemma. The other two algorithms perform similarly. In particular, for a small number of devices, we see that selecting $q_j = \sum_i \lambda_i/n$ results in performance deterioration (up to 4 times worse), due to the random locations of the devices. However, as the number of devices increases, the traffic becomes more uniform and the choice $q_j = \sum_i \lambda_i/n$ becomes as good as the maxSINR solution.

3.5 Learning RRH Load

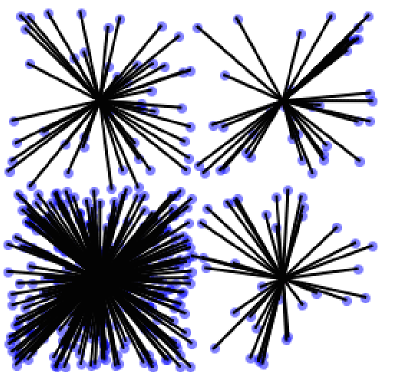
In this section we consider non-uniform traffic, as for example in figures 3.5(a)-3.5(b). When the traffic is non-uniform, both the maxSINR rule and our OT heuristics based on equalizing traffic at RRH will perform badly. Instead, we must discover the correct traffic balancing at RRHs which will provide the average completion time optimization.

We propose an iterative algorithm, where at iteration k Sinkhorn algorithm is used with $\boldsymbol{q}^{(k)}$ to obtain an association which results in a specific RRH loading $\rho_j(\boldsymbol{\pi}^{(k)})$. According to this loading, a new marginal $\boldsymbol{q}^{(k+1)}$ is computed, and the process repeats until an accuracy criterion is satisfied. More specifically, our iterative algorithm picks the RRH with the highest load (step 5) and then decreases its aggregate traffic by a fixed term δ, dispersing the traffic to all other RRHs (step 6). We mention that increasing the traffic in a remote RRH will result in a large number of association changes in the Sinkhorn algorithm which will ensure that the steered traffic maintains minimum transportation cost. We provide the algorithm flow here.

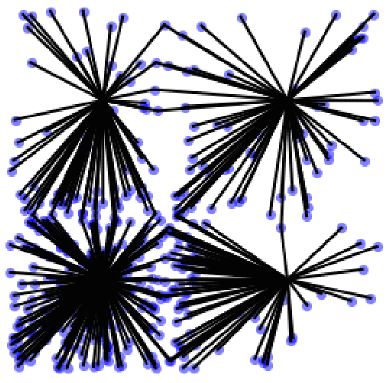
Adaptive Sinkhorn Algorithm

```
  Input  : $C_{ij} = 1/\mu R_{ij}$, $\boldsymbol{p} = \boldsymbol{\lambda}$, $\epsilon$
  Output : $\boldsymbol{\pi}$
1 initialize  $q_j = \sum_i \lambda_i/n$;
2 while accuracy do
3     $k++$ ;
4     $\boldsymbol{\pi}^{(k+1)} \leftarrow \text{Sinkhorn}(\boldsymbol{C}, \boldsymbol{p}, \boldsymbol{q}^{(k)}, \epsilon)$ ;
5     $j^* \in \arg\max\{\rho_j(\boldsymbol{\pi}^{(k+1)})\}$ ;
6     $q_j^{(k+1)} = \begin{cases} q_j^{(k)} - \delta & j = j^* \\ q_j^{(k)} + \delta/(n-1) & j \neq j^* \end{cases}$ ;
7 end
8 $\boldsymbol{\pi} \leftarrow \boldsymbol{\pi}^{(k+1)}$
```

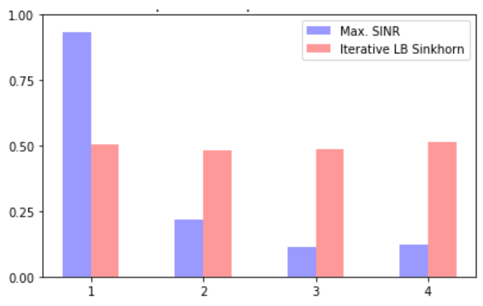
Figures 3.5(a)-3.5(c) show the results. First, comparing the two association rules we see that although the maxSINR rule associates the devices according to interference and not traffic, our adaptive Sinkhorn algorithm considers both, and converges to an association where some cell edge devices are steered to the neighboring RRHs. This is done to alleviate the load of the bottom-left RRH. Indeed, figure 3.5(c) shows the resulting loads of the 4 RRH. Our approach successfully equalizes the loads of the different RRH, while the maxSINR rule fails to do so. Ultimately, our scheme achieves an average completion time 6.3msec while the maxSINR rule 24msec, which



(a) maxSINR



(b) Adaptive OT



(c) Load (non-uniform traffic)

Figure 3.5: Comparison of maxSINR and Optimal Transport.

corresponds to almost 4 times improvement.

3.6 Conclusion

In this chapter, we studied the device association in C-RAN, using the Sinkhorn algorithm inspired by the theory of Optimal Transport. This algorithm has been previously used in large-scale problems in imaging and machine learning, and here it is applied to provide a heuristic load balancer in C-RAN systems. First, it is shown that a very simple version of this algorithm is capable of providing low delay associations when the traffic is uniformly spread in the covered geographical area. In case of non-uniform traffic, we extend the algorithm to an iterative version which progressively improves the load balancing. We show that our scheme scales to very large problem instances, and has the potential to provide great improvements over the simple baseline approach.

We note that, this chapter's method relies on a good estimate of the current traffic and into the ability of the network to compute and update the network configuration in real time. If this decision update process is slower than the change of status in the system (in the meantime new users appear, traffic volume changes, etc), the new configurations will be stale and suboptimal and will fail the QoS guarantees. In the following, we will present a pro-active optimization method, that will not require knowledge of the demand and real time configurations, but will extract the demand pattern and compute robust associations based on collected data.

Chapter 4

Data-Driven User Association based on Robust Optimization

4.1 Introduction

The explosion of wireless traffic is driving network operators to deploy heterogeneous sites and constantly increase base station density, in an effort to improve the spectrum reuse [6, 10]. This trend is expected to culminate with the emerging *Ultra Dense Networks (UDNs)* in the 5G and beyond era, where a user located in an urban area will be surrounded by hundreds of sites, while the available cells may be more than the number of active users [5–7]. Nevertheless, providing a copious amount of resources is only the first step. A second equally important step is to develop efficient resource management mechanisms in order to balance base station loads, under the high spatio-temporal traffic variability resulting from the small number of users per base station [9]. In the demanding environment of dense 5G networks, user association (choosing a base station for each user among a large number of candidates) and traffic steering (serving a traffic flow from the right base station, carrier, etc.) must be surgically engineered for proper exploitation of the increased density [9].

Including the distributed method presented in chapter 2, a number of recent works formalize the QoS user association problem and attempt to find an optimal solution, [9, 17–24]. Nevertheless, these frameworks are ineffective for our setup for two main reasons:

- *Spatio-temporal variability*: Due to smaller user/site ratios, the traffic demand will vary significantly more over time and space, giving rise to unpredictable traffic spikes.
- *Increased QoS requirements*: With the rise of vertical applications, 5G networks are expected to support slices that provide guaranteed Quality of

Service (QoS). Unexpected traffic spikes combined with dynamic association decisions reacting to them, might lead the system to oscillations, instability, and violation of QoS requirements.

Towards addressing these issues, this chapter proposes a radically different approach based on two main components:

1. *Data-driven user association maps* pre-calculated for each day and time period (e.g. per hour), based on estimates of average traffic demand for that time, day, and location; these maps *proactively* associate users/flows from certain locations to certain base stations, rather than constantly oscillating association decisions due to traffic spikes.

2. *A robust optimization framework for user association* that takes into account the prediction error and protects system QoS from traffic spikes.

To our best knowledge, this is the first work to propose such robust pre-calculated user association maps for dense heterogeneous networks. Moreover, our work is the first to explore an interesting new user association tradeoff between improving network performance vs. facilitating traffic prediction. These two goals are not always aligned, as it will become clear in our analysis.

4.1.1 Related Work and our Contribution

Selecting the association rule vector $\boldsymbol{\pi}$, that assigns each new user to a base station, can be formulated as an optimization problem that targets to maximize a utility function of the resulting base station loads [17–24, 48].

Such optimization problems are usually difficult to solve, due to the coupling of the user association decisions; adding a user alters the base station load and affects the performance for all the users connected to it. In the context of UDNs, the size of the problem grows (100s of base stations in small areas with many locations or users to be associated) and mounts an extra difficulty. Complex optimization approaches fall short, since practical systems require fast and lightweight solutions.

The integral user association problem is a combinatorial problem [24], but in [19] the authors show that the optimal solution of their convex relaxation problem is in fact integral, enabling in this way optimization of a general class of objective functions. Later, [21] introduced backhaul constraints in the model, while in chapter 2 and [18] we introduced quality of service guarantees for premium users. A dynamic biasing scheme is proposed in [22] to load balance heterogeneous wireless cells. In chapter 3 and [17], we developed a scalable approximate algorithm by using optimal transport theory to achieve lower user latency. Most of the prior work, including the algorithms mentioned, cannot handle the fast evolving traffic that follows human activity in a densely populated area. The goal of the presented solution is to fill this gap in the literature. The framework introduced here, instead of monitoring and trying to follow the rapidly-changing status of the network, takes decisions based on the predicted traffic demand based on the data.

A modern trend in networking problems is to tap into the power of available data to deal with uncertainty [71]. For wireless traffic, many prior works identify structure, like the diurnal pattern during the day or the similarity of traffic during the weekdays and weekends/holidays [13, 14]. However, up to now it is far from clear how to best utilize the data and the observed patterns for improving the performance of user association. *Our contributions in this chapter are the following:*

- We propose the idea of precomputing maps, that can be used later to determine user association in real-time.
- To study the map performance we propose an analytical data-driven framework for user association in the context of dense wireless networks with unpredictable traffic spikes. This leads us to the formulation of the Robust User Association Map (RUAM) problem.
- We then propose the Generalized Robust Map Algorithm (GRMA) that provably produces optimal robust maps, for a large class of objective functions. Moreover, we provide methods for creating approximate robust maps to enable and accelerate the gradient computation when the objective function is challenging (e.g. α-optimal, or α-fair with $\alpha > 2$).
- Further, we demonstrate the efficiency of our framework on real telecom traces [33], compared to a popular user association algorithm [19]. Our simulations show that we achieve more stability (up to 25% improvement) and decreased average latency, especially in periods of high traffic activity. Our framework adapts to design choices, balancing trade-offs in stability and cost, by tuning the Service Layer Agreement (SLA) guarantees.
- Finally, we discuss advanced time series forecasting methods. Using these methods on the data [33], we validate the gaussian estimator error model and improve the quality of the map produced by GRMA; decreasing the cost gap compared to an optimal map down to 5%.

4.2 Architecture

4.2.1 System Model

We consider a region $\mathcal{L} \subseteq \boldsymbol{R}^2$ with ultra dense cellular coverage from a set of $\mathcal{B}$ (possibly heterogeneous) base stations. This region we envision it as a 2D representation of a dense urban environment with fixed base station positions. We note that, in the following analysis we focus on downlink traffic and association, assuming that downlink and uplink can be independently optimized (5G, [21]), we can use our framework on uplink data to calculate the uplink robust association maps.

Spatial traffic. Users at location $x \in \mathcal{L}$, generate flow requests according to an inhomogeneous Poisson point process with spatial intensity $\lambda(x)$ and have independently and generically distributed file sizes with mean $\frac{1}{\mu}$.

Service Rate. The flows generated at a point $x \in \mathcal{L}$ that are associated to a base station $i \in \mathcal{B}$ are served with rate $C_i(x)$. In our model, $C_i(x)$ is a location-dependent metric that depicts the wireless signal degradation due to distance of x from the base station $i \in \mathcal{B}$.

$$C_i(x) = W \log(1 + \mathrm{SINR}_i(x)),$$

where W is the available frequency bandwidth, and $\mathrm{SINR}_i(x)$ is given by:

$$\mathrm{SINR}_i(x) = \frac{P_i G_i(x)}{\sum_{j \neq i} P_j G_j(x) + N_0}.$$

P_i denotes the transmission power of base station i, N_0 denotes noise power and $G_i(x)$ is the path loss between the antenna and the UE. This model has been shown to accurately capture the average behavior of wireless systems including shadowing, interference, and path-loss [17–19, 21, 22, 49].

Association Rules. Let $\pi_i(x) \in [0, 1]$ be the association rules, indicating the fraction of traffic of location x associated to base station i. To associate the total traffic of location x we enforce the constraint $\sum_{i \in \mathcal{B}} \pi_i(x) = 1$. The association variables $\pi_i(x),\ \forall x \in \mathcal{L}$ will be the means to control the performance of the system.

Base Station Load. The load ρ_i is the fraction of time base station i is busy. The load contribution from a specific location depends on the association rules, and it is equal to $\frac{\lambda(x)}{\mu C_i(x)} \pi_i(x)$. Therefore, considering the area $\mathcal{L}$:

$$\rho_i = \int_{\mathcal{L}} \frac{\lambda(x)}{\mu C_i(x)} \pi_i(x) dx. \tag{4.1}$$

The vector of base station loads $\boldsymbol{\rho} = (\rho_i)$ is an important performance metric of the system. For example, [44] suggests that assuming a temporal fair scheduler (e.g. round robin, proportional fair) the dynamics of the base station queues can be accurately described by an M/G/1 processor sharing system, where the expected number of active users at base station i is given by $\mathbb{E}[N_i] = \frac{\rho_i}{1-\rho_i}$. This is tightly related with average response time for a flow in base station i, which from Little's law is $\mathbb{E}[T_i] = \frac{1}{\lambda_i} \frac{\rho_i}{1-\rho_i}$, where $\lambda_i = \int_{\mathcal{L}} \lambda(x) \pi_i(x) dx$, and with the average delay experienced at a location $x \in \mathcal{L}$: $\mathbb{E}[T | X = x] = \sum_{i \in \mathcal{B}} \frac{1}{\mu C_i(x)(1-\rho_i)} \pi_i(x)$, which is derived from the flow throughput equation in [44].

In the following sections we will focus on how to choose association rules $\pi_i(x), \forall x \in \mathcal{L}$ to achieve specific vectors $\boldsymbol{\rho}$ that correspond to important network-wide objectives, e.g. total throughput, average queuing delay, or balancing base station loads.

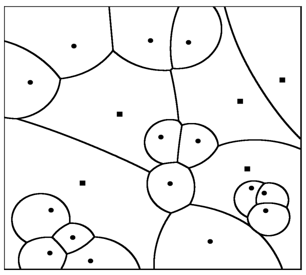


Figure 4.1: Representation of an association map.

Table 4.1: Notation Description Table

Notation	Description
$C_i(x)$	Service rate of base station i to location x
$\hat{\lambda}(x)$	Actual traffic value $\hat{\lambda}(x) = \bar{\lambda}(x) + N_n(x)$
$\hat{\rho}_i(\boldsymbol{\pi}_i)$	Actual base station i load, $\hat{\rho}_i = \rho_i + Y_i$
Y_i	Load prediction error $Y_i \sim N(0, S_i^2)$
$\pi_i(x)$	Association probability of location x to base station i
$S_i^2(\boldsymbol{\pi}_i)$	Actual load error variance
c_i	Actual load threshold for base station i
ϵ_i	Allowed probability of overloading for base station i
$\phi(\boldsymbol{\pi})$	Convex seperable objective function
$\Phi(\boldsymbol{\pi})$	Partially relaxed Lagrangian function
γ_i	Lagrangian multiplier for base station i

4.2.2 User Association Maps

In this chapter we are interested in precalculated association rules $\boldsymbol{\pi}$, which we call *user association maps*. When a request is generated at a given location, the map probabilistically determines the base station that will serve the user. A *feasible* map $\boldsymbol{\pi}$ for given traffic $\lambda(x)$ must (a) associate the entire traffic of every location x and (b) ensure through Eq.(4.1) that the base station loads are limited to < 1 (since $\rho_i > 1$ means that base station i is unstable).

Definition 1 (Feasible user association map)**.**

$$\mathcal{F} = \left\{ \boldsymbol{\pi} \in [0,1]^{\mathcal{L}\times\mathcal{B}} \;\middle|\; \begin{array}{ll} \rho_i \leq 1-\epsilon, & \forall i \in \mathcal{B}, \\ \sum_{i\in\mathcal{B}} \pi_i(x) = 1, & \forall x \in \mathcal{L}. \end{array} \right\}$$

As given in Def.1, $\mathcal{F}$ includes only the most generic and necessary constraints. Our model can be extended to include application-specific constraints. For example, in the context of UDNs user experience can be improved in crowded locations by further restricting the load of certain base stations, e.g., ensuring $\rho_i < c_i < 1$ for some i, as in [18]; this constraint is handled in this chapter. In the context of network slicing, it is useful to define multiple classes of users and design a different map per class such that a user can enjoy a slice-specific QoS level, similar to [18]. For clarity of exposition, this extension is left for future work.

Definition 2 (Objective Function)**.** $\phi(\boldsymbol{\pi})$ *is a generic differentiable and seperable convex function.*

Problem 1 (P1: Generic User Association Problem)**.**

$$\underset{\boldsymbol{\pi}\in\mathcal{F}}{minimize}\ \phi(\boldsymbol{\pi}).$$

We give here some example objective functions, choosing: (i) $\phi(\boldsymbol{\pi}) = \sum_i \rho_i$ maximizes the total system throughput, (ii) $\phi(\boldsymbol{\pi}) = \sum_i \rho_i^2$ balances proportionally the base station loads [72], (iii) $\phi(\boldsymbol{\pi}) = \sum_i \rho_i^{\alpha},\ \alpha \to \infty$ makes the base station loads as equal as possible. In fact, it can be shown that $\phi(\boldsymbol{\pi})$ as given in Definition 2 can be chosen to yield as optimal solution any vector $\boldsymbol{\pi}$ in the feasible set $\mathcal{F}$, and therefore we do not need more generic functions.

Definition 3 (Optimal User Association Map)**.** *A solution of P1* $\boldsymbol{\pi}^{\star}$ *is called the optimal user association map.*

Observe that the optimal solution of P1 strongly depends on knowing the demand $\lambda(x)$ (through Eq.(4.1)) at a fine spatial granularity x. A number of related works take these as known assuming that they can be estimated from data, cf. [19] and followups. With accurate knowledge of demand, the average number of competing for resources at base station i can be expressed as $\mathbb{E}[N_i] = \frac{\rho_i}{1-\rho_i}$, thus setting a threshold c_i such that $rho_i \leq c_i < 1$ also ensures a bound on the average active flows. Additionally, as explained in [18], this can be used to tune a bound on average delay experienced at that station.

In practice, due to the natural demand fluctuations (especially related to non-stationary phenomena) *there will always be discrepancies between actual and estimated demand, even with the best estimators.* In the context of UDNs, this poses a great threat to user association, as the discrepancies are expected to be larger. To this end, we introduce next our proposed estimators, and then describe how to rigorously treat these unpredicted discrepancies and avoid violating SLAs.

4.2.3 Statistical Methods for Mobile Traffic Prediction

Mobile traffic exhibits strong diurnal patterns, which make it predictable; the interested reader is referred to [14], [13] and [15] for extensive analyses. We use a publicly available dataset collected in the Milano area, analyzed

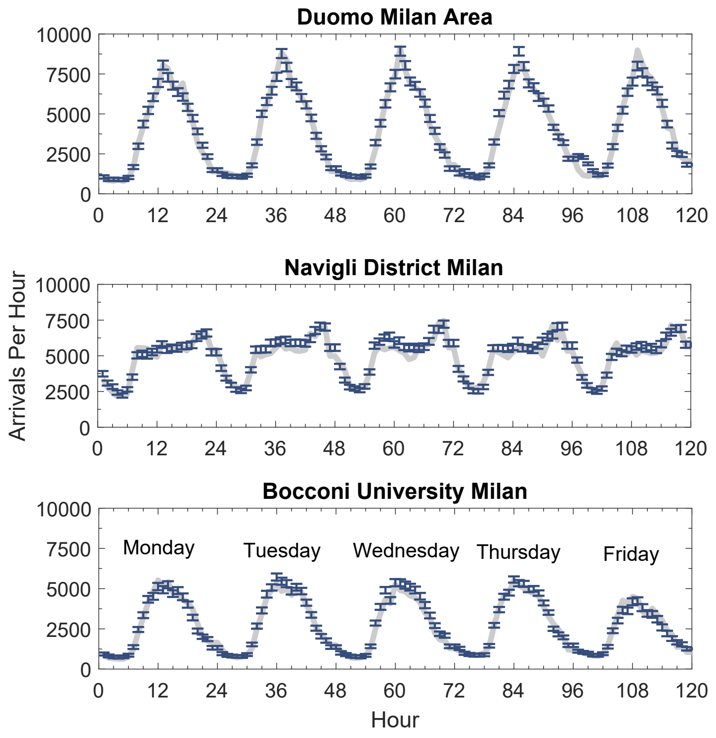


Figure 4.2: Traffic Prediction based on Eq.(4.3) for 3 different areas of Milano from Monday 2/12/2013 to Friday 6/12/2013 and comparison with the actual traffic (a) Duomo Area (b) Navigli District (c) Bocconi University.

in [13], wherein it has been observed that the daily pattern is stronger when considering each day of the week and each location on the Milano grid separately. Motivated by this and simplicity for exposition, we introduce the following traffic predictor.

Definition 4 (Traffic Predictor). *Let $X_i^{t,d}(x)$ denote the measured intensity (#of arrivals/h) at location x, hour t, and day d of the week, i weeks before the current. The predicted spatial intensity is based on the data of the last n weeks:*

$$\bar{\lambda}^{t,d}(x) = \frac{1}{n}\sum_{i=1}^{n} X_i^{t,d}(x). \tag{4.2}$$

Hereinafter, we focus on a single hour/day slot of the week and drop the notation t, d. The actual value of traffic intensity is modeled to be equal to the predicted one $\bar{\lambda}(x)$, plus a zero-mean Gaussian prediction error:

$$\hat{\lambda}(x) = \bar{\lambda}(x) + N_n(x), \tag{4.3}$$

where $N_n(x) \sim N(0, \sigma_n^2(x))$, and $\sigma_n^2(x)$ is the sample variance, which is given by

$$\sigma_n^2(x) = \frac{1}{n}\sum_{i=1}^{n}(X_i(x) - \bar{\lambda}(x))^2.$$

In Eq.(4.3) we have implicitly assumed that the observed data of a specific hour of a week are drawn from the same distribution across different weeks, in which case Eq.(4.3) follows from Eq.(4.2) and the central limit theorem.

In Fig.4.2 we evaluate our simple predictor on the Milano dataset [33]. We have trained the predictor for the whole November and used it to predict the first week of December (2-6/12/2013). We see that our model predicts accurately the traffic in most situations for three areas with heterogeneous behavior: 1) the Duomo, with tourist activity 2) the Navigli District, with nightlife activity and 3) the Bocconi University, with scholar activity. Advanced techniques like the ARIMA model [34] and LSTM networks [35] are explored in Sect.4.6.

Although the described traffic prediction is fairly accurate, there are spikes in the traffic which are non-stationary and cannot be predicted based on past data, for example see in Fig.4.2 at the peak hour on Friday at Duomo (109 hour). If we design the association map disregarding the prediction error, these unpredictable spikes can lead to constraint violations. The next lemma characterizes the behavior of the base station load as a function of the prediction error.

Lemma 1. *Fix (t, d), and let the hourly spatial intensity of traffic $\hat{\lambda}(x)$ be related to the predicted one as explained in Eq.(4.3). Fix the user association vector $\boldsymbol{\pi}$. The actual load $\hat{\rho}_i$ of base station i is related to the estimated one ρ_i as follows:*

$$\hat{\rho}_i = \rho_i(\boldsymbol{\pi}_i) + Y_i(\boldsymbol{\pi}_i), \tag{4.4}$$

where Y_i – the base station load prediction error – is zero mean Gaussian random variable with variance:

$$S_i^2(\boldsymbol{\pi}_i) = \int_{\mathcal{L}} \frac{\pi_i^2(x)}{\mu^2 C_i^2(x)} \sigma_n^2(x) dx. \tag{4.5}$$

Proof. Analytically, starting from the estimate of $\lambda(x)$:

$$\hat{\lambda_n}(x) = \lambda(x) + N_n(x)$$
$$\frac{\hat{\lambda_n}(x)}{\mu C_i(x)} = \frac{\lambda(x)}{\mu C_i(x)} + \frac{N_n(x)}{\mu C_i(x)}$$
$$\int_{x\in\mathcal{L}} \frac{\hat{\lambda_n}(x)}{\mu C_i(x)}\pi_i(x)dx = \rho_i + \int_{x\in\mathcal{L}} \frac{N_n(x)}{\mu C_i(x)}\pi_i(x)dx$$
$$\hat{\rho}_i = \rho_i + Y_i.$$

The aggregate noise that is generated from all the locations $x \in \mathcal{L}$ associated with a base station i is described by the random variable: $Y_i = \int_{x\in\mathcal{L}} \frac{N_n(x)}{\mu C_i(x)}\pi_i(x)dx$ and $Y_i \sim N(0, \int_{\mathcal{L}} \frac{\pi_i^2(x)}{\mu^2 C_i^2(x)}\sigma_n^2(x)dx)$, since $N_n(x) \sim N(0, \sigma_n^2(x))$. □

We emphasize that the variance of the error of the actual load $S_i^2(\boldsymbol{\pi}_i)$ depends on the association rule $\boldsymbol{\pi}$, Eq.(4.5). Intuitively, the less actual traffic we associate to a base station, the less confident we are that the actual load will match the predicted load, and thus the more conservative we need to be to avoid the violation of its load constraint. This leads us to an interesting observation: *to optimize user association maps under uncertainty we must select the rules $\boldsymbol{\pi}$ considering jointly their impact on the base station load objective and the prediction error.* Next, we introduce the concept of RUAM which accurately captures this tradeoff.

4.2.4 Robust User Association Maps

In order to optimize user association maps under uncertainty, we will reformulate P1 using the theory of robust optimization [73], [31]. In terms of objective function, we seek to optimize $\mathbb{E}_Y\left[\phi\left(\hat{\boldsymbol{\rho}}(\boldsymbol{\pi})\right)\right]$, where ϕ is the objective function of Def.2, and the expectation is taken with respect to the Gaussian prediction error. In terms of constraints, we require that the actual BS load does not exceed a tunable parameter c_i with a selected probability ϵ_i:

$$\mathbb{P}(\hat{\rho}_i \geq c_i) \leq \epsilon_i. \tag{4.6}$$

Therefore, the *robust* feasibility set $\mathcal{F}^r$ contains association maps $\boldsymbol{\pi}$, such that the predicted base station load $\boldsymbol{\rho}(\boldsymbol{\pi})$ and the prediction error variance $\boldsymbol{S}(\boldsymbol{\pi})$ satisfy certain conditions as explained below. For the remainder of the text we simplify notation from $\mathbb{E}_Y\left[\phi\left(\hat{\boldsymbol{\rho}}(\boldsymbol{\pi})\right)\right]$ to $\mathbb{E}\left[\phi(\boldsymbol{\pi})\right]$, with the understanding that the expected cost is taken with respect to the distribution of the prediction error $\boldsymbol{Y}$, which is controlled by $\boldsymbol{\pi}$.

Problem 2 (P2: Robust User Association Problem)**.**

$$\underset{\boldsymbol{\pi}\in\mathcal{F}^r}{\mathit{minimize}}\ \mathbb{E}\left[\phi(\boldsymbol{\pi})\right], \tag{4.7}$$

where $\mathcal{F}^r$:

$$\mathcal{F}^r = \left\{ \boldsymbol{\pi} \in [0,1]^{\mathcal{L}\times\mathcal{B}} \middle| \begin{array}{ll} \mathbb{P}(\hat{\rho}_i(\boldsymbol{\pi}_i)) > c_i) \leq \epsilon_i, & \forall i \in \mathcal{B}, \\ \sum_{i\in\mathcal{B}} \pi_i(x) = 1, & \forall x \in \mathcal{L}. \end{array} \right\} \tag{4.8}$$

Definition 5 (Robust User Association Map)**.** *The solution of P2 $\boldsymbol{\pi}^\star$ is called the robust user association map.*

The robust user association map is our novel proposition in this chapter. Using past data, we precompute robust maps that will be used at runtime on traffic to determine the association. The maps are easy to use, while at the same time they provide minimum expected cost, and allow us to control the probability of constraint violation. Hence, it is a disciplined and practical approach to optimize the system using data.

Solving P2 is challenging. There is a great number of optimization variables, increasing with the quantization accuracy for area $\mathcal{L}$. Also, in its current form, Eq.(4.7) is a stochastic program. In the next section we will present an optimal generalized algorithm that overcomes these challenges.

4.3 Generalized Robust Map Algorithm

We present GRMA: a generalized algorithm for solving the Robust User Association Problem P2 when $\phi(\boldsymbol{\pi})$ is a differentiable and separable convex objective function. First, we transform the problem into a convex program by replacing the stochastic constraint Eq.(4.6) with an equivalent convex constraint. In the following, we relax the new constraint; the feasibility set of the relaxed problem is a simplex and we can solve it with an efficient projected gradient algorithm. Finally we present the GRMA algorithm, which is based on a dual subgradient method with averaging on the primal sequence $\boldsymbol{\pi}^{(k)}$ and show that it converges to the optimal robust map.

4.3.1 Convex Formulation

To make P2 a convex program, we will replace the stochastic constraint $\mathbb{P}(\hat{\rho}_i \geq c_i) \leq \epsilon_i$ in $\mathcal{F}^r$ with an equivalent convex constraint.

Lemma 2. *The inequality $\mathbb{P}(\hat{\rho}_i \geq c_i) \leq \epsilon_i$ is equivalent to $\rho_i + \alpha_i S_i \leq c_i$, when $\hat{\rho}_i = \rho_i + Y_i$ is taken according to Eq.(4.4), where Y_i is normally distributed with zero mean and variance $S_i^2(\boldsymbol{\pi}_i)$ and $\alpha_i = Q^{-1}(\epsilon_i)$, where $Q(\cdot)$ is the tail probability of the standard normal distribution.*

Proof. Starting by the probabilistic constraint we have:

$$\mathbb{P}(\hat{\rho}_i \geq c_i) \leq \epsilon_i \Leftrightarrow \mathbb{P}(Y_i \geq c_i - \rho_i) \leq \epsilon_i,$$

Y_i is normally distributed with $Y_i \sim N(0, S_i^2)$. If Q is the Q-function (tail probability of the standard normal distribution), we can rewrite the above equation as:

$$Q(\frac{c_i - \rho_i}{S_i}) \leq \epsilon_i.$$

The inequality is satisfied for all $\frac{c_i - \rho_i}{S_i}$ that:

$$\frac{c_i - \rho_i}{S_i} \geq Q^{-1}(\epsilon_i) \Leftrightarrow \rho_i + Q^{-1}(\epsilon_i) S_i \leq c_i.$$

□

From $Q^{-1}(\epsilon_i)$ it is evident that ϵ_i is a design parameter that affects the feasibility region $\mathcal{F}^r$. Small values of ϵ_i protect from violations, but can lead to inefficient association vectors. Now, we can define the new set $\mathcal{F}^c$ and prove it is convex.

Definition 6 (Convex Feasibility Set $\mathcal{F}^c$)**.**

$$\mathcal{F}^c = \left\{ \boldsymbol{\pi} \in [0,1]^{\mathcal{L}\times\mathcal{B}} \middle| \begin{array}{ll} \rho_i + \alpha_i S_i \leq c_i, & \forall i \in \mathcal{B}, \\ \sum_{i\in\mathcal{B}} \pi_i(x) = 1, & \forall x \in \mathcal{L}. \end{array} \right\} \tag{4.9}$$

Lemma 3. *The constraint $\rho_i + \alpha S_i \leq c_i$ is convex.*

Proof. Consider two vectors $\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 \in \mathcal{F}^c$. We first show that $S_i(\theta\boldsymbol{\pi}_1 + (1-\theta)\boldsymbol{\pi}_2) \leq \theta S_i(\boldsymbol{\pi}_1) + (1-\theta)S_i(\boldsymbol{\pi}_2)$, where $\theta \in [0,1]$. Denote $w(x) = \frac{\sigma_n(x)}{\mu C_i(x)}$. We begin by:

$$\begin{aligned} S_i^2(\theta\boldsymbol{\pi}_1 + (1-\theta)\boldsymbol{\pi}_2) &= \int w^2(x)(\theta\pi_1(x) + (1-\theta)\pi_2(x))^2 dx \\ &= S_i^2(\theta\boldsymbol{\pi}_1) + S_i^2((1-\theta)\boldsymbol{\pi}_2) + 2\theta(1-\theta)\int w^2(x)\pi_1(x)\pi_2(x)dx \end{aligned}$$

while

$$\begin{aligned} (S_i(\theta\boldsymbol{\pi}_1) + S_i((1-\theta)\boldsymbol{\pi}_2))^2 =& S_i^2(\theta\boldsymbol{\pi}_1) + S_i^2((1-\theta)\boldsymbol{\pi}_2) \\ &+ 2S_i(\theta\boldsymbol{\pi}_1)S_i((1-\theta)\boldsymbol{\pi}_2). \end{aligned}$$

From the Cauchy-Swartz[1] inequality we have that:

$$2\theta(1-\theta)\int w^2(x)\pi_1(x)\pi_2(x)dx \leq 2S_i(\theta\boldsymbol{\pi}_1)S_i((1-\theta)\boldsymbol{\pi}_2),$$

hence:

$$\begin{aligned} &S_i^2(\theta\boldsymbol{\pi}_1 + (1-\theta)\boldsymbol{\pi}_2) \leq (S_i(\theta\boldsymbol{\pi}_1) + S_i((1-\theta)\boldsymbol{\pi}_2))^2 \Leftrightarrow \\ &S_i(\theta\boldsymbol{\pi}_1 + (1-\theta)\boldsymbol{\pi}_2) \leq S_i(\theta\boldsymbol{\pi}_1) + S_i((1-\theta)\boldsymbol{\pi}_2) \Leftrightarrow \\ &S_i(\theta\boldsymbol{\pi}_1 + (1-\theta)\boldsymbol{\pi}_2) \leq \theta S_i(\boldsymbol{\pi}_1) + (1-\theta)S_i(\boldsymbol{\pi}_2). \end{aligned}$$

We have proven that S_i is convex, by inspection ρ_i is also convex, and since the sum of positive weighted convex terms is also convex, it follows that the constraint is convex. □

The set $\mathcal{F}^c$ is convex because all the constraints in Eq.(4.9) are convex. Based on Lemmas 2 and 3, and the fact that the expectation of a convex function of a random variable is also convex, we can now recast P2 as a convex program:

[1]Define $f(x) = w(x)\pi_1(x)$ and $g(x) = w(x)\pi_2(x)$, then we have that $|\int f(x)g(x)dx|^2 \leq \int |f(x)|^2 dx \int |g(x)|^2 dx$

Problem 3 (P3: Convex Robust User Association Problem)**.**

$$\underset{\boldsymbol{\pi}\in\mathcal{F}^c}{minimize}\ \mathbb{E}\left[\phi(\boldsymbol{\pi})\right]. \tag{4.10}$$

Hence, in the next subsections we focus on resolving the issue of high dimension optimization with the coupled constraints.

4.3.2 Partial Lagrangian Relaxation

The feasiblity set $\mathcal{F}^c$ is convex, but the constraints couple locations and base stations for every association vector and make the implementation of an efficient algorithm challenging. To efficiently solve this, we propose to relax the load constraint. The remaining feasibility set after the relaxation is the simplex $\mathcal{F}' = \{\boldsymbol{\pi} \in [0,1]^{\mathcal{L}\times\mathcal{B}} | \sum_{i\in\mathcal{B}} \pi_i(x) = 1\}$, and there is rich literature on how to apply projected gradient algorithms in simplices, cf. [74], [75]. Our idea here is to keep as many constraints as possible as long as we know how to project infeasible solutions on them, while we relax the rest. Notice that if we would relax all the constraints, we would get an easy unconstrained convex problem, but the coordination of a large number of Lagrangian multipliers would prohibitively delay the solution. Let us consider the following partial dual maximization, which will be instrumental in solving our problem:

Problem 4 (P4: Partial Dual Robust Problem)**.**

$$\underset{\boldsymbol{\gamma}\geq 0}{maximize}\left\{\underset{\boldsymbol{\pi}\in\mathcal{F}'}{min}\left\{\Phi(\boldsymbol{\pi},\boldsymbol{\gamma})\right\}\right\}, \tag{4.11}$$

where the partially relaxed Lagrangian is:

$$\Phi(\boldsymbol{\pi},\boldsymbol{\gamma}) = \mathbb{E}[\phi(\boldsymbol{\pi})] + \sum_{i\in\mathcal{B}} \gamma_i(\rho_i + \alpha_i S_i - c_i). \tag{4.12}$$

In Eq.(4.12) the vector $\boldsymbol{\gamma}$ contains the Lagrangian multipliers. The multipliers penalize association maps which violate the load constraint with extra cost ($\gamma_i \geq 0$), which increases linearly the more overloaded a base station gets. Henceforth we assume that the Slater's Condition holds, which we expect to be the case for all practical purposes in our problem; therefore, the optimal solution of P4 has equal cost with the optimal primal for the P3 (Strong duality [52]).

4.3.3 Projected Gradient Descent

In this subsection we design an algorithm to efficiently solve the inner minimization subproblem in Eq.(4.11). For a given $\gamma^\star$, we have to find the map that minimizes the cost:

$$\underset{\boldsymbol{\pi}\in\mathcal{F}'}{\text{minimize}}\left\{\Phi(\boldsymbol{\pi},\boldsymbol{\gamma}^\star)\right\}.$$

We design the Projected Gradient Descent (PGD) algorithm to solve this problem motivated by the fact that gradient algorithms have been shown in the literature to have independent convergence rate from the dimension (number of variables) of the problem [76, Ch. 3]. Also, the projection onto $\mathcal{F}'$ (simplex) can be solved exactly and efficiently. The algorithm is:

Projected Gradient Descent (PGD) on $\mathcal{F}'$

```
Initialize: $\boldsymbol{\pi}^{(0)}$ (can be infeasible), $\boldsymbol{\gamma}^\star$.
Iterate: over n, until convergence
    $\boldsymbol{y}^{(n+1)} = \boldsymbol{\pi}^{(n)} - \eta^{(n)} \nabla_{\boldsymbol{\pi}} \Phi(\boldsymbol{\pi}^{(n)}, \boldsymbol{\gamma}^\star)$        (4.13)
    $\boldsymbol{\pi}^{(n+1)} = \Pi_{splx}[\boldsymbol{y}^{(n+1)}]$        (4.14)
Where $\Pi_{splx}$ is the Euclidean projection on $\mathcal{F}'$:
    Sort $\boldsymbol{y}^{(n+1)}$ in descending order $(y_1 \geq y_2 \geq \ldots \geq y_{|\mathcal{B}|})$
    Select $m = \underset{j \in \mathcal{B}}{\text{argmax}} \{ j \mid y_j + \frac{1}{j}(1 - \sum_{i=1}^{j} y_i) > 0 \}$
    $\pi_i^{(n+1)} = \left[ y_i + \frac{1}{m}(1 - \sum_{i=1}^{m} y_i) \right]^+, \ i = 1, \ldots, |\mathcal{B}|$
```

Eq.(4.13) implements the gradient update of the user association $\boldsymbol{\pi}^{(n)}$ one step along the direction of the gradient with fixed step size $\eta^{(n)}$. Eq.(4.14) is the Euclidean projection of $\boldsymbol{y}^{(n+1)}$ onto the set $\mathcal{F}'$. Π_{splx}, as described here, is shown in [75] to give an exact solution to the projection in $\mathcal{O}(|\mathcal{B}| \log |\mathcal{B}|)$.

Proposition 1 (Convergence Rate of PGD). *Let $\boldsymbol{\pi}^{(n)}$ be the projected output of PGD algorithm at iteration n, and $\boldsymbol{\pi}^\star$ be an optimal solution of* (4.3.3)*, it is shown in [76, Ch. 3.2]:*

$$||\boldsymbol{\pi}^{(n)} - \boldsymbol{\pi}^\star|| = \mathcal{O}(1/n).$$

4.3.4 Dual Subgradient Method

We return to the task of solving P4. The objective of this problem $D(\boldsymbol{\gamma}) = \min_{\boldsymbol{\pi} \in \mathcal{F}'} \{\Phi(\boldsymbol{\pi}, \boldsymbol{\gamma})\}$ is not differentiable everywhere, hence we will resort to a subgradient method [53], [77] for updating the value of the multipliers.

Proposition 2 (Subgradient Vector). *The vector:*

$$\boldsymbol{g}^{(k)} = \boldsymbol{\rho}(\boldsymbol{\pi}^{(k)}) + \boldsymbol{\alpha} \circ \boldsymbol{S}(\boldsymbol{\pi}^{(k)}) - \boldsymbol{c},$$

where $\circ$ is the hadamard (entrywise) product, and

$$\boldsymbol{\pi}^{(k)} \in \underset{\boldsymbol{\pi} \in \mathcal{F}'}{argmin} \left\{ \mathbb{E}[\phi(\boldsymbol{\pi})] + \sum_{i \in \mathcal{B}} \gamma_i^{(k)} (\rho_i + \alpha_i S_i - c_i) \right\}, \tag{4.15}$$

satisfies $||D(\boldsymbol{\gamma}^{(k+1)}) - D(\boldsymbol{\gamma}^{(k)})|| \leq \boldsymbol{g}^{(k)} ||\boldsymbol{\gamma}^{(k+1)} - \boldsymbol{\gamma}^{(k)}||$ as shown in [53, Ch. 6.1], hence is a subgradient of $D(\boldsymbol{\gamma})$ at $\boldsymbol{\gamma}^{(k)}$.

We now show that the norm of the subgradients is bounded; a necessary property for the convergence of the method.

Lemma 4 (Bounds on the Subgradient)**.** *The subgradient sequence* $\{\boldsymbol{g}^{(k)}\}$ *is bounded:*

$$||\boldsymbol{g}^{(k)}|| < L,$$

where

$$L = \sqrt{\sum_{i\in\mathcal{B}} \left(\int_{\mathcal{L}} \frac{\lambda(x)}{\mu C_i(x)} dx + \alpha_i \left(\int_{\mathcal{L}} \frac{\sigma_n^2(x)}{\mu^2 C_i(x)^2} dx \right)^{\frac{1}{2}} \right)^2}. \tag{4.16}$$

Proof. The set $\mathcal{F}'$ is a simplex (compact). The functions $\boldsymbol{g}_i,\ i \in \mathcal{B}$ are convex over $\mathbb{R}^n$, hence they are continuous over $\mathbb{R}^n$. The norm of the subgradients is upper bounded by $\max_{\boldsymbol{\pi}\in\mathcal{F}'} ||\boldsymbol{g}(\boldsymbol{\pi})||$. This is smaller than assigning all the locations $x \in \mathcal{L}$ to all base stations, hence the norm of the subgradients is bounded by the easy to calculate Eq.(4.16). $\square$

The subgradient method updates the multipliers $\boldsymbol{\gamma}^{(k)}$ by making a step along the direction of the subgradient vector:

$$\boldsymbol{\gamma}^{(k+1)} = [\boldsymbol{\gamma}^{(k)} + s^{(k)}\boldsymbol{g}^{(k)}]^+. \tag{4.17}$$

For dual problems with a unique solution, the above algorithm converges to the unique optimal dual vector $\boldsymbol{\gamma}^\star$, and with this we can calculate the optimal robust map $\boldsymbol{\pi}^\star$ by a single run of PGD algorithm. However, P4 is not strictly convex, and therefore its dual may have multiple solutions. The subgradient method may converge to a solution $(\boldsymbol{\pi}, \boldsymbol{\gamma})$ which does not satisfy complementary slackness and hence $\boldsymbol{\pi}$ is not feasible in P3 (it will violate the load constraint). To alleviate this issue we will use the technique of averaging: the idea is to output as a solution the average of the primal iterates $\boldsymbol{\pi}^{(k)}$ (feasible or not). We will show that the sequence of averages $\bar{\boldsymbol{\pi}}^{(k)}$ converges to the optimal solution of P3.

Generalized Robust Map Algorithm (GRMA)

```
**Initialize:** $\boldsymbol{\pi}^{(0)}$ (e.g. *MaxSINR*, can be infeasible), $\boldsymbol{\gamma}^{(0)}$.
**Iterate:** over k, until convergence:

    $\boldsymbol{\gamma}^{(k+1)} = [\boldsymbol{\gamma}^{(k)} + s^{(k)}\boldsymbol{g}^{(k)}]^+$
    $\boldsymbol{\pi}^{(k+1)} \leftarrow \mathrm{PGD}(\boldsymbol{\gamma}^{(k+1)})$
*Keep the running average of the* $\boldsymbol{\pi}^{(k)}$ *(Eq.(4.15)):*
$\bar{\boldsymbol{\pi}}^{(k)} = \frac{1}{k}\sum_{i=0}^{k-1} \boldsymbol{\pi}^{(i)}$
```

Theorem 1 (Convergence to Primal Optimal). *The average of the primal iterates* $\bar{\boldsymbol{\pi}}^{(k)} = \frac{1}{k}\sum_{i=0}^{k-1} \boldsymbol{\pi}^{(i)}$, *where*

$$\boldsymbol{\pi}^{(i)} \in \underset{\boldsymbol{\pi}\in\mathcal{F}'}{argmin} \left\{ \mathbb{E}[\phi(\boldsymbol{\pi})] + \sum_{j\in\mathcal{B}} \gamma_j^{(i)}(\rho_j + \alpha_j S_j - c_j) \right\}, \tag{4.18}$$

asymptotically converges to (or approximates) the optimal robust association map $\boldsymbol{\pi}^\star$, *i.e.:*

$$\lim_{k\to\infty} ||g(\bar{\boldsymbol{\pi}}^{(k)})^+|| \to 0 \textit{ and } \lim_{k\to\infty} \mathbb{E}[\phi(\bar{\boldsymbol{\pi}}^{(k)})] = \mathbb{E}[\phi(\boldsymbol{\pi}^\star)].$$

Proof. We use a constant step size, hence $s^{(k)} = s$. We also denote $\boldsymbol{\gamma}^\star$ as the optimal multipliers and

$$d^\star = \max_{\gamma\geq 0} \left\{ \min_{\boldsymbol{\pi}\in\mathcal{F}'} \{\Phi(\boldsymbol{\pi},\gamma\} \right\}$$

is the optimal value of the dual problem. First we prove that the load constraint violation for the vector $\bar{\boldsymbol{\pi}}^{(k)}$ is upper bounded as follows:

$$||\boldsymbol{g}(\bar{\boldsymbol{\pi}}^{(k)})^+|| \leq \frac{||\gamma^{(k)}||}{ks}. \tag{4.19}$$

By updating the dual as described in Eq.(4.17), we have:

$$s\boldsymbol{g}(\boldsymbol{\pi}^{(k)}) \leq \boldsymbol{\gamma}^{(k+1)} - \boldsymbol{\gamma}^{(k)}, \ \forall k \geq 0.$$

Summing telescopically for $i = 0, 1, \dots, k-1$ we get:

$$\sum_{i=0}^{k-1} s\boldsymbol{g}(\boldsymbol{\pi}^{(i)}) \leq \boldsymbol{\gamma}^{(k)} - \boldsymbol{\gamma}^{(0)} \leq \boldsymbol{\gamma}^{(k)}, \ \forall k \geq 1. \tag{4.20}$$

Also, since $\boldsymbol{g}(\bar{\boldsymbol{\pi}}^{(\boldsymbol{k})})$ is convex, we have that:

$$\begin{aligned}\boldsymbol{g}(\bar{\boldsymbol{\pi}}^{(\boldsymbol{k})}) &= \boldsymbol{g}(\frac{1}{k}\sum_{i=0}^{k-1}\boldsymbol{\pi}^i) \leq \frac{1}{k}\sum_{i=0}^{k-1}\boldsymbol{g}(\boldsymbol{\pi}^{(i)}) \\ &= \frac{1}{ks}\sum_{i=0}^{k-1} s\boldsymbol{g}(\boldsymbol{\pi}^{(i)}) \overset{Eq.(4.20)}{\leq} \frac{\boldsymbol{\gamma}^{(k)}}{ks}.\end{aligned}$$

Taking norms for the active constraints $(\boldsymbol{g}(\bar{\boldsymbol{\pi}}^{(k))} \geq 0)$ gives Eq.(4.19). Since the Lagrangian multipliers are bounded [77, Lem.3], the first result follows. Next, we will prove that the objective function for the vector $\bar{\boldsymbol{\pi}}^{(k)}$ is upper bounded by:

$$\mathbb{E}[\phi(\bar{\boldsymbol{\pi}}^{(k)})] \leq d^\star + \frac{||\boldsymbol{\gamma}^{(0)}||^2}{2ks} + \frac{s}{2k}\sum_{i=0}^{k-1} ||\boldsymbol{g}(\boldsymbol{\pi}^{(i)})||^2. \tag{4.21}$$

From Eq.(4.17):

$$||\boldsymbol{\gamma}^{(i+1)}||^2 \leq ||\boldsymbol{\gamma}^{(i)}||^2 + s^2||\boldsymbol{g}(\boldsymbol{\pi}^{(i)})||^2 + 2s\boldsymbol{\gamma}^{(i)\top}\boldsymbol{g}(\boldsymbol{\pi}^{(i)}) \Leftrightarrow$$
$$-\boldsymbol{\gamma}^{(i)\top}\boldsymbol{g}(\boldsymbol{\pi}^{(i)}) \leq \frac{||\boldsymbol{\gamma}^{(i)}||^2 - ||\boldsymbol{\gamma}^{(i+1)}||^2 + s^2||\boldsymbol{g}(\boldsymbol{\pi}^{(i)})||^2}{2s}.$$

By taking the telescoping sum we have:

$$\begin{aligned} -\frac{1}{k}\sum_{i=0}^{k-1}\boldsymbol{\gamma}^{(i)\top}\boldsymbol{g}(\boldsymbol{\pi}^{(i)}) &\leq \frac{||\boldsymbol{\gamma}^{(0)}||^2 - ||\boldsymbol{\gamma}^{(k)}||^2}{2ks} + \frac{s}{2k}\sum_{i=0}^{k-1}||\boldsymbol{g}(\boldsymbol{\pi}^{(i)})||^2 \\ &\leq \frac{||\boldsymbol{\gamma}^{(0)}||^2}{2ks} + \frac{s}{2k}\sum_{i=0}^{k-1}||\boldsymbol{g}(\boldsymbol{\pi}^{(i)})||^2. \end{aligned} \tag{4.22}$$

As before, since $\phi(\boldsymbol{\pi})$ is convex:

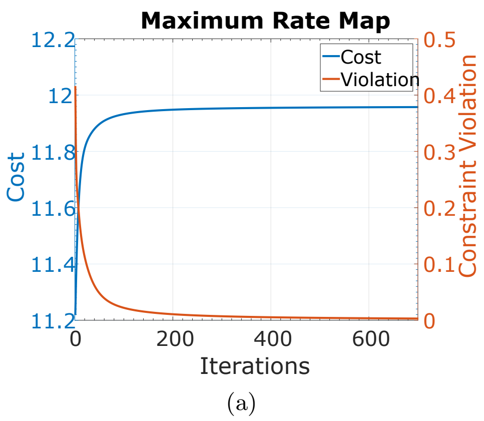
$$\begin{aligned} \mathbb{E}[\phi(\bar{\boldsymbol{\pi}}^{(k)})] &= \mathbb{E}\left[\phi(\frac{1}{k}\sum_{i=0}^{k-1}\boldsymbol{\pi}^{(i)})\right] \leq \frac{1}{k}\sum_{i=0}^{k-1}\mathbb{E}[\phi(\boldsymbol{\pi}^{(i)})] \\ &= \frac{1}{k}\sum_{i=0}^{k-1}\left\{\mathbb{E}[\phi(\boldsymbol{\pi}^{(i)})] + \boldsymbol{\gamma}^{(i)\top}\boldsymbol{g}(\boldsymbol{\pi}^{(i)}) - \boldsymbol{\gamma}^{(i)\top}\boldsymbol{g}(\boldsymbol{\pi}^{(i)})\right\} \\ &\overset{Eq.(4.18)}{=} \frac{1}{k}\sum_{i=0}^{k-1}D(\boldsymbol{\gamma}^{(i)}) - \frac{1}{k}\sum_{i=0}^{k-1}\boldsymbol{\gamma}^{(i)\top}\boldsymbol{g}(\boldsymbol{\pi}^{(i)}) \\ &\leq d^\star - \frac{1}{k}\sum_{i=0}^{k-1}\boldsymbol{\gamma}^{(i)\top}\boldsymbol{g}(\boldsymbol{\pi}^{(i)}) \\ &\overset{Eq.(4.22)}{\leq} d^\star + \frac{||\boldsymbol{\gamma}^{(0)}||^2}{2ks} + \frac{s}{2k}\sum_{i=0}^{k-1}||\boldsymbol{g}(\boldsymbol{\pi}^{(i)})||^2. \end{aligned}$$

The second line is true because $\boldsymbol{\pi}^{(i)}$ is a minimizer of the partial Lagrangian according to Eq.(4.18), hence $\Phi(\boldsymbol{\pi}^{(i)}, \boldsymbol{\gamma}^{(i)}) = D(\boldsymbol{\gamma}^{(i)})$. By letting $k \to \infty$ on Eq.(4.19) shows that $\bar{\boldsymbol{\pi}}^{(k)}$ is feasible and on Eq.(4.21) shows that $\mathbb{E}[\phi(\bar{\boldsymbol{\pi}}^{(k)})] \leq d^\star$. Since $d^\star \leq \mathbb{E}[\phi(\boldsymbol{\pi})]$, $\forall \boldsymbol{\pi} \in \mathcal{F}^c$, the result follows. $\square$

4.3.5 Example Applications of GRMA

First, we consider the maximum expected rate objective, where the optimal map will associate every location x to the base stations that provide the highest physical rate $C_i(x)$. This, according to Eq.(4.1) is identical to minimizing the sum of loads. Hence, taking expectation of the cost of the actual load $\hat{\rho}$:

$$\mathbb{E}_Y\left[\sum_{i\in\mathcal{B}}\hat{\rho}_i\right] = \sum_{i\in\mathcal{B}}\mathbb{E}_Y[\rho_i + Y_i] = \sum_{i\in\mathcal{B}}\rho_i.$$



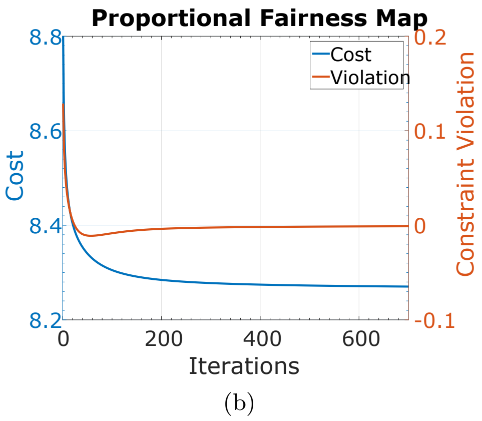


Figure 4.3: Convergence on cost and violation of GRMA for different objectives (a) max rate (b) proportional fairness.

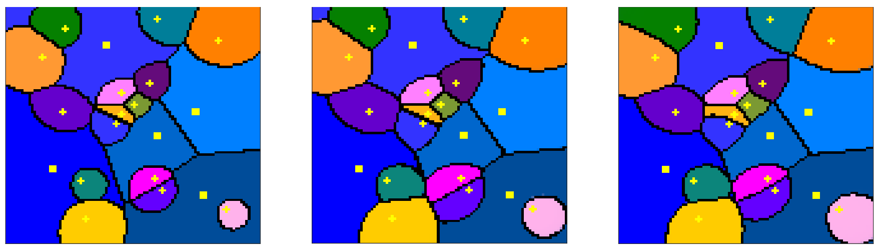


Figure 4.4: Examples of User Association maps, colors and borders indicate areas covered by each base station. (a) MaxSINR Cells (b) robust maximum rate map ($\epsilon = 0.05$) (c) robust proportional fair map ($\epsilon = 0.05$).

Robust Map 1 (RM1: Maximum Expected Rate Map)**.** The optimal maximum expected rate map $\pi^\star$ is the solution of:

$$\underset{\boldsymbol{\pi}\in\mathcal{F}^c}{\text{minimize}}\{\sum_{i\in\mathcal{B}}\rho_i\}.$$

Next we consider the *penalty proportional fairness* associated to the objective $\phi(\boldsymbol{\pi}) = \sum_i \frac{\rho_i^2}{2}$ [72, 78]. This objective function leads to a load balancing trade-off, where base stations in high traffic areas are allowed to be more loaded in the benefit of higher total throughput. Taking expectation:

$$\mathbb{E}_Y\left[\sum_{i\in\mathcal{B}}\frac{(\rho_i+Y_i)^2}{2}\right]=\sum_{i\in\mathcal{B}}\frac{\mathbb{E}[\rho_i^2+S_i^2+2\rho_iS_i]}{2}$$
$$=\sum_{i\in\mathcal{B}}\frac{\rho_i^2+S_i^2}{2}.$$

Robust Map 2 (RM2: Proportional Fair Map)**.** The optimal proportional fair map $\pi^\star$ is given by:

$$\underset{\boldsymbol{\pi}\in\mathcal{F}^c}{\text{minimize}}\left\{\sum_{i\in\mathcal{B}}\frac{\rho_i^2+S_i^2}{2}\right\}.$$

Figure 4.3 shows the progress in iterations towards convergence and feasibility of GRMA and Figure 4.4 shows an illustration of the optimal robust map produced when applied on a network setup of heterogeneous base stations in an area with highly variable and dense traffic. In Fig.4.3(a) and 4.3(b), we can see that for both objectives GRMA produces feasible solutions with almost optimal cost after 200 iterations. In the maximum rate map Fig.4.4(b), we can see the similarities with the MaxSINR cells Fig.4.4(a), at locations where the expected traffic is low, while at heavy traffic locations we have curved boundaries, enforced by the load protection constraint. In proportional fair map Fig.4.4(c), the cells are very different from the other cases.

Note that for higher order α-fair functions ($\alpha > 2$), as well as other objective functions, the calculation of the gradient of the expectation is highly complicated. We propose using the Approximate Robust Maps in the following section.

4.4 Approximate Robust Maps

In this section we expand the framework to problem formulations in which calculating the gradient of the expectation is prohibitively complicated. The expectation for a convex, separable and differentiable $\phi=\sum_{i\in\mathcal{B}}\varphi(\rho_i(\boldsymbol{\pi}_i)+Y_i(\boldsymbol{\pi}_i))$ is:

$$\mathbb{E}_{Y_i}[\varphi(\rho_i+Y_i)]=\int_{-\infty}^{\infty}\mathbb{P}(Y_i=y_i,\boldsymbol{\pi}_i)\varphi(\rho_i+y_i)dy_i, \tag{4.23}$$

where $Y_i\sim N(0,S_i^2(\boldsymbol{\pi}_i))$ and

$$\mathbb{P}(Y_i=y_i,\boldsymbol{\pi}_i)=\frac{e^{-\frac{y_i^2}{2S_i^2(\boldsymbol{\pi}_i)}}}{\sqrt{2\pi S_i^2(\boldsymbol{\pi}_i)}}.$$

In Eq.(4.23), we can see that to calculate the gradient we need to consider both the rate of change of the objective function φ and also the effect of changing the distribution of Y_i due to selecting $\boldsymbol{\pi}_i$.

We can counter the tough derivative calculations by approximating the objective function, we will discuss three techniques. First, we can use the Taylor series expansion of the expectation, second we can use convexity and Jensen's Inequality to have a lower bound approximation (MCEL) and third, we can minimize the worst case expected cost (MWCC) (excluding the ϵ probability tail, as in the robust constraint). The technique of MWCC produces solutions which guarantee that the cost in operation will be less than the cost of the optimal solution of MWCC ($f^{Real}(\boldsymbol{\pi}^\star_{MWCC}) \leq f^{MWCC}(\boldsymbol{\pi}^\star_{MWCC})$) inside the confidence interval. Hence, MWCC is risk averse towards high cost due to low probability events at the tail of the distribution.

4.4.1 Taylor Approximation on RUA

The first approach we consider is the Taylor series expansion of the objective function around the average load $\mathbb{E}_Y[\hat{\boldsymbol{\rho}}] = \boldsymbol{\rho}$. Since the support of the distribution of Y_i is infinite, the series can be shown not to be convergent and thus approximating by keeping the first terms of the series has no guarantee. However, we argue that we will rarely deviate 'a lot' from the average (predicted) load, while deviating 'a lot' usually means that the problem is infeasible. Expanding at ρ_i we get:

$$\begin{aligned}\mathbb{E}_{Y_i}\left[\varphi(\rho_i + Y_i)\right] &= \mathbb{E}_{Y_i}\left[\sum_{n=0}^{\infty} \frac{\varphi^{(n)}(\rho_i)}{n!}(Y_i)^n\right] \\ &= \sum_{n=0}^{\infty} \frac{\varphi^{(n)}(\rho_i)}{n!} E[Y_i^n]. \qquad (4.24)\end{aligned}$$

Keeping the 2 first non zero terms we get:

$$\mathbb{E}_{Y_i}\left[\varphi(\rho_i + Y_i)\right] \approx \varphi(\rho_i) + \frac{\varphi^{(2)}(\rho_i)}{2} S_i^2. \qquad (4.25)$$

We can calculate the derivatives now and solve the problem with this objective function to a local minima[2], with the GRMA. We show in the numerical section that this is in practice the best approach[3].

4.4.2 Minimize the Cost of the Expected Load (MCEL)

The calculation of the gradients can be simplified by minimizing the cost of the expected load of the objective function, instead of minimizing the expected

[2]The function in Eq.(4.25) is of the type $f(x)g(x)$, which is most of the times non-convex even if $f(x)$ and $g(x)$ are convex.

[3]A heuristic that works well in practice is to start from an optimal point which is calculated by one of the other approximate convex methods.

cost. This is a known lower bound for a convex function $\phi(\mathbb{E}_Y[\boldsymbol{\rho}+\boldsymbol{Y}]) \leq \mathbb{E}_Y[\phi(\boldsymbol{\rho}+\boldsymbol{Y})]$, called Jensen's Inequality. Since $Y_i \sim N(0, S_i^2)$, we have:

$$\underset{\boldsymbol{\pi}\in\mathcal{F}^c}{\text{minimize}}\{\phi(\boldsymbol{\rho}(\boldsymbol{\pi}))\},$$

this problem is equivalent to the zero order Taylor Approximation Eq.(4.24).

4.4.3 Minimize the Worst Case Cost (MWCC)

The objective is to minimize the cost that corresponds to the worst case uncertainty for the resulting robust user association map. In section 4.3.1 we have shown that we can bound the uncertainty set to $U = (-\infty, Q^{-1}(\epsilon_i)S_i]^B$ with probability ϵ_i and that for a given the vector $\boldsymbol{\pi}_i$ the variance of $\hat{\rho}_i$ is $S_i^2 = \int_{\mathcal{L}} \frac{\pi_i^2(x)}{\mu^2 C_i^2(x)} \sigma_n^2(x)dx$. The formulation of the optimization problem is:

$$\underset{\boldsymbol{\pi}\in\mathcal{F}^c}{\text{minimize}} \left\{ \sup_{y\in\mathcal{U}}\{\phi(\boldsymbol{\rho}+\boldsymbol{y})\} \right\}.$$

Assuming $\phi = \sum_i \varphi(\rho_i + y_i)$ and φ is a non-decreasing positive function for $\forall\boldsymbol{\pi} \in \mathcal{F}^c$, we have that:

$$\sup_{y\in\mathcal{U}}\{\phi(\boldsymbol{\rho}+\boldsymbol{y})\} = \phi(\boldsymbol{\rho}+\sup_{y\in\mathcal{U}}\{\boldsymbol{y}\}).$$

We define $\boldsymbol{R}$ vector with $R_i = \rho_i + \sup\limits_{y\in\mathcal{U}}\{y_i\} = \rho_i + Q^{-1}(\epsilon_i)S_i$. Hence the optimization boils down to:

$$\underset{\boldsymbol{\pi}\in\mathcal{F}^c}{\text{minimize}}\{\phi(\boldsymbol{R}(\boldsymbol{\pi}))\}.$$

Lemma 5. *The function $\phi(\boldsymbol{R}(\boldsymbol{\pi}))$ is convex.*

Proof. Convexity of $\phi(\boldsymbol{R}(\boldsymbol{\pi}))$ can be verified by composition rules. $\boldsymbol{R}(\boldsymbol{\pi})$ is convex, since it is a sum of convex functions, $\phi(\boldsymbol{R})$ is convex and non-decreasing, hence the composition $\phi(\boldsymbol{R}(\boldsymbol{\pi}))$ is convex. □

4.4.4 Evaluation of Approximate Robust Maps

Here, we will evaluate the approximate robust maps with the family of α-optimal functions. The α-optimal cost functions, defined below, have the nice property of producing vectors of solutions that optimize a desirable performance metric at the base stations [19]. For $\alpha \in [0, \infty)$, we define the family of convex functions called α-optimal functions parametrized by α:

$$\phi_\alpha(\boldsymbol{\rho}) = \begin{cases} \sum_i \frac{(1-\rho_i)^{1-\alpha}}{\alpha-1} & \alpha \neq 1. \\ \sum_i \log(\frac{1}{1-\rho_i}) & \alpha = 1. \end{cases} \tag{4.26}$$

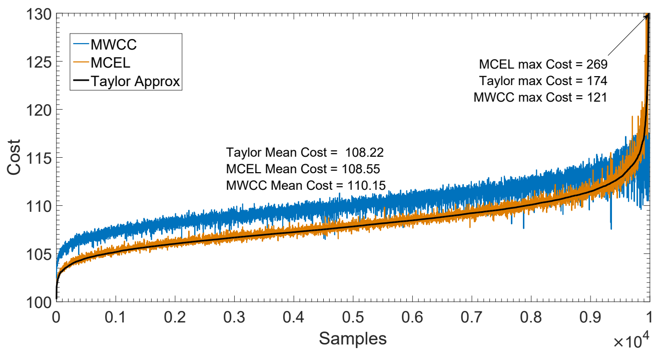


Figure 4.5: Cost plot of approximate methods, robust maps are generated for each of the approximate methods for the delay minimization objective ($\alpha = 2$), 10000 samples are generated based on the prediction of traffic for a peak hour in the dataset, samples are ordered by taylor approximation cost.

Table 4.2: Expected Performance for different α objective functions

α	Load Variance	Mean Load	Maximum Load	Mean Delay (s)	Maximum Delay (s)
0	0.0970	**0.44**	0.94	2.52	12.98
1	0.0873	0.45	0.93	2.24	7.77
2	0.0663	0.49	0.85	**1.97**	**4.45**
5	**0.0103**	0.66	**0.82**	3.57	8.58

For example, as shown in [19], by selecting: (i) $\alpha = 0$, we maximize the expected rate of the users (Robust Map 1 - Section 4.3.5), (ii) $\alpha = 1$ we maximize the effective throughput of the base stations (iii) for $\alpha = 2$ we minimize the number of users in the base stations and hence the overall delay, and (iv) for $\alpha = \infty$ we achieve max-min load fairness, described in [32]. Notice that to calculate the gradients for $\mathbb{E}_Y[\phi_\alpha(\boldsymbol{\pi})]$ we use the techniques described in Sect.4.4-A-C.

For Fig.4.5, we generate 10^4 samples of traffic ($\boldsymbol{\lambda}_i$), based on the predicted traffic and its distribution for a peak hour (Wednesday at 13h) in the $\mathcal{L}$ Milano area. For each realization of traffic, we calculate the cost incurred by the policies derived from each approximation method. In the figure, we can see that MWCC incurs extra cost for most of the realizations, but for the extreme 0.2% it provides an efficient cost, while the other policies explode. Taylor approximation, initialized from an optimal point of MWCC policy, has the best average performance. We note that the feasibility set is common for all the methods $\mathcal{F}^c$ and all the points in Fig.4.5 are feasible. In table 4.2, we summarize important metrics for the expected performance of approximate robust maps calculated with different parameter α, for the same peak hour.

4.5 Numerical Evaluation

4.5.1 Simulation Setup

We will experiment on the Milano dataset [33], which provides spatially aggregated data about the telecommunication activity. The data are grouped on a regular grid overlaying the territory of Milano with 100×100 squares. Consequently, the grid designates the area $\mathcal{L}$ and every square is a location $x \in \mathcal{L}$ to be associated with base stations. For every square of this grid the data set contains the aggregate per ten minutes telecommunication events in the period of 01/11/13-01/01/14.

In this work we consider weekdays (Monday to Friday) which are non-holidays, since then the volume of traffic is increased. Our method targets very small cells, where traffic (spatio-)temporal variability will be significantly higher than today's macro-cells; hence we focused on instances where the variability is large enough to resemble what one would see on a regular basis in denser, smaller cells. On the other hand, our framework can also be used for holidays, and other rare but predictable occasions, like a football match or a concert, for which network operators can reserve a special map, tuned to an exceptional increase/decrease in predicted traffic.

We choose an architected setup of 40 base stations with fixed positions, spread over the area, with higher density on the area that is the city center. We specifically design this subset of base stations, to accurately simulate a simplified environment of a UDN, bringing in the front all the aspects of the user association problem. In the simulations scenarios we will consider two alternative base station setups. One, in which all the traffic is served by the small cells, as envisioned for future 5G UDNs, cf. [6], and one is with the two tier structure, which is dominant in current networks. The LTE parameters used in the simulation are given in table 4.3, also found in [20].

Table 4.3: Simulation Parameters

Parameter	Variable	Value
Transmission Power Macro BS	P_M	43 dbm
Transmission Power Micro BS	P_m	33 dbm
System Bandwidth	W	10 MHz
Noise Density	No	-174dbm/Hz
Path Loss Exponent	P_{lo}	3

4.5.2 Robust Map 1 vs MaxSINR

Here, we will compare the maximum expected rate maps (RM1) to the policy that is implemented currently in practice, which is to associate users to the strongest signal (MaxSINR). The accrued cost is the value of the

Table 4.4: Robust Map 1 vs MaxSINR

$\epsilon = 0.001$	Average Cost	Violations (%)
MaxSINR	12.38	72.1
RM1	12.75	0.3

objective function, based on the actual input at that time slot and on the user association policies currently active. Finally, we count violations as the percentage of time in which the system has an or some overloaded base stations; this happens when either an SLA with a load threshold c_i is violated or when some of the base stations are overcumbered by traffic ($\rho_i > 1$).

In the considered model MaxSINR is a static map assigning every location to the base station that provides the maximum rate, disregarding load or traffic changes. Meanwhile, Robust Map 1 (RM1) will make sure that the base stations are protected by overload with ϵ_i, hence will produce different maps for every hour based on the predicted traffic and its variance. From table 4.4, we can see that RM1, vastly outperforms MaxSINR, increasing slightly the average cost (3%), while protecting the SLA.

4.5.3 Robust Map 2 vs Adaptive Algorithm

In this subsection, we compare the proportional fair maps (RM2) to an adaptive version of a popular user association algorithm from the literature [19], on the proportional fairness objective (RM2). At every network update (10 minutes), the adaptive algorithm calculates the average load experienced on the previous slot and settles to a new association vector, while on the other hand we apply our precalculated map for every hour. We add average delay to our metrics, which is the average response time for a flow in the network.

In the first experiment we focus on the choice of ϵ (we choose $\epsilon_i = \epsilon,\ \forall i \in \mathcal{B}$) and the effect it has to the performance of a robust map. In theory the choice of smaller ϵ shrinks the feasibility space, allowing only maps that provide an ϵ probability protection guarantee (Eq.(4.6)). This should correlate with the violation metric in the results, where we also expect slightly increased cost due to eliminating cheaper but riskier configurations. This behavior is well observed in the results.

Table 4.5: Aggregate Results 1st week of December Micro Setup

ϵ	Average Cost	Average Delay (s)	Violations (%)
Adaptive	6.001	1.700	13.1
10%	6.462	1.353	2.3
5%	6.485	1.319	1.6
0.1%	6.596	1.267	0

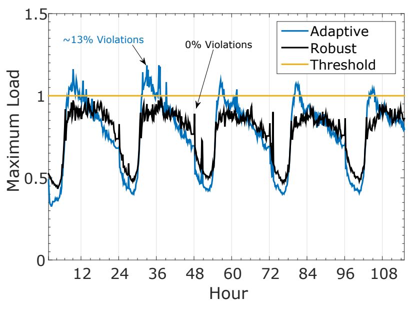
Table 4.6: Peak Traffic 1st week of December Micro Setup

ϵ	Average Cost	Average Delay (s)	Violations (%)
Adaptive	9.671	2.858	24.6
10%	10.911	2.251	7.7
5%	11.000	2.140	3.1
0.1%	11.415	2.030	0

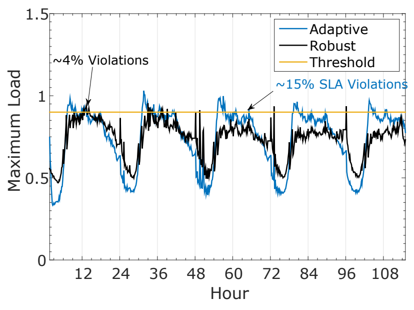
In tables 4.5, 4.6, we present performance results for different values of ϵ during rush hours and in general. We observe that the robust maps typically incur a small increase of cost ($< 10\%$ in average, and $< 18\%$ during peak hours) with respect to the adaptive algorithm. On the other hand, the robust approach provides extraordinary guarantees against traffic fluctuations. In particular, we observe a significantly better average delay ($\approx 30\%$ better) and much less violations (0 instead of 25% of the adaptive algorithm). We also observe that selecting a more relaxed $\epsilon = 5\%, 10\%$, reduces the cost, but deteriorates the performance with respect to average delay and violations of the max load in the duration of the simulation.

Moreover, in the left column of Fig.4.6 we depict the time evolution of the system under the two considered approaches (the robust map and the adaptive algorithm). Comparing the two approaches in this scenario, we see that the robust maps yield 0% violations vs 13% of the adaptive algorithm, this improvement leads to a much better delay performance. Notice that the points with 0 average delay in Fig.4.6c correspond to load constraint violations in Fig.4.6a and thus to infeasible points ($\rho > 1$ at some base stations). Additionally, the improvements in the performance guarantees come at a very small cost increase Fig.4.6e. Remarkably our scheme incurs no extra cost when the traffic is low, a benefit that arises from using different maps per hour and exploiting the past data for prediction. Last, the right column of Fig.4.6 presents a scenario in which we enforce SLAs, in the form of a cap on the base station loads ($\rho_i \leq 0.9$). In practice, SLA violations are extremely important and must be avoided at all costs. The results of Fig.4.6 show how our robust maps ($\epsilon = 5\%$) protect the SLA from violations, resulting in only 4% violations instead of 15% of the adaptive algorithm; this would be further improved by a more conservative ϵ.

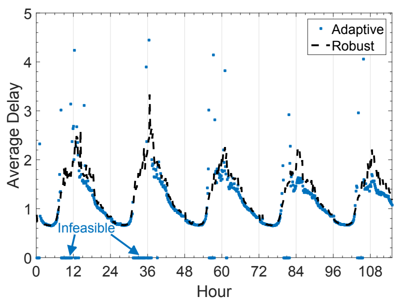
In both time evolution figures we can see that the adaptive algorithms have a natural way of adapting to fluctuations, however this takes a lot of time and in the meantime the system tends to exhibit unstable behavior. Finally, the robust maps pay an increased optimization cost (we have already argued that is of lesser significance than SLA failures) for being conservative against these failures. This is more evident during peak traffic hours, where the more conservative handling of the robust maps infers greater cost, but also improved delay performance and protection against the fluctuating traffic, for example see Fig.4.6(c) and Fig.4.6(a) arround hour 36.



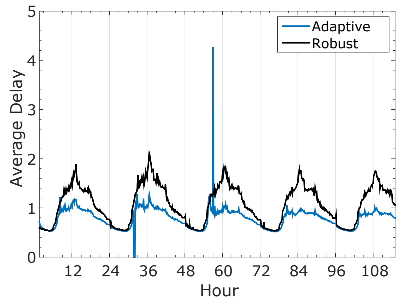
(a)



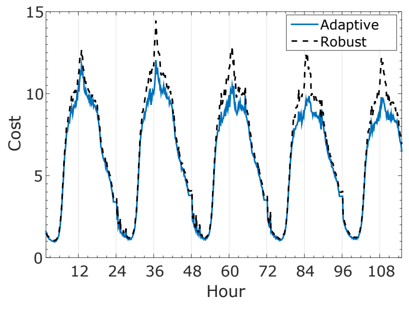
(b)



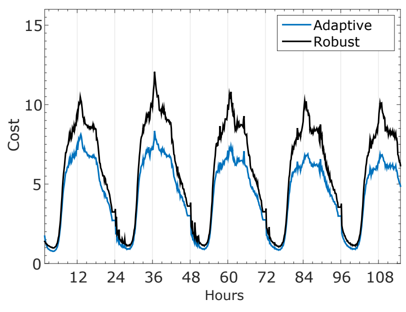
(c)



(d)



(e)



(f)

Figure 4.6: Left Column: Micro Base Station Setup, strong SLA protection $\epsilon = 0.001$, 2-6 of December (a) Violation (c) Average System Delay (e) Cost. Right Column: 2-Tier Base Station Setup, light SLA protection $\epsilon = 0.05$, 2-6 of December (b) Violation (d) Average System Delay (f) Cost.

4.6 Advanced Prediction Methods

In this section we use state of the art methods for time series forecast to get our estimators (Eq.(4.3)), which translates to more accurate estimation of the mean (Eq.(4.4)) and variance (Eq.(4.5)) of the aggregate load. Specifically we use: (i) the analytical Seasonal AutoRegressive Integrated Moving Average methods (SARIMA) [34] and (ii) the Long Short-Term Memory (LSTM) Neural Networks [35]. We present simulation results on the Milano data [33] to validate the gaussian error model (Eq.(4.3)) and to quantify the gains from using improved predictions as input to GRMA.

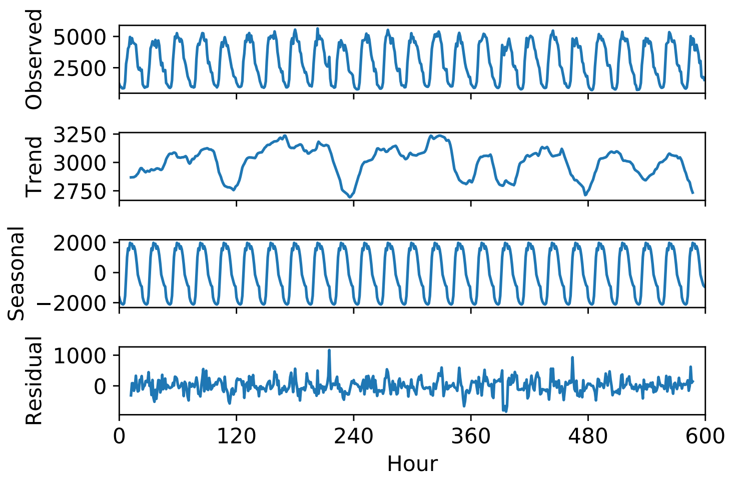
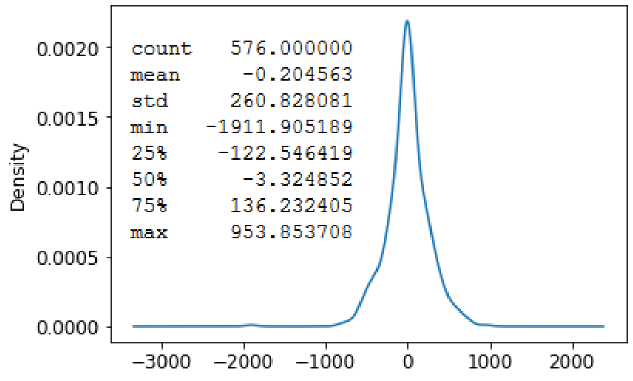


Figure 4.7: Decomposition of the Bocconi area time series.

4.6.1 Gaussian Estimator Error Model Validation

In this subsection we forecast the traffic in locations of Milano by using a Seasonal ARIMA model, then we proceed to show that the residuals (the difference between the observed and the forecasted traffic) approximately follow a Gaussian distribution and that they are uncorrelated in time. Similar to Sect.4.5, we train for the month of November (20 days) and we forecast the traffic for the first week of December.

A Seasonal ARIMA model is described by the parameters $(p, d, q) \times (P, D, Q)S$, with p non-seasonal AR order, d non-seasonal differencing, q non-seasonal MA order, P seasonal AR order, D seasonal differencing, Q seasonal MA order, and S time span of repeating seasonal pattern. By inspecting the signal decomposition for the time series for the area in Milano of Bocconi University, i.e. Fig.4.7, one can distinguish the dominant components of the mobile traffic evolution in time of a location in Milano city. Primarily, the strong seasonal component of the 24 hour period (S) and secondly the



```
count     576.000000
mean       -0.204563
std       260.828081
min     -1911.905189
25%      -122.546419
50%        -3.324852
75%       136.232405
max       953.853708
```

Figure 4.8: Distribution of the residuals for SARIMA(2,0,0)x(2,1,0)24

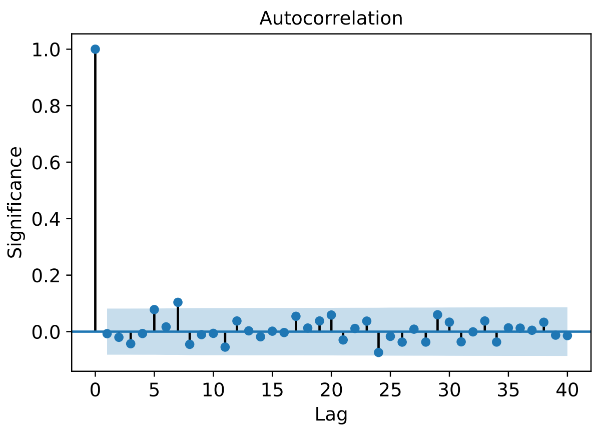


Figure 4.9: ACF plot of residuals.

trend, where traffic fluctuates during the duration of the week (decreasing significantly every Friday). We successfully capture these patterns by using the parameters $(2, 0, 0) \times (2, 1, 0)24$.

To validate the effectiveness of the selected model, we present Fig.4.8 and Fig.4.9. In Fig.4.8, we see the statistics and the distribution of the residuals, which approximates a gaussian. Finally, to create Fig.4.9, we inspect the time series of the residuals and make the AutoCorrelation Function (ACF) plot. Clearly there is no strong correlation between future samples of the residuals, hence the model captures the patterns in time evolution of the traffic. The predictor error is closely approximated by zero-mean gaussian noise with its variance given by the SARIMA model fit.

Table 4.7: Comparison between traffic prediction methods.

Prediction	MSE (10^6)	Avg Cost	11h-13h	Violations
Sample Mean	1,290	6,60	11,42	<0,1%
SARIMA	0,280	6,22	10,43	<0,1%
LSTM NN	0,266	6,18	10,28	<0,1%
Oracle	-	*5,90*	*9,86*	*0%*

4.6.2 Numerical Comparison

Our numerical results are summarized in Table 4.7. We use the same setup as in Fig.4.6, Tables 4.5 and 4.6; the parameters can be found on the caption of Fig.4.6. The different robust maps are calculated using as input the traffic $\boldsymbol{\lambda}$ and variance $\boldsymbol{\sigma}$ vectors, estimated by the method named on the first column. In the second column of Table 4.7 we give the achieved Mean Square Error (MSE) of the forecasts for each prediction method with respect to the observed (actual) value. Finally, we compare the average play out cost of the configuration prescribed by each of the robust maps and we compare to the optimal cost achieved by optimizing based on the oracle predictions. We note that refining the prediction method increases the network performance reducing the gap to the optimal up to 5%, in comparison to greater than 12% obtained by the simplest predictor. In accordance to our model, all of the prediction methods achieve the robustness guarantee set by the ϵ parameter at 0.1%. Selecting the appropriate prediction method is a design choice, as performance can be sacrificed for decreased computational complexity, or vice versa.

4.7 Conclusion

The problem of user association in dense networks becomes challenging due to frequent unexpected traffic fluctuations. We showed that past traffic data can be exploited towards precalculating association maps, which are designed to be robust and can be tuned to protect the base stations from overload. Accordingly, we proposed a theoretical framework for efficiently computing the optimal robust map, parametrized to a large class of utility functions that allow the system designer to tune the base station load. Finally, we evaluated our approach in Milano dataset, and found that our methodology is very effective at protecting UDNs from unexpected spikes, allowing the offering of premium wireless service.

The robust framework serves as an introduction to our resource reservation body of work later, as it could be used to economically reserve resources, while guaranteeing QoS. For example a routing map, computed by GRMA, could associate flows to a cloud, optimizing the expectation and variance of the cost of traffic load on the server. The map would give the routing decisions and indirectly the amount of resources (computation, memory, etc) required to be reserved. This framework though, depends on the predictability of future demand, which as described in the following chapters (5 and 6) is not always true for cloud workloads. For such applications, we will need a new framework that will adapt the reservations based on the changing distributions of the demand.

Chapter 5

Online Cloud Resource Reservation with Budget Constraints

5.1 Introduction

A fundamental challenge in cloud computing is to reserve just enough resources (e.g. memory, CPU, and bandwidth) to meet application runtime requirements [25]. We desire reservations that accurately meet the requirements: resource over provisioning causes excessive operation costs, while under provisioning may severely degrade service quality, causing interruptions and real-time deployment of extra resources, which costs heavily [26]. The problem resembles the well-known *newsvendor model* [27], where we seek an inventory level that maximizes the vendor revenue versus a forecast demand. Similarly, the robust framework presented in chapter 4 could be applied to capture constraints and guarantee QoS. In cloud computing, however, the common assumption of demand predictability does not hold. Recent experimentation in a Google cluster [1] shows that the profile of cloud resources exhibits highly non-stationary behaviour, and prediction is very difficult, if not impossible. Furthermore, in the increasingly relevant scenario of edge computing, the workload is expected to vary quickly with geography, mobility, and user application trends, and therefore its fluctuations will be even more unpredictable. All these motivate the approach in this chapter; *to design a model-free online reservation policy for cloud computing using ideas from machine learning.*

A concern with machine learning approaches is that their exploration phase combined with occasional unpredictability, may lead to an unforeseen violation of an important constraint. In our case, reserving fewer resources than needed for long time periods may potentially mount a serious threat on the operation of the cloud system. Therefore, apart from the demand

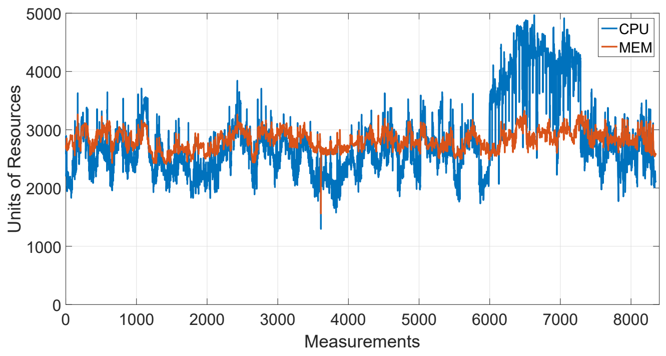


Figure 5.1: Aggregate resource utilization of the Google Cluster. The resources are normalized with respect to the server with the highest memory and CPU. Every point corresponds to 5mins, up to 29 days measured. The fluctuations are characterized as unpredictable in [1].

unpredictability, we are faced with the extra challenge of guaranteeing that the average resource violations will not exceed a pre-defined threshold. To address both aforementioned challenges at once, we cast the problem of resource reservation in the setting of constrained-*Online Convex Optimization* (OCO), seeking to find a no regret policy with violation guarantee. This is an extension of the standard machine learning framework OCO, where the online policy competes versus adversarial resource demands. The performance metric is the regret, i.e. the cost difference between our policy and the best static reservation with knowledge of the entire demand sample path. We seek to find an online policy that achieves zero average regret under its worst adversary (a condition known as "no regret"), while we require from our policy to satisfy a time-average constraint that concerns the number of violations occurring in the studied time period. To the best of our knowledge, no previous work has addressed the problem of reserving cloud resources in this setting. The main contribution of this chapter is to design the Time Horizon Online Reservation (THOR): a feasible "no regret" resource reservation policy.

5.1.1 Prior Work

The framework of OCO is used to minimize the sum of convex functions $\sum_t f_t(x_t)$ where f_t is revealed to the optimizer after the action x_t is taken. It was inspired by the seminal paper of Zinkevich [39], who proposed to predict f_t as a linearization of f_{t-1} and take a gradient step in the direction of $\nabla f_{t-1}(x_{t-1})$. OCO allows us to design model-free algorithms that are data-driven and robust to environment changes cf. [79, 80], since the "no regret" property is extremely powerful; it implies that our online algorithm learns to allow the same average losses as a static policy that knows the future. We study a slightly perturbed setting of OCO, where the function f is static and known, but the constraint function $\sum_t g_t(x_t) \leq 0$ is chosen by the adversary.

A number of past works have focused on constrained-OCO problems. The simplest case is when the constraint is not adversarial, and limits the policy actions in the same manner at each time slot. This is addressed in the online gradient of Zinkevich via a projection, but to extend to cases with complicated sets [81,82] proposed an alternative approach based on Lagrangian relaxation. If we have a time-average constraint coupling the decisions over time, [83] uses self-concordant barrier functions to relax it, while [81,82] provides a dual algorithm. While these methods ensure asymptotic feasibility–the constraint residual $\sum_t g(x_t)$ scales as $o(T)$–they do not apply to our problem, where the constraint set is shaped by adversary-selected (time changing) convex functions.

The well-known result of [37] states that in general it is impossible to simultaneously achieve $o(T)$ regret and asymptotic feasibility in constrained-OCOs where both objective and constraint set are tinkered by an adversary. However, [40] showed that it is possible when there exists a static solution which strictly satisfies all constraint functions at every slot (a Slater vector). With the approach of [40], however, the "no regret" property is provided with respect to the Slater vector, meaning that applying [40] to our problem will yield a feasible resevation policy, but with a poor cost guarantee due to the restrictive assumption of ensuring the constraint at every slot. Consider a benchmark static reservation that only satisfies the average constraint every K slots. Then as K increases, the constraint is looser and we obtain a stronger guarantee, but establishing the guarantee may become harder. While [40] proves the case $K = 1$, in this chapter we prove the case where $K = O(T^{1-\varepsilon})$, and propose the online policy THOR, which is asymptotically feasible and provides $o(T)$ regret.

5.1.2 Our Contribution

We formulate the problem of reserving resources for cloud computing as a constrained-OCO. At each slot, *(i)* an online reservation policy decides a reservation vector, then *(ii)* the adversary decides a demand vector, and last *(iii)* a cost is paid for the reserved resources and a violation is noted if the demand was not covered by the reservation. A reservation policy is feasible if at the end of the T-slot horizon the number of violations of resource i are no more than a configurable $\epsilon_i T$. *We seek to find a feasible policy that achieves no regret with respect to the best static reservation in hindsight while satisfying the violation constraint at all K-slot windows within T.* Our contributions are summarized as follows:

- We introduce a natural machine learning approach for reserving resources for cloud computing. Our constrained-OCO framework is an ideal setup for investigating more complicated scenarios with reservations, e.g., reserving resource slices in wireless networks.

- We propose THOR, a policy we prove to achieve asymptotic feasibility and "no regret" with respect to a benchmark constrained to $K = O(T^{1-\varepsilon})$ slots, the first of its kind. The performance guarantees of THOR are obtained by a novel combination of the Lyapunov K-slot drift technique with the linearization idea of Zinkevich. THOR inherits the simplicity of online gradient, and therefore is straightforward to implement in practical systems.
- We have validated THOR resource reservations using a public dataset provided by Google [2]. THOR vastly outperforms our implementation of the textbook Follow The Leader (FTL) policy in guaranteeing the violations constraint, while it achieves similar or sometimes better performance than the static oracle T-slot policy, in the challenging, non-stationary CPU workload.

5.2 System Model

Requests. Our system operates in slots $t = 1, \dots, T$, with T being the horizon. In slot t the cloud users request λ_i^t units of resource i (for example $i = 1$ refers to CPU and $i = 2$ to memory). We consider I types of resources. To model the fact that the vectors $\boldsymbol{\lambda}^t$ are drawn from a general distribution $D(\boldsymbol{\lambda}^1, \dots, \boldsymbol{\lambda}^T)$ (hence model-free), we allow them to be selected by an adversary who aims to harm our system.

Reservations. A reservation policy π decides at each slot to reserve $x_i^{t,\pi}$ units of resource i. Formally, at time t an *online reservation policy* is a mapping from past requests to a vector of nonnegative values:

$$\pi : (\boldsymbol{\lambda}^1, \dots, \boldsymbol{\lambda}^{t-1}, \boldsymbol{x}^1, \dots, \boldsymbol{x}^{t-1}) \to \mathbb{R}_+^I.$$

The above is depictive of the action order within a slot, i.e., first a reservation $\boldsymbol{x}^{t,\pi}$ is made, and then the adversary reveals the values $\boldsymbol{\lambda}^t$. There is a cost c_i attached to each resource, and therefore at the end of slot t, policy π incurs a cost

$$C(\boldsymbol{x}^{t,\pi}) = \sum_{i=1}^{I} c_i x_i^{t,\pi}.$$

Violation guarantees. Let v_i^t denote the event of resource i violation in slot t, which occurs when the request for a resource exceeds the reservation, i.e., $v_i^t \triangleq \mathbb{1}\left\{\lambda_i^t > x_i^{t,\pi}\right\}$. A policy π is called *feasible* if the time-average violations of resource i do not exceed a pre-determined threshold ϵ_i:

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\left[v_i^t\right] \le \epsilon_i, \quad \text{for all } \; i = 1, \dots, I.$$

Hence, the feasibility constraint of resource i can also be written in the form:

$$\sum_{t=1}^{T} \mathbb{P}\left(\lambda_i^t > x_i^t\right) \leq \epsilon_i T. \tag{5.1}$$

Observe that the above constraint couples the decisions across the entire horizon. Our initial objective is to find a *feasible* policy that minimizes the total cost $\sum_{t=1}^{T} C(\boldsymbol{x}^{t,\pi})$, however, since in slot t the arrivals $\boldsymbol{\lambda}^t$ are unknown, such an objective is out of reach. Next, we provide an alternative approach through the framework of *Online Convex Optimization* (OCO).

Regret. We introduce the performance metric of *regret*, which is commonly used in the literature of machine learning to measure the robustness of online algorithms [79, 80]. The regret R_T^π is the cumulative difference of losses between policy π and a benchmark policy which is aware of the entire sample path $\boldsymbol{\lambda}^1, \dots, \boldsymbol{\lambda}^T$ but forced to take a static action throughout the horizon–often called *best static policy in hindsight.* Specifically, let $\boldsymbol{x}^*$ denote our benchmark, which is calculated as the solution to the following problem:

$$\boldsymbol{x}^* \in \arg\min_{\boldsymbol{x} \in \mathbb{R}_+^I} T\, C(\boldsymbol{x}) \;\; \text{s.t.} \;\; \sum_{t=1}^{T} \mathbb{P}\left(\lambda_i^t > x_i^*\right) \leq \epsilon_i T.$$

Then the regret is defined as follows:

$$R_T^\pi = \inf_{D} \; \mathbb{E}\left[\sum_{t=1}^{T} C(\boldsymbol{x}^{t,\pi}) - T\, C(\boldsymbol{x}^*)\right],$$

where the infimum is taken w.r.t. the supported distributions of the adversary, and the expectation w.r.t. the (possibly) randomized $\boldsymbol{x}^{t,\pi}, \boldsymbol{\lambda}^t$. If $R_T^\pi = o(T)$, then we say that policy π has "no regret", since $R_T^\pi/T \to 0$ as $T \to \infty$, i.e., the average losses from the benchmark are amortized. Our goal is to obtain a *feasible* online reservation policy with "no regret".

5.2.1 Modified Adversary

In this subsection we introduce two innovations with respect to the classical OCO framework, one related to the decisions of the adversary, and one to the benchmark we compare against.

Probabilistic adversary support. It is customary to limit the actions of the adversary to be no more than a finite value $\Lambda_{i,\max}$ for resource i. In our problem, this hard constraint is problematic: *(i)* small $\Lambda_{i,\max}$ (e.g. set equal to the maximum observed value) will cause our algorithms to "think" that an action $x_i^t = \Lambda_{i,\max}$ ensures a violation-free slot, which will incur instability should a flow of larger values occurs in the future, while *(ii)* large $\Lambda_{i,\max}$ will force our algorithms to consistently overbook in order to ensure no violation, leading to very poor performance.

To address this issue, we propose the idea of bounding the adversary with a stationary process with known distribution; the distribution is configured based on the data. Specifically, in this chapter we set Λ_i^t to be an i.i.d. Gaussian process, and then restrict the adversary to $\lambda_i^t \in [0, \Lambda_i^t]$. The benefit of this approach lies in the elasticity offered by the stationary process. Due to the concavity of its cumulative distribution, our algorithms will be able to learn tradeoffs between the probability of violations and the investment cost. We mention that despite $\boldsymbol{\Lambda}^t$ being stationary, the actual arrivals $\boldsymbol{\lambda}^t$ remain model-free and possibly non-stationary.

K-slot feasibility. When showing that policy π has no regret, we are effectively showing that π achieves the same average performance with the benchmark. It is useful then to introduce a class of benchmark policies that ensure the violation constraint for all windows of K slots within the horizon T:

$$\boldsymbol{x}^*(K) \in \arg\min_{\boldsymbol{x}\in\mathbb{R}_+^I} T\,C(\boldsymbol{x}) \tag{5.2}$$
$$\text{s.t.} \sum_{k=0}^{K-1} \mathbb{P}\left(\lambda_i^{t+k} > x_i^*\right) \le \epsilon_i K, \quad \forall t = 1, \ldots, T-K.$$

Observe that for $K = T$ we obtain the original benchmark. However, as K decreases, we will have an interesting trade off: on one hand, the benchmark should ensure the average violations in shorter periods, hence it will incur higher investment costs (as the optimization above will have a stricly smaller constraint set), on the other hand it might be easier to prove the “no regret” property. Indeed, prior work [40] produced a “no regret” policy versus a benchmark which satisfies the violation constraint at every slot ($K = 1$). Note, however, that such a guarantee is compromised in our problem. For example, in Fig. 5.2 we plot the performance of K benchmarks, where we observe greatly increased cost for $K = 1$. We also observe that, initially increasing K enhances greatly the achieved performance, but the returns are diminishing. In this chapter, we will obtain a “no regret” guarantee for the case when $K = O(T^{1-\varepsilon})$.

5.3 Queue Assisted Online Learning Algorithm

In this section we present our algorithm, and provide intuition into its functionality. An online reservation policy is called in slot t to update the decisions from $\boldsymbol{x}^{t-1}$ to $\boldsymbol{x}^t$. In order to explain how THOR performs this update, we will discuss some intermediate steps, namely *(i)* the constraint convexification *(ii)* the predictor queue, and *(iii)* the drift plus penalty plus smoothness. Finally, we will present THOR and show how it naturally arises from these three steps. Formal performance guarantees are presented in the following section.

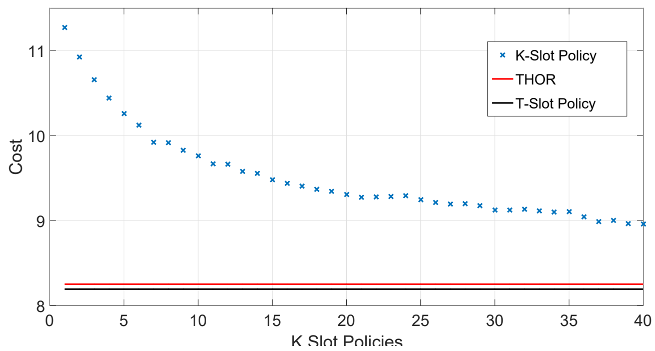


Figure 5.2: Average cost comparison in a repeated randomly generated instance. Adversary uniformly selects $\lambda_i^t \in [0, \Lambda_i^t]$ in the course of the experiment, while Λ_i^t is maintained between the multiple runs of the experiment. The plotted points represent the cost of $K(=[1,40])$-slot benchmark policy, the black line is the T-slot benchmark policy and the red line is our policy.

5.3.1 Constraint Convexification

In this subsection, we propose a convex approximation for the feasibility constraint Eq.(5.1). This constraint will be tighter as it will become obvious below. We presented in the system model $\mathbb{P}\left(\lambda_i^t > x_i^t\right)$ to be the quantile function of the adversary for every slot t and resource i. This function is a non-increasing function, but can be non-convex. For the approximation we will use $\mathbf{\Lambda}$ which we assume to be an i.i.d. Gaussian process that caps the distribution of the adversary $\lambda_i^t \leq \Lambda_i^t$. For ease of exposition, we define $F_{\lambda_i^t}(x_i^t) \triangleq \mathbb{P}\left(\lambda_i^t > x_i^t\right)$. By our assumption, it is true that:

$$F_{\lambda_i^t}(x_i^t) \leq F_{\Lambda_i^t}(x_i^t).$$

Furthermore, since the quantile function of the Gaussian distribution is quasiconvex we can design a convex envelope function for F_Λ:

$$F_{E_i^t}(x_i^t) = \begin{cases} G_i^t\, x_i^t + \beta_i, & x^t < \mu_i \\ F_{\Lambda_i^t}(x_i^t), & \text{otherwise}, \end{cases}$$

where $G_i^t = F'_{\Lambda_i^t}(\mu_i)$ and $F_{\Lambda_i^t}(x_i^t) \leq F_{E_i^t}(x_i^t)$. Hence, for reservations less than μ_i the CCDF is enveloped by a linear function that decreases by G_i^t. We note that this is defined for the theoretical proofs to follow through, but also it is relevant to practice, in the unlikely case of loose guarantees (big ϵ_i) or extremely optimistic error predictions. This envelope will produce strong derivatives to increase the reservations (while the gaussian CCDF will have a smaller -but still negative- slope). Hereinafter, we simplify the notation to $F_t(x_i^t)$, to describe $F_{E_i^t}(x_i^t)$. We further note that, $0 \leq F_{E_i^t}(x_i^t) \leq F$, where F is a constant and that there exists a constant $G \geq G_i^t$.

5.3.2 Predictor Queue

Intuitively, we could use $F_t(x_i^t)$ to take a good step in slot t towards ensuring the time-average constraint, but this information is not available in our model: the function F_t relates to the choice of the adversary that takes place after we commit our decision $\boldsymbol{x}_t$. What is available is the previous value $F_{t-1}(x_i^{t-1})$. Inspired by the work of Zinkevich [39], we introduce a linear *prediction of the violation probability* $F_t(x_i^t)$:

$$b_i(x_i^t) \triangleq F_{t-1}(x_i^{t-1}) + F'_{t-1}(x_i^{t-1})(x_i^t - x_i^{t-1}), \tag{5.3}$$

where $F'_{t-1}(x_i^{t-1})$ denotes the derivative, hence $b_i(x_i^t)$ is the first order Taylor expansion of F_{t-1} around x_i^{t-1} evaluated at x_i^t. Note that only x_i^t in Eq.(5.3) is to be determined at time t.

Definition 1 (Predictor Queue Vector)**.** *We define as* $\boldsymbol{Q}$ *the Predictor Queue Vector. Every element of the vector is a virtual queue for each resource, containing the sum of the predicted probabilities of violation in the past iterations.*

$$Q_i(t+1) = [Q_i(t) - \epsilon_i + b_i(x_i^t)]^+. \tag{5.4}$$

Virtual queue $Q_i(t)$ is a counter that increases with our controllable predictions $b_i(x_i^t)$ and decreases at a steady rate ϵ_i. If we limit the growth of $Q_i(t)$ to $o(T)$, then the average (predicted) violations would only overshoot ϵ_i by an amortizable amount, which can be manipulated into providing asymptotic feasibility. In fact, we will rigorously prove this intuition in section 5.4.2.

5.3.3 Drift Plus Penalty Plus Smoothness for Online Learning

Having obtained a handle on the time-average constraint (and hence also asymptotic feasibility) via the predictor queue, it remains to explain how $\boldsymbol{x}_t$ is updated in THOR. To combine the consideration of the cost and the predictor queue we will use the technique of Drift Plus Penalty (DPP), a framework used to solve constrained stochastic network optimization problems [38]. In DPP, the tradeoff between the constraints (queue lengths) and the cost is controlled via the penalty parameter V. Specifically, first we consider the quadratic Lyapunov drift (defined as $\Delta(t) = \frac{1}{2}\sum_i Q_i(t+1)^2 - \frac{1}{2}\sum_i Q_i(t)^2$), which measures the impact of our policy on the norm of the predictor queue vector. The drift can be bounded as in Lemma 4.2 in [84],

$$\Delta(t) \le B + \sum_{i=1}^{I} Q_i(t)[b_i(x_i^t) - \epsilon_i],$$

where $B = \frac{1}{2}I(\max\{GD, F\})^2$ is a constant. Then, adding to both parts of the drift inequality the *penalty* term, defined as the cost function multiplied by a weight V, we arrive at the drift plus penalty inequality:

$$\Delta(t) + VC(\boldsymbol{x}) \leq B + \sum_{i=1}^{I} Q_i(t)[b_i(x_i^t) - \epsilon_i] + VC(\boldsymbol{x}). \tag{5.5}$$

A large $\boldsymbol{x}$ tends to minimize the drift term $\Delta(t)$ but incurs a high cost $VC(\boldsymbol{x})$, while small $\boldsymbol{x}$ has the opposite effect. Clearly, minimizing DPP achieves a balance between the two conflicting objectives. Remarkably, prior work in DPP shows that finding the minimizer of the upperbound in Eq.(5.5) at every slot, eventually produces an online policy that simultaneously achieves a cost within $O(\frac{1}{V})$ of the optimal and bounds the queue with $O(V)$.

Here, we will further add a quadratic (Tikhonov) regularizer [85] centered at the previous iterate $\boldsymbol{x}^{t-1}$, this will encourage the new reservation $\boldsymbol{x}^t$ to not drastically change its value compared to the last iteration $\boldsymbol{x}^{t-1}$.

Definition 2 (Drift Plus Penalty Plus Smoothness)**.** *By adding the penalty term $\alpha||\boldsymbol{x}^t - \boldsymbol{x}^{t-1}||^2$ on both sides of the inequality Eq.*(5.5) *we get the upper bound on Drift Plus Penalty Plus Smoothness (DPPPS).*

$$\begin{aligned} &\Delta(t) + VC(\boldsymbol{x}^t) + \alpha||\boldsymbol{x}^t - \boldsymbol{x}^{t-1}||^2 \leq \\ &\quad B + \sum_{i=1}^{I} Q_i(t)[b_i(x_i^t) - \epsilon_i] + VC(\boldsymbol{x}^t) + \alpha||\boldsymbol{x}^t - \boldsymbol{x}^{t-1}||^2. \end{aligned} \tag{5.6}$$

also we define the upper bound as a function of $\boldsymbol{x}$:

$$g(\boldsymbol{x}) \triangleq B + \sum_{i=1}^{I} Q_i(t)[b_i(x_i^t) - \epsilon_i] + VC(\boldsymbol{x}) + \alpha||\boldsymbol{x} - \boldsymbol{x}^{t-1}||^2.$$

to be used in the theoretical analysis later.

Asymptotically the effect of the regularizer fades, so the optimality results are unaffected. While, however, the original DPP yields online policies that constantly operate at two extremes (called bang-bang policies), with this regularizer THOR update is transformed to a smooth gradient step, as we show next.

5.3.4 Online Reservation Policy

Proposition 1 (THOR minimizes DPPPS bound)**.** *The updates*

$$x_i^t = \left[x_i^{t-1} - \frac{1}{2\alpha}(Vc_i + Q_i(t)F'_{t-1}(x_i^{t-1}))\right]^+. \tag{5.7}$$

minimize at each slot t the upper bound on the predicted DPPPS Eq.(5.6).

Proof. The function $g(\boldsymbol{x})$, found in Def. 2, is decomposable to the I resources and the minimizer of each component is:

$$\begin{aligned} x_i^t &= \operatorname*{argmin}_{x_i \geq 0}\{g(\boldsymbol{x})\} \\ &= \operatorname*{argmin}_{x_i \geq 0}\{Q_i(t)F'_{t-1}(x_i^{t-1})x_i + Vc_i x_i + \alpha(x_i - x_i^{t-1})^2\}. \end{aligned}$$

From this expression, it becomes apparent that, by removing the regularizer ($\alpha = 0$), since F is a quantile function and F' is negative, we get the following:

$$x_i^t = \begin{cases} 0, \text{ if } Vc_i \geq |Q_i(t)F'_{t-1}(x_i^{t-1})| \\ +\infty, \text{ otherwise.} \end{cases}$$

This generates a bang-bang reservation policy for every time slot t. Bang-bang policies are ideal for scheduling or routing, but are not practical for a cloud resource reservation environment. Here, we must maintain as stable reservations as possible, allowing slow installation or removal of servers and resources. Meanwhile, with the added regularizer, the reservation update is a gradient minimization step with step size $\frac{1}{2\alpha}$. This comes naturally by finding the stationary point:

$$\begin{aligned} Q_i(t)F'_{t-1}(x_i^{t-1}) + Vc_i + 2\alpha x_i - 2\alpha x_i^{t-1} &= 0 \\ x_i &= \left[x_i^{t-1} - \frac{1}{2\alpha}(Vc_i + Q_i(t)F'_{t-1}(x_i^{t-1}))\right]^+. \end{aligned}$$

□

In conclusion the evolution of THOR policy can be described as follows:

Time Horizon Online Reservations (THOR)

Initializition: Predictor queue initial length $\boldsymbol{Q}(1) = \boldsymbol{0}$, initial reservation vector $\boldsymbol{x}^0 \in \mathbb{R}_+^I$.
Parameters: penalty constant V, step size α, cost of resource unit c_i, constraint requirement per resource $\epsilon_i \leq 0.5$.
Updates at every time slot $t \in \{1, \ldots, T\}$:

$$x_i^t = \left[x_i^{t-1} - \tfrac{1}{2\alpha}(Vc_i + Q_i(t)F'_{t-1}(x_i^{t-1})\right]^+, \tag{5.8}$$

$$Q_i(t+1) = [Q_i(t) + b_i(x_i^t) - \epsilon_i]^+. \tag{5.9}$$

Where $b_i(x_i^t)$ is given in Eq.(5.3) and $F'_t(x_i^t)$ is the derivative of the convexified CCDF $F_{E_i^t}$.

In the following section we will use the above-explained intuition to establish rigorous proofs that our THOR algorithm is asymptotically feasible and has "no regret" against the K benchmark. These results are harder to achieve than those from the standard DPP framework, because we have no knowledge of the current status of the queue ($F_t(x_i^t)$) and also the average violations (arrivals to the virtual queue) are arbitrarily distributed (selected by a constrained adversary), hence there are no Markovian statistics to be learned.

5.4 Performance Analysis

In this section we prove that THOR is a feasible policy with no regret against K-slot policies. First we will show a general upper bound on THOR DPPPS by using the strong convexity property of $g(\boldsymbol{x})$. Then we will use this general bound to achieve a sublinear in time bound on the predictor queues of THOR, which will be used to prove feasibility, i.e. no time average constraint violation in a time horizon T. Finally, we compare THOR with a K benchmark, using the K benchmark properties, to prove the THOR's no regret. The results are summarized in a corollary at the end of the section.

We note that since $\boldsymbol{x}^t$ by Eq.(5.7) is the minimizer of the upper bound on DPPPS (Eq.(5.6)) for every time slot t; any static reservation $\boldsymbol{y} \in \mathbb{R}_+^I$ bounds THOR's DPPPS by above. This is an important aspect for the analysis to follow.

$$B + \sum_{i=1}^{I} Q_i(t)[b_i(x_i^t) - \epsilon_i] + VC(\boldsymbol{x}^t) + \alpha||\boldsymbol{x}^t - \boldsymbol{x}^{t-1}||^2 \leq$$
$$B + \sum_{i=1}^{I} Q_i(t)[b_i(y_i) - \epsilon_i] + VC(\boldsymbol{y}) + \alpha||\boldsymbol{y} - \boldsymbol{x}^{t-1}||^2.$$

A stronger condition is required in our analysis, a more refined upper bound which is achieved due to the imposed strong convexity of the Tikhonov regularizer. The following lemma is a key lemma for our results.

Lemma 1. *[Strong Convexity Bound] Let $\boldsymbol{y} \in \mathbb{R}_+^I$ be a static reservation, then the DPPPS of THOR $\boldsymbol{x}^t$ is bounded by:*

$$\Delta(t) + VC(\boldsymbol{x}^t) + \alpha||\boldsymbol{x}^t - \boldsymbol{x}^{t-1}||^2 \leq B + VC(\boldsymbol{y}) +$$
$$\sum_{i=1}^{I} Q_i(t)[F_t(y_i) - \epsilon_i] + \alpha||\boldsymbol{y} - \boldsymbol{x}^{t-1}||^2 - \alpha||\boldsymbol{y} - \boldsymbol{x}^t||^2.$$

Proof. Due to $g(\boldsymbol{x})$ being 2α-strongly convex, we have $g(\boldsymbol{x}_{min}) = g(\boldsymbol{y}) - \frac{2\alpha}{2}||\boldsymbol{y} - \boldsymbol{x}_{min}||^2$, for all $\boldsymbol{y} \in \mathbb{R}_+^I$ static policies. This refines THOR's upper bound:

$$\begin{aligned}
&\Delta(t) + VC(\boldsymbol{x}^t) + \alpha||\boldsymbol{x}^t - \boldsymbol{x}^{t-1}||^2 \leq \\
&B + \sum_{i=1}^{I} Q_i(t)[b_i(x_i^t) - \epsilon_i] + VC(\boldsymbol{x}^t) + \alpha||\boldsymbol{x} - \boldsymbol{x}^{t-1}||^2 \leq \\
&B + \sum_{i=1}^{I} Q_i(t)[b_i(y_i) - \epsilon_i] + VC(\boldsymbol{y}) + \\
&\qquad\qquad \alpha||\boldsymbol{y} - \boldsymbol{x}^{t-1}||^2 - \alpha||\boldsymbol{y} - \boldsymbol{x}^t||^2 \overset{(1)}{\leq} \\
&B + \sum_{i=1}^{I} Q_i(t)[F_t(y_i) - \epsilon_i] + VC(\boldsymbol{y}) + \\
&\qquad\qquad \alpha||\boldsymbol{y} - \boldsymbol{x}^{t-1}||^2 - \alpha||\boldsymbol{y} - \boldsymbol{x}^t||^2.
\end{aligned}$$

For (1) we use the convexity of the envelope CCDF function $F_{E_i^t}$, $F_t(y_i) \geq b_i(y_i)$. □

5.4.1 Predictor Queue Upper Bound

The first important step is to prove that under our policy Eq.(5.8)-(5.9) the predictor virtual queues increase sublinearly with time. This will later give us a bound for the constraint violation which ensures THOR policy's feasibility.

Lemma 2 (Upper Bound on Predictor Queues)**.** *Let D be a finite upper bound on the maximum reservation, such that $x_i \leq x_{max} = D$. The queue predictor vector $\boldsymbol{Q}$ at time slot T satisfies the following inequalities:*

$$||Q(T+1)||_2 \leq \sqrt{2BT + 2VT\sum_{i=1}^{I} c_i r_i(\epsilon_i) + 2\alpha I D^2},$$

$$||Q(T+1)||_1 \leq \sqrt{I}||Q(T+1)||_2,$$

where $r_i(\epsilon_i) \triangleq \mu_i + \sigma_i \mathcal{Q}^{-1}(\epsilon_i)$.

Proof. We will use Lemma 1, to prove that a static reservation $\bar{\boldsymbol{y}}$ achieves sublinear growth in T of the $\boldsymbol{Q}$ vector. Then, since our policy x_i^t is the minimizer of the DPPPS, our policy achieves (at least) the same upper bound. Select $\bar{\boldsymbol{y}}$ to be a static reservation that always satisfies the constraint $F_{\Lambda_i^t}(\bar{y}_i) \leq \epsilon_i$, $\forall i, t$. It is easy to show that $\bar{y}_i \geq \mu_i + \sigma_i \mathcal{Q}^{-1}(\epsilon_i)$, where $\mathcal{Q}$ is the quantile function of the standard normal distribution. We take $\bar{y}_i = \mu_i + \sigma_i \mathcal{Q}^{-1}(\epsilon_i)$, which gives the cost:

$$C(\bar{\boldsymbol{y}}) = \sum_{i=1}^{I} c_i(\mu_i + \sigma_i \mathcal{Q}^{-1}(\epsilon_i)) = \sum_{i=1}^{I} c_i r_i(\epsilon_i).$$

According to Lem.1:

$$\Delta(t) + \underbrace{VC(\boldsymbol{x}^t) + \alpha||\boldsymbol{x}^t - \boldsymbol{x}^{t-1}||^2}_{(a)} \leq B + VC(\bar{\boldsymbol{y}}) +$$

$$\underbrace{\sum_{i=1}^{I} Q_i(t)[F_{E_i^t}(\bar{y}_i) - \epsilon_i]}_{(b)} + \alpha||\bar{\boldsymbol{y}} - \boldsymbol{x}^{t-1}||^2 - \alpha||\bar{\boldsymbol{y}} - \boldsymbol{x}^t||^2.$$

Here, terms (a) and (b) can be discarded. (a) is positive, while (b) is negative, due to $F_{E_i^t}(\bar{y}_i) \leq \epsilon_i, \forall i, t$. This leaves us with:

$$\Delta(t) \leq B + VC(\bar{\boldsymbol{y}}) + \alpha||\bar{\boldsymbol{y}} - \boldsymbol{x}^{t-1}||^2 - \alpha||\bar{\boldsymbol{y}} - \boldsymbol{x}^t||^2.$$

We take the telescopic sum over the time slots $\{1, ..., T\}$:

$$\sum_{t=1}^{T} \Delta(t) \leq BT + VC(\bar{\boldsymbol{y}})T +$$

$$\alpha \sum_{t=1}^{T} ||\bar{\boldsymbol{y}} - \boldsymbol{x}^{t-1}||^2 - \alpha \sum_{t=1}^{T} ||\bar{\boldsymbol{y}} - \boldsymbol{x}^t||^2.$$

By taking the telescopic sum the intermediate terms of (i) the quadratic Lyapunov drift $\Delta(t)$ and (ii) the norms cancel each other. Furthermore, $Q(1) = 0$ and the (negative) norm term $-\alpha||\bar{\boldsymbol{y}} - \boldsymbol{x}^T||^2$ can be dropped. We replace $VC(\bar{\boldsymbol{y}})$ with its cost and we arrive at:

$$\frac{1}{2} \sum_{i=1}^{I} Q_i(T+1)^2 \leq VT \sum_{i=1}^{I} c_i r_i(\epsilon_i) + BT + \alpha||\bar{\boldsymbol{y}} - \boldsymbol{x}^0||^2.$$

Since $x_i \leq x_{max} = D$, then $||\boldsymbol{x} - \boldsymbol{y}|| \leq \sqrt{I}D, \ \forall \boldsymbol{y}, \boldsymbol{x} \in \mathbb{R}_+^I$.

$$\sum_{i=1}^{I} Q_i(T+1)^2 \leq 2BT + 2VT \sum_{i=1}^{I} c_i r_i(\epsilon_i) + 2\alpha I D^2$$

$$||Q(T+1)||_2 \leq \sqrt{2BT + 2VT \sum_{i=1}^{I} c_i r_i(\epsilon_i) + 2\alpha I D^2}.$$

The result for the bound on the 1-norm follows by using the norm inequality $||x||_1 \leq \sqrt{I}||x||_2$, for I vector elements. □

5.4.2 Constraint Residual

Having established a bound on $\boldsymbol{Q}$, our next objective is to use it to bound the constraint violations at time slot T and obtain asymptotic feasibility.

For simplicity, hereon we denote

$$Q^B(T+1) \triangleq \sqrt{2BT + 2VT\sum_{i=1}^{I} c_i r_i(\epsilon_i) + 2\alpha I D^2}.$$

Theorem 1 (Upper Bound on Constraint Violation)**.** *The constraint violation for each resource is bounded by the size of the predictor queue at time $T+1$, $Q_i(T+1)$, which is upper bounded by $Q^B(T+1)$:*

$$\sum_{t=0}^{T-1} F_t(x_i^t) - \epsilon_i T \le Q^B(T+1) + \frac{G^2\sqrt{2B}T(T+1)}{4\alpha},$$

where $G \ge |G_i^t|$ is the upper bound of the absolute value of the derivative of the quantile function F defined in Sect.5.3.1.

Proof. We take the predictor queue update equation Eq.(5.4):

$$\begin{aligned} Q_i(t+1) = [Q_i(t) + b_i(x_i^t) - \epsilon_i]^+ &\ge Q_i(t) + b_i(x_i^t) - \epsilon_i \\ &\ge Q_i(t) + F_{t-1}(x_i^{t-1}) - G(x_i^t - x_i^{t-1}) - \epsilon_i, \end{aligned}$$

We have by projection properties and by the queue update policy Eq.(5.7) that $||x_i^t - x_i^{t-1}||_1 \le \frac{||Vc_i - Q_i(t)G||_1}{2\alpha} \le \frac{GQ_i(t)}{2\alpha}$:

$$F_{t-1}(x_i^{t-1}) - \epsilon_i \le Q_i(t+1) - Q_i(t) + \frac{G^2 Q_i(t)}{2\alpha}.$$

We sum for the time slots $t = \{1, ..., T\}$:

$$\sum_{t=1}^{T} F_{t-1}(x_i^{t-1}) - \epsilon_i T \le Q_i(T+1) + \frac{G^2\sum_{t=1}^{T} Q_i(t)}{2\alpha}.$$

We have $Q_i(t+1) \le Q_i(t) + ||b_i(x_i^t) - \epsilon_i||_1 \le Q_i(t) + \sqrt{2B}$, hence $Q(t) \le t\sqrt{2B}$, $\forall t$ and $Q_i(T+1) \le Q^B(T+1)$. Using those on the above equation we get:

$$\sum_{t=0}^{T-1} F_t(x_i^t) - \epsilon_i T \le Q^B(T+1) + \frac{G^2\sum_{t=1}^{T} t\sqrt{2B}}{2\alpha}.$$

The result follows. □

By properly selecting the constants V, α and for growing T the residual of the constraint on average goes to zero. In the next subsection, we will show that our policy guarantees similar cost against the K-Slot best static policy.

5.4.3 No Regret Against the K Benchmark Policy

Here, we will prove that our policy achieves no regret against K-slot benchmark policy, these are defined in Sect. 5.2 by Eq.(5.2). We start by the upper bound on DPPPS by Lem.(1), we prove a small technical lemma to be able to use the properties of the K slot benchmark; enabling us to upper bound THOR's performance in comparison to the benchmark.

Theorem 2 (Upper Bound Against K Benchmark)**.** *Regret, against the optimal static policy in $\mathbb{R}_+^I$ that achieves feasibility in K slots, is upper bounded by:*

$$\sum_{t=1}^{T} C(\boldsymbol{x}^t) - \sum_{t=1}^{T} C(\boldsymbol{x}^\star) \leq$$
$$\frac{BKT}{V} + \frac{\alpha I D^2}{V} + \frac{IB(K+1)(2K+1)}{6V} + IDc_i(K-1).$$

Proof. We begin by Lem.(1), by setting $\boldsymbol{y} = \boldsymbol{x}^*(K)$ and summing the inequality for K slots, $t = \{1, 2, \ldots, K\}$:

$$\sum_{\tau=0}^{K-1} \left[\Delta(t+\tau) + VC(\boldsymbol{x}^{t+\tau}) + \underbrace{\alpha ||\boldsymbol{x}^{t+\tau} - \boldsymbol{x}^{t+\tau-1}||^2}_{(a)} \right] \leq$$
$$BK + V\sum_{\tau=0}^{K-1} C(\boldsymbol{y}) + \underbrace{\sum_{\tau=0}^{K-1}\sum_{i=1}^{I} Q_i(t+\tau)[F_{t+\tau}(y_i) - \epsilon_i] +}_{(b)}$$
$$\alpha \sum_{\tau=0}^{K-1} ||\boldsymbol{y} - \boldsymbol{x}^{t+\tau-1}||^2 - \alpha \sum_{\tau=0}^{K-1} ||\boldsymbol{y} - \boldsymbol{x}^{t+\tau}||^2. \tag{5.10}$$

Lemma 3. *For policy $\boldsymbol{y} = \boldsymbol{x}^*(K)$, for any $t \in \{1, \ldots, T\}$:*

$$\sum_{\tau=0}^{K-1}\sum_{i=1}^{I} Q_i(t+\tau)[F_t(y_i) - \epsilon_i] \leq IBK(K-1).$$

Proof. We give a lower bound for $Q(t+K)$ for any $K \geq 1$:

$$Q_i(t+K) \geq Q_i(t) + \sum_{\tau=0}^{K-1} [F_{t+\tau}(y_i) - \epsilon_i], \tag{5.11}$$

and an upper bound:

$$Q_i(t+K) \leq Q_i(t) + \sum_{\tau=0}^{K-1} ||F_{t+\tau}(y_i) - \epsilon_i||_1. \tag{5.12}$$

Next, we use these bounds of the queue derived above, so that $Q_i(t)$ appears as a common term in the sum. We will prove for a single Queue $q_i(t)$. We denote $\max\{0, f(\cdot)\} \triangleq [f(\cdot)]^+$ and $\min\{0, f(\cdot)\} \triangleq [f(\cdot)]^-$:

$$
\begin{aligned}
&\sum_{\tau=0}^{K-1} Q_i(t+\tau)[F_{t+\tau}(y_i) - \epsilon_i] = \\
&\quad \sum_{\tau=0}^{K-1} Q_i(t+\tau)\left\{[F_{t+\tau}(y_i) - \epsilon_i]^+ + [F_{t+\tau}(y_i) - \epsilon_i]^-\right\} \overset{(a)}{\leq} \\
&\quad \sum_{\tau=0}^{K-1} \left\{ Q_i(t) + \sum_{j=0}^{\tau-1} ||F_{t+j}(y_i) - \epsilon_i||_1 \right\} [F_{t+\tau}(y_i) - \epsilon_i]^+ + \\
&\qquad \sum_{\tau=0}^{K-1} \left\{ Q_i(t) + \sum_{j=0}^{\tau-1} [F_{t+j}(y_i) - \epsilon_i] \right\} [F_{t+\tau}(y_i) - \epsilon_i]^- \overset{(b)}{\leq} \\
&\quad Q_i(t) \sum_{\tau=0}^{K-1} \{[F_{t+\tau}(y_i) - \epsilon_i]^+ + [F_{t+\tau}(y_i) - \epsilon_i]^-\} + \\
&\qquad \sum_{\tau=0}^{K-1} \left\{ \sum_{j=0}^{\tau-1} ||F_{t+j}(y_i) - \epsilon_i||_1 \right\} [F_{t+\tau}(y_i) - \epsilon_i)]^+ + \\
&\qquad \sum_{\tau=0}^{K-1} \left\{ \sum_{j=0}^{\tau-1} [F_{t+j}(y_i) - \epsilon_i] \right\} [F_{t+\tau}(y_i) - \epsilon_i]^- \overset{(c)}{\leq} \\
&\quad Q_i(t) \sum_{\tau=0}^{K-1} [F_{t+\tau}(y_i) - \epsilon_i] + \\
&\qquad \sum_{\tau=0}^{K-1} \left\{ \sum_{j=0}^{\tau-1} ||F_{j+\tau}(y_i) - \epsilon_i||_1 \right\} ||F_{t+\tau}(y_i) - \epsilon_i||_1 \overset{(d)}{\leq} \\
&\quad Q_i(t) \sum_{\tau=0}^{K-1} [F_{t+\tau}(y_i) - \epsilon_i] + 2B \sum_{\tau=0}^{K-1} \sum_{j=0}^{\tau-1} 1 \leq \\
&\quad Q_i(t) \sum_{\tau=0}^{K-1} [F_{t+\tau}(y_i) - \epsilon_i] + BK(K-1) \leq BK(K-1).
\end{aligned}
$$

For (a) we take the upper bound for queue on the positive terms (Eq.(5.12)) and the lower bound on the queue for the negative terms (Eq.(5.11)), this gives an upper bound on the total. In (b) we rewrite the equation by bringing in the front the common $Q(t)$ terms. Next, at (c) we upper bound the non-$Q(t)$ terms with their norm and we pass the norm to every element. Finally, (d) follows by taking the upper bound $||F_t(y_i)||_1 \leq \sqrt{2B}$. Summing for I queues proves the lemma. □

We take Eq.5.10, where we drop (a) and replace (b) by Lem.(3). We sum the inequality for $t = \{1, 2, \dots, T-K\}$:

$$\underbrace{\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} \Delta(t+\tau)}_{(c)} + \underbrace{V\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} \big[C(\boldsymbol{x}^{t+\tau})) - C(\boldsymbol{y})\big]}_{(d)} \leq$$

$$BK^2T + \underbrace{\alpha\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} ||\boldsymbol{y}-\boldsymbol{x}^{t-1}||^2 - \alpha\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} ||\boldsymbol{y}-\boldsymbol{x}^{t}||^2}_{(e)}.$$

To continue with the proof we need to lower bound the negative terms of the drift expression (c):

$$\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} \Delta(t+\tau) = \frac{1}{2}\sum_{\tau=0}^{K-1} Q(T-K+\tau+1)^2 -$$
$$\frac{1}{2}\sum_{\tau=0}^{K-1} Q(\tau+1)^2 \geq -\frac{1}{2}\sum_{\tau=0}^{K-1} Q(\tau+1)^2 \geq$$
$$-\frac{1}{2}\sum_{\tau=0}^{K-1} ((\tau+1)\sqrt{2B})^2 \geq -B\frac{K(K+1)(2K+1)}{6}.$$

The term (e) is bounded by $(e) \leq \alpha I D^2 K$, this is easy to show by the cancellation of the terms of the telescopic sum and by dropping the negative terms, it is omitted due to space limitation. Finally, to complete the sum of regret K times we need to add and subtract terms in (d):

$$\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} \big[C(\boldsymbol{x}^{t+\tau})) - C(\boldsymbol{y})\big] = K\sum_{t=1}^{T}\big[C(\boldsymbol{x}^{t})) - C(\boldsymbol{y})\big] -$$
$$\sum_{\tau=0}^{K-1}\sum_{t=1}^{\tau}\big[C(\boldsymbol{x}^{t})) - C(\boldsymbol{y})\big] - \sum_{\tau=0}^{K-1}\sum_{t=T-K+\tau+1}^{T}\big[C(\boldsymbol{x}^{t})) - C(\boldsymbol{y})\big] \geq$$
$$K\sum_{t=1}^{T}\big[C(\boldsymbol{x}^{t})) - C(\boldsymbol{y})\big] - DIc_iK(K-1).$$

where D is the max reservation, hence IKc_iD is the cost of assigning the maximum reservation for K iterations at all the I resources. By combining the results for (c),(d) and (e) and dividing by V and K we finish the proof of the lemma. $\square$

In the next subsection, we give a corollary that summarizes the performance guarantees of THOR, utilizing the results of Th.(1) and Th.(2).

5.4.4 THOR has No Regret and is Feasible

The following corollary, combines the theoretical results of the previous subsections to prove the performance guarantees of THOR.

Corollary 1 (No Regret against $K = o(T)$)**.** *Fix $\varepsilon > 0$, setting $\alpha = \frac{T^{\frac{3}{2}}}{V^{\frac{1}{2}}}$ and taking $K = T^{1-\varepsilon}$ and $V = T^{1-\frac{\varepsilon}{2}}$, THOR has no regret and is feasible:*

$$R_K(T) = O\left(T^{1-\frac{\varepsilon}{2}}\right),$$
$$Ctr(T) = O\left(T^{1-\frac{\varepsilon}{4}}\right).$$

Proof. We substitute in the expressions of Th.(1) and Th.(2) the values of α,K and V. Dropping the dominated terms completes the proof. □

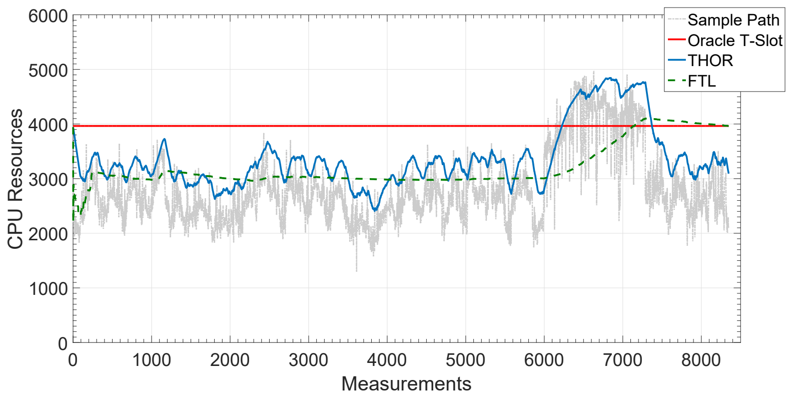
We have proven that using THOR reservations we achieve "no regret" against any K benchmark policy that has $K = o(T)$. This is an important finding that bridges the gap between [40], which proves "no regret" against $K = 1$ benchmark and [37], which proves that "no regret" is impossible against T benchmark, with adversarial time varying constraints.

5.5 Numerical Evaluation

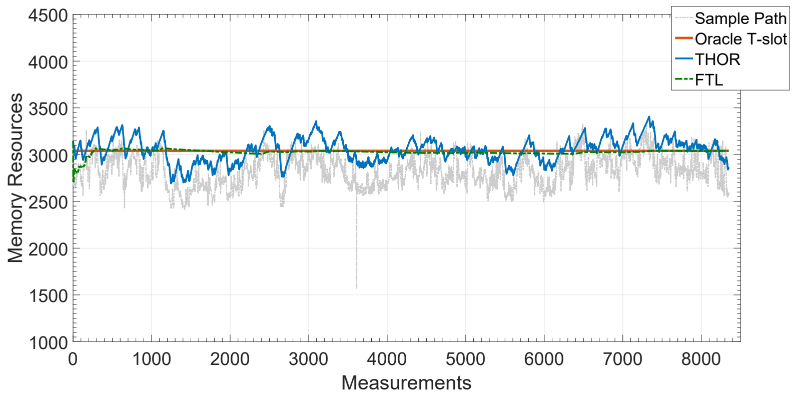
5.5.1 Google Cluster Data Analysis

In this subsection we compare the performance of our algorithm against an implementation of Follow The Leader (FTL), decribed in [79] and against the oracle best T slot reservation. The comparison is run on a public dataset from Google [2, 86]. The dataset contains detailed information about (i) measurements of actual usage of resources (ii) request for resources (iii) constraints of placing the resources in a big cluster comprised by 12500 machines. We will use the measurement of the actual usage of resources aggregated over the cluster. The time granularity of the measurements is 5 minutes for the period of 29 days and it is visualized in Fig. 5.1.

In Fig. 5.3 colored light gray is the time evolution of the workload demand sample path of the aggregate CPU (in subfig. 5.3(a)) and memory resources (in subfig. 5.3(b)). The blue line is THOR reservations, the red is the static oracle T-slot reservation and with dotted green the FTL reservations. In this experiment, ϵ_i is selected to be 10%. We see that for both resources, THOR predicts the resource reservation or quickly reacts to changes. On the other hand FTL is late to adapt, especially in the CPU case, causing many violations. In Table 5.3, the above are expressed in numbers, THOR achieves significantly reduced violations while also achieving the lowest cost ($\sim$ 10%) less than the best static T-slot policy.



(a) CPU Reservations



(b) Memory Reservations

Figure 5.3: Comparison Plots of reservation updates for different policies. Light gray on the background is the sample path of the actual resources required at the server. From the subfigures we can see that in the non-stationary case of CPU resource requirement, FTL policy is lagging significantly in resource reservation, incurring many more violations than the maximum 10%. For memory we can see that FTL is similar to the best static, while our algorithm achieves better performance by tracking the fluctuations. See Table 5.3 for the numerical comparison.

We run the same experiment, for a range of ϵ_i from very loose to very conservative violation guarantees. The numerical results are found in Tables 5.1 and 5.2. The result shows that THOR always achieves the violation guarantee, but also that as the constraint gets stricter, THOR is becoming more protective than necessary. This is most probably due to overestimation of the $\mathbf{\Lambda}$ distribution that caps the adversary decisions in theory, which is estimated by random sampling of max values. Furthermore, FTL again struggles to achieve the constraint in the whole range of guarantees, with slightly better performance in memory reservation. Even though THOR is more protective than necessary cost wise the performance is very similar to the best T-slot static policy, with the maximum difference to be less than 4% for very loose guarantee and actually achieving better performance on the tough CPU reservation, especially as the guarantee gets stronger.

Table 5.1: CPU performance comparison table according to guarantee

Guarantee	25%	20%	5%	1%	0.5%
T-Slot Vio	25.00	20.00	5.00	1.00	0.05
FTL Vio	36.28	31.29	13.63	5.41	3.57
THOR Vio	21.37	15.38	0.5	0	0
T-Slot Cost	2984	3124	4312	4682	4752
FTL Cost	2816	2896	3418	3740	3799
THOR Cost	3013	3193	3786	4238	4412

Table 5.2: Memory performance comparison table according to guarantee

Guarantee	25%	20%	5%	1%	0.5%
T-Slot Vio	25.00	20.00	5.00	1.00	0.05
FTL Vio	26.38	21.43	6.94	2.61	1.64
THOR Vio	16.38	10.95	0.4	0	0
T-Slot Cost	2943	2970	3096	3189	3225
FTL Cost	2937	2962	3071	3129	3159
THOR Cost	2968	3006	3181	3258	3293

Table 5.3: Policy Comparison Table, $\epsilon = 10\%$

Performance	T-slot Oracle	FTL	THOR
Average CPU Cost	3964	3203	3365
Average Violations (%)	10.00	21.41	5.64
Average MEM cost	3041	3054	3027
Average Violations (%)	10.00	11.84	3.7

5.6 Conclusion

In this chapter we introduce THOR, an online resource reservation policy for clouds. Cloud environments are very challenging due to volatile and unpredictable workloads. THOR uses (i) predictor queues and (ii) drift plus penalty plus smoothness, to fine tune reservations, such that overprovisioning is minimized while underprovisioning is maintained below a guaranteed ϵ_i parameter chosen by the system administrator. We prove that THOR is feasible in a growing horizon T and that it achieves similar performance to any sublinear to T, K-slot oracle static reservation. In simulation results using a public dataset from Google cluster, we vastly outperform the heavy to implement in practice follow the leader algorithm and perform similarly to the T-slot oracle policy.

In the next chapter, we will generalize THOR to a new framework for online learning and resource reservation with budget constraints. In this framework we strengthen the adversary to pick both the cost and constraint functions, we introduce the K-benchmark in the combined adversarial setting, a set of K-slot best cost K-slot feasible static policies, we generalize some of the base assumptions of prior work and we prove that our new policy, Cautious Online Lagrangian Descent (COLD), has “no regret” against K-benchmarks when $K = o(T)$.

Chapter 6

General Online Resource Reservation with Budget Constraints: A New Framework

6.1 Introduction

Consider the following bare-bones model of an online portfolio management problem: At each stage $t = 1, 2, \ldots$, an investor places an investment over d diverse goods. This investment is modeled as a vector of unit bid prices $x_t = (x_t^1, \ldots, x_t^d)$, with each x_i^t representing the amount of money the investor is willing to pay for a unit of the i-th good – for instance, for an advertiser requesting ad space from different publishers, x_i^t would denote the cost-per-click (CPC). Based on the performance of each individual asset (e.g., the number of clicks), the investor pays a total cost as a function of x_t^i and a performance parameter p_t^i of the i-th asset, and concurrently collects a corresponding reward w_t from their investment portfolio.

In our running example of online ad placement, the agent's utility $u_t(x_t; p_t)$ would be typically assumed concave in the agent's investment vector (to model diminishing returns), but otherwise stage-dependent, reflecting the variability of the performance parameter p_t^i of each investment. As such, utility maximization in this setting leads to an *online optimization problem* with the goal of maximizing the total reward $\sum_{t=1}^{T} u_t(x_t)$ accrued over T stages.

A considerable complication arises in this problem when the agent also needs to balance their total investment against an allotted budget (daily, monthly, or otherwise). In more detail, assume that the agent must meet a long-term budget constraint of the form $\sum_{t=1}^{T} c_t \leq b_T$, where $c_t = \langle p_t, x_t \rangle$ denotes the total expenditure of the consumer at time t. Since the performance parameters p_t are not known ahead of time (nor can they be assumed to follow a stationary probability law), techniques based on dynamic programming and optimal control cannot be applied in this context.

More generally, long-term budget constraints of this type can be formulated as

$$\sum_{t=1}^{T} g_t(x_t) \leq 0, \tag{6.1}$$

where g_t is a convex function representing the impact to the budget at time t. For instance, in our previous example, we have $g_t(x_t) = \langle p_t, x_t \rangle - b_T/T$, so (6.1) simply represents the target $\sum_{t=1}^{T} \langle p_t, x_t \rangle \leq b_T$. Importantly, these constraint functions are not only a priori unknown, but their evolution could even be adversarial: for instance, in online ad markets, competitors may click on ads to deplete their rivals' advertising budget, fraudulent publishers may attempt to manufacture revenue by increasing the click-through-rate (CTR) without legitimate buying intent, etc. [87].

Source	Constraint	K-window	Regret	Residual$^{(a)}$	Assumption
[88]	Fixed	T	$\mathcal{O}(\sqrt{T} + \frac{T}{V})$	$\mathcal{O}(\sqrt{VT})$	-
[89]	Stochastic	T	$\mathcal{O}(\sqrt{T})$	$\mathcal{O}(\sqrt{T})$	St. Slater$^{(b)}$
[40]	Adversarial	1	$\mathcal{O}(\sqrt{T})$	$\mathcal{O}(\sqrt{T})$	Slater$^{(c)}$
[90]	Adversarial	1	$\mathcal{O}(\sqrt{T})$	$\mathcal{O}(T^{3/4})$	-
[37]	Adversarial	T	$\Omega(T)$	$o(T)$	-
Us	Adversarial	K	$\mathcal{O}(\sqrt{T} + \frac{KT}{V})$	$\mathcal{O}(\sqrt{VT})$	-

Table 6.1: State of the art results in OCO with long-term budget constraints. All papers assume f_t, g_t are convex and Lipschitz continuous. (a) Residual refers to the long-term budget constraint violation. (b) Stochastic Slater assumes there exists an action $x_* \in \mathcal{X}$ such that $\mathbb{E}[g_t(x_*)] < 0$ for all t. (c) Slater assumes there exists an action $x_* \in \mathcal{X}$ such that $g_t(x_*) < 0$ for all t.

In this way, we obtain the following archetype of an *online optimization problem with long-term budget constraints:*

$$\begin{array}{ll} \text{minimize} & \sum_{t=1}^{T} f_t(x_t), \\ \text{subject to} & \sum_{t=1}^{T} g_t(x_t) \leq 0. \end{array} \tag{6.2}$$

In the above, the problem's loss and constraint functions (f_t and g_t respectively) are assumed convex and Lipschitz on their definition domain, but are otherwise arbitrary. Our aim in the rest of this chapter will thus be to *a*) quantify the trade-offs between regret minimization and budget violations in this setting; and *b*) propose online algorithms capable of achieving the problem's minimization objective while exceeding the allotted budget by a minimal amount.

6.1.1 Related work

When the agent's action are constrained to lie on a fixed convex set $\mathcal{X}$ – i.e., in the absence of long-term constraints – standard methods based on online gradient/mirror descent enjoy an $\mathcal{O}(\sqrt{T})$ bound on the incurred regret [39, 79], which is well-known to be optimal in this setting [91].

Beyond this classic regret minimization framework, the first work to examine online optimization problems with long-term budget constraints is [81], where the constraint functions are the same for all t, i.e., the constraint is of the form $\sum_t g(x_t) \leq 0$. For deterministic, non-adversarial constraints of this form, [81] achieved $\mathcal{O}(\sqrt{T})$ regret and an $\mathcal{O}(T^{2/3})$ constraint residual (defined here as $\sum_{t=1}^{T} g(x_t)$). These bounds were subsequently improved by [82] to $\mathcal{O}(T^{\max[\beta,1-\beta]})$ and $\mathcal{O}(T^{1-\beta/2})$ respectively, with $\beta \in (0,1)$ a free parameter, using varying stepsizes and regularization parameters. Finally, [88] generalized these works by proving the same regret and $\mathcal{O}(T^{1-\beta})$ constraint residual for the tighter constraints $\sum_{t=1}^{T} \left([g(x_t)]^+\right)^2$. All above works use roughly the same algorithm: at each round the dual variable of the long-term budget constraint is updated with g, and the algorithm takes a step along the (online) subgradient of the instantaneous augmented Lagrangian.

Moving on to time-varying constraint functions g_t, [89] examined the case where the losses f_t are adversarial but g_t are stochastic (non-adversarial), drawn from some unknown (but otherwise *stationary*) distribution. They define the regret with respect to the best action that satisfies $\mathbb{E}\left[g_t(x_*)\right] < 0$ at each round. Assuming such an action exists, a combination of OGD with a virtual queue playing the role of a dual relaxation variable guarantees a bound $\mathcal{O}(\sqrt{T})$ on both regret and constraint residual.

In a concurrent line of work by [92] and [93], the performance of an online optimization algorithm is compared to that of an instantaneous minimizer of f_t subject to $g_t(x) \leq 0$. As expected, regret guarantees against this dynamic comparator require very strong assumptions – for instance, that the total variation of the sequence of losses faced by the optimizer is bounded by the slack of a constant action (whose existence is difficult to guarantee).[1]

Our aim in this chapter is to study online convex optimization problems with time-varying (and possibly *adversarial*) long-term budget constraints. In this general setting, prior work by [37] provided a simple counterexample showing that the regret of any causal algorithm is lower bounded as $\Omega(T)$; as such, achieving no regret is impossible if the functions defining the agent's budget are chosen by an adversary. More recently, [40] proved that a combination of OGD with a virtual queue can indeed provide no regret, compared to a static action that is strictly feasible *for all* functions g_t, i.e., x_* must satisfy $g_t(x_*) < 0$ for all $t = \{1, \ldots, T\}$. Especially in an

[1]We should mention here that our work can also be extended in this direction using the work of [94]. However, because we want to focus on regret minimization with minimal assumptions, we only use static comparators throughout.

adversarial setting, this assumption might not be easy to achieve since, even a small degree of residual constraint violation injected by the adversary could disqualify *any* comparator action. More importantly, since x_* is artificially constrained in this way, the obtained regret guarantee can be fairly loose.

The state of the art regarding the above works is summarized in Table 6.1.

6.1.2 Our contributions

Our overarching objective is to examine the various trade-offs between regret minimization and long-term budget constraint violations. Our first contribution in this direction is the introduction of a refined regret metric which compares the agent's incurred losses to those of a "K-benchmark", i.e., a comparator which meets the problem's allotted budget over any window of length K. In other words, a K-benchmark comparator satisfies:

$$\sum_{\tau=t}^{t+K-1} g_\tau(x) \leq 0, \quad \forall t \in \{1, \ldots, T-K+1\} \tag{6.3}$$

and the "regret over a K-benchmark" compares the loss accrued by an online algorithm to that of the best K-benchmark in hindsight.

By varying K, this refined regret metric provides sufficient flexibility to study the difficult question of adversarial long-term budget constraints. Specifically, letting $K = T^\kappa$ for some $\kappa \in [0, 1]$, we recover the result of [37] for $\kappa = 1$ (i.e., every causal algorithm is regretful in the long run). At the other end of the spectrum, for $\kappa = 0$ – i.e., $K = \Theta(1)$ – we recover the framework of [40], where no regret *is* achievable. In this way, we are led to the following fundamental questions: (i) What is the largest κ for which "no regret over K-benchmark" can be achieved? and (ii) For a given $\kappa \in (0, 1)$, what is the regret guarantee for a given tolerance on the residual constraint violation?

Building on prior work by [82], [88], [89] and [40], we attack the first question by means of an online optimization policy which we call *cautious online Lagrangian descent* (COLD). As we show in the sequel, COLD achieves no regret over any benchmark of length $K = T^\kappa$, for all $\kappa \in [0, 1)$. Since no regret is impossible without further assumptions for $\kappa = 1$, our result closes the gap with respect to achieving no regret with adversarial long-term budget constraints. Finally, regarding the second question above, we show that the COLD algorithm can simultaneously achieve the tradeoffs

$$\underbrace{\mathcal{O}(KT/V + \sqrt{T})}_{\text{regret over } K\text{-benchmark}} \quad \text{and} \quad \underbrace{\mathcal{O}(\sqrt{VT})}_{\text{constraint residual}}$$

for any choice of $V \in [K, T)$. The "cautiousness parameter" V can be tuned at will by the optimizer and, in so doing, we derive the region of COLD relative to the trade-off between regret minimization and long-term residual constraint violation.

Our theoretical findings are also validated by a series of numerical experiments which suggest that increasing K – that is, enlarging the window over which the budget must be balanced – makes the K-benchmark guarantee tighter. Hence, proving "no regret over K-benchmark" for large K results in tighter performance guarantees – an observation which is not a priori obvious in a bona fide adversarial setting.

6.2 OCO with long-term budget constraints

To motivate the formal setup of the problem under study, we begin by discussing in more detail the online ad placement problem presented in Section 6.1.

Specifically, assume that, at every round, an advertiser chooses an investment vector of bid prices $x_t = (x_t^1, \dots, x_t^d)$ over d different websites, with x_t^i denoting the cost-per-click (CPC) for the i-th website. It is implied that each website offers different deals for ad display with varying position prominence and frequency of display, and accordingly arranged prices per click. The ultimate cost of an investment in dollars is determined when the number of click the ad receives (denoted here by p_t^i and measuring the performance of the corresponding investment) is revealed, and is equal to $\langle p_t, x_t \rangle$. In this setting, the values of p_t^i fluctuate in an unpredictable manner – for instance, following website popularity, viewer interest, and/or possible attacks by competitors who click on an ad without a legitimate intent to buy, but only to increase the cost to the advertiser. As a result, satisfying the monthly budget given by $\sum_t \langle p_t, x_t \rangle \leq b_T$ is an adversarial long-term budget constraint of the form (6.1).

The goal of the customer in this framework is to invest the available budget wisely. Specifically, the reward from ad display at site i also fluctuates unpredictably according to the website's popularity and the relation of the users to the advertized product (in some websites the value of a user click is higher since the user is more probable to eventually become a customer). With these considerations in mind, the collected utility is given by an unknown concave function $u_t(x_t) = \sum_i u_t^i(x_t^i)$. The concavity of u_t reflects the diminishing returns for a fixed website characteristic (e.g. making the ad more visible will attract proportionally less extra viewers). Needless to say, this task is very challenging because of the unpredictable fluctuations of price and reward, but also because an early agressive choice might consume the budget resulting in missing out on opportunities towards the end of the horizon.

6.2.1 Problem formulation and assumptions

To state the above in a more formal framework, we will focus on the online optimization problem (6.2) over a play horizon of $t = 1, \dots, T$ rounds. In round t, the action $x_t \in \mathcal{X}$ incurs $f_t(x_t)$ loss and impacts the budget by the amount $g_t(x_t)$. Here, functions f_t and g_t are not required to be differentiable and $f'_t(x_t), g'_t(x_t)$ denote subgradients at x_t. To analyze this problem we require the following basic assumptions.

(A1) The set $\mathcal{X}$ is convex and compact with diameter D.

(A2) For all $t = 1, \dots, T$ functions $f_t, g_t : \mathcal{X} \to \mathbb{R}$ are convex and Lipschitz, with $\|f'_t\|_2 \leq G$ and $\|g'_t\|_2 \leq G$.

(A3) For a given $K \leq T$, consider the set of all actions that maintain a balanced budget within all windows of K rounds:

$$\mathcal{X}_K = \left\{ x \in \mathcal{X} : \sum_{\tau=t}^{t+K-1} g_\tau(x) \leq 0,\ 1 \leq t \leq T - K + 1 \right\} \tag{6.4}$$

We assume that $\mathcal{X}_K$ is non-empty.

Since $\mathcal{X}$ is compact and f_t, g_t Lipschitz, it follows that they are also bounded, i.e., $|f_t(x)| \leq F$ and $|g_t(x)| \leq F$, for all $x \in \mathcal{X}$.

Assumptions A.1–A.2 are the blanket assumptions of all previous OCO papers. A.3 is essential in order to define the regret metric that we use, and is significantly less stringent than the nonempty interior Slater assumption $\cap_t\{x : g_t(x) < 0\}$ of [40]. For example, if g_t are nonnegative the Slater assumption cannot hold. This case appears in problems where we want to ensure that a rate of failures does not cross a threshold, as e.g., in [88].

6.2.2 Performance metric

We classify algorithms based on how they fare with respect to the constraint and the aggregate loss.

6.2.2.1 Feasibility

Regarding the constraint (6.1), we take the common approach in the OCO literature, that of a relaxed notion of feasibility.

Definition 1 (Asymptotic feasibility)**.** *An algorithm is asymptotically feasible if it satisfies:*

$$\sum_{t=1}^{T} g_t(x_t) = o(T).$$

An asymptotically feasible algorithm has the desirable property that it learns to produce a vanishingly small *constraint residual* $\sum_{t=1}^{T} g_t(x_t)$ over a large horizon T, i.e., we have $\sum_{t=1}^{T} g_t(x_t)/T \to 0$ as $T \to \infty$, and hence produces an asymptotically feasible solution of (6.2).

6.2.2.2 Regret over K-benchmark

The algorithmic efficiency is measured in OCO with the *static regret*: the aggregate loss difference between the algorithm and that of a benchmark action with hindsight. In our work, however, we encounter a further complication that does not appear in the literature. The appropriate regret definition must clarify how the benchmark action will behave with respect to the long-term budget constraint. To this end, we introduce the following family of benchmarks.

Definition 2 (K-benchmark)**.** *Fix a $K \in \{1, \dots, T\}$. The K-benchmark x_*^K is an action that satisfies:*

$$x_*^K \in \operatorname*{argmin}_{x \in \mathcal{X}_K} \sum_{t=1}^{T} f_t(x), \tag{6.5}$$

where $\mathcal{X}_K$ is defined in A.3.

This allows us to extend the definition of regret in the following manner.

Definition 3 (Regret of x_t over x_*^K)**.** *Fix $K \in \{1, \dots, T\}$, and suppose x_*^K is a K-benchmark. The regret of x_t over x_*^K is defined to be:*

$$R_K(T) = \sum_{t=1}^{T} f_t(x_t) - \sum_{t=1}^{T} f_t(x_*^K).$$

We make the following remarks:

Remark 1: If an asymptotically feasible online algorithm has *no regret* over x_*^K, it follows that the average losses $R_K(T)/T \to 0$ as $T \to \infty$, which implies that our policy approximates the benchmark x_*^K under any sequence of functions, while additionally asymptotically satisfying the long-term budget constraint.

Remark 2: The actual guarantee provided by the novel regret criterion depends on K. As K increases, actions in $\mathcal{X}_K$ must balance the budget in longer periods, and therefore become more aggressive. In fact, it can be checked that $\mathcal{X}_1 \subseteq \mathcal{X}_K$, hence $x_*^1 \leq x_*^K$ for all K. Consequently, a no regret guarantee over x_*^K is tighter than a no regret guarantee over x_*^1; section 6.5 illustrates this with a numerical example. We are, therefore, motivated to prove no regret for as large K as possible.

Interestingly, however, for $K > 1$ the K-benchmarks are not necessarily monotonic in K as the next example shows.

Example 1 (Non-monotonicity of K-benchmark)**.** Consider the ad display example with only one website. The K-benchmark is the largest action x constrained to the use of Kb_T/T budget within every K-round window.

Given hindsight and prices $p_1, \dots, p_T$, and assuming non-increasing f_t (i.e. more investment means larger utility and smaller loss), we have

$$x_*^K = \frac{K b_T}{T \max_{t=0,\dots,T-K-1} \sum_{\tau=t+1}^{t+K} p_\tau}.$$

Consider the instance where $T = 3$, $p = (10, 0, 8)$, and $b_T = 30$. We have $x_*^1 = 1, x_*^2 = 2, x_*^3 = 5/3$, and surprisingly $x_*^2 > x_*^3$. The latter is the consequence of the window of size 3 having the value at the two ends higher than the mean.

The lack of monotonicity is indicative of a powerful adversary who can tweak functions g_t to disturb the agent's algorithm in non-trivial ways. In spite of this complication, we present next a general algorithm that establishes no regret over x_*^K for any $K = o(T)$. Additionally, although a strong monotonicity result can not be established, in the numerical section we observe an improvement trend with increasing K, verifying the intuition that a larger window allows the agent to handle its budget in a better manner.

6.3 The algorithm

The main idea behind handling the long-term budget constraints is to weigh their importance against the loss in a Lagrangian fashion. Specifically, consider a *regularized instantaneous Lagrangian* for problem (6.2) that takes the following form in round t:

$$L_t(x, Q(t)) = V f_t(x) + Q(t) g_t(x) + \alpha \|x - x_{t-1}\|^2, \tag{6.6}$$

where *a)* V is a configurable *cautiousness parameter*; *b)* $Q(t)$ is a virtual queue that plays the role of the Lagrangian multiplier; *c)* the term $\|x - x_{t-1}\|^2$ is a L_2 regularizer that smoothens the differences between consecutive actions; and *d)* α is the strength of the regularization.

The main difference between (6.6) and traditional Lagrangian relaxations is that the cautiousness parameter V can be used to control the tradeoff between regret and constraint residual (smaller V makes the algorithm more cautious); likewise, the regularization parameter α can be tuned to enhance the algorithm's robustness to fluctuations.

To estimate the value of the (otherwise unknown) functions f_t and g_t, we will employ their linear surrogates:

$$\hat{f}_t(x) \triangleq f_{t-1}(x_{t-1}) + \langle f'_{t-1}(x_{t-1}), x - x_{t-1} \rangle, \tag{6.7}$$

$$\hat{g}_t(x) \triangleq g_{t-1}(x_{t-1}) + \langle g'_{t-1}(x_{t-1}), x - x_{t-1} \rangle. \tag{6.8}$$

Let $L'_t(x, Q)$ denote the subgradient of $L_t(x, Q(t))$ at x. Plugging the above surrogate terms in (6.6) and using basic subgradient algebra, we get:

$$L'_t(x, Q) = V f'_{t-1}(x_{t-1}) + Q(t) g'_{t-1}(x_{t-1}) + 2\alpha(x - x_{t-1})$$

Our algorithm is designed to compute a stationary point of L_t in round t, and then project it on $\mathcal{X}$, while updating queue $Q(t+1)$ with the surrogate $\hat{g}_t(x_t)$. The latter is in fact the subgradient of $L_t(x, Q)$ with respect to Q.

Cautious Online Lagrangian Descent (COLD)

For $t = 1, \dots, T$:

$$x_t = \Pi_{\mathcal{X}}\left[x_{t-1} - \frac{V f'_{t-1}(x_{t-1}) + Q(t) g'_{t-1}(x_{t-1})}{2\alpha}\right] \tag{6.9}$$

$$Q(t+1) = [Q(t) + \hat{g}_t(x_t)]^+. \tag{6.10}$$

with initialization $Q(1) = 0$, $x_0 \in \mathcal{X}$, and where:

- $\Pi_{\mathcal{X}}[.]$ denotes the Euclidean projection on set $\mathcal{X}$,
- V is the configurable cautiousness parameter,
- f'_{t-1}, g'_{t-1} are the subgradient vectors in round $t-1$,
- $Q(t)$ is a virtual queue that is updated according to (6.10), and it is called the *predictor queue*,
- α is the configurable regularization strength parameter,
- $\hat{g}_t(x_t)$ is the surrogate of $g_t(x_t)$ from (6.8),
- $[.]^+$ is $\max\{., 0\}$,

We remark that if we fix $Q(t) = 0, \forall t$, COLD reduces to the OGD of [39] with stepsize $V/2\alpha$, which however would fail to address the long-term budget constraints. In what follows, we provide the logical steps that are used in the design of COLD, as well as to prove its performance guarantees. The same steps can be used to derive variations of COLD for different problems.

6.3.1 Regularized drift plus loss framework

The framework is inspired by the unification of two theories, namely the theory of stochastic network optimization [38] that handles time-average constraints by stabilizing virtual queues, and the standard framework of OCO [39, 79, 80].

Our mathematical analysis is based on an instantaneous metric called *Drift plus loss plus smoothness* (DPLPS), which at each round measures the quality of an action by weighing three competing factors: (i) the *quadratic Lyapunov Drift* of the predictor queue (which reflects the urgency of the constraint), (ii) the predicted instantaneous loss, and (iii) the L_2 regularizer to smoothen the changes in the sequence of actions. The values of all three above depend on the action x_t, and we will show that our algorithm arises as the action that minimizes an upper bound of the DPLPS. The remaining of this subsection provides further detail.

We first define the quadratic *Lyapunov Drift* as the change in the quadratic predictor queue length after action x_t is taken:

$$\Delta(x_t) \triangleq \frac{1}{2}[Q^2(t+1) - Q^2(t)]. \tag{6.11}$$

Since x_t determines $\hat{g}_t(x_t)$, it implicitly affects $\Delta(x_t)$ via $Q(t+1)$, see (6.10). By taking actions to minimize the drift $\Delta(x_t)$ we may keep the queue length small at the end of the horizon. We will then show that a bound on $Q(T)$ can be manipulated into proving asymptotic feasibility. In summary, the minimization of $\Delta(x_t)$ at each round pushes towards actions that satisfy the long-term budget constraint.

From the literature of stochastic network optimization [38], the drift can be combined with the instantaneous loss (here adapted to its surrogate from (6.7)): $\Delta(x_t) + V\hat{f}_t(x_t)$. Minimizing this weighted metric pushes towards actions that simultaneously satisfy the long-term budget constraints and achieve low aggregate loss in the stochastic setting, see [95]. However, such algorithms have a bang-bang behavior, oscillating between extreme actions, which is inappropriate for many real applications. Therefore, in this work we further add the L_2 regularizer (following the methodology of [40]), which brings us to the DPLPS metric:

$$DPLPS(x_t) \triangleq \Delta(x_t) + V\hat{f}_t(x_t) + \alpha||x_t - x_{t-1}||_2^2. \tag{6.12}$$

With some work on the definition of the Lyapunov drift (see also Lemma 4.2 in [84]), we have the following inequality:

$$\Delta(x_t) \leq B + Q(t)\hat{g}_t(x),$$

where $B \triangleq (F + GD)^2/2$ is a constant. Therefore, we have

$$DPLPS(x) \leq \underbrace{B + V\hat{f}_t(x) + Q(t)\hat{g}_t(x) + \alpha||x - x_{t-1}||_2^2}_{r_t(x)}.$$

Observe that the term $r_t(x)$ equals (6.6), plus a constant B (which does not affect its stationary point). The following Lemma proves formally that $r_t(x)$ is minimized by COLD at each round.

Lemma 1 (DPLPS bound minimizer)**.** *COLD minimizes $r_t(x)$ at each round.*

Proof. First, observe that by definition our policy is:

$$x_t \triangleq \Pi_{\mathcal{X}}\left[x_{t-1} + \frac{H_t}{2\alpha}\right],$$

where H_t is the following constant:

$$H_t \triangleq Vf'_{t-1}(x_{t-1}) + Q(t)g'_{t-1}(x_{t-1}).$$

It suffices to show that this projection actually returns a minimizer of $r_t(x)$. We have that, $\langle H_t, x_t - x_{t-1}\rangle = Q(t)\langle g'_{t-1}(x_{t-1}), (x_t - x_{t-1})\rangle +$

$V\langle f'_{t-1}(x_{t-1}), (x_t - x_{t-1})\rangle$.

$$\begin{aligned}
x_t &= \underset{x\in\mathcal{X}}{\operatorname{argmin}} \{r_t(x)\} \\
&\overset{(a)}{=} \underset{x\in\mathcal{X}}{\operatorname{argmin}} \{\langle H_t, x - x_{t-1}\rangle + \alpha||x - x_{t-1}||_2^2\} \\
&\overset{(b)}{=} \underset{x\in\mathcal{X}}{\operatorname{argmin}} \left\{\langle H_t, x - x_{t-1}\rangle + \alpha||x - x_{t-1}||_2^2 + \frac{H_t^2}{4\alpha}\right\} \\
&= \underset{x\in\mathcal{X}}{\operatorname{argmin}} \left|\left|\frac{H_t}{2\sqrt{\alpha}} + \sqrt{\alpha}(x - x_{t-1})\right|\right|_2^2 \\
&\overset{(c)}{=} \underset{x\in\mathcal{X}}{\operatorname{argmin}} \left|\left|x - \left(x_{t-1} - \frac{H_t}{2\alpha}\right)\right|\right|_2^2 = \Pi_{\mathcal{X}}\left[x_{t-1} - \frac{H_t}{2\alpha}\right].
\end{aligned}$$

In (a) we discard the terms that do not depend on x. In (b) we add a constant term that completes the square norm but does not change the minimizer. Finally, (c) is the definition of Euclidean projection on set $\mathcal{X}$, hence the last equality follows and completes the proof. □

This methodology constitutes a powerful framework, where for a new problem we may define appropriate virtual queues for the constraints, determine the corresponding DPLPS metric, and then extract asymptotically feasible online algorithms by minimizing a bound on the DPLPS. In the technical proofs, we show how this bound minimization can be used to derive the performance guarantees of our algorithm.

6.4 Performance analysis

In this section we provide our main theoretical results, which characterize the performance of COLD algorithm. Recall that T is the horizon, V, α are configurable parameters, F, G are universal constants that bound functions and subgradient norms (see section 6.2.1), D is the diameter of set $\mathcal{X}$, and $B = (F + GD)^2/2$.

Proposition 1 (COLD performance)**.** *If the assumptions in Sec.* (6.2.1) *are satisfied, and actions are taken according to the COLD algorithm, the* *constraint residual is bounded by:*

$$\begin{aligned}
Ctr(T) \triangleq \sum_{t=1}^{T} g_t(x_t) \leq \Bigg\{ 2BKT + 4FVT + \frac{V^2G^2T}{\alpha} + \\
2\alpha D^2 + 2BK^2 + 2(T+1)BK \Bigg\}^{\frac{1}{2}} + \frac{GVT}{2\alpha} + \\
\frac{G^2\left[\sqrt{2BK} + \sqrt{4FV} + \sqrt{\frac{V^2G^2}{\alpha}} + \sqrt{2BK}\right]\frac{T^{\frac{3}{2}}+T}{\sqrt{2}}}{2\alpha} + \\
\frac{G^2\left[\sqrt{2\alpha D^2} + \sqrt{2BK^2} + \sqrt{2BK}\right]T}{2\alpha}
\end{aligned} \tag{6.13}$$

and the regret over the K-benchmark is bounded by:

$$\begin{aligned}
R_K(T) \triangleq \sum_{t=1}^{T} f_t(x_t) - \sum_{t=1}^{T} f_t(x_*^K) \leq \\
\leq \frac{BKT}{V} + \frac{G^2VT}{2\alpha} + B\frac{(K+1)(2K+1)}{6V} + \\
+ \frac{D^2\alpha}{V} + 2F(K-1).
\end{aligned} \tag{6.14}$$

Based on the above fundamental bounds, we optimize the parameters V, α to get a region of achievable asymptotic laws, such that both the constraint residual and the regret are $o(T)$. In general, we may choose the value of parameter V in the range (K, T) and different choices provide different tradeoffs. For instance, choosing V close to K makes the algorithm cautious and provides the best guarantees on the constraint residual, while choosing it close to T makes the algorithm to aggressively pursue the best regret.

Theorem 1 (Achievable tradeoffs)**.** *Fix $K \geq 1$ such that $K = o(T)$ (higher K makes the K-benchmark tighter). Choose some $V \in (K, T)$, and $\alpha = \max\{T, V\sqrt{T}\}$. Then,* (6.13)-(6.14) *simplify to:*

$$\underbrace{\mathcal{O}(KT/V + \sqrt{T})}_{\text{regret over } K\text{-benchmark}} \quad \text{and} \quad \underbrace{\mathcal{O}(\sqrt{VT})}_{\text{constraint residual}}$$

Furthermore, suppose $K = T^{1-\epsilon}$ for some small $\epsilon > 0$ and choose $V = T^{1-\frac{\epsilon}{2}}$, and $\alpha = V\sqrt{T}$. Then, (6.13)-(6.14) *simplify to:*

$$\underbrace{\mathcal{O}(T^{1-\frac{\epsilon}{2}})}_{\text{regret over } T^{1-\epsilon}\text{-benchmark}} \quad \text{and} \quad \underbrace{\mathcal{O}(T^{1-\frac{\epsilon}{4}})}_{\text{constraint residual}}$$

In fig. 6.1 we showcase the results of theorem 1, where the black curves indicate the Pareto frontier for different values of $K = T^k$; all the values

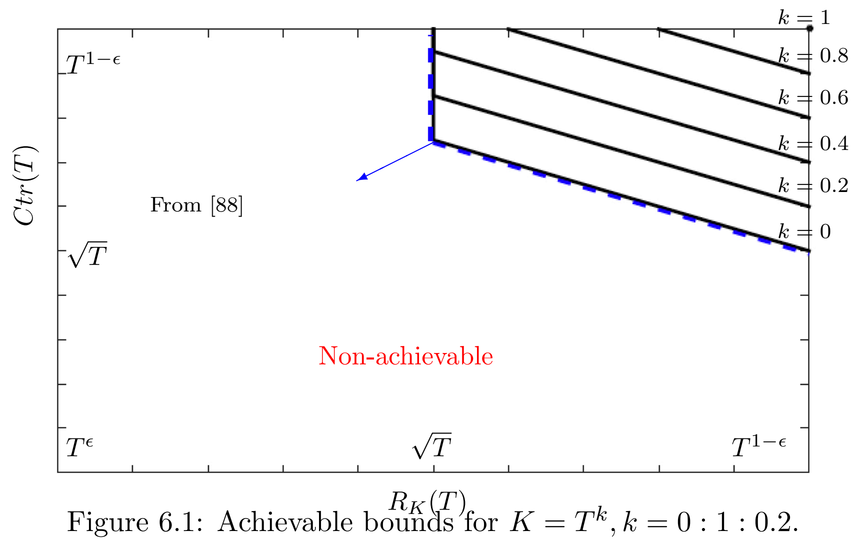


Figure 6.1: Achievable bounds for $K = T^k, k = 0 : 1 : 0.2$.

north-east of the frontier are achievable. As $K \to T$, the Pareto frontier vanishes to the north-east corner point $\mathcal{O}(T^{1-\epsilon/2}), \mathcal{O}(T^{1-\epsilon/4})$. The blue dotted line shows the tradeoffs achieved by [88], which interestingly coincide with our case of $K = 1$, though we mention that they address fixed (non-adversarial) constraints. Finally, the point $\mathcal{O}(\sqrt{T}), \mathcal{O}(\sqrt{T})$ achieved by [40] for $K = 1$ is not part of our achievable guarantees, but this is attributed to their stricter Slater assumption.

6.4.1 Outline of the proofs

The technical proofs of Prop. (1) and Th. (1) are deferred to the appendix (in supplemental material) due to space limitations. Here, we provide a brief outline.

The COLD algorithm is designed to minimize the DPLPS, hence one can directly compare it to the 1-benchmark, as for example in [40]. The novel element in our analysis is that we compare K steps of DPLPS of COLD to the DPLPS of the K-benchmark (Def.(2)). This comparison forms the basis of our analysis, and it is given in Lem.(7) and Cor.(2) in the appendix.

First, based on the feasibility of the K-benchmark in windows of size K, we establish a bound on $Q(t)$ of COLD for any $t \in \{1, \dots, T+1\}$. The bound builds on the above-explained comparison between COLD and the K-benchmark. Then, the bound on $Q(t)$ is manipulated into proving the upper bound on the constraint residual Eq. (6.13). We note that an important part of the proof is to obtain a good upper bound of $\sum_{t=1}^{T} Q(t)$. A similar strategy is used in comparing the losses of COLD to those of the K-benchmark and proving the regret bound Eq. (6.14).

We mention that the bound on $Q(t)$ can be strengthened if we make the Slater assumption, i.e., assume the existence of a vector x_* such that

$g_t(x_*) < -\eta, \quad \forall t$. In particular, this is the approach taken by [40] achieving the point $(\mathcal{O}(\sqrt{T}), \mathcal{O}(\sqrt{T}))$ for $K = 1$. In our case, we would assume the existence of actions satisfying $\sum_{\tau=t}^{t+K-1} g_\tau(x) < -\eta, \quad \forall t$, with which we could improve further the bounds on $Q(t)$, $\sum_{t=1}^{T} Q(t)$ and eventually our Proposition. In this work, we chose to present the most general bounds, and left this direction for future research.

Finally, regarding the proofs of Th. (1), our strategy is to restrict progressively the values of K, α, and V, such that both $Ctr(T)$ and $R_K(T)$ are $o(T)$. An optimization over parameters α, V provides the presented trade-offs.

6.5 Numerical results

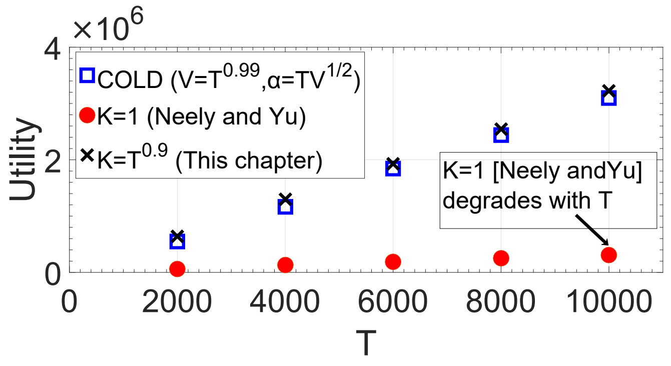
In this section we test COLD and the performance guarantees given by K-benchmarks on an instance of our example application of online ad placement. We also showcase the effect of the cautiousness parameter V on COLD's utility and constraint residual.

6.5.1 Accuracy of performance guarantee

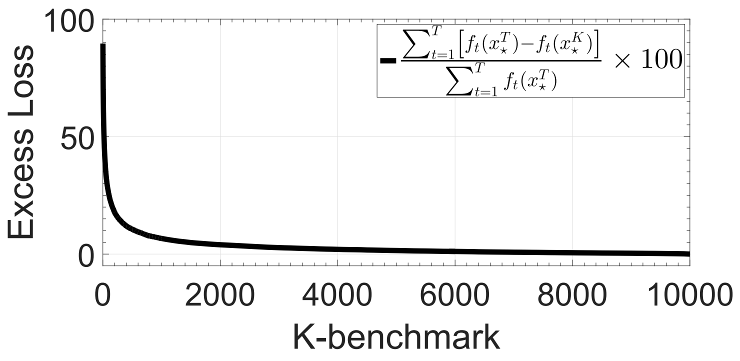
We simulate a scenario with one website, where $x_t \in [0, \infty)$, $f_t(x_t) = -w_t x_t$, and $g_t(x_t) = p_t x_t - b_T/T$, where w_t, p_t are generated by exponential distributions $w_t \sim Exp(11)$ and $p_t \sim Exp(10)$. We run the experiment for different horizons $T = \{2000, 4000, \ldots, 10000\}$, budget $b_T = 300T$, and parameters set to $\alpha = \max\{T, V\sqrt{T}\}$ and $V = T^{0.99}$ for each of the experiments. Fig. (6.2(a)) shows the utility (minus the loss) for the 1-benchmark [40], the $T^{0.9}$-benchmark (an instance of this chapter) and finally the utility achieved by the COLD algorithm.

In Fig. (6.2(a)) we observe that COLD's utility is approximated much more accurately by the $T^{0.9}$-benchmark than by 1-benchmark. In fact, the approximation by the 1-benchmark becomes even worse as T increases. Since proving no regret over K-benchmark essentially shows that the algorithm's losses approximate those of the K-benchmark, we conclude that the importance of the regret guarantee lies with how large K is.

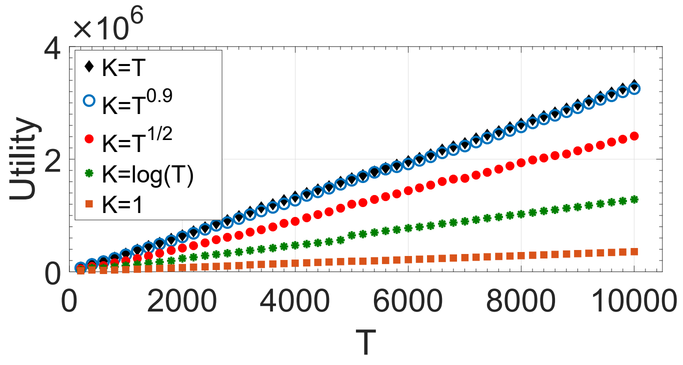
In a similar experiment, Fig. (6.2(b)) presents the relative excess loss of the K-benchmark with respect to the T-benchmark, for all values of $K = 1, \ldots, T$. This relative excess loss is depictive of the approximation error of a K-benchmark with respect to the T-benchmark. We remind the reader that the excess loss is due to constraining the benchmark action to be feasible on a window that is $K < T$, and that the regret guarantees can be established for $K = o(T)$ only. The points are averages over 150 sample paths. Here we may observe a monotonous behavior of the K-benchmarks due to averaging, but more importantly, we can see that the 1-benchmark can have as much as 85% excess loss.



(a) COLD vs $K = 1, K = T^{0.9}$



(b) K-benchmark vs T-benchmark



(c) Utility for different scaling of K

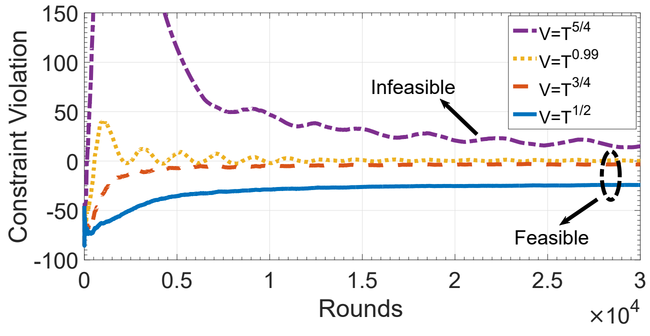
Figure 6.2: Simulations of the example of online ad placement. (a) COLD utility comparison versus 1-benchmark and $T^{0.9}$-benchmark. (b) Relative excess loss of K-benchmark compared to T-benchmark. (c) Utilities of K-benchmark for different values of K.

On Fig. (6.2(c)), we observe that a K-benchmark that scales sublinearly with the horizon ($K = o(T)$) has a much better approximation compared to a constant K, like $K = 1$, which is very pessimistic even for moderate values of T. All the above experiments support that intuition that taking K large produces a tighter regret guarantee.

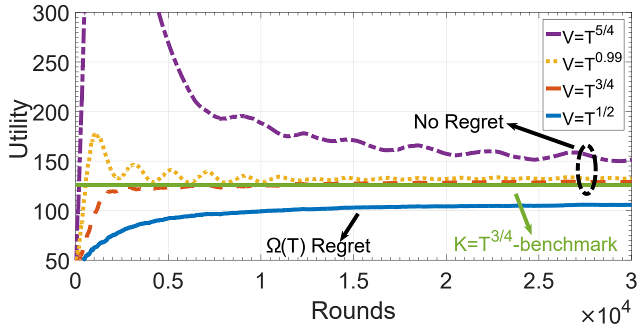
6.5.2 Impact of the Cautiousness Parameter V

In this subsection we explore the effect of the cautiousness parameter V on the performance of COLD algorithm. On the same example as before, we choose $K = T^{3/4}$ and $V = \{T^{1/2}, T^{3/4}, T^{0.99}, T^{5/4}\}$. In our theoretical analysis, we have proven that V has to be greater than K to achieve no regret. On Fig. (6.3(b)) the running average utility for $V = T^{1/2}$ indeed fails to reach the benchmark. Furthermore, we can deduce from Fig. (6.3(a)) that, for $V > T$, the constraint residual is not sub-linear to the horizon T, which is in accordance to our bounds.

On the other hand when $K < V < T$, indeed the constraint residuals are sub-linear and the average utility approximates the utility of the K-benchmark. Increasing V up to T, one can observe in Fig. (6.3) the different tradeoffs between the regret and the constraint residual.



(a) $\sum_{\tau=1}^{t} g_\tau(x_\tau)/t$ of COLD



(b) $-\sum_{\tau=1}^{t} f_\tau(x_\tau)/t$ of COLD

Figure 6.3: Running averages of constraint residual and utility performance of COLD for different values of parameter V.

6.6 Conclusion

In this chapter we studied Online Convex Optimization with long-term budget constraints. In particular, we deal with the case where the constraints are adversarial, which captures the relevant scenario where the long-term budget constraint must be addressed in the presence of poor prediction quality. We introduce the notion of K-benchmark, which allows us to refine the regret metric used to provide performance guarantees for online algorithms. Although for $K = T$ prior work has established that no algorithm can provide no regret, we prove that the COLD algorithm achieves no regret for any $K = o(T)$. Our numerical results suggest that a K-benchmark with K large can provide a more accurate performance guarantee than the previous state of the art $K = 1$. Finally, we provide a new region of regret-constraint residual tradeoffs that characterize the performance of COLD in the general setting.

Chapter 7

Conclusions and Future Research

7.1 Conclusions

The exponential increase of wireless traffic, followed by the proliferation of wireless devices, services and network resources leads to a new paradigm of management and organization of wireless networks. We identify optimization as a very important tool, which will greatly increase resource utilization efficiency and greatly reduce operational cost. In the manuscript, we propose various algorithmic frameworks for wireless networks, based both on classical or data-driven optimization and machine learning to augment the arsenal of algorithms against resource allocation network problems.

The contributions of this thesis are separated into two main parts. The first part corresponds to chapters 2 and 3, where we focus on optimizing resource utilization of networks in real time; while in chapters 5 and 6 we consider proactive resource reservations. Meanwhile, chapter 4, serves as a bridge between the two parts, as it tackles an online problem with an offline data-driven solution. In the thesis, we present two main use cases: *user association* and *cloud resource reservation*, but the solutions presented can be applied to a plethora of problems, sometimes out of the scope of networking (prominent examples are chapters 4 and 6).

The baseline approach for *user association*, connecting wireless devices to the base station that provides the MaxSINR, can lead to very inefficient configurations in current and future wireless networks. We focused on tailoring user association based on resource efficiency and service requirements satisfaction (QoS guarantees), depending on the underlying network demand.

In chapter 2 we have proposed a framework for user association based on distributed constrained optimization for 5G New Radio (NR) in the context of future Ultra Dense Networks (UDNs). We have derived distributed association rules, that provably converge to the optimum point of operation.

The resulting association guarantees performance to the VIP flows whilst balancing the load between both service types. The method is based on non-invasive extensions to current wireless networks, while it can be generalized in future work for multiple class priorities and applications in wireless network slicing. Initial Simulation results demonstrate the capabilities of the framework in bounding the mean number of VIP flows at the base stations and also the improved performance over best effort only policies.

Meanwhile, In chapter 3, we studied centralized device association, where the computation is done in the C-RAN, using the Sinkhorn algorithm inspired by the theory of Optimal Transport. This algorithm has been previously used in large-scale problems in imaging and machine learning, and here it is applied to provide a heuristic load balancer in C-RAN systems. First, it is shown that a very simple version of this algorithm is capable of providing low delay associations when the traffic is uniformly spread in the covered geographical area. In case of non-uniform traffic, we extend the algorithm to an iterative version which progressively improves the load balancing. We show that our scheme scales to very large problem instances, and has the potential to provide great improvements over the simple baseline approach.

Finally, in chapter 4, we showed that past traffic data can be exploited towards precalculating association maps, which are designed to be robust and can be tuned to protect the base stations from overload. Accordingly, we proposed a theoretical framework for efficiently computing the optimal robust map, parametrized to a large class of utility functions that allow the system designer to tune the base station load. Finally, we evaluated our approach in Milano dataset, and found that our methodology is very effective at protecting UDNs from unexpected spikes, allowing the offering of premium wireless service.

Moving to the topic of *cloud resource reservation*, we developed a framework to handle resource reservation in worst-case scenaria, where the demand is engineered by an adversary aiming to harm our performance. We provide policies that have “no regret” and guarantee *asymptotic feasibility in budget constraints*, under such workloads, complementing the results of recent literature in cloud computing and more importantly in OCO.

In chapter 5 we introduce THOR, an online resource reservation policy for clouds. Cloud environments exhibit volatile and unpredictable workloads. THOR uses (i) predictor queues and (ii) drift plus penalty plus smoothness, to fine tune reservations, such that overprovisioning is minimized while underprovisioning is maintained below a guaranteed ϵ_i parameter chosen by the system administrator. We prove that THOR is feasible in a growing horizon T and that it achieves similar performance to any sublinear to T, K-slot oracle static reservation. In simulation results using a public dataset from Google cluster, we vastly outperform the heavy to implement in practice follow the leader algorithm and perform similarly to the T-slot oracle policy.

We conclude our thesis contributions with chapter 6, which generalizes the results of chapter 5. We studied the general case of OCO with adversarial cost and with long-term budget constraints. In particular, we deal with the case where the constraints are adversarial, which captures the relevant scenario where the long-term budget constraint must be addressed in the presence of poor prediction quality. We introduce the notion of K-benchmark, which allows us to refine the regret metric used to provide performance guarantees for online algorithms. Although for $K = T$ prior work has established that no algorithm can provide no regret, we prove that the COLD algorithm achieves no regret for any $K = o(T)$. Our numerical results suggest that a K-benchmark with K large can provide a more accurate performance guarantee than the previous state of the art $K = 1$. Finally, we provide a new region of regret-constraint residual trade-offs that characterize the performance of COLD in the general setting.

7.2 Future Work

We mainly consider three promising directions for future research on the topics presented in the thesis. First, we could focus on generalizing the theoretical frameworks to improve the performance guarantees and the areas of application; next, we could consider different problems that can be cast and solved efficiently by our proposed frameworks (more notably in chapters 4 and 6) and lastly, it would be interesting to perform the numerical evaluation of different optimization methods on similar use cases, giving concrete performance comparisons.

Theory Generalization. A promising direction would be to generalize the theory in chapter 6 and derive policies for different regularizers, which fit the structure of the underlying problem. One prominent example, found in convex optimization literature, is the entropic regularizer, which has great properties when the control variables are in the probability simplex.

Different Applications. Furthermore, we have envisioned application of the online budget constrained framework (seen in chapter 6) on wireless optimization problems. The problems considered include medium access control with constrained power budget (IoT, sensors) and adaptive streaming algorithms, which control the download video quality to devices.

Method Comparison. In the manuscript we have applied a plethora of mathematical tools for optimizing resource allocation in wireless networks, studying varying regimes. An important future direction would be to compare these methods, highlighting the pros and cons of each method depending on the underlying network environment. This could give an answer to the important challenging question of "which is the correct optimization technique to apply?". In the following subsection, some preliminary work can be found, on providing direct quantitative comparison between different optimization techniques on the user association (traffic steering) use case, using real data. This work has been published in [41].

7.2.1 Online Network Optimization on Traffic Steering

In this, work presented in [41], we work towards automating online network optimization, and we examine several approaches such as data-driven optimization chapter 4, online learning[1], and model-based AI techniques. We perform a direct comparison of these techniques for the challenging problem of traffic steering and load balancing in dense, heterogeneous networks. Our study demonstrates how the inherent variability of network traffic can be successfully addressed by optimization methodologies. The most prominent in the specific scenario, is based on artificial intelligence, but also makes use of deep modeling insights of the problem.

7.2.1.1 Performance Evaluation and Comparison

The performance evaluation results are obtained by feeding our simulator with the Milan dataset [33], also used in chapter 4 and the associated network topology.

Table 7.1: Summarized statistics per day and for the whole period: (i) average delay in *ms* over all BSs at daytime (i.e., between 7:30 and 20:30); (ii) mean squared error (MSE) of average delay over all BSs against the optimal delay; (iii) rejected traffic in percentage of the total traffic. To compute average delay and MSE values, time slots with infinite delay are omitted. We highlight the lowest value (per day and overall) in bold.

	Method	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**	**Avg**
Avg delay	Robust UAM	10.95	11.90	10.01	9.50	9.41	10.35
	Online learning	11.62	11.50	9.73	**8.79**	8.52	10.03
	Problem-adapted AI	**9.90**	**10.09**	**9.13**	9.07	**8.43**	**9.32**
	Oracle	*9.57*	*9.81*	*8.86*	*8.54*	*8.35*	*9.02*
MSE	Robust UAM	2.30	17.23	1.61	0.95	1.21	4.66
	Online learning	8.63	21.66	9.29	**0.18**	0.04	7.96
	Problem-adapted AI	**0.49**	**0.95**	**0.25**	2.63	**0.02**	**0.87**
Rejected	Robust UAM	0.00	0.06	0.00	0.00	0.00	**0.01**
	Online learning	1.10	0.24	0.00	0.00	0.00	0.27
	Problem-adapted AI	0.00	0.26	0.00	0.00	0.00	0.05

[1]without budget (QoS) constraints as in chapters 5 and 6.

Table 7.1 presents average KPIs of applying the 3 ONO techniques on the same data. The experimental results lead us to the following observations:

1) Robust optimization can be tuned to trade-off average cost / delay against excessive overload and thus protect against SLA violations. In the simulated scenario, we configured the method to limit the percentage of rejected traffic to less than 0.1%, and this requirement is indeed satisfied (see Table 7.1). However, due to the strong conservative parameter against failure, the average delay is increased compared to the other methods.

2) Online learning adapts relatively quickly to a good state, though sometimes fails to react to a rapid increase in traffic. As our algorithm makes no assumptions on the statistics of traffic fluctuations, it is best suited for applications where $\boldsymbol{\lambda}_t$ is highly non-stationary. When $\boldsymbol{\lambda}_t$ follows a cyclo-stationary process (as in our case) alternative methods which make use of acquired knowledge on the statistics of the underline process are advantageous. However, as the presented solution does not apply any "margin" to guard against sudden traffic fluctuations, a steep traffic increase might lead to large volumes of rejected traffic. This is confirmed by our experiments on Monday and Tuesday when traffic radically changes from nighttime to morning hours. In contrast, steep traffic increases can be handled more easily when the average cumulative traffic is lower: during the last three days, online gradient achieves delays close to the oracle with low or no constraint violations.

3) Problem-adapted AI exhibits the lowest average delay among the three techniques while keeping rejected traffic at very low levels. It achieves this by directly predicting the optimal BS loads instead of predicting the traffic and calculating the "optimal" loads while trying to guard against traffic prediction errors. This way it both averages out a large number of small prediction errors and learns to be only as conservative as needed in its decisions.

7.2.1.2 Which Method to Choose?

Although the simulations results lean in favor of problem-adapted AI, the answer is not trivial. Problem-specific AI seems to provide the best tradeoff between robustness, performance and scalability, while robust optimization requires significant overhead to calculate good predictions and optimal maps, especially with finer space granularity. However, the AI performance largely depends on the quality of training. Proper training requires good training data and a potentially huge amount of resources (i.e., memory, CPU) depending on the dimensionality of the problem, which are not always available. This is the reason why it is important to combine AI with modeling intuition in order to significantly reduce the problem state space. On the other hand, online learning algorithms are very simple and quick, but sometimes fail to protect the system from rapid traffic surges, this could although be circumvented by applying the constraint budget framework of chapters 5 and 6, allowing a parametrizable budget of violation similar to the robust method. Deciding which method to choose is therefore largely dependent on the application, the problem size and the available resources.

Appendices

Appendix A

Online Convex Optimization with Long-Term Budget Constraints

A.1 Proof of Proposition 1

In this section we prove that the COLD algorithm is a feasible policy with no regret against any K-benchmark with $K = o(T)$. We first prove an upper bound on the predictor queue $Q(t)$ at the beginning of round t. Then we use this to upper bound the constraint residual, which will, in turn be used to prove feasibility; picking x_t according to Eq.(6.9) will give sub-linear long-term constraint residual in a time horizon T (see A.1.1). After having proved feasibility, we compare our policy with the K-benchmark and prove, in A.1.1.1, that it achieves no regret for any $K = o(T)$. For proving these an intermediate technical result regarding the behavior of the sums of $Q(t)\hat{g}_t(x_t)$ over any window of K consecutive rounds for any sequence of actions $\{x_t\}$ is necessary; the statement and its proof are deferred to A.1.2 at the end of this section.

A.1.1 Proof of the bound on the Constraint Residual

In the first part of the proof, we will prove the bound (6.13) on the constraint residual. First, we consider the quadratic Lyapunov Drift (defined as $\Delta(t) \triangleq \frac{1}{2}Q(t+1)^2 - \frac{1}{2}Q(t)^2$); this measures the impact of our policy on the predictor queue. We bound the Lyapunov drift by the following lemma:

Lemma 1 (Drift Upper Bound)**.** *The predictor queue drift is upper bounded by:*

$$\Delta(t) \leq B + Q(t)\hat{g}_t(x_t), \ \forall t \in \{0, \ldots, T\}, \forall x \in \mathcal{X}. \tag{A.1}$$

where $B \triangleq \frac{(F+GD)^2}{2}$*, is a constant. Reminder:*

$$\hat{g}_t(x_t) \triangleq g_{t-1}(x_{t-1}) + \langle g'_{t-1}(x_{t-1}), (x_t - x_{t-1})\rangle.$$

Proof. Squaring the queue update function Eq.(6.10), since $\max\{0,x\}^2 \leq x^2$ we get:

$$\begin{aligned} Q(t+1)^2 &= \left([Q(t) + \hat{g}_t(x_t)]^+\right)^2 \\ &\leq (Q(t) + \hat{g}_t(x_t))^2 \\ &\leq Q(t)^2 + 2Q(t)\hat{g}_t(x_t) + \hat{g}_t(x_t)^2. \end{aligned}$$

To continue we bound $\hat{g}_t(x_t)$:

$$\begin{aligned} |\hat{g}_t(x_t)| &\leq |g_{t-1}(x_{t-1}) + \langle g'_{t-1}(x_{t-1}), (x_t - x_{t-1})\rangle| \\ &\leq |g_{t-1}(x_{t-1})| + |\langle g'_{t-1}(x_{t-1}), (x_t - x_{t-1})\rangle| \\ &\leq F + GD. \end{aligned}$$

Using the above we get that:

$$\begin{aligned} Q(t+1)^2 &\leq Q(t)^2 + 2Q(t)\hat{g}_t(x_t) + \hat{g}_t(x_t)^2 \\ &\leq Q(t)^2 + 2Q(t)\hat{g}_t(x_t) + (F+GD)^2. \end{aligned}$$

Rearranging the terms, dividing by 2 and using the definitions of B and $\Delta(t)$ completes the proof. □

Corollary 1. *Plugging Lem.*(1) *into DPLPS definition, Eq.*(6.12) *we get the resulting upper bound:*

$$\begin{aligned} \Delta(t) + V\hat{f}_{t-1}(x_{t-1}) + \alpha||x_t - x_{t-1}||_2^2 \leq B + \\ Q(t)\hat{g}_t(x_t) + V\hat{f}_{t-1}(x_{t-1}) + \alpha||x_t - x_{t-1}||_2^2. \end{aligned}$$

We define $r_t(x)$*:*

$$r_t(x) \triangleq B + Q(t)\hat{g}_t(x) + V\hat{f}_{t-1}(x_{t-1}) + \alpha||x - x_{t-1}||_2^2.$$

Hence, $DPLPS(x_t) \leq r_t(x_t)$*.*

Lemma 2. *Our policy* x_t *minimizes* $r_t(x)$

$$x_t \in \underset{x\in\mathcal{X}}{\operatorname{argmin}} \{r_t(x)\}.$$

Proof. First, observe that by definition our policy is:

$$x_t \triangleq \Pi_{\mathcal{X}}\left[x_{t-1} + \frac{H_t}{2\alpha}\right],$$

where H_t is the constant from past observations:

$$H_t \triangleq V f'_{t-1}(x_{t-1}) + Q(t) g'_{t-1}(x_{t-1}).$$

It suffices to show that this projection actually returns a minimizer of $r_t(x)$. We have that, $\langle H_t, x_t - x_{t-1}\rangle = Q(t)\langle g'_{t-1}(x_{t-1}), (x_t - x_{t-1})\rangle + V\langle f'_{t-1}(x_{t-1}), (x_t - x_{t-1})\rangle$.

$$\begin{aligned}
x_t &= \operatorname*{argmin}_{x\in\mathcal{X}} \{r_t(x)\} \\
&\overset{(a)}{=} \operatorname*{argmin}_{x\in\mathcal{X}} \{\langle H_t, x - x_{t-1}\rangle + \alpha||x - x_{t-1}||_2^2\} \\
&\overset{(b)}{=} \operatorname*{argmin}_{x\in\mathcal{X}} \left\{\langle H_t, x - x_{t-1}\rangle + \alpha||x - x_{t-1}||_2^2 + \frac{H_t^2}{4\alpha}\right\} \\
&= \operatorname*{argmin}_{x\in\mathcal{X}} \left|\left|\frac{H_t}{2\sqrt{\alpha}} + \sqrt{\alpha}(x - x_{t-1})\right|\right|_2^2 \\
&\overset{(c)}{=} \operatorname*{argmin}_{x\in\mathcal{X}} \left|\left|x - \left(x_{t-1} - \frac{H_t}{2\alpha}\right)\right|\right|_2^2 \\
&= \Pi_{\mathcal{X}}\left[x_{t-1} - \frac{H_t}{2\alpha}\right].
\end{aligned}$$

Here, in the (a) we discard the constant terms $Q(t)g_{t-1}(x_{t-1}) + Vf_{t-1}(x_{t-1})$ that do not depend on variable x. The (b) equality does not change the minimizer since we add a constant term that completes the square norm. Finally, (c) is the definition of euclidean projection on set $\mathcal{X}$, hence the last equality follows and completes the proof. □

Using the fact that our policy minimizes the DPLPS, by Lem.(2) and the bound on the drift by Lem.(1), we will prove that the predictor queue is bounded; this result is an intermediate step towards proving that the constraint residuals are bounded. The idea is to compare our policy's DPLPS, with the upper bound for a policy that satisfies the constraint at a rolling window of K rounds, namely the K-benchmark policy x_*^K. The properties of the K -benchmark policy will help us bound the performance of our policy, that minimizes DPLPS.

Lemma 3 (Queue Bound)**.** *For any $t \in \{0, 1, \ldots, T\}$, the queue is bounded as:*

$$Q(t+1) \leq \left\{2BKt + 4FVt + \frac{V^2G^2t}{\alpha} + 2\alpha D^2 + 2BK^2 + 2BK(t+1)\right\}^{\frac{1}{2}}$$

Proof. First, by Lem.(2), x_t is the minimizer of $r_t(x)$ at round t. Furthermore, $r_t(x)$ is a 2α-strongly convex function, hence $r_t(x_t) \leq r_t(y) - \alpha||y - x_t||_2^2$ for any $y \in \mathcal{X}$. We have that $DPLPS(x_t) \leq r_t(x_t) \leq r_t(y) - \alpha||y - x_t||_2^2$. We expand and re-arrange terms:

$$\begin{aligned}\Delta(t) \leq B + &\underbrace{Q(t)\hat{g}_t(y)}_{\text{(a)}} + V \underbrace{\langle f'_{t-1}(x_{t-1}), (y - x_{t-1})\rangle}_{\text{(b)}} \\ &\underbrace{-V\langle f'_{t-1}(x_{t-1}), (x_t - x_{t-1})\rangle - \alpha||x_t - x_{t-1}||_2^2}_{\text{(c)}} + \\ &\underbrace{\alpha||y - x_{t-1}||_2^2 - \alpha||y - x_t||_2^2}_{\text{(d)}}. \end{aligned} \tag{A.2}$$

To continue with the proof we need to work on the terms (a),(b),(c) and (d) appearing on the right hand side. Since $f_t(x)$ is convex, (b) term is upper bounded by:

$$\begin{aligned} f_{t-1}(y) &\geq f_{t-1}(x_{t-1}) + \langle f'_{t-1}(x_{t-1}), (y - x_{t-1})\rangle \\ \langle f'_{t-1}(x_{t-1}), (y - x_{t-1})\rangle &\leq f_{t-1}(y) - f_{t-1}(x_{t-1}) \\ \langle f'_{t-1}(x_{t-1}), (y - x_{t-1})\rangle &\leq 2F. \end{aligned}$$

For (c) term we prove a technical Lemma:

Lemma 4. *[40]*

$$-\langle Vf'_{t-1}(x_{t-1}), (x_t - x_{t-1})\rangle - \alpha||x_t - x_{t-1}||_2^2 \leq \frac{V^2G^2}{2\alpha}.$$

Proof. Since $||\alpha||_2^2 + ||\beta||_2^2 - 2\langle\alpha, \beta\rangle = ||\alpha + \beta||_2^2$:

$$\begin{aligned} &-\langle Vf'_{t-1}(x_{t-1}), (x_t - x_{t-1})\rangle - \alpha||x_t - x_{t-1}||_2^2 \leq \\ &-\left|\left|\frac{Vf'_{t-1}(x_{t-1})}{\sqrt{2\alpha}} + \frac{\sqrt{\alpha}(x_t - x_{t-1})}{\sqrt{2}}\right|\right|_2^2 + \frac{V||f'_{t-1}(x_{t-1})||_2^2}{2\alpha} \\ &\leq \frac{V^2G^2}{2\alpha}. \end{aligned}$$

□

Now we pick $y = x_\star^K$, according to Def.(2), and we sum the inequality for K consecutive rounds.

$$\begin{aligned}\sum_{\tau=0}^{K-1} \Delta(t+\tau) \leq BK + &\underbrace{\sum_{\tau=0}^{K-1} Q(t+\tau)\hat{g}_{t+\tau}(y)}_{\text{(a)}} + 2FVK + \\ \frac{V^2G^2K}{2\alpha} + &\underbrace{\alpha\sum_{\tau=0}^{K-1}\left\{||y - x_{t+\tau-1}||_2^2 - ||y - x_{t+\tau}||_2^2\right\}}_{\text{(d)}}. \end{aligned}$$

For the term (a) we use Cor.(2) (see A.1.2 at the end of this subsection for a proof),

$$\sum_{\tau=0}^{K-1} Q(t+\tau)\hat{g}_{t+\tau}(y) \leq BK(K-1).$$

We get to:

$$\sum_{\tau=0}^{K-1} \Delta(t+\tau) \leq BK^2 + 2FVK + \frac{V^2G^2K}{2\alpha} + (d).$$

We take the telescopic sum for $t = \{1, \dots, T-K\}$:

$$\begin{aligned}\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} \Delta(t+\tau) \leq BK^2(T-K) + 2FVK(T-K)+ \\ \frac{V^2G^2K(T-K)}{2\alpha} + \sum_{t=1}^{T-K}(d) \\ \leq BK^2T + 2FVKT + \frac{V^2G^2KT}{2\alpha} + \sum_{t=1}^{T-K}(d).\end{aligned}$$

For the term (d) we have that:

$$\begin{aligned}\alpha \sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} \left(||y - x_{t+\tau-1}||_2^2 - ||y - x_{t+\tau}||_2^2\right) = \\ \alpha \sum_{\tau=0}^{K-1} \left(||y - x_{\tau}||_2^2 - ||y - x_{T-K+\tau}||_2^2\right) \leq \\ \alpha \sum_{\tau=0}^{K-1} ||y - x_{\tau}||_2^2 \leq \alpha K D^2.\end{aligned} \tag{A.3}$$

The intermediate drift terms in the telescopic sum are canceled out, hence:

$$\begin{aligned}\underbrace{\sum_{\tau=0}^{K-1} \frac{Q(T-K+\tau+1)^2}{2}}_{\text{e}} - \underbrace{\sum_{\tau=0}^{K-1} \frac{Q(\tau+1)^2}{2}}_{\text{f}} \leq \\ BK^2T + 2FVKT + \frac{V^2G^2KT}{2\alpha} + \alpha K D^2.\end{aligned}$$

Since $\hat{g}_t(x_t) \leq \sqrt{2B}$, the following holds for all t:

$$\begin{aligned} Q(t+1) &= [Q(t) + b_t(x_t)]^+ \leq |Q(t) + \hat{g}_t(x_t)| \\ &\leq |Q(t)| + |\hat{g}_t(x_t)| \leq Q(0) + \sum_{\tau=0}^{T} |\hat{g}_t(x_t)| \\ &\leq \sum_{\tau=0}^{t} \sqrt{2B} \leq (t+1)\sqrt{2B} \end{aligned} \tag{A.4}$$

We use Eq.(A.4) to upperbound (f):

$$(f) \leq \frac{1}{2} \sum_{\tau=0}^{K-1} ((\tau+1)\sqrt{2B})^2 \leq B\frac{K(K+1)(2K+1)}{6},$$

giving us the new bound:

$$\begin{aligned} \underbrace{\sum_{\tau=0}^{K-1} \frac{Q(T-K+\tau+1)^2}{2}}_{\text{(e)}} &\leq BK^2T + 2FVKT + \\ &\frac{V^2G^2KT}{2\alpha} + \alpha KD^2 + B\frac{K(K+1)(2K+1)}{6}. \end{aligned} \tag{A.5}$$

To continue we need to come up with a lower bound for term (e) based on $Q(T+1)$. We will use Eq.(A.13):

$$Q(T-K+\tau+1) \geq Q(T+1) - \sum_{n=T-K+\tau+1}^{T} |\hat{g}_n(x_n)| \overset{(i)}{\geq} 0. \tag{A.6}$$

here (i) is true if $Q(T+1) \geq K\sqrt{2B} \geq (\tau+1)\sqrt{2B}$[1]. We take the square of each side of Eq.(A.6):

$$\begin{aligned} Q(T-K+\tau+1)^2 &\geq \left(Q(T+1) - \sum_{n=T-K+\tau+1}^{T} |\hat{g}_n(x_n)| \right)^2 \\ &\geq Q(T+1)^2 - 2Q(T+1) \sum_{n=T-K+\tau+1}^{T} |\hat{g}_n(x_n)|. \end{aligned}$$

[1]We will later prove our bound holds even if this assumption is not true.

Now, we sum the resulting inequality for K consecutive slots, arriving at:

$$\begin{aligned}
&\sum_{\tau=0}^{K-1} Q(T-K+\tau+1)^2 \geq \\
&\quad \geq KQ(T+1)^2 - \sum_{\tau=0}^{K-1} 2Q(T+1) \sum_{n=T-K+\tau+1}^{T} |\hat{g}_n(x_n)| \\
&\quad \overset{(i)}{\geq} KQ(T+1)^2 - \sum_{\tau=0}^{K-1} 2(T+1)\sqrt{2B} \sum_{n=T-K+\tau+1}^{T} \sqrt{2B} \\
&\quad \geq KQ(T+1)^2 - 4(T+1)B \sum_{\tau=0}^{K-1} (K-\tau-1) \\
&\quad \geq KQ(T+1)^2 - 4(T+1)B\frac{K(K-1)}{2},
\end{aligned}$$

where (i) follows from Eq.(A.4) and $\hat{g}_t(x_t) \leq \sqrt{2B}$. We rearrange and use Eq.(A.5):

$$\begin{aligned}
&KQ(T+1)^2 \leq \\
&\quad \leq \sum_{\tau=0}^{K-1} Q(T-K+\tau+1)^2 + 4(T+1)B\frac{K(K-1)}{2} \\
&\quad \overset{(i)}{\leq} 2BK^2T + 4FVKT + \frac{V^2G^2KT}{\alpha} + 2\alpha KD^2 + \\
&\qquad B\frac{K(K+1)(2K+1)}{3} + 2(T+1)BK^2,
\end{aligned}$$

where (i) is true due to Eq.(A.5). Dividing by K and taking the square root, we get

$$\begin{aligned}
Q(T+1) &\leq \Bigg\{ 2BKT + 4FVT + \frac{V^2G^2T}{\alpha} + 2\alpha D^2 + \\
&\quad + B\frac{(K+1)(2K+1)}{3} + 2(T+1)B(K+1) \Bigg\}^{\frac{1}{2}} \\
&\overset{(i)}{\leq} \Bigg\{ 2BKT + 4FVT + \frac{V^2G^2T}{\alpha} + 2\alpha D^2 + \\
&\quad 2BK^2 + 2(T+1)BK \Bigg\}^{\frac{1}{2}}, \qquad \text{(A.7)}
\end{aligned}$$

where in (i) $B\frac{(K+1)(K+2)}{3} \leq 2BK^2$, for $K \geq 1$. We note here that we proved Eq.(A.7) for $Q(T+1) \geq K\sqrt{2B}$, for $Q(T+1) < K\sqrt{2B}$ obviously Eq.(A.7) is an upper bound. This result can be easily generalized for any $T' \in \{K, K+1, \ldots, T\}$. For $T' \in \{1, \ldots, K-1\}$, we have from Eq.(A.4) that $Q(T') \leq T'\sqrt{2B} \leq K\sqrt{2B}$, which is again upper bounded by Eq.(A.7). This finishes the proof. $\square$

We may now use the queue bound from Lem.(3) to prove that our policy guarantees sublinear growth of constraint residual.

Lemma 5 (Constraint Residual Bound)**.** *The Constraint Residual is bounded as:*

$$\sum_{t=1}^{T} g_t(x_t) \leq \Bigg\{ 2BKT + 4FVT + \frac{V^2G^2T}{\alpha} + 2\alpha D^2 + 2BK^2 + 2(T+1)BK \Bigg\}^{\frac{1}{2}} + \frac{GVT}{2\alpha} + \frac{G^2\left[\sqrt{2BK} + \sqrt{4FV} + \sqrt{\frac{V^2G^2}{\alpha}} + \sqrt{2BK}\right]\frac{T^{\frac{3}{2}}+T}{\sqrt{2}}}{2\alpha} + \frac{G^2\left[\sqrt{2\alpha D^2} + \sqrt{2BK^2} + \sqrt{2BK}\right]T}{2\alpha}.$$

Proof. We start with the queue update from Eq.(6.10):

$$\begin{aligned} Q(t+1) = [Q(t) + \hat{g}_t(x_t)]^+ &\geq Q(t) + \hat{g}_t(x_t) \\ &\geq Q(t) + g_{t-1}(x_{t-1}) + \langle g'_{t-1}(x_{t-1}), (x_t - x_{t-1})\rangle. \end{aligned}$$

Re-arranging terms in the above, we have:

$$\begin{aligned} g_{t-1}(x_{t-1}) &\leq Q(t+1) - Q(t) - \langle g'_{t-1}(x_{t-1}), (x_t - x_{t-1})\rangle \\ &\overset{(a)}{\leq} Q(t+1) - Q(t) + ||g'_{t-1}(x_{t-1})||_2 ||(x_t - x_{t-1})||_2., \end{aligned} \tag{A.8}$$

where in (a) we use the Cauchy-Schwarz inequality. Let us now denote:

$$y \triangleq x_{t-1} - \frac{1}{2\alpha}\left(Vf'_{t-1}(x_{t-1}) + Q(t)g'_{t-1}(x_{t-1})\right),$$

as in Eq.(6.9). We have:

$$\begin{aligned} ||x_t - x_{t-1}||_2 = ||\Pi_{\mathcal{X}}(y) - x_{t-1}||_2 &\overset{(a)}{\leq} ||y - x_{t-1}||_2 \\ &\overset{(b)}{\leq} ||y - x_{t-1}||_1, \end{aligned}$$

where (a) is true due to the non-expansiveness of euclidean projection on a convex set, while for (b) we used the norm inequality. Combing this finding, Eq.(A.8) and the fact that $||g'_t(x)||_2 \leq G$ we get:

$$g_{t-1}(x_{t-1}) \leq Q(t+1) - Q(t) + G||y - x_{t-1}||_1.$$

We now expand y as

$$\begin{aligned} g_{t-1}(x_{t-1}) \leq Q(t+1) - Q(t) + \\ G\frac{||Vf'_{t-1}(x_{t-1}) + Q(t)g'_{t-1}(x_{t-1})||_1}{2\alpha} \\ \leq Q(t+1) - Q(t) + \frac{VG^2}{2\alpha} + \frac{G^2Q(t)}{2\alpha}. \end{aligned}$$

We take the sum for T rounds and we drop $Q(1) = 0$:

$$\sum_{t=1}^{T} g_{t-1}(x_{t-1}) \leq Q(T+1) + \frac{G^2VT}{2\alpha} + \frac{G^2\sum_{t=1}^{T}Q(t)}{2\alpha}. \tag{A.9}$$

We now use the queue bound (Lem.(3)), from which it follows that

$$\begin{aligned} \sum_{t=1}^{T} Q(t) \leq \sum_{t=1}^{T} \Bigg\{ 2BKt + 4FVt + \frac{V^2G^2t}{\alpha} + \\ 2\alpha D^2 + 2BK^2 + 2(t+1)BK \Bigg\}^{\frac{1}{2}} \\ \leq \sum_{t=1}^{T} \Bigg[\sqrt{2BKt} + \sqrt{4FVt} + \sqrt{\frac{V^2G^2t}{\alpha}} + \\ \sqrt{2\alpha D^2} + \sqrt{2BK^2} + \sqrt{2BK(T+1)} \Bigg] \\ \leq \left[\sqrt{2BK} + \sqrt{4FV} + \sqrt{\frac{V^2G^2}{\alpha}} + \sqrt{2BK} \right] \sum_{t=1}^{T} \sqrt{t} + \\ \left[\sqrt{2\alpha D^2} + \sqrt{2BK^2} + \sqrt{2BK} \right] T. \end{aligned}$$

To proceed futher, we use the norm inequality (Cauchy-Schwarz) for the vector $[\sqrt{1}, \sqrt{2}, \sqrt{3}, ..., \sqrt{t}, ..., \sqrt{T}]$, from which we get

$$\sqrt{\sum_{t=1}^{T} t} \leq \sum_{t=1}^{T} \sqrt{t} \leq \sqrt{T\sum_{t=1}^{T} t} \leq \frac{T^{\frac{3}{2}} + T}{\sqrt{2}},$$

hence:

$$\begin{aligned} \sum_{t=1}^{T} Q(t) \leq \left[\sqrt{2BK} + \sqrt{4FV} + \sqrt{\frac{V^2G^2}{\alpha}} + \sqrt{2BK} \right] \times \\ \frac{T^{\frac{3}{2}} + T}{\sqrt{2}} + \left[\sqrt{2\alpha D^2} + \sqrt{2BK^2} + \sqrt{2BK} \right] T. \end{aligned}$$

By using the bounds for $Q(t)$ and $\sum_t Q(t)$ in inequality Eq.(A.9) we complete the proof.

$\square$

A.1.1.1 Proof of the Regret Bound

In the second part of the proof, we will prove the regret bound (6.14) against any K-benchmark. Since our policy x_t minimizes the DPLPS by Lem.(2) and $x_\star^K$ by Def.(2) has has minimum cost subject to no constraint residuals over any window of K rounds, comparing $x_\star^K$ with our policy will produce the result.

Lemma 6 (Regret Bound)**.** *The upper bound of the regret of our policy against an $x_\star^K$ (Eq.(2)) benchmark is:*

$$\sum_{t=0}^{T-1} f_t(x_t) - \sum_{t=0}^{T-1} f_t(y) \leq \frac{BKT}{V} + \frac{VG^2T}{2\alpha} + B\frac{(K+1)(2K+1)}{6V} + \frac{\alpha D^2}{V} + 2F(K-1).$$

Proof. We begin by Eq.(A.2), where we add on both sides the term $Vf_{t-1}(x_{t-1})$, with $y = x_\star^K$. We remind that:

$$\hat{f}_t(x_t) \triangleq f_{t-1}(x_{t-1}) + f'_{t-1}(x_{t-1})(x_t - x_{t-1}).$$

Using Lem.(4) on Eq.(A.2)(c) and summing the resulting inequality for $K = \{t, \ldots, t+K-1\}$ consequent rounds, we get:

$$\sum_{\tau=0}^{K-1} \Delta(t+\tau) + \sum_{\tau=0}^{K-1} Vf_{t+\tau-1}(x_{t+\tau-1}) \leq BK + \underbrace{\sum_{\tau=0}^{K-1} Q(t+\tau)\hat{g}_{t+\tau}(y)}_{(a)} + V\sum_{\tau=0}^{K-1} \underbrace{\hat{f}_{t+\tau}(y)}_{(b)} + \frac{V^2G^2K}{2\alpha} + \alpha\sum_{\tau=0}^{K-1} ||y - x_{t+\tau-1}||_2^2 - \alpha\sum_{\tau=0}^{K-1} ||y - x_{t+\tau}||_2^2. \tag{A.10}$$

We now use Lem.7 (see A.1.2 at the end of this subsection for a proof) in order to bound term (a) in a way so we can use the sample path property of the static policy $x_\star^K$, and convexity of $f_t(x)$, hence $\hat{f}_t(x) \leq f_{t-1}(x)$, in order to bound term (b).

Taking Eq.(A.10) and using the Cor.(2) for term (a) and convexity of

$f_t(x)$ for term (b), we get the following:

$$\sum_{\tau=0}^{K-1} \Delta(t+\tau) + V\sum_{\tau=0}^{K-1} f_{t+\tau-1}(x_{t+\tau-1}) \leq BK+$$
$$\underbrace{BK(K-1)}_{(a)} + V\sum_{\tau=0}^{K-1} \underbrace{f_{t+\tau-1}(y)}_{(b)} + \frac{V^2G^2K}{2\alpha}+$$
$$\alpha \sum_{\tau=0}^{K-1} \left[||y-x_{t+\tau-1}||_2^2 - ||y-x_{t+\tau}||_2^2\right].$$

We sum the inequality from $t=\{1,\dots,T-K\}$:

$$\underbrace{\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} \Delta(t+\tau)}_{(d)} + V\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} f_{t+\tau-1}(x_{t+\tau-1}) \overset{(i)}{\leq}$$
$$BK^2T + V\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} f_{t+\tau-1}(y) + \frac{V^2G^2KT}{2\alpha}+$$
$$\underbrace{\alpha\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} ||y-x_{t+\tau-1}||_2^2 - \alpha\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} ||y-x_{t+\tau}||_2^2}_{(e)}. \tag{A.11}$$

where in (i), we use where needed that $T > T-K$. To continue with the proof we need to lower bound the negative terms of the drift expression (d):

$$\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} \Delta(t+\tau) =$$
$$\frac{1}{2}\sum_{\tau=0}^{K-1} Q(T-K+\tau+1)^2 - \frac{1}{2}\sum_{\tau=0}^{K-1} Q(\tau+1)^2$$
$$\geq -\frac{1}{2}\sum_{\tau=0}^{K-1} Q(\tau+1)^2 \overset{Eq.(A.4)}{\geq} -\frac{1}{2}\sum_{\tau=0}^{K-1}((\tau+1)\sqrt{2B})^2$$
$$\geq -B\frac{K(K+1)(2K+1)}{6}.$$

Furthermore, term (e) is upper bounded by $(e) \leq \alpha KD^2$, as shown by Eq.(A.3). Realigning Eq.(A.11) with the new bounds we get:

$$V\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} f_{t+\tau-1}(x_{t+\tau-1}) - V\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K} f_{t+\tau-1}(y) \leq$$
$$BK^2T + \frac{V^2G^2KT}{2\alpha} + B\frac{K(K+1)(2K+1)}{6} + \alpha KD^2.$$

Here, this expression already compares the cost of our policy against the $x_\star^K$ policy. To continue with the proof we add and subtract terms in order to make Regret $(R_K(T) = \sum_{t=0}^{T-1}[f_t(x_t) - f_t(y)])$ appear K times:

$$\begin{aligned}
&\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K}(f_{t+\tau-1}(x_{t+\tau-1}) - f_{t+\tau-1}(y)) = \\
&\sum_{\tau=0}^{K-1}\sum_{t=1}^{T-K}(f_{t+\tau-1}(x_{t+\tau-1}) - f_{t+\tau-1}(y)) + \\
&\sum_{\tau=0}^{K-1}\sum_{t=0}^{\tau-1}(f_t(x_t) - f_t(y)) - \sum_{\tau=0}^{K-1}\sum_{t=0}^{\tau-1}(f_t(x_t) - f_t(y)) + \\
&\sum_{\tau=0}^{K-1}\sum_{t=T-K+\tau+1}^{T-1}\left[f_t(x_t) - f_t(y) - f_t(x_t) - f_t(y)\right] = \\
&K\sum_{t=1}^{T}(f_{t-1}(x_{t-1}) - f_{t-1}(y)) - \\
&\sum_{\tau=0}^{K-1}\sum_{t=0}^{\tau-1}(f_t(x_t) - f_t(y)) - \sum_{\tau=0}^{K-1}\sum_{t=T-K+\tau+1}^{T-1}(f_t(x_t) - f_t(y))
\end{aligned}$$

Then, we upper bound the red and blue terms:

$$\sum_{\tau=0}^{K-1}\sum_{t=0}^{\tau-1}(f_t(x_t) - f_t(y)) - \sum_{\tau=0}^{K-1}\sum_{t=T-K+\tau+1}^{T-1}(f_t(x_t) - f_t(y)) \leq 2FK(K-1)$$

By dividing both sides with V, K and using the above inequalities we can complete the proof:

$$\sum_{t=0}^{T-1} f_t(x_t) - \sum_{t=0}^{T-1} f_t(y) \leq \frac{BKT}{V} + \frac{VG^2T}{2\alpha} + B\frac{(K+1)(2K+1)}{6V} + \frac{\alpha D^2}{V} + 2F(K-1).$$

□

A.1.2 Upper Bound on $\sum_{\tau=0}^{K-1} Q(t+\tau)\hat{g}_{t+\tau}(x_{t+\tau})$

Finally, we prove an important technical result regarding upper bounding the quantity $\sum_{\tau=0}^{K-1} Q(t+\tau)\hat{g}_{t+\tau}(x_{t+\tau})$ for any sequence of actions x_t, and then specifically for the action selected by the K-benchmark.

Lemma 7. *For any sequence of actions $\{x_t\}$:*

$$\sum_{\tau=0}^{K-1} Q(t+\tau)\hat{g}_{t+\tau}(x_{t+\tau}) \leq \\ Q(t)\sum_{\tau=0}^{K-1} \hat{g}_{t+\tau}(x_{t+\tau}) + BK(K-1).$$

Proof. We start by the queue update equation Eq.(6.10) and give a lower bound for $Q(t+K)$ for any $K \geq 1$:

$$\begin{aligned} Q(t+K) &= [Q(t+K-1) + \hat{g}_{t+K-1}(x_{t+K-1})]^+ \\ &\geq Q(t+K-1) + \hat{g}_{t+K-1}(x_{t+K-1}) \\ &\geq Q(t) + \sum_{\tau=0}^{K-1} \hat{g}_{t+\tau}(x_{t+\tau}). \end{aligned} \tag{A.12}$$

and an upper bound:

$$\begin{aligned} Q(t+K) &= [Q(t+K-1) + \hat{g}_{t+K-1}(x_{t+K-1})]^+ \\ &\leq Q(t+K-1) + |\hat{g}_{t+K-1}(x_{t+K-1})| \\ &\leq Q(t) + \sum_{\tau=0}^{K-1} |\hat{g}_{t+\tau}(x_{t+\tau})|. \end{aligned} \tag{A.13}$$

Next, we use these bounds so that $Q(t)$ appears as a common term in the sum. We denote $\max\{0, f(\cdot)\} \triangleq [f(\cdot)]^+$ and $\min\{0, f(\cdot)\} \triangleq [f(\cdot)]^-$:

$$\begin{aligned}
&\sum_{\tau=0}^{K-1} Q(t+\tau)\hat{g}_{t+\tau}(x_{t+\tau}) = \\
&\sum_{\tau=0}^{K-1} Q(t+\tau)\left\{[\hat{g}_{t+\tau}(x_{t+\tau})]^+ + [\hat{g}_{t+\tau}(x_{t+\tau})]^-\right\} \overset{(a)}{\leq} \\
&\sum_{\tau=0}^{K-1}\left\{Q(t) + \sum_{i=0}^{\tau-1}|\hat{g}_{t+i}(x_{t+i})|\right\}[\hat{g}_{t+\tau}(x_{t+\tau})]^+ + \\
&\quad \sum_{\tau=0}^{K-1}\left\{Q(t) + \sum_{i=0}^{\tau-1}\hat{g}_{t+i}(x_{t+i})\right\}[\hat{g}_{t+\tau}(x_{t+\tau})]^- \overset{(b)}{\leq} \\
&Q(t)\sum_{\tau=0}^{K-1}\{[\hat{g}_{t+\tau}(x_{t+\tau})]^+ + [\hat{g}_{t+\tau}(x_{t+\tau})]^-\} + \\
&\quad \sum_{\tau=0}^{K-1}\left\{\sum_{i=0}^{\tau-1}|\hat{g}_{t+i}(x_{t+i})|\right\}[\hat{g}_{t+\tau}(x_{t+\tau})]^+ + \\
&\quad \sum_{\tau=0}^{K-1}\left\{\sum_{i=0}^{\tau-1}\hat{g}_{t+i}(x_{t+i})\right\}[\hat{g}_{t+\tau}(x_{t+\tau})]^- \overset{(c)}{\leq} \\
&Q(t)\sum_{\tau=0}^{K-1}\hat{g}_{t+\tau}(x_{t+\tau}) + \\
&\quad \sum_{\tau=0}^{K-1}\left\{\sum_{i=0}^{\tau-1}|\hat{g}_{t+i}(x_{t+i})|\right\}|\hat{g}_{t+\tau}(x_{t+\tau})| \overset{(d)}{\leq} \\
&Q(t)\sum_{\tau=0}^{K-1}\hat{g}_{t+\tau}(x_{t+\tau}) + 2B\sum_{\tau=0}^{K-1}\sum_{i=0}^{\tau-1}1 \leq \\
&Q(t)\sum_{\tau=0}^{K-1}\hat{g}_{t+\tau}(x_{t+\tau}) + BK(K-1).
\end{aligned}$$

For (a) we take the upper bound for queue on the positive terms (Eq.(A.13)) and the lower bound on the queue for the negative terms (Eq.(A.12)), this gives an upper bound on the total. In (b) we rewrite the equation by bringing in the front the common $Q(t)$ terms. Next, at (c) we upper bound the non-$Q(t)$ terms with their norm and by passing the norm to every element the inequality follows. Finally, (d) follows by taking the upper bound $|\hat{g}_t(x)| \leq \sqrt{2B}$. □

Corollary 2. *For policy $y = x_\star^K$:*

$$\sum_{\tau=0}^{K-1} Q(t+\tau)\hat{g}_{t+\tau}(y) \leq BK(K-1).$$

Proof. Usingthe above Lem.(7), replacing $x_t + \tau$ for all τ with y, since y has the property that $\sum_{\tau=0}^{K-1} g_{t+\tau-1}(y) \leq 0,\ \forall t$ and $\hat{g}_t(x) \leq g_{t-1}(x)$ due to convexity, the proof follows. □

A.2 Proof of Theorem 1

Keeping in mind that K is a free variable, we restrict our analysis to choices of α, V such that both $Ctr(T), R_K(T)$ are $o(T)$, and the bounds are optimized. These restrictions allow us to drop some terms in (6.13)-(6.14) which are of smaller order than the rest terms.

First, we note that $1 \leq K \leq T$. Hence, we may eliminate from (6.13) the following dominated terms: $2BK^2 = \mathcal{O}(2BKT)$, $\sqrt{K^2/T} = \mathcal{O}(\sqrt{K})$.

Second, we observe that $K < V < T$. This is because: if $V \leq K$, then the term BKT/V will become $\Omega(T)$, and if $V \geq T$, then the term $\sqrt{4FVT}$ will become $\Omega(T)$. Accordingly, we have the following dominated terms: $2BKT = \mathcal{O}(4FVT)$, $\sqrt{K} = \mathcal{O}(\sqrt{4FV})$, $B(K-1)(2K-1)/(6V) = \mathcal{O}(2F(K-1))$, $2F(K-1) = \mathcal{O}(BKT/V)$.

Dropping constants, and applying the above relations, we have arrived at the following simplified bounds:

$$Ctr(T) = \mathcal{O}\left(\frac{VT}{\alpha} + \frac{T}{\sqrt{\alpha}} + \sqrt{VT + \alpha + \frac{V^2T}{\alpha} +} \right. \qquad \text{(A.14)}$$
$$\left. + \frac{T^{\frac{3}{2}}}{\alpha}\left(\sqrt{V} + \frac{V}{\sqrt{\alpha}}\right)\right),$$

$$R_K(T) = \mathcal{O}\left(\frac{KT}{V} + \frac{VT}{\alpha} + \frac{\alpha}{V}\right). \qquad \text{(A.15)}$$

Next, we work to simplify (A.14). From the terms $\frac{VT}{\alpha}, \frac{\alpha}{V}$ in the regret expression, we may infer that: $V = o(\alpha)$, $\alpha = o(VT)$.

Hence, we have the following dominated terms: $V\frac{T}{\alpha} = \mathcal{O}(\sqrt{VT}\frac{T}{\alpha})$, $\frac{T}{\sqrt{\alpha}} = \mathcal{O}(\frac{\sqrt{VT}}{\sqrt{\alpha}}\frac{T}{\sqrt{\alpha}})$, $\alpha = \mathcal{O}(\sqrt{VT})$, $\frac{V\sqrt{T}}{\sqrt{\alpha}} = \sqrt{VT}\frac{\sqrt{V}}{\sqrt{\alpha}} = \mathcal{O}(\sqrt{VT})$, $\frac{V}{\sqrt{a}} = \mathcal{O}(\sqrt{V})$. Therefore (A.14) simplifies to:

$$Ctr(T) = \mathcal{O}\left(\sqrt{VT} + \sqrt{VT}\frac{T}{\alpha}\right).$$

Now comparing the above with (A.15), we observe that the only term that benefits from setting $\alpha < T$ is the term $\frac{\alpha}{V}$, and all other terms improve when

α increases. However, for $\alpha < T$ the term $\frac{\alpha}{V}$ is dominated by $\frac{KT}{V}$. Hence, we may restrict $\alpha \geq T$ with no loss of optimality. In the sequel we further restrict $\alpha \geq T$. This brings us to the following simplified bounds:

$$\begin{aligned} Ctr(T) &= \mathcal{O}(\sqrt{VT}), \\ R_K(T) &= \mathcal{O}\left(\frac{KT}{V} + \frac{VT}{\alpha} + \frac{\alpha}{V}\right). \end{aligned}$$

These bounds contain all cases where both constraint residuals and regret are sublinear to T. Next, consider two cases:

Case 1: $K < V < \sqrt{T}$. This case exists only when $K < \sqrt{T}$. In this case, we obtain no benefit by increasing α beyond T (since the benefiting term VT/α is already less than $\sqrt{T}$), hence the best choice is $\alpha = T$. Then, notice that $\frac{T}{V} = \mathcal{O}(\frac{KT}{V})$, and $V < \sqrt{T} = \mathcal{O}(\frac{KT}{V})$. Thus we get:

$$Ctr(T) = \mathcal{O}(\sqrt{VT}), \quad R_K(T) = \mathcal{O}\left(\frac{KT}{V}\right).$$

Case 2: $\sqrt{T} \leq V < T$. In this case, there are admissible values for α to make the two terms (containing α) equal, hence we select $\alpha = V\sqrt{T}$. This yields:

$$Ctr(T) = \mathcal{O}(\sqrt{VT}), \quad R_K(T) = \mathcal{O}\left(\frac{KT}{V} + \sqrt{T}\right).$$

Last, we observe that in the first case, we also have $\frac{KT}{V} = \Omega(\sqrt{T})$ due to the restricted values of V, hence we can express both cases with the expression:

$$Ctr(T) = \mathcal{O}(\sqrt{VT}), \quad R_K(T) = \mathcal{O}\left(\frac{KT}{V} + \sqrt{T}\right).$$

This concludes the proof.

Chapter 8

Résumé [Français]

8.1 Introduction

Récemment, nous avons assisté à une explosion de la disponibilité des données. Actuellement, des quantités massives de données sont régulièrement collectées dans toutes les entreprises, y compris les opérateurs de réseaux sans fil, les plates-formes de cloud computing (en français l'informatique en nuage ou nuagique ou encore l'infonuagique), et les détaillants. Cela a provoqué une tendance dans les problèmes de réseau; exploiter la puissance des données disponibles pour extraire des modèles et traiter les incertitudes. Le but de cette thèse est d'augmenter l'arsenal d'algorithmes afin de résoudre des problèmes de réseau, en proposant des cadres algorithmiques pour des réseaux sans fil, basés à la fois sur l'optimisation classique ou pilotée des données et sur l'apprentissage automatique.

Dans les réseaux sans fil, l'application de l'optimisation gagne du terrain, à la fois dans la pratique et dans la recherche, dans la mesure où il est envisagé d'améliorer considérablement l'efficacité de l'utilisation des ressources. Une application rigoureuse des techniques d'optimisation sur les ressources existantes peut réduire le coût d'exploitation. En réduisant par exemple la puissance, en augmentant les performances (maximiser le débit, en équilibrant la charge ou en minimisant les délais [32]), la qualité de service peut s'améliorer en offrant des garanties de performance (Quality of Service). À tour de rôle, différents outils d'optimisation peuvent être utilisés pour la future planification du réseau; l'expansion, le partage de ressources entre opérateurs et l'installation ou la réservation proactive de ressources.

En outre, l'intégration de l'optimisation dans des systèmes futurs est rendue possible par les avancées technologiques récentes, qui peuvent être utilisées pour renouveler l'architecture de réseau. Les principaux facteurs que nous identifions sont les suivants: i) la Softwarisation des services de réseau - officiellement désignée comme Network Function Virtualization (NFV) et Software Defined Networking (SDN) et la centralisation des ressources de calcul pour les réseaux – l'architecture Cloud-Radio Access Network (C-

RAN). Le groupement des deux permet une gestion centralisée des ressources (orchestration) et fournit une connaissance collective et une puissance de calcul, capables de résoudre les principaux problèmes d'optimisation. ii) Collecte massive de données et leurs techniques d'exploration. Les données collectées par le réseau de manière centralisée peuvent nous donner des informations sur le système et sur le comportement de l'utilisateur et être utilisées par des techniques d'apprentissage automatique afin d'améliorer l'efficacité du réseau.

Malgré l'évolution récente de la conception et de l'orchestration du réseau, le provisionnement de ressources dans les réseaux sans fil demeure un défi. Ceci peut être expliqué par les facteurs suivants: i) grande variation spatiotemporelle de la demande - l'environnement sans fil évolue rapidement suivant l'activité humaine, ii) grande dimension de l'optimisation - les problèmes rencontrés sont à grande échelle avec des millions de variables et (iii) le couplage des décisions - le réseau doit être reconfiguré en fonction de l'évolution de la demande dans le but de conserver son efficacité. Le défi consiste à proposer des méthodes d'optimisation rapides, évolutives en s'adaptant rapidement aux modifications apportées par le signal d'entrée; tout en assurant une résistance contre les fluctuations afin de répondre aux exigences du service.

Les propositions de cette thèse sont séparées en deux grandes parties. La première partie correspond aux chapitres 3 et 4, où nous nous concentrons sur l'optimisation de l'utilisation des ressources des réseaux en temps réel. Dans les chapitres 6 et 7 qui suivent, nous analyserons les réservations de ressources proactives. Tandis que le chapitre 5, sert de pont entre les deux parties, car il traite un problème en ligne avec une solution hors ligne basée sur les données. Dans cette étude, nous présentons deux cas d'utilisation principaux: l'association d'utilisateurs et la réservation de ressources. Les solutions présentées peuvent être également appliquées à une multitude de problèmes, à la fois hors du champ de la mise en réseau (vous trouverez les principaux exemples dans les chapitres 5 et 7).

En somme, les chapitres sont organisés de la manière suivante:

Chapitre 2 – Association d'Utilisateurs Distribuée avec des Garanties de Qualité de Service.

Chapitre 3 – Association d'Utilisateurs Centralisée et Scalable Basée sur le Transport Optimal Computationnel.

Chapitre 4 – Association d'Utilisateurs Data-Driven, Basée sur l'Optimisation Robuste.

Chapitre 5 – Réservation en Ligne de Ressources en Nuages avec des Contraintes Budgétaires.

Chapitre 6 – Réservation Générale de Ressources en Nuages en Ligne avec Contraintes Budgétaires: Un Nouveau Cadre.

Dans un premier temps, nous présenterons brièvement le contenu de chaque chapitre ainsi que les résultats les plus importants de ceci.

8.2 Association d'Utilisateurs Distribuée avec des Garanties de Qualité de Service

On étudie le problème d'association distribuée d'utilisateurs dans des Réseaux Ultra Dense (RUD) pour deux services réseau; l'un requérant des garanties QdS pour les flux VIP, et le service meilleur effort. Le but est de tirer avantage du multiplexage statistique pour optimiser l'utilisation des ressources, tout en s'assurant que les flux VIP disposent de garanties de performance active.

Regardons Fig.8.1 comme un example servant de motivation. Une méthode d'association de base qui ne prend pas en compte la différentiation des flux (pas de priorités ni de contraintes VIP) aboutira a de fortes charges pour les flux VIP Fig.8.1a pour la SB 1, et une qualité de service faible. Améliorer le réseau avec un système qui considère en priorité les flux VIP (Fig.8.1b), fixe temporairement la garantie de preformance pour les flux VIP, sans impacter la charge totale. Cependant, lorsqu'un autre flux VIP est ajouté á la SB 1 (Fig.8.1c), la QdS de la SB 1 ne peut être atteinte, du fait de la forte concentration de charge VIP. Dans cette configuration, l'algorithme que nous proposons va passer un des deux flux VIP á la SB 2, permettant d'atteindre la QdS VIP pour les deux SB, au coût d'une diminution du SINR ou du taux instantané pour le flux que l'on a changé de SB.

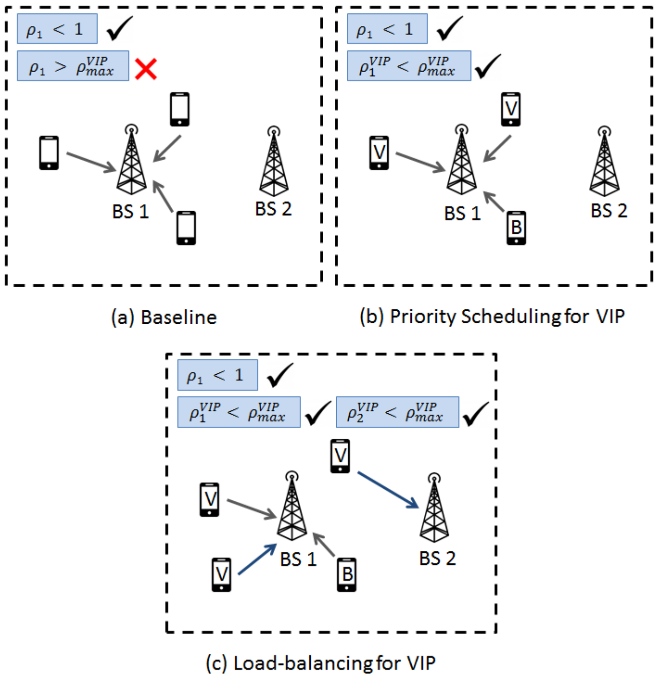

$\rho_1 < 1$ ✓
$\rho_1 > \rho_{max}^{VIP}$ ✗
BS 1
BS 2
(a) Baseline
$\rho_1 < 1$ ✓
$\rho_1^{VIP} < \rho_{max}^{VIP}$ ✓
V
B
BS 1
BS 2
(b) Priority Scheduling for VIP
$\rho_1 < 1$ ✓
$\rho_1^{VIP} < \rho_{max}^{VIP}$ ✓
$\rho_2^{VIP} < \rho_{max}^{VIP}$ ✓
V
B
BS 1
BS 2
(c) Load-balancing for VIP

Figure 8.1: Exemple simplifié pour différentes stratégies d'association

Dans notre travail, nous étendons le cadre de l'association d'utilisateurs de [19], et faisons les contributions suivantes : (i) A la différence de [19], pour s'assurer de l'isolation, nous supposons qu'á chaque SB on a un 'priority-based MAC scheduler', pour lesquels les flux VIP sont toujours servis avant les flux BE. Nous montrons qu'un tel scheduler peut être modélisé par une queue Processor Sharing (PS) á deux priorités. (ii) Nous introduisons une contrainte sur la charge VIP á chaque SB, qui sert á apporter des garanties de performance pour les VIP. (iii) Nous établissons de nouvelles règles l'association d'utlisitateurs distribuée, pour les services VIP et BE, pour lesquelles on prouve qu'elles convergent et optimisent une fonction α-optimale de la charge *totale* des stations de base. Cela maximise les gains du 'statistical multiplexing' dans la région de faisabilité. (iv) Nous montrons que notre politique d'association surpasse en performance la politique originale de 'meilleur effort' de [19], ainsi qu'une version améliorée de celle-ci, adaptée á la configuration 2 classes, qui utilise la règle d'association originale, tout en donnant la priorité aux flux VIP, en utilisant des traces telecom de la région de Milan [33].

8.2.1 Garanties de Qualité de Service

Nous motivons la priorité stricte donnée aux flux VIP ainsi que la contrainte cap pour chaque station de base en prouvant la proposition suivante:

Proposition 1 (Isolation et garanties de performance)**.** *En considérant que les stations de base suivent aussi une politique préemptive de priorité aux flux VIP, toutes les équations ci-dessus sont écrites pour ρ_i^V. En limitant la charge des VIP $\rho_i^V \le c_i$, on obtient les bornes supérieures suivantes:*

$$E[N_i^V] \le \frac{1}{1-c_i} - 1,$$
$$E[D_i^V] \le \frac{1}{\lambda_i}(\frac{1}{1-c_i} - 1).$$

Ainsi, pour garantir une certaine performance moyenne dans le delai, on optimise $\boldsymbol{\rho}$ pour la contrainte $\rho_i^V \le c_i$ à chaque station de base.

8.2.2 Formulation du problème

Nous définissons donc l'ensemble de faisabilité convexe des vecteurs de charge qui atteignent la QdS en se basant sur la proposition 1, et nous formulons le problème d'optimisation convexe:

Definition 1. *$\mathcal{F}$ est l'ensemble des vecteurs de charge à flux différenciés* *faisables* $\boldsymbol{\rho}^B, \boldsymbol{\rho}^V$*:*

$$\begin{aligned}\mathcal{F} = \{\boldsymbol{\rho} \mid \rho_i = & \int_{\mathcal{L}} \left(\varrho_i^V(x)\pi_i^V(x) + \varrho_i^B(x)\pi_i^B(x)\right) dx \\ & 0 \leq \rho_i \leq 1-\epsilon, \quad \forall i \in \mathcal{B} \\ & 0 \leq \rho_i^V \leq c_i, \ \forall i \in \mathcal{B} \\ & \sum_{i \in \mathcal{B}} \pi_i^T(x) = 1, \ \forall x \in \mathcal{L}, \ T = V, B \\ & 0 \leq \pi_i^T(x) \leq 1, \ \forall i \in \mathcal{B}, \ \forall x \in \mathcal{L}, \ T = V, B\},\end{aligned}$$

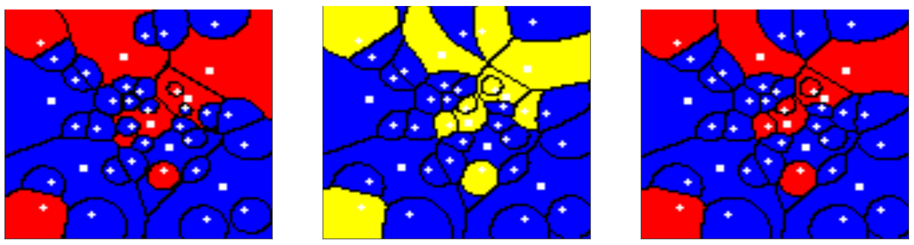
où c_i *est le seuil de la charge VIP de la station de base i.*

Problem 1 (P1: Le Problème de l'Association d'Utilisateurs avec de Différenciation de Service)**.**

$$\underset{\boldsymbol{\rho} \in \mathcal{F}}{\textit{minimize}} \ \phi_\alpha(\boldsymbol{\rho}) = \sum_{i \in \mathcal{B}} \frac{(1-\rho_i)^{1-\alpha}}{\alpha - 1}. \tag{8.1}$$

8.2.3 Règles d'Association d'Utilisateurs Distribuée

On procède à la relaxation des contraintes sur les charges VIP, ce qui crée une formulation du lagrangien partiellement relaxée, et on dérive des nouvelles règles d'association d'utilisateurs distribuée. Enfin, on présente le Distributed Constrained User Association Algorithm (DCUAA) qui résout le problème relaxé en combinant les règles optimales d'association d'utilisateurs dérivées, et en utilisant le sous-gradient pour calculer les multiplicateurs de Lagrange. L'algorithme est 'trigger based' et va continuer les itérations jusqu'à convergence. La sortie sera une carte d'association optimale pour les deux types de flux, connectant la localisation $x \in \mathcal{L}$ à la station de base $i \in \mathcal{B}$.



(a) MaxSINR (b) DCUAA VIP (c) Kim et al VIP

Figure 8.2: Comparaison de violation de contrainte entre les algorithmes, rouge signifie violation, jaune est étroit, bleu signifie inférieur au seuil. (b) et (c) sont calculés pour $\alpha = 1$.

Distributed Constrained User Association Algorithm (DCUAA)

```
Itérer sur t jusqu'à convergence
Les stations de base calculer $\gamma_i^{(t+1)} \leftarrow \left[\gamma_i^{(t)} + s^{(t)}\nabla_g \Phi_\alpha\right]^+$
Diffuser $\gamma_i^{(t+1)}$
    Itérer sur k jusqu'à convergence
    Utilisateur sur place $x \in \mathcal{L}$ calcule $\pi_i(x)$:
        $\pi_i^V(x) = 1\left\{ i^V(x) = \underset{j\in\mathcal{B}}{\operatorname{argmax}} \left\{ \frac{C_j(x)(1-\rho_j^{(k)})^\alpha}{1+\gamma_j^{(t+1)}(1-\rho_j^{(k)})^\alpha} \right\} \right\}$
        $\pi_i^B(x) = 1\left\{ i^B(x) = \underset{j\in\mathcal{B}}{\operatorname{argmax}} \left\{ C_j(x)(1-\rho_j^{(k)})^\alpha \right\} \right\}$
    La station de base $i \in \mathcal{B}$ mesure l'utilisation:
        $U_i^{(k)} = \min\left[\int_{\mathcal{L}} (\varrho_i^V(x)\pi_i^V(x) + \varrho_i^B(x)\pi_i^B(x))dx, 1-\epsilon\right]$
        $\rho_i^{(k+1)} = \beta\rho_i^{(k)} + (1-\beta)U_i^{(k)}$
    Diffuser $\rho_i^{(k+1)}$
```

8.2.4 Résultats

Dans nos expériences, nous montrons qu'il n'y a pas de violations de la contrainte sur les flux VIP sur des données traces réelles de traffic de réseau mobile, tandis que la politique distribuée baseline 'meilleur effort' appliquée à cette configuration amène jusquà 46,5% de violations. Ainsi, nous concluons que, dans un environnement qui évolue lentement, et pour lequel l'estimateur de charge moyenne converge, l'algorithme distribué dérivé fournit la QdS tout en utilisant efficacement les ressources du réseau.

Table 8.1: Résultats de Simulation sur le Milan Ensemble de Données

Algorithm	Constr. Violation %	Av. Delay Tot (s)
DCUAA $\alpha = 1$	0	0.2575
Kim et al. $\alpha = 1$	46.5	0.2471
DCUAA $\alpha = 2$	0	0.2814
Kim et al. $\alpha = 2$	38.9	0.2874
DCUAA $\alpha = 5$	0	0.4404
Kim et al. $\alpha = 5$	9.5	0.4368

8.2.5 Publications

Le travail exposé dans ce chapitre a été publié dans l'article suivant:

- *N. Liakopoulos, G. Paschos, Thr. Spyropoulos, "User Association for Wireless Network Slicing with Performance Guarantees", IEEE GLOBECOM, Abu Dhabi, UAE, December 2018*

8.3 Association d'Utilisateurs Centralisée et Scalable Basée sur le Transport Optimal Computationnel.

Dans ce chapitre, on étudie le problème qui consiste à connecter le traffic mobile aux stations Clous Radio Access Network (C-RAN). L'approche centralisée de ce chapitre est motivée par l'arrivée de l'architecture 5G pour les réseaux sans-fil. L'architecture C-RAN permet d'économiser en computation et en traitement du signal en migrant la partie calcul des stations de base vers un cloud centralisé. Le fait d'avoir les informations dans le controleur C-RAN permet, à la différence du chapitre précédent, de prendre des décisions informées sur la centralisation de l'association d'utilisateurs.

La très grande échelle des futurs réseaux sans-fil va encore causer des limitations computationnelles pour l'optimisation des performances. En pratique, un système générique de résolution de système linéaire échouera à résoudre un probmème d'optimisation de cette échelle, qui a des millions de variables [54]. Pour attaquer ce problème de connectivité massive, nous proposons une approche basée sur la théorie du transport optimal [29,30,54], qui étudie le transfert à moindre coût de deux distributions de probabilité. La méthodologie proposée peut ainsi inspirer des algorithmes scalables pour des problèmes d'optimisation à grande échelle dans les réseaux sans-fil.

8.3.1 Transport Optimal Régularisé

Le Transport Optimal (OT) discret est essentiellement un programme linéaire [29]. Le TO discret correspond à trouver un flux s-t à une unité, de minimum coût. En utilisant un réseau simplicial [62], on peut obtenir une solution en $O(E^2 \log V$, ce qui, pour $V = m+n, E = mn$, and $n = m$, devient $O(n^4 \log n)$, donc essentiellement quadratique en la taille de l'entrée mn. Même si une telle solution est polynomiale en la taille de l'entrée, la dimension de notre problème est si grande que le degré du polynôme devient important. Dans ce qui suit, nous décrivons le TO régularisé, une méthode qui donne une approximation du TO en $O(n^2 \log n)$.

En 2013, Cuturi [54] propose d'approximer le TO par une version régularisée. En particulier, il a proposé de modifier l'objectif du TO en soustrayant l'entropie $H(\boldsymbol{\pi}) = -\sum_{ij} \pi_{ij}(\log \pi_{ij} - 1)$, pondérée par le *coefficient force de régularisation* $\epsilon > 0$. Le TO régularisé devient:

$$\begin{aligned}\min_{\pi_{ij}} &\sum_{ij} C_{ij}\pi_{ij} + \epsilon \sum_{ij} \pi_{ij}(\log \pi_{ij} - 1) \\ \text{s.t.} &\sum_j \pi_{ij} = p_i, \quad i = 1, \dots, m, \\ &\sum_i \pi_{ij} = q_j, \quad j = 1 \dots, n.\end{aligned} \tag{8.2}$$

8.3.2 Sinkhorn Algorithme

On peut reformuler le problème Eq.8.2 et le résoudre par l'algorithme Sinkhorn:

Sinkhorn Algorithme

```
Input  : $\boldsymbol{C}$, $\boldsymbol{p}$, $\boldsymbol{q}$, $\epsilon$
Output : $\boldsymbol{\pi}$
initialize $\boldsymbol{b}^{(0)} = \mathbf{1}$, $\xi_{ij} = e^{-C_{ij}/\epsilon}$;
while accuracy do
    $k \leftarrow k + 1$ ;
    $a_i^{(k)} \leftarrow \frac{p_i}{\sum_j b_j^{(k-1)}\xi_{ij}}, \quad \forall i$ ;
    $b_j^{(k)} \leftarrow \frac{q_j}{\sum_i a_i^{(k)}\xi_{ij}}, \quad \forall j$ ;
end
$\pi_{ij} \leftarrow \xi_{ij} a_i^{(k)} b_j^{(k)}, \quad \forall (i, j)$
```

On peut montrer le théorème suivant sur la complexité de l'algorithme:

Theorem 1 (De [69]). *Supposons que $m = n$, fixons $\tau > 0$, et choisissons $\gamma = \frac{4 \log n}{\tau}$. L'algorithme Sinkhorn calcule une τ-approximation d'une solution de (4.2) en $O(n^2 \log n \tau^{-3})$ opérations.*

8.3.3 Choix de design pour l'association d'utilisateurs

Pour obtenir une règle faisable d'association d'appareils, on doit sélectionner les éléments suivants:

- Pour $\boldsymbol{C}$: nous proposons d'utiliser (i) la distance euclidienne entre la position x_i de l'appareil i, et la position x_j du RRH, i.e. $C_{ij} = ||x_i - x_j||$, ou bien la charge subie par unité de trafic : $C_{ij} = \frac{1}{\mu R_{ij}}$.
- Pour $\boldsymbol{p}$: le trafic d'entrée, $p_i = \lambda_i$.

- Pour $\boldsymbol{q}$: nous proposons d'utiliser (i) trafic RRH égal $q_j = \sum_i \lambda_i / n$, (où n est le nombre de RRHs) ou bien (ii) le trafic RRH à partir de la règle maxSINR, $\Lambda_j = \sum_i \pi_{ij}^{SINR} \lambda_i$.

8.3.4 Association d'Utilisateurs Performance d'Algorithme de Sinkhorn

Devices	RRHs	*LP-glpk*	*Sinkhorn* $\tau = 1\%$	*Sinkhorn* $\tau = 0.1\%$
10	25	6	2.27	5.02
50	25	88	4.09	8.95
100	25	315	8.09	9.7
500	25	8130	10.1	31.6
1000	25	out of memory	27	37.9
5000	25	out of memory	135	204
10000	25	out of memory	434	568

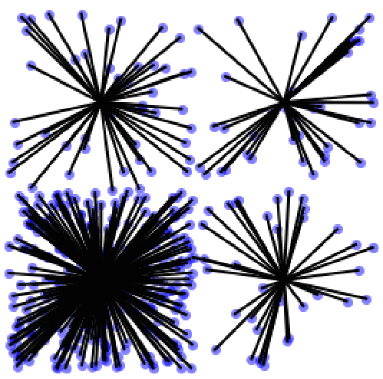
Table 8.2: Comparaison d'Exécution (msec) Sinkhorn vs LP-glpk.

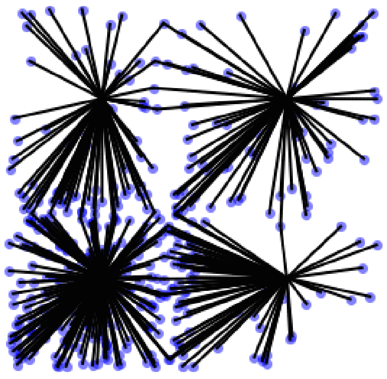
8.3.5 Equilibrage Heuristique de la Charge

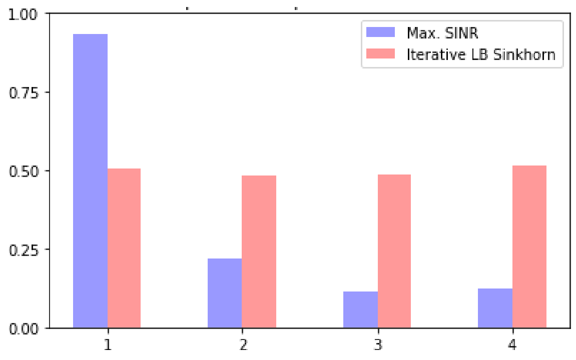
Nous proposons également un algorithme itératif, où à chaque itération k l'algorithme Sonkhorn est utilisé avec $\boldsymbol{q}^{(k)}$ pour obtenir une association qui donne une charge RRH spécifique $\rho_j(\boldsymbol{\pi}^{(k)})$. A partir de cette charge, une nouvelle marginale $\boldsymbol{q}^{(k+1)}$ est calculée, et le procédé se répète jusqu'à satisfaction du critère de précision. Plus spécifiquement, notre algorithme sélectionne le RHH avec la plus grande charge, et diminue son trafic agrégé par un term fixé δ, en dispersant le trafic à tous les autres RHH. Nous précisons qu'augmenter le trafic pour un RHH lointain entraînera un grand nombre de changements dans les associations de l'algorithme Sinkhorn, ce qui assure que le trafic dirigé maintient un coût de transport minimal.

8.3.6 Résultats

Nous montrons que l'heuristique décrite ci-dessus équilibre la charge et éloigne le trafic des stations surchargées, ce qui réduit le délai par un facteur 4. Notre procédé atteint une complétion moyenne de 6.3 msec, tandis que la règle maxSINR donne 24 msec, ce qui correspond à une amélioration par un facteur de proche de 4.


(a) maxSINR


(b) TO Adaptatif



(c) Charge (trafic non uniforme)

Figure 8.3: Comparaison de MaxSINR et du Transport Optimal.

8.3.7 Publications

Le travail exposé dans ce chapitre a été publié dans l'article suivant:

- *G. Paschos, N. Liakopoulos, Mérouane Debbah, Tong Wen, "Computational Optimal Transport for 5G Massive C-RAN Device Association", IEEE GLOBECOM, Abu Dhabi, UAE, December 2018*

8.4 Association d'Utilisateurs Data-Driven, Basée sur l'Optimisation Robuste

Dans ce qui suit, nous étudions le problème d'association d'utilisateurs avec garanties QdS, dans le contexte d'UDN, où le trafic de demande évolue rapidement en temps et en espace, en suivant l'activité humaine. Dans ce contexte, les algorithmes d'adaptation standard deviennent inefficaces [9, 16, 96]. Ici, nous proposons des décisions d'association faites par l'intelligence centrale du réseau sans-fil, le C-RAN, basées sur des données d'activités de télécommunication collectées et traitées par le réseau.

Nous montrons d'abord comment prédire la vraie demande de trafic et son erreur, puis nous montrons via un lemme que le vrai charge d'une station de base (charge et erreur moyenne) dépend de la décision d'association, c'est pourquoi nous optimisons les statistiques de second ordre (à la fois la moyenne et la variance de l'erreur), et nous montrons comment rendre le problème convexe et le résoudre, pour n'importe quel objectif convexe et séparable, soit exactement soit approximativement, selon la difficulté à calculer les dérivées. Par la suite, nous présentons les résultats numériques, et concluons par la validation du modèle d'erreur gaussien sur l'ensemble de données choisi, et mettons en lumière les améliorations obtenues en utilisant des techniques de prédiction avancées.

8.4.1 Méthodes Statistiques Pour la Prédiction du Trafic Mobile

Le trafic mobile présente des régularités diurnes, ce qui le rend sa prédiction possible [13–15]. Nous utilisons un ensemble de données public collecté dans la région de Milan, analysé dans [13], où il a été observé que le schéma journalier est plus marqué lorsque l'on considère chaque jour de la semaine et chaque lieu de Milan séparément.

Definition 2 (Prédicteur de Charge de Trafic). *Soit $X_i^{t,d}(x)$ l'intensité mesurée (nombre d'arrivées/h) à la position x, l'heure t, et le jour d de la semaine, i jour avant le jour actuel. La prédiction de l'intensité spatiale est basée sur les données des n dernières semaines;*

$$\bar{\lambda}^{t,d}(x) = \frac{1}{n}\sum_{i=1}^{n} X_i^{t,d}(x). \tag{8.3}$$

Nous nous concentrons sur un unique intervalle de temps (heure/jour) et omettons la notation t, d. La vraie valeur de l'intensité du trafic est modélisée par:

$$\hat{\lambda}(x) = \bar{\lambda}(x) + N_n(x), \tag{8.4}$$

où $N_n(x) \sim N(0, \sigma_n^2(x))$, et $\sigma_n^2(x)$ et la variance de l'échantillon, qui est donnée par

$$\sigma_n^2(x) = \frac{1}{n}\sum_{i=1}^{n}(X_i(x) - \bar{\lambda}(x))^2.$$

Le lemme qui suit caractérise le comportement de la charge de la station de base en fonction de l'erreur de prédiction.

Lemma 1. *On fixe (t, d). Soit l'intensité horaire spatiale du trafic $\hat{\lambda}(x)$ liée à celle prédite, comme dans Eq.(8.4). On fixe le vecteur d'association d'utilisateurs $\boldsymbol{\pi}$. La vrai charge $\hat{\rho}_i$ de la station de base i est liée à celle estimée ρ_i par:*

$$\hat{\rho}_i = \rho_i(\boldsymbol{\pi}_i) + Y_i(\boldsymbol{\pi}_i), \tag{8.5}$$

où Y_i - l'erreur de prédiction de la charge de la station de base - est un v.a. gaussienne de moyenne nulle et de variance:

$$S_i^2(\boldsymbol{\pi}_i) = \int_{\mathcal{L}} \frac{\pi_i^2(x)}{\mu^2 C_i^2(x)}\sigma_n^2(x)dx. \tag{8.6}$$

On crée des 'Robust User Association Maps' (RUAM) qui sélectionnent $\boldsymbol{\pi}$ en considérant de manière jointe leur impact sur l'objectif de charge de la station de base et l'erreur de prédiction.

8.4.2 Robust User Association Maps

Pour créer des cartes robustes en termes de fonction objectif, on cherche à optimiser $\mathbb{E}_Y[\phi(\hat{\boldsymbol{\rho}}(\boldsymbol{\pi}))]$, et l'espérance est calculée par rapport à l'erreur de prédiction gaussienne. En termes de contraintes, on requiert que la vraie charge de la SB ne dépasse pas un paramètre réglable c_i avec une certaine probabilité ϵ_i:

$$\mathbb{P}(\hat{\rho}_i \geq c_i) \leq \epsilon_i. \tag{8.7}$$

On prouve le lemme suivant pour pouvoir remplacer la contrainte stochastique par une contrainte convexe équivalente:

Lemma 2. *L'inégalité $\mathbb{P}(\hat{\rho}_i \geq c_i) \leq \epsilon_i$ est équivalente à $\rho_i + \alpha_i S_i \leq c_i$, quand $\hat{\rho}_i = \rho_i + Y_i$ est choisi selon Eq.(8.5), où Y_i suit une loi normale centrée et de variance $S_i^2(\boldsymbol{\pi_i})$, et $\alpha_i = Q^{-1}(\epsilon_i)$, où $Q(\cdot)$ est la queue de probabilité de la loi standard.*

La formulation du problème d'optimisation robuste est:

Definition 3 (Ensemble de Faisabilité Convexe $\mathcal{F}^c$)**.**

$$\mathcal{F}^c = \left\{ \boldsymbol{\pi} \in [0,1]^{\mathcal{L}\times\mathcal{B}} \middle| \begin{array}{ll} \rho_i + \alpha_i S_i \leq c_i, & \forall i \in \mathcal{B}, \\ \sum_{i\in\mathcal{B}} \pi_i(x) = 1, & \forall x \in \mathcal{L}. \end{array} \right\} \tag{8.8}$$

Problem 2 (P: Convex Robust User Association Problem)**.**

$$\underset{\boldsymbol{\pi}\in\mathcal{F}^c}{\text{minimize}}\ \mathbb{E}\left[\phi(\boldsymbol{\pi})\right]. \tag{8.9}$$

8.4.3 Algorithme de Carte Robuste Généralisé

Dans ce qui suit, on relaxe la contrainte sur la charge; l'ensemble de faisabilité du problème relaxé (intérieur) pour un multiplieur donné $\boldsymbol{\gamma}$ est un simplexe, et on peut le résoudre avec un algorithme de gradient projeté efficace.

Projected Gradient Descent (PGD) on $\mathcal{F}'$

```
**Initialiser:** $\boldsymbol{\pi}^{(0)}$ (peut être infaisable), $\boldsymbol{\gamma}^{\star}$.
**Répéter:** sur n, jusqu'à convergence
    $\boldsymbol{y}^{(n+1)} = \boldsymbol{\pi}^{(n)} - \eta^{(n)}\nabla_{\boldsymbol{\pi}}\Phi(\boldsymbol{\pi}^{(n)}, \boldsymbol{\gamma}^{\star})$    (8.10)
    $\boldsymbol{\pi}^{(n+1)} = \Pi_{splx}[\boldsymbol{y}^{(n+1)}]$    (8.11)
Où $\Pi_{splx}$ est la projection euclidienne sur $\mathcal{F}'$:
    Trier $\boldsymbol{y}^{(n+1)}$ par ordre décroissant $(y_1 \geq y_2 \geq \ldots \geq y_{|\mathcal{B}|})$
    Sélectionner $m = \underset{j\in\mathcal{B}}{\text{argmax}}\{j \mid y_j + \frac{1}{j}(1 - \sum_{i=1}^{j} y_i) > 0\}$
    $\pi_i^{(n+1)} = \left[y_i + \frac{1}{m}(1 - \sum_{i=1}^{m} y_i)\right]^+, \; i = 1, \ldots, |\mathcal{B}|$
```

Enfin, on présente l'algorithme GRMA, qui est basé sur une méthode de sous-gradient dual avec moyennage que la suite primale $\boldsymbol{\pi}^{(k)}$, et on montre qu'il converge vers la carte robuste optimale.

Generalized Robust Map Algorithm (GRMA)

```
**Initialiser:** $\boldsymbol{\pi}^{(0)}$ (e.g. *MaxSINR*, peut être infaisable), $\boldsymbol{\gamma}^{(0)}$.
**Répéter:** sur k, jusqu'à la convergence:

    $\boldsymbol{\gamma}^{(k+1)} = [\boldsymbol{\gamma}^{(k)} + s^{(k)}\boldsymbol{g}^{(k)}]^+$
    $\boldsymbol{\pi}^{(k+1)} \leftarrow \text{PGD}(\boldsymbol{\gamma}^{(k+1)})$
*Garder la moyenne courante du* $\boldsymbol{\pi}^{(k)}$ *(Eq.*(4.15)*):*
$\bar{\boldsymbol{\pi}}^{(k)} = \frac{1}{k}\sum_{i=0}^{k-1}\boldsymbol{\pi}^{(i)}$
```

Nous prouvons que GRMA résout de manière optimale P2:

Theorem 2 (Convergence à Primal Optimal)**.** *La moyenne des itérées primales* $\bar{\boldsymbol{\pi}}^{(k)} = \frac{1}{k}\sum_{i=0}^{k-1}\boldsymbol{\pi}^{(i)}$*, où*

$$\boldsymbol{\pi}^{(i)} \in \underset{\boldsymbol{\pi}\in\mathcal{F}'}{\mathit{argmin}}\left\{\mathbb{E}[\phi(\boldsymbol{\pi})] + \sum_{j\in\mathcal{B}}\gamma_j^{(i)}(\rho_j + \alpha_j S_j - c_j)\right\}, \tag{8.12}$$

converge asymptotiquement vers (ou se rapproche) de la carte d'association robuste optimale $\boldsymbol{\pi}^{\star}$*, i.e.:*

$$\lim_{k\to\infty} ||g(\bar{\boldsymbol{\pi}}^{(k)})^+|| \to 0 \;\mathit{and}\; \lim_{k\to\infty} \mathbb{E}[\phi(\bar{\boldsymbol{\pi}}^{(k)})] = \mathbb{E}[\phi(\boldsymbol{\pi}^{\star})].$$

8.4.4 Approximation de Cartes Robustes

Selon le choix de la fonction d'objectif, le calcul des dérivées peut être très complexe, c'est pourquoi nous proposons des méthodes pour approximer la carte optimale, basées sur des approximations de Taylor, l'inégalité de Jensen (MCEL), et l'$\boldsymbol{\epsilon}$- intervalle de confiance supérieur (MWCC).

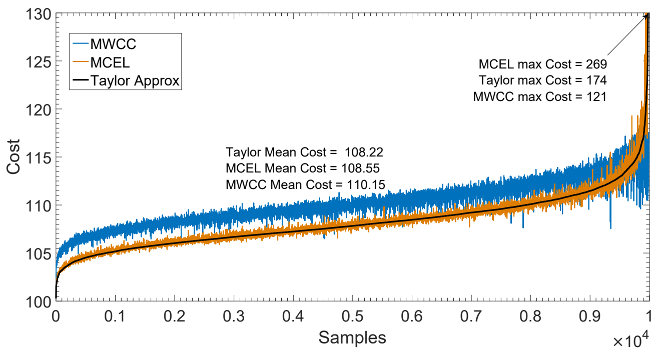


Figure 8.4: Courbes de coûts pour les méthodes d'approximation. Les cartes robustes sont générées pour chaque méthode d'approximation pour l'objectif de minimisation du delai ($\alpha = 2$). 10000 échantillons sont générés en se basant sur la prédiction du trafic pour une heure de pointe de l'ensemble de données. Les échantillons dont ordonnés par le coût de l'approximation de Taylor. On observe que le coût pour MCEL explose en s'approchant de la queue de distribution, tandis que les autres méthodes restent robustes.

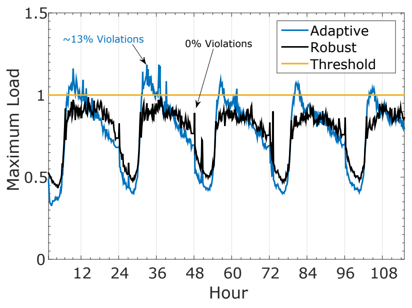
8.4.5 Résultats

Dans la section résultats numériques, on valide nos cartes robustes sur les traces de la région de Milan [33], avec couverture dense, et trouvons que l'on peut réduire les violations de 25% (infligées par un algorithme de base adaptatif) à presque zéro. De plus, on motnre l'effet des α-objectifs [19, 32].

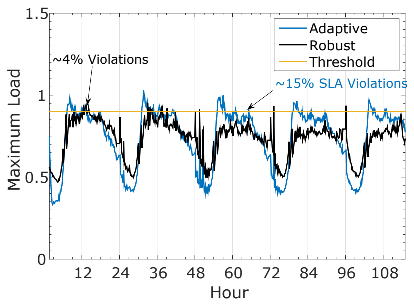
Table 8.3: Pic de Trafic 1ère Semaine de Décembre Micro Setup

ϵ	Average Cost	Average Delay (s)	Violations (%)
Adaptive	9.671	2.858	24.6
10%	10.911	2.251	7.7
5%	11.000	2.140	3.1
0.1%	11.415	2.030	0

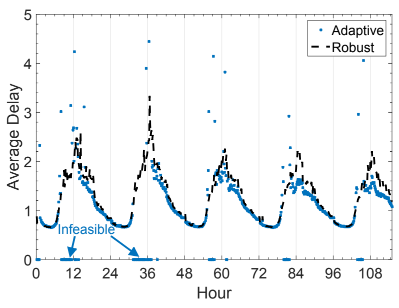
Enfin, on discute les méthodes avancées de prévision de séries temporelles, comme SARIMA [34] et les réseaux de neurones LSTM [34]. En utilisant ces méthodes sur les données [33], nous validons le modèle gaussien d'estimation de l'erreur et améliorons la qualité de la carte produite par GRMA en diminuant l'écart dans le coût avec une carte optimale à 5%.



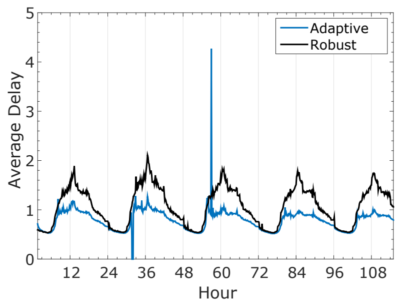
(a)



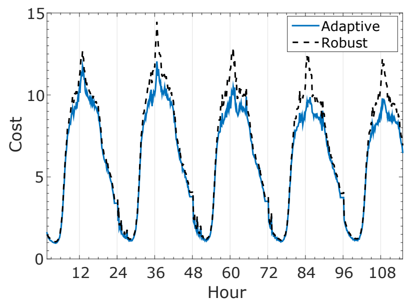
(b)



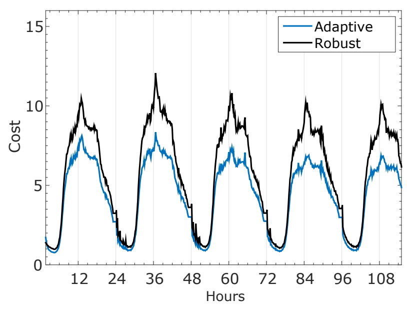
(c)



(d)



(e)



(f)

Figure 8.5: Colonne de gauche: Configuration de la station de base Micro, protection élevée des SLA $\epsilon = 0.001$, 2 au 6 décembre (a) Violation (c) Délai système moyen (e) Coût. Colonne de droite: Configuration de la station de base à 2 niveaux, protection SLA légère $\epsilon = 0.05$, 2 au 6 décembre (b) Violation (d) Délai système moyen (f) Coût.

Table 8.4: Comparaison Entre les Méthodes de Prévision du Trafic.

Prediction	MSE (10^6)	Avg Cost	11h-13h	Violations
Sample Mean	1,290	6,60	11,42	<0,1%
SARIMA	0,280	6,22	10,43	<0,1%
LSTM NN	0,266	6,18	10,28	<0,1%
Oracle	-	*5,90*	*9,86*	*0%*

8.4.6 Publications

Le travail exposé dans ce chapitre a été publié dans les articles suivant:

- *N. Liakopoulos, G. Paschos, Thr. Spyropoulos, "Robust User Association for Ultra Dense Networks", IEEE INFOCOM, Honolulu, HI, April 2018*
- *N. Liakopoulos, G. Paschos, Thr. Spyropoulos, "Robust Optimization Framework for Proactive User Association in UDNs: A Data-Driven Approach", to appear in IEEE Transactions on Networking, 2019*

8.5 Réservation en Ligne de Ressources en Nuages avec des Contraintes Budgétaires

Nous étudions ici l'apprentissage d'une réservation de ressources à la fois économique et robuste pour le informatique en nuages, à savoir réservation de ressources juste satisfaisants pour répondre aux exigences d'exécution de l'application [25]. L'objectif est d'obtenir des réservations qui répondent exactement aux exigences: le surapprovisionnement en ressources entraîne des frais d'exploitation excessifs, tandis que le sous-approvisionnement peut dégrader considérablement la qualité du service, en entraînant des interruptions et du déploiement en temps réel des ressources supplémentaires, ce qui coûte très cher [26].

8.5.1 Charge de Travail sur le Cloud Non-Stationnaire

La réservation de ressources dans le cloud computing (informatique en nuages) est un défi, étant donné que l'hypothèse de prévisibilité de la demande ne tient pas [1]. Notre expérimentation sur un ensemble de données de cluster Google [2, 86] récupère des résultats similaires, v. Fig.8.6.

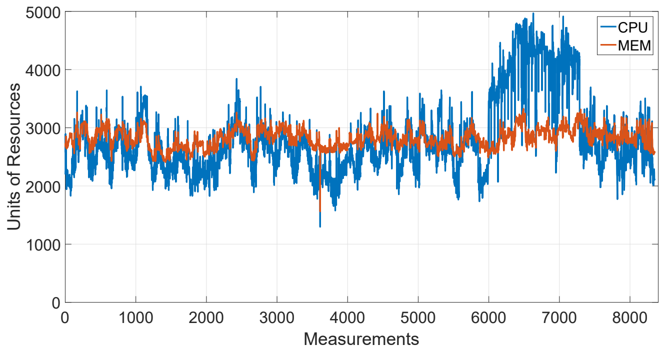


Figure 8.6: Utilisation globale des ressources du cluster Google [2]. Les ressources sont normalisées par rapport au serveur avec la mémoire et le CPU les plus importants. Chaque point correspond à 5 minutes, jusqu'à 29 jours mesurés. Les fluctuations sont caractérisées comme imprévisibles dans [1]. The fluctuations are characterized as unpredictable in [1].

8.5.2 Modèle de Système

Violation garantie. Le v_i^t dénote le cas de violation de la resource i dans le créneau t, qui se produit lorsque la demande d'une ressource dépasse la réservation, c'est-à-dire $v_i^t \triangleq \mathbb{1}\left\{\lambda_i^t > x_i^{t,\pi}\right\}$. Une stratégie π est *faisable* si les violations de moyenne temporelle de la ressource i ne dépassent pas un seuil prédéterminé ϵ_i:

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\left[v_i^t\right] \leq \epsilon_i, \quad \text{pour tous} \;\; i = 1, \ldots, I.$$

Par conséquent, la contrainte de faisabilité de la ressource i peut également être écrite sous la forme:

$$\sum_{t=1}^{T}\mathbb{P}\left(\lambda_i^t > x_i^t\right) \leq \epsilon_i T. \tag{8.13}$$

Énoncé du Problème.

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \textstyle\sum_{t=1}^{T}\sum_i c_i(x_i^t), \\ \text{subject to} \quad & \textstyle\sum_{t=1}^{T}\mathbb{P}\left(\lambda_i^t > x_i^t\right) \leq \epsilon_i T, \; \forall i, \end{aligned} \tag{8.14}$$

oú $\boldsymbol{\lambda}^t$ sont choisis par un adversaire et sont révélés au décideur après que la réservation $\boldsymbol{x}^t$ soit décidée.

Le Regret. Le regret R_T^π est la différence cumulative de pertes entre la stratégie π et un benchmark stratégie qui est au courant de tout le trajet de l'échantillon mais qui est forcé de prendre une action statique tout au long de l'horizon-souvent appelé *meilleure stratégie statique a posteriori* [79, 80]. En particulier, nous laissons $\boldsymbol{x}^*$ dénotent notre benchmark, qui est calculé comme la solution au problème suivant:

$$\boldsymbol{x}^* \in \arg\min_{\boldsymbol{x}\in\mathbb{R}_+^I} T\,C(\boldsymbol{x}) \;\; \text{s.t.} \;\; \sum_{t=1}^{T}\mathbb{P}\left(\lambda_i^t > x_i^*\right) \leq \epsilon_i T.$$

Le regret est ainsi défini comme suit:

$$R_T^\pi = \mathbb{E}\left[\sum_{t=1}^{T} C(\boldsymbol{x}^{t,\pi}) - T\,C(\boldsymbol{x}^*)\right].$$

Si $R_T^\pi = o(T)$, dans ce cas la stratégie π n'a "aucun regret" (no regret), car $R_T^\pi/T \to 0$ comme $T \to \infty$, à savoir les pertes moyennes du benchmark sont amorties.

K-slot Benchmark. Le K-benchmark policy garantit la violation contrainte pour toutes les fenêtres de K slots dans l'horizon T, tout en minimisant le coût:

$$\begin{aligned} & \boldsymbol{x}^*(K) \in \arg\min_{\boldsymbol{x}\in\mathbb{R}_+^I} T\,C(\boldsymbol{x}) \qquad (8.15) \\ & \text{s.t.} \sum_{k=0}^{K-1}\mathbb{P}\left(\lambda_i^{t+k} > x_i^*\right) \leq \epsilon_i K, \;\; \forall t = 1, \ldots, T-K. \end{aligned}$$

Nous présentons THOR et nous prouvons que cette stratégie n'a "aucun regret" contre le K-benchmark, lorsque $K = o(T)$. Il convient de noter que ceci équivaut à montrer que π atteind la même performance moyenne que le benchmark.

8.5.3 Time Horizon Online Reservation

Puisque les fonctions de contrainte sont révélées après que nous déterminions la réservation du vecteur $\boldsymbol{x}$, à chaque créneau, nous décidons sur la base de la prédiction linéaire de probabilité d'erreur suivante :

$$b_i(x_i^t) \triangleq F_{t-1}(x_i^{t-1}) + F'_{t-1}(x_i^{t-1})(x_i^t - x_i^{t-1}). \tag{8.16}$$

Notre algorithme décide des nouvelles réservations en fonction de la valeur de la predictor queue qui rassemble l'erreur de prédiction $b_i(x_i^t)$ à chaque itération, le paramètre de prudence V, le coût de la ressource c_i, la taille du pas α et le dérivé de la CCDF convexi du créneau horaire précédente $(F'_{t-1}(x_i^{t-1}))$.

Time Horizon Online Reservations (THOR)

Initialisation: Valeur initiale de la Predictor queue $\boldsymbol{Q}(1) = \boldsymbol{0}$, reservation initial du vecteur $\boldsymbol{x}^0 \in \mathbb{R}_+^I$.

Paramètres: pénalité constante V, taille du pas α, coût de l'unité de ressource c_i, exigence de contrainte par ressource $\epsilon_i \leq 0.5$.
Mises à jour à chaque créneau horaire $t \in \{1, \ldots, T\}$**:**

$$x_i^t = \left[x_i^{t-1} - \tfrac{1}{2\alpha}(Vc_i + Q_i(t)F'_{t-1}(x_i^{t-1})\right]^+, \tag{8.17}$$
$$Q_i(t+1) = [Q_i(t) + b_i(x_i^t) - \epsilon_i]^+. \tag{8.18}$$

Oú $b_i(x_i^t)$ est donné en Eq.(5.3) et $F'_t(x_i^t)$ c'est le dérivé de la CCDF convexi.

8.5.4 Résultats de Performance

Dans nos résultats théoriques, nous prouvons une sous-linéaire dans le temps délimité à la predictor queue de THOR, qui servira à prouver la faisabilité, à savoir, en moyenne temporelle, aucune violation de contrainte dans un horizon T. De plus, nous comparons THOR avec le K-benchmark, en utilisant les capacités du K-benchmark et nous prouvons l' "aucun regret". En combinant les résultats, nous obtenons le théorème suivant:

Theorem 3 (No Regret contre $K = o(T)$)**.** *Fix $\varepsilon > 0$, en mettant $\alpha = \frac{T^{\frac{3}{2}}}{V^{\frac{1}{2}}}$ et en prenant $K = T^{1-\varepsilon}$ et $V = T^{1-\frac{\varepsilon}{2}}$, THOR n'a aucun regret contre le*

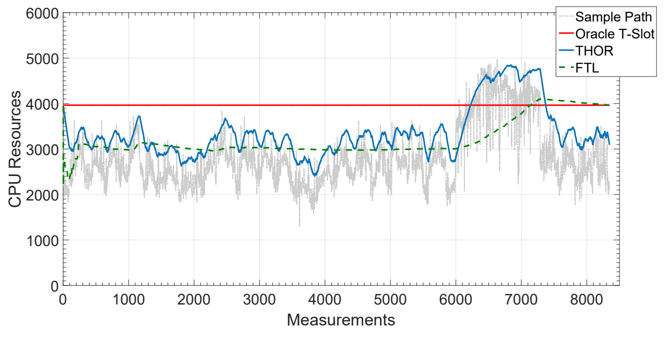
K-benchmark and il est faisable:

$$R_K(T) = O\left(T^{1-\frac{\varepsilon}{2}}\right),$$

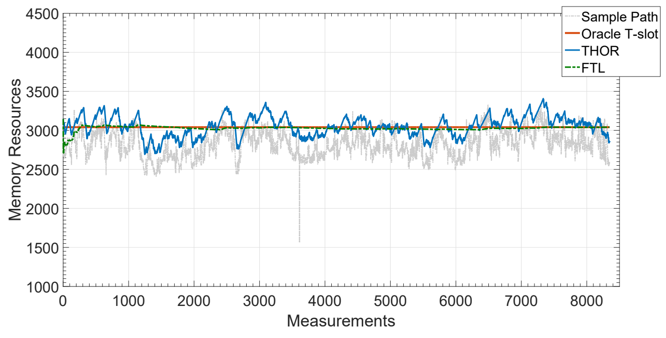
$$Ctr(T) = \sum_{t=1}^{T} \mathbb{P}\left(\lambda_i^t > x_i^t\right) - \epsilon_i T = O\left(T^{1-\frac{\varepsilon}{4}}\right).$$

8.5.5 Résultats Numériques

Dans la section numérique, nous validons les réservations de ressources THOR en utilisant un ensemble de données publiques fournies par Google [2]. THOR surpasse largement notre implementation du manuel Follow the leader (FTL) stratégie, garantissant les violations des contrainte, tout en réalisant des performances similaires ou parfois meilleures que la stratégie de statique oracle T-slot, dans la CPU charge de travail rigoureuse et non-stationnaire.



(a) CPU Réservations



(b) Mémoire Réservations

Figure 8.7: Parcelles de comparaison des mises à jour de réservation pour des différentes stratégies sur les traces de cluster Google [2]. Voir le tableau 8.5 pour la comparaison numérique.

Table 8.5: Tableau de Comparaison des Politiques, $\epsilon = 10\%$

Performance	T-slot Oracle	FTL	THOR
Average CPU Cost	3964	3203	3365
Average Violations (%)	10.00	21.41	5.64
Average MEM cost	3041	3054	3027
Average Violations (%)	10.00	11.84	3.7

8.5.6 Publications

Le travail exposé dans ce chapitre a été publié dans l'article suivant:

- *N. Liakopoulos, G. Paschos, Thr. Spyropoulos, "No Regret in Cloud Resources Reservation with Violation Guarantees", IEEE INFOCOM, Paris, FR, May 2019*

8.6 Réservation Générale de Ressources en Nuages en Ligne avec Contraintes Budgétaires: Un Nouveau Cadre

Dans le dernier chapitre du corpus principal du travail, nous étudions une catégorie de problèmes d'optimisation convexe en ligne avec des contraintes budgétaires à long terme qui se posent naturellement comme des garanties de fiabilité ou des contraintes de consommation totale. Dans ce contexte général, les travaux antérieurs de [37] ont montré qu'il est impossible d'atteindre aucun regret si les fonctions qui définissent le budget de l'agent, sont choisies par un adversaire. Pour surmonter cet obstacle, nous affinons la métrique de regret de l'agent en introduisant la notion de K-benchmark, à savoir un comparateur qui fait face au problème de budget alloué sur chaque fenêtre de longueur K.

L'analyse d'impossibilité de [37] est récupérée lorsque $K = T$; cependant, pour $K = o(T)$, nous montrons qu'il est possible de minimiser le regret tout en faisant face au problème des contraintes budgétaires à long terme. Nous y parvenons via un algorithme d'apprentissage en ligne fondé sur le Cautious Online Lagrangian Descent (COLD) pour lequel nous dérivons des limites explicites, tant en termes de regret engagé et de violations du budget résiduel.

8.6.1 Hypothèses

Dans le rond t, l'action $x_t \in \mathcal{X}$ s'expose à une perte $f_t(x_t)$ et affecte le budget d'un montant $g_t(x_t)$. Les fonctions f_t et g_t ne doivent pas nécessairement être dérivables et $f'_t(x_t), g'_t(x_t)$ indiquent des sous-gradients à x_t.

(A1) L'ensemble $\mathcal{X}$ est convexe et compact de diamètre D.

(A2) Pour toutes $t = 1, \dots, T$, les fonctions $f_t, g_t : \mathcal{X} \to \mathbb{R}$ sont convexes et Lipschitz, avec $\|f'_t\|_2 \leq G$ et $\|g'_t\|_2 \leq G$.

(A3) Pour un donné $K \leq T$, nous considérons l'ensemble des toutes les actions qui conservent un budget équilibré dans toutes les fenêtres de K ronds :

$$\mathcal{X}_K = \left\{ x \in \mathcal{X} : \sum_{\tau=t}^{t+K-1} g_\tau(x) \leq 0,\ 1 \leq t \leq T - K + 1 \right\} \tag{8.19}$$

Nous supposons que $\mathcal{X}_K$ n'est pas vide.

Etant donné que $\mathcal{X}$ est compact et f_t, g_t Lipschitz, il s'ensuit qu'ils sont également bornés, à savoir $|f_t(x)| \leq F$ et $|g_t(x)| \leq F$, pour tout $x \in \mathcal{X}$.

Les hypothèses A.1 à A.2 sont les hypothèses générales de tous les documents OCO précédents. A.3 est essentiel afin de définir la métrique de regret que nous utilisons, et c'est considérablement moins strict que l'hypothèse d'interieur non vide de Slater $\cap_t \{x : g_t(x) < 0\}$ de [40]. Par exemple, si gt est non négatif, l'hypothèse de Slater ne peut pas être efficace. Ce cas figure dans des problèmes où nous voulons nous assurer qu'un taux d'échecs ne dépasse pas un seuil, comme par exemple dans [88].

8.6.1.1 Regret sur K-benchmark

L'efficacité algorithmique est mesurée en OCO avec le regret statique : la différence de perte totale entre l'algorithme et celle d'une action benchmark à posteriori. Dans ce cas, la définition de regret appropriée doit préciser la façon dont l'action benchmark se comporte par rapport à la contrainte budgétaire à long terme. Dans ce but, nous introduisons la famille benchmark suivante.

Definition 4 (K-benchmark)**.** *Nous fixons $K \in \{1, \dots, T\}$. Le K-benchmark x_*^K est une action qui répond à:*

$$x_*^K \in \operatorname*{argmin}_{x \in \mathcal{X}_K} \sum_{t=1}^{T} f_t(x), \tag{8.20}$$

où $\mathcal{X}_K$ est défini en A.3.

Cela nous permet d'étendre la définition de regret de la manière suivante.

Definition 5 (Regret de x_t sur x_*^K)**.** *Nous fixons $K \in \{1, \dots, T\}$, et supposons qu x_*^K soit un K-benchmark. Le regret de x_t sur x_*^K est défini comme:*

$$R_K(T) = \sum_{t=1}^{T} f_t(x_t) - \sum_{t=1}^{T} f_t(x_*^K).$$

8.6.2 Cautious Online Lagrangian Descent

Cautious Online Lagrangian Descent (COLD)

Pour $t = 1, \dots, T$:

$$x_t = \Pi_{\mathcal{X}}\left[x_{t-1} - \frac{V f'_{t-1}(x_{t-1}) + Q(t) g'_{t-1}(x_{t-1})}{2\alpha}\right] \tag{8.21}$$

$$Q(t+1) = [Q(t) + \hat{g}_t(x_t)]^+. \tag{8.22}$$

avec l'initialisation $Q(1) = 0$, $x_0 \in \mathcal{X}$, et oú:

- $\Pi_{\mathcal{X}}[.]$ signifie la projection euclidienne sur l'ensemble $\mathcal{X}$,
- V est le paramètre de prudence configurable,
- f'_{t-1}, g'_{t-1} sont les vecteurs sous-gradients en rond $t-1$,
- $Q(t)$ est une file d'attente virtuelle qui est mise à jour en fonction de (8.22), et elle est appelée le *predictor queue*,
- α est le paramètre de force de régularisation configurable,
- $\hat{g}_t(x) \triangleq g_{t-1}(x_{t-1}) + \langle g'_{t-1}(x_{t-1}), x - x_{t-1}\rangle$,
- $[.]^+$ est $\max\{., 0\}$,

8.6.3 Analyse des Performances

Proposition 2 (COLD performance)**.** *Si les hypothèses de Sec.* (8.6.1) *sont satisfaites et des actions sont prises conformément à COLD, la contrainte résiduelle est bornée par:*

$$Ctr(T) \triangleq \sum_{t=1}^{T} g_t(x_t) \leq \Bigg\{ 2BKT + 4FVT + \frac{V^2G^2T}{\alpha} + 2\alpha D^2 + 2BK^2 + 2(T+1)BK \Bigg\}^{\frac{1}{2}} + \frac{GVT}{2\alpha} + \frac{G^2\left[\sqrt{2BK} + \sqrt{4FV} + \sqrt{\frac{V^2G^2}{\alpha}} + \sqrt{2BK}\right]\frac{T^{\frac{3}{2}}+T}{\sqrt{2}}}{2\alpha} + \frac{G^2\left[\sqrt{2\alpha D^2} + \sqrt{2BK^2} + \sqrt{2BK}\right]T}{2\alpha} \tag{8.23}$$

et le regret sur K-benchmark est borné par:

$$R_K(T) \triangleq \sum_{t=1}^{T} f_t(x_t) - \sum_{t=1}^{T} f_t(x_*^K) \leq \tag{8.24}$$
$$\leq \frac{BKT}{V} + \frac{G^2VT}{2\alpha} + B\frac{(K+1)(2K+1)}{6V} + \frac{D^2\alpha}{V} + 2F(K-1).$$

Theorem 4 (Compromis Réalisables)**.** *Fix $K \geq 1$ tel que $K = o(T)$ (le plus grand K rend le K-benchmark plus solide). Choisis des $V \in (K,T)$, et $\alpha = \max\{T, V\sqrt{T}\}$. Ensuite,* (6.13)-(6.14) *simplifie à :*

$$\underbrace{\mathcal{O}(KT/V + \sqrt{T})}_{\textit{regret sur } K\textit{-benchmark}} \quad \textit{and} \quad \underbrace{\mathcal{O}(\sqrt{VT})}_{\textit{contrainte résiduelle}}$$

De plus, suppose que $K = T^{1-\epsilon}$ pour certains inférieurs $\epsilon > 0$ et choisis $V = T^{1-\frac{\epsilon}{2}}$, et $\alpha = V\sqrt{T}$. Ensuite, (6.13)-(6.14) *simplifie à:*

$$\underbrace{\mathcal{O}(T^{1-\frac{\epsilon}{2}})}_{\textit{regret sur } T^{1-\epsilon}\textit{-benchmark}} \quad \textit{et} \quad \underbrace{\mathcal{O}(T^{1-\frac{\epsilon}{4}})}_{\textit{contrainte résiduelle}}$$

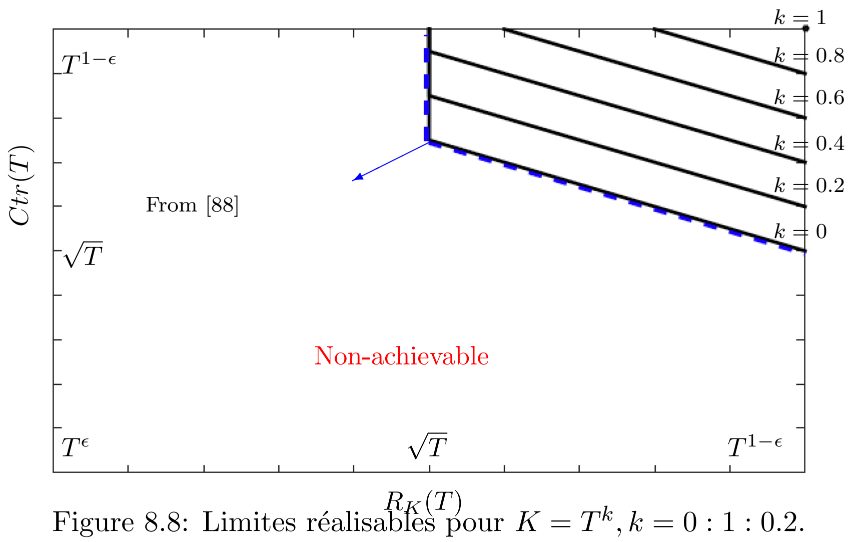


Figure 8.8: Limites réalisables pour $K = T^k, k = 0 : 1 : 0.2$.

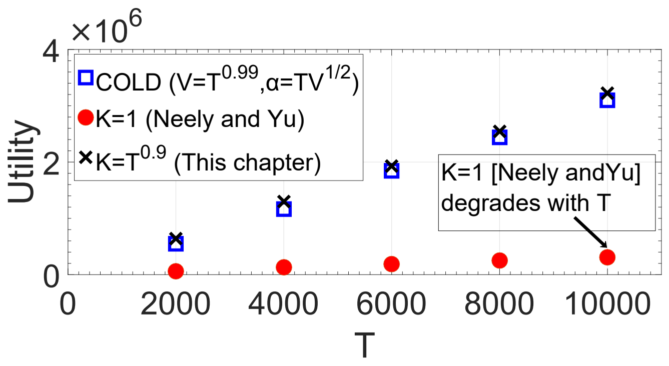
8.6.4 Résultats Numériques

Nos conclusions théoriques sont également validées par une série d'expériences numériques (voir la Fig.8.9), qui suggèrent que l'augmentation de K - c'est-à-dire l'élargissement de la fenêtre sur laquelle le budget doit être équilibré – renforce la garantie de K-benchmark. Par conséquent, la démonstration non regret sur K-benchmark pour un grand K aboutit aux garanties de performance plus solides - une observation qui n'est pas a priori évidente dans un contexte bona fide antagoniste. De plus, nous démontrons l'effet du paramètre de prudence V, voir Fig.8.10.

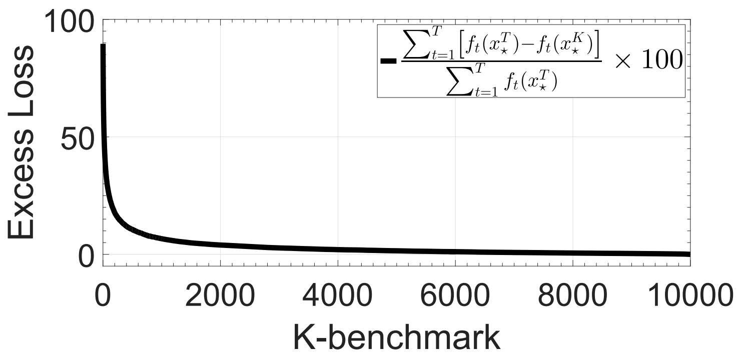
8.6.5 Publications

Le travail de ce chapitre a été publié dans le document suivant:

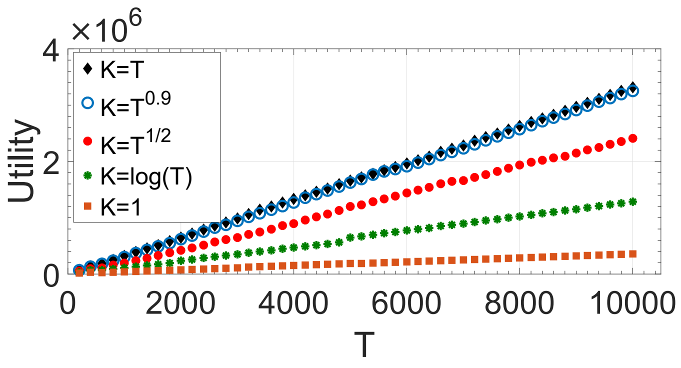
- *N. Liakopoulos, A. Destounis, G. Paschos, Thr. Spyropoulos, Panayotis Mertikopoulos, "Cautious Regret Minimization: Online Optimization with Long-Term Budget Constraints", to appear at ICML, Long Beach, CA, June 2019*



(a) COLD vs $K = 1, K = T^{0.9}$

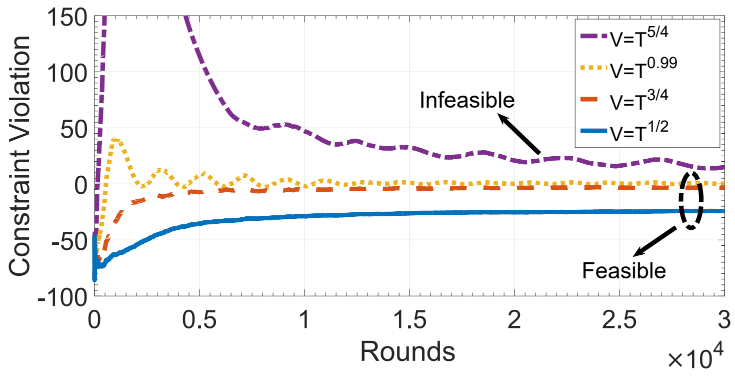


(b) K-benchmark vs T-benchmark

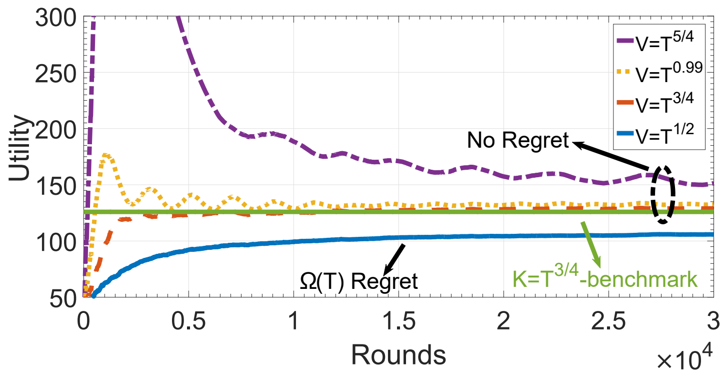


(c) Utility for different scaling of K

Figure 8.9: Les simulations de l'exemple de placement d'annonce en ligne. (a) Comparaison de l'utilité de COLD par rapport 1-benchmark et $T^{0.9}$-benchmark. (b) La perte relative absolue de K-benchmark par rapport à T-benchmark. (c) Utilités de K-benchmark pour des valeurs différentes de K.



(a) $\sum_{\tau=1}^{t} g_\tau(x_\tau)/t$ of COLD



(b) $-\sum_{\tau=1}^{t} f_\tau(x_\tau)/t$ of COLD

Figure 8.10: Les moyennes flottantes des performances des contraintes résiduelles et des contraintes d'utilisation de COLD pour des valeurs différentes du paramètre V.

Bibliography

[1] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch, "Heterogeneity and Dynamicity of Clouds at Scale: Google Trace Analysis," in *ACM Symposium on Cloud Computing (SoCC)*, San Jose, CA, USA, Oct. 2012.

[2] C. Reiss, J. Wilkes, and J. L. Hellerstein, "Google Cluster-Usage Traces: Format + Schema," Google Inc., Mountain View, CA, USA, Technical Report, Nov. 2011, revised 2014-11-17 for version 2.1. Posted at https://github.com/google/cluster-data.

[3] D. P. Bertsekas, *Network Optimization: Continuous and Discrete Models*. Athena Scientific Belmont, 1998.

[4] "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016–2021,," Cisco, February 2017.

[5] M. Kamel, W. Hamouda, and A. Youssef, "Ultra-Dense Networks: A Survey," *IEEE Communications Surveys Tutorials*, vol. 18, no. 4, pp. 2522–2545, 2016.

[6] N. Bhushan, J. Li, D. Malladi, R. Gilmore, D. Brenner, A. Damnjanovic, R. T. Sukhavasi, C. Patel, and S. Geirhofer, "Network Densification: The Dominant Theme for Wireless Evolution into 5G," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 82–89, February 2014.

[7] Nokia, "Ultra Dense Network UDN," 2016.

[8] "Making 5G a reality: Addressing the strong mobile broadband demand in 2019 & beyond," Qualcomm and Nokia, 2018. [Online]. Available: https://www.qualcomm.com/documents/whitepaper-making-5g-nr-reality

[9] D. Liu, L. Wang, Y. Chen, M. Elkashlan, K. K. Wong, R. Schober, and L. Hanzo, "User Association in 5G Networks: A Survey and an Outlook," *IEEE Communications Surveys Tutorials*, vol. 18, no. 2, pp. 1018–1044, 2016.

[10] M. Agiwal, A. Roy, and N. Saxena, "Next Generation 5G Wireless Networks: A Comprehensive Survey," *IEEE Communications Surveys Tutorials*, vol. 18, no. 3, pp. 1617–1655, 2016.

[11] N. Nikaein, E. Schiller, R. Favraud, K. Katsalis, D. Stavropoulos, I. Alyafawi, Z. Zhao, T. Braun, and T. Korakis, "Network Store: Exploring Slicing in Future 5G Networks," ser. MobiArch '15. ACM.

[12] S. Vassilaras, L. Gkatzikis, N. Liakopoulos, I. N. Stiakogiannakis, M. Qi, L. Shi, L. Liu, M. Debbah, and G. S. Paschos, "The algorithmic aspects of network slicing," *IEEE Communications Magazine*, 2017.

[13] G. Barlacchi et al., "A Multi-source Dataset of Urban Life in the City of Milan and the Province of Trentino," 2015. [Online]. Available: http://dx.doi.org/10.1038/sdata.2015.55

[14] F. Xu, Y. Li, H. Wang, P. Zhang, and D. Jin, "Understanding Mobile Traffic Patterns of Large Scale Cellular Towers in Urban Environment," *IEEE/ACM Transactions on Networking*, vol. PP, no. 99, pp. 1–15, 2016.

[15] D. Naboulsi, M. Fiore, and R. Stanica, "Human Mobility Flows in the City of Abidjan," 2013. [Online]. Available: https://hal.inria.fr/hal-00908277

[16] N. Liakopoulos, G. Paschos, and T. Spyropoulos, "Robust User Association for Ultra Dense Networks," in *IEEE INFOCOM*, April 2018.

[17] G. Paschos, N. Liakopoulos, M. Debbah, and T. Wen, "Computational Optimal Transport for 5G Massive C-RAN Device Association," in *2018 IEEE Global Communications Conference (GLOBECOM)*, Dec 2018.

[18] N. Liakopoulos, G. Paschos, and T. Spyropoulos, "User Association for Ultra Dense Networks with QoS Guarantees," in *2018 IEEE Global Communications Conference (GLOBECOM)*, Dec 2018.

[19] H. Kim, G. de Veciana, X. Yang, and M. Venkatachalam, "Distributed alpha-Optimal User Association and Cell Load Balancing in Wireless Networks," *IEEE/ACM Transactions on Networking*, Feb 2012.

[20] N. Sapountzis, T. Spyropoulos, N. Nikaein, and U. Salim, "An Analytical Framework for Optimal Downlink-Uplink User Association in HetNets with Traffic Differentiation," in *IEEE GLOBECOM*, 2015.

[21] ——, "Optimal Downlink and Uplink User Association in Backhaul-limited HetNets," in *IEEE INFOCOM*, 2016.

[22] Q. Ye, B. Rong, Y. Chen, M. Al-Shalash, C. Caramanis, and J. G. Andrews, “User Association for Load Balancing in Heterogeneous Cellular Networks,” *IEEE Transactions on Wireless Communications*, vol. 12, no. 6, pp. 2706–2716, June 2013.

[23] D. Fooladivanda and C. Rosenberg, “Joint Resource Allocation and User Association for Heterogeneous Wireless Cellular Networks,” *IEEE Transactions on Wireless Communications*, January 2013.

[24] T. Bu, L. Li, and R. Ramjee, “Generalized Proportional Fair Scheduling in Third Generation Wireless Data Networks,” in *Proceedings IEEE INFOCOM*, April 2006, pp. 1–12.

[25] R. B. Q. Zhang, M. F. Zhani and J. L. Hellerstein, “Dynamic Heterogeneity-Aware Resource Provisioning in the Cloud,” *IEEE Transactions on Cloud Computing*, Jan 2014.

[26] “How AWS Pricing Works,” Amazon, 2018. [Online]. Available: https://aws.amazon.com/whitepapers/

[27] N. C. Petruzzi and M. Dada, “Pricing and the Newsvendor Problem: A Review with Extensions,” *Operations research*, 1999.

[28] G. Peyré and M. Cuturi, “Computational optimal transport,” *Foundations and Trends® in Machine Learning*, vol. 11, 2019. [Online]. Available: http://dx.doi.org/10.1561/2200000073

[29] L. Kantorovich, “On the Transfer of Masses,” 1942.

[30] G. Monge, “Mémoire sur la Théorie des Déblais et des Remblais,” 1781.

[31] D. Bertsimas, D. B. Brown, and C. Caramanis, “Theory and Applications of Robust Optimization,” *SIAM review*, vol. 53, no. 3, pp. 464–501, 2011.

[32] J. Mo and J. Walrand, “Fair End-to-End Window-based Congestion Control,” *IEEE/ACM Transactions Networking*.

[33] Telecom Italia, “Telecommunications - SMS, Call, Internet - MI,” 2015. [Online]. Available: http://dx.doi.org/10.7910/DVN/EGZHFV

[34] P. J. Brockwell and R. A. Davis, *Introduction to Time Series and Forecasting*. Springer, 2016.

[35] F. A. Gers, J. Schmidhuber, and F. Cummins, “Learning to Forget: Continual Prediction with LSTM,” 1999.

[36] N. Liakopoulos, G. Paschos, and T. Spyropoulos, “No Regret in Cloud Resources Reservation with Violation Guarantees,” in *IEEE INFOCOM*, May 2019.

[37] S. Mannor, J. N. Tsitsiklis, and J. Y. Yu, “Online Learning with Sample Path Constraints,” *Journal of Machine Learning Research*, 2009.

[38] M. J. Neely, “Stochastic Network Optimization with Application to Communication and Queueing Systems,” *Synthesis Lectures on Communication Networks*, 2010.

[39] M. Zinkevich, “Online Convex Programming and Generalized Infinitesimal Gradient Ascent,” ser. ICML, 2003.

[40] M. J. Neely and H. Yu, “Online Convex Optimization with Time-Varying Constraints,” *arXiv preprint arXiv:1702.04783*, 2017.

[41] S. Vassilaras, L. Vigneri, N. Liakopoulos, G. S. Paschos, A. Destounis, T. Spyropoulos, and M. Debbah, “Problem-Adapted Artificial Intelligence for Online Network Optimization,” 2018.

[42] S. Sesia, I. Toufik, and M. Baker, “LTE-the UMTS Long Term Evolution: From Theory to Practise,” 2015.

[43] K. Samdanis, X. Costa-Perez, and V. Sciancalepore, “From Network Sharing to Multi-Tenancy: The 5G Network Slice Broker,” *IEEE Communications Magazine*, July 2016.

[44] T. Bonald and A. Proutière, “Wireless Downlink Data Channels: User Performance and Cell Dimensioning,” in *MobiCom 2003*. ACM.

[45] A. Destounis, G. S. Paschos, J. Arnau, and M. Kountouris, “Scheduling URLLC Users with Reliable Latency Guarantees,” *WiOpt*, May 2018.

[46] E. Altman, K. Avrachenkov, and U. Ayesta, “A Survey on Discriminatory Processor Sharing,” *Queueing Systems*, 2006.

[47] K. Son, H. Kim, Y. Yi, and B. Krishnamachari, “Toward Energy-Efficient Operation of Base Stations in Cellular Wireless Networks,” *Green Communications: Theoretical Fundamentals, Algorithms, and Applications*, 2012.

[48] P. Caballero, A. Banchs, G. de Veciana, and X. Costa-Pérez, “Multi-tenant radio access network slicing: Statistical multiplexing of spatial loads,” *IEEE/ACM Transactions on Networking*, Oct 2017.

[49] K. Shen and W. Yu, “Distributed Pricing-Based User Association for Downlink Heterogeneous Cellular Networks,” *CoRR*, 2014.

[50] “eLTE2.3 DBS3900 LTE TDD Basic Feature Description,” Huawei Technologies Co. Ltd., 2014. [Online]. Available: http://www.huawei.com/ilink/cnenterprise/download/HW_328213

[51] M. Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*, 1st ed. New York, NY, USA: Cambridge University Press, 2013.

[52] S. Boyd and L. Vandenberghe, *Convex Optimization.* New York, NY, USA: Cambridge University Press, 2004.

[53] D. P. Bertsekas, *Nonlinear Programming.* Athena Scientific, 1995.

[54] M. Cuturi, "Sinkhorn Distances: Lightspeed Computation of Optimal Transport," ser. NeurIPS, 2013.

[55] "The C-RAN (Centralized Radio Access Network) Ecosystem: 2017 - 2030 - Opportunities, Challenges, Strategies & Forecasts," SNS Research Report, July 2017.

[56] A. Checko, H. Christiansen, Y. Yan, L. Scolari, G. Kardaras, M. Berger, and L. Dittmann, "Cloud RAN for Mobile Networks–A Technology Overview," *IEEE Communications Surveys & Tutorials*, 2015.

[57] Gartner Newsroom, February 2017. [Online]. Available: http://www.gartner.com/newsroom/id/3598917

[58] Y. L. Lee, L. Wang, T. C. Chuah, and J. Loo, "Joint Resource Allocation and User Association for Heterogeneous Cloud Radio Access Networks," in *ITC 28*, Sep. 2016.

[59] G. Athanasiou, P. C. Weeraddana, C. Fischione, and L. Tassiulas, "Optimizing Client Association for Load Balancing and Fairness in Millimeter-Wave Wireless Networks," *IEEE/ACM Transactions on Networking*, June 2015.

[60] M. Awais, A. Ahmed, M. Naeem, M. Iqbal, W. Ejaz, A. Anpalagan, and H. S. Kim, "Efficient Joint User Association and Resource Allocation for Cloud Radio Access Networks," *IEEE Access*, 2017.

[61] R. Irmer, H. Droste, P. Marsch, M. Grieger, G. Fettweis, S. Brueck, H. Mayer, L. Thiele, and V. Jungnickel, "Coordinated Multipoint: Concepts, Performance, and Field Trial Results," *IEEE Communications Magazine*, February 2011.

[62] J. B. Orlin, S. A. Plotkin, and É. Tardos, "Polynomial Dual Network Simplex Algorithms," *Mathematical Programming*, Jun 1993.

[63] J. B. Orlin, "A polynomial time primal network simplex algorithm for minimum cost flows," *Mathematical Programming*, Aug 1997.

[64] R. E. Tarjan, "Dynamic Trees as Search Trees via Euler Tours, Applied to the Network Simplex Algorithm," *Mathematical Programming*, Aug 1997.

[65] J. Benamou, G. Carlier, M. Cuturi, L. Nenna, and G. Peyré, "Iterative Bregman Projections for Regularized Transportation Problems," *SIAM Journal on Scientific Computing*, 2015.

[66] L. Bregman, "The Relaxation Method of Finding the Common Point of Convex Sets and its Application to the Solution of Problems in Convex Programming," 1967.

[67] H. H. Bauschke and A. S. Lewis, "Dykstra's Algorithm with Bregman Projections: A Convergence Proof," *Optimization*, 2000.

[68] R. Sinkhorn, "A Relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices," *The Annals of Mathematical Statistics*, June 1964.

[69] J. Altschuler, J. Weed, and P. Rigollet, "Near-Linear Time Approximation Algorithms for Optimal Transport via Sinkhorn Iteration," in *Proceedings NeurIPS*, 2017.

[70] "GLPK (GNU linear programming kit)," 2006. [Online]. Available: http://www.gnu.org/software/glpk

[71] D. Bertsimas, V. Gupta, and N. Kallus, "Data-driven Robust Optimization," *Mathematical Programming*, Feb 2017.

[72] C. p. Li, G. S. Paschos, L. Tassiulas, and E. Modiano, "Dynamic Overload Balancing in Server Farms," in *IFIP Networking Conference*, June 2014, pp. 1–9.

[73] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust Optimization.* Princeton University Press, 2009.

[74] Y. Chen and X. Ye, "Projection Onto A Simplex," 2011.

[75] W. Wang and M. Á. Carreira-Perpiñán, "Projection onto the Probability Simplex: An Efficient Algorithm with a Simple Proof, and an Application," 2013. [Online]. Available: http://arxiv.org/abs/1309.1541

[76] S. Bubeck *et al.*, "Convex Optimization: Algorithms and Complexity," *Foundations and Trends in Machine Learning*, 2015.

[77] A. Nedić and A. Ozdaglar, "Approximate Primal Solutions and Rate Analysis for Dual Subgradient Methods," *SIAM Journal on Optimization*, vol. 19, no. 4, pp. 1757–1780, 2009.

[78] C. p. Li and M. J. Neely, "Delay and Rate-Optimal Control in a Multi-Class Priority Queue with Adjustable Service Rates," in *Proceedings IEEE INFOCOM*, March 2012.

[79] S. Shalev-Shwartz, "Online Learning and Online Convex Optimization," *Foundations and Trends® in Machine Learning*, 2012.

[80] E. V. Belmega, P. Mertikopoulos, R. Negrel, and L. Sanguinetti, "Online Convex Optimization and No-Regret Learning: Algorithms, Guarantees and Applications," 2018. [Online]. Available: http://arxiv.org/abs/1804.04529

[81] M. Mahdavi, R. Jin, and T. Yang, "Trading Regret for Efficiency: Online Convex Optimization with Long Term Constraints," *Journal of Machine Learning Research*, 2012.

[82] R. Jenatton, J. C. Huang, and C. Archambeau, "Adaptive Algorithms for Online Convex Optimization with Long-Term Constraints," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, 2016.

[83] J. D. Abernethy, E. Hazan, and A. Rakhlin, "Interior-Point Methods for Full-Information and Bandit Online Learning," *IEEE Transactions on Information Theory*, 2012.

[84] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource Allocation and Cross-Layer Control in Wireless Networks," *Foundations and Trends® in Networking*, vol. 1, 2006. [Online]. Available: http://dx.doi.org/10.1561/1300000001

[85] N. Parikh, S. Boyd *et al.*, "Proximal Algorithms," *Foundations and Trends® in Optimization*, 2014.

[86] J. Wilkes, "More Google Cluster Data," Google Research Blog, Nov. 2011.

[87] "Truth in advertising: "Click fraud poses a threat to the boom in Internet advertising"," 2005.

[88] J. Yuan and A. Lamperski, "Online convex optimization for cumulative constraints," ser. NeurIPS, 2018.

[89] H. Yu, M. J. Neely, and X. Wei, "Online Convex Optimization with Stochastic Constraints," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017.

[90] W. Sun, D. Dey, and A. Kapoor, "Safety-Aware Algorithms for Adversarial Contextual Bandit," in *ICML*, Sydney, Australia, August 2017, pp. 3280–3288.

[91] J. Abernethy, P. L. Bartlett, A. Rakhlin, and A. Tewari, "Optimal Strategies and Minimax Lower Bounds for Online Convex Games," 2008.

[92] T. Chen, Q. Ling, and G. B. Giannakis, "An online convex optimization approach to proactive network resource allocation," *IEEE Transactions on Signal Processing*, Dec 2017.

[93] X. Cao and K. J. R. Liu, "Online convex optimization with time-varying constraints and bandit feedback," *IEEE Transactions on Automatic Control*, 2018.

[94] O. Besbes, Y. Gur, and A. Zeevi, "Non-stationary stochastic optimization," *Operations research*, vol. 63, no. 5, pp. 1227–1244, 2015.

[95] M. J. Neely, "Universal scheduling for networks with arbitrary traffic, channels, and mobility," in *Decision and Control (CDC), 2010 49th IEEE Conference on.* IEEE, 2010, pp. 1822–1829.

[96] N. Liakopoulos, G. Paschos, and T. Spyropoulos, "Robust Optimization Framework for Proactive User Association in UDNs: A Data-Driven Approach," in *IEEE Transactions on Networking*, August 2019.