AN ABSTRACT OF THE THESIS OF

Sagar Shrestha for the degree of Master of Science in Computer Science presented on March 3, 2023.

Title: Graph Imitation Learning For Optimal Joint Beamforming and Antenna Selection

Abstract approved: _____

Xiao Fu

Transmit beamforming is an important technique employed to improve efficiency and signal quality in wireless communication systems by steering signals towards their intended users. It often arises jointly with the antenna selection problem due to various reasons, such as limited number of radio frequency (RF) chains and energy/resource efficiency considerations. The joint robust beamforming and antenna selection (RBF&AS) problem is a mixed integer nonlinear program. Due to the NP-hard combinatorial nature of this problem, majority of existing methods rely on various heuristics, e.g., continuous approximations, greedy search, and supervised machine learning. However, these heuristics do not guarantee the optimality (or even feasibility) of the considered problem. To address this issue, we design an effective branch-and-bound (B&B) based method that guarantees optimal solutions to the problem of interest. To avoid the potentially costly nature of the proposed B&B algorithm, a machine learning-based scheme is proposed that expedites the B&B search by skipping intermediate steps of the algorithm. The learning model is based on a *graph neural network* (GNN) that provides resilience to commonly encountered problems in wireless comunications, namely, the change of problem size (e.g., the number of users) across the training and test stages. Finally, we provide a comprehensive theoretical analysis, which shows the proposed GNN-based method can reduce the complexity of the B&B method while retaining global optimality under reasonable conditions. Extensive numerical simulations show that the proposed

method can provide near-optimal solution with an order-of-magnitude speedup relative to the B&B.

Graph Imitation Learning For Optimal Joint Beamforming and Antenna Selection

by

Sagar Shrestha

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented March 3, 2023
Commencement June 2023

Master of Science thesis of Sagar Shrestha presented on March 3, 2023.

APPROVED:

Major Professor, representing Computer Science

Head of the School of Electrical Engineering and Computer Science

---

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Sagar Shrestha, Author

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor Dr. Xiao Fu for his unwavering support, invaluable guidance, and mentorship. His expertise and feedback have been critical to the success of this work. I would like to thank my committee members for their feedback and support to my work. I would also like to thank other faculty members who have supported and guided me throughout my academic journey. Finally, I would like to thank my family for their love and support throughout all my endeavors.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF APPENDIX FIGURES

Figure Page

Chapter 1: Introduction

Most of the materials of this thesis is from the following article:

Sagar Shrestha, Xiao Fu, and Mingyi Hong, "Optimal Solutions for Joint Beamforming and Antenna Selection: From Branch and Bound to Graph Neural Imitation Learning", in *IEEE Trans. Signal Process.*, doi: 10.1109/TSP.2023.3244096.

Proper re-organization and modifications were incorporated to enrich the details and to serve for the clarity of the thesis.

1.1 Background

In multi-antenna wireless communication systems, a base station (BS) uses multiple antennas to transmit private information to the respective users. By using channel state information (CSI) of the users, the base station (BS) can steer each message in the direction of its intended user by selecting different weights for the antenna elements. This procedure, called Beamforming, helps to boost signal strength and reduce interference at the receiver compared to transmitting isotropically. In the past decade, a plethora of beamforming algorithms have been proposed under various scenarios; see, e.g., [24, 26, 29, 51, 66, 67, 72].

An important problem that arises in practice is that the BS may not be able to operate all antennas simultaneously. Often, hardware elements called radio frequancy (RF) chains, required to operate antenna elements, are fewer in number than the antenna elements. This is due to the costly and power hungry nature of RF chains compared to antenna elements [21, 48, 51, 63]. Various other factors can also contribute to the limitation of simultaneously operable antennas—such as energy consumption considerations [4, 30], problem size reduction [55], overhead minimization [41], and algorithm accommodations [62]. Thus, one needs to consider the problem of jointly selecting a subset of antennas and the beamforming weights for maximizing efficiency and signal quality.

The problem is called joint beamforming and antenna selection (BF&AS) [29, 51, 66].

Jointly designing the beamformers and selecting antennas is a *mixed integer and nonlinear program*, which is known to be NP-hard [14, 35]. A large portion of the literature tackles this problem using continuous programming-based approximations. For example, [2, 29, 51, 66] used convex and nonconvex group sparsity-promoting regularization to encourage turning off antenna elements. However, the continuous approximations are often NP-hard problems as well (especially when the sparsity promotion is done via nonconvex quasi-norms as in [51]), and thus it is unclear if they can solve the problem of interest optimally. In addition, works using greedy methods to assist antenna selection also exist (see, e.g., [13, 18, 35, 47, 52]). But the optimality of joint (R)BF&AS is still not addressed in these works.

In recent years, machine learning (ML) approaches are employed to handle the joint BF and AS problem. In [28], a supervised learning approach was proposed. The basic idea is to use a continuous optimization algorithm to produce training pairs (i.e., channel matrices and sparse beamformers), and then learn a neural network-based regression function using such pairs. Similar ideas were used in [19, 71] with various settings. This type of approach in essence mimics the training pair-generating algorithms at best, and thus the optimality of their solutions is again not guaranteed.

1.2 Contributions

In this work, we revisit the joint BF and AS and its extension under imperfect channel state information (CSI), namely, the joint robust beamforming (RBF) and AS problem. We are interested in the unicast BF and RBF formulations in [6] and [73], respectively. The goal is to satisfy the users' quality-of-service (QoS) constraints while minimizing the power consumption, with only a subset of the antenna elements activated. Our detailed contributions are as follows:

- **Optimal Joint (R)BF&AS via Branch and Bound.** Our first contribution lies in an optimal computational framework to attain the *global optimal solutions* to the joint (R)BF&AS problems. To this end, we propose a *Branch and Bound* (B&B) [15, 36] framework that is tailored for the problems of interest. Our design leverages problem structures of unicast BF and RBF, which allows for branching

only on a subset of the optimization variables—thereby having reduced complexity and being effective. Unlike continuous optimization-based approximations in [2, 29, 51, 66] whose solutions are often sub-optimal or infeasible, the proposed B&B is guaranteed to return an optimal solution.

- **An ML-based Acceleration Scheme.** B&B is known for its relatively weak scalability. To improve efficiency, an idea from the ML community (see, e.g., [27, 56]) is to learn a binary classifier offline using multiple problem instances. The classifier determines whether or not any encountered intermediate state of the B&B algorithm could be "skipped", as skipping these states saves computational resources and expedites the B&B process. Generic ML learning functions (e.g., support vector machines (SVM)) used in existing works like [27, 37] do not reflect the problem structure in wireless communications. In this work, we propose a *graph neural network* (GNN) [64] based learning function designed to exploit the physics of the (R)BF problem—which offers an enhanced classification accuracy. More importantly, the GNN is agnostic to the change of scenarios (e.g., problem size) during training and testing. This feature is designed to meet the need of wireless communication systems, as the number of users served by a base station could change quickly in practice.

- **Theoretical Understanding.** We present comprehensive performance characterizations for the proposed approaches. In particular, we show that the ML-based acceleration retains the global optimality of the B&B procedure with high probability, under reasonable conditions. ML-based B&B acceleration has limited theoretical studies, and the results were developed under often overly ideal settings (e.g., convex classifier) [27, 61]. There is a lack of understanding of the impacts of key factors such as nonconvexity, limited training samples, and the employed ML model's structure. Our analysis takes important aspects into consideration, such as the nonconvexity of the GNN learning process, the GNN's structure and complexity, the GNN's function approximation error, and the amount of available samples. As a consequence, the analysis offers insights to reveal key trade-offs in practice.

1.3 Related Works

B&B was proposed for beamforming problems in [42, 43], and antenna selection problems in [20, 39, 58]. Particularly, the work in [42] considered a single group multicast beamforming problem, the work in [39] considered a joint power allocation and antenna selection problem, the work in [58] considered antenna selection-assisted rate maximization in wiretap channels, and [20, 21] considered receive antenna selection for sum rate maximization. However these are different from the QoS-constrained downlink transmit beamfroming formulation considered in our work, which requires new B&B designs. ML-based B&B acceleration so far has been mostly used for *mixed integer and linear programs* (MILPs) in the ML community, e.g., [23, 27], where the B&B design is standard. Such methods have also been adopted in wireless communications in [37, 65] where resource allocation tasks are framed as *mixed integer and nonlinear programs* (MINPs). However, the joint (R)BF&AS problem has not been considered. In addition, comprehensive theoretical understanding to such ML-acceleration procedures has been elusive.

1.4 Overview of the Thesis

The organization of this thesis is as follows:

Chapter 2 introduces the problem of (R)BF&AS and summarizes representative existing methods along with their challenges. Chapter 3 proposes an optimality guaranteed method based on B&B for solving the problem under consideration. Chapter 4 presents an acceleration scheme based on GNN and imitation learning for acclerating the proposed B&B procedure. Further, a comprehensive theoretical analysis of the resulting accerated B&B scheme is provided in Chapter 4. Chapter 5 provides extensive numerical results that demonstrate the efficacy of the proposed method and validates our theoretical analysis. Finally conclusion and discussions are provided in Chapter 6

1.5 Notation

x, $\boldsymbol{x}$ and $\boldsymbol{X}$ denote a scalar, a vector, and a matrix, respectively. $\boldsymbol{x}_n$ denotes the nth column of $\boldsymbol{X}$. We use the matlab notation $\boldsymbol{X}(n,:)$ to denote the nth row of $\boldsymbol{X}$. $[N]$ denotes the set $\{1, 2, \ldots, N\}$. $\|\boldsymbol{x}\|_2$, $\|\boldsymbol{x}\|_\infty$, $\|\boldsymbol{X}\|_2$, $\|\boldsymbol{X}\|_F$, $\|\boldsymbol{X}\|_{\text{row}-0}$ denote the vector ℓ_2 norm, vector ℓ_∞ norm, matrix spectral norm, matrix Frobenius norm, and the number

of non-zero rows in the matrix, respectively. $\mathrm{Tr}(\boldsymbol{X})$, $\boldsymbol{X}^H$, and $\boldsymbol{X}^\top$ denote the trace, hermitian, and transpose of $\boldsymbol{X}$. $|\mathcal{X}|$ denotes the cardinality of the set $\mathcal{X}$. $\mathbb{E}[\cdot]$ denotes the expectation operator. $\boldsymbol{X} \succeq \boldsymbol{0}$ denotes that $\boldsymbol{X}$ is positive semi-definite matrix. $\boldsymbol{X}(\mathcal{S},:)$ with $\mathcal{S} \subseteq [N]$ denotes the submatrix of $\boldsymbol{X} \in \mathbb{C}^{N\times M}$ containing only the rows of $\boldsymbol{X}$ contained in the set $\mathcal{S}$. $\boldsymbol{X}_{-n}$ denotes the submatrix of $\boldsymbol{X}$ with the nth column removed. $\boldsymbol{f}(\cdot)$ is C-Lipschitz continuous iff $\|\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{y})\|_2 \leq C\|\boldsymbol{x} - \boldsymbol{y}\|_2$.

Chapter 2: Problem Statement and Challenges

We consider a typical downlink communication scenario where a *base station* (BS) is serving M single antenna users. We suppose that the BS has N antennas. Denote $\boldsymbol{w}_m \in \mathbb{C}^N$ as the beamforming vector for serving user m. The message signal for user m is represented by $s_m(t)$. Given the channel $\boldsymbol{h}_m \in \mathbb{C}^N$ between the BS and user m, the signal received by user m can be expressed as follows:

$$y_m(t) = \underbrace{\boldsymbol{h}_m^H \boldsymbol{w}_m s_m(t)}_{\text{signal}} + \underbrace{\sum_{\ell \neq m} \boldsymbol{h}_m^H \boldsymbol{w}_\ell s_\ell(t)}_{\text{interference}} + \underbrace{n_m}_{\text{noise}},$$

where n_m is zero-mean circular symmetric white Gaussian noise with variance σ_m^2. Assume w.l.o.g. that $\{s_m(t)\}_{m=1}^M$ are mutually uncorrelated and temporally white with zero-mean and unit-variance. Beamforming scenario for 3 antennas and 3 users is depicted in Figure 2.1. The total transmission power is given by

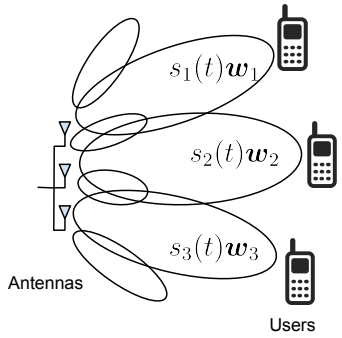

Figure 2.1: Illustraction of beamforming with 3 antennas and 3 users

$$\sum_{m=1}^{M} \|\boldsymbol{w}_m\|_2^2 := \|\boldsymbol{W}\|_{\mathrm{F}}^2,$$

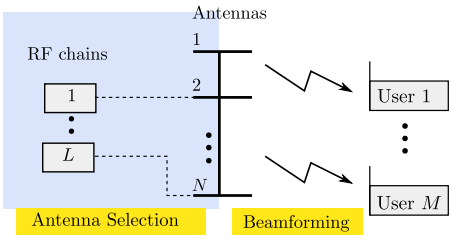

Figure 2.2: Downlink communication scenario depicting antenna selection and beamforming.

where $\boldsymbol{W} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_M]$. The *signal to interference and noise ratio* (SINR) at the mth receiver is expressed as:

$$\text{SINR}_m = \frac{|\boldsymbol{w}_m^H \boldsymbol{h}_m|_2^2}{\sum_{\ell \neq m} |\boldsymbol{w}_\ell^H \boldsymbol{h}_m|_2^2 + \sigma_m^2}. \tag{2.1}$$

In this work, our interest lies in a scenario where the BS only activates L anntennas as shown in Figure 2.2. Hence, we aim to select a subset $\mathcal{A} \subseteq \{1, \ldots, N\}$ such that $|\mathcal{A}| \leq L$ and $\boldsymbol{w}_m(i) \neq 0$ only when $i \in \mathcal{A}$. Thus we want to design $\boldsymbol{w}_m$ such that $\boldsymbol{w}_m(n) = 0, \forall n \in \mathcal{A}, \forall m \in [M]$.

2.1 Unicast Beamforming and SOCP

One of the most popular formulations for beamforming is the so-called Quality of Service (QoS) formulation [32,60,70] that tries to maintain a pre-specified SINR level for all users while minimizing the total power consumed at the BS. When $\boldsymbol{h}_m$ is known, the unicast BF problem can be formulated as follows:

$$\underset{\boldsymbol{W}}{\text{minimize}} \quad \|\boldsymbol{W}\|_F^2 \tag{2.2a}$$

$$\text{subject to} \quad \frac{|\boldsymbol{w}_m^H \boldsymbol{h}_m|^2}{\sum_{\ell \neq m} |\boldsymbol{w}_\ell^H \boldsymbol{h}_m|^2 + \sigma_m^2} \geq \gamma_m, \quad m \in [M], \tag{2.2b}$$

where γ_m is the pre-specified SINR for mth user. Problem (2.2) is called "unicast" BF because every user receives its own message, in contrast to "multicast" where each group of users receive a common message. Problem (2.2) appears to be nonconvex, but it can

be recast as a *second-order cone program* (SOCP). The following lemma shows that one can re-write (2.2b) as a second-order cone constraint:

Lemma 1 ([6]). *Eq.* (2.2b) *can be equivalently written as a second-order cone constraint:*

$$\frac{1}{\sqrt{\gamma_m \sigma_m^2}} \mathrm{Re}(\boldsymbol{w}_m^H \boldsymbol{h}_m) \geq \sqrt{\sum_{\ell \neq m} |\boldsymbol{w}_\ell^H \boldsymbol{h}_m|^2 + 1}, \tag{2.3}$$

for all $m \in [M]$. *Therefore, any algorithm for solving SOCP can be used to solve* (2.2) *optimally.*

The intuition behind the proof of Lemma 1 is to note that the absolute value $|\boldsymbol{w}_m^H \boldsymbol{h}_m|^2$ is equal for any rotation of $\boldsymbol{w}_m$, i.e., $\exp(i\theta_m)\boldsymbol{w}_m$. This allows us to select θ_m such that $\boldsymbol{w}_m^H \boldsymbol{h}_m$ is a real number. Hence one can obtain (2.3) by taking square root on both sides of (2.2b). The fact that (2.2) is a convex program allows us to solve it efficiently using any convex optimization method. This will be instrumental in the development of our proposed B&B algorithm in Chapter 3

2.2 Robust Beamforming and SDR

In practice, the BS cannot have perfect knowledge of CSI at the users. The CSI is usually estimated using feedback information from the users [44, 49]. In this scenario, the following worst-case RBF formulation is often considered [10, 33, 46, 68, 73]:

$$\underset{\boldsymbol{W}}{\text{minimize}} \ \ \|\boldsymbol{W}\|_F^2 \tag{2.4a}$$

$$\text{subject to} \ \ \min_{\overline{\boldsymbol{h}}_m \in \mathcal{U}_m} \frac{|\boldsymbol{w}_m^H \overline{\boldsymbol{h}}_m|^2}{\sum_{\ell \neq m} |\boldsymbol{w}_\ell^H \overline{\boldsymbol{h}}_m|^2 + \sigma_m^2} \geq \gamma_m, \quad \forall m \in [M], \tag{2.4b}$$

where $\mathcal{U}_m := \{\boldsymbol{h}_m + \boldsymbol{e}_m \mid \|\boldsymbol{e}_m\|_2 \leq \varepsilon_m\}$ is the uncertainty region around the approximate channel in which the true channel vector lies, $\boldsymbol{h}_m$ is the approximate channel vector available at the BS, and ε_m is the bound on the approximation error. Problem (2.4) cannot be directly converted to a convex program as in the perfect CSI case (cf. Lemma 1). However, Problem (2.4) can be tackled by a convex relaxation technique, namely, *semidefnite*

relaxation (SDR) [45]. Let $\boldsymbol{W}_m := \boldsymbol{w}_m \boldsymbol{w}_m^H$. Then the SDR of (2.4) is given by

$$\begin{aligned} \underset{\{\boldsymbol{W}_m \in \mathbb{C}^{N\times N}\}_{m=1}^M}{\text{minimize}} \quad & \sum_{i=1}^M \mathrm{Tr}(\boldsymbol{W}_m) && \text{(2.5a)} \\ \text{subject to} \quad & \min_{\overline{\boldsymbol{h}}_m \in \mathcal{U}_m} \frac{\overline{\boldsymbol{h}}_m^H \boldsymbol{W}_m \overline{\boldsymbol{h}}_m}{\sum_{j\neq m} \overline{\boldsymbol{h}}_m^H \boldsymbol{W}_j \overline{\boldsymbol{h}}_m + \sigma_m^2} \geq \gamma_m, && \text{(2.5b)} \\ & \boldsymbol{W}_m \succeq \boldsymbol{0}, \quad \forall m \in [M]. \end{aligned}$$

Note that (2.5) and (2.4) are equivalent if the constraint $\boldsymbol{W}_m = \boldsymbol{w}_m \boldsymbol{w}_m^H$ (or, $\mathrm{rank}(\boldsymbol{W}_m) = 1$) has not been relaxed. The semi-infinite constraint (2.5b) can be equivalently written as a linear matrix inequality using the $\mathcal{S}$-lemma (see [46] for more details):

$$\begin{aligned} \underset{\{\boldsymbol{W}_m, \boldsymbol{Y}_m\}_{m=1}^M, \boldsymbol{t}}{\text{minimize}} \quad & \sum_{m=1}^M \mathrm{tr}(\boldsymbol{W}_m) && \text{(2.6)} \\ \text{subject to} \quad & \boldsymbol{Y}_m = \begin{bmatrix} \boldsymbol{Q}_m + t_m \boldsymbol{I} & \boldsymbol{r}_m \\ \boldsymbol{r}_m^H & s_m - t_m \varepsilon_m^2 \end{bmatrix}, \quad m \in [M], \\ & \boldsymbol{Y}_m \succeq 0, \ \boldsymbol{W}_m \succeq 0, t_m \geq 0 \quad m \in [M], \\ \text{where} \quad & \boldsymbol{Q}_m = \frac{1}{\gamma_m} \boldsymbol{W}_i - \sum_{j\neq i} \boldsymbol{W}_j \\ & \boldsymbol{r}_m = \boldsymbol{Q}_m \boldsymbol{h}_m \\ & s_m = \boldsymbol{h}_m^H \boldsymbol{Q}_m \boldsymbol{h}_m - \sigma_m^2. \end{aligned}$$

Interestingly, this relaxation procedure turns out to be tight under reasonable conditions:

Lemma 2 ([46, Theorem 1])**.** *Suppose that Problem* (2.4) *is feasible. Let* $\boldsymbol{\Pi}_m := \boldsymbol{I} - \boldsymbol{H}_{-m}(\boldsymbol{H}_{-m}^H \boldsymbol{H}_{-m})^{-1} \boldsymbol{H}_{-m}^H$ *be the orthogonal complement projector of* $\boldsymbol{H}_{-m}$*. If*

$$\frac{\|\boldsymbol{\Pi}_m \boldsymbol{h}_m\|_2^2}{\varepsilon_m^2} > 1 + M + (M - \frac{1}{M})\gamma_m, \forall m, \tag{2.7}$$

then the optimal solution of (2.4) *can be obtained using SDR.*

The condition in (2.7) means that if the downlink channels associated with different

users are sufficiently different, then the SDR is tight.

When the condition of Lemma 2 is satisfied, the optimal beamforming matrix $\boldsymbol{W}^\star = [\boldsymbol{w}_1^\star, \boldsymbol{w}_2^\star, \ldots, \boldsymbol{w}_M^\star]$ can be obtained from optimal solution, $\boldsymbol{W}_m^\star, \forall m$, to the problem (2.6) as follows:

$$\boldsymbol{w}_m^\star = \sqrt{\lambda_1^{(m)}}\boldsymbol{v}_1^{(m)}, \forall m \in [M], \tag{2.8}$$

where λ_1 and $\boldsymbol{v}_1$ are the principal eigenvalue and eigenvector of $\boldsymbol{W}_m^\star$, respectively.

2.3 Joint (R)BF&AS: Existing Approaches

The joint (R)BF&AS problem frequently arises in practice for many reasons. For example, due to the the costly and power-hungry nature of RF chains, in some antenna arrays, the number of RF chains may be fewer than that of the antenna elements [21, 48, 51, 63]. Furthermore, AS is also used for energy-efficiency considerations [30], problem size reduction, overhead control, and algorithm design accommodations—see, e.g., [41,54,55,62,63] and the discussions therein. The problem considered in this work is as follows:

$$\begin{aligned} \underset{W}{\text{minimize}} \ & \|\boldsymbol{W}\|_F^2 && \text{(2.9a)} \\ \text{subject to } & \mathcal{C}(\boldsymbol{w}_m, \boldsymbol{h}_m, \varepsilon_m, \sigma_m) \geq \gamma_m, && \text{(2.9b)} \\ & \|\boldsymbol{W}\|_{\text{row-0}} \leq L. && \text{(2.9c)} \end{aligned}$$

where the row-0 function $\|\cdot\|_{\text{row-0}}$ counts the number of nonzero rows in $\boldsymbol{W}$ and

$$\mathcal{C}(\boldsymbol{w}_m, \boldsymbol{h}_m, \varepsilon_m, \sigma_m) := \begin{cases} \frac{|\boldsymbol{w}_m^H \boldsymbol{h}_m|^2}{\sum_{\ell \neq m} |\boldsymbol{w}_\ell^H \boldsymbol{h}_m|^2 + \sigma_m^2}, & \text{if BF is considered,} \\ \min_{\overline{\boldsymbol{h}} \in \mathcal{U}_m} \frac{|\boldsymbol{w}_m^H \overline{\boldsymbol{h}}_m|^2}{\sum_{\ell \neq m} |\boldsymbol{w}_\ell^H \overline{\boldsymbol{h}}_m|^2 + \sigma_m^2}, & \text{if RBF is considered.} \end{cases}$$

Problem (2.9) is a non-convex combinatorial problem, and it is NP-hard [14]. Some representative approaches for tackling joint (R)BF&AS problems are as follows:

2.3.1 Continuous Approximations

In the literature, Problem (2.9) and other joint (R)BF&AS formulations are often handled by continuous approximation. For example, a representative continuous approximation technique was used in [51] for handling a multicast version of (2.9). Using the idea from [51], one can recast the unicast problem in (2.9) as a regularized formulation as follows:

$$\begin{aligned} \underset{\boldsymbol{W}}{\text{minimize}} \;\; & \|\boldsymbol{W}\|_F^2 + \lambda \|\boldsymbol{W}\|_{\text{row-0}} \\ \text{subject to} \;\; & \mathcal{C}(\boldsymbol{w}_m, \boldsymbol{h}_m, \varepsilon_m, \sigma_m) \geq \gamma_m, \quad m \in [M]. \end{aligned} \tag{2.10}$$

The idea in [51] is to approximate the row-0 function using a group sparsity-inducing norm, namely, the $\ell_{\infty,1}$ norm, i.e., $\|\boldsymbol{W}\|_{\text{row-0}} \approx \sum_{n=1}^{N} \|\boldsymbol{W}(n,:)\|_\infty$ and its nonconvex counterpart $\|\boldsymbol{W}\|_{\text{row-0}} \approx \sum_{n=1}^{N} \log\left(\|\boldsymbol{W}(n,:)\|_\infty + \varepsilon\right)$ [9]. Similar ideas were used in [2]. Such continuous approximations allow the use of standard nonlinear program techniques to tackle (2.10). However, as mentioned, these methods do not provide any optimality guarantees. In addition, the feasiblity of $\boldsymbol{W}$ is often not met by the approximate solutions.

2.3.2 Greedy Methods

A number of greedy approaches also exist for tackling various formulations of the joint (R)BF&AS problem; see, e.g., [13, 18, 35, 47, 52]. The major idea is to activate or shut down an antenna in every iteration using a certain criterion that is often defined by the optimization problem's objective function—see an example in Chapter 5.2.1. Notably, such greedy algorithms are not necessarily computationally light, as will be seen in our simulations.

2.3.3 Supervised Learning

More recently, a number of learning-based approaches are proposed to tackle the joint (R)BF&AS problem; see, e.g., [11, 28, 31]. In [28], a multicast version of (2.9) was considered. The idea is to use an existing joint multicast BF&AS algorithm (e.g., the

algorithm from [51]) to generate "training pairs" $\{\boldsymbol{H}_t, \widehat{\boldsymbol{W}}_t\}_{t=1}^T$ by simulating a large number of problem instances, where t is the instance index, $\widehat{\boldsymbol{W}}_t$ is a (row-sparse) solution produced by the training pair-generating algorithm, and $\boldsymbol{H}_t$ is the channel matrix of instance t. Note that the training pairs can take other forms, e.g., $\{\boldsymbol{H}_t, \widehat{\boldsymbol{z}}_t\}$ where $\widehat{\boldsymbol{z}}_t \in \mathbb{R}^M$ is a binary vector found by the training pair-producing algorithm, indicating which antenna is activated [28, 71]. Then, a deep neural network (DNN) $\boldsymbol{f}_{\boldsymbol{\theta}}(\cdot)$ is trained via

$$\widehat{\boldsymbol{\theta}} \leftarrow \arg\min_{\boldsymbol{\theta}} \ \frac{1}{T}\sum_{t=1}^{T} \ell(\widehat{\boldsymbol{W}}_t, \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{H}_t)), \tag{2.11}$$

where $\boldsymbol{\theta}$ represents the parameters of the DNN and $\ell(x, y)$ measures the divergence between x and y. When a new $\boldsymbol{H}$ is seen in the test stage, one can use the learned DNN to predict the solution, i.e., $\widehat{\boldsymbol{W}} = \boldsymbol{f}_{\widehat{\boldsymbol{\theta}}}(\boldsymbol{H})$. This "supervised learning" idea is similar to a line of work in deep learning-based wireless system design; see, e.g., [38, 69]. Notably, it cannot exceed the performance of the algorithm that produces the training pairs or ensure producing a feasible solution in the test stage. Other deep learning-based ideas were seen in [11, 19, 31, 40] using either supervised learning or unsupervised learning variants, but similar challenges remain.

2.4 Summary

In this chapter, we have formulated the problem of unicast beamforming and antenna selection along with its robust version. If we omit the antenna selection part, the unicast beamforming problem can be re-written as a SOCP. Similarly, under reasonable conditions, the SDR of the robust unicast beamforming problem is tight. Hence, one can solve the (R)BF&AS problem by solving $\binom{N}{L}$ convex problems (SOCP/SDR). However, this is not feasible due to the exponential time complexity with respect to the number of antennas. Representative existing methods attempt to resolve this issue by resorting to approximate solutions. Hence, the issue of optimality remains. The following chapter is devoted to addressing this issue by proposing an optimality guaranteed B&B procedure.

Chapter 3: Optimal Joint (R)BF&AS via B&B

A natural idea for solving hard optimization problems is to employ a *global optimization* technique, e.g., the B&B procedure [8, 15, 36] described in Chapter **??**. Designing a practically working B&B algorithm is often an art—it normally involves judicious exploitation of problem structures. That is, not every hard problem enjoys an efficient B&B algorithm. Nonetheless, as we will see, the special properties of BF and RBF allows for an effective B&B design.

3.1 Preliminaries of B&B

We follow the notations from the tutorial in [8] to give a brief overview of B&B's design principles. Consider a nonconvex problem:

$$\underset{\boldsymbol{x}}{\text{minimize}}\ f(\boldsymbol{x}) \tag{3.1a}$$

$$\text{subject to } \boldsymbol{x} \in \mathcal{X}. \tag{3.1b}$$

where both the objective function and the constraint can be nonconvex. Suppose that there is a partition of the space $\mathcal{X} = \mathcal{X}_1 \cup \ldots \cup \mathcal{X}_S$, and that lower and upper bounds of $f(\boldsymbol{x})$ over each $\mathcal{X}_i$ are easier to find (relative to directly solving (3.1)). Let $\Phi_{\rm lb}(\mathcal{X}_i)$ and $\Phi_{\rm ub}(\mathcal{X}_i)$ be the algorithms that return lower and upper bounds of the optimal solution of (3.1) over the set $\mathcal{X}_i$, respectively. Then, the following holds:

$$l_G := \min_{1\le i\le S} \Phi_{\rm lb}(\mathcal{X}_i) \le \Phi(\mathcal{X}) \le \min_{1\le i\le S} \Phi_{\rm ub}(\mathcal{X}_i) =: u_G. \tag{3.2}$$

where $\Phi(\mathcal{X})$ represents the optimal solution of (3.1) over the feasible region $\mathcal{X}$. l_G and u_G are the global lower and upper bounds of the optimal solution $\Phi(\mathcal{X})$.

A premise of the success of B&B is that one could find a partition $\mathcal{X}_i$ for $i = 1, \ldots, S$

and a pair of functions $\Phi_{\rm lb}$ and $\Phi_{\rm ub}$ which can make the following hold:

$$\min_{1\le i\le S} \Phi_{\rm ub}(\mathcal{X}_i) - \min_{1\le i\le S} \Phi_{\rm lb}(\mathcal{X}_i) \le \epsilon \tag{3.3}$$

where $\epsilon > 0$ is a pre-specified error tolerance parameter. Algorithm 1 describes the basic working of the B&B algorithm for solving (3.1). One can see that we have maintained a list of unbranched nodes (subsets) in $\mathcal{F}^{(t)}$. In each iteration, we select a node from this list and divide it into smaller subsets. The process of selecting the node and dividing it is referred to as "branching".

Algorithm 1 B&B FRAMEWORK

$t = 0$;
$\mathcal{F}^{(0)} = \{\mathcal{X}\}$;
$l_G^{(t)} = \Phi_{\rm lb}(\mathcal{X})$;
$u_G^{(t)} = \Phi_{\rm lb}(\mathcal{X})$;
while $u_G^{(t)} - l_G^{(t)} > \varepsilon$ **do**
  Select a node $\widetilde{\mathcal{X}} \in \mathcal{F}^{(t)}$ using some heuristics;
  Divide $\widetilde{\mathcal{X}}$ into two subsets such that $\widetilde{\mathcal{X}}_1 \cup \widetilde{\mathcal{X}}_2 = \widetilde{\mathcal{X}}$;
  $\mathcal{F}^{(t+1)} = (\mathcal{F}^{(t)} \backslash \widetilde{\mathcal{X}}) \cup \{\widetilde{\mathcal{X}}_1, \widetilde{\mathcal{X}}_2\}$;
  $l_G^{(t)} = \min_{\overline{\mathcal{X}} \in \mathcal{F}^{(t+1)}} \Phi_{\rm lb}(\overline{\mathcal{X}})$;
  $u_G^{(t)} = \min_{\overline{\mathcal{X}} \in \mathcal{F}^{(t+1)}} \Phi_{\rm ub}(\overline{\mathcal{X}})$;
  $t = t + 1$;
end while

The effectiveness of B&B relies on two key factors. First, the design of the lower and upper bounding algorithms represented by $\Phi_{\rm lb}(\mathcal{X}_i)$ and $\Phi_{\rm ub}(\mathcal{X}_i)$, respectively, plays a central role. Second, the method of branching also matters. It often requires a problem-specific way to progressively and judiciously partition the constraint set $\mathcal{X}$ (usually from rough to fine-grid), so that the difference in (3.3) could shrink quicker than exhaustive search. Meeting either of the design requirements is not necessarily easy. Moreover, the key designs in B&B algorithms (e.g., the $\mathcal{X}$ partition strategies) are highly problem-dependent; that is, there is hardly a "standard recipe" for B&B algorithm design.

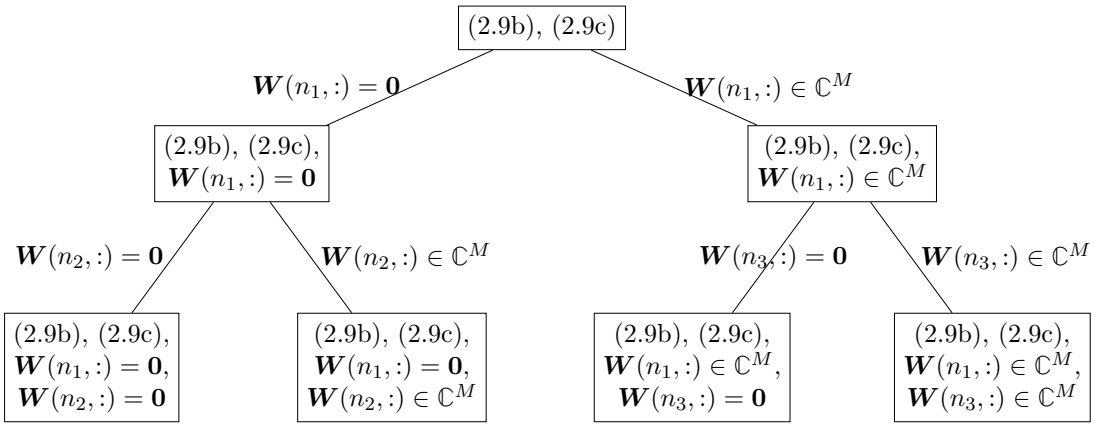

Figure 3.1: Illustration of B&B tree for problem (2.9). Here $n_i \in [N]$ are the branching variables selected at each node.

3.2 Proposed B&B for Joint (R)BF&AS

Problem (2.9) involves optimization in discrete and continuous spaces in the constraints. Designing a B&B algorithm for such problems can be difficult due to the large search space that consists of both types of constraints. However, the special structure of (R)BF in (2.9) allows us to efficiently obtain bounds over the entire range of the values of the continuous (beamforming) parameters once the discrete (antenna selection) parameters have been chosen; this will be clearer in (3.5) and (3.6). As such, we only need to construct a B&B tree over the discrete space.

3.2.1 B&B Tree Construction

We illustrate the idea of systematically partitioning the feasible region of Problem (2.9) in Figure 3.1. Here, $\mathcal{N}_i^{(\ell)}$ denotes the feasible region corresponding to the ith node at the ℓth level. In the sequel, we will use the term "node" and the associated feasible region interchangeably. The root is denoted as $\mathcal{N}^{(0)}$, and we have

$$\mathcal{N}^{(0)} = \{\boldsymbol{W} \mid \boldsymbol{W} \text{ satisfies (2.9b), (2.9c)}\}.$$

In the first level, the region represented by the root node is split into two regions represented by two child nodes, namely,

$$\mathcal{N}_1^{(1)} = \{\boldsymbol{W} \mid \boldsymbol{W}(n_1,:) = \mathbf{0},\ \boldsymbol{W} \text{ satisfies (2.9b), (2.9c)}\}$$
$$\mathcal{N}_2^{(1)} = \{\boldsymbol{W} \mid \boldsymbol{W}(n_1,:) \in \mathbb{C}^M,\ \boldsymbol{W} \text{ satisfies (2.9b), (2.9c)}\}.$$

where $n_1 \in [N]$ is an antenna index selected by a designed antenna selection criterion (e.g., via random sampling). Up to the first level of the tree, the status ("include (activate)" or "exclude (shut down)") of all antennas other than antenna n_1 have not been decided.

Note that the nodes in the B&B tree could constitute a partition in various forms. For example, for nodes in the same level, we have

$$\mathcal{N}_1^{(\ell)} \cup \ldots \cup \mathcal{N}_{S_\ell}^{(\ell)} = \mathcal{N}^{(0)},$$

where $S_\ell = 2^\ell$ is the number of nodes in the ℓth level of the tree. In addition, we have

$$\mathcal{N}_s^{(\ell)} = \mathcal{N}_{s_1}^{(\ell+1)} \cup \mathcal{N}_{s_2}^{(\ell+1)}, \tag{3.4}$$

where $s_1 := 2(s-1)+1$ and $s_2 := 2(s-1)+2$ represent the left and right children developed from $\mathcal{N}_s^{(\ell)}$ in the full tree. In fact, the children of $\mathcal{N}_s^{(\ell)}$ in any level and $\mathcal{N}_{-s}^{(\ell)}$ also present a partition of the root node, where $\mathcal{N}_{-s}^{(\ell)}$ is the union of $\mathcal{N}_1^{(\ell)}, \ldots, \mathcal{N}_{S_\ell}^{(\ell)}$ with $\mathcal{N}_s^{(\ell)}$ excluded.

The B&B algorithm starts from the first level to compute lower and upper bounds of (2.9) over the node-defined regions. Then, the B&B algorithm picks a node to "branch", i.e., to further partition oftentimes using a heuristic-based metric; see [15]. Going deeper in the tree towards the final leaves will allow us to progressively decide which antennas to activate or shut off. Let t denote the iteration index of the B&B algorithm, where an iteration corresponds to a branching (partitioning a node) operation. Use $\mathcal{P}^{(t)}$ to denote the collection of (s,ℓ) corresponding to the unbranched nodes. Then, the union of $\mathcal{N}_s^{(\ell)}$'s for $(s,\ell) \in \mathcal{P}^{(t)}$ represents a partitioning of the root in iteration t. In each iteration t, the stopping criterion in (3.3) is evaluated. It follows that the following two quantities

need to be evaluated:

$$l_G^{(t)} = \min_{(s,\ell)\in\mathcal{P}^{(t)}} \Phi_{\rm lb}(\mathcal{N}_s^{(\ell)}),\ u_G^{(t)} = \min_{(s,\ell)\in\mathcal{P}^{(t)}} \Phi_{\rm ub}(\mathcal{N}_s^{(\ell)}),$$

where $l_G^{(t)}$ and $u_G^{(t)}$ are the global lower and upper bounds in iteration t. In particular, the lower and upper bounds over the *newly* created two child nodes need to be found—since other nodes have been evaluated in a certain previous iteration. The hope is that one would not need to visit all nodes of tree before reaching the stopping criterion in (3.3).

3.2.2 Lower and Upper Bounds

In order to compute $\Phi_{\rm lb}(\mathcal{N}_s^{(\ell)})$ and $\Phi_{\rm ub}(\mathcal{N}_s^{(\ell)})$, let us define $\mathcal{A}_s^{(\ell)} \subseteq [N]$ and $\mathcal{B}_s^{(\ell)} \subseteq [N]\backslash\mathcal{A}_s^{(\ell)}$ to be the index sets of the antennas that have been activated and shut down at node s in level ℓ, respectively. Note that $\mathcal{A}_s^{(\ell)} \cup \mathcal{B}_s^{(\ell)} \subseteq [N]$ constitute the set of decided antennas at the node. Then, finding the upper and lower bounds of $\|\boldsymbol{W}\|_F^2$ at this node amounts to finding those of the following optimization problem:

$$\begin{aligned} \underset{\boldsymbol{W}}{\text{minimize}}\ & \|\boldsymbol{W}\|_F^2 \\ \text{subject to}\ & \mathcal{C}(\boldsymbol{w}_m, \boldsymbol{h}_m, \varepsilon_m, \sigma_m) \geq \gamma_m,\ \forall m, \\ & \boldsymbol{W}(n,:) = \boldsymbol{0}, \quad \forall n \in \mathcal{B}_s^{(\ell)}, \\ & \boldsymbol{W}(n,:) \in \mathbb{C}^M, \quad \forall n \in \mathcal{A}_s^{(\ell)}, \\ & \|\boldsymbol{W}\|_{\rm row-0} \leq L, \quad n \in [N]. \end{aligned} \tag{3.5}$$

For any given node $\mathcal{N}_s^{(\ell)}$, the lower bound can be obtained by solving the following relaxation of (3.5):

$$\Phi_{\rm lb}(\mathcal{N}_s^{(\ell)}) = \underset{\boldsymbol{W}}{\text{minimize}}\ \|\boldsymbol{W}\|_F^2 \tag{3.6a}$$

$$\begin{aligned} \text{subject to}\ & \mathcal{C}(\boldsymbol{w}_m, \boldsymbol{h}_m, \varepsilon_m, \sigma_m) \geq \gamma_m, \forall m, \\ & \boldsymbol{W}(n,:) = \boldsymbol{0}, \quad \forall n \in \mathcal{B}_s^{(\ell)}, \end{aligned} \tag{3.6b}$$

where we have dropped $\|\boldsymbol{W}\|_{\rm row-0} \leq L$. If Problem (3.6) is not feasible, $\Phi_{\rm lb}(\mathcal{N}_s^{(\ell)})$ is set to $+\infty$.

In the following lemma, we show that (3.6) can be optimally solved for all nodes in the B&B tree. It also helps derive a procedure for $\Phi_{\rm ub}(\cdot)$.

Lemma 3. *Regarding* (3.6)*, the following hold:*

(a) Consider the BF case where perfect CSI is given. Then, (3.6) *can be optimally solved by using SOCP.*

(b) Consider the RBF case where imperfect CSI is given. Assume that

$$\frac{\|\mathbf{\Pi}_m \widetilde{\boldsymbol{h}}_m\|_2^2}{\varepsilon_m^2} > 1 + M + (M - \frac{1}{M})\gamma_m, \forall m, \tag{3.7}$$

where $\mathbf{\Pi}_m := \boldsymbol{I} - \widetilde{\boldsymbol{H}}_{-m}(\widetilde{\boldsymbol{H}}_{-m}^H \widetilde{\boldsymbol{H}}_{-m})^{-1}\widetilde{\boldsymbol{H}}_{-m}^H$*, holds for* $\widetilde{\boldsymbol{H}} \in \{\boldsymbol{H}(\mathcal{S},:) | \forall \mathcal{S} \in [N], |\mathcal{S}| \geq L\}$*. Then, Problem* (3.6) *can be optimally solved using SDR.*

(c) Under the same conditions of (a) and (b), solving the following gives a valid upper bound of (3.5) *under the BF and RBF cases, respectively:*

$$\Phi_{\rm ub}(\mathcal{N}_s^{(\ell)}) = \underset{\boldsymbol{W}}{\text{minimize}} \ \|\boldsymbol{W}\|_F^2 \tag{3.8a}$$

$$\text{subject to} \ \ \mathcal{C}(\boldsymbol{w}_m, \boldsymbol{h}_m, \varepsilon_m, \sigma_m) \geq \gamma_m, \forall m, \tag{3.8b}$$

$$\boldsymbol{W}(n,:) = \boldsymbol{0}, \quad \forall n \in \widetilde{\mathcal{B}}_s^{(\ell)},$$

where $\widetilde{\mathcal{B}}_s^{(\ell)} = \mathcal{C}_s^{(\ell)} \cup \mathcal{B}_s^{(\ell)}$ *represents the set of* $N - L$ *antennas to be excluded, and* $\mathcal{C}_s^{(\ell)} \subseteq [N] \backslash (\mathcal{A}_s^{(\ell)} \cup \mathcal{B}_s^{(\ell)})$ *is the index set of undecided antennas that have been assigned the minimum power in the solution of* (3.6)*. If Problem* (3.8) *is not feasible,* $\Phi_{\rm ub}(\mathcal{N}_s^{(\ell)})$ *is notationally set to* $+\infty$*.*

The proof of Lemma 3 is relegated to Appendix A.

3.2.3 Node Selection and Branching

After (3.6) and (3.8) are computed in iteration t, $l_G^{(t+1)}$ and $u_G^{(t+1)}$ are updated. If the stopping criterion $u_G^{(t)} - l_G^{(t)} \leq \varepsilon$ is not met, one needs to pick a node in $\mathcal{P}^{(t)}$ to further partition. To this end, we employ the "lowest lower bound first" principle that is often

used in the literature [15]. To be specific, we pick a non-leaf node $\mathcal{N}_{s^\star}^{(\ell^\star)}$ such that

$$(\ell^\star, s^\star) \in \arg \min_{(s,\ell)\in\mathcal{P}^{(t)}\backslash\mathcal{S}_{\text{leaf}}} \Phi_{\text{lb}}(\mathcal{N}_s^{(\ell)}), \tag{3.9}$$

where $\mathcal{S}_{\text{leaf}} := \{(\ell, s) : |\mathcal{A}_s^{(\ell)}| = L, |\mathcal{B}_s^{(\ell)}| = N - L\}$ is the set of leaf nodes. To partition the region $\mathcal{N}_{s^\star}^{(\ell^\star)}$, we need to pick an *undecided* antenna and decide whether to include or exclude it in our solution. We select the antenna that has been assigned the largest power among the undecided antennas in iteration t, i.e.,

$$n^\star = \arg \max_{i\in[N]\backslash(\mathcal{A}_{s^\star}^{(\ell^\star)}\cup\mathcal{B}_{s^\star}^{(\ell^\star)})} \|\boldsymbol{W}_{s^\star}^{(\ell^\star)}(i,:)\|_2^2, \tag{3.10}$$

where $\boldsymbol{W}_{s^\star}^{(\ell^\star)} := \arg\min_{\boldsymbol{W}}$ (3.6) at $\mathcal{N}_{s^\star}^{(\ell^\star)}$. Then, $n^\star$ is used to partition $\mathcal{N}_{s^\star}^{(\ell^\star)}$ into two child nodes (i.e., excluding and including antenna $n^\star$ on top of the decided antennas in $\mathcal{N}_{s^\star}^{(\ell^\star)}$). The associated include/exclude sets in the child nodes, $\mathcal{N}_{s_i^\star}^{(\ell^\star+1)}, i \in \{1, 2\}$, are updated as follows:

$$\begin{aligned} \mathcal{B}_{s_1^\star}^{(\ell+1)} &= \mathcal{B}_{s^\star}^{(\ell)} \cup \{n^\star\}, \quad \mathcal{A}_{s_1^\star}^{(\ell+1)} = \mathcal{A}_{s^\star}^{(\ell)}, \\ \mathcal{A}_{s_2^\star}^{(\ell+1)} &= \mathcal{A}_{s^\star}^{(\ell)} \cup \{n^\star\}, \quad \mathcal{B}_{s_2^\star}^{(\ell+1)} = \mathcal{B}_{s^\star}^{(\ell)}. \end{aligned}$$

Note that if any of the child nodes, have L included or $N - L$ excluded antennas, we apply the following update:

$$\begin{aligned} \mathcal{B}_{s_i^\star}^{(\ell^\star+1)} &= [N]\backslash\mathcal{A}_{s_i^\star}^{(\ell^\star+1)} \text{ if } |\mathcal{A}_{s_i^\star}^{(\ell^\star+1)}| = L \\ \mathcal{A}_{s_i^\star}^{(\ell^\star+1)} &= [N]\backslash\mathcal{B}_{s_i^\star}^{(\ell^\star+1)} \text{ if } |\mathcal{B}_{s_i^\star}^{(\ell^\star+1)}| = N - L. \end{aligned} \tag{3.11}$$

This ensures that we do not generate any new nodes that do not satisfy (2.9c). Finally, the two children replace $\mathcal{N}_{s^\star}^{(\ell^\star)}$ in $\mathcal{P}^{(t)}$ to form $\mathcal{P}^{(t+1)}$.

Note during the process, some nodes in the B&B tree can be simply discarded, or, "fathomed"—as in the standard terminologies of B&B [15]. After iteration t, one can potentially find a set of (s', ℓ') such that

$$\Phi_{\text{lb}}(\mathcal{N}_{s'}^{(\ell')}) > u_G^{(t)}.$$

The above means that $\mathcal{N}_{s'}^{(\ell')}$ needs not to be further partitioned in the next iteration. Hence, we can form a set $\mathcal{F}^{(t)}$ in each iteration, which only contains the nodes that need to be further considered, i.e.,

$$\mathcal{F}^{(t)} = \left\{ (s', \ell') \in \mathcal{P}^{(t)} \;\middle|\; \Phi_{\text{lb}}\left(\mathcal{N}_{s'}^{(\ell')}\right) \leq u_G^{(t)} \right\}$$

This is arguably the most important for attaining efficiency against exhaustive search. A summary of the B&B procedure is presented in Algorithm 3.2.3.

Algorithm 2 BB

Input: Problem instance $(\boldsymbol{h}_m, \sigma_m, \gamma_m, \varepsilon_m), \forall m$, trained pruning policy $\boldsymbol{\pi_\theta}$, relative error ϵ; # Add the root node first

$\mathcal{A}_1^{(0)} \leftarrow \{\}, \mathcal{B}_1^{(0)} \leftarrow \{\}$;
Select node using (3.9) for $\mathcal{N}_1^{(0)}$;
$\boldsymbol{W}_{\text{incumbent}} \leftarrow$ solution to (3.8);
$l_G^{(t)} \leftarrow \|\boldsymbol{W}_1^{(0)}\|_F^2$, $u_G^{(0)} \leftarrow \|\boldsymbol{W}_{\text{incumbent}}\|_F^2$;
$\mathcal{F}^{(0)} \leftarrow \{(0,1)\}$;
$t \leftarrow 0$;
while $|\mathcal{F}^{(t)}| > 0$ and $\left|u_G^{(t)} - l_G^{(t)}\right| / l_G^{(t)} > \epsilon$ **do**
  Select a non-leaf node $(\ell^\star, s^\star)$ using (3.9)
  Remove the selected node $\mathcal{F}^{(t)} \leftarrow \mathcal{F}^{(t)} \backslash \mathcal{N}_{s^\star}^{(\ell^\star)}$;
  Select variable $n^\star$ using (3.10);
  Generate child nodes $\mathcal{N}_{s_1^\star}^{(\ell^\star+1)}$ and $\mathcal{N}_{s_2^\star}^{(\ell^\star+1)}$ using (3.4) and append to $\mathcal{F}^{(t)}$;
  $k \leftarrow \arg\min_{i \in \{1,2\}} \Phi_{\text{ub}}\left(\mathcal{N}_{s_i^\star}^{(\ell^\star+1)}\right)$;
  if $\Phi_{\text{ub}}\left(\mathcal{N}_{s_k^\star}^{(\ell^\star+1)}\right) \leq u_G^{(t)}$ **then**
    $u_G^{(t+1)} \leftarrow \Phi_{\text{ub}}\left(\mathcal{N}_{s_k^\star}^{(\ell^\star+1)}\right)$;
    $\boldsymbol{W}_{\text{incumbent}} \leftarrow$ solution to (3.8) for $\mathcal{N}_{s_k^\star}^{(\ell^\star+1)}$;
  end if
  $l_G^{(t+1)} \leftarrow \min_{(\ell,s) \in \mathcal{F}^{(t)}} \Phi_{\text{lb}}\left(\mathcal{N}_s^{(\ell)}\right)$;
  $\mathcal{F}^{(t+1)} \leftarrow \left\{ (s', \ell') \in \mathcal{F}^{(t)} \mid \Phi_{\text{lb}}\left(\mathcal{N}_{s'}^{(\ell')}\right) \leq u_G^{(t+1)} \right\}$;
  $t \leftarrow t + 1$;
end while
Return $\boldsymbol{W}_{\text{incumbent}}$;

3.2.4 An Alternative B&B Method

It is interesting to note that there is often more than one way to come up with a B&B procedure for a given problem. For example, a commonly used approach for deriving B&B of *mixed integer and linear programs* (MILPs), and more generally, subset selection problems (see, e.g., [7, 37]) can also be used for our problem (2.9). The method is by introducing auxiliary Boolean variables. Specifically, problem (2.9) can be expressed as follows:

$$\begin{aligned} \underset{\boldsymbol{W},\boldsymbol{z}}{\text{minimize}} \ & \|\boldsymbol{W}\|_F^2 && \text{(3.12a)} \\ \text{subject to } & \mathcal{C}(\boldsymbol{w}_m, \boldsymbol{h}_m, \varepsilon_m, \sigma_m) \geq \gamma_m, \\ & \boldsymbol{z} \in \{0,1\}^N, && \text{(3.12b)} \\ & \boldsymbol{z}^\top \mathbf{1} \leq L, \\ & \|\boldsymbol{W}(n,:)\|_2 \leq C z(n), \ \forall n \in [N]. \end{aligned}$$

where $C < \infty$ is a large positive constant and $z(n) = 0$ means that the nth antenna is excluded whereas $z(n) = 1$ indicates the opposite. The constraint in (3.12b) can be relaxed to be $\boldsymbol{z} \in [0,1]^N$ for finding the lower bound (see Appendix B.2.2 for details). In this procedure, the branching operations are imposed on the new variable $\boldsymbol{z}$ [7, 37]. The reason that we do not choose formulation (3.12) to design B&B for our joint (R)BF&AS problem is that this approach could be computationally (much) less efficient compared to the proposed approach (see a proof in Theorem 1 in the next section). The computational efficiency of our method comes from the fact that the computation of upper and lower bounds in (3.6) and (3.8) can be reused for many nodes; see the proof of Theorem 1. However, it is not obvious if such kind of computation reduction is still possible for the formulation in (3.12).

3.3 Optimality

We show that the proposed algorithm will produce optimal solutions for the problem of interest:

Theorem 1. *Regarding the proposed B&B procedure in Algorithm 3.2.3, the following*

statements hold:

(a) When BF is considered, the proposed B&B solves (2.9) *optimally.*

(b) When RBF is considered, if the conditions in Lemma 3(b) are satisfied, the proposed B&B solves (2.9) *optimally.*

(c) The total number of SOCPs/SDRs solved by the proposed B&B is upper bounded by

$$Q_{\text{Compute}} = \binom{N}{L} + \sum_{i=2}^{N-L+1} \binom{N-i}{L-1}.$$

The number of SOCPs/SDRs needed by the B&B associated with the alternative formulation in Sec. 3.2.4 is upper bounded by $Q'_{\text{Compute}} = 2\binom{N}{L} - 1$.

The proof of Theorem 1 is in Appendix B. At the first glance, it feels a bit surprising that the B&B algorithms could use more than $\binom{N}{L}$ SOCP/SDRs to find the optimal solution, since this seems to be worse than exhaustive search. This is because, in the worst case, B&B visits many more intermediate states in the search tree—but exhaustive search only visits the leaves. Nonetheless, in practice, B&B is often much more efficient than exhaustive search since B&B does not really exhaust all the nodes. Theorem 1 (c) spells out the advantage of our B&B design relative to the more classic B&B idea as in (3.12) from the MILP literature. Note that the reduction of complexity shown in (c) could be substantial. For example, when $(N, L) = (12, 8)$, Q_{Compute} =660, whereas Q'_{Compute} =989. Hence, there is a potential saving of 339 SOCPs/SDRs (reduction by 34%) in the worst case.

Remark 1. *Under approximate CSI, the conditions in Lemma 3(b) is the premise for our theorem to hold [cf. Theorem 1(b)]. When the condition is violated, it is possible that the SDR in* (3.5) *might return solutions whose rank is higher than one in theory—which would hinder the optimality of the B&B procedure. Nonetheless, such higher-rank solutions were never seen in our simulations—which is consistent with observations from the literature [10, 46, 68, 73]. Our conjecture is that the sufficient condition in Lemma 3(b) is far from necessary. In rare cases where rank-one solutions do not exist for* (3.5)*, standard procedures like randomization [45] may be resorted to for finding rank-one approximations.*

3.4 Summary

In this chapter, we proposed an effective B&B procedure to solve (R)BF&AS problem optimally. B&B design is problem-specific and often an art. An effective B&B procedure needs to exploit the problem structure to obtain optimal solution faster than the exhaustive search. Due to the hidden convexity of the beamforming problem, the proposed B&B procedure could "bypass" the continuous optimization related to beamforming and only branch on the discrete antenna selection constraint. The proposed approach provides optimal solution using smaller computation budget than commonly used frameworks for subset selection using B&B [7, 37]. Although more effective than the exhaustive search, as will be seen in Chapter 5, it still suffers from exponential time complexity and cannot be used for solving large-sized problems. In the next chapter, we discuss a machine learning based solution based on imitation learning and graph neural networks that can significantly speedup the branch and bound procedure while retaining the optimality of the solution under some conditions.

Chapter 4: Accelerated Joint (R)BF&AS via ML

The challenge of any B&B algorithm lies in the large number of nodes in the tree. This means that in the worst case, many SOCPs and SDRs need to be solved. An idea from the ML community is to "train" a classifier to recognize the *relevant nodes*, i.e., nodes that lead to leaves containing the optimal solution [27]. If a node is deemed to be "irrelevant", the B&B algorithm would simply skip branching on this node, and thus could save a substantial amount of time. In this section, we will show that a similar idea can be used for accelerating our B&B based joint (R)BF&AS algorithm—with carefully designed neural models to meet the requirements arising in wireless communications. More importantly, we will present comprehensive performance characterizations, including sample complexity and global optimality retention, which are currently lacking in the existing literature.

4.1 Preliminaries: Node Classification and Imitation Learning

4.1.1 Node Classification

Let us denote

$$\boldsymbol{\pi}_{\boldsymbol{\theta}} : \mathbb{R}^P \to [0, 1]$$

as the node classifier parameterized by $\boldsymbol{\theta}$, which returns the probability of a node being relevant. Let

$$\boldsymbol{\phi}(\mathcal{N}_s^{(\ell)}) \in \mathbb{R}^P$$

be the mapping from a node to its feature representation. When $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}(\mathcal{N}_s^{(\ell)})) < 0.5$, then the node is deemed irrelevant. Otherwise, the node is branched.

To train such a classifier, denote $\{(\mathcal{N}_s, y_s)\}_{s=1}^T$ as the (node, label) training data, where we have removed the level indices of the nodes for notation simplicity. To create the training pairs, one could run random problem instances of (2.9) using the B&B

procedure. Note that the label y_s is annotated according to the following rule:

$$y_s = \begin{cases} 1, & \mathcal{A}_s \subseteq \mathcal{A}^\star \text{ and } \mathcal{B}_s \subseteq [N] \backslash \mathcal{A}^\star, \\ 0, & \text{otherwise}, \end{cases} \tag{4.1}$$

where $\mathcal{A}_s$ and $\mathcal{B}_s$ are the index sets of included and excluded antennas at node s, respectively, and $\mathcal{A}^\star$ is the index set of the active antennas of the optimal solution found by B&B of the associated problem instance.

4.1.2 Imitation Learning

The simplest supervised learning paradigm would learn $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ using the following risk minimization criterion:

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \ \frac{1}{T} \sum_{s=1}^{T} \mathcal{L}\left(\boldsymbol{\pi}_{\boldsymbol{\theta}}\left(\boldsymbol{\phi}_s\right), y_s\right) + r(\boldsymbol{\theta}), \tag{4.2}$$

where $\boldsymbol{\phi}_s := \boldsymbol{\phi}(\mathcal{N}_s)$, $\mathcal{L}(x, y)$ is a certain loss function, e.g., the logistic loss, and $r(\boldsymbol{\theta})$ is a regularization term, e.g., $r(\boldsymbol{\theta}) = \lambda \|\boldsymbol{\theta}\|_2^2$. Unfortunately, such a supervised learning approach often does not work well, since it ignores the fact that the node generating process is *sequential* and *interactive* with the node classifier in the test stage. In ML-based MILP, the remedy is to adopt the *imitation learning* (IL) [61] approach, where $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ is integrated in the training data generating process [27]. To be more specific, the training data generation process is done in a batch-by-batch manner with *online optimization*. The IL training criterion is as follows (see Chapter 4.3 for data generation and training process):

$$\boldsymbol{\theta}^{(i+1)} = \arg\min_{\boldsymbol{\theta}} \frac{1}{i} \sum_{t=1}^{i} \frac{1}{|\mathcal{D}_t|} \sum_{(\boldsymbol{\phi}_s, y_s) \in \mathcal{D}_t} \mathcal{L}\left(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s), y_s\right) + r(\boldsymbol{\theta}),$$

where $\mathcal{D}_t$ is the tth batch of training pairs. The learned model parameter $\widehat{\boldsymbol{\theta}}$ is selected from $\boldsymbol{\theta}^{(i)}$'s via the following:

$$\widehat{\boldsymbol{\theta}} = \arg \min_{\boldsymbol{\theta} \in \{\boldsymbol{\theta}^{(i)}\}_{i=1}^{I}} \mathbb{E}_{(\boldsymbol{\phi}_s, y_s)} \left[\mathcal{L}\left(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s), y_s\right)\right], \tag{4.3}$$

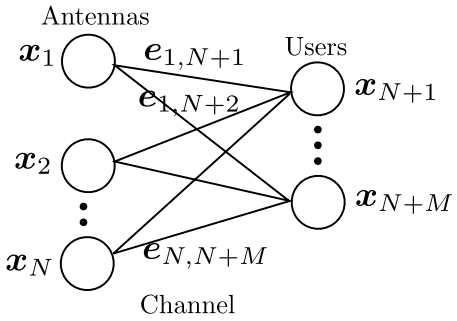

Figure 4.1: Illustration of the input graph representation for a node.

where I is the total number of batches generated during the training process. In practice, one can use a validation set to approximate the above expectation. In the test stage, the proposed B&B algorithm is run with the assistance of $\boldsymbol{\pi}_{\widehat{\boldsymbol{\theta}}}$.

The key of using IL to accelerate the proposed B&B for joint (R)BF&AS is twofold, namely, a practical node classifier tailored for wireless communications and a convergent online training algorithm. We will detail our designs to address the two requirements in the next sections.

4.2 GNN-based Node Classifier for Joint (R)BF&AS

To design the node classifier, a critical consideration in wireless communications is that the number of users to serve could drastically change from time to time. This requires us to design an ML model that is agnostic to such changes, as re-training a model when change happens is not affordable. Towards this end, we design a GNN-based node classifier [64]. Note that GNNs learn aggregation operators over a graph, and thus is naturally robust to the change of entities on the graph. We will leverage this property to design our node classifier.

4.2.1 Neural Architecture Design

To describe the GNN-based node classifier, we first define a graph to represent $\mathcal{N}_s^{(\ell)}$. Figure 4.1 illustrates the idea, where the antennas and users represent the vertices, and the channel represent the edge between the vertices. It is important to design the features

of the vertices and the edges, so that they represent the essential information of the node $\mathcal{N}_s^{(\ell)}$. To be specific, we let

$$\begin{aligned} &\boldsymbol{x}_n \in \mathbb{R}^{V_a}, n \in [N], \quad \boldsymbol{x}_{N+m} \in \mathbb{R}^{V_u}, m \in [M], \text{ and} \\ &\boldsymbol{e}_{n,N+m} \in \mathbb{R}^{V_e}, n \in [N], m \in [M] \end{aligned} \tag{4.4}$$

represent the feature vectors of antenna n (a vertex), user m (a vertex), and the channel between the antenna n and the user m (an edge), respectively. Layer d of the GNN "aggregates" the embedding of graph neighbors to update the uth vertex for all $u \in [M+N]$. The definition of such aggregation can be flexible. For example, in the *message passing neural network* [25], the aggregation is done by the following:

$$\boldsymbol{q}_u^{(d)} = \boldsymbol{\xi}(\boldsymbol{Z}_1 \boldsymbol{q}_u^{(d-1)} + \sum_{v \in \mathcal{E}_u} \boldsymbol{\xi}(\boldsymbol{Z}_2 \boldsymbol{q}_v^{(d-1)} + \boldsymbol{Z}_3 \boldsymbol{e}_{u,v})), \tag{4.5}$$

where $\boldsymbol{q}_u^{(0)} = \boldsymbol{x}_u$; $\boldsymbol{Z}_i$ for $i = 1, 2, 3$ are the *aggregation operators* of the GNN; $\boldsymbol{\xi}(\cdot)$ represents the activation functions of layer d; and $\mathcal{E}_u$ is the index set of all the one-hop neighbors of vertex u on the graph. The output of the GNN is

$$\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s) = \frac{1}{U} \sum_{u \in [U]} \boldsymbol{\zeta}\left(\boldsymbol{\beta}^\top \boldsymbol{q}_u^{(D)}\right), \ \boldsymbol{\phi}_s = \boldsymbol{\phi}(\mathcal{N}_s) \in \mathbb{R}^P$$

where $U = M+N$ is the total number of vertices; $\boldsymbol{\phi}(\mathcal{N}_s) = [\boldsymbol{x}_1^\top, \ldots, \boldsymbol{x}_{N+M}^\top, \boldsymbol{e}_{1,N+1}^\top, \ldots, \boldsymbol{e}_{N,N+M}^\top]^\top$; and $\boldsymbol{\zeta}(\cdot)$ is a sigmoid function. Here, the parameter to be optimized is given by $\boldsymbol{\theta} := [\mathrm{vec}(\boldsymbol{Z}_1)^\top, \mathrm{vec}(\boldsymbol{Z}_2)^\top, \mathrm{vec}(\boldsymbol{Z}_3)^\top, \boldsymbol{\beta}^\top]^\top$.

4.2.2 Feature Design

Table 4.1 shows the detailed feature descriptions. We design two types of features to represent the B&B nodes. To be specific, Type I features represent the features whose dimensions are not affected by the problem size parameters N, M, L. For example, Φ_{lb} is a Type I feature as it is always a scalar under any (N, M, L). Type II features are those whose dimensions change when (N, M, L) changes. For instance, the channel matrix $\boldsymbol{H} \in \mathbb{C}^{M \times N}$ is a Type II feature.

Table 4.1: Feature Design for the GNN based node classifier.

Type I Features	**Type II Features**
$l_G^{(t)}$	$\mathcal{A}_s^{(\ell)}$
$u_G^{(t)}$	$\mathcal{B}_s^{(\ell)}$
$\Phi_{\text{lb}}(\mathcal{N}_s^{(\ell)})$	$[\|\|\boldsymbol{W}_{\ell,s}(1,:)\|\|_2^2, \ldots, \|\|\boldsymbol{W}_{\ell,s}(N,:)\|\|_2^2]$
$\Phi_{\text{ub}}(\mathcal{N}_s^{(\ell)})$	$\boldsymbol{H}$
ℓ	$\boldsymbol{W}_{\text{incumbent}}$ (see Algorithm 4.3)
$\mathbb{1}(\Phi_{\text{ub}}(\mathcal{N}_s^{(\ell)}) - u_G^{(t)} < \epsilon)$.	$\boldsymbol{W}_{\ell,s}$
	$\lvert\boldsymbol{W}_{\ell,s}(:,m)^H \boldsymbol{h}_m\rvert^2$.
	Aggregate Interference using $\boldsymbol{W}_{\ell,s}$.

Table 4.2: Classification error (%) attained by SVM, FCN and GNN based classifier for classifying relevance of the nodes. $\gamma_m = \sigma_m = 1, \varepsilon = 0.1$.

	Perfect CSI		Approximate CSI	
Problem sizes (N, M, L)	(4,3,2)	(8,6,4)	(4,3,2)	(8,5,4)
SVM	8.49	16.67	7.17	11.67
FCN	**6.93**	13.95	26.95	10.18
GNN	7.26	**12.23**	**6.62**	**8.49**

Note that the special structure of GNN allows us to employ both Type I and Type II features. The reason is that the change of M, N and L only changes the number of vertices/edges of the graph in Figure 4.1. This does not necessarily change V_a, V_e and V_u that determines the size of $\boldsymbol{Z}_i$ [cf. Eq. (4.4)]—if $\boldsymbol{x}_n$ and and $\boldsymbol{e}_{n,m}$ are designed properly under the GNN framework (see Appendix 4.2.2.1). However, if one uses SVM as in [27] or other types of neural networks (e.g., fully connected network (FCN) and convolutional neural network (CNN)), Type II features are much less flexible to use. We should remark that our feature design is not "optimal" in any sense, but using Type II features arguably provides more comprehensive information about the node and could often enhance the node classification accuracy.

Table 4.2 shows numerical evidence to support our postulate. There, different classifiers are trained by IL using problem instances as described in Chapter 5. The FCN has two hidden layers with 32 hidden units in each layer, a sigmoid activation function on the output layer, and ReLU activations on the remaining layers. The architecture of the GNN is described in Chapter 5.2.3. The SVM and FCN could only use the Type I features. The GNN with both types of features clearly offers a lower node classification error.

4.2.2.1 Construction of input features

We assign the features tabulated in Table 4.1 among the elements of the following sets: $\{\boldsymbol{x}_i \mid i \in [N]\}$, $\{\boldsymbol{x}_{N+i} \mid i \in [M]\}$, and $\{\boldsymbol{e}_{i,N+j} \mid i \in [N], j \in [M]\}$. Specifically, the Type II features that can be represented with a vector of dimension N (i.e., $\mathcal{A}_s^{(\ell)}$, and $\mathcal{B}_s^{(\ell)}$, $[\|\boldsymbol{W}_{\ell,s}(1,:)\|_2^2, \ldots, \|\boldsymbol{W}_{\ell,s}(N,:)\|_2^2]$) are assigned to the elements of $\{\boldsymbol{x}_i \mid i \in [N]\}$ as follows:

$$x_i(1) = \begin{cases} 1, & \text{if } i \in \mathcal{A}_s^{(\ell)} \\ 0, & \text{otherwise,} \end{cases} \qquad x_i(2) = \begin{cases} 1, & \text{if } i \in \mathcal{B}_s^{(\ell)} \\ 0, & \text{otherwise, and} \end{cases}$$

$$x_i(3) = \|\boldsymbol{W}_{\ell,s}(i,:)\|_2^2.$$

Similarly, the Type II features that can be represented by a vector of dimension M (i.e., $\boldsymbol{W}_{\ell,s}(:,m)^H \boldsymbol{h}_m$ and the aggregated interference under $\boldsymbol{W}_{\ell,s}$) are assigned to be the

elements of $\{\boldsymbol{x}_{N+i} \mid i \in [M]\}$ as follows:

$$x_{N+i}(1) = \left|\boldsymbol{W}_{\ell,s}(:,i)^H \boldsymbol{h}_i\right|^2,\ x_{N+i}(2) = \sum_{j \neq i} \left|\boldsymbol{W}_{\ell,s}(:,j)^H \boldsymbol{h}_i\right|^2.$$

The remaining Type II features can be represented by a vector of dimension NM, and are assigned to the elements of $\{\boldsymbol{e}_{i,N+j} \mid i \in [N], j \in [M]\}$ as follows:

$$\begin{aligned}
&(e_{i,N+j}(1), e_{i,N+j}(2), e_{i,N+j}(3)) = (\mathrm{Re}(\boldsymbol{H}(i,j)),\\
&\qquad \mathrm{Im}(\boldsymbol{H}(i,j)), |\boldsymbol{H}(i,j)|)\\
&(e_{i,N+j}(4), e_{i,N+j}(5), e_{i,N+j}(6)) = (\mathrm{Re}(\boldsymbol{W}_{\text{incumbent}}(i,j)),\\
&\qquad \mathrm{Im}(\boldsymbol{W}_{\text{incumbent}}(i,j)), |\boldsymbol{W}_{\text{incumbent}}(i,j)|)\\
&(e_{i,N+j}(7), e_{i,N+j}(8), e_{i,N+j}(9)) = (\mathrm{Re}(\boldsymbol{W}_{\ell,s}(i,j)),\\
&\qquad \mathrm{Im}(\boldsymbol{W}_{\ell,s}(i,j)), |\boldsymbol{W}_{\ell,s}(i,j)|),
\end{aligned}$$

where $\mathrm{Re}(\cdot)$ and $\mathrm{Im}(\cdot)$ returns the real and imaginary part of the complex number.

Finally, the Type I features are assigned to the set $\{\boldsymbol{x}_{N+i} \mid i \in [M]\}$ as follows:

$$\begin{aligned}
&(x_{N+i}(3), x_{N+i}(4), \ldots, x_{N+i}(8))\\
&= (l_G^{(t)}, u_G^{(t)}, \Phi_{\mathrm{lb}}(\mathcal{N}_s^{(\ell)}), \Phi_{\mathrm{ub}}(\mathcal{N}_s^{(\ell)}), \ell, \mathbb{1}(\Phi_{\mathrm{ub}}(\mathcal{N}_s^{(\ell)}) - u_G^{(t)} < \epsilon)).
\end{aligned}$$

Remark 2. *In addition to being able to work with both types of features, another important benefit of using GNN is as folows: Since $\boldsymbol{\theta}$ of the GNN model does not depend on (N, M, L), the learned model can naturally work when the numbers of users and antennas change, as long as V_a, V_u, and V_e remain the same. That is, the model trained on problem instances with (N, M, L) can be seamlessly tested on cases with $(N', M', L') \neq (N, M, L)$. This property of GNN will be vital for applying the proposed method in real-world scenarios where the problem size changes constantly (as the number of users to be served by a BS changes all the time). It also helpd scale up the proposed method for coping with large (N, M, L) using a $\boldsymbol{\theta}$ trained from small problem sizes, which could save a substantial amount of computational resources.*

We should emphasize that GNN is "insenstive" to the change of problem size across training and testing. However, drastic change of other aspects (e.g., channel model and

noise level) across the two stages does affect the performance more substantially. In other words, beyond the problem size, our GNN-based method still expects that the training and testing data to share similar characteristics, as other machine learning models do.

4.3 Data Generation and Online Training

We use an IL framework to train the GNN, which is summarized in Algorithm 4.3. The framework is based on the online learning method in [61]. The work in [61] was proposed for convex learning criteria. Necessary modifications are made in Algorithm 4.3 to accommodate our nonconvex learning problem.

Algorithm 4.3 consists of two steps in each iteration: data collection and classifier improvement. In the ith iteration, the accumulated dataset $\mathcal{D}_i$ is obtained by solving B&B on R problem instances using the current classifier learned from the previous data batches, $\boldsymbol{\pi}_{\boldsymbol{\theta}^{(i)}}$. Then, the classifier is retrained using $\cup_{l=1}^{i}\mathcal{D}_i$ and

$$\widehat{\boldsymbol{\theta}}^{(i+1)} = \arg\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \; \boldsymbol{g}_i(\boldsymbol{\theta}) + r(\boldsymbol{\theta})$$

where $\boldsymbol{\Theta}$ specifies the constraints of the GNN parameters [cf. Eq (4.7)]; the loss function $g_i(\cdot)$ is defined as follows:

$$\boldsymbol{g}_i(\boldsymbol{\theta}) := \frac{1}{i}\sum_{t=1}^{i}\frac{1}{|\mathcal{D}_t|}\sum_{(\boldsymbol{\phi}_s,y_s)\in\mathcal{D}_t}\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s),y_s); \tag{4.6}$$

additionally, we select $\boldsymbol{r}(\boldsymbol{\theta}) = -\boldsymbol{\psi}^\top\boldsymbol{\theta}$ in which $\boldsymbol{\psi}$ is sampled from exponential distribution in each iteration. This specific choice of $\boldsymbol{r}(\boldsymbol{\theta})$ plays an important role in our nonconvex learning problem (where the nonconvexity arises due to the use of GNN). To be more specific, such a random perturbation-based $r(\boldsymbol{\theta})$ is advocated by recent developments from nonconvex online learning [1]. It was shown in [1] that using $\boldsymbol{r}(\boldsymbol{\theta}) = -\boldsymbol{\psi}^\top\boldsymbol{\theta}$ ensures no-regret type convergence of nonconvex online learning. This property is a critical stepping stone towards establishing learning guarantees of our GNN-based framework. This will become clearer in the proofs of Theorem 2.

The training data generation subroutine is given in Algorithm 4.3. To generate $\mathcal{D}_i$, the algorithm first runs B&B on a given problem instance to find the optimal solution.

Algorithm 3 ONLINE GNN LEARNING

1: Input: I, R(number of training instances per batch), η;
2: $\mathcal{D}_1 = \{\}$;
3: **for** $i = 1$ to I **do**
4: Sample $\boldsymbol{\psi} \sim (\text{Exp}(\eta))^B$;
$\text{Exp}(\eta)$ is the exponential distribution with pdf $p(x) = \eta \exp(-\eta x)$; $\boldsymbol{\theta}^{(i)} \in \mathbb{R}^B$;
5: **for** $r = 1$ to R **do**
6: Generate problem instance Q;
7: **if** i=1 **then**
8: $\mathcal{D}^{(\mathsf{Q})} \leftarrow$ run $\texttt{BB}(\mathsf{Q})$ and label the nodes using optimal solution;
9: **else**
10: $\mathcal{D}^{(\mathsf{Q})} \leftarrow \texttt{Algorithm\_4.3}(\mathsf{Q}, \boldsymbol{\pi}_{\boldsymbol{\theta}^{(i)}})$;
11: **end if**
12: $\mathcal{D}_i \leftarrow \mathcal{D}_i \cup \mathcal{D}^{(\mathsf{Q})}$;
13: **end for**
14: $\boldsymbol{\theta}^{(i+1)} = \arg\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \ \frac{1}{i} \sum_{t=1}^{i} \frac{1}{|\mathcal{D}_t|} \sum_{(\boldsymbol{\phi}_s, y_s) \in \mathcal{D}_t} \mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s), y_s) - \boldsymbol{\psi}^\top \boldsymbol{\theta}$
15: **end for**
16: Return $\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta} \in \boldsymbol{\theta}_{1:I}} \frac{1}{|\mathcal{D}_i^{\text{valid}}|} \sum_{(\boldsymbol{\phi}_s, y_s) \in \mathcal{D}_i^{\text{valid}}} [\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s), y_s)]$;
where $\mathcal{D}_i^{\text{valid}}$ validation batch i generated by B&B with $\boldsymbol{\pi}_{\boldsymbol{\theta}^{(i)}}$

Next, B&B is run again but with $\boldsymbol{\pi}_{\boldsymbol{\theta}^{(i)}}$ to generate nodes. The training pairs $(\boldsymbol{\phi}_s, y_s)$ are annotated by utilizing the optimal solution obtained in the first run.

The overall GNN-accelerated B&B procedure is summarized in Algoirthm 4.3. The algorithm is termed as *MachINe learning-based joInt beaMforming and Antennas seLection* (`MINIMAL`) The node classifier is used in Line 11.

4.4 Performance Characterizations

Our goal is to characterize the performance of `MINIMAL`, e.g., under what conditions (e.g., the amount of training samples and the complexity of the GNN) `MINIMAL` can accelerate the proposed B&B without losing its optimality. To our best knowledge, such performance characterization have not been provided for ML-based B&B acceleration, even when the learning problem is convex.

To proceed, we will use the following assumptions:

Assumption 1. *Assume that the following statements about the data features and the*

Algorithm 4 TRAINING DATA GENERATION

Input: $\mathtt{Q}$, $\boldsymbol{\pi}_{\boldsymbol{\theta}}$;
  # optimal solution and optimal selected antenna subset to problem $\mathtt{Q}$
$(\boldsymbol{W}^\star, \mathcal{A}^\star) = \mathtt{BB}(\mathtt{Q})$; (see Algorithm 3.2.3 for $\mathtt{BB}$)
  # Initialization
Execute Line 2 to Line 7 in Algorithm 4.3;
$\mathcal{D} \leftarrow \{\}$;
while B&B termination criteria is not met **do**
  Execute Line 9 to Line 22 from Algorithm 4.3;
  if $\mathcal{N}_{s^\star}^{(\ell^\star)}$ is relevant **then**
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{\boldsymbol{\phi}_{s^\star}^{(\ell^\star)}, 0\}$;
  else
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{\boldsymbol{\phi}_{s^\star}^{(\ell^\star)}, 1\}$;
  end if
end while
Return $\mathcal{D}$;

GNN in Sec. 4.2 hold:

(a) The input features are bounded, i.e., $\|\boldsymbol{x}_u\|_2, \|\boldsymbol{e}_{u,v}\|_2 \leq B_{\boldsymbol{x}}, \forall u, v$.

(b) The activation functions $\boldsymbol{\xi}(\cdot)$ *and* $\boldsymbol{\zeta}(\cdot)$ *are* $C_{\boldsymbol{\xi}}$*-Lipschitz and* $C_{\boldsymbol{\zeta}}$*-Lipschitz continuous, respectively. In addition,* $\boldsymbol{\xi}(\mathbf{0}) = \mathbf{0}$.

(c) Let $\mathcal{L} : \mathbb{R} \times \mathbb{R} \to [-B_{\mathcal{L}}, B_{\mathcal{L}}]$ *be* $C_{\mathcal{L}}$*-Lipschitz in its first argument, i.e.,* $|\mathcal{L}(x, y) - \mathcal{L}(x', y)| \leq C_{\mathcal{L}}|x - x'|$.

(d) The parameters of the GNN are bounded; i.e., $\|\boldsymbol{Z}_i\|_2 \leq B_{\boldsymbol{Z}}, \forall i \in \{1, 2, 3\}$ *and* $\|\boldsymbol{\beta}\|_2 \leq B_{\boldsymbol{\beta}}$.

Let us define the set of parameters $\boldsymbol{\Theta}$ as follows:

$$\boldsymbol{\Theta} := \big\{\boldsymbol{\theta} = [\mathrm{vec}(\boldsymbol{Z}_1)^\top, \mathrm{vec}(\boldsymbol{Z}_2)^\top, \mathrm{vec}(\boldsymbol{Z}_3)^\top, \boldsymbol{\beta}^\top]^\top \;\big|\; \|\boldsymbol{Z}_i\|_2 \leq B_{\boldsymbol{Z}}, \boldsymbol{\beta} \leq B_{\boldsymbol{\beta}}, i \in \{1, 2, 3\}\big\}. \tag{4.7}$$

Using the above, we first characterize the generalization error of the GNN with the following Lemma:

Algorithm 5 MAIN ALGORITHM: MINIMAL

Input: Problem instance $(\boldsymbol{h}_m, \sigma_m, \gamma_m, \varepsilon_m), \forall m$, trained pruning policy $\boldsymbol{\pi_\theta}$, relative error ϵ;
  # Add the root node first
$\mathcal{A}_1^{(0)} \leftarrow \{\}, \mathcal{B}_1^{(0)} \leftarrow \{\}$;
Select node using (3.9) for $\mathcal{N}_1^{(0)}$;
$\boldsymbol{W}_{\text{incumbent}} \leftarrow$ solution to (3.8);
$l_G^{(0)} \leftarrow \|\boldsymbol{W}_1^{(0)}\|_F^2$, $u_G^{(0)} \leftarrow \|\boldsymbol{W}_{\text{incumbent}}\|_F^2$;
$\mathcal{F}^{(0)} \leftarrow \{(0,1)\}$;
$t \leftarrow 0$;
while $|\mathcal{F}^{(t)}| > 0$ and $\left|u_G^{(t)} - l_G^{(t)}\right| / l_G^{(t)} > \epsilon$ **do**
  Select a non-leaf node $(\ell^\star, s^\star)$ using (3.9);
  Remove the selected node $\mathcal{F}^{(t)} \leftarrow \mathcal{F}^{(t)} \backslash \mathcal{N}_{s^\star}^{(\ell^\star)}$;
  if $\boldsymbol{\pi_\theta}\left(\boldsymbol{\phi}_{s^\star}^{(\ell^\star)}\right) \geq 0.5$ **then**
    Select variable $n^\star$ using (3.10);
    Generate child nodes $\mathcal{N}_{s_1^\star}^{(\ell^\star+1)}$ and $\mathcal{N}_{s_2^\star}^{(\ell^\star+1)}$ using (3.4) and append to $\mathcal{F}^{(t)}$;
    $k \leftarrow \arg\min_{i \in \{1,2\}} \Phi_{\text{ub}}\left(\mathcal{N}_{s_i^\star}^{(\ell^\star+1)}\right)$;
    if $\Phi_{\text{ub}}\left(\mathcal{N}_{s_k^\star}^{(\ell^\star+1)}\right) \leq u_G^{(t)}$ **then**
      $u_G^{(t+1)} \leftarrow \Phi_{\text{ub}}\left(\mathcal{N}_{s_k^\star}^{(\ell^\star+1)}\right)$;
      $\boldsymbol{W}_{\text{incumbent}} \leftarrow$ solution to (3.8) for $\mathcal{N}_{s_k^\star}^{(\ell^\star+1)}$;
    end if
    $l_G^{(t+1)} \leftarrow \min_{(\ell,s) \in \mathcal{F}^{(t)}} \Phi_{\text{lb}}\left(\mathcal{N}_s^{(\ell)}\right)$;
  end if
  $\mathcal{F}^{(t+1)} \leftarrow \left\{(s', \ell') \in \mathcal{F}^{(t)} \mid \Phi_{\text{lb}}\left(\mathcal{N}_{s'}^{(\ell')}\right) \leq u_G^{(t+1)}\right\}$;
  $t \leftarrow t + 1$;
end while
Return $\boldsymbol{W}_{\text{incumbent}}$;

Lemma 4 (Generalization Error of GNN)**.** *Consider a GNN $\boldsymbol{\pi_\theta}$ in Sec. 4.2 and $\mathcal{G} = \{\boldsymbol{\phi}_k, y_k\}_{k=1}^K$ of i.i.d. samples. Then, for $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, the following holds with probability at*

least $1-\delta$*:*

$$\begin{aligned}
&\mathsf{Gap}(\delta, K) \\
&:= \mathbb{E}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}), y)] - {}^1\!/_K \sum_{(\boldsymbol{\phi}_k, y_k)\in\mathcal{G}} \mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_k), y_k) \\
&\leq \frac{8C_{\mathcal{L}}}{K} + \frac{24C_{\mathcal{L}}B_{\mathcal{L}}}{\sqrt{K}}\sqrt{(3E^2+E)\log\Lambda} + 3B_{\mathcal{L}}\sqrt{\frac{\log(2/\delta)}{2K}},
\end{aligned} \tag{4.8}$$

where $\alpha = ((1+UC_{\boldsymbol{\xi}})C_{\boldsymbol{\xi}}B_{\boldsymbol{Z}})$,

$$\begin{aligned}
\Lambda &= 1 + 12\sqrt{EK}B_{\boldsymbol{Z}}\max\{\Sigma_{\boldsymbol{Z}_1}, \Sigma_{\boldsymbol{Z}_2}, \Sigma_{\boldsymbol{Z}_3}, {}^{B_{\boldsymbol{\beta}}}\!/_{B_{\boldsymbol{Z}}}\Sigma_{\boldsymbol{\beta}}\}, \\
\Sigma_{\boldsymbol{Z}_1} &= C_{\boldsymbol{\zeta}}B_{\boldsymbol{\beta}}UC_{\boldsymbol{\xi}}^3B_{\boldsymbol{Z}}B_{\boldsymbol{x}}\frac{\alpha^{(D+1)}-2\alpha+1}{(\alpha-1)^2}, \Sigma_{\boldsymbol{Z}_2} = UC_{\boldsymbol{\xi}}\Sigma_{\boldsymbol{Z}_1}, \\
\Sigma_{\boldsymbol{Z}_3} &= C_{\boldsymbol{\zeta}}B_{\boldsymbol{\beta}}UC_{\boldsymbol{\xi}}^2B_{\boldsymbol{Z}}B_{\boldsymbol{x}}\frac{\alpha^D-1}{\alpha-1}, \\
\Sigma_{\boldsymbol{\beta}} &= C_{\boldsymbol{\zeta}}B_{\boldsymbol{x}}\alpha^D + C_{\boldsymbol{\zeta}}UC_{\boldsymbol{\xi}}^2B_{\boldsymbol{Z}}B_{\boldsymbol{x}}\frac{\alpha^D-1}{\alpha-1},
\end{aligned}$$

where the expectation is taken w.r.t. the distribution of (ϕ_k, y_k)*.*

Note that our GNN generalization error bound is rather different from some existing results, e.g., [22], as edge features (i.e., $\boldsymbol{e}_{u,v}$) were not considered in their work. Lemma 4 can be used to understand the GNN's performance with a single batch. To characterize the node classification accuracy of the GNN learned through the described imitation learning algorithm, we need the following assumptions:

Assumption 2. *Let* $\sup_{\boldsymbol{\theta}_1,\boldsymbol{\theta}_2\in\boldsymbol{\Theta}}\|\boldsymbol{\theta}_1-\boldsymbol{\theta}_2\|_\infty \leq H$*, for some* $H<\infty$*. Let all the loss functions* $\boldsymbol{g}_i(\cdot)$ *[cf. Eq. (4.6)] for* $i=1,\ldots,I$ *are* G*-Lipschitz continuous with respect to the* ℓ_1*-norm, i.e.* $|\boldsymbol{g}_i(\boldsymbol{\theta}_1)-\boldsymbol{g}_i(\boldsymbol{\theta}_2)| \leq G\|\boldsymbol{\theta}_1-\boldsymbol{\theta}_2\|_1, \forall i$*.*

Assumption 3. *The minimal empirical loss over the aggregated dataset is bounded by* ν*.*

$$\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \frac{1}{IJ}\sum_{i=1}^{I}\sum_{(\boldsymbol{\phi}_s, y_s)\in\mathcal{D}_i} \mathbb{E}_{\boldsymbol{\psi}}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s), y_s)] \leq \nu.$$

Assumption 2 is not hard to meet if the data features and the network parameters are bounded. Assumption 3 characterizes the expressiveness of the GNN.

To present our main theory, we compute the expected number of nodes that will be visited (with the associated SOCPs/SDRs solved) by Algorithm 4.3 when run with $\boldsymbol{\pi}_{\widehat{\boldsymbol{\theta}}}$ in the testing stage. Let us denote $\rho_{\widehat{\boldsymbol{\theta}}}$ as the probability with which the classifier accurately classifies a node. Also denote $\mathcal{S}$ as the set of all possible B&B trees that can be realized by Algorithm 4.3 under a given instance. Let $\Pr(s;\widehat{\boldsymbol{\theta}}), s \in \mathcal{S}$ be the probability with which a particular tree s is realized. Let $Q^s_{\widehat{\boldsymbol{\theta}}}$ denote the number of visited nodes in tree s. Let $Q_{\widehat{\boldsymbol{\theta}}} = \mathbb{E}[Q^s_{\widehat{\boldsymbol{\theta}}}]$ where the expectation is taken over the probability mass function $\Pr(s;\widehat{\boldsymbol{\theta}}), s \in \mathcal{S}$. In the following theorem, we characterize the classification accuracy, $\rho_{\widehat{\boldsymbol{\theta}}}$, and present a bound on $\boldsymbol{Q}_{\widehat{\boldsymbol{\theta}}}$.

Theorem 2. *Suppose that Assumptions 2-3 hold, and that the GNN in* `MINIMAL` *is prameterized by* $\widehat{\boldsymbol{\theta}}$ *in* (4.3)*. In addition, assume that every single batch* $\mathcal{D}_i$ *consists of i.i.d. samples, and that Algorithm 4.3 is used for GNN learning. Then, we have*

$$Q_{\widehat{\boldsymbol{\theta}}} \leq \frac{2N\left(2\rho_{\widehat{\boldsymbol{\theta}}} - \rho^N_{\widehat{\boldsymbol{\theta}}}\right)}{2\rho_{\widehat{\boldsymbol{\theta}}} - 1} + 1.$$

Further, when $\widehat{\boldsymbol{\theta}}$ *is selected using* (4.3)*, with a probability at least* $1-\delta$*,*

$$\begin{aligned} &\mathbb{E}_{p_{\widehat{\boldsymbol{\theta}}},\boldsymbol{\psi}}\left[\mathcal{L}\left(\boldsymbol{\pi}_{\widehat{\boldsymbol{\theta}}}(\boldsymbol{\phi}_s), y_s\right)\right] \\ &\leq \nu + \mathcal{O}\left(1/I^{1/3}\right) + \mathsf{Gap}\left(\frac{\delta}{2}, J\right)\sqrt{\frac{2\log(2/\delta)}{I}}. \end{aligned} \tag{4.9}$$

Assume the logistic loss function $\mathcal{L}$ *is employed. Then, the node classification accuracy*

$$\rho_{\widehat{\boldsymbol{\theta}}} \geq \exp\left(-\mathbb{E}_{p_{\widehat{\boldsymbol{\theta}}},\boldsymbol{\psi}}\left[\mathcal{L}\left(\boldsymbol{\pi}_{\widehat{\boldsymbol{\theta}}}(\boldsymbol{\phi}_s), y_s\right)\right]\right).$$

In addition, `MINIMAL` *returns an optimal solution with probability at least* $\rho^N_{\widehat{\boldsymbol{\theta}}}$*.*

The proof of Theorem 2 is relegated to Appendix D. This result bounds the number of nodes visited by the proposed algorithm under a given classification accuracy. It also characterizes the classification accuracy that can be achieved by the proposed training procedure. One can see that when the batch size is large enough, Gap is close to zero. Additionally, when the GNN is expressive (and thus ν is small) and the algorithm is run for large enough iterations I, the accuracy of the classifier, i.e., $\rho_{\widehat{\boldsymbol{\theta}}}$, approaches 1 [cf.

Eq. (4.9)]. Consequently, the total number of nodes visited will be close to $2N + 1$ at most. This shows linear dependence of the computational complexity of the proposed method on N, which is a significant saving compared to $\binom{N}{L}$ for the exhaustive search.

Remark 3. *We should remark that the results in Theorem 2 has a couple of caveats. First, we assumed that the samples in each $\mathcal{D}_i$ are i.i.d. If every node created by $\boldsymbol{\pi}_{\boldsymbol{\theta}^{(i)}}$ in Algorithm 4.3 is used, then the samples in $\mathcal{D}_i$ are likely not i.i.d., as the nodes in the same B&B tree are generated in a sequential manner. Nonetheless, simple remedies can assist creating an i.i.d. batch $\mathcal{D}_i$—e.g., by taking only one random node from a B&B tree. This is inevitably more costly, and seems not to be necessary in practice—as using nodes from Algorithm 4.3 for training works fairly well in our simulations. Second, the expectation based criterion* (4.3) *is only approximated in practice, e.g., via using empirical averaging. Characterizing the empirical version of* (4.3) *can be done via concentration theorems in a straightforward manner. However, this would substantially complicate the expressions yet reveals little to no additional insight. Hence, we leave it out of this work.*

4.5 Summary

In this chaper, we detailed the proposed ML-based acceleration scheme for B&B based on GNN and imitation learning. To avoid the high computation complexity of the proposed B&B procedure, one can train a classifier to recognize and skip the intermediate steps in B&B procedure without affecting the solution. To this end, an imitation learning scheme is proposed to learn a classifier to recognize relevant nodes in the B&B procedure. To meet the special demands of wireless communication systems, our classifier should be agnostic to the change in problem dimension. Graph neural network is proposed as a natural choice for the task. The learnt classifier can then be "plugged" into the B&B procedure to make decisions about whether to branch or skip the selected node. Finally, we presented a comprehensive analysis of the resulting ML-assisted B&B which reveals the role of GNN design, sample complexity, number of iterations of the training algorithm in determining the size of the B&B tree. In the next chapter, we provide experimental evidence to support our theoretical analysis and demonstrate how the proposed method can be used to tackle large-scale problems while obtaining high-quality or near-optimal solutions.

Chapter 5: Numerical Experiments

In this section, we showcase the effectiveness of the proposed B&B algorithm and its machine learning based acceleration using numerical simulations. We use CVXPY [17] which calls MOSEK [3] to solve the SOCPs/SDRs in (3.6) and (3.8). The elements of Rayleigh fading channel vectors $\{\boldsymbol{h}_m\}_{m=1}^M$ are sampled independently from circularly symmetric zero mean Gaussian distribution with unit variance. Implementation of the proposed methods can be found on the authors' website[1].

5.1 Evaluation of B&B for Joint (R)BF&AS

In Figure 5.1, we verify the convergence of the proposed B&B algorithm under both the perfect and the approximate CSI cases. The figure shows the convergence of the global upper and lower bounds (i.e., $u_G^{(t)}$ and $l_G^{(t)}$) computed by the proposed B&B procedure for $(N, M, L) = (8, 4, 4)$. One can see that the global bounds converge to the optimal objective value in both the perfect and approximate CSI case. This verifies our optimality claim in Theorem 1. Note that the B&B algorithm for both cases converges in less than 24 iterations (i.e., visiting ≤ 48 nodes). This is much less than the worst-case complexity of B&B, i.e., visiting 139 nodes. The empirical complexity is also better than the worst-case complexity of exhaustive search, which is 70 node visits in this case.

Table 5.1 gives a closer look at the effectiveness of the proposed B&B framework. Specifically, Table 5.1 shows the performance of the proposed B&B procedure for various problem sizes, compared to the exhaustive search strategy for the perfect CSI case. The result is averaged over 30 Monte Carlo trials. One can see that the B&B algorithm can constantly attain reduced complexity, in terms of the number of nodes visited (i.e., the number of SOCPs solved). In particular, when the number of users is relatively small, the B&B can attain an around 8-fold acceleration (cf. the case where $(N, M, L) = (12, 2, 8)$). Similar results can be seen in Table 5.2, where the imperfect CSI case is considered.

Table 5.3 compares our B&B and the alternative B&B using the formulation (3.12)

[1] `https://github.com/XiaoFuLab/Antenna-Selection-and-Beamforming-with-BandB-and-ML.git`

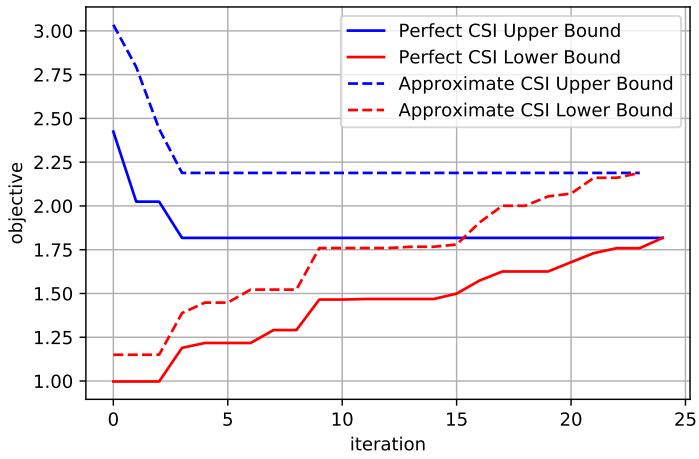

Figure 5.1: Convergence of the global upper and lower bounds, computed by the proposed B&B algorithm, to the optimal solution. Problem instance of size $(N, M, L) = (8, 4, 4)$.

Table 5.1: Performance of the proposed B&B algorithm for various problem sizes in the perfect CSI case compared to the exhaustive search. $\sigma_m^2 = 1.0, \gamma_m = 1.0, \forall m \in [M]$.

Problem size	Proposed B&B		Exhaustive Search	
(N, M, L)	Time	SOCPs	Time	SOCPs
(8, 2, 4)	1.58	34.07	2.95	70
(8, 3, 4)	2.29	40.67	2.58	70
(8, 4, 4)	3.30	47.30	4.53	70
(8, 5, 4)	5.31	63.27	5.46	70
(8, 6, 4)	8.24	82.93	6.10	70
(10, 2, 6)	2.28	50.20	9.11	210
(10, 4, 6)	6.47	88.37	14.75	210
(10, 6, 6)	14.55	141.80	20.00	210
(10, 8, 6)	24.56	186.90	25.59	210
(12, 2, 8)	2.95	65.53	21.39	495
(12, 4, 8)	10.57	137.80	33.45	495
(12, 6, 8)	21.89	211.87	46.53	495
(12, 8, 8)	37.69	279.67	62.46	495
(12, 10, 8)	69.48	398.40	80.94	495

Table 5.2: Performance of the proposed B&B algorithm for various problem sizes in the Approximate CSI case compared to the exhaustive search. $\sigma_m^2 = 1.0, \gamma_m = 1.0, \forall m \in [M]$.

Problem size	Proposed B&B		Exhaustive Search	
(N, M, L)	Time	SDPs	Time	SDPs
(8, 2, 4)	7.09	31.60	12.71	70
(8, 3, 4)	15.09	39.37	21.25	70
(8, 4, 4)	28.39	49.00	32.58	70
(10, 2, 6)	19.49	65.27	51.38	210
(10, 4, 6)	80.47	85.73	133.38	210
(10, 6, 6)	236.26	137.37	262.10	210
(10, 8, 6)	520.81	180.13	452.76	210
(12, 2, 8)	26.83	62.80	157.62	495
(12, 4, 8)	175.45	122.13	471.54	495

Table 5.3: Number of SOCPs solved by two B&B Strategies. $\sigma_m^2 = 1.0, \gamma_m = 1.0, \forall m \in [M]$.

Problem size) (N, M, L)	(4,2,2)	(8,4,6)	(8,6,6)	(10,5,6)
Proposed B&B	6.86	16.73	22.63	117.67
Alternative Using (3.12)	8.06	24.66	33.8	159.6

in the perfect CSI case. One can see that the proposed procedure consistently solves fewer SOCPs. This supports Theorem 1 (c).

5.2 Evaluation of ML-accelerated B&B for Joint (R)BF&AS

In this section, we demonstrate the efficacy of `MINIMAL`.

5.2.1 Baselines

A number of baselines are as follows:

1. **Supervised Learning**: We follow the supervised learning (`SL`) ideas in [28, 71] to train a neural network for antenna selection (cf. Sec. 2.3). Specifically, we use the proposed B&B algorithm to generate training pairs with optimal antenna selection

as the labels, i.e., $\{\boldsymbol{H}_t, \boldsymbol{z}_t\}_{t=1}^T$, where $\boldsymbol{z}_t$ is a binary vector representing optimal antenna selection for the tth training instance. The learned deep model predicts a vector $\boldsymbol{z}$ which may not satisfy $\|\boldsymbol{z}\|_0 \leq L$, and thus we take the L elements that have the largest magnitude as in [28]. For this baseline, we use an $\boldsymbol{f_\theta}$ that is a 3-layer neural network, where the first two layers are convolutional layers with ReLU activations and the last layer is a fully connected layer with sigmoid activation.

2. **Greedy Method**: A plethora of greedy algorithms exist for different variants of joing BF&AS problems; see, e.g., [13, 18, 35, 47, 52]. We design a greedy baseline for (2.9) following the general idea of [13], which is described as follows:

   (a) Let $\mathcal{H} = \{1, \ldots, N\}$ denote the set of all antennas (set to active initially).

   (b) Solve SOCPs with $\widetilde{\mathcal{H}}_{-n} = \mathcal{H} \backslash \{n\}, \forall n \in \mathcal{H}$. Let $\widehat{\mathcal{H}}_{-\widehat{n}}$ correspond to the smallest objective value. Then, set $\mathcal{H} = \mathcal{H} \backslash \widehat{n}$.

   (c) Repeat (ii) if $|\mathcal{H}| > L$; otherwise return $\mathcal{H}$.

   We call this method `Greedy`. Note that `Greedy`'s computational burden is not necessarily light, as a total amount of $\mathcal{O}(N^2)$ SOCPs have to be solved (e.g., ≈ 1000 SOCPs have to be solved for $N = 32$).

3. **Continuous Approximation**: As the third baseline, we use the continuous optimization-based idea in [51] and modify it to solve the unicast cases in this work. Although [51] did not explore their method for the approximate CSI case, we note that the same idea can be used after proper modifications to the subproblems (i.e., using the S-lemma to come up with an SDR formulation of the subproblem). We term this method *iteratively reweighted convex relaxation-based optimization* (`IrCvxOpt`).

   Following the implementation instruction of [51], we run `IrCvxOpt` for at most 30 iterations with its bisection-based λ-tuning method for 30 iterations as well. The algorithm is stopped if the relative change of the reweighting matrix is smaller than 10^{-4} or a solution comprising of $\leq L$ antennas is found. If the algorithm returns $> L$ antennas, we select the L antennas from the returned antennas that is assigned the maximum power in the returned beamforming solution $\widehat{\boldsymbol{W}}$. All of

the evaluation metrics (see Sec 5.2.4) are computed using the final L antennas and $\widehat{\boldsymbol{W}}$ output by the algorithms.

5.2.2 Training Setups

We use a GNN tailored for our beamforming setting (see details in Chapter 5.2.3). We set $(R, I) = (30, 20)$ in Algorithms 4.3-4.3. The loss function $\mathcal{L}$ is selected to be the binary cross-entropy loss, i.e., $\mathcal{L}(x, y) = -y\log(x) - (1-y)\log(1-x)$. In batch i, the parameters of the classifier is initialized with $\boldsymbol{\theta}^{(i)}$, and updated using the Adam algorithm [34] for 10 epochs, where the sample size of Adam is set to be 128. The initial step size of Adam is set to 0.001. As described in Section 4.1.2, we select $\widehat{\boldsymbol{\theta}}$ from $\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(I)}$ using 30 validation problem instances using a sample average version on (4.3).

In order to account for the class imbalance (number of relevant nodes usually much smaller than number of irrelevant nodes in the training set), we apply a larger positive weight on the "positive" training pairs. Further, premature/early pruning of the B&B tree (i.e., when ℓ is small) should be discouraged as it is more risky. Hence, we weight each term $\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}^{(i)}}(\boldsymbol{\phi}_s^{(\ell)}), y_s^{(\ell)})$ using $(q\mathbb{1}[y_s^{(\ell)} = 1] + 1)\frac{1}{\ell}$, where $q \in \mathbb{R}$ offsets the imbalance ratio, and $\mathbb{1}[\cdot]$ denotes the indicator function. We select $q = 11$ via trial and error, and use the same q in all experiments.

5.2.3 GNN Architecture

The GNN is designed to accommodate the unequal input feature dimensions for antennas and users. We enhance the expressiveness GNN by letting different layers to have different aggregation matrices in our experiments. The initial embeddings of a common size E are obtained using a single layer fully connected neural network, i.e.,

$$\begin{aligned}
&\boldsymbol{q}_n^{(0)} = \mathsf{ReLU}(\boldsymbol{Z}_1\boldsymbol{x}_n),\ \boldsymbol{q}_{N+m}^{(0)} = \mathsf{ReLU}(\boldsymbol{Z}_2\boldsymbol{x}_m) \\
&\boldsymbol{e}_{u,v} = \mathsf{ReLU}(\boldsymbol{Z}_3\widetilde{\boldsymbol{e}}_{u,v}).
\end{aligned}$$

where $\boldsymbol{Z}_1 \in \mathbb{R}^{E\times V_a}$, $\boldsymbol{Z}_2 \in \mathbb{R}^{E\times V_u}$, $\boldsymbol{Z}_3 \in \mathbb{R}^{E\times V_e}$, and $\mathsf{ReLU} : \mathbb{R}^E \to \mathbb{R}^E$ deontes elementwise nonlinear function such that $\mathsf{ReLU}(x) = \max\{x, 0\}$.

The first layer of GNN only updates the antenna vertices, i.e., $\boldsymbol{q}_n, n \in [N]$, as follows

$$\begin{aligned}\boldsymbol{q}_n^{(1)} = \boldsymbol{Z}_9\Bigg(\mathsf{ReLU}\Bigg(\boldsymbol{Z}_8\boldsymbol{q}_n^{(0)} + \sum_{m=1}^{M}\boldsymbol{Z}_7\Big(\mathsf{ReLU}\Big(\boldsymbol{Z}_6\boldsymbol{q}_n^{(0)} + \\ \boldsymbol{Z}_5\boldsymbol{q}_{N+m}^{(0)} + \boldsymbol{Z}_4\boldsymbol{e}_{n,N+m}\Big)\Big)\Bigg)\Bigg), \forall n \in [N] \\ \boldsymbol{q}_{N+m}^{(1)} = \boldsymbol{q}_{N+m}^{(0)}, \forall m \in [M].\end{aligned}$$

The second layer only updates the user vertices as follows

$$\begin{aligned}\boldsymbol{q}_{N+m}^{(2)} = \boldsymbol{Z}_{15}\Bigg(\mathsf{ReLU}\Bigg(\boldsymbol{Z}_{14}\boldsymbol{q}_{N+m}^{(1)} + \sum_{n=1}^{N}\boldsymbol{Z}_{13}\Big(\mathsf{ReLU}\Big(\boldsymbol{Z}_{12}\boldsymbol{q}_{N+m}^{(1)} \\ + \boldsymbol{Z}_{11}\boldsymbol{q}_n^{(1)} + \boldsymbol{Z}_{10}\boldsymbol{e}_{n,N+m}\Big)\Big)\Bigg)\Bigg), \forall m \in [M] \\ \boldsymbol{q}_n^{(2)} = \boldsymbol{q}_n^{(1)}, \forall n \in [N].\end{aligned}$$

Such "split updating" of different nodes' embeddings in two layers has been advocated in [23] for the type of graph structure used in this work (i.e., a bipartite graph). Moreover, there is a potential saving in the computational cost in both training and testing [57] compared to updating all nodes' embeddings in each layer.

Finally, $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi})$ is computed using the $\boldsymbol{q}_{N+m}^{(2)}, \forall m \in [M]$ as follows:

$$\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}) = \mathsf{Sigmoid}\left(\frac{1}{M}\sum_{m=1}^{M}\boldsymbol{\beta}^{\top}\mathsf{ReLU}(\boldsymbol{Z}_{16}\boldsymbol{q}_{N+m}^{(2)})\right),$$

where $\boldsymbol{Z}_4, \ldots \boldsymbol{Z}_{16} \in \mathbb{R}^{E\times E}$, $\boldsymbol{\beta} \in \mathbb{R}^E$, and $\mathsf{Sigmoid} : \mathbb{R} \to \mathbb{R}$ is the sigmoid function, i.e., $\mathsf{Sigmoid}(x) = \frac{1}{1+\exp(-x)}$.

5.2.4 Evaluation Metrics

We define the *optimality gap* (Ogap) as follows:

$$\text{Ogap} := \frac{\|\widehat{\boldsymbol{W}}\|_{\text{F}}^2 - \|\boldsymbol{W}^{\star}\|_{\text{F}}^2}{\|\boldsymbol{W}^{\star}\|_{\text{F}}^2} \times 100\%,$$

Table 5.4: Performance of algorithms for $N \leq 16$ cases with perfect CSI. $\sigma_m^2 = 0.1, \gamma_m = 10.0, \forall m \in [M]$.

Problem Size (N, M, L)	Metric	MINIMAL	Greedy	IrCvxOpt	SL
$(6, 3, 3)$	Ogap	0.00	1.18	20.54	64.39
	speedup	1.73	0.92	4.68	17.70
	SOCPs	10.25	15.00	6.65	1
$(8, 4, 4)$	Ogap	0.0	0.83	20.19	38.08
	speedup	2.72	1.35	6.40	40.52
	SOCPs	14.9	26.0	12.05	1
$(10, 5, 5)$	Ogap	0.85	2.83	68.34	-
	speedup	4.10	2.46	8.47	-
	SOCPs	28.05	40.00	22.60	-
$(12, 6, 6)$	Ogap	2.16	3.43	234.88	-
	speedup	5.87	4.72	10.96	-
	SOCPs	49.00	57.00	27.90	-
$(16, 8, 8)$	Ogap	2.94	6.59	159.28	-
	speedup	12.39	23.88	78.62	-
	SOCPs	234.50	100.00	29.00	-

Table 5.5: Objective values, $\|\boldsymbol{W}\|_F^2$, attained by the algorithms for $N \geq 32$ cases with perfect CSI. $\sigma_m^2 = 0.1, \gamma_m = 10.0, \forall m \in [M]$.

Problem Size (N, M, L)	MINIMAL	Greedy	IrCvxOpt
$(32, 12, 12)$	4.35	21.73	12.44
$(64, 16, 16)$	5.23	61.66	72.73
$(128, 8, 8)$	1.86	22.45	3.13
$(128, 16, 16)$	4.60	40.29	163.93

where $\boldsymbol{W}^\star$ is the optimal solution provided by the B&B algorithm and $\widehat{\boldsymbol{W}}$ is the solution provided by an algorithm under test. We also define the runtime speedup as follows:

$$\text{speedup} := \frac{\text{Run-time of B\&B (seconds)}}{\text{Run-time of method under test (seconds)}}.$$

5.2.5 Results

Table 5.4 shows the performance of all methods under $\gamma_m = 10.0, \sigma_m^2 = 0.1, \forall m \in [M]$ for cases where $N \leq 16$. Results are averaged over 20 random test instances. One can see that `MINIMAL` consistently attains a very small Ogap ($< 3\%$ for all cases), whereas the baselines have much larger Ogaps. The `SL` method only requires solving a single SOCP, as the antenna selection part is done by the learned $\boldsymbol{f}_{\widehat{\boldsymbol{\theta}}}$. However, the solution quality is not acceptable, indicating that the learned neural network for AS performs poorly. Notably, in our simulations, we observed that `SL` needs a large amount of problem instances to generate its training data for a given (N, M, L). For example, under the settings in Table 5.4, $T = 12,000$ instances were used for `SL`, but only 600 instances were used for the proposed method.

Table 5.5 shows the performance of the algorithms in cases where $N \geq 32$. Note that generating training samples for `SL` is too costly in these case, and thus we drop this baseline in this table. This is because for each (N, M, L), one has to re-train $\boldsymbol{f}_{\boldsymbol{\theta}}$ from scratch under `SL`—but generating training examples for large size N is not affordable. For the proposed algorithm, we use the GNN trained on smaller problem size, i.e., $(N, M, L) = (16, 8, 8)$ (cf. Remark 2), which allows us to avoid re-training. In this simulation, we test all methods under limited computational budget (i.e., every method is allowed to use up to $2N$ SOCPs), for controlling the runtime. Unlike the previous cases where the Ogap is presented, we could only compare the objective values in this simulation, as obtaining the optimal solution is very costly. One can see that the proposed method attains objective values that are oftentimes order-of-magnitude smaller than those of the baselines. `IrCvxOpt` sometimes attains small objective values (e.g., when $(N, M, L) = (128, 8, 8)$), but the performance is not consistent across different cases.

Table 5.6 shows the performance of the algorithms under imperfect CSI using the RBF constraints. For $(N, M, L) = (16, 8, 8)$, we use the model trained on $(N, M, L) = (10, 5, 5)$, and limit the number of SDRs to $2N$. Similar to the perfect CSI case, the proposed method attains the smallest Ogap/objective value compared to all baselines. The `IrCvxOpt` again sometimes outputs acceptable results, but could not maintain consistently good performance over all cases.

Table 5.7 tests the algorithms' ability of finding feasible solutions of (2.9). Note that finding a feasible solution for QCQP problems is often highly nontrivial [50]. As

Table 5.6: Performance of algorithms under approximate CSI. $\sigma_m^2 = 0.1, \gamma_m = 10.0, \varepsilon_m = 0.02, \forall m \in [M]$.

Problem Size (N, M, L)	Metric	`MINIMAL`	`Greedy`	`IrCvxOpt`	`SL`
$(8, 4, 4)$	Ogap	0.09	1.27	4.97	21.97
	speedup	3.54	1.43	10.64	47.08
	SDRs	13.30	26.00	4.70	1.0
$(10, 5, 5)$	Ogap	2.04	2.20	10.72	-
	speedup	4.19	1.90	18.89	-
	SDRs	23.90	40.00	7.75	-
$(16, 8, 8)$	$\|\boldsymbol{W}\|_F^2$	2.93	24.39	3.15	-
	SDRs	34.00	34.00	18.25	-

Table 5.7: Performance of Algorithms under Various γ_m's with Approximate CSI. $(N, M, L) = (8, 4, 4)$, $\varepsilon_m = 0.02, \sigma_m^2 = 0.1, \forall m \in [M]$.

γ_m(dB) (# feasible ins.)	Metric	`MINIMAL`	`Greedy`	`IrCvxOpt`
30.00	Ogap	0.40	4.63	17.76
(50)	# feasible solutions	50	50	44
33.01	Ogap	0.51	11.21	45.07
(40)	# feasible solutions	40	39	32
34.77	Ogap	0.00	19.02	133.88
(25)	# feasibile solutions	25	25	21
36.02	Ogap	0.00	72.19	31.65
(10)	# feasible solutions	10	10	7

making $\|\boldsymbol{W}\|_{\text{row}-0} \leq L$ [cf. Eq. (2.9c)] can be easily done via simple post-processing (e.g., by thresholding some rows of the solution $\boldsymbol{W}$ to zeros), we primarily examine if the algorithms could find $\boldsymbol{W}$'s that satisfy the SINR specifications in (2.9b). To be specific, the algorithms are tested using various γ_m's. Naturally, higher values of γ_m may make all the SINR constraints hard to satisfy. We run 50 random trials. One can see that under γ_m =30dB, all the problem instances have at least a feasible solution for (2.9b). Both `MINIMAL` and `Greedy` can find solutions that are feasible for all instances, but `MINIMAL` enjoys a much smaller Ogap. When γ_m grows, the problem admits fewer infeasible instances. However, `MINIMAL` always returns a feasible solution, as long as the instance has one. `Greedy` also works fine for finding feasible solutions, but the Ogap

becomes much larger when γ_m increases. `IrCvxOpt` is less competitive in terms of both Ogap and feasibility.

5.3 Summary

In this chapter, we provided experimental results to demonstrate the efficacy of the proposed B&B algorithm and the ML-assisted B&B algorithm. First, we showed that the B&B procedure provides optimal solution under significantly less computation budget compared to the exhaustive search for the (R)BF&AS problem. Next, for small problem sizes, we showed that our ML-assisted scheme, `MINIMAL`, provides near-optimal solution compared to the baselines with comparable run-time. For large problem sizes, we showed that we could obtain high quality solutions using classifier trained on small problem size $N \leq 16$. This demonstrates the size-insensitivity property of the proposed GNN classifier.

Chapter 6: Conclusion and Discussion

In this work, we revisited the joint beamforming and antenna selection problem under perfect and imperfect CSI and proposed a machine learning-assisted B&B algorithm to attain its optimal solution. Unlike the vast majority of existing algorithms that rely on continuous optimization to approximate the hard mixed integer and nonconvex optimization problem without optimality guarantees, our B&B algorithm leverages the special properties of joint (R)BF&AS to come up with optimal solutions. More importantly, we proposed a GNN-based machine learning method to help accelerate the B&B algorithm. Our analysis showed that the design ensures provable acceleration and retains optimality with high probability, under proper GNN design and given a sufficiently enough sample size. To our best knowledge, this is the first comprehensive characterization for ML-based B&B. Our GNN design also easily handles a commonly seen challenge in communications, namely, the problem size change across training and test sets, without visible performance losses. Simulations corroborated our design goals and theoretical analyses.

Moving forward, a natural question is if the proposed ML-accelerated B&B method can be extended to offer efficient and optimal solutions to other joint (R)BF&AS criteria, e.g., those in [16, 26, 29, 51, 66, 72]. This can *in principle* be done, but the caveat lies in designing an effective B&B algorithm for the problem of interest. In our case, our B&B design leveraged the fact that (2.9) is optimally solvable when given a fixed set of antennas, which is a property that not all the joint BF&AS formulations enjoy—e.g., the multicast version of (2.9) cannot be handled by a similar B&B. Therefore, a meaningful future direction is to consider such more challenging cases and come up with a ML-assisted (near)-optimal method.

Bibliography

[1] Naman Agarwal, Alon Gonen, and Elad Hazan. Learning in non-convex games with an optimization oracle. In *Proc. COLT*, pages 18–29. PMLR, 2019.

[2] Ammar Ahmed, Shuimei Zhang, and Yimin D Zhang. Antenna selection strategy for transmit beamforming-based joint radar-communication system. *Digital Signal Process.*, 105:102768, 2020.

[3] MOSEK ApS. Mosek optimization suite.

[4] Aakash Arora, Christos G Tsinos, Symeon Chatzinotas, Björn Ottersten, et al. Analog beamforming with antenna selection for large-scale antenna arrays. In *Proc. IEEE ICASSP*, pages 4795–4799, 2021.

[5] Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Proc. NeurIPS*, volume 30, 2017.

[6] Mats Bengtsson and Björn Ottersten. Optimum and suboptimum transmit beamforming. In *Handbook of antennas in wireless communications*. CRC press, 2001.

[7] Dimitris Bertsimas, Angela King, and Rahul Mazumder. Best subset selection via a modern optimization lens. *The Annals of Statistics*, 44(2):813–852, 2016.

[8] Stephen Boyd and Jacob Mattingley. Branch and bound methods. *Notes for EE364b, Stanford University*, pages 2006–07, 2007.

[9] Emmanuel J Candes, Michael B Wakin, and Stephen P Boyd. Enhancing sparsity by reweighted ℓ 1 minimization. *J. Four. Analy. Appl.*, 14(5):877–905, 2008.

[10] Tsung-Hui Chang, Wing-Kin Ma, and Chong-Yung Chi. Worst-case robust multiuser transmit beamforming using semidefinite relaxation: Duality and implications. In *Proc. IEEE ASILOMAR*, pages 1579–1583, 2011.

[11] Jienan Chen, Siyu Chen, Yunlong Qi, and Shengli Fu. Intelligent massive MIMO antenna selection using monte carlo tree search. *IEEE Trans. Signal Process.*, 67(20):5380–5390, 2019.

[12] Minshuo Chen, Xingguo Li, and Tuo Zhao. On generalization bounds of a family of recurrent neural networks. In *Proc. AISTATS*, 2019.

[13] Runhua Chen, Jeffrey G Andrews, and Robert W Heath. Efficient transmit antenna selection for multiuser MIMO systems with block diagonalization. In *IEEE GLOBECOM*, pages 3499–3503, 2007.

[14] Ali Civril and Malik Magdon-Ismail. On selecting a maximum volume sub-matrix of a matrix and related problems. *Theoretical Computer Science*, 410(47-49):4801–4811, 2009.

[15] Jens Clausen. Branch and bound algorithms-principles and examples, 1999.

[16] Lin Dai, Sana Sfar, and Khaled Ben Letaief. Optimal antenna selection based on capacity maximization for MIMO systems in correlated channels. *IEEE Trans. Commun.*, 54(3):563–573, 2006.

[17] Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *JMLR*, 17(83):1–5, 2016.

[18] Ming Ding, Shi Liu, Hanwen Luo, and Wen Chen. MMSE based greedy antenna selection scheme for AF MIMO relay systems. *IEEE Signal Process. Lett.*, 17(5):433–436, 2010.

[19] Ahmet M Elbir and Kumar Vijay Mishra. Joint antenna selection and hybrid beamformer design using unquantized and quantized deep learning networks. *IEEE Trans. Wireless Commun.*, 19(3):1677–1688, 2019.

[20] Yuan Gao, Wei Jiang, and Thomas Kaiser. Bidirectional branch and bound based antenna selection in massive MIMO systems. In *Proc. IEEE PIMRC*, pages 563–568, 2015.

[21] Yuan Gao, Han Vinck, and Thomas Kaiser. Massive MIMO antenna selection: Switching architectures, capacity bounds, and optimal antenna selection algorithms. *IEEE Trans. Signal Process.*, 66(5):1346–1360, 2017.

[22] Vikas Garg, Stefanie Jegelka, and Tommi Jaakkola. Generalization and representational limits of graph neural networks. In *Proc. ICML*, pages 3419–3430, 2020.

[23] Maxime Gasse, Didier Chételat, Nicola Ferroni, Laurent Charlin, and Andrea Lodi. Exact combinatorial optimization with graph convolutional neural networks. In *Proc. NeurIPS*, volume 32, 2019.

[24] Alex B Gershman, Nicholas D Sidiropoulos, Shahram Shahbazpanahi, Mats Bengtsson, and Bjorn Ottersten. Convex optimization-based beamforming. *IEEE Signal Process. Mag.*, 27(3):62–75, 2010.

[25] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *Proc. ICML*, pages 1263–1272, 2017.

[26] Mohammad-Hossein Golbon-Haghighi. Beamforming in wireless networks. *InTech Open*, pages 163–192, 2016.

[27] He He, Hal Daume III, and Jason M Eisner. Learning to search in branch and bound algorithms. In *Proc. NeurIPS*, volume 27, pages 3293–3301, 2014.

[28] Mohamed S Ibrahim, Ahmed S Zamzam, Xiao Fu, and Nicholas D Sidiropoulos. Learning-based antenna selection for multicasting. In *Proc. IEEE SPAWC*, pages 1–5, 2018.

[29] Mohamed Salah Ibrahim, Aritra Konar, and Nicholas D Sidiropoulos. Fast algorithms for joint multicast beamforming and antenna selection in massive MIMO. *IEEE Trans. Signal Process.*, 68:1897–1909, 2020.

[30] Chenzi Jiang and Leonard J. Cimini. Antenna selection for energy-efficient MIMO transmission. *IEEE Wireless Commun. Lett.*, 1(6):577–580, 2012.

[31] Jingon Joung. Machine learning-based antenna selection in wireless communications. *IEEE Commun. Lett.*, 20(11):2241–2244, 2016.

[32] Eleftherios Karipidis, Nicholas D Sidiropoulos, and Zhi-Quan Luo. Quality of service and max-min fair transmit beamforming to multiple co-channel multicast groups. *IEEE Trans. Signal Process.*, 56(3):1268–1279, 2008.

[33] Seung-Jean Kim, Alessandro Magnani, Almir Mutapcic, Stephen P Boyd, and Zhi-Quan Luo. Robust beamforming via worst-case SINR maximization. *IEEE Trans. Signal Process.*, 56(4):1539–1547, 2008.

[34] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. ICLR*, 2015.

[35] Aritra Konar and Nicholas D Sidiropoulos. A simple and effective approach for transmit antenna selection in multiuser massive MIMO leveraging submodularity. *IEEE Trans. Signal Process.*, 66(18):4869–4883, 2018.

[36] Ailsa H Land and Alison G Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497–520, 1960.

[37] Mengyuan Lee, Guanding Yu, and Geoffrey Ye Li. Learning to branch: Accelerating resource allocation in wireless networks. *IEEE Trans. Veh. Technol.*, 69(1):958–970, 2019.

[38] Woongsup Lee, Minhoe Kim, and Dong-Ho Cho. Deep power control: Transmit power control scheme based on convolutional neural network. *IEEE Commun. Lett.*, 22(6):1276–1279, 2018.

[39] Yuzhou Li, Min Sheng, Xijun Wang, Yan Shi, and Yan Zhang. Globally optimal antenna selection and power allocation for energy efficiency maximization in downlink distributed antenna systems. In *Proc. IEEE GLOBCOM*, pages 3856–3861, 2014.

[40] Bo Lin, Feifei Gao, Shun Zhang, Ting Zhou, and Ahmed Alkhateeb. Deep learning-based antenna selection and CSI extrapolation in massive MIMO systems. *IEEE Trans. Wireless Commun.*, 20(11):7669–7681, 2021.

[41] An Liu and Vincent KN Lau. Joint power and antenna selection optimization in large cloud radio access networks. *IEEE Trans. Signal Process.*, 62(5):1319–1328, 2014.

[42] Cheng Lu and Ya-Feng Liu. An efficient global algorithm for single-group multicast beamforming. *IEEE Trans. Signal Process.*, 65(14):3761–3774, 2017.

[43] Cheng Lu, Ya-Feng Liu, and Jing Zhou. An enhanced SDR based global algorithm for nonconvex complex quadratic programs with signal processing applications. *IEEE Open J. Signal Process.*, 1:120–134, 2020.

[44] Changqing Luo, Jinlong Ji, Qianlong Wang, Xuhui Chen, and Pan Li. Channel state information prediction for 5g wireless communications: A deep learning approach. *IEEE Transactions on Network Science and Engineering*, 7(1):227–236, 2018.

[45] Zhi-Quan Luo, Wing-Kin Ma, Anthony Man-Cho So, Yinyu Ye, and Shuzhong Zhang. Semidefinite relaxation of quadratic optimization problems. *IEEE Signal Proc. Mag.*, 27(3):20–34, 2010.

[46] Wing-Kin Ma, Jiaxian Pan, Anthony Man-Cho So, and Tsung-Hui Chang. Unraveling the rank-one solution mystery of robust MISO downlink transmit optimization: A verifiable sufficient condition via a new duality result. *IEEE Trans. Signal Process.*, 65(7):1909–1924, 2017.

[47] Hasan F Mahdi, Ahmed T Alheety, Nather A Hamid, and Sefer Kurnaz. Quantization-aware greedy antenna selection for multi-user massive MIMO systems. *Progress in Electromagnetics Research C*, 111:15–24, 2021.

[48] José Carlos Marinello, Taufik Abrão, Abolfazl Amiri, Elisabeth de Carvalho, and Petar Popovski. Antenna selection for improving energy efficiency in XL-MIMO systems. *IEEE Trans. Veh. Technol.*, 69(11):13305–13318, 2020.

[49] Thomas L Marzetta and Bertrand M Hochwald. Fast transfer of channel state information in wireless systems. *IEEE Transactions on Signal Processing*, 54(4):1268–1278, 2006.

[50] Omar Mehanna, Kejun Huang, Balasubramanian Gopalakrishnan, Aritra Konar, and Nicholas D Sidiropoulos. Feasible point pursuit and successive approximation of non-convex QCQPs. *IEEE Signal Process. Lett.*, 22(7):804–808, 2014.

[51] Omar Mehanna, Nicholas D Sidiropoulos, and Georgios B Giannakis. Joint multicast beamforming and antenna selection. *IEEE Trans. Signal Process.*, 61(10):2660–2674, 2013.

[52] Marcele OK Mendonca, Paulo SR Diniz, Tadeu N Ferreira, and Lisandro Lovisolo. Antenna selection in massive MIMO based on greedy algorithms. *IEEE Trans. Wireless Commun.*, 19(3):1868–1881, 2019.

[53] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.

[54] Andreas F Molisch and Moe Z Win. Mimo systems with antenna selection. *IEEE Microw. Mag.*, 5(1):46–56, 2004.

[55] Andreas F Molisch, Moe Z Win, Yang-Seok Choi, and Jack H Winters. Capacity of MIMO systems with antenna selection. *IEEE Trans. Wireless Commun.*, 4(4):1759–1772, 2005.

[56] Vinod Nair, Sergey Bartunov, Felix Gimeno, Ingrid von Glehn, Pawel Lichocki, Ivan Lobov, Brendan O'Donoghue, Nicolas Sonnerat, Christian Tjandraatmadja, Pengming Wang, et al. Solving mixed integer programs using neural networks. *arXiv preprint arXiv:2012.13349*, 2020.

[57] Marcel Nassar. Hierarchical bipartite graph convolution networks. *arXiv preprint arXiv:1812.03813*, 2018.

[58] Chongjun Ouyang, Zeliang Ou, Lu Zhang, and Hongwen Yang. Optimal transmit antenna selection algorithm in massive MIMOME channels. In *Proc. IEEE WCNC*, pages 1–6, 2019.

[59] David Pollard. Empirical processes: Theory and applications. *NSF-CBMS Reg. Conf. Ser. Prob. Stat.*, 2:i–86, 1990.

[60] Farrokh Rashid-Farrokhi, KJ Ray Liu, and Leandros Tassiulas. Transmit beamforming and power control for cellular wireless systems. *IEEE J. Sel. Areas Commun.*, 16(8):1437–1450, 1998.

[61] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proc. AISTATS*, pages 627–635, 2011.

[62] Mirette Sadek, Alireza Tarighat, and Ali H. Sayed. Active antenna selection in multiuser MIMO communications. *IEEE Trans. Signal Process.*, 55(4):1498–1510, 2007.

[63] Shahab Sanayei and Aria Nosratinia. Antenna selection in MIMO systems. *IEEE Commun. Mag.*, 42(10):68–73, 2004.

[64] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Trans. Neural Netw.*, 20(1):61–80, 2008.

[65] Yifei Shen, Yuanming Shi, Jun Zhang, and Khaled B Letaief. LORM: Learning to optimize for resource management in wireless networks with few training samples. *IEEE Trans. Wireless Commun.*, 19(1):665–679, 2019.

[66] Yuanming Shi, Jun Zhang, and Khaled B Letaief. Group sparse beamforming for green cloud-RAN. *IEEE Trans. Wireless Commun.*, 13(5):2809–2823, 2014.

[67] Nikos D Sidiropoulos, Timothy N Davidson, and Zhi-Quan Luo. Transmit beamforming for physical-layer multicasting. *IEEE Trans. Signal Process.*, 54(6):2239–2251, 2006.

[68] Enbin Song, Qingjiang Shi, Maziar Sanjabi, Ruo-Yu Sun, and Zhi-Quan Luo. Robust SINR-constrained MISO downlink beamforming: When is semidefinite programming relaxation tight? *EURASIP J. Wireless Commun. Netw.*, 2012(1):1–11, 2012.

[69] Haoran Sun, Xiangyi Chen, Qingjiang Shi, Mingyi Hong, Xiao Fu, and Nicholas D Sidiropoulos. Learning to optimize: Training deep neural networks for interference management. *IEEE Trans. Signal Process.*, 66(20):5438–5453, 2018.

[70] Eugene Visotsky and Upamanyu Madhow. Optimum beamforming using transmit antenna arrays. In *Proc. IEEE VTC*, volume 1, pages 851–856, 1999.

[71] Thang X Vu, Symeon Chatzinotas, Van-Dinh Nguyen, Dinh Thai Hoang, Diep N Nguyen, Marco Di Renzo, and Björn Ottersten. Machine learning-enabled joint antenna selection and precoding design: From offline complexity to online performance. *IEEE Trans. Wireless Commun.*, 20(6):3710–3722, 2021.

[72] Kun-Yu Wang, Anthony Man-Cho So, Tsung-Hui Chang, Wing-Kin Ma, and Chong-Yung Chi. Outage constrained robust transmit optimization for multiuser MISO downlinks: Tractable approximations by conic optimization. *IEEE Trans. Signal Process.*, 62(21):5690–5705, 2014.

[73] Gan Zheng, Kai-Kit Wong, and Tung-Sang Ng. Robust linear MIMO in the downlink: A worst-case optimization with ellipsoidal uncertainty regions. *EURASIP J. Adv. Signal Process.*, 2008:1–15, 2008.

APPENDICES

Appendix A: Poof of Lemma 3

(a) The BF setting implies that $\mathcal{C}(\boldsymbol{w}_m, \boldsymbol{h}_m, \varepsilon_m, \sigma_m)$ is from (2.2b). Then, the equivalence of (2.2b) and (2.3) implies that (3.6) for any node $\mathcal{N}_s^{(\ell)}$ can be optimally solved using SOCP. Hence Lemma 3(a) holds due to Lemma 1.

(b) Note that (3.6) with $\mathcal{B}_s^{(\ell)}$ is equivalent to (2.4) with antennas restricted to the set $[N]\backslash\mathcal{B}_s^{(\ell)}$. Hence, when the condition in (3.7) is satisfied for $\boldsymbol{H}([N]\backslash\mathcal{B}_s^{(\ell)}, :)$, then (3.6) with $\mathcal{B}_s^{(\ell)}$ can be optimally solved using SDR due to Lemma 2. Further, the B&B procedure ensures that $|\mathcal{B}_s^{(\ell)}| \leq N - L, \forall(s, \ell)$. Hence, the set $\{\boldsymbol{H}(\mathcal{S}, :)|\mathcal{S} \in [N], |\mathcal{S}| \geq L\}$ includes all possible instances of (3.6) encountered during the B&B procedure. Therefore, Lemma 3(b) holds.

(c) Note that $|\widetilde{\mathcal{B}}_s^{(\ell)}| = N - L$. Hence, the solution of Problem (3.8) satisfies the constraint (2.9c). Further, due to Lemma 3 (a) and (b), Problem (3.8) can be optimally solved using SOCP and SDR for the BF and RBF cases, respectively. Hence, $\Phi_{\text{ub}}(\mathcal{N}_s^{(\ell)})$ is a valid upper bound of the optimum of (2.9).

Appendix B: Proof of Theorem 1

B.1 Proof of (a) and (b)

Note that if the SOCP and SDR return optimal solutions to every leaf node of the B&B tree, then the B&B procedure is ensured to find the optimal solutions of the the joint BF/RBF&AS problems. The reason is that the B&B tree only has a finite number of leaves.

For the BF setting with perfect CSI, the subproblem at a leaf node (ℓ, s) can be expressed as

$$\begin{aligned} \underset{\boldsymbol{W}}{\text{minimize}} \ & \|\boldsymbol{W}\|_F^2 \\ \text{subject to} \ & \frac{|\boldsymbol{w}_m^H \boldsymbol{h}_m|^2}{\sum_{\ell \neq m} |\boldsymbol{w}_\ell^H \boldsymbol{h}_m|^2 + \sigma_m^2} \geq \gamma_m, \quad \forall m \in [M] \\ & \boldsymbol{W}(n,:) = \boldsymbol{0}, \quad \forall n \in \mathcal{B}_s^{(\ell)}, \end{aligned} \tag{B.1}$$

where $|\mathcal{B}_s^{(\ell)}| = N - L$. Since $\|\boldsymbol{W}\|_{\text{row}-0} \leq L$ is automatically satisfied, it is omitted. Problem (B.1) can be rewritten as

$$\begin{aligned} \underset{\boldsymbol{W}_s^{(\ell)}}{\text{minimize}} \ & \|\boldsymbol{W}_s^{(\ell)}\|_F^2 \\ \text{subject to} \ & \frac{|\boldsymbol{w}_m^H \boldsymbol{h}_m|^2}{\sum_{\ell \neq m} |\boldsymbol{w}_\ell^H \boldsymbol{h}_m|^2 + \sigma_m^2} \geq \gamma_m, \quad \forall m \in [M] \end{aligned} \tag{B.2}$$

where $\boldsymbol{W}_s^{(\ell)} = \boldsymbol{W}([N] \backslash \mathcal{B}_s^{(\ell)}, :)$, and we let $\boldsymbol{w}_m = \boldsymbol{W}_s^{(\ell)}(:, m)$ by slightly abusing the notation. Since Problem (B.2) can be recast as a convex problem as detailed in (2.3), the solution to the above is indeed optimal.

Similarly, under the RBF setting with imperfect CSI, the subproblem at a leaf node

can be written as

$$\begin{aligned} \underset{\boldsymbol{W}}{\text{minimize}} \;\; & \|\boldsymbol{W}\|_F^2 \\ \text{subject to} \;\; & \min_{\overline{\boldsymbol{h}}_m \in \mathcal{U}_m} \frac{\overline{\boldsymbol{h}}_m^H \boldsymbol{W}_m \overline{\boldsymbol{h}}_m}{\sum_{j \neq m} \overline{\boldsymbol{h}}_m^H \boldsymbol{W}_j \overline{\boldsymbol{h}}_m + \sigma_m^2} \geq \gamma_m, \\ & \boldsymbol{W}(n,:) = \boldsymbol{0}, \quad \forall n \in \mathcal{B}_s^{(\ell)}, \end{aligned} \tag{B.3}$$

where $|\mathcal{B}_s^{(\ell)}| = N - L$. Problem (B.3) can be further rewritten as

$$\begin{aligned} \underset{\boldsymbol{W}_s^{(\ell)}}{\text{minimize}} \;\; & \|\boldsymbol{W}_s^{(\ell)}\|_F^2 \\ \text{subject to} \;\; & \min_{\overline{\boldsymbol{h}}_m \in \mathcal{U}_m} \frac{\overline{\boldsymbol{h}}_m^H \boldsymbol{W}_m \overline{\boldsymbol{h}}_m}{\sum_{j \neq m} \overline{\boldsymbol{h}}_m^H \boldsymbol{W}_j \overline{\boldsymbol{h}}_m + \sigma_m^2} \geq \gamma_m, \end{aligned} \tag{B.4}$$

where $\boldsymbol{W}_s^{(\ell)}$ and $\boldsymbol{w}_m$ are defined as in (B.2), and $\boldsymbol{h}_m = \boldsymbol{H}_s^{(\ell)}(:,m)$ with $\boldsymbol{H}_s^{(\ell)} = \boldsymbol{H}([N] \backslash \mathcal{B}_s^{(\ell)}, :)$ (recall that $\mathcal{U}_m := \{\boldsymbol{h}_m + \boldsymbol{e} | \|\boldsymbol{e}\|_2 \leq \varepsilon_m\}$). Using the condition in Theorem 1 (b), and invoking Lemma 3, one can see that (B.4) can be solved optimally using SDR.

B.2 Proof of (c)

B.2.1 Amount of SOCPs/SDRs Solved by Proposed B&B

In our B&B procedure, (3.6) and (3.8) are equivalent for any node and its right child node, i.e.,

$$\Phi_{\text{lb}}\left(\mathcal{N}_s^{(\ell)}\right) = \Phi_{\text{lb}}\left(\mathcal{N}_{s_2}^{(\ell+1)}\right), \Phi_{\text{ub}}\left(\mathcal{N}_s^{(\ell)}\right) = \Phi_{\text{ub}}\left(\mathcal{N}_{s_2}^{(\ell+1)}\right).$$

The first equation is because $\mathcal{B}_s^{(\ell)} = \mathcal{B}_{s_2}^{(\ell+1)}$ and the second because $\widetilde{\mathcal{B}}_s^{(\ell)}, \forall(\ell, s)$ in (3.8) is determined using the solution to (3.6). Hence, one can avoid redundant computations in the nodes by storing and reusing the results of (3.6) and (3.8). Using this observation, we derive an upper bound of the number of SOCPs/SDRs that need to be solved by the B&B.

Consider a B&B tree where none of the nodes are fathomed (Fig. B.1). Note that

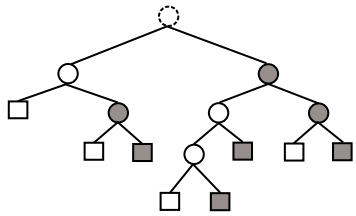

Figure B.1: Illustration of a B&B tree (where no nodes are fathomed).

there are $Q_{\text{Leaf}} = \binom{N}{L}$ leaf nodes (squares in Fig. B.1). Therefore, there are $Q_{\text{Total}} = 2\binom{N}{L} - 1$ nodes in total (all circles and squares). Each non-leaf node (circles) is branched into a right child node and a left child node. Hence, there are $Q_{\text{Right}} = \binom{N}{L} - 1$ right child nodes (shaded solid circles and squares) and $Q_{\text{Left}} = \binom{N}{L} - 1$ left child nodes (unshaded solid circles and squares).

The constraints of the SOCPs/SDRs corresponding to the leaf nodes can be different from that of its parent even if they correspond to a right child node, i.e., shaded squares. This is because of the update step in (3.11) for the leaf nodes. To explain, a right child node, $\mathcal{N}_s^{(\ell)}$, is converted into a leaf node if L of the decided antennas are included, i.e., $|\mathcal{A}_s^{(\ell)}| = L$. For this node, $\mathcal{B}_s^{(\ell)} = [N] \backslash \mathcal{A}_s^{(\ell)}$, i.e., all remaining undecided antennas are excluded. Since $\mathcal{B}_s^{(\ell)}$ will be different from that of its parent node, the solutions of (3.6) and (3.8) can be different from that of its parent node.

Therefore, only the non-leaf right child nodes (shaded solid circles) can reuse previously stored upper bound and lower bound solutions from their parents. Let $Q_{\text{RightLeaf}}$ denote the number of right child leaf nodes (shaded squares). Then, the total number of nodes whose associated SOCPs/SDRs that need to be solved in the worst case is $Q_{\text{Compute}} = Q_{\text{Total}} - Q_{\text{Right}} + Q_{\text{RightLeaf}}$.

To count $Q_{\text{RightLeaf}}$, notice that the right and left child nodes of a parent node correspond to 'including' and 'excluding' an antenna, respectively. A parent node is branched into a right child leaf node if it contains exactly $L-1$ included antennas and fewer than or equal to $N-L-1$ excluded antennas. This implies that there can be fewer than or equal to $(L-1)+(N-L-1) = N-2$ decided antennas. Hence, a right child leaf node is created whenever a node has $\leq N-2$ decided antennas, where $L-1$

of them are included, is branched. Therefore, we have the following holds:

$$\begin{aligned} Q_{\text{RightLeaf}} &= \binom{N-2}{L-1} + \binom{N-3}{L-1} + \cdots + \binom{L-1}{L-1} \\ &= \sum_{i=2}^{N-L+1} \binom{N-i}{L-1}. \end{aligned}$$

Consequently,

$$Q_{\text{Compute}} = \binom{N}{L} + \sum_{i=2}^{N-L+1} \binom{N-i}{L-1}.$$

Note that Q_{Compute} nodes may correspond to $2Q_{\text{Compute}}$ SOCPs/SDRs (cf. (3.6) and (3.8) for each node). However, for the leaf nodes (3.6) and (3.8) are identical. Hence there are only $Q_{\text{Compute}} - \binom{N}{L}$ instances of (3.6). Moreover, there can be at most $\binom{N}{L}$ instances of (3.8), since $\binom{N}{L}$ correspond to selecting L out of N antennas. Therefore, there are at most Q_{Compute} SDRs/SOCPs solved by the B&B procedure.

B.2.2 The SOCPS/SDRs Needed in B&B for Problem (3.12)

To complete the proof, let us examine the number of SOCPs/SDRs that are needed to exhaust the B&B tree of the formulation in (3.12).

A node problem of (3.12), for the node $\mathcal{N}_s^{(\ell)}$ is as follows:

$$\begin{aligned} \underset{\boldsymbol{W},\boldsymbol{z}}{\text{minimize}} \ & \|\boldsymbol{W}\|_F^2 \\ \text{subject to } & \mathcal{C}(\boldsymbol{w}_m, \boldsymbol{h}_m, \varepsilon_m, \sigma_m) \geq \gamma_m, \\ & \boldsymbol{z} \in \{0,1\}^N, \ \boldsymbol{z}^\top \mathbf{1} \leq L, \\ & z(n) = 0, \ n \in \mathcal{B}_s^{(\ell)}, \ z(n) = 1, \ n \in \mathcal{A}_s^{(\ell)}, \\ & \|\boldsymbol{W}(n,:)\|_2 \leq C z(n), \ \forall n \in [N]. \end{aligned} \tag{B.5}$$

The lower bound is obtained by solving the convex relaxation of the above, i.e., $\boldsymbol{z} \in \{0,1\}$ is relaxed to $\boldsymbol{z} \in [0,1]^N$. One can see that the lower bounds obtained at the parent node and both child nodes may be different.

It is because (B.5) depends upon both $\mathcal{A}_s^{(\ell)}$ and $\mathcal{B}_s^{(\ell)}$ and each child node will differ from its parent in one of the two sets, i.e, $\mathcal{B}_{s_1}^{(\ell+1)} \neq \mathcal{B}_s^{(\ell)}$ and $\mathcal{A}_{s_2}^{(\ell+1)} \neq \mathcal{A}_s^{(\ell)}$. The above

implies that the number of SOCPs/SDRs with B&B using (B.5) has an upper bound of $Q'_{\text{Compute}} = 2\binom{N}{L} - 1$ (specially, with $\binom{N}{L}$ instances of (3.8) and $\binom{N}{L} - 1$ instances of (3.6)).

Appendix C: Proof of Lemma 4

We use the empirical Rademacher complexity of the GNN class to assist finding the expected risk's error, which is a classic way of establishing generalization bounds [5, 12, 53]. To proceed, let us define the sets

$$\begin{aligned}
\mathcal{X}_{\boldsymbol{\phi}} &:= \big\{\boldsymbol{\phi} = \Big[\boldsymbol{x}_1^\top, \ldots, \boldsymbol{x}_U^\top, \boldsymbol{e}_{1,1}^\top, \ldots, \boldsymbol{e}_{U,U}^\top\Big] \\
&\quad \big| \ \|\boldsymbol{x}_u\|_2, \|\boldsymbol{e}_{u,v}\|_2 \leq B_{\boldsymbol{x}}, \forall u, v \in [U]\big\}, \\
\mathcal{X}_{\boldsymbol{Z}} &:= \{\boldsymbol{Z} \in \mathbb{R}^{E\times E} | \|\boldsymbol{Z}\|_2 \leq \mathcal{B}_{\boldsymbol{Z}}\}, \text{ and} \\
\mathcal{X}_{\boldsymbol{\beta}} &:= \{\boldsymbol{a} \in \mathbb{R}^E \mid \|\boldsymbol{a}\|_2 \leq B_{\boldsymbol{\beta}}\}.
\end{aligned}$$

First, consider the following lemma:

Lemma 5 ([53, Theorem 3.1])**.** *Let $\mathcal{T}$ be a family of functions mapping from $\mathcal{X}_{\boldsymbol{\phi}} \times \{0,1\}$ to $[-b, b]$. Assume $\mathcal{G}$ consists of K i.i.d. samples $\{\boldsymbol{\phi}_k, y_k\}_{k=1}^K$. With probability at least $1-\delta$ over the samples $\mathcal{G}$, for any $\tau \in \mathcal{T}$,*

$$\mathbb{E}[\tau(\boldsymbol{\phi}, y)] - \frac{1}{K}\sum_{(\boldsymbol{\phi}_k, y_k)\in\mathcal{G}} \tau(\boldsymbol{\phi}_k, y_k) \leq 2\widehat{\mathcal{R}}_{\mathcal{G}}(\mathcal{T}) + 3b\sqrt{\frac{\log 2/\delta}{2K}},$$

where $\widehat{\mathcal{R}}_{\mathcal{G}}(\mathcal{T})$ is the empirical Rademacher complexity [53] of the set $\mathcal{T}$ with respect to the samples $\mathcal{G}$.

Let us define the set $\mathcal{T} := \{(\boldsymbol{\phi}, y) \mapsto \mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}), y) \mid \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$, a class of functions that maps from $\mathcal{X}_{\boldsymbol{\phi}} \times \{0,1\}$ to $[-B_{\mathcal{L}}, B_{\mathcal{L}}]$. Then, applying Lemma 5 to $\mathcal{T}$ over the set $\mathcal{G}$ ensures that with probability at least $1-\delta$ over $\mathcal{G}$, $\forall \boldsymbol{\theta} \in \boldsymbol{\Theta}$,

$$\begin{aligned}
&\mathbb{E}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}), y)] - \frac{1}{K}\sum_{i\in[K]} \mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_i), y_i) \\
&\leq 2\widehat{\mathcal{R}}_{\mathcal{G}}(\mathcal{T}) + 3B_{\mathcal{L}}\sqrt{\frac{\log 2/\delta}{2K}},
\end{aligned} \tag{C.1}$$

In the following, we derive an upper bound on $\widehat{\mathcal{R}}_{\mathcal{G}}(\mathcal{T})$. To this end, we instead define a set $\mathbf{\Pi} := \{\boldsymbol{\phi} \mapsto \boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}) | \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$, and derive $\widehat{\mathcal{R}}_{\mathcal{G}}(\mathbf{\Pi})$. With this, we can use Talagrand's Lemma [53, Lemma 4.2] to obtain $\widehat{\mathcal{R}}_{\mathcal{G}}(\mathcal{T})$ as $\widehat{\mathcal{R}}_{\mathcal{G}}(\mathcal{T}) = C_{\mathcal{L}}\widehat{\mathcal{R}}_{\mathcal{G}}(\mathbf{\Pi})$.

In order to derive $\widehat{\mathcal{R}}_{\mathcal{G}}(\mathbf{\Pi})$, we use Dudley's entropy integral [5, Lemma A.5], which provides an upper bound on the empirical Rademacher complexity by using the *covering number* of $\mathbf{\Pi}$. To clarify, a μ-cover of $\mathbf{\Pi}$ is any set $\mathcal{C} \subseteq \mathbf{\Pi}$ such that $\forall \boldsymbol{\pi}_{\boldsymbol{\theta}} \in \mathbf{\Pi}$, $\exists \boldsymbol{\pi}_{\widetilde{\boldsymbol{\theta}}} \in \mathcal{C}$ such that

$$\max_{\boldsymbol{\phi} \in \mathcal{X}_{\boldsymbol{\phi}}} \left| \boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}) - \boldsymbol{\pi}_{\widetilde{\boldsymbol{\theta}}}(\boldsymbol{\phi}) \right| \leq \mu.$$

Similarly, the covering number of the set $\mathbf{\Pi}$ at scale μ is denoted by $\mathsf{N}(\mathbf{\Pi}, \mu)$ and defined as the minimum cardinality of a μ-cover set of $\mathbf{\Pi}$. The following lemma summarizes the Dudley's entropy integral that uses the covering number of a set to bound its empirical Rademacher complexity.

Lemma 6 ([5, Lemma A.5])**.** *Given samples $\mathcal{G}$ of size K, the empirical Rademacher complexity of the set $\mathbf{\Pi}$ with respect to the samples $\mathcal{G}$ is upperbounded as follows:*

$$\widehat{\mathcal{R}}_{\mathcal{G}}(\mathbf{\Pi}) \leq \inf_{a>0} \left(\frac{4a}{\sqrt{K}} + \frac{12}{K} \int_{a}^{\sqrt{K}} \sqrt{\log \mathsf{N}(\mathbf{\Pi}, \mu)} d\mu \right). \tag{C.2}$$

To proceed with the derivation of $\log(\mathsf{N}(\mathbf{\Pi}, \mu))$, we first characterize the Lipschitz constants of the GNN with respect to its parameters. Consider parameters $\boldsymbol{\theta}$ and $\widetilde{\boldsymbol{\theta}}$, which correspond to $(\boldsymbol{Z}_1, \boldsymbol{Z}_2, \boldsymbol{Z}_3, \boldsymbol{\beta})$ and $(\widetilde{\boldsymbol{Z}}_1, \widetilde{\boldsymbol{Z}}_2, \widetilde{\boldsymbol{Z}}_3, \widetilde{\boldsymbol{\beta}})$, respectively. Let $\boldsymbol{q}_u^{(d)}$ and $\widetilde{\boldsymbol{q}}_u^{(d)}$ denote the embeddings learned for the uth vertex at the end of dth layer of the GNN with parameters $\boldsymbol{\theta}$ and $\widetilde{\boldsymbol{\theta}}$, respectively. Then, for any input $\boldsymbol{\phi}$, the absolute difference

between the outputs of the two GNNs can be written as

$$\begin{aligned}
&\left|\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}) - \boldsymbol{\pi}_{\widetilde{\boldsymbol{\theta}}}(\boldsymbol{\phi})\right| \\
&= \left| \frac{1}{U} \sum_{u\in[U]} \left(\boldsymbol{\zeta}(\boldsymbol{\beta}^\top \boldsymbol{q}_u^{(D)}) - \boldsymbol{\zeta}(\widetilde{\boldsymbol{\beta}}^\top \widetilde{\boldsymbol{q}}_u^{(D)}) \right) \right| \\
&\leq \frac{1}{U} \sum_{u\in[U]} C_{\boldsymbol{\zeta}} \left| \boldsymbol{\beta}^\top \boldsymbol{q}_u^{(D)} - \widetilde{\boldsymbol{\beta}}^\top \boldsymbol{q}_u^{(D)} + \widetilde{\boldsymbol{\beta}}^\top \boldsymbol{q}_u^{(D)} + \widetilde{\boldsymbol{\beta}}^\top \widetilde{\boldsymbol{q}}_u^{(D)} \right| \\
&\leq \frac{C_{\boldsymbol{\zeta}}}{U} \sum_{u\in[U]} \left(\left\| \boldsymbol{q}_u^{(D)} \right\|_2 \left\| \boldsymbol{\beta} - \widetilde{\boldsymbol{\beta}} \right\|_2 + B_{\boldsymbol{\beta}} \left\| \boldsymbol{q}_u^{(D)} - \widetilde{\boldsymbol{q}}_u^{(D)} \right\|_2 \right).
\end{aligned} \tag{C.3}$$

First, we can bound $\|\boldsymbol{q}_u^{(D)}\|_2$ as follows:

$$\begin{aligned}
&\left\| \boldsymbol{q}_u^{(D)} \right\|_2 \\
&= \left\| \boldsymbol{\xi} \left(\boldsymbol{Z}_1 \boldsymbol{q}_u^{(D-1)} + \sum_{(u,v)\in\mathcal{E}} \boldsymbol{\xi} \left(\boldsymbol{Z}_2 \boldsymbol{q}_v^{(D-1)} + \boldsymbol{Z}_3 \boldsymbol{e}_{u,v} \right) \right) - \boldsymbol{\xi}(0) \right\|_2 \\
&\leq C_{\boldsymbol{\xi}} \|\boldsymbol{Z}_1\|_2 \left\| \boldsymbol{q}_u^{(D-1)} \right\|_2 \\
&+ C_{\boldsymbol{\xi}}^2 \sum_{(u,v)\in\mathcal{E}} \left(\|\boldsymbol{Z}_2\|_2 \left\| \boldsymbol{q}_v^{(D-1)} \right\|_2 + \|\boldsymbol{Z}_3\|_2 \|\boldsymbol{e}_{u,v}\|_2 \right) \\
&\leq C_{\boldsymbol{\xi}} B_{\boldsymbol{Z}} \left\| \boldsymbol{q}_u^{(D-1)} \right\|_2 + C_{\boldsymbol{\xi}}^2 U \max_v \left(B_{\boldsymbol{Z}} \left\| \boldsymbol{q}_v^{(D-1)} \right\|_2 + B_{\boldsymbol{Z}} B_{\boldsymbol{x}} \right).
\end{aligned}$$

Solving the recursion from the final inequality, we obtain

$$\left\| \boldsymbol{q}_u^{(D)} \right\|_2 \leq \alpha^D B_{\boldsymbol{x}} + U C_{\boldsymbol{\xi}}^2 B_{\boldsymbol{Z}} B_{\boldsymbol{x}} \frac{\alpha^D - 1}{\alpha - 1}, \tag{C.4}$$

where $\alpha = ((1 + U C_{\boldsymbol{\xi}}) C_{\boldsymbol{\xi}} B_{\boldsymbol{Z}})$.

Next, we bound $\Gamma_u^{(D)} := \left\| \boldsymbol{q}_u^{(D)} - \widetilde{\boldsymbol{q}}_u^{(D)} \right\|_2$ from (C.3) as follows:

$$
\begin{aligned}
&\Gamma_u^{(D)} \\
&= \Bigg\| \boldsymbol{\xi} \left(\boldsymbol{Z}_1 \boldsymbol{q}_u^{(D-1)} + \sum_{(u,v)\in\mathcal{E}} \boldsymbol{\xi} \left(\boldsymbol{Z}_2 \boldsymbol{q}_v^{(D-1)} + \boldsymbol{Z}_3 \boldsymbol{e}_{u,v} \right) \right) \\
&- \boldsymbol{\xi} \left(\widetilde{\boldsymbol{Z}}_1 \widetilde{\boldsymbol{q}}_u^{(D-1)} + \sum_{(u,v)\in\mathcal{E}} \boldsymbol{\xi} \left(\widetilde{\boldsymbol{Z}}_2 \widetilde{\boldsymbol{q}}_v^{(D-1)} + \widetilde{\boldsymbol{Z}}_3 \boldsymbol{e}_{u,v} \right) \right) \Bigg\|_2
\end{aligned}
$$

$$
\begin{aligned}
&\leq C_{\boldsymbol{\xi}} \left\| \boldsymbol{Z}_1 \boldsymbol{q}_u^{(D-1)} - \widetilde{\boldsymbol{Z}}_1 \widetilde{\boldsymbol{q}}_u^{(D-1)} \right\|_2 \\
&+ U C_{\boldsymbol{\xi}}^2 \max_v \left(\left\| \boldsymbol{Z}_2 \boldsymbol{q}_v^{(D-1)} - \widetilde{\boldsymbol{Z}}_2 \widetilde{\boldsymbol{q}}_v^{(D-1)} \right\|_2 + \left\| \boldsymbol{Z}_3 - \widetilde{\boldsymbol{Z}}_3 \right\|_2 B_{\boldsymbol{x}} \right) \\
&\leq C_{\boldsymbol{\xi}} \left(\left\| \boldsymbol{q}_u^{(D-1)} \right\|_2 \left\| \boldsymbol{Z}_1 - \widetilde{\boldsymbol{Z}}_1 \right\|_2 + B_{\boldsymbol{Z}} \Gamma_u^{(D-1)} \right) \\
&+ U \boldsymbol{C}_{\boldsymbol{\xi}}^2 \max_v \Bigg(\left\| \boldsymbol{q}_v^{(D-1)} \right\|_2 \left\| \boldsymbol{Z}_2 - \widetilde{\boldsymbol{Z}}_2 \right\|_2 + B_{\boldsymbol{Z}} \Gamma_v^{(D-1)} \\
&+ B_{\boldsymbol{x}} \left\| \boldsymbol{Z}_3 - \widetilde{\boldsymbol{Z}}_3 \right\|_2 \Bigg).
\end{aligned}
$$

Solving the recursion in the last inequality, and using $\Gamma_u^{(0)} = 0, \forall u$, we get

$$
\begin{aligned}
\Gamma_u^{(D)} &\leq \widetilde{\Sigma}_{\boldsymbol{Z}_1} \left\| \boldsymbol{Z}_1 - \widetilde{\boldsymbol{Z}}_1 \right\|_2 + \widetilde{\Sigma}_{\boldsymbol{Z}_2} \left\| \boldsymbol{Z}_2 - \widetilde{\boldsymbol{Z}}_2 \right\|_2 \\
&\quad + \widetilde{\Sigma}_{\boldsymbol{Z}_3} \left\| \boldsymbol{Z}_3 - \widetilde{\boldsymbol{Z}}_3 \right\|_2, \\
\text{where } \widetilde{\Sigma}_{\boldsymbol{Z}_1} &= U C_{\boldsymbol{\xi}}^3 B_{\boldsymbol{Z}} B_{\boldsymbol{x}} \frac{\alpha^{(D+1)} - 2\alpha + 1}{(\alpha - 1)^2}, \\
\widetilde{\Sigma}_{\boldsymbol{Z}_2} &= U^2 C_{\boldsymbol{\xi}}^4 B_{\boldsymbol{Z}} B_{\boldsymbol{x}} \frac{\alpha^{(D+1)} - 2\alpha + 1}{(\alpha - 1)^2}, \\
\widetilde{\Sigma}_{\boldsymbol{Z}_3} &= U C_{\boldsymbol{\xi}}^2 B_{\boldsymbol{Z}} B_{\boldsymbol{x}} \frac{\alpha^D - 1}{\alpha - 1}.
\end{aligned}
$$

Using the above bound on $\Gamma_u^{(D)}$ in (C.3), we get

$$\begin{aligned}\left|\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}) - \boldsymbol{\pi}_{\widetilde{\boldsymbol{\theta}}}(\boldsymbol{\phi})\right| \leq \Sigma_{\boldsymbol{\beta}} \left\|\boldsymbol{\beta} - \widetilde{\boldsymbol{\beta}}\right\|_2 + \Sigma_{\boldsymbol{Z}_1} \left\|\boldsymbol{Z}_1 - \widetilde{\boldsymbol{Z}}_1\right\|_2 \\ + \Sigma_{\boldsymbol{Z}_2} \left\|\boldsymbol{Z}_2 - \widetilde{\boldsymbol{Z}}_2\right\|_2 + \Sigma_{\boldsymbol{Z}_3} \left\|\boldsymbol{Z}_3 - \widetilde{\boldsymbol{Z}}_3\right\|_2, \end{aligned} \tag{C.5}$$

where $\Sigma_{\boldsymbol{\beta}} = C_{\boldsymbol{\zeta}} B_{\boldsymbol{x}} \alpha^D + C_{\boldsymbol{\zeta}} U C_{\boldsymbol{\xi}}^2 B_{\boldsymbol{Z}} B_{\boldsymbol{x}} \frac{\alpha^D - 1}{\alpha - 1}$, $\Sigma_{\boldsymbol{Z}_1} = C_{\boldsymbol{\zeta}} B_{\boldsymbol{\beta}} \widetilde{\Sigma}_{\boldsymbol{Z}_1}$, $\Sigma_{\boldsymbol{Z}_2} = C_{\boldsymbol{\zeta}} B_{\boldsymbol{\beta}} \widetilde{\Sigma}_{\boldsymbol{Z}_2}$, and $\Sigma_{\boldsymbol{Z}_3} = C_{\boldsymbol{\zeta}} B_{\boldsymbol{\beta}} \widetilde{\Sigma}_{\boldsymbol{Z}_3}$.

Eq. (C.5) implies that for any $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, the existence of $\widetilde{\boldsymbol{\theta}}$ in the cover set such that $|\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}) - \boldsymbol{\pi}_{\widetilde{\boldsymbol{\theta}}}(\boldsymbol{\phi})| \leq \mu$ can be satisfied by ensuring the existence of $(\widetilde{\boldsymbol{\beta}}, \widetilde{\boldsymbol{Z}}_1, \widetilde{\boldsymbol{Z}}_2, \widetilde{\boldsymbol{Z}}_3)$ such that the right hand side of (C.5) $\leq \mu$. Hence, if we construct $\mu/4\Sigma_{\boldsymbol{\beta}}$-cover of $\mathcal{X}_{\boldsymbol{\beta}}$, and $\mu/4\Sigma_{\boldsymbol{Z}_i}$-cover of $\mathcal{X}_{\boldsymbol{Z}}$, $\forall i \in \{1, 2, 3\}$, the Cartesian product of the four sets correspond to a μ-cover of $\boldsymbol{\Pi}$. Hence, the covering number of $\boldsymbol{\Pi}$ at scale μ can be upper bounded by the product of the covering numbers of the four sets as follows:

$$\mathsf{N}\left(\boldsymbol{\Pi}, \mu\right) \leq \mathsf{N}\left(\mathcal{X}_{\boldsymbol{\beta}}, \frac{\mu}{4\Sigma_{\boldsymbol{\beta}}}\right) \times \prod_{i=1}^{3} \mathsf{N}\left(\mathcal{X}_{\boldsymbol{Z}}, \frac{\mu}{4\Sigma_{\boldsymbol{Z}_i}}\right). \tag{C.6}$$

In addition, the covering number for $\mathcal{X}_{\boldsymbol{Z}}$ and $\mathcal{X}_{\boldsymbol{\beta}}$ can be upper bounded using [12, Lemma 8] and [59], respectively, as follows:

$$\mathsf{N}(\mathcal{X}_{\boldsymbol{Z}}, \mu) \leq \left(1 + \frac{2\sqrt{E} B_{\boldsymbol{Z}}}{\mu}\right)^{E^2}, \mathsf{N}(\mathcal{X}_{\boldsymbol{\beta}}, \mu) \leq \left(\frac{3B_{\boldsymbol{\beta}}}{\mu}\right)^{E}$$

Using the above bounds in (C.6), we get

$$\begin{aligned}\mathsf{N}\left(\boldsymbol{\Pi}, \mu\right) &\leq \left(\frac{12 B_{\boldsymbol{\beta}} \Sigma_{\boldsymbol{\beta}}}{\mu}\right)^{E} \times \prod_{i=1}^{3} \left(1 + \frac{8\sqrt{E} B_{\boldsymbol{Z}} \Sigma_{\boldsymbol{Z}_i}}{\mu}\right)^{E^2} \\ &\leq \left(1 + \frac{12\sqrt{E} B_{\boldsymbol{Z}} \max\left\{\frac{B_{\boldsymbol{\beta}}}{B_{\boldsymbol{Z}}} \Sigma_{\boldsymbol{\beta}}, \Sigma_{\boldsymbol{Z}_1}, \Sigma_{\boldsymbol{Z}_2}, \Sigma_{\boldsymbol{Z}_3}\right\}}{\mu}\right)^{3E^2 + E}. \end{aligned}$$

Finally, we can use Lemma 6 to obtain a bound on $\widehat{\mathcal{R}}_{\mathcal{G}}(\boldsymbol{\Pi})$.

To this end, we upper bound the integral on the right hand side of (C.2) as follows:

$$\int_{a}^{\sqrt{K}} \sqrt{\log \mathsf{N}(\mathbf{\Pi}, \mu)} d\mu \;\leq\; \sqrt{K}\sqrt{\log \mathsf{N}(\mathbf{\Pi}, a)}.$$

The above inequality holds because $\sqrt{\log \mathsf{N}(\mathbf{\Pi}, \mu)}$ increases monotonically with the decrease of μ. Taking $a = 1/\sqrt{K}$, we get the following:

$$\widehat{\mathcal{R}}_{\mathcal{G}}(\mathbf{\Pi}) \leq \frac{4}{K} + \frac{12\sqrt{3E^2 + E}}{\sqrt{K}} \times \sqrt{\log\left(1 + 12\sqrt{EK} B_{\boldsymbol{Z}} \max\left\{\frac{B_{\boldsymbol{\beta}}}{B_{\boldsymbol{Z}}}\Sigma_{\boldsymbol{\beta}}, \Sigma_{\boldsymbol{Z}_1}, \Sigma_{\boldsymbol{Z}_2}, \Sigma_{\boldsymbol{Z}_3}\right\}\right)}.$$

Combining the above with $\widehat{\mathcal{R}}_{\mathcal{G}}(\mathcal{T}) \leq C_{\mathcal{L}}\widehat{\mathcal{R}}_{\mathcal{G}}(\mathbf{\Pi})$ and substituting in (C.1), we get the final result.

Appendix D: Proof of Theorem 2

Proof of Theorem 2 can be divided into two parts. In the first part we bound the expected loss under of the learned GNN. For this we will use the proof idea from [61]. However, the proof technique in [61] hinges on the convexity of their online learning problem. Hence, we make appropriate modifications to accommodate our non-convex GNN-based learning problem. In the second part, using the expected loss, we characterize the number of nodes needed to be visited by Algorithm 4.3 for solving a given problem instance optimally.

D.1 Expected Loss of Algorithm 4.3

Note that the online learning algorithm in Algorithm 4.3 is a no-regret algorithm. The definition of regret is as follows:

Definition 1 (Regret)**.** *Regret of an online algorithm that produces a sequence of policies* $\boldsymbol{\theta}_{1:I} = \{\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \ldots, \boldsymbol{\theta}^{(I)}\}$ *is denoted by* Reg_I*. It is the average loss of all policies with respect to the best policy in hindsight, i.e.,*

$$\mathrm{Reg}_I := \frac{1}{I}\sum_{i=1}^{I}\frac{1}{|\mathcal{D}_i|}\sum_{(\boldsymbol{\Phi}_s, y_s)\in\mathcal{D}_i}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}^{(i)}}(\boldsymbol{\phi}_s), y_s)] - \min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}}\frac{1}{I}\sum_{i=1}^{I}\frac{1}{|\mathcal{D}_i|}\sum_{(\boldsymbol{\Phi}_s, y_s)\in\mathcal{D}_i}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s), y_s)].$$

Definition 2 (No-regret Algorithm)**.** *A no-regret algorithm is an algorithm that produces a sequence of policies* $\boldsymbol{\theta}_{1:I}$ *such that the average regret goes to* 0 *as* N *goes to* ∞*:*

$$\mathrm{Reg}_I \leq \gamma_I \quad \textit{and} \quad \lim_{I\to\infty}\gamma_I \to 0.$$

For strongly convex $\mathcal{L}$, the work in [61] shows that Algorithm 4.3 is a no-regret algorithm with $\eta = \infty$, i.e., $\boldsymbol{\psi} = \mathbf{0}$ (recall that η is the parameter of the exponential distribution, i.e., $\boldsymbol{\psi} \sim \mathrm{Exp}(\eta)$, where $\mathrm{Exp}(\eta) := \eta(\exp(-\eta))$). However, for non-convex $\mathcal{L}$

we cannot guarantee that Algorithm 4.3 is a no-regret algorithm [1]. But with $0 < \eta < \infty$, under Assumption 2, Algorithm 4.3 was shown to be a no-regret algorithm [1].

Lemma 7. *[1, Theorem 1] When Assumption 2 holds, the regret after N iterations can be bounded by:*

$$\mathbb{E}_{\boldsymbol{\psi}\sim\mathrm{Exp}(\eta)}[\mathrm{Reg}_I] \leq \gamma_I \leq \mathcal{O}(1/I^{1/3}).$$

Finally, the following lemma establishes the expected loss of the policy returned by Algorithm 4.3.

Lemma 8. *For Algorithm 4.3, with probability at least $1-\delta$,*

$$\begin{aligned}
&\min_{\boldsymbol{\theta}\in\boldsymbol{\theta}_{1:I}} \mathbb{E}_{(\boldsymbol{\phi}_s,y_s)\sim p_{\boldsymbol{\theta}},\boldsymbol{\psi}}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s),y_s)] \\
&\leq \min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \frac{1}{I}\sum_{i=1}^{I}\frac{1}{J}\sum_{(\boldsymbol{\phi}_s,y_s)\in\mathcal{D}_i} \mathbb{E}_{\boldsymbol{\psi}}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s),y_s)] \\
&+ \gamma_I + \mathsf{Gap}\left(\frac{\delta}{2},J\right)\sqrt{\frac{2\log(\frac{2}{\delta})}{I}}. \qquad \text{(D.1)}
\end{aligned}$$

Proof. Define $\omega_i, \forall i \in [I]$ as:

$$\begin{aligned}
\omega_i :=& \mathbb{E}_{p_{\boldsymbol{\theta}^{(i)}},\boldsymbol{\psi}}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}^{(i)}}(\boldsymbol{\phi}_s),y_s)] \\
&- \frac{1}{J}\sum_{(\boldsymbol{\phi}_s,y_s)\in\mathcal{D}_i} \mathbb{E}_{\boldsymbol{\psi}}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}^{(i)}}(\boldsymbol{\phi}_s),y_s)].
\end{aligned}$$

Next, we use Lemma 4 to obtain a bound on $\omega_i, \forall i$; i.e., with probability at least $1-\delta/2$, the following holds simultaneously for $\omega_i, \forall i \in [I] : \omega_i \leq \mathsf{Gap}\left(\frac{\delta}{2},J\right)$. Consequently, $\Omega_i := \sum_{t=1}^{i}\omega_t, i = \{1,\ldots,I\}$ forms a martingale sequence, i.e., $\mathbb{E}[\Omega_i|\Omega_1,\ldots,\Omega_{i-1}] = \Omega_{i-1}$. Also, we have $|\Omega_{i+1}-\Omega_i| \leq \mathsf{Gap}(\delta/2,J), \forall i \in [I-1]$ with probability $1-\delta/2$. Next, consider the following lemma:

Lemma 9 (Azuma-Hoeffding's Inequality)**.** *Let $X_0,\ldots,X_I$ be a martingale sequence and $|X_i - X_{i-1}| \leq c_i$. Then with probability $1-\delta$,*

$$\Pr(X_I - X_0 \geq \epsilon) \leq \exp\left(\frac{-\epsilon^2}{2\sum_{i=1}^{I}c_i^2}\right).$$

Using Lemma 9, we have the following holds with probability of at least $(1-\delta/2)^2 \geq 1-\delta$,

$$\Omega_I \leq \mathsf{Gap}\left(\frac{\delta}{2}, J\right)\sqrt{2I\log(2/\delta)}. \tag{D.2}$$

Now, consider the following inequality:

$$\begin{aligned}
&\min_{\boldsymbol{\theta}\in\boldsymbol{\theta}_{1:I}} \mathbb{E}_{p_{\boldsymbol{\theta}},\boldsymbol{\psi}}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s), y_s)] \\
&\leq \frac{1}{I}\sum_{i=1}^{I} \mathbb{E}_{p_{\boldsymbol{\theta}_i}} \mathbb{E}_{\boldsymbol{\psi}}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}^{(i)}}(\boldsymbol{\phi}_s), y_s)] \\
&= \frac{1}{I}\sum_{i=1}^{I}\frac{1}{J}\sum_{(\boldsymbol{\phi}_s,y_s)\in\mathcal{D}_i} \mathbb{E}_{\boldsymbol{\psi}}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}^{(i)}}(\boldsymbol{\phi}_s), y_s)] + \frac{1}{I}\sum_{i=1}^{I}\omega_i.
\end{aligned}$$

Hence, with probability of at least $1-\delta$, we have

$$\begin{aligned}
&\min_{\boldsymbol{\theta}\in\boldsymbol{\theta}_{1:I}} \mathbb{E}_{p_{\boldsymbol{\theta}},\boldsymbol{\psi}}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s), y_s)] \\
&\overset{(a)}{\leq} \min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \frac{1}{I}\sum_{i=1}^{I}\frac{1}{J}\sum_{(\boldsymbol{\phi}_s,y_s)\in\mathcal{D}_i} \mathbb{E}_{\boldsymbol{\psi}}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s), y_s)] + \mathcal{O}(1/I^{1/3}) \\
&+ \mathsf{Gap}\left(\frac{\delta}{2}, J\right)\sqrt{\frac{2\log(2/\delta)}{I}} \\
&\overset{(b)}{\leq} \nu + \mathcal{O}(1/I^{1/3}) + \mathsf{Gap}\left(\frac{\delta}{2}, J\right)\sqrt{\frac{2\log(2/\delta)}{I}},
\end{aligned}$$

where (a) is by Lemma 7 and (D.2), and (b) is obtained via using Assumption 3. □

When the loss function $\mathcal{L}$ is selected to be binary cross-entropy loss, i.e.,

$$\mathcal{L}(x, y) = -y\log(x) - (1-y)\log(1-x),$$

$1 - e^{-\mathcal{L}(x,y)}$ corresponds to the classification error. Therefore, classification accuracy for

any $\boldsymbol{\theta}$, i.e., $\rho_{\boldsymbol{\theta}}$ is given by

$$\rho_{\boldsymbol{\theta}} = \mathbb{E}_{p_{\boldsymbol{\theta}},\boldsymbol{\psi}}[\exp(-\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s), y_s))].$$

Note that $\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}\in\boldsymbol{\theta}_{1:I}} \mathbb{E}_{p_{\boldsymbol{\theta}},\boldsymbol{\psi}}[\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s), y_s)]$. Next, we characterize $\rho_{\widehat{\boldsymbol{\theta}}}$. To that end, the following follows from Lemma 8.

$$\begin{aligned}
&\exp(\mathbb{E}_{p_{\boldsymbol{\theta}},\boldsymbol{\psi}}[-\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s), y_s)]) \\
&\geq \exp\left(-\nu - \mathcal{O}(1/I^{1/3}) - \mathsf{Gap}\left(\frac{\delta}{2}, J\right)\sqrt{\frac{2\log(2/\delta)}{I}}\right) \\
\implies \rho_{\widehat{\boldsymbol{\theta}}} &= \mathbb{E}_{p_{\boldsymbol{\theta}},\boldsymbol{\psi}}[\exp(-\mathcal{L}(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_s), y_s))] \\
&\overset{(b)}{\geq} \exp\left(-\nu - \mathcal{O}(1/I^{1/3}) - \mathsf{Gap}\left(\frac{\delta}{2}, J\right)\sqrt{\frac{2\log(2/\delta)}{I}}\right),
\end{aligned}$$

where (b) follows from Jensen's inequality.

D.2 B&B expected number of nodes and optimality

Let ϵ_{FP} denote the false positive error rate, i.e., the probability of classifying an irrelevant node as relevant. Also define ϵ_{FN} denote the false negative error rate, i.e., the probability of classifying a relevant node as irrelevant. Then the expected number of branches generated by using pruning policy on B&B was derived in [27]:

Lemma 10 ([27, Theorem 1])**.** *Assume that the node selection method in* (3.9) *ranks an irrelevant node higher than a relevant node with probability* ϵ_r*. Then the expected number of branches (number of non-leaf nodes) is*

$$\begin{aligned}
\frac{Q_{\widehat{\boldsymbol{\theta}}} - 1}{2} \leq \Bigg(\left(\frac{1-\epsilon_{\mathrm{FN}}}{1-2\epsilon_r\epsilon_{\mathrm{FP}}} + \frac{\epsilon_{\mathrm{FN}}}{1-2\epsilon_{\mathrm{FP}}}\right)\epsilon_r\epsilon_{\mathrm{FP}} \sum_{n=0}^{N}(1-\epsilon_{\mathrm{FN}})^n \\
+(1-\epsilon_{\mathrm{FN}})^{N+1}\frac{(1-\epsilon_r)\epsilon_{\mathrm{FP}}}{1-2\epsilon_{\mathrm{FP}}} + 1\Bigg)N,
\end{aligned}$$

Our node selection strategy is the lowest lower bound first as detailed in Section **??**. In the worst case scenario, $\epsilon_r = 1$. Therefore, using Lemma 10, the expected number of

branches is

$$\leq N\left(\frac{1-\rho_{\widehat{\boldsymbol{\theta}}}}{2\rho_{\widehat{\boldsymbol{\theta}}}-1}\sum_{n=0}^{N}\rho_{\widehat{\boldsymbol{\theta}}}^{n}+1\right)\overset{(c)}{=}N\left(\frac{1-\rho_{\widehat{\boldsymbol{\theta}}}^{N+1}}{2\rho_{\widehat{\boldsymbol{\theta}}}-1}+1\right)$$
$$=\frac{N(2\rho_{\widehat{\boldsymbol{\theta}}}-\rho_{\widehat{\boldsymbol{\theta}}}^{N})}{2\rho_{\widehat{\boldsymbol{\theta}}}-1}.$$

Since the expected number of branches correspond to the expected number of non-leaf nodes, the total number of nodes in the tree is $\leq \frac{2N(2\rho_{\widehat{\boldsymbol{\theta}}}-\rho_{\widehat{\boldsymbol{\theta}}}^{N})}{2\rho_{\widehat{\boldsymbol{\theta}}}-1}+1$. Next, we we characterize the probability that Algorithm 4.3 provides the optimal solution. To this end, observe that there is only one relevant node at any depth n of the B&B algorithm. The probability of not pruning a relevant node is $\geq \rho_{\widehat{\boldsymbol{\theta}}}$. Therefore, the probability of not pruning a relevant node at any depth of the branch and bound tree is $\geq \rho_{\widehat{\boldsymbol{\theta}}}^{N}$ (since N is the maximum depth of the tree). Hence, the probability of obtaining an optimal solution is at least $\rho_{\widehat{\boldsymbol{\theta}}}^{N}$.