Computational Complexity of the Pilot Assignment problem in Cell-Free Massive MIMO

A Thesis

submitted to
Indian Institute of Science Education and Research Pune
in partial fulfillment of the requirements for the
BS-MS Dual Degree Programme

by

Shruthi Prusty

Indian Institute of Science Education and Research Pune
Dr. Homi Bhabha Road,
Pashan, Pune 411008, INDIA.

May, 2023

Supervisor: Dr. Sofiat Olaosebikan

Certificate

This is to certify that this dissertation entitled *Computational Complexity of the Pilot Assignment problem in Cell-Free Massive MIMO* towards the partial fulfilment of the BS-MS dual degree programme at the Indian Institute of Science Education and Research, Pune represents study/work carried out by Shruthi Prusty at the School of Computing Science, University of Glasgow, UK, under the supervision of Dr. Sofiat Olaosebikan, Lecturer, School of Computing Science, University of Glasgow, UK, during the academic year 2022-2023.

Dr. Sofiat Olaosebikan

Committee:

Dr. Sofiat Olaosebikan

Dr. Soumen Maity

"Education never ends, Watson. It is a series of lessons, with the greatest for the last.

(...)

For the world is full of obvious things which nobody by any chance ever observes."

— Sir Arthur Conan Doyle, *His Last Bow*
and *The Hound of the Baskervilles*

This thesis is dedicated to my parents, and all those who taught me to love Mathematics at IISER Pune.

Declaration

I hereby declare that the matter embodied in the report entitled *Computational Complexity of the Pilot Assignment problem in Cell-Free Massive MIMO* are the results of the work carried out by me at the School of Computing Science, University of Glasgow, UK, under the supervision of Dr. Sofiat Olaosebikan and the same has not been submitted elsewhere for any other degree.

Shruthi Prusty

Shruthi Prusty

Acknowledgements

Like all academic endeavours, this thesis too, stands on the shoulders of giants. I am privileged and grateful to have had the help and support of various people, without which this thesis would have never seen the light of day.

First and foremost, I would like to express my deepest gratitude to my supervisor, Dr. Sofiat Olaosebikan, for agreeing to supervise my thesis and suggesting to work on such an exciting problem that lies at the intersection of mathematics and computer science research. Her brilliant guidance and support throughout this project, coupled with the immense freedom to pursue whatever mathematical direction I wished to, have made for an incredibly enriching experience. Her constant focus on the bigger picture has led this thesis to completion. She is the single most important person to have influenced my academic decisions during my MS and beyond. I'll always be grateful to her for keeping my passion for research alive.

I must express my gratitude to the University of Glasgow, the School of Computing Science, and the FATA research section in particular, for all the academic opportunities and privileges I received for being a part of them. I am glad for this exposure that has had a deep impact on my academic and mental growth. I am indebted to Prof. David Manlove for his hospitality during my visit, and for his valuable suggestions and comments on my thesis draft. I am grateful to Peace Ayegba, whose research on cell-free massive MIMO made the task of understanding a problem from a previously unfamiliar field much easier. As a friend, she made it possible for me to feel at home in a new city like Glasgow in a short period of time. I also thank Dr. Michael McKay for suggesting approximation-preserving reductions as a direction to pursue in this thesis, and for keeping me company during an intensely stressful application period. All the conversations and banter with Mike, Douglas and Kai in G161, SAWB have made my experience in Glasgow indeed a fun and memorable one.

I am grateful to the Department of Mathematics at IISER Pune for playing such a crucial role in shaping my interests. Over the last 5 years, I have received a lot of help and encouragement from various people in the department. I am grateful to Dr. Soumen Maity for introducing me to research in algorithms, and for his interest and support during this project. I must thank Dr. Chandrasheel Bhagwat for playing a vital role in my study of mathematics at IISER, through courses and discussions that spanned academics and beyond.

My maths friends and classmates have been an important part of my journey at IISER. Working on this thesis would have been much more exhausting without the encouragement from Ajay and Anirudh. I am grateful to Ipsa, Mihir, Karthik and Aravint for keeping me company in many of my maths classes in the last three years.

Working on this thesis required more than academic support. I couldn't thank my "Maula" group of friends enough – Adithyan, Jatin, Chinmay, Shivani, Siddharth and Suryo – for always listening to me and tolerating me. Navigating application season would have been incredibly difficult if not for them. Srishti is responsible for teaching me how to explore the UK. During the pandemic, I made several friends whose presence was invaluable for my mental health. I'm glad to have known my "Discord gang" (you all know who you are), who became my home away from home despite being in junior batches. I am grateful to Pooja, Krishna and Kashika for letting me show up to their rooms and listening to my rants about everything in the initial days of this thesis. I am indebted to Jason for constantly being by my side, checking up on me even when a reply was unlikely, and spamming me with content that can perhaps only entertain the two of us. For all the strength you found in me, I hope you'll be glad to know that the favour was more than returned in kind. Arjun and Gaurav have accompanied me in a lot of my classes at IISER, and have played an important role in helping me build mathematical confidence and emotional maturity over the years, especially during the pandemic, that culminated in this project. Shikhara, who has known me the longest at IISER, has helped me grow as a person and shared his love for animals with me. Thank you for all the cat-related mental health boosts on a daily basis during the pandemic and thesis years.

I am incredibly fortunate to have had the companionship of Reetish ever since I decided to work on this project and move to Glasgow. He has been my pillar of strength, sustaining me through this thesis year with his tireless care and support. Unconditionally giving me all the emotional strength I could have asked for, and more. He has been my go-to person, making life exciting with his fun presence and long conversations. Without him, this thesis would have been incomplete. You have my unwavering gratitude and respect.

Finally, I'm eternally grateful to my parents for their financial and mental support. For having faith in me, and believing that I'll figure things out on my own. For all their encouragement, patience and love. For all the sacrifices that I can never repay them for. This project would have never happened without them. Thank you for everything.

Abstract

Wireless communication is of paramount importance in today's world, enabling billions of people to connect to each other and the internet, transforming every sector of the economy, and building the foundations for powerful new technologies that hold great promise to improve lives at an unprecedented rate and scale. The rapid increase in the number of devices and the associated demands for higher data rates and broader network coverage fuels the need for more robust wireless technologies. The key technology identified to address this problem is called Cell-Free Massive MIMO (CF-mMIMO).

CF-mMIMO is accompanied by many challenges, one of which is how to efficiently manage limited resources, giving rise to the resource allocation problem in wireless networks. In this thesis, we focus on a major problem that hinders resource allocation in wireless networks, namely the Pilot Assignment problem (PA). While this is a problem that is the focus of active research in engineering, it has received little attention from theoretical computer scientists, despite having several graph-theoretic schemes to tackle it. In the course of this thesis, we delve into the theory of computational complexity and approximability, and establish results pertaining to these concepts for the Pilot Assignment problem with the help of original reductions. We further discuss our findings in the context of the algorithms already proposed in the literature to tackle this problem.

Contents

Introduction

0.1 Background: Cell-Free Massive MIMO

Wireless networks is an essential technology for enabling flexible communication and connectivity between individuals (or machines) across regions. In addition, it is transforming every sector of the economy (transportation, healthcare, education, etc.), and powerful new technologies (artificial intelligence, internet of things, etc.) are being built upon it. Cellular networks is the technology that 1G to 5G relies on [20, 32]. They are designed primarily for voice and data transmissions, and the overall network performance is measured by the spectral efficiency (SE) or the uplink/downlink throughput attained by every user in the network. Here, the coverage area is divided up into non-overlapping cells and we have a single AP coordinating data transmissions amongst user devices within its cell. The entire spectrum (used for wireless communication) is divided up into channels and a set of channels is often assigned to each cell such that neighbouring cells don't share the same channels. Figure 1 depicts a typical cellular network.

However, as the number of devices that depend on wireless communication networks continues to grow, each needing a high connection rate and better coverage with minimal interference, this technology will no longer be suitable [24, 32]. Particularly, resources (e.g. channels, power, spectrum) in a wireless network are limited and must be managed efficiently. Further, the User Equipments (UEs) at the edge of the cells get poor service due to inter-cell interference, as the channels of UEs in the cells neighbouring such UEs are in close proximity, and at the same time, are often not orthogonal (distinct). For the same reason, we also see interference at the APs due to signals received from UEs in cells other than the cell where the UE under consideration is located. These interferences are depicted in Figure 2. For future wireless communications (e.g., 6G), the key technology that has the potential to enhance connectivity and provide better coverage for billions of users is referred

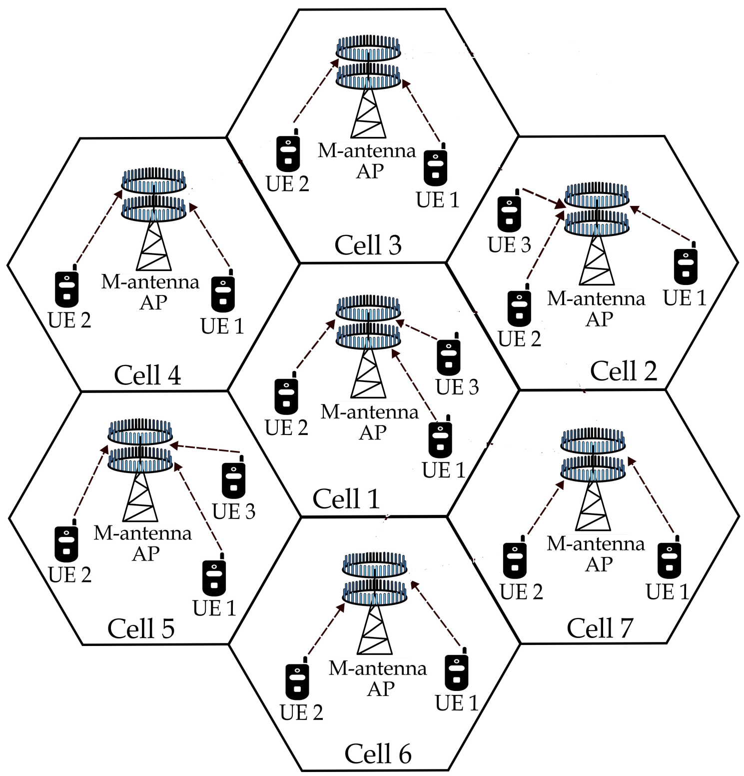


Figure 1: Schematic diagram of Cellular network architecture

to as Cell-Free Massive Multiple-Input Multiple-Output (CF-mMIMO).

As the name suggests, CF-mMIMO allows a device/user (UE) to be served by multiple access points (APs) that are within its range without the notion of boundaries; in contrast to the current technology, which allows each UE to be served by only one AP within a defined boundary. Essentially, all the UEs can be served by all the APs in the network. All APs are connected to the CPU through a backhaul link and perform signal processing in the CPU. The CPU is responsible for coordinating various resource allocations. Due to the user-centric (UC) cooperation relationship between APs, the cell boundary is eliminated to a certain extent, which limits the effects of cell interference. Moreover, the distance between the APs and the user is shorter than that in a traditional cellular network, which leads to lower path loss and higher diversity gain [6, 22]. Refer to Figure 3 for a schematic of a cell-free network architecture. The goal of this network is to reduce inter-cell interferences, improve the uniform distribution of spectral efficiency amongst users and enhance network reliability [24]. CF-mMIMO is accompanied by many challenges, one of which is how to efficiently manage limited resources (spectrum, pilot signals, energy, and power) – the so-

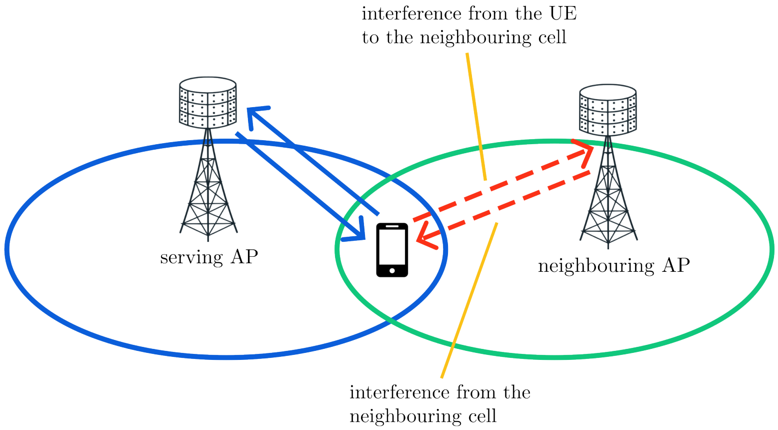


Figure 2: Schematic representation of inter-cell interference

called resource allocation problem in wireless networks.

For the sake of this thesis, we will consider a CF-mMIMO system with M single-antenna APs and K ($K \ll M$) UEs where the APs are randomly distributed in a large area. At first glance, the condition that the number of APs is far greater than the number of users might appear strange, given our current understanding of APs as transmission towers. However, the idea for 6G is that these APs will now be small enough to be on "strips" that shall be pasted all over the walls of rooms inhabited by users. This also gives us an understanding of why the distance between the APs and the users is shorter than before. Since a large number of distributed APs jointly provide uniform service to a small number of UEs using the same time-frequency resource without any notion of boundaries, it is often the case that *AP selection* is done for each user, such that only a subset of the APs providing service above a certain threshold are considered for any energy or spectral efficiency calculations pertaining to that user [31, 23]. This is depicted in Figure 4. We can ignore the contribution of the APs which are far from the user in the data transmission. Therefore, it is not necessary to consider all APs providing service to a certain user. AP selection can reduce interference between users and increase the achievable rate per user by improving system Energy Efficiency (EE) and reducing backhaul load [22]. An important assumption made in this thesis is that AP selection is always done for any CF-mMIMO system we consider. We now turn our attention towards a major problem that hinders resource allocation in wireless communication networks, and is a resource allocation problem itself: the Pilot Assignment problem.

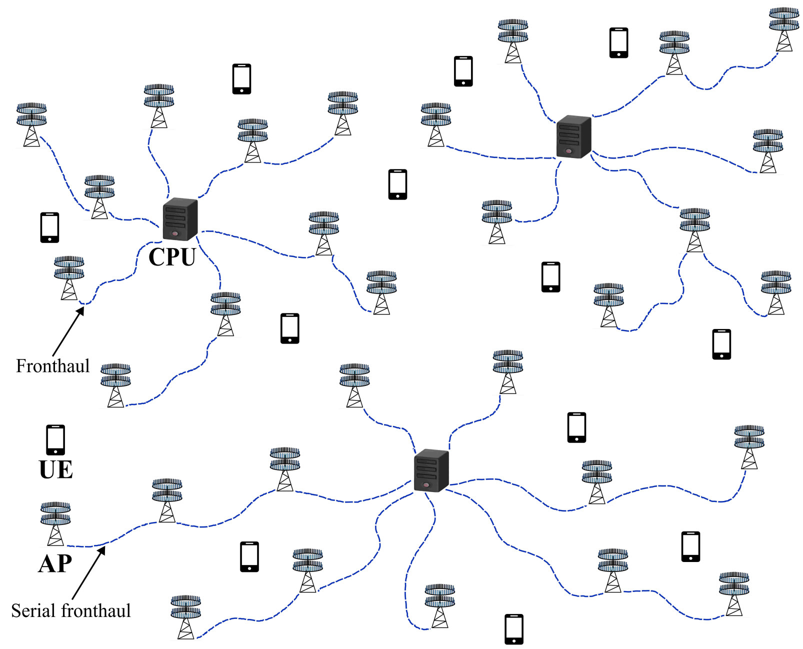


Figure 3: Schematic diagram of Cell-free network architecture

0.2 Introduction to the Pilot Assignment problem

0.2.1 Description of the problem

It is essential to acquire accurate *Channel State Information (CSI)* between the UEs and the APs to reap all the benefits potentially provided by the distributed user-centric, cell-free, massive MIMO architecture [7]. Channel estimation allows APs to process data signals from the UEs, which in turn facilitates the decoding and encoding of data signals at the CPU. To perform channel estimation for a user, a *pilot signal* needs to be assigned to it. In the system setting, it is assumed that APs and UEs do not have *a priori* CSI at the beginning of a coherent interval. The CSI is estimated in what is called a *pilot training phase*, which usually happens in the uplink. All communications are carried out on the same frequency band and work in time division duplex (TDD) mode. Due to the channel reciprocity of the TDD system, only uplink channel estimation is performed [22]. Thus uplink training or uplink pilot transmission is necessary and sufficient for both uplink and downlink data transmission between UEs and APs. So, channel estimation is needed only at the beginning of a coherent interval τ_c. Thus, τ pilot sequences (or signals) of length τ each are assigned to the UEs prior to uplink data transmission for channel estimation. Each UE is assigned one pilot. The received pilot information is used for channel estimation. The estimated channel

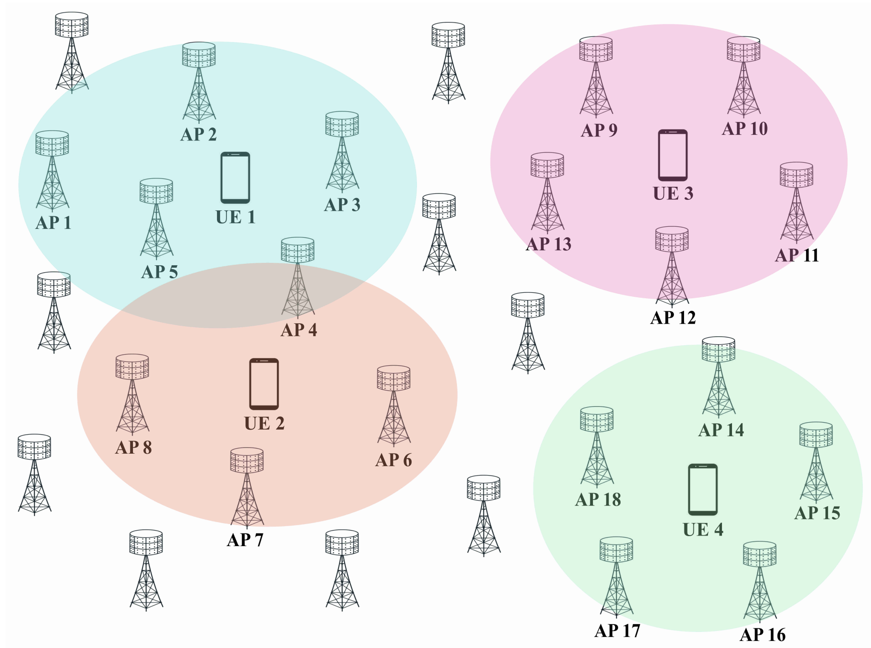


Figure 4: AP selection in cell-free massive mMIMO system

is used to detect the received data, thereby allowing us to calculate the Spectral Efficiency (SE) of each UE [11]. However, due to the limited length of the coherence interval, the available number of orthogonal pilot sequences is normally smaller than the number of UEs ($\tau \ll K$) and some UEs have to reuse a given pilot. Hence, the orthogonality among the pilot sequences for all UEs is typically not achieved. The pilot reuse causes an impairment known as pilot contamination, which can degrade the system performance, by lowering the achievable uplink/downlink rates and signal-to-interference-plus-noise ratio (SINR) of the system [31, 23]. This phenomenon is depicted in Figure 5.

0.2.2 Existing work, Objective and Scope of the thesis

The Pilot Assignment PA problem has been extensively researched from the engineers' perspective to find heuristics-based algorithms that return efficient solutions for this distributed architecture. The most straightforward and naive pilot allocation strategy is the *random* pilot assignment scheme, in which the available pilots are assigned randomly to each user [24]. Of course, this does not address any of the problems, including two nearby users sharing the same pilot, and thus turns out to be the worst scheme. The *greedy* pilot assignment scheme proposed in [24] works by iteratively improving the downlink rate for the worst user. However, such a method can only improve the worst user's performance at any

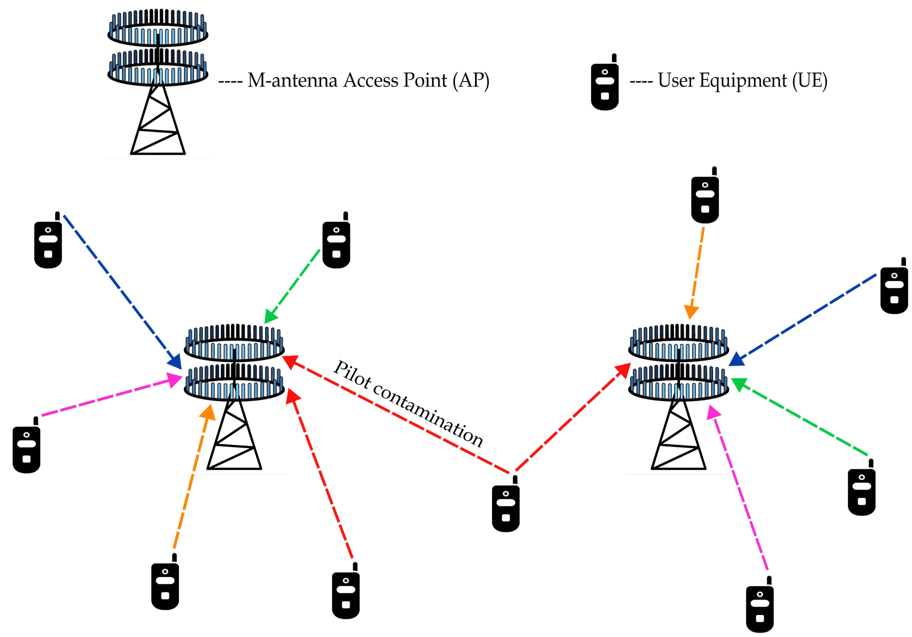


Figure 5: Pilot contamination in cell-free architecture

given point, and cannot guarantee an improvement in the whole system's performance. This means that this algorithm is not guaranteed to converge stably to a global maximum value. The *location-based greedy* pilot assignment scheme utilizes the location information of the users for the initial assignment of pilots in the greedy scheme, instead of random assignment. This method, however, does not prove to be very effective in practice and only promotes the throughput performances of a few users [33, 23]. The *structured* pilot assignment scheme maximizes the minimum distance among UEs that share the same pilot using a clustering algorithm, but the implementation in a real-world cell-free massive MIMO system is hindered by the difficulty of finding the centroid APs in such practical systems [2, 23]. The *Maximal increment* (MI) algorithm maximizes the achievable (downlink) rate by maximizing the increment in an iterative algorithm, but it has high time complexity [30]. The *Hungarian Algorithm* based pilot assignment scheme is an iterative procedure based on the Hungarian algorithm to enhance system throughput and fairness by avoiding pilot reuse among nearby users. In fact, it has been a common theme to design algorithms by avoiding pilot reuse between nearby users. However, it has been observed that it is not sufficiently accurate to measure pilot contamination solely based on the geographical distance between the users [6, 31].

Recently, two graph-based pilot assignment schemes have been used: The *graph-colouring* based pilot assignment scheme and the *Max-k-Cut* based pilot assignment scheme in a weighted graphic framework. In both cases, the CF-mMIMO architecture is visualized as a graph, with the UEs forming the vertices and the edges depicting the interference between the UEs. The pilot assignment optimization is then solved on this graph. Since both the graph-colouring and the Max-k-Cut problems fall in the class of NP-hard problems, heuristic graph algorithms have been employed to find reasonable solutions to the PA problem, supported by experimental results obtained via simulations, where they outperform other non-graph theoretic algorithms with reasonable time complexity. Further, the Max-k-Cut scheme even outperforms the graph-colouring scheme [31, 23], making it the state of the art. This leads us to believe that there must be some sort of natural relation between the pilot assignment optimization problem, and classical graph-theoretic optimization problems. The experimental results seen so far aim to optimize a certain Quality of Service (QoS) metric that, such as the system or sum-user SE, the system throughput and the system Energy Efficiency (EE), all of which can be expressed via the uplink/downlink achievable rates, which further depend on the Signal-to-Interference-plus-Noise Ratio (SINR) [7, 22, 6, 23, 31].

Despite all the research effort that has been put into solving the pilot contamination problem in cell-free massive MIMO systems, the pilot assignment problem has received very little attention from a theoretical computer science perspective. In particular, an explicit proof of the computational hardness of this problem is yet to be seen. Further, assuming the problem is indeed hard to solve, which justifies the various heuristics algorithms being employed to arrive at a reasonably close-to-optimal solution, there are no theoretical guarantees on how well one can approximate the optimal solution in polynomial time. This thesis aims to answer the above questions. Modification of existing heuristics for PA, and the design or implementation of better heuristics based on the graph problems we study to prove theoretical results for the PA problem are out of the scope of this thesis. We also limit ourselves to achieving high spectral efficiency or system throughput, and do not take into consideration the trade-off that exists between the energy efficiency and the spectral efficiency of the system [22].

0.3 Original Contributions

All the definitions and results pertaining to the Pilot Assignment problem in Chapter 2 are original. While many of the ideas are heavily borrowed from [12], the main proof in the chapter, the NP-hardness of the Pilot Assignment problem, is completely original. The approximability results for the Pilot Assignment problem in Chapter 3 are also original and are inspired by the results in [12]. This thesis refers to the textbook [3] for all the definitions and results on complexity theory in Chapter 1 and approximation theory in Chapter 3.

The big picture here is that the thesis provides greater insights into the very structure of the Pilot Assignment problem on a graph. As noted in Subsection 0.2.2, the MAX-k-CUT problem has been used to develop the current state of the art for pilot assignment. Our contribution here is to focus on its dual problem, the MIN-k-PARTITION, to prove theoretical approximation bounds for the Pilot Assignment problem. With a clear distinction made between the approximability of the MAX-k-CUT and the MIN-k-PARTITION problems, this thesis is able to shed light on the classical graph problem that is perhaps the best-suited for future work on heuristic algorithms for PA.

Chapter 1

Preliminaries

The goal of this chapter is to introduce the basic concepts related to the complexity of optimization problems. Since the major concepts in complexity theory are defined in terms of decision problems, we will first discuss them by stating the main definitions and results concerning decision problems. In particular, we will introduce the two complexity classes P and NP, and we will briefly discuss their relationship by introducing the concepts of polynomial-time reductions, complexity class closure, and hard and complete problems. Later, we will define the same concepts and notions in the context of optimization problems. These definitions and results can be found in Chapter 1 of [3]. Finally, we introduce the concept of *strong* NP-hardness for problems in both NP and NPO.

1.1 Complexity of decision problems

Definition 1.1.1 (Decision problem). *A* ***decision problem*** *is a tuple (I, SOL) where I is the set of instances and $SOL : I \to \{0, 1\}$ associates with every instance x either zero or one. The problem P is identified with the language $L_P = \{x \in I \mid SOL(x) = 1\}$. The problem asks, for any instance $x \in I$, to verify whether $x \in L_P$, also known as verifying if x is a "yes"-instance of P, and if the language L_P is recognizable.*

Definition 1.1.2 (The class P). *The class of decision problems that can be solved in polynomial time. In other words, it is the set of all such decision problems P for which there exists a polynomial-time algorithm $\mathcal{A}$ such that, for a given instance $x \in I_P$, $\mathcal{A}$ returns* YES *if and only if $x \in L_P$. We also say that the language L_P is recognized by $\mathcal{A}$.*

In the above definitions, given an instance x of a problem P, the corresponding solution is a binary value that indicates whether the instance is a "yes" instance. In most cases, determining this also involves finding an object $y(x)$, which could be a string, set, array, graph, etc., whose characteristics are stated in the problem instance. The decision problem asks for the existence of such an object. We shall call this object $y(x)$ a *solution* of P.

Definition 1.1.3 (**The class NP**). *The class of decision problems whose solutions can be verified in polynomial time. In other words, it is the set of all such decision problems for which there exists a polynomial-time algorithm* (*known as an efficient certifier*) *that takes as input an instance of the problem, and a certificate for the problem instance* (*whose length is polynomial in the size of the problem instance*)*, and verifies the certificate.*

A problem $P \in \mathrm{NP}$ if there exists an efficient verifier B such that for a given instance x of P, x is a "yes"-instance of P if and only if there exists a certificate $t(x)$, which depends on x, such that $B(s,t) = \text{YES}$.

Remark 1.1.1. While it is easy to see that $\mathrm{P} \subseteq \mathrm{NP}$, the question of whether $\mathrm{P} = \mathrm{NP}$ is fundamental in the area of algorithms, and is one of the most famous problems in computer science. It is widely believed that $\mathrm{P} \neq \mathrm{NP}$, and this is taken as a working hypothesis throughout the field, although there is little technical evidence for it.

1.1.1 Polynomial-time reducibility among problems

A **reduction** from a problem P_1 to a problem P_2 presents a method to solve P_1 using an algorithm for P_2. This means that P_2 is at least as difficult as P_1, as long as the reduction involves "simple enough" steps. In this section, let I_P be the set of problem instances and S_P be the set of problem solutions for any problem P.

Definition 1.1.4 (**Polynomial time many-one reducibility or Karp reducibility**). *A polynomial-time many-one reduction or a* ***Karp reduction*** *from a decision problem P_1 to a decision problem P_2 is a polynomial-time algorithm that transforms a given instance $x \in I_{P_1}$ of P_1 to an instance $y \in I_{P_2}$ of P_2 in such a way that $x \in L_{P_1}$ if and only if $y \in L_{P_2}$. A reduction of this type is denoted by $P_1 \leq_m^p P_2$. If both $P_1 \leq_m^p P_2$ and $P_2 \leq_m^p P_1$ we say that P_1 and P_2 are Karp-equivalent, and we write $P_1 \equiv_m^p P_2$.*

A more general form of Karp reducibility is defined as follows:

Definition 1.1.5 (**Oracle**). *Let P be the problem of computing a* (*possibly multivalued*) *function $f : I_P \to S_P$. An **oracle** for problem P is an abstract device or a subroutine which, for any $x \in I_P$, returns a value $f(x) \in S_P$. It is assumed that the oracle may return the value in just one computation step.*

Definition 1.1.6 (**Polynomial time Turing reducibility or Cook reducibility**). *Let P_1 be the problem of computing a (possibly multivalued) function $f : I_{P_1} \to S_{P_1}$. A polynomial-time **Turing reduction** or a Cook reduction from problem P_1 to a problem P_2 (defined similarly as above) is an algorithm that solves P_1 using a polynomial number of queries to an oracle for P_2, and polynomial time outside of those queries. A reduction of this type is denoted by $P_1 \leq_T^p P_2$. If both $P_1 \leq_T^p P_2$ and $P_2 \leq_T^p P_1$ we say that P_1 and P_2 are Turing-equivalent, and we write $P_1 \equiv_T^p P_2$.*

It is clear that Karp-reducibility is a special case of Turing-reducibility, where the problems P_1 and P_2 are decision problems, the oracle is queried just once for P_2, and the value returned by the reduction is the same value as the one returned by the oracle.

Definition 1.1.7 (**Complexity Class Closure**). *A complexity class C is said to be **closed** with respect to a reducibility $\leq_r$ if, for any pair of decision problems P_1, P_2 such that $P_1 \leq_r P_2$, $P_2 \in C \Rightarrow P_1 \in C$.*

Definition 1.1.8 (**Complete and Hard problems**). *For any complexity class C, a decision problem $P \in C$ is said to be **complete in C** (equivalently, **C-complete**) with respect to a reducibility $\leq_r$ if, for any other decision problem $P_1 \in C$, $P_1 \leq_r P$. When the condition $P \in C$ is relaxed, the problem is said to be **C-hard**.*

We immediately note the following facts from the above definitions:

Remark 1.1.2.

1. For any two problems P_1 and P_2 which are C-complete with respect to a reducibility $\leq_r$, $P_1 \equiv_r P_2$.
2. For any pair of complexity classes C_1 and C_2 such that $C_1 \subset C_2$ and C_1 is closed with respect to a reducibility $\leq_r$, any C_2-complete problem P belongs to $C_2 \setminus C_1$.

We now state a few well-known results without proof:

Result 1.1.1.

1. *The complexity class NP is closed with respect to Karp reducibility, $\leq_m^p$.*

2. *Karp reductions have the transitive property, i.e., if $P_1 \leq_m^p P_2$ and $P_2 \leq_m^p P_3$, then $P_1 \leq_m^p P_3$.*

Definition 1.1.9 (NP-complete and NP-hard decision problems). *A decision problem P is said to be* NP-complete *if it is complete in* NP *with respect to $\leq_m^p$, i.e., $P \in$* NP *and, for any decision problem $P_1 \in$* NP, *$P_1 \leq_m^p P$. When the first condition of $P \in$* NP *is relaxed, P is said to be* NP-hard.

1.2 Complexity of optimization problems

1.2.1 Optimization problems

Definition 1.2.1 (Optimization problem). *An* ***optimization problem*** *P is a tuple $(I_P, SOL_P, m_P, \text{goal}_P)$ where:*

- *I_P is the set of instances of P,*
- *SOL_P is a function that associates to an instance $x \in I_P$, the set of feasible solutions of x,*
- *m_P is a measure function, defined for a pair (x, y) where $x \in I_P$ and $y \in SOL_P(x)$, that returns a positive rational which is the value of the feasible solution y.*
- $\text{goal}_P \in \{\text{MIN}, \text{MAX}\}$ *denotes whether P is a minimization or a maximization problem.*

Notation 1.2.1. Given an input instance x, we denote by $SOL_P^*(x)$ the set of optimal solutions of x. More formally, for every $y^*(x) \in SOL_P^*(x)$:

$$m_P(x, y^*(x)) = \text{goal}_P\{v \mid v = m_P(x, z) \wedge z \in SOL_P(x)\}.$$

The value of any optimal solution $y^*(x)$ of x is denoted as $m_P^*(x)$.

The definition of an optimization problem P naturally leads to the following three different problems, depending on what kind of solution is sought:

1. **Constructive Problem (P_C)** - Given an instance $x \in I$, find an optimal solution $y^*(x) \in SOL^*(x)$ and its measure $m^*(x)$.

2. **Evaluation Problem (P_E)** - Given an instance $x \in I$, compute the value $m^*(x)$.

3. **Decision Problem (P_D)** - Given an instance $x \in I$ and a positive rational $q \in \mathbb{Q}_+$, decide if $m^*(x) \geq q$ (if goal $=$ MAX) or if $m^*(x) \leq q$ (if goal $=$ MIN). As seen in Definition 1.1.1, if goal $=$ MAX, the set $\{(x,q) \mid x \in I \wedge m^*(x) \geq q\}$ (or $\{(x,q) \mid x \in I \wedge m^*(x) \leq q\}$ if goal $=$ MIN) is called the underlying language of P.

We drop the subscript "P" above as the context is clear. Note that for any optimization problem P, the corresponding decision problem P_D is not harder than the constructive problem P_C.

1.2.2 PO and NPO problems

Analogous to the NP class of decision problems, we define the NPO class of optimization problems:

Definition 1.2.2 (The class NPO). *An optimization problem $P = (I_P, SOL_P, m_P, \text{goal}_P)$ belongs to the class* **NPO** *if the following hold:*

1. *the set of instances I_P is recognizable in polynomial time,*

2. *there exists a polynomial q such that, given an instance $x \in I_P$,*

   (a) $|y| \leq q(|x|) \;\; \forall\, y \in SOL_P(x)$,

   (b) and it is decidable in polynomial time whether $y \in SOL_P(x)$ for every y with $|y| \leq q(|x|)$.

3. *the measure function m_P is computable in polynomial time.*

We note the expected relationship between the classes NP and NPO:

Theorem 1.2.1. *For any optimization problem P in NPO, the corresponding decision problem P_D belongs to NP.*

Proof. The proof of this theorem is straightforward and short, and is presented as Theorem 1.1 in Chapter 1, Section 1.4.2 of [3]. ∎

Definition 1.2.3 (**The class PO**). *An optimization problem P belongs to the class* **PO** *if it is in* NPO *and there exists a polynomial-time algorithm that, for any instance $x \in I_P$, returns an optimal solution $y \in SOL^*(x)$, together with its value $m^*(x)$.*

1.2.3 NP-hard optimization problems

To talk about the relative complexity of optimization problems as we did in Section 1.1.1 for decision problems using (polynomial-time) Karp reductions and the concept of NP-completeness, we must use polynomial-time Turing reductions as Karp reductions are only defined for decision problems.

Definition 1.2.4 (**NP-hard optimization problems**). *An optimization problem P is called* **NP-hard** *if, for every decision problem $P' \in \mathrm{NP}$, $P' \leq_T^p P$, that is, P' can be solved in polynomial time by an algorithm which queries an oracle that, for any instance $x \in I_P$, returns an optimal solution $y^*(x)$ of x along with its value $m_P^*(x)$.*

Remark 1.2.1. We note that as a consequence of the definition of NP-completeness, to conclude that an optimization problem P is NP-hard, it is enough to show that for an NP-complete problem P', $P' \leq_T^p P$ holds true.

We state a simple lemma:

Lemma 1.2.2. *Let a problem $P \in$* NPO*. The underlying language (or the corresponding decision version) of P is* NP-complete *if and only if P is* NP-hard*.*

Note that we do not call optimization problems with NP-complete decision versions NPO-complete. This is due to the connection between approximation algorithms and computational optimization problems. Reductions which *preserve approximation* in some respect are preferred to the usual Turing and Karp reductions. This is discussed in greater detail in Chapter 3.

We formally state another lemma, the proof of which can be found in Section 1.4.4, Chapter 1 of [3]:

Lemma 1.2.3.

1. *For any problem $P \in$* NPO*, $P_D \equiv_T^p P_E \leq_T^p P_C$.*

2. *If $P \in$* NPO *and its corresponding decision problem P_D is* NP-complete*, then $P_C \leq_T^p$*

P_D.

The above result shows that given an optimization problem, when the decision problem is NP-complete, the constructive, evaluation, and decision problems are equivalent.

1.3 Strong NP-hardness

In this final section, we discuss the concept of *strong* NP-hardness. This is exclusive to *numeric* problems, that is, problems with numerical input parameters, whose encoding lengths are generally exponential in the input length/size n of the problem. Sometimes, the NP-hardness (or NP-completeness) of an optimization or decision problem is only due to instances involving "large numbers". Such problems have *pseudo-polynomial time* algorithms. We formalize these terms in the following definitions:

Definition 1.3.1 (**Pseudo-polynomial time algorithm**). *A **pseudo-polynomial time** algorithm for a problem P is an algorithm that, given an instance $x \in I_P$, runs in time bounded by a polynomial in two variables - the numeric value of the input* (*usually defined to be the maximum value present in the input,* $\max(x)$), *and the length of the input* (*the encoding length or the number of bits required to represent it, denoted by* $|x|$), *unlike polynomial time algorithms.*

Definition 1.3.2 (**Strong NP-completeness**). *Let P be a decision problem in NP, with an instance $x \in I_P$. Let* $\max(x)$ *denote the value of the largest number occurring in the instance x. Given a polynomial p, let $P^{\max,p}$ be the restriction of P to those instances with the property that* $\max(x) \leq p(|x|)$. *If $P^{\max,p}$ remains an* NP-complete *problem for some polynomial p, then P is called **strongly*** **NP-complete**.

A decision problem is said to be *strongly NP-hard* if a strongly NP-complete problem has a polynomial reduction to it.

Definition 1.3.3 (**Strong NP-hardness**). *Let an optimization problem $P \in NPO$ and p be a polynomial. Define $P^{\max,p}$ as above. P is said to be **strongly*** **NP-hard** *if $P^{\max,p}$ is* NP-hard *for some polynomial p.*

We prove the following theorem:

Theorem 1.3.1. *Unless* P = NP, *no strongly* NP-hard *problem can have a pseudo-polynomial* *time algorithm.*

Proof. Let us assume that there exists a problem P which is strongly NP-hard and also has a pseudo-polynomial time algorithm. By the definition of pseudo-polynomial time algorithm, there exists an algorithm that solves P in time $O(q(\max(x), |x|))$ for any instance $x \in I_P$. That means, the restricted set of instances $P^{\max,p}$ can be solved in time $O(q(p(x), |x|))$ for all polynomials p, implying that $P^{\max,p} \in \text{P}\ \forall$ polynomials p. By the definition of strong NP-hardness of P, there exists a polynomial p such that $P^{\max,p}$ is NP-hard. Thus, we conclude that P = NP, which contradicts our initial hypothesis. ■

Observation 1.3.1. We list a few elementary observations to highlight the significance of the above definitions: [15]

(1) If a problem is solvable by a pseudo-polynomial time algorithm, then for every polynomial p, the problem is solvable by a polynomial time algorithm when the numerical parameters to the problem are given in unary notation.

(2) If a problem is not a numeric problem, then an algorithm for it is a pseudo-polynomial time algorithm if and only if it is a polynomial time algorithm.

(3) If a problem is not a numeric problem, then it is NP-complete (or NP-hard) if and only if it is NP-complete (or NP-hard) in the strong sense.

We conclude that while any NP-complete problem without numerical data is strongly NP-complete, it is not enough to show that a numeric problem is NP-complete. We must proceed to ask if the problem is also strongly NP-complete. If not, then we must ask if it has a pseudo-polynomial time algorithm. Strongly NP-complete problems, as noted in [15], are far more likely to be intractable than those that are simply NP-complete, as they cannot even be solved by a pseudo-polynomial time algorithm unless P = NP.

Chapter 2

The Pilot Assignment problem: Formulation and Results

In this chapter, we formally define the *Pilot Assignment* (PA) problem described in the Introduction. We define feasible pilot assignments for our problem and determine the objective function to be minimized. Next, we define the MIN-k-PARTITION problem and its dual, the MAX-k-CUT problem. We briefly discuss what is known about these problems in literature, although the choice of problem for the reduction to prove our main result in this chapter, the NP-hardness of PA, will become clear in Chapter 3. The most important result in this chapter is the proof of the fact that PA is NP-hard, by giving an explicit reduction from MIN-k-PARTITION to PA. Finally, we explicitly prove that MIN-k-PARTITION is strongly NP-hard. Throughout this chapter, we follow the notation used in [31, 23], which seems to be the standard notation used in the field.

2.1 System Model

The Pilot Assignment problem describes a major problem that hinders resource allocation in wireless networks. We consider the case of a cell-free massive MIMO system consisting of M single-antenna APs and K ($K \ll M$) UEs, which are randomly distributed in a large area. We know that in cell-free massive MIMO systems, a large number of distributed APs jointly provide uniform service to a small number of UEs using the same time-frequency resource. At the beginning of a coherence interval τ_c, during uplink training, channel state information between APs and UEs is required for both uplink and downlink data transmission. Thus,

τ pilot sequences of length τ each are assigned to the UEs for channel estimation. Each UE is assigned one pilot. However, due to the limited length of the coherence interval, the available number of orthogonal pilot sequences is normally smaller than the number of UEs ($\tau \ll K$). The number of available pilots is independent of the number of UEs. This leads to the reuse of the same pilot for different UEs. The pilot reuse causes an impairment known as pilot contamination, which can degrade the system performance.

The channel estimation error caused by pilot contamination translates into affected achievable rates and eventually leads to an observable degradation of system throughput. Thus, the system performance of cell-free massive MIMO has been characterized by the system throughput in literature [31], which is defined as $\sum_{k=1}^{K} R_k^u$, where K is the number of users and R_k^u denotes the uplink achievable rate for user k. The quantity R_k^u is further defined as $R_k^u =$

$$\frac{1-\tau/\tau_c}{2}\log_2\left(1+\frac{\rho^u \eta_k \left(\sum_{m\in A(k)} \gamma_{km}\right)^2}{\rho^u \sum_{k'\in O(k)} \eta_{k'} \left(\sum_{m\in A(k)} \gamma_{km}\frac{\beta_{k'm}}{\beta_{km}}\right)^2 + \rho^u \sum_{k'=1}^{K} \eta_{k'} \sum_{m\in A(k)} \gamma_{km}\beta_{k'm} + \sum_{m\in A(k)} \gamma_{km}}\right) \tag{2.1}$$

where

- η_k is the uplink power control coefficient,
- ρ^u is the normalized uplink SNR (signal-to-noise ratio),
- β_{km} denotes the large-scale fading coefficient between user k and AP m including geometric path loss and shadowing,
- γ_{km} denotes the mean-square of the channel estimation of the channel coefficient between user k and AP m,
- $A(k)$ denotes the indices of the APs serving user k, and
- $O(k)$ denotes the set of indices of users k' with the same pilot as user k, excluding user k.

In this problem, we assume that AP selection for each user has already been done. Thus, every quantity in Equation (2.1) is a constant. The pilot assignment problem seeks to find the optimum sets $O(k)$ for all k. Put simply, we seek a partition of the users into τ non-empty disjoint sets so that the system throughput is maximized. Thus, our optimization problem can be defined as

$$F_{opt} = \max_{\mathcal{P}} \sum_{k=1}^{K} R_k^u \tag{2.2}$$

where $\mathcal{P} = \{P_t, t \in \{1, 2, \ldots, \tau\}\}$ is a partition of the UEs into τ non-empty disjoint sets, depicting a pilot assignment scheme where P_t denotes the set of users assigned to pilot t. In this vein, notice that only the first term in the sum in the denominator of the huge fraction inside the logarithm expression in Equation (2.1) changes with a change in the assignment of pilots to users. As the logarithm is an increasing function, we may focus only on the huge fraction term inside it. Further, as noted before, only a single term in the denominator changes, the rest are all constants. Thus we focus on the term

$$\rho^u \sum_{k' \in O(k)} \eta_{k'} \Big(\sum_{m \in A(k)} \gamma_{km} \frac{\beta_{k'm}}{\beta_{km}} \Big)^2,$$

which should be minimized in order to maximize Equation (2.1). Notice that upto a scaling of the large-scale coefficients, minimizing the above term is equivalent to minimizing the term

$$\sum_{k' \in O(k)} \sum_{m \in A(k)} \left(\beta_{k'm}/\beta_{km}\right)^2 .$$

This results in maximization of the uplink achievable rate for user k, R_k^u. Since our goal is to minimize system throughput with an optimum pilot assignment for all the users, we shall consider the sum of the above terms for all the users, i.e., we aim to minimize

$$\sum_{k=1}^{K} \sum_{k' \in O(k)} \sum_{m \in A(k)} \left(\beta_{k'm}/\beta_{km}\right)^2 .$$

There is also some physical intuition behind dropping the above constants. The large-scale coefficient for a user and an AP is higher when they are geographically closer [27]. It is safe to assume that for a user k, AP selection yields only those APs m in the set $A(k)$ for which the large-scale coefficients β_{km} are reasonably high. The above equations tell us that given a user k, the potential disturbance from users k' is severe when the proximity between the interfering users k' and the APs in the set $A(k)$ is higher than that between the user k and the APs in $A(k)$. In a way, we want to put those users k' in $O(k)$, whose large-scale coefficients with respect to the APs that have been selected to serve k are low. In other words, they are farther from those APs than user k.

2.2 Problem formulation

Given M APs, K users, with each user assigned a set $A(k)$ $(1 \leq k \leq K)$ which denotes the indices of the subset of APs serving it, large-scale coefficients β_{km} between a user k and AP m, and τ pilots, we need to find a partition of the K users into τ disjoint sets (where a set contains the users served by the same pilot) such that

$$\sum_{k=1}^{K} \sum_{k' \in O(k)} \sum_{m \in A(k)} \left(\beta_{k'm}/\beta_{km}\right)^2 \tag{2.3}$$

is minimized, where $O(k)$ is the set of indices users k' with the same pilot as user k, excluding user k. Note that since AP selection is done before pilot assignment [31], the innermost summation remains untouched in this optimization problem.

Definition 2.2.1 (**Cell-Free Massive MIMO system**). *Let $\mathcal{A}$ denote the set of APs with cardinality M, U denote the set of users with cardinality K, and Ψ denote the set of pilots with cardinality τ. Let $\boldsymbol{\beta}$ be a $K \times M$ matrix, with the (k, m)-th element denoting the large-scale coefficient β_{km} between user k and AP m. For each user k, the set $A(k)$ $(1 \leq k \leq K)$ denotes the indices of the subset of APs serving it, as described above. We shall refer to the tuple $\left(\mathcal{A}, U, \{A(k)\}_{k=1}^{K}, \boldsymbol{\beta}, \Psi\right)$ as a* ***cell-free massive MIMO (CF-mMIMO) system S****.*

Definition 2.2.2 (**Feasible Pilot Assignment**). *A **feasible** pilot assignment for a CF-mMIMO system S is a well-defined, surjective function $f : U \to \Psi$.*

We state an easy-to-see lemma:

Lemma 2.2.1. *Finding feasible pilot assignments for the users in a given cell-free massive MIMO system S can be done in polynomial time.*

Proof. All we really need is the partition of the set of users U such that no two subsets in the partition intersect. As long as $K \gg \tau$, this is always possible. A simple greedy approach to see that this is true would be to randomly select $\tau - 1$ users from U, and assign them to the first $\tau - 1$ pilots in Ψ respectively. Then assign the remaining $K - \tau + 1$ users to the remaining τ-th pilot in Ψ. By construction, this is a feasible pilot assignment for the system S. ■

Let the τ subsets of U induced by f be $V_1, V_2, \ldots V_\tau$. In Equation (2.3), observe the outer two summations. Given a user k, we only have to consider those users k' which are served by the same pilot as it. In other words, given a set in a partition, we must look at all possible pairs of users within it. We must then look at the contribution of the *pair* k, k' to the second sum, which by symmetry turns out to be

$$\sum_{m \in A(k)} (\beta_{k'm}/\beta_{km})^2 + \sum_{m' \in A(k')} (\beta_{km'}/\beta_{k'm'})^2 \tag{2.4}$$

As mentioned in [31], we can regard the above term as the quantity of interference (thus leading to potential pilot contamination) between users k and k'. Since the order or permutation of the elements of the pair k, k' matters, each pair of elements in a partition contributes *two* terms to the second sum.

We then rewrite the outer two summations in Equation (2.3) as a summation over all possible pairs k, k' of Equation (2.4), and then a summation of this over all sets in the partition:

$$\sum_{t=1}^{\tau} \sum_{k,k' \in V_t} \left(\sum_{m \in A(k)} (\beta_{k'm}/\beta_{km})^2 + \sum_{m' \in A(k')} (\beta_{km'}/\beta_{k'm'})^2 \right) \tag{2.5}$$

Definition 2.2.3 (**Pilot Assignment Problem (PA)**). *Given a cell-free massive MIMO system S, we call the optimization problem*

$$\min_{f\, feasible} \sum_{\substack{k,k' \in U \\ f(k)=f(k')}} \left(\sum_{m \in A(k)} (\beta_{k'm}/\beta_{km})^2 + \sum_{m' \in A(k')} (\beta_{km'}/\beta_{k'm'})^2 \right)$$

*the **Pilot Assignment (PA)** problem.*

Finally, to talk about the computational complexity of the Pilot Assignment problem, we assume that all numerical values appearing as input data for PA are rational and that they are encoded in binary form. We shall define both the optimization and decision versions of the problem. This will enable us to prove hardness results for the problem later:

Definition 2.2.4 (**Decision version of the PA problem**). *The decision version of PA is defined by a triple (I_{PA}, q, SOL_{PA}), where I_{PA} is the set of all cell-free massive MIMO systems*

S, $q \in \mathbb{Q}_+$ and $SOL_{\mathsf{PA}} : I_{\mathsf{PA}} \to \{0, 1\}$ is a function that assigns to each pair (S, q) either the value 0 or 1. The assignment is done based on whether an instance S has a value of at most q for Equation (2.5). The problem asks if there exists an instance $S \in I_{\mathsf{PA}}$ such that given a number q, $SOL_{\mathsf{PA}}(S, q) = 1$.

Definition 2.2.5 (**Optimization version of the PA problem**). *The minimization problem PA is characterized by the triple $(I_{\mathsf{PA}}, SOL_{\mathsf{PA}}, m_{\mathsf{PA}})$, where I_{PA} is the set of all cell-free massive MIMO systems S (instances of PA), SOL_{PA} is a function that assigns every instance $S \in I_{\mathsf{PA}}$ to its set of feasible pilot assignments f, and m_{PA} is a measure function that assigns to each pair (S, f), where $S \in I_{\mathsf{PA}}$ and $f \in SOL_{\mathsf{PA}}(S)$, the value of Equation (2.5). The problem asks for a given instance S, a feasible assignment $f \in SOL_{\mathsf{PA}}(S)$ such that $m_{\mathsf{PA}}(S, f) = \min_{f' \in SOL_{\mathsf{PA}}(S)} m_{\mathsf{PA}}(S, f')$.*

2.3 The Min-k-Partition and the Max-k-Cut problems

In this section, we define the two very well-studied problems that we will use to prove hardness results for the Pilot Assignment problem.

Definition 2.3.1 (**Min-k-Partition**). *Given an undirected graph $G = (V, E)$ with n vertices and weight $\omega_{i,j} \in \mathbb{Q}_+$ for the edge joining vertices i and j $\forall\, 1 \leq i, j \leq n$, the* Min-$k$-Partition *problem seeks to find a partition $\mathcal{V}$ of V into k disjoint sets $\{V_1, V_2, \ldots, V_k\}$ such that the total weight of the edges with endpoints within the same set is minimum.*

The objective of the above problem can be formulated as:

$$\min_{\mathcal{V} = \{V_1, V_2, \ldots, V_k\}} \sum_{p=1}^{k} \sum_{i,j \in V_p} \omega_{i,j} \tag{2.6}$$

The Min-k-Partition is also known as the Min-Sum k-Clustering problem [5, 10, 4].

Definition 2.3.2 (**Max-k-Cut**). *Given an undirected graph $G = (V, E)$ with weighted edges, the* Max-k-Cut *problem seeks to find a partition of the vertices into k disjoint subsets such that the total weight of the edges with endpoints in disjoint sets is maximum.*

Lemma 2.3.1. *Solving the* Max-k-Cut *problem on a graph is equivalent to solving the* Min-k-Partition *problem on the same graph. In other words, the* Min-k-Partition *problem* *is the dual of the* MAX-k-CUT *problem.*

Proof. The proof of this fact is trivial: For every partition, an edge either has both its endpoints within the same block or in distinct blocks. Therefore the MAX-k-CUT problem is also known as the MIN-k-PARTITION problem since minimizing the total weight of the edges between vertices in the same set of the partition is equivalent to maximizing the total weight of edges joining different sets of the partition [12, 28]. ■

Example 1. Below is an example of a MIN-2-PARTITION problem, with the minimum 2-partition denoted by the dotted orange and pink lines. The measure or weight of the minimum partition is 9. Notice that this is also the maximum 2-cut for the graph, giving us the solution to the corresponding MAX-2-CUT problem. The weight of the maximum cut is 24.

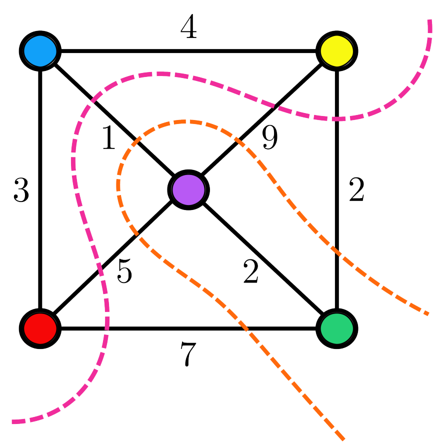


Figure 2.1: The optimal solution to a MIN-2-PARTITION problem.

Both the MIN-k-PARTITION and the MAX-k-CUT problems have been extensively studied in the literature. They are known to be NP-hard [9, 1, 16]. In particular, Linear Programming (LP) and Semi-definite Programming (SDP) relaxation approaches have been used to formulate and solve both the problems [12, 14, 9, 13, 25, 29, 18]. In the case of MIN-k-PARTITION, sparse instances of the problems can be solved faster with linear methods than SDP-based methods, while SDP approaches perform better on dense instances [18, 29]. Exploring these methods in further detail, however, is out of the scope of this thesis. For the sake of completion, we conclude this section by giving an edge-only $0-1$ ILP formulation of MIN-k-PARTITION, following [9, 8, 18, 29]:

For an instance of MIN-k-PARTITION following Definition 2.3.1, the graph G can be completed to $K_{|V|}$ without loss of generality, by adding zero-weight edges. The edge set is then $E = \{(i,j) | 1 \leq i < j \leq |V|\}$. Define the binary variable x_{ij} as

$$x_{ij} = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are in the same partition} \\ 0 & \text{otherwise.} \end{cases}$$

The ILP formulation given by the authors of [9, 8] is:

$$\min \quad \sum_{(i,j)\in E} w_{ij} x_{ij} \tag{2.7a}$$

$$\text{s.t.} \quad x_{ih} + x_{hj} - x_{ij} \leq 1 \quad \forall\, i,j,h \in V,\ i < j < h \tag{2.7b}$$

$$\sum_{i,j \,\in Q \,:\, i<j} x_{ij} \geq 1 \quad \forall\, Q \subseteq V \text{ where } |Q| = k+1 \tag{2.7c}$$

$$x_{ij} \in \{0,1\} \quad \forall\, i,j \in V,\ i < j \tag{2.7d}$$

The *triangle* constraint 2.7b requires the values of the variables to be consistent with respect to partition membership. If $x_{ih} = 1$ and $x_{hj} = 1$, indicating that i, h and j are in the same partition, by transitivity this implies $x_{ij} = 1$. For every subset of $k+1$ vertices, the *clique* constraint 2.7c forces at least two vertices to be in the same set. Together with constraint 2.7b, this implies that there are at most k sets. In total, there are $3\binom{|V|}{3}$ triangle inequalities and $\binom{|V|}{k+1}$ clique inequalities. We refer to this formulation as the *edge formulation*.

2.4 Primary hardness results for the Pilot Assignment problem

We state and prove the first major theorem in this thesis:

Theorem 2.4.1. *The following results hold for the time complexity of the PA problem:*

(*i*) PA $\in$ NPO.

(*ii*) *The optimization version of the* PA *problem, or simply the Pilot Assignment problem, is* NP-hard.

(*iii*) *The* PA *problem is strongly* NP-hard.

Proof.

(*i*) It is recognizable in polynomial time whether a string encodes a tuple representing a cell-free massive MIMO system, as it primarily involves a check of the cardinality of multiple sets, and the dimension of a matrix. If all the users and pilots are listed individually, then the encoding length of a feasible pilot assignment for the users cannot exceed the encoding length of the CF-mMIMO system. From Lemma 2.2.1, we know that it is decidable in polynomial time if a pilot assignment is feasible. Finally, it is clear that Equation (2.5) is computable in polynomial time. Therefore, PA $\in$ NPO.

(*ii*) We prove that the Pilot Assignment problem is NP-hard by establishing the following claim: MIN-k-PARTITION $\leq_T^p$ PA.

Consider an arbitrary instance of the MIN-k-PARTITION problem, specified by a weighted graph $G = (V, E, \omega)$, where $\omega : E \to \mathbb{R}_+$ is a function mapping edges to their weights.

We construct an instance of PA as follows:
Set $K = |V|$. The set of users U is set to be the set of vertices V. Thus, $|U| = K$. The number of pilots τ is set to be k. This determines the set $\Psi = \{1, \dots, k\}$. For all users indexed by $1 \leq i \leq K$, set $|A(i)| = 1$. This determines K number of APs. Since we know that in a practical cell-free massive MIMO system, we have $M \gg K$, we could define many dummy APs to achieve this condition. We set the total number of APs in our PA instance as some arbitrarily large constant $M \gg K$, which determines the set $\mathcal{A} = \{1, \dots, K, \dots, M\}$. Further, $\forall\, 1 \leq i \leq K$, let $A(i) = \{i\}$. Thus, a single, distinct AP indexed by i serves the user i. Now, $\forall\, 1 \leq i \leq K$ and $\forall\, 1 \leq m \leq M$ set

$$\beta_{im} = \begin{cases} 1 & \text{if } m = i, \\ \sqrt{\frac{\omega((i,m))}{2}} & \text{if } m \neq i \text{ and } m \leq K, \\ 0 & \text{otherwise.} \end{cases}$$

This determines the matrix $\boldsymbol{\beta}$ in our instance of the problem.

($\Rightarrow$) If we have a solution to an instance of the MIN-k-PARTITION problem, then by Definition 2.3.1, we have a partition $\mathcal{V} = \{V_1, V_2, \ldots, V_k\}$ of the vertex set V such that the expression $\sum_{t=1}^{k} \sum_{i,j \in V_t} \omega((i,j))$ is minimized. By our construction, the set of vertices V is the set of users U and k is the cardinality of the set Ψ, τ. Replacing k by τ and i, j by the arbitrary indices k, k' to denote the users in our constructed instance of the PA problem, we get that the minimized equation is $\sum_{t=1}^{\tau} \sum_{k,k' \in V_t} \omega((k,k'))$. Recall that the objective function to be minimized in a general instance of the PA problem is Equation (2.5), which can be simplified to

$$\begin{aligned}
&\sum_{t=1}^{\tau} \sum_{k,k' \in V_t} \left(\sum_{m \in A(k)} (\beta_{k'm}/\beta_{km})^2 + \sum_{m' \in A(k')} (\beta_{km'}/\beta_{k'm'})^2 \right) \\
&= \sum_{t=1}^{\tau} \sum_{k,k' \in V_t} \left((\beta_{k'k}/\beta_{kk})^2 + (\beta_{kk'}/\beta_{k'k'})^2 \right) \\
&= \sum_{t=1}^{\tau} \sum_{k,k' \in V_t} \omega((k,k'))
\end{aligned}$$

Thus we have a solution to our PA problem instance.

($\Leftarrow$) By the above simplification of Equation (2.5), if we have a solution for the above-constructed instance of the PA problem, we end up minimizing the expression

$$\sum_{t=1}^{\tau} \sum_{k,k' \in V_t} \omega((k,k'))$$

Now, we map the users k, k' back to the vertices of the graph via arbitrary indices i, j, and note that the number of pilots τ is in fact the desired number of cuts, k. So we end up minimizing the expression

$$\sum_{t=1}^{k} \sum_{i,j \in V_t} \omega((i,j))$$

which represents the total weight of such edges which have both endpoints in the same partition. Thus, due to the construction of our PA instance, we also have a solution to the corresponding MIN-k-PARTITION instance.

Therefore, the MIN-k-PARTITION instance has a solution *if and only if* if the above PA instance has a solution. This proves the claim that the MIN-k-PARTITION problem is polynomial-time reducible to the Pilot Assignment problem.

From the discussion in Section 2.3, we see that an NP-hard problem optimization problem is Turing-reducible, in polynomial time, to the problem of our interest. Thus it is clear that the Pilot Assignment problem is NP-hard.

(iii) We just gave a polynomial time reduction from MIN-k-PARTITION to PA. It is stated in [13] that MIN-k-PARTITION is strongly NP-hard due to a reduction from the GRAPH k-COLOURABILITY problem, which has been proven to be strongly NP-complete [15]. As we could not find a reference for an explicit proof of this fact in the literature, we give a simple proof of the aforementioned reduction: The CHROMATIC NUMBER or GRAPH k-COLOURABILITY is a non-numeric decision problem which asks whether the vertices of a graph G can be coloured using at most k colours such that no two adjacent vertices have the same colour. In other words, it asks if we can partition the vertices into at most k sets such that each of the sets is an independent set.

We give a simple Turing reduction from GRAPH k-COLOURABILITY to MIN-k-PARTITION as follows: Given an instance (G, E, k) of the GRAPH k-COLOURABILITY problem, associate to it an instance of MIN-k-PARTITION defined by $G = (V, E, \omega)$, such that $\omega : E \to 1$. It is easy to see that G is k-colourable if and only if the optimal solution to the associated instance of MIN-k-PARTITION yields a value of 0. This is due to the fact that a solution to the k-COLOURING problem on G induces k independent sets in G. In our instance of MIN-k-PARTITION, these k independent sets translate to k subsets of the set of vertices V of the graph, such that no two vertices in a given subset are adjacent. Thus, the endpoints of any edge in G must be in two different sets. This gives us the minimum possible value of the sum of the weights of the edges with endpoints in the same partition: 0. Thus, we see that independent sets in G give us the optimum solution to the constructed instance of MIN-k-PARTITION. On the other hand, if the solution to our MIN-k-PARTITION instance has a value of 0, then all the edges in G have endpoints in disjoint sets of the partition. This follows from the fact that the weight of each edge is 1. Thus, the vertices in a given subset are pairwise disjoint, forming an independent set. Colouring the k independent sets in this partition of V using k different colours gives us the desired solution to the GRAPH k-COLOURABILITY problem. ■

We end this section with a simple corollary of the above theorem:

Corollary 2.4.2. *For every $q \in \mathbb{Q}_+$, the decision version of the PA problem is* NP-complete.

Proof. The proof of this fact follows from Lemma 1.2.2. One should note that since all decision problems in NP are Turing-reducible (in polynomial time) to PA, an oracle that gives the optimal solution for PA with its value automatically returns YES or NO for an instance of any problem in NP mapped to an instance of PA. By Theorem 1.2.1, this is true of the corresponding decision version of every instance of PA too. Thus, we have a Karp reduction from any decision problem in NP to the corresponding decision problem of PA. ■

Chapter 3

Approximation theory and results

In the previous chapter, we proved that the Pilot Assignment problem is strongly NP-hard. This tells us that unless P = NP, there exists no polynomial-time algorithm to solve PA. In this chapter, we focus on the more practical aspects of tackling this problem. Instead of trying to obtain optimal solutions, we look at how well we can approximate the optimal solution to the problem in polynomial time. The varying behaviour of NP-hard optimization problems with respect to their approximability properties is captured by the definition of approximation classes that form a strict hierarchy under the assumption of P $\neq$ NP, whose levels correspond to different degrees of approximation.

Further, even though the decision problems corresponding to most NP-hard optimization problems are Karp-reducible to each other, the optimization problems do not share the same approximability properties. This is due to the fact that Karp reductions do not always preserve the measure function and, even if they do, they seldom preserve the quality of the solutions. This is especially true of Karp reductions from the decision versions of maximization problems to minimization problems and vice versa. As will be clear, this is the reason we gave a reduction from MIN-k-PARTITION to PA instead of using MAX-k-CUT, even though these are equivalent problems. We introduce a stronger kind of reducibility, namely AP-reducibility, that not only maps instances of a problem P_1 to instances of a problem P_2, but also maps back good solutions for P_2 to good solutions for P_1.

Approximation-preserving reductions induce an order on optimization problems based on their "difficulty" of being approximated. They are also an essential tool for proving non-approximability results. In this chapter, we prove that our reduction in Chapter 2 is approximation-preserving, and also give an explicit approximation-preserving reduction from PA to MIN-k-PARTITION. The latter is especially important from a practical viewpoint, as it allows us to translate the performance ratios for various algorithms/heuristics that may be designed for the MIN-k-PARTITION problem. Finally, we use the reduction from Chapter 2 to prove negative approximability results for the Pilot Assignment problem.

3.1 Approximation algorithms: Definitions and Results

The definitions and results in this section can be found in Chapters 3 and 8 of [3].

3.1.1 Approximation classes

Definition 3.1.1 (**Approximation algorithm**). *Given an optimization problem $P = (I_P, SOL_P, m_P, \text{goal}_P)$, an algorithm $\mathcal{A}$ is an **approximation algorithm** for P if, for any given instance $x \in I$, it returns an **approximate solution**, that is, a feasible solution $\mathcal{A}(x) \in SOL(x)$.*

In order to express the quality of an *approximate solution*, the most widely used notions are that of the *relative error* and the *performance ratio*.

Definition 3.1.2 (**Relative error**). *Given an optimization problem P, for any instance x of P and for any feasible solution y of x, the **relative error** of y with respect to x is defined as*

$$E(x, y) = \frac{|m^*(x) - m(x, y)|}{\max\{m^*(x), m(x, y)\}}.$$

We notice immediately that in the case of both maximization and minimization problems, the relative error is 0 when the solution obtained is optimal, and approaches 1 as the approximate solution gets poorer.

Definition 3.1.3 (**ε-approximate algorithm**). *Given an optimization problem P and an approximation algorithm $\mathcal{A}$ for P, we say that $\mathcal{A}$ is an **ε-approximate algorithm** for P if, given any input instance x of P, the relative error of the approximate solution $\mathcal{A}(x)$ provided by algorithm $\mathcal{A}$ is bounded by ε, that is*

$$E(x, \mathcal{A}(x)) \leq \varepsilon.$$

Definition 3.1.4 (**Performance ratio**). *Given an optimization problem P, for any instance*

x of P and for any feasible solution y of x, the ***performance ratio*** *of y with respect to x is defined as*

$$R(x,y) = \max\left(\frac{m(x,y)}{m^*(x)}, \frac{m^*(x)}{m(x,y)}\right).$$

Remark 3.1.1. The performance ratio and the relative error of an approximate solution y with respect to an instance x of an optimization problem P are related as follows:

$$E(x,y) = 1 - \frac{1}{R(x,y)}.$$

Definition 3.1.5 ($r(n)$-approximate algorithm). *Given an optimization problem P in* NPO*, an approximation algorithm $\mathcal{A}$ for P, and a function $r : \mathbb{N} \mapsto (1, \infty)$, we say that $\mathcal{A}$ is an* ***$r(n)$-approximate algorithm*** *for P if, for any instance x such that $SOL(x) \neq \emptyset$, the performance ratio of the feasible solution $\mathcal{A}(x)$ with respect to x satisfies the following inequality:*

$$R(x, \mathcal{A}(x)) \leq r(|x|).$$

An NP-hard optimization problem P is called ε-*approximable* (respectively, $r(n)$-*approximable*) if there exists a polynomial-time ε-*approximate* (respectively, $r(n)$-*approximate* algorithm for P.

We now proceed to define several classes of optimization problems in NPO with special approximability properties:

Definition 3.1.6 (Class APX). *The class of all* NPO *problems P such that, for some fixed constant $c > 1$, there exists a polynomial-time c-approximate algorithm (also called constant-factor approximation algorithm) for P.*

Lemma 3.1.1. *If* $\mathrm{P} \neq \mathrm{NP}$*, then* $\mathrm{APX} \subset \mathrm{NPO}$*.*

Proof. The proof of this result follows from Theorem 3.3 and Corollary 3.4 under Section 3.1.3 in Chapter 3 of [3]. ■

Definition 3.1.7 (Class PTAS). *Let P be an* NPO *problem. A* ***Polynomial Time Approximation Scheme*** *is an algorithm $\mathcal{A}$ that, for any given $\epsilon > 0$ and an instance x of*

P, when applied to input (x, ϵ) approximates x within a factor $1 + \epsilon$ in time that is polynomial in $|x|$, and can depend arbitrarily upon $1/\epsilon$. The class of NPO *problems P with such approximation schemes is called* **PTAS**.

Lemma 3.1.2. *If* $\mathrm{P} \neq \mathrm{NP}$, *then* $\mathrm{PTAS} \subset \mathrm{APX}$.

Proof. The proof of this result follows from Theorem 3.12 and Corollary 3.13 under Section 3.2.2 in Chapter 3 of [3]. ■

Definition 3.1.8 (Class FPTAS). *Let P be an* NPO *problem. A **Fully Polynomial Time Approximation Scheme** is an algorithm $\mathcal{A}$ that, for any given $\epsilon > 0$ and an instance x of P, when applied to input (x, ϵ) approximates x within a factor $1 + \epsilon$ in time that is polynomial in $|x|$ and $1/\epsilon$. The class of* NPO *problems P with such approximation schemes is called* **FPTAS**.

Definition 3.1.9 (Polynomially bounded optimization problem). *An optimization problem is **polynomially bounded** if there exists a polynomial p such that, for any instance x of the problem and for any $y \in SOL(x)$, $m(x, y) \leq p(|x|)\cdot$*

Theorem 3.1.3. *No* NP-hard *polynomially bounded optimization problem belongs to the class* FPTAS *unless* $\mathrm{P} = \mathrm{NP}$.

Proof. This theorem has been proved as Theorem 3.15 under Section 3.3.3 of Chapter 3 of [3]. ■

Lemma 3.1.4. *If* $\mathrm{P} \neq \mathrm{NP}$, *then* $\mathrm{FPTAS} \subset \mathrm{PTAS}$.

Proof. See the proof of corollary 3.16 in Section 3.3.3, Chapter 3 of [3]. ■

Definition 3.1.10 (Class F-APX). *Given a class of functions F, F-*APX *is the class of all NPO problems P such that, for some function $r \in F$, there exists a polynomial-time $r(n)$-approximate algorithm for P.*

The following table defines various F-APX classes:

Class F-APX	Set of functions denoted by F
APX	constant functions
log-APX	$O(\log n)$
poly-APX	$\bigcup_{k>0} O(n^k)$
exp-APX	$\bigcup_{k>0} O(2^{n^k})$

Lemma 3.1.5 ([3]). *If* P $\neq$ NP*,* exp-APX $\subset$ NPO*.*

Proof. Let $P = (I,\ SOL,\ m,$ goal$)$ be a problem in NPO. Since m is computable in polynomial time, there exist h and k such that for any $x \in I$ with $|x| = n$ and for any $y \in SOL(x)$, $m(x, y) \leq h2^{n^k}$. This is because the range of possible values of $m(x, y)$ has an upper bound given by $M = 2^{p(|x|)}$ for some polynomial p, which is again due to the properties of NPO problems which state that the length $|y|$ of any solution $y \in SOL(x)$ is bounded by $q(|x|)$ for some polynomial q, and m is computable in polynomial time (see Definition 1.2.2). This implies that any feasible solution has a performance ratio bounded by $h2^{n^k}$. Indeed, the polynomial bound on the computation time of the measure function for all NPO problems implies that they are $h2^{n^k}$-approximable for some h and k. This seems to imply that the classes exp-APX and NPO are the same. However, we note that there exist several problems in NPO for which it is hard even to decide whether any feasible solution exists, (and thus to find such a feasible solution) unless P = NP. An example of such a problem is the Minimum $\{0, 1\}$-Linear Programming problem, which belongs to NPO. The problem instance consists of a matrix $A \in \mathbb{Z}^{m \times n}$ and vectors $b \in \mathbb{Z}^m$, $w \in \mathbb{N}^n$. The problem asks for a solution $x \in \{0, 1\}^n$ such that $Ax \geq b$, and the measure function $\sum_{i=1}^{n} w_i x_i$ is minimized. Given an integer matrix A and an integer vector b, deciding whether a binary vector x exists such that $Ax \geq b$ is NP-hard, as the Satisfiability problem is polynomial-time reducible to this decision problem (see Example 1.10 in Section 1.3.1, Chapter 1 of [3]). This implies that, if P $\neq$ NP, then Minimum $\{0, 1\}$-Linear Programming does not belong to exp-APX. ∎

Result 3.1.1. *Finally, we have the following inclusion:*

$$\text{FPTAS} \subseteq \text{PTAS} \subseteq \text{APX} \subseteq \text{log-APX} \subseteq \text{poly-APX} \subseteq \text{exp-APX} \subseteq \text{NPO},$$

and the inclusions are all strict unless P = NP*.*

3.1.2 Approximation-preserving reduction: AP-reducibility

A reduction from a problem P_1 to a problem P_2 gives us a method to solve P_1 if we have an algorithm that can solve P_2. In the context of approximation algorithms, the reduction should guarantee that an approximate solution of P_2 yields an approximate solution for P_1. Thus, the polynomial-time Karp reducibility defined for decision problems must be modified, since we need not only a function f mapping instances of P_1 into instances of P_2, but also a function g mapping back solutions of P_2 into solutions of P_1 while preserving certain properties of the solutions.

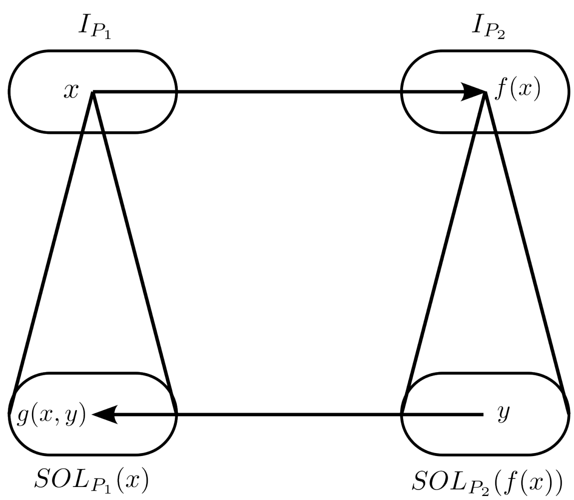


Figure 3.1: Approximation-preserving reduction between two optimization problems.

We now define an approximation-preserving reducibility called *AP-reducibility.* Although different types of approximation-preserving reducibilities have been defined in the literature, AP-reducibility is sufficiently general to incorporate the properties of almost all such reducibilities, while also establishing a *linear relation* between performance ratios. The latter property is especially important in order to preserve membership in all approximation classes.

Definition 3.1.11 (AP-reducibility). *Let P_1 and P_2 be two optimization problems in NPO. P_1 is said to be* ***AP-reducible*** *to P_2, written $P_1 \leq_{AP} P_2$, if there exist two functions f and g and a constant $\alpha \geq 1$ such that:*

1. *For an instance $x \in I_{P_1}$, and for any rational $r > 1$, $f(x, r) \in I_{P_2}$.*

2. *For an instance* $x \in I_{P_1}$*, and for any rational* $r > 1$*, if* $SOL_{P_1}(x) \neq \emptyset$*, then* $SOL_{P_2}(f(x,r)) \neq \emptyset$*.*

3. *For any instance* $x \in I_{P_1}$*, for any rational* $r > 1$*, and for any* $y \in SOL_{P_2}(f(x,r))$*,* $g(x,y,r) \in SOL_{P_1}(x)$*.*

4. f *and* g *are computable by two algorithms* $\mathcal{A}_f$ *and* $\mathcal{A}_g$*, respectively, whose running time is polynomial for any fixed rational* r*.*

5. *For any instance* $x \in I_{P_1}$*, for any rational* $r > 1$*, and for any* $y \in SOL_{P_2}(f(x,r))$*,*

$$R_{P_2}(f(x,r),y) \leq r \;\Rightarrow\; R_{P_1}(x, g(x,y,r)) \leq 1 + \alpha(r-1).$$

In the rest of this thesis, this condition will be referred to as the ***AP-condition****.*

The triple (f,g,α) *is said to be an* ***AP-reduction*** *from* P_1 *to* P_2*.*

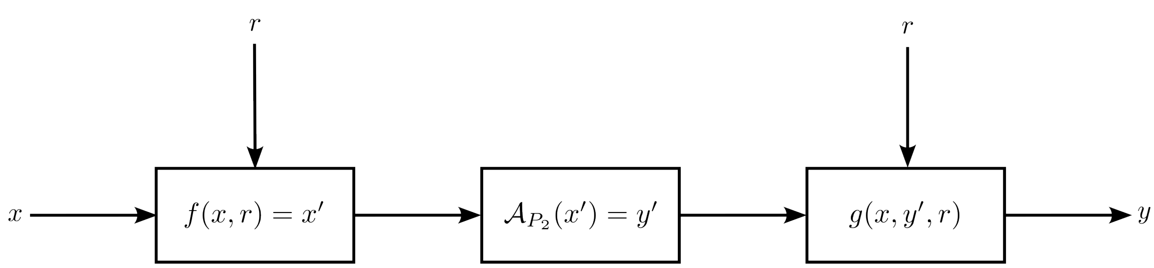


Figure 3.2: Schematic representation of how to use AP-reduction.

The significance of the above reducibility is reflected in the following lemma:

Lemma 3.1.6 ([3])**.** *If* $P_1 \leq_{AP} P_2$ *and* $P_2 \in$ APX *(respectively,* $P_2 \in$ PTAS*), then* $P_1 \in$ APX *(respectively,* $P_1 \in$ PTAS*).*

Proof. Let (f,g,α) be an AP-reduction from P_1 to P_2. If $P_2 \in$ APX and $\mathcal{A}_{P_2}$ is an algorithm for P_2 with performance ratio at most r, then (see Figure 3.2)

$$\mathcal{A}_{P_1}(x) = g(x,\, \mathcal{A}_{P_2}(f(x,r)),\, r)$$

is an algorithm for P_1 with performance ratio at most $(1+\alpha(r-1))$. Similarly, if $P_2 \in$ PTAS and $\mathcal{A}_{P_2}$ is a polynomial-time approximation scheme for P_2, then

$$\mathcal{A}_{P_1}(x, \epsilon') = g(x,\ \mathcal{A}_{P_2}(f(x, 1+\epsilon),\ \epsilon),\ 1+\epsilon)$$

is a polynomial-time approximation scheme for P_1, where $\epsilon' = \alpha\epsilon$. This follows from the fact that $\mathcal{A}_{P_1}$ has a performance ratio that is at most $1 + \alpha((1+\epsilon) - 1) = 1 + \alpha\epsilon$. ■

Remark 3.1.2. APX reductions not only preserve membership in PTAS and APX, but also in the larger classes log-APX and poly-APX, subject to the additional condition that for a fixed problem size, the computation time of f, g must be non-increasing as the approximation ratio increases.

Remark 3.1.3. As noted in Section 8.2, Chapter 8 of [3], Definition 3.1.11 actually introduces the notion of an approximation preserving reducibility *scheme* since, according to Figure 3.1, for any r, two functions that depend on r, namely f_r and g_r, exist that reduce P_1 to P_2 while linearly preserving approximation ratios. This extension makes sense as we can't justify ignoring the quality of the solution we are looking for *a priori*, when reducing one optimization problem to another. However, this information is not required in most (but not all) reductions seen in the literature. Hence, from now on whenever functions f and g do not depend on r, we will not specify this dependency. We shall replace $f(x, r)$ and $g(x, y, r)$ with $f(x)$ and $g(x, y)$, respectively.

Lemma 3.1.7. AP-reducibility *is transitive, i.e., if* $P_1 \leq_{AP} P_2$ *and* $P_2 \leq_{AP} P_3$*, then* $P_1 \leq_{AP} P_3$.

Proof. The proof of the above lemma is straightforward and only requires applying the conditions in Definition 3.1.11 successively on the reduction from P_1 to P_2 and from P_2 to P_3 to get the same properties for the reduction from P_1 to P_3. ■

Thus, AP-reducibility induces a partial order on problems in the same approximation class.

Definition 3.1.12 (C-hard problem). *Given a class* C *of* NPO *problems, a problem P is* **C-hard** *with respect to AP-reducibility if, for any $P' \in$ C, $P' \leq_{AP} P$. A* C-hard *problem is* **C-complete** *with respect to the AP-reducibility if it belongs to* C.

Section 8.2.1, Chapter 8 of [3] states that if a problem P is NPO-complete, then $P \notin$ APX, unless P = NP. Moreover, if a problem P is APX-complete, then $P \notin$ PTAS, unless P = NP. Hence, completeness results automatically yield non-approximability results.

3.2 Approximability results for the Pilot Assignment problem

We will now address the disparity in the approximability of the equivalent MIN-k-PARTITION and MAX-k-CUT problems. Indeed, in general, the approximability results that hold for an optimization problem, be it a minimization or a maximization problem, are vastly different from those that hold for its dual. We know that while the MAX-k-CUT has a polynomial-time $(1 - k^{-1})$-approximate algorithm [26, 14], it is not possible to find an ε-approximate algorithm for the MIN-k-PARTITION problem in polynomial time for any ε unless P = NP [26]. In fact, MAX-k-CUT is APX-complete, while MIN-k-PARTITION is not in APX (see Appendix B of [3]).

The outline of this section is as follows: We first give an AP-reduction from PA to MIN-k-PARTITION. This will help us translate the performance ratios that exist for the approximation of special cases of MIN-k-PARTITION (for example, when $k = 2$ or 3) into performance ratios for the corresponding PA problem [21]. We then present a few approximability results for the MIN-k-PARTITION problem from the literature in the form of a lemma, and then show that the reduction we gave from MIN-k-PARTITION to PA is an AP-reduction. Together, these will yield both positive and negative approximability results for the PA problem.

3.2.1 Preserving approximation bounds from MIN-k-PARTITION to PA

From Definition 3.1.11, it is clear that if we mean to use the performance ratios that have been studied in the literature for the MIN-k-PARTITION problem for the PA problem, we must provide an AP-reduction from PA to MIN-k-PARTITION.

Theorem 3.2.1. PA $\leq_{AP}$ MIN-k-PARTITION.

Proof. Since we've shown that the decision version of PA is NP-complete, it follows from the definition of NP-completeness (and Remark 1.1.2) that PA and MIN-k-PARTITION are equivalently hard. We first give an explicit polynomial-time reduction from PA to MIN-k-PARTITION, and then prove that it is in fact an AP-reduction.

Consider an arbitrary instance of the PA problem, specified by the cell-free massive MIMO system $S = (\mathcal{A}, U, A(k), \boldsymbol{\beta}, \Psi)$ with $|\mathcal{A}| = M$, $|U| = K$, $|\Psi| = \tau$. We construct an instance of MIN-k-PARTITION as follows: Define a *complete* graph $G = (V, E, \omega)$ (where $\omega : E \to \mathbb{R}_+$ is a function that maps the edges in our graph to positive real values), by setting $V = U$. By construction, we have $|V| = K$ and $|E| = \frac{K(K+1)}{2}$. For any edge $(k, k') \in E$, where $k, k' \in V$, we set

$$\omega((k,k')) = \sum_{m \in A(k)} (\beta_{k',m}/\beta_{k,m})^2 + \sum_{m' \in A(k')} (\beta_{k,m'}/\beta_{k',m'})^2 .$$

The weight function ω defined above satisfies a symmetric property, i.e., $\omega((k, k')) = \omega((k', k))$. It is important to note that we set the value of k that is referred to in the title of the MIN-k-PARTITION problem to τ. So we have constructed an instance of a MIN-τ-PARTITION problem.

($\Rightarrow$) If we have a solution to an instance of the PA problem, then we have minimized the optimization function given by Equation (2.5), which we state again:

$$\sum_{t=1}^{\tau} \sum_{k,k' \in V_t} \left(\sum_{m \in A(k)} (\beta_{k'm}/\beta_{km})^2 + \sum_{m' \in A(k')} (\beta_{km'}/\beta_{k'm'})^2 \right)$$

But notice that the objective function to be minimized in a general instance of the MIN-k-PARTITION problem is $\sum\limits_{t=1}^{k} \sum\limits_{i,j \in V_t} \omega((i,j))$. By the construction of our instance of the MIN-k-PARTITION problem, the users U form the vertices V of the graph and the cardinality τ of the set of pilots Ψ is k (which is in the title of the problem, denoting the number of subsets of the vertices). Replacing τ by k and the user indices k, k' by the arbitrary indices i, j to denote the vertices of our graph, we get that the function minimized by solving the PA instance is

$$\sum_{t=1}^{k} \sum_{i,j \in V_t} \left(\sum_{m \in A(i)} (\beta_{jm}/\beta_{im})^2 + \sum_{m' \in A(j)} (\beta_{im'}/\beta_{jm'})^2 \right) = \sum_{t=1}^{k} \sum_{i,j \in V_t} \omega((i,j))$$

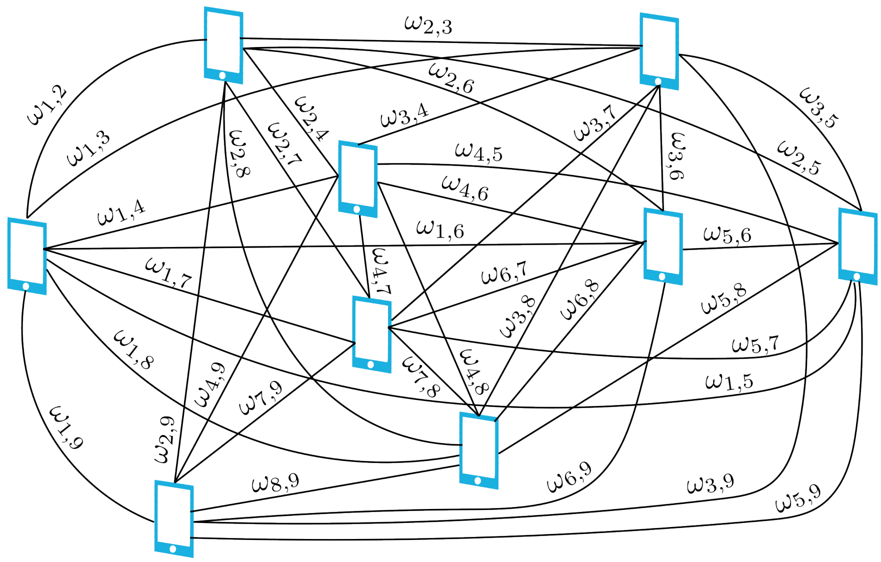
Therefore, we have a solution to our instance of the MIN-τ-PARTITION problem.

($\Leftarrow$) By the above argument, it is clear that if we have a solution to our constructed instance of the MIN-τ-PARTITION problem, we also have a solution to the corresponding PA instance by a reverse change of indices, as seen in Theorem 2.4.1.(ii).

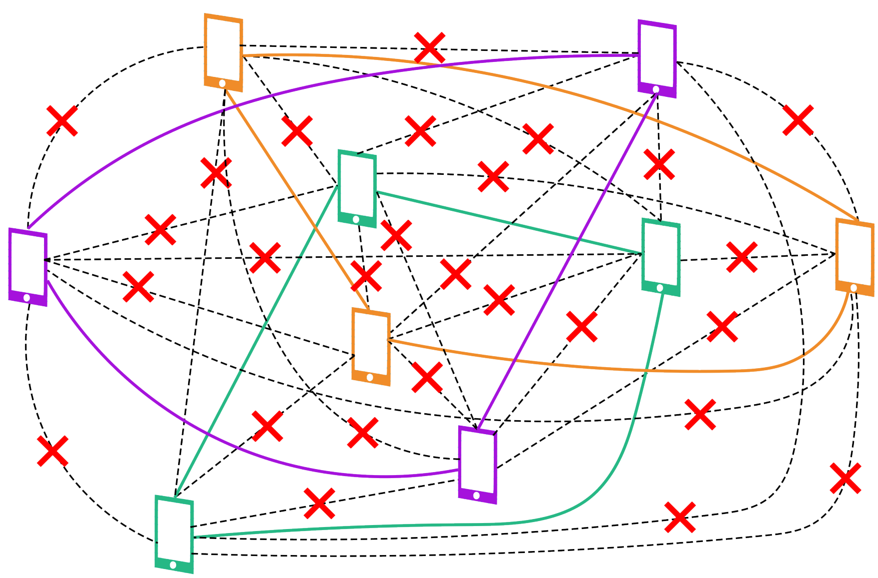
Hence, we have shown an explicit polynomial-time reduction from PA to MIN-k-PARTITION. What remains to be verified is that this is an AP-reduction. To see this, consider an instance of PA given by $S = \big(\mathcal{A}, U, \{A(k)\}_{k=1}^{K}, \boldsymbol{\beta}, \Psi\big)$. A general instance of MIN-k-PARTITION is given by an undirected graph $G = (V, E, \omega)$. In the above reduction, we have that G is a complete graph with $V = U$ and $\omega((i,j)) = \sum\limits_{m \in A(i)} (\beta_{j,m}/\beta_{i,m})^2 + \sum\limits_{m' \in A(j)} (\beta_{i,m'}/\beta_{j,m'})^2 \quad \forall\, i, j \in V$. Recall Definition 3.1.11. Thus we have $f : \big(\mathcal{A}, U, \{A(k)\}_{k=1}^{K}, \boldsymbol{\beta}, \Psi\big) \mapsto (U, E(K_U), w)$ where $E(K_U)$ denotes the set of edges of the complete graph on the vertices denoted by U, and $\omega((k,k')) = \sum\limits_{m \in A(k)} (\beta_{k',m}/\beta_{k,m})^2 + \sum\limits_{m' \in A(k')} (\beta_{k,m'}/\beta_{k',m'})^2$. Further, if we have obtained the partition $\mathcal{V} = \{V_1, V_2, \ldots, V_k\}$ of the vertex set V in our constructed instance of MIN-k-PARTITION, we get that the feasible assignment h which is the solution to the corresponding PA instance is defined as $h(k) = t$ such that $k \in V_t \quad \forall\, k \in U$. We set $g : (S, \mathcal{V}) \mapsto h$, where $h(i) = t$ such that $i \in V_t \in \mathcal{V}$. Notice that f and g do not depend on the performance ratio. Finally, for any instance S of PA, with $G = f(S)$ as the constructed instance of MIN-k-PARTITION, $\mathcal{V}$ as the solution of G, and $h = g(S, \mathcal{V})$ as the solution to the original instance S of PA, we see that $m_{\mathsf{PA}}(S, h) = m_{\mathrm{M}k\mathrm{P}}(G, \mathcal{V})$ (where MkP is short for MIN-k-PARTITION), by the nature of the constructed instances and solutions. We conclude that f and g satisfy the AP-condition with $\alpha = 1$, and thus the polynomial-time reduction described above is an AP-reduction. ■

As a consequence of this theorem, any existing performance ratios for approximating the solution to a MIN-k-PARTITION problem directly translate to the same performance ratios for approximating the solution to the corresponding Pilot Assignment problem.

The figures on the next page are inspired from Fig. 1 in [31], where the authors depict the mapping of a Pilot Assignment problem with 9 users and 3 pilots to a MAX-3-CUT problem:



(a) Min-3-Partition instance corresponding to the PA instance



(b) Solution of the Min-3-Partition instance. Edges of the same colour lie in the same subset, while those marked with a cross have endpoints in different subsets of the partition

Figure 3.3: The above figures show the reduction from a PA instance with $K = 9$ and $\tau = 3$ to its corresponding Min-3-Partition instance, and the solution of the Min-3-Partition instance.

3.2.2 Approximability results for PA

We state a few results on the approximability of MIN-k-PARTITION stated in [21, 12] as a lemma:

Lemma 3.2.2. *Assuming* P $\neq$ NP*, the following statements hold with respect to the computational complexity of* MIN-k-PARTITION*:*

(*i*) *The problem is not in* APX. [26]

(*ii*) *It is NP-hard to approximate* MIN-k-PARTITION *within* $O(|E|)$*, even when restricting the instances to graphs with* $|E| = \Omega(|V|^{2-\epsilon})$*, for a fixed* ϵ*,* $0 < \epsilon < 1$*.* [21]

(*iii*) *No polynomial time algorithm can achieve a better performance ratio than 1.058 in the case of* $k = 2$*.* [19]

(*iv*) *In case of* $k = 2$*, a polynomial time algorithm with a performance guarantee of* $\log |V|$ *is known.* [17]

(*v*) *In case of* $k = 3$*, a polynomial time algorithm with a performance guarantee of* $\epsilon |V|^2$ *for any* $\epsilon > 0$ *is known.* [21]

We now prove that the reduction we gave from MIN-k-PARTITION to PA in Section 2.4 is approximation preserving:

Lemma 3.2.3. *The polynomial-time reduction from* MIN-k-PARTITION *to PA given in Theorem 2.4.1.(ii) is an AP-reduction.*

Proof. Consider an instance $G = (V, E, \omega)$ of MIN-k-PARTITION, we need to determine the function f that maps it to an instance of PA. A general instance of PA is determined by the tuple $\big(\mathcal{A}, U, \{A(k)\}_{k=1}^{K}, \boldsymbol{\beta}, \Psi\big)$. From our reduction, we get that $f : (V, E, \omega) \mapsto \Big(\{1, \ldots, |V|, \ldots, M\},\ V,\ \{i\}_{i=1}^{|V|},\ \boldsymbol{\beta},\ \{1, \ldots, k\}\Big)$, where M is an arbitrary constant such that $M \gg |V|$, and the im-th element of the matrix $\boldsymbol{\beta}$ is $\beta_{im} = 1$ if $m = i$, $\sqrt{\frac{\omega((i,m))}{2}}$ if $m \neq i$ and $m \leq K$, and 0 otherwise. Moreover, if h is the feasible assignment that forms the solution of our constructed instance of PA, then the partition of vertices $\mathcal{V} = \{V_1, V_2, \ldots, V_k\}$ that forms the solution of our original MIN-k-PARTITION instance is defined as $V_t = \{i \,|\, h(i) = t\}$ where $i \in V$. We set $g : (G, h) \mapsto \mathcal{V}$, where $V_t = \{k \,|\, k \in U \text{ and } h(k) = t\} \quad \forall\ 1 \leq t \leq \tau$. Notice that f and g do not depend on the performance ratio.

Finally, for any instance G of MIN-k-PARTITION, with $S = f(G)$ as the constructed instance of PA, h as the solution of S, and $\mathcal{V} = g(G, \mathcal{V})$ as the solution to the original instance G of MIN-k-PARTITION, we see that $m_{\mathrm{M}k\mathrm{P}}(G, \mathcal{V}) = m_{\mathsf{PA}}(S, h)$ (where MkP is short for MIN-k-PARTITION), by the nature of the constructed instances and solutions. We conclude that f and g satisfy the AP-condition with $\alpha = 1$, and thus the polynomial-time reduction described above is an AP-reduction. ■

Using Theorem 3.2.1 and Lemmas 3.2.2 and 3.2.3, we are now ready to prove the following theorem on the approximability of the Pilot Assignment problem, inspired by Section 3.2.3 of [12]:

Theorem 3.2.4. *Assuming P $\neq$ NP, The following statements hold true for the Pilot Assignment problem :*

(*i*) *PA is in* exp-APX.

(*ii*) *An approximation of the PA problem within $\mathcal{O}(K^2)$ for $\tau \geq 3$ is impossible in polynomial time.*

(*iii*) *In the special case of $\tau = 2$, while no polynomial time algorithm can achieve a performance ratio better than 1.058, there exists an algorithm with a performance guarantee of* $\log |K|$ *in polynomial time.*

(*iv*) *In the special case of $\tau = 3$, there exists a polynomial time algorithm with a performance guarantee of $\epsilon|K|^2$ for any $\epsilon > 0$.*

Proof.

(*i*) From Lemmas 2.2.1 , 3.1.5 and 3.2.2.(i), the result follows immediately.

(*ii*) From Lemma 3.2.2.(ii), we have that MIN-k-PARTITION cannot be approximated within $\mathcal{O}(|E|)$ in polynomial time for $k \geq 3$. Further, in the PA problem, we typically deal with complete (or dense) graphs. So we have that $|E| = \Omega(K^2)$, where K is the number of users, or vertices in the graph. The number of pilots available, or the number of partitions required in the graph is τ. Hence, using Lemma 3.2.3, we get that unless P = NP, it is impossible to approximate PA within $\mathcal{O}(K^2)$ in polynomial time for $\tau \geq 3$.

(*iii*) The first part of this claim follows from Lemma 3.2.3 and Lemma 3.2.2.(iii), while the second part follows from Theorem 3.2.1 and Lemma 3.2.2.(iv).

(*iv*) As above, this follows easily from Theorem 3.2.1 and Lemma 3.2.2.(v). ■

Finally, we would like to remark the following: While the fact that MIN-k-PARTITION is AP-reducible to PA is used to prove negative approximation results for the Pilot Assignment problem, this cannot be applied to the positive approximation results on MIN-k-PARTITION to prove similar results for the Pilot Assignment problem. As is clear from Definition 3.1.11, we need the fact that PA is also AP-reducible to MIN-k-PARTITION to carry over explicit lower bounds for the MIN-k-PARTITION problem (from Lemma 3.2.2) to the Pilot Assignment problem.

Chapter 4

Conclusion and Open Problems

In this thesis, we explored the problem of pilot assignment in cell-free massive MIMO systems to reduce pilot contamination, so as to enable efficient data transmission between users in a wireless communication network. We did so from a theoretical computer science perspective, where our aim was to study the inherent hardness of the problem and find provable guarantees on the quality of achievable solutions to this problem. We defined the cell-free massive MIMO system mathematically, which is crucial in any theoretical study of this problem with problem instances and feasible solutions. In Chapter 2 and Chapter 3, we addressed the NP-hardness and the approximability of the PA optimization problem respectively. We proved that the Pilot Assignment problem is NP-hard, and at the same time, does not belong to the APX class of problems.

From our results in Chapter 3, we find it interesting to note that any research on the classical MIN-k-PARTITION problem can be directly applied to the PA problem. This thesis does not explore the SDP or LP formulations of the MIN-k-PARTITION problem, which is an active area of research. Further research in this area seems to be a step in the right direction towards finding better optimal solutions for the pilot assignment problem. It is also interesting to note that there has been very little research on the parameterized complexity of the MAX-k-CUT problem for $k \geq 3$, and more importantly, of the MIN-k-PARTITION problem for even the case $k = 2$. A practical question that arises here is if these problems even admit an XP algorithm for some suitable parameter. Such an algorithm may yield a Pilot Assignment scheme with superior performance, depending on the parameter used in the algorithm. In the same vein, *parameterized approximation algorithms* might be an intersting topic of research, which involves designing algorithms that aim to find approximate solutions to NP-hard optimization problems in time that is polynomial in the input size and a function of a specific parameter. Finally, a question of purely theoretical interest to address from our work in Chapter 3 is whether the MIN-k-PARTITION problem is NPO-complete, which makes it exp-APX complete. We suspect that this problem might be NPO-intermediate, that is, an optimization problem which is in the class NPO but is neither in the class PO nor NPO-complete under the assumption that P $\neq$ NP. Thus, we will be able to characterize the approximation complexity class of PA accordingly.

Bibliography

[1] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. 1998. Proof Verification and the Hardness of Approximation Problems. *J. ACM* 45, 3 (May 1998), 501–555. `https://doi.org/10.1145/278298.278306`

[2] Masoud Attarifar, Aliazam Abbasfar, and Angel Lozano. 2018. Random vs Structured Pilot Assignment in Cell-Free Massive MIMO Wireless Networks. In *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 1–6. `https://doi.org/10.1109/ICCW.2018.8403508`

[3] Giorgio Ausiello, Alberto Marchetti-Spaccamela, Pierluigi Crescenzi, Giorgio Gambosi, Marco Protasi, and Viggo Kann. 1999. *Complexity and Approximation* (1 ed.). Springer Berlin, Heidelberg. `https://doi.org/10.1007/978-3-642-58412-1`

[4] Yair Bartal, Moses Charikar, and Danny Raz. 2001. Approximating Min-Sum k-Clustering in Metric Spaces. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing* (Hersonissos, Greece) *(STOC '01)*. Association for Computing Machinery, New York, NY, USA, 11–20. `https://doi.org/10.1145/380752.380754`

[5] Babak Behsaz, Zachary Friggstad, Mohammad R. Salavatipour, and Rohit Sivakumar. 2019. Approximation Algorithms for Min-Sum k-Clustering and Balanced k-Median. *Algorithmica* 81, 3 (Mar 2019), 1006–1030. `https://doi.org/10.1007/s00453-018-0454-1`

[6] Stefano Buzzi, Carmen D'Andrea, Maria Fresia, Yong-Ping Zhang, and Shulan Feng. 2021. Pilot Assignment in Cell-Free Massive MIMO Based on the Hungarian Algorithm. *IEEE Wireless Communications Letters* 10, 1 (2021), 34–37. `https://doi.org/10.1109/LWC.2020.3020003`

[7] Shuaifei Chen, Jiayi Zhang, Jing Zhang, Emil Björnson, and Bo Ai. 2022. A survey on user-centric cell-free massive MIMO systems. *Digital Communications and Networks* 8, 5 (2022), 695–719. https://doi.org/10.1016/j.dcan.2021.12.005

[8] Sunil Chopra and M.R. Rao. 1995. Facets of the k-partition polytope. *Discrete Applied Mathematics* 61, 1 (1995), 27–48. https://doi.org/10.1016/0166-218X(93)E0175-X

[9] Sunil Chopra and M. R. Rao. 1993. The partition problem. *Mathematical Programming* 59 (1993), 87–115. https://doi.org/10.1007/BF01581239

[10] Artur Czumaj and Christian Sohler. 2007. Small Space Representations for Metric Min-Sum k-Clustering and Their Applications. In *STACS 2007, Proceedings of the 24th Annual Symposium on Theoretical Aspects of Computer Science* (Aachen, Germany) *(Lecture Notes in Computer Science)*. Springer-Verlag, Berlin, Heidelberg, 536–548. https://doi.org/10.1007/978-3-540-70918-3_46

[11] Zhiwen Duan, Feng Zhao, and Yingjie Wang. 2022. Pilot Allocation and Data Power Optimization Based on Access Point Selection in Cell-Free Massive MIMO. *Wirel. Commun. Mob. Comput.* 2022 (Jan 2022), 10 pages. https://doi.org/10.1155/2022/4044783

[12] Andreas Eisenblätter. 2001. *Frequency Assignment in GSM Networks: Models, Heuristics, and Lower Bounds.* Ph. D. Dissertation.

[13] Jamie Fairbrother and Adam N. Letchford. 2017. Projection results for the k-partition problem. *Discrete Optimization* 26 (2017), 97–111. https://doi.org/10.1016/j.disopt.2017.08.001

[14] A. Frieze and M. Jerrum. 1997. Improved Approximation Algorithms for MAX k-CUT and MAX BISECTION. *Algorithmica* 18, 1 (May 1997), 67–81. https://doi.org/10.1007/BF02523688

[15] M. R. Garey and D. S. Johnson. 1978. “Strong” NP-Completeness Results: Motivation, Examples, and Implications. *J. ACM* 25, 3 (July 1978), 499–508. https://doi.org/10.1145/322077.322090

[16] Michael R. Garey and David S. Johnson. 1990. *Computers and Intractability; A Guide to the Theory of NP-Completeness.* W. H. Freeman & Co., USA.

[17] Naveen Garg, Vijay V. Vazirani, and Mihalis Yannakakis. 1996. Approximate Max-Flow Min-(Multi)Cut Theorems and Their Applications. *SIAM J. Comput.* 25, 2 (1996), 235–251. https://doi.org/10.1137/S0097539793243016

[18] Bissan Ghaddar, Miguel F. Anjos, and Frauke Liers. 2011. A branch-and-cut algorithm based on semidefinite programming for the minimum k-partition problem. *Annals of Operations Research* 188 (2011), 155–174. https://doi.org/10.1007/s10479-008-0481-4

[19] Johan Håstad. 2001. Some Optimal Inapproximability Results. *J. ACM* 48, 4 (Jul 2001), 798–859. https://doi.org/10.1145/502090.502098

[20] Giovanni Interdonato, Emil Björnson, Hien Quoc Ngo, Pål Frenger, and Erik G. Larsson. 2019. Ubiquitous Cell-Free Massive MIMO Communications. *EURASIP Journal on Wireless Communications and Networking* 2019, 1 (Aug 2019), 1–13. https://doi.org/10.1186/s13638-019-1507-0

[21] Viggo Kann, Sanjeev Khanna, Jens Lagergren, and Alessandro Panconesi. 1997. *On the Hardness of Approximating Max K-Cut and Its Dual.* Technical Report.

[22] Na Li, Yuanyuan Gao, and Kui Xu. 2021. On the Optimal Energy Efficiency and Spectral Efficiency Trade-off of CF Massive MIMO SWIPT System. *EURASIP J. Wirel. Commun. Netw.* 2021, 1 (Aug 2021), 22 pages. https://doi.org/10.1186/s13638-021-02035-w

[23] Heng Liu, Jiayi Zhang, Shi Jin, and Bo Ai. 2020. Graph Coloring Based Pilot Assignment for Cell-Free Massive MIMO Systems. *IEEE Transactions on Vehicular Technology* 69, 8 (2020), 9180–9184. https://doi.org/10.1109/TVT.2020.3000496

[24] Hien Quoc Ngo, Alexei Ashikhmin, Hong Yang, Erik G. Larsson, and Thomas L. Marzetta. 2017. Cell-Free Massive MIMO Versus Small Cells. *IEEE Transactions on Wireless Communications* 16, 3 (2017), 1834–1850. https://doi.org/10.1109/TWC.2017.2655515

[25] Vilmar Jefté Rodrigues de Sousa, Miguel F. Anjos, and Sébastien Le Digabel. 2019. Improving the linear relaxation of maximum k-cut with semidefinite-based constraints. *EURO Journal on Computational Optimization* 7, 2 (2019), 123–151. https://doi.org/10.1007/s13675-019-00110-y

[26] Sartaj Sahni and Teofilo Gonzalez. 1976. P-complete approximation problems. *J. ACM* 23, 3 (07 1976), 555–565. https://doi.org/10.1145/321958.321975

[27] T. Schwengler. 2019. *Wireless and Cellular Communications (paperback)*. Lulu.com. https://books.google.co.in/books?id=afdbzQEACAAJ

[28] E.R. van Dam and R. Sotirov. 2016. New bounds for the max-k-cut and chromatic number of a graph. *Linear Algebra Appl.* 488 (2016), 216–234. https://doi.org/10.1016/j.laa.2015.09.043

[29] Guanglei Wang and Hassan L. Hijazi. 2017. Exploiting sparsity for the min k-partition problem. *Mathematical Programming Computation* 12 (2017), 109 – 130. https://doi.org/10.1007/s12532-019-00165-3

[30] Xudong Yin, Jianxin Dai, Jin yuan Wang, Junxi Zhao, and Chonghu Cheng. 2019. Pilot Allocation and Power Control in Cell-Free Massive MIMO Systems. *Wireless Personal Communications* 109 (2019), 2489 – 2506. https://doi.org/10.1007/s11277-019-06694-3

[31] Wenbo Zeng, Yigang He, Bing Li, and Shudong Wang. 2021. Pilot Assignment for Cell Free Massive MIMO Systems Using a Weighted Graphic Framework. *IEEE Transactions on Vehicular Technology* 70, 6 (2021), 6190–6194. https://doi.org/10.1109/TVT.2021.3076440

[32] Jiayi Zhang, Emil Björnson, Michail Matthaiou, Derrick Wing Kwan Ng, Hong Yang, and David J. Love. 2020. Prospective Multiple Antenna Technologies for Beyond 5G. *IEEE Journal on Selected Areas in Communications* 38, 8 (2020), 1637–1660. https://doi.org/10.1109/JSAC.2020.3000826

[33] Yao Zhang, Haotong Cao, Peng Zhong, Chang Qi, and Longxiang Yang. 2018. Location-Based Greedy Pilot Assignment for Cell-Free Massive MIMO Systems. In *2018 IEEE 4th International Conference on Computer and Communications (ICCC)*. IEEE, 392–396. https://doi.org/10.1109/CompComm.2018.8780756