ReDBeam: Real-time MU-MIMO Beamforming with Limited CSI Data Samples

Shaoran Li[†] Nan Jiang[‡] Chengzhang Li[§] Y. Thomas Hou[*] Wenjing Lou[*] Weijun Xie[‡]
[†]NVIDIA Corp, Santa Clara, CA [‡]Georgia Tech, Atlanta, GA
[§]The Ohio State University, Columbus, OH [*]Virginia Tech, Blacksburg, VA

***Abstract*—MU-MIMO beamforming is a key technology for 5G/NextG networks. In practice, MU-MIMO beamforming requires Channel State Information (CSI) and is prone to uncertainty. Furthermore, a beamforming solution must be derived within a millisecond (ms) to be useful for real-time (RT) 5G applications. We present ReDBeam—a RT data-driven beamforming solution for MU-MIMO using limited CSI data samples. The main contribution of ReDBeam is a parallel algorithm and an optimized GPU implementation. ReDBeam minimizes the base station (BS)'s power consumption while offering a probabilistic guarantee of users' data rates. It is purposefully designed to take advantage of the vast parallel processing capability in commercial off-the-shelf GPUs. Through extensive experiments, we show that ReDBeam can meet the 1 ms RT requirement and is orders of magnitude faster than other state-of-the-art algorithms for the same problem.**

I. Introduction

Beamforming is the key to increasing spectral efficiency in MU-MIMO [1], [2]. For beamforming, Channel State Information (CSI) is required to ensure transmit signals are precoded in the correct directions. There is a large body of works on MU-MIMO beamforming by assuming knowledge of perfect CSI (see, e.g., [3], [4]). Such an assumption is unrealistic, due to issues such as channel estimation errors [5], limited feedback [6], and hardware imbalance [7]. Therefore, a practical MU-MIMO beamforming solution must address the inherent *channel uncertainty* in CSI.

Existing works addressing CSI uncertainty can be classified into two branches: *model-based* and *data-driven.* Under the model-based approach, CSI is assumed to follow some known distributions [8], [9], channel statistics [10], [11], or worst-case boundaries [12], [13]. These works typically offer tractable mathematical formulations and subsequently can be leveraged to develop solutions with performance guarantees. However, the efficacy of this approach hinges upon the validity of the assumed models. In contrast, a data-driven approach (a.k.a. model-free) does not assume any models but rather uses CSI data samples directly to derive a beamforming solution. The prevailing examples of this approach are learning-based solutions (see, e.g., [14], [15]). Data-driven solutions are highly adaptive to a wide range of scenarios and can easily meet real-time (RT) requirements. However, their performance hinges upon a large high-quality dataset for training and lacks a theoretical performance guarantee.

Recently, a new approach called D^{2}BF was proposed in [16], which aimed to combine the strengths of both model-based and data-driven approaches and avoid their pitfalls. Instead of requiring a large dataset as in a data-driven approach, it only uses a small number of CSI data samples (which we call "small data"). Similar to a model-based approach, it is able to offer probabilistic performance guarantees to the UEs. The only concern of D^{2}BF is its high computational complexity, which poses a challenge to meet the RT requirement in 5G/NextG. By "RT", we mean am MU-MIMO beamforming solution must be derived within one Transmission Time Interval (TTI).

In this paper, we address the RT challenge associated with the new small-data approach in [16]. Assuming the most common 5G-NR numerology 0 [17], we aim to derive an MU-MIMO beamforming solution within 1 ms. The main contributions of this paper are summarized as the following:

- We address the key limitation in D^{2}BF [16] for MU-MIMO beamforming: How to make it work in RT? This "RT challenge" is especially important in 5G/NextG networks, where the available time for computation is only 1 ms.
- We propose an RT solution called ReDBeam, short for Real-time Data-Driven Beamforming, with two primary objectives: (i) deriving a beamforming solution within 1 ms, and (ii) minimizing BS power consumption while guaranteeing probabilistic data rates for the UEs. We structure ReDBeam as a parallel algorithm and make it suitable for parallel processing by Commercial Off-The-Shelf (COTS) GPUs.
- We implement ReDBeam on an NVIDIA V100 GPU and optimize our hardware implementation to minimize total time consumption. Specifically, we design three kernels and optimize each kernel by properly choosing the computation steps run in parallel for the GPU threads and using shared memory to reduce data access time.
- Through experiments, we show that ReDBeam can meet the 1 ms timing requirement and guarantee probabilistic SINR thresholds (equivalent to data rate requirements) for the UEs. In addition, we find that the performance of ReDBeam is very close to D^{2}BF and is significantly better than other model-based algorithms (i.e., Gaussian Approximation).

This research was supported in part by NSF under grants CNS-2312447 and CMMI-2246414, ONR under MURI grant N00014-19-1-2621, Commonwealth Cyber Initiative (CCI), and Virginia Tech Institute for Critical Technology and Applied Science (ICTAS).

ICC 2024 - IEEE International Conference on Communications | 978-1-7281-9054-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICC51166.2024.10622851

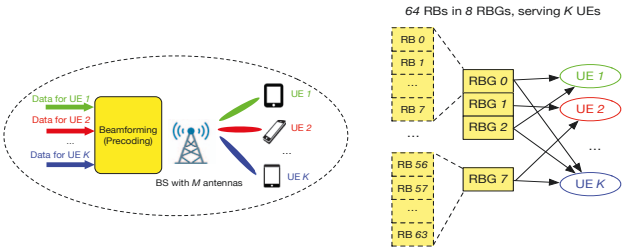


Fig. 1. Downlink MU-MIMO in a 5G cell (left) and grouping of RBs into RBGs for resource allocation (right).

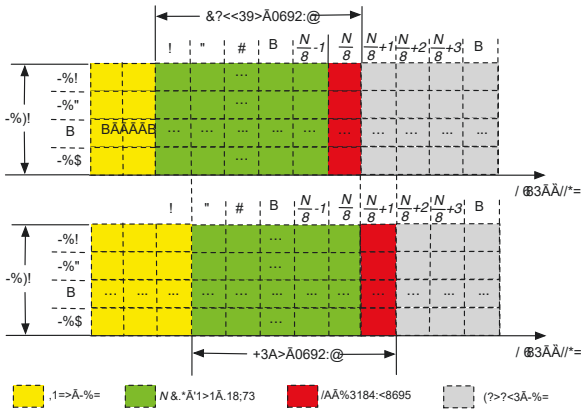


Fig. 2. Sliding window for CSI data samples.

II. System Model and Mathematical Formulation

A. MU-MIMO Beamforming

Fig. 1 (left) shows a 5G cell that employs the MU-MIMO scheme to serve a group of UEs. Denote M as the number of antennas of the BS and denote $\mathcal{K} = \{1, 2, 3, \cdots, K\}$ as the set of K UEs. We consider downlink and assume that each UE only has one antenna. As defined in 5G-NR [17], one Resource Block (RB) covers 12 sub-carriers in one TTI. To reduce the scheduling overhead, the BS can group multiple RBs into an RB Group (RBG) and use RBG as the granularity for scheduling. In Fig. 1, 64 RBs are grouped into 8 RBGs. Each RBG serves a subset (multiple) of UEs and a UE can be scheduled on multiple RBGs. Denote $\mathcal{G} = \{1, 2, \cdots, G\}$ as the set of G RBGs at the BS. For RBG $g \in \mathcal{G}$, denote $\mathcal{K}_g$ as the subset of UEs that are selected to receive data from RBG g. In this work, we assume $\mathcal{K}_g$'s are given *a priori*.

In the context of MU-MIMO, UEs in $\mathcal{K}_g$ simultaneously receive different data streams from the BS on the same RBG. To achieve this, we need to design a unique precoding vector for each UE on an RBG g. Denote $\mathbf{w}_{(g,i)}$ (an $M \times 1$ complex column vector) as the precoding vector for UE i on RBG g. These $\mathbf{w}_{(g,i)}$'s should be optimized using the CSI from the BS to the UEs.

B. CSI Data Samples from Sliding Window

Denote $\mathbf{h}_{(g,i)}$ (an $M \times 1$ complex column vector) as the CSI from the BS to UE i on RBG g. $\mathbf{h}_{(g,i)}$ can be estimated during a channel sounding process and can be performed on each RB [18]. Fig. 2 shows a sliding window mechanism on RBG 0, where each small rectangle represents an RB. Denote S as the number of RBs in an RBG and we have $S = 8$ in Fig. 2. Each window has $(N+S)$ RBs spanning over $(N/S+1)$ TTIs. We will use the N CSI data samples collected in the most recent N/S TTIs (green) to design precoding vectors for the S RBs in the upcoming TTI (red). The same mechanism applies to other RBGs.

Denote $\mathbb{P}_{\mathbf{h}_{(g,i)}}$ as the probability density function (PDF) of the unknown distribution of $\mathbf{h}_{(g,i)}$, i.e., $\mathbf{h}_{(g,i)} \sim \mathbb{P}_{\mathbf{h}_{(g,i)}}$. Then we have the N CSI data samples of $\mathbf{h}_{(g,i)}$ drawn from the unknown distribution $\mathbb{P}_{\mathbf{h}_{(g,i)}}$. We denote this sampling process from unknown channel distribution as:

$$N \text{ samples from unknown distribution } \mathbf{h}_{(g,i)} \sim \mathbb{P}_{\mathbf{h}_{(g,i)}}. \quad (1)$$

Under the sliding window mechanism shown in Fig. 2, we need to design an MU-MIMO beamforming solution within one TTI, which is 1 ms under 5G numerology 0.

C. Problem Formulation

We consider two requirements for the precoding vectors $\mathbf{w}_{(g,i)}$. The first is that the total transmission power over all RBGs (to all UEs) cannot exceed a power budget, i.e.,

$$\sum_{g \in \mathcal{G}} \sum_{i \in \mathcal{K}_g} ||\mathbf{w}_{(g,i)}||_2^2 \leq P^{\max}, \quad (2)$$

where $||\cdot||_2$ is the L_2-norm and $P^{\max}$ is the BS power budget.

The second is on UE's service requirement. We assume each UE has a data rate requirement to be met, which is equivalent to meeting an SINR threshold given the bandwidth of each RBG. Per 5G standards [19], a UE must use the same Modulation and Coding Scheme (MCS) on all its RBGs, which means SINR thresholds must be the same on all its RBGs. Denote γ_i^{req} as the SINR threshold for UE i. To cope channel uncertainty, we employ probabilistic guarantee for γ_i^{req} as:

$$\mathbb{P}\left\{ \frac{|(\mathbf{w}_{(g,i)})^H \mathbf{h}_{(g,i)}|^2}{\sum_{k \in \mathcal{K}_g}^{k \neq i} |(\mathbf{w}_{(g,k)})^H \mathbf{h}_{(g,i)}|^2 + \sigma_i^2} \geq \gamma_i^{\text{req}} \right\} \geq 1 - \epsilon_i \quad (i \in \mathcal{K}_g, g \in \mathcal{G}), \quad (3)$$

where $(\cdot)^H$ denotes conjugate transpose, σ_i^2 is the power of thermal noise at UE i, $\mathbb{P}\{\cdot\}$ denotes the probability function, and ϵ_i is called *risk level*. Constraints (3) state that the actual SINR on RBG g should be greater or equal to the required SINR threshold γ_i^{req} with a probability at least $1 - \epsilon_i$.

In this work, we are interested in minimizing the BS's power consumption while meeting the UEs' probabilistic data rate requirements. Our problem (P1) can be stated as follows:

$$\text{(P1)} \quad \min_{\mathbf{w}_{(g,i)} \in \mathbb{C}^{M \times 1}} \sum_{g \in \mathcal{G}} \sum_{i \in \mathcal{K}_g} ||\mathbf{w}_{(g,i)}||_2^2$$

$$\text{s.t.} \quad \text{BS power budget (2)},$$

$$\text{Probabilistic SINR guarantees (3)},$$

$$\text{CSI data samples with unknown distribution (1)},$$

where $\mathbb{C}^{M \times 1}$ is the set of all complex $M \times 1$ column vectors.

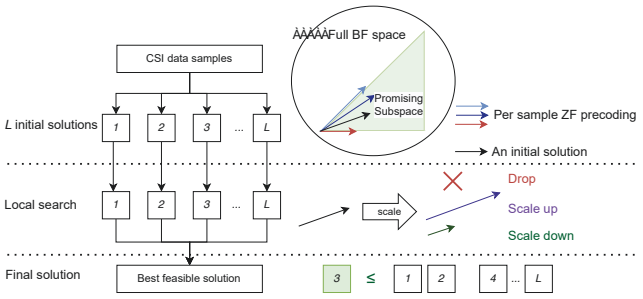


Fig. 3. An illustration of key steps of ReDBeam.

P1 is a *chance-constrained program*, which is hard to solve. In [16], the authors showed that P1 can be decomposed into G parallel and independent subproblems, where the g-th subproblem corresponds to MU-MIMO beamforming on RBG g. Then each subproblem can be equivalently reformulated into a *deterministic* problem based on the empirical distribution of $\mathbf{h}_{(g,i)}$ from its N data samples using ∞-Wasserstein ambiguity set [20]. After the G subproblems are solved, we can recover the solution to P1. This solution recovery process does not introduce relaxation and has negligible computation efforts.

The g-th ($g \in \mathcal{G}$) subproblem is given as:

$$\text{(P2)} \min_{\mathbf{w}_{(g,i)}} \sum_{\mathbf{w}_{(g,i)} \in \mathbb{C}^{M\times 1}} ||\mathbf{w}_{(g,i)}||_2^2$$
$$\text{s.t.} \frac{1}{N}\sum_{n=1}^{N} \mathbb{I}\Big\{\hat{f}\big(\mathbf{w}_{(g,i)}, \hat{\mathbf{h}}_{(g,i)}(n)\big) \geq \sigma_i^2\Big\} \geq 1-\epsilon_i \ (i \in \mathcal{K}_g),$$

where $\hat{f}\big(\mathbf{w}_{(g,i)}, \hat{\mathbf{h}}_{(g,i)}(n)\big)$ is defined as

$$\hat{f}\big(\mathbf{w}_{(g,i)}, \hat{\mathbf{h}}_{(g,i)}(n)\big) = \Bigg\{ \min_{\mathbf{c}_i \in \mathbb{C}^{M\times 1}} \Big(\frac{|(\mathbf{w}_{(g,i)})^H \mathbf{c}_i|^2}{\gamma_i^{\text{req}}} - \sum_{k\in\mathcal{K}_g}^{k\neq i} |(\mathbf{w}_{(g,k)})^H \mathbf{c}_i|^2 \Big) \ : \ ||\mathbf{c}_i - \hat{\mathbf{h}}_{(g,i)}(n)||_2 \leq \theta_{(g,i)} \Bigg\} \tag{4}$$

In P2, $\hat{\mathbf{h}}_{(g,i)}(n)$ is the n-th data sample of $\mathbf{h}_{(g,i)}$, $\mathbb{I}(\cdot)$ is the indicator function, and $\theta_{(g,i)}$ is a small constant that represents the search space (distance) of $\mathbf{c}_i$ around each CSI data sample [16], [20], [21]. We will show a simple approach to set $\theta_{(g,i)}$ based on the N CSI data samples in Section V.

Technical Challenges Though P2 is a deterministic problem, it is mathematically complex. In particular, its constraints include a non-convex Quadratically Constrained Quadratic Program (QCQP) defined in $\hat{f}(\mathbf{w}_{(g,i)}, \hat{\mathbf{h}}_{(g,i)}(n))$. Iterative algorithms for MU-MIMO (e.g., [3], [16]) require substantial computation time and cannot meet our RT requirement (1 ms). In this work, we will develop a 1 ms RT solution that effectively explores the search space $\mathbb{C}^{M\times 1}$ for $\mathbf{w}_{(g,i)}$.

III. Real-time Data-Driven Beamforming

In this section, we present ReDBeam—a Real-time Data-Driven Beamforming solution and its GPU implementation. Fig. 3 shows the three key steps of ReDBeam. In the first step, we identify a "promising" search space and generate a sufficiently large number (denoted as L) of initial solutions. In the second step, we employ a scaling-based local search to find feasible solutions based on these L initial solutions while trying to improve the objective if possible. In the final step, we find the feasible solution with the minimum objective value as our solution to P2.

A. Generating A Population of Initial Solutions

In this step, we generate L initial solutions within a promising search space. The "promising search space" is a subspace formed by some basis vectors and it should contain many feasible beamforming vectors with satisfactory performance. It is possible that the optimal beamforming solution may fall outside of this search space. But as long as we can find a good solution (i.e., close to the optimal) within this space, we have achieved our goals.

A Promising Search Space For ease of exposition, we drop the subscript g when there is no confusion. To narrow down the original search space $\mathbb{C}^{M\times 1}$, we observe that a promising direction for $\mathbf{w}_i \in \mathbb{C}^{M\times 1}$ should enhance the received power and suppress the received interference for UE i. Based on this observation, we identify a promising search space to be a cone whose basis vectors are derived from the widely used Zero-Forcing (ZF) precoding based on CSI data samples.

Given that we have N CSI data samples in the current window, it is natural to use N basis vectors to form the cone where each basis vector corresponds to a specific CSI data sample. Denote $\mathbf{v}_i(n)$, $n = 1, 2, \cdots, N$ s the N basis vectors. Each $\mathbf{v}_i(n)$ is the ZF precoding vector for UE i under the n-th CSI data sample. Clearly, $\mathbf{v}_i(n)$ depends on the CSI data samples $\hat{\mathbf{h}}_i(n), i \in \mathcal{K}_g$. Define an $M \times |\mathcal{K}_g|$ matrix $\hat{\mathbf{H}}(n)$ as:

$$\hat{\mathbf{H}}(n) = \Big[\hat{\mathbf{h}}_1(n) \quad \hat{\mathbf{h}}_2(n) \quad \cdots \quad \hat{\mathbf{h}}_{|\mathcal{K}_g|}(n)\Big].$$

We can calculate the ZF precoding vectors $\mathbf{v}_i(n)$'s based on $\hat{\mathbf{H}}(n)$ following the deterministic CSI model. This means we need to calculate the Moore-Penrose pseudo-inverse of $\hat{\mathbf{H}}(n)$, denoted as $\hat{\mathbf{H}}(n)^\dagger$ (a $|\mathcal{K}_g| \times M$ complex matrix). Here we calculate $\hat{\mathbf{H}}(n)^\dagger$ using QR decomposition and forward/backward substitutions since it has many parallelizable steps.

Denote $\mathbf{u}_i(n)$, an $M \times 1$ complex column vector, as the complex conjugate of the i-th row of $\hat{\mathbf{H}}(n)^\dagger$, i.e.,

$$\hat{\mathbf{H}}(n)^\dagger = \big[\mathbf{u}_1(n) \quad \mathbf{u}_2(n) \quad \cdots \quad \mathbf{u}_{|\mathcal{K}_g|}(n)\big]^H .$$

Then $\mathbf{v}_i(n)$ is given as:

$$\mathbf{v}_i(n) = \sigma_i \sqrt{\gamma_i^{\text{req}}} \cdot \mathbf{u}_i(n) \quad (i \in \mathcal{K}_g, \ n = 1, 2, \cdots, N), \tag{5}$$

which means that $\mathbf{v}_i(n)$ follows the same direction as $\mathbf{u}_i(n)$. Due to the zero interference property of ZF precoding, and the fact that $\big(\mathbf{u}_i(n)\big)^H \hat{\mathbf{h}}_i(n) = 1$, the received SINR at UE i is $|\big(\mathbf{v}_i(n)\big)^H \hat{\mathbf{h}}_i(n)|^2/\sigma_i^2 = \gamma_i^{\text{req}}$. Since there are N $\hat{\mathbf{H}}(n)$'s, we can calculate the ZF precoding vectors for each $\hat{\mathbf{H}}(n)$ in parallel for $n = 1, 2, \cdots, N$. After calculating $\mathbf{v}_i(n)$ in (5), we obtain the promising search space for $\mathbf{w}_i$, given as:

$$\mathbf{w}_i \in \Big\{\mathbf{e}\colon \mathbf{e} = \sum_{n=1}^{N} \alpha_i(n)\mathbf{v}_i(n), \ \alpha_i(n) \geq 0\Big\} \ (i \in \mathcal{K}_g), \tag{6}$$

where each vector inside this cone is a linear combination of the N basis vectors $\mathbf{v}_i(n)$'s with $\alpha_i(n) \geq 0$, $i \in \mathcal{K}_g$.

Sampling The L initial solutions are randomly sampled inside this promising cone. Denote the ℓ-th initial solution as $\mathbf{z}_i^\ell$ where $\ell = 1, 2, 3, \cdots, L$. To generate $\mathbf{z}_i^\ell$, we choose $\alpha_i^\ell(n)$, $n = 1, 2, \cdots, N$ in (6) following a uniform distribution between $[0, 1]$. Then we scale the $\alpha_i^\ell(n)$'s proportionally so that their sum is normalized to 1, i.e., $\sum_{n=1}^{N} \alpha_i^\ell(n) = 1$. This gives each initial solution as $\mathbf{z}_i^\ell = \sum_{n=1}^{N} \alpha_i^\ell(n)\mathbf{v}_i(n)$. Clearly, finding the L initial solutions can be done in parallel since they are independent of each other.

B. Finding Promising Solutions via Local Search

Now we have L initial solutions $\mathbf{z}_i^\ell$, $\ell = 1, 2, \cdots, L$, $i \in \mathcal{K}_g$. However, since they are randomly generated from the "promising space", they neither guarantee feasibility (meet the UEs' data rate requirements) nor good performance (minimize the BS's transmission power over all RBGs). Therefore, we will perform a local search on each of these L initial solutions so that i) each new solution is feasible (if possible), and ii) the objective of P2 is minimized.

Main Idea With 1 ms time constraint, a local search must be simple and fast, with as few steps as possible. Thus, we limit our local search only to the scaling of the length (or norm) of $\mathbf{z}_i^\ell$, i.e., without creating new directions. Denote $\mathbf{w}_i^\ell$ as the solution after scaling of $\mathbf{z}_i^\ell$, which is given as

$$\mathbf{w}_i^\ell = \lambda^\ell \cdot \mathbf{z}_i^\ell \quad (i \in \mathcal{K}_g)\ . \tag{7}$$

Here $\lambda^\ell > 0$ is the scaling factor and is independent of UEs.

With the scaling in (7), the objective function of P2 becomes $\sum_{i\in\mathcal{K}_g} ||\lambda^\ell \cdot \mathbf{z}_i^\ell||_2^2$, which is $(\lambda^\ell)^2 \cdot \sum_{i\in\mathcal{K}_g} ||\mathbf{z}_i^\ell||_2^2$. Since $\sum_{i\in\mathcal{K}_g} ||\mathbf{z}_i^\ell||_2^2$ is a constant when $\mathbf{z}_i^\ell$'s are given, the objective of P2 can be replaced by $\min\ \lambda^\ell$. Further, based on the definition of $\hat{f}(\mathbf{w}_i^\ell, \hat{\mathbf{h}}_i(n))$ in P2, we have

$$\hat{f}(\mathbf{w}_i^\ell, \hat{\mathbf{h}}_i(n)) = \hat{f}(\lambda^\ell \mathbf{z}_i^\ell, \hat{\mathbf{h}}_i(n)) = (\lambda^\ell)^2 \hat{f}\big(\mathbf{z}_i^\ell, \hat{\mathbf{h}}_i(n)\big)\ . \tag{8}$$

Therefore, with given $\mathbf{z}_i^\ell$'s, we can rewrite P2 as follows:

$$\begin{aligned}
\text{(P3)} \min_{\lambda^\ell}\ & \lambda^\ell \\
\text{s.t.} \frac{1}{N}\sum_{n=1}^{N} \mathbb{I}\Big\{(\lambda^\ell)^2 \hat{f}\big(\mathbf{z}_i^\ell, \hat{\mathbf{h}}_i(n)\big) \geq \sigma_i^2\Big\} \geq 1 - \epsilon_i\ (i \in \mathcal{K}_g), \\
& \text{Definition of } \hat{f}\big(\mathbf{z}_i^\ell, \hat{\mathbf{h}}_i(n)\big) \text{ in (4)},\ \lambda^\ell > 0\ .
\end{aligned}$$

Thus, we substitute the complicated multi-dimensional local search for $\mathbf{z}_i^\ell$'s with finding λ^ℓ. Clearly, the main difficulty of P3 is $\hat{f}\big(\mathbf{z}_i^\ell, \hat{\mathbf{h}}_i(n)\big)$.

Calculation of $\hat{f}\big(\mathbf{z}_i^\ell, \hat{\mathbf{h}}_i(n)\big)$ In P3, $\hat{f}\big(\mathbf{z}_i^\ell, \hat{\mathbf{h}}_i(n)\big)$ contains $N \cdot |\mathcal{K}_g|$ terms. Since these terms are independent of each other, we can solve them in parallel. Based on (4), for a specific $\hat{f}\big(\mathbf{z}_i^\ell, \hat{\mathbf{h}}_i(n)\big)$, we need to solve

$$\begin{aligned}
\text{(P4)}\ \min_{\mathbf{c}_i}\quad & \Big(\frac{|(\mathbf{z}_i^\ell)^H \mathbf{c}_i|^2}{\gamma_i^{\text{req}}} - \sum_{k\in\mathcal{K}_g}^{k\neq i} |(\mathbf{z}_k^\ell)^H \mathbf{c}_i|^2\Big) \\
\text{s.t.}\quad & ||\mathbf{c}_i - \hat{\mathbf{h}}_i(n)||_2 \leq \theta_i\ .
\end{aligned}$$

Unfortunately, P4 is a non-convex QCQP, which is hard to solve. Due to our strict 1 ms time constraint, we will find a lower bound for the optimal objective of P4 and use it for $\hat{f}\big(\mathbf{z}_i^\ell, \hat{\mathbf{h}}_i(n)\big)$ in P3. This may lead to a slightly larger value for λ^ℓ in the objective of P3. Nevertheless, in our final solution to P1, all constraints (probabilistic guarantee of UEs' data rates and BS's power budget) remain satisfied, except the objective (BS's power consumption) may be slightly higher than that of the optimal solution.

Note that in the objective of P4, the first term is related to the received signal, while the other $(|\mathcal{K}_g|-1)$ terms are related to interference. To obtain a lower bound for P4, we relax its objective function by separating the $|\mathcal{K}_g|$ terms, i.e.,

$$\min_{\mathbf{c}_i}\ \frac{|(\mathbf{z}_i^\ell)^H \mathbf{c}_i|^2}{\gamma_i^{\text{req}}} - \sum_{k\in\mathcal{K}_g}^{k\neq i} \max_{\mathbf{c}_i} |(\mathbf{z}_k^\ell)^H \mathbf{c}_i|^2. \tag{9}$$

Then we can decompose P4 into $\mathcal{K}_g$ subproblems where each subproblem corresponds to an item in (9):

$$\begin{aligned}
\text{(P4-A)}\quad \min_{\mathbf{c}_i}\quad & \frac{|(\mathbf{z}_i^\ell)^H \mathbf{c}_i|^2}{\gamma_i^{\text{req}}} \\
\text{s.t.}\quad & ||\mathbf{c}_i - \hat{\mathbf{h}}_i(n)||_2 \leq \theta_i\ ,
\end{aligned}$$

and for $k \in \mathcal{K}_g,\ k \neq i$,

$$\begin{aligned}
\text{(P4-B)}\quad \max_{\mathbf{c}_i}\quad & |(\mathbf{z}_k^\ell)^H \mathbf{c}_i|^2 \\
\text{s.t.}\quad & ||\mathbf{c}_i - \hat{\mathbf{h}}_i(n)||_2 \leq \theta_i\ .
\end{aligned}$$

We will have one instance of (P4-A) for the received signal, and $|\mathcal{K}_g|-1$ instances of (P4-B). Both (P4-A) and (P4-B) have only one decision variable $\mathbf{c}_i$ in their objective function and constraint, which promises closed-form solutions. To conserve space, we omit the derivations and directly provide the lower bound for P4's objective function (i.e., $\hat{f}\big(\mathbf{z}_i^\ell, \hat{\mathbf{h}}_i(n)\big)$) as:

$$\begin{aligned}
\hat{f}_i^{\text{LB}}(n) = & \frac{1}{\gamma_i^{\text{req}}} \Big(\max\big\{0, \big(|(\mathbf{z}_i^\ell)^H \hat{\mathbf{h}}_i(n)| - \theta_i \cdot ||\mathbf{z}_i^\ell||_2\big)^2\big\}\Big) \\
& - \sum_{k\in\mathcal{K}_g}^{k\neq i} \big(|(\mathbf{z}_k^\ell)^H \hat{\mathbf{h}}_i(n)| + \theta_i \cdot ||\mathbf{z}_k^\ell||_2\big)^2\ .
\end{aligned} \tag{10}$$

Solution to P3 Substituting $\hat{f}\big(\mathbf{z}_i^\ell, \hat{\mathbf{h}}_i(n)\big)$ with its lower bound $\hat{f}_i^{\text{LB}}(n)$, we can rewrite the constraints in P3 as

$$\sum_{n=1}^{N} \mathbb{I}\Big\{(\lambda^\ell)^2 \cdot \hat{f}_i^{\text{LB}}(n) \geq \sigma_i^2\Big\} \geq N(1-\epsilon_i) \quad (i \in \mathcal{K}_g). \tag{11}$$

There are $|\mathcal{K}_g|$ constraints in (11). Denote β_i^ℓ as the minimum λ^ℓ to satisfy the i-th constraint of (11). β_i^ℓ can be easily found by sorting N real numbers in non-decreasing order and set β_i^ℓ as the $(1-\epsilon_i)$-quantile. There are two cases:

i) If for all $i \in \mathcal{K}_g$, $\hat{f}_i^{\text{LB}}(n) > 0$ holds for at least $\lceil N \cdot (1-\epsilon_i)\rceil$ CSI data samples, then $\beta_i^\ell > 0$. To satisfy all constraints in (11), we simply set λ^ℓ to:

$$\lambda^\ell = \max_{i\in\mathcal{K}_g}\ \beta_i^\ell\ . \tag{12}$$

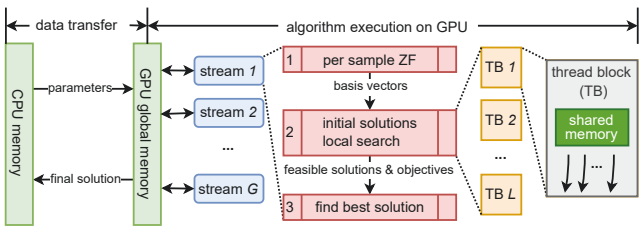


Fig. 4. A GPU implementation of ReDBeam.

Using λ^ℓ from (12) to (7), we have a feasible solution:

$$\mathbf{w}_i^\ell = \left(\max_{i\in\mathcal{K}_g} \beta_i^\ell\right) \cdot \sum_{n=1}^{N} \alpha_i^\ell(n)\mathbf{v}_i(n) \quad (i \in \mathcal{K}_g). \tag{13}$$

ii) Otherwise, i.e., for some $i \in \mathcal{K}_g$, $\hat{f}_i^{\mathrm{LB}}(n) > 0$ only holds for fewer than $\lceil N \cdot (1-\epsilon_i)\rceil$ CSI data samples, then there is no feasible solution to (11) under this initial solution $\mathbf{z}_i^\ell$. We can simply drop this initial solution $\mathbf{z}_i^\ell$.

C. Finding the Final Solution

After the local search, we have at most L feasible solutions, we can calculate their objectives (i.e., $\sum_{i\in\mathcal{K}_g} ||\mathbf{w}_i^\ell||_2^2$) and find the one with the smallest objective (since P2 is a minimization problem). The feasible solution $\mathbf{w}_i^\ell$ associated with this smallest objective value is our final solution to P2 on RBG g.

IV. GPU Implementation

Now we have a solution to P2. But we still need to address how to make the solution to meet the 1 ms time constraint in the real world. To do this, we choose COTS GPU as our implementation platform, due to its massive parallel computing capability. On a GPU, we need to address how to efficiently allocate the available resources, such as *threads*[1] for computation and *memory* for data storage.

Fig. 4 shows our GPU implementation of ReDBeam. The total time consumption consists of data transfer and algorithm execution. We use *asynchronous* data transfer between CPU and GPU to reduce these data transfer times. As shown in Fig. 4, there are G streams (blue) and each stream calculates the beamforming solution on a specific RBG (one P2 instance). We overlap data transfers and kernel executions across different streams to reduce overall time consumption [22]. For each stream, we design three kernels (red):

- Kernel 1 calculates the ZF precoding vectors $\mathbf{v}_i(n)$ based on CSI data samples $\hat{\mathbf{h}}_i(n)$, $i \in \mathcal{K}_g$, $n = 1, 2, \cdots, N$.
- Kernel 2 generates initial solutions $\mathbf{z}_i^\ell$ based on $\mathbf{v}_i(n)$, finds the scaling factors λ^ℓ (if exists), applies λ^ℓ to obtain $\mathbf{w}_i^\ell = \lambda^\ell \cdot \mathbf{z}_i^\ell$, and calculates its objective.
- Kernel 3 finds the best solution from the feasible solutions provided by kernel 2, which is our solution to a P2.

Due to space limitations, we will focus our discussion on kernel 2 to demonstrate how to optimize the GPU implementation of ReDBeam.

[1]A *thread* is the minimum processing unit for algorithm execution and threads are grouped into *thread blocks (TBs)* to execute *kernels*.

Kernel 2 For kernel 2, we use L TBs, and each TB has $N|\mathcal{K}_g|$ threads to maximize the parallelization capability of our GPU. Specifically, each TB generates an initial solution $\mathbf{z}_i^\ell$ from (6), finds the scaling factor λ^ℓ, obtains $\mathbf{w}_i^\ell = \lambda^\ell \cdot \mathbf{z}_i^\ell$ by (7), and calculates the objective value.

The core step of generating an initial solution is to calculate $|\mathcal{K}_g|M$ sums of N complex numbers with randomly generated $\alpha_i(n)$. Note that the promising search space (6) will be implicitly included during this process. A common technique to reduce execution time in GPU implementation is parallel reduction, which is suitable for comparison or summation over a large number of terms [23]. For a sum of N numbers, we need $\log_2(N)$ iterations and $N/2$ threads. Since timing is our main concern, we employ the parallel reduction technique to calculate an initial solution $\mathbf{z}_i^\ell$, $i \in \mathcal{K}_g$.

For the lower bounds $\hat{f}_i^{\mathrm{LB}}(n)$ derived in (10), we need to calculate $N|\mathcal{K}_g|^2$ times of multiplication of two $M \times 1$ complex vectors in the form of $(\mathbf{z}_k^\ell)^H \cdot \hat{\mathbf{h}}_i(n)$. So each thread will compute one such term and $N|\mathcal{K}_g|$ terms can be calculated in parallel. Then we can easily calculate $\hat{f}_i^{\mathrm{LB}}(n)$ based on (10).

To find λ^ℓ, we can directly sort N numbers and then check the sorted numbers to see whether we employ (13) to obtain $\mathbf{w}_i^\ell$ or drop this initial solution $\mathbf{z}_i^\ell$. Specifically, we need to perform $|\mathcal{K}_g|$ times of sorting of N real numbers. We employ a parallel sorting algorithm called odd-even sorting, which uses $\lfloor N/2 \rfloor$ threads and N iterations to sort N numbers. Then λ^ℓ can be easily found by comparing $|\mathcal{K}_g|$ real numbers or we declare it does not exist and the initial solution is infeasible.

If λ^ℓ is found, we only need to multiply a real number λ^ℓ to $|\mathcal{K}_g|$ $M \times 1$ complex vectors $\mathbf{z}_i^\ell$, $i \in \mathcal{K}_g$. To reduce computation time, we use $2M|\mathcal{K}_g|$ threads, where the first $|\mathcal{K}_g|M$ threads are for the real part of $\mathbf{w}_i^\ell$, $i \in \mathcal{K}_g$ and the remaining $|\mathcal{K}_g|M$ threads are for the imaginary part of $\mathbf{w}_i^\ell$, $i \in \mathcal{K}_g$. Then we can calculate the objective of this feasible solution using parallel reduction.

In terms of *memory*, we use both *global memory* and *shared memory*. Global memory has a large volume (e.g., $\sim$ 10 gigabytes) and can exchange data with external platforms (i.e., CPU or other GPU) while shared memory has faster access but a smaller volume (e.g., 48 kilobytes per SM in our NVIDIA V100 GPU) and no external access outside of a thread block. Thus, shared memory is more suitable for repeatedly accessed data within a TB. In kernel 2, all intermediate results are stored in the shared memory, which includes $\mathbf{z}_i^\ell$, $\hat{f}_i^{\mathrm{LB}}(n)$, β_i^ℓ, and λ^ℓ. The output feasible solutions $\mathbf{w}_i^\ell$ and their objectives are stored in global memory for kernel 3.

V. Experimental Results

In this section, we evaluate the running time and performance of ReDBeam. We implement ReDBeam using CUDA 11.2 on an NVIDIA Tesla V100.

A. Parameter Settings

Fig. 5 shows the topology of a 5G cell with a 500-meter radius and 30 UEs (i.e., $K = 30$). We assume the BS has 8 antennas (i.e., $M = 8$). Following Fig. 1, the BS has $G = 8$

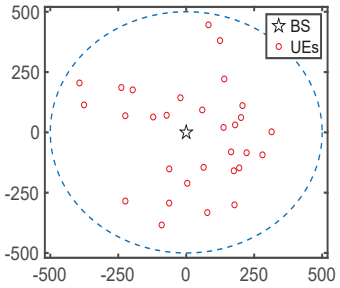
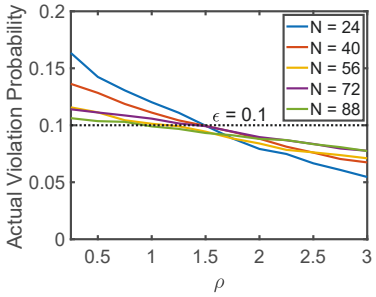
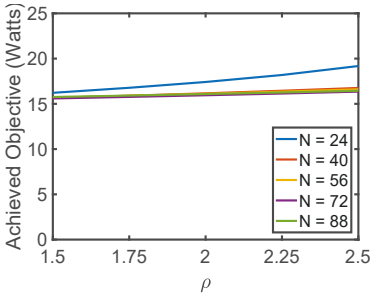


Fig. 5. A BS with 30 UEs.



(a) Violation probability



(b) Objective value

Fig. 6. Impact of ρ and N.

RBGs and each RBG consists of 8 RBs. We set the number of UEs per RBG $|\mathcal{K}_g| = 2$. The BS has a power budget $P^{\max} = 46$ dBm and the thermal noise σ_i^2 is set to -150 dBm/Hz for all UEs. We set the SINR threshold $\gamma_i^{\text{req}} = 2^{(500/d_i)} - 1$ where d_i is the distance between UE i and the BS in meters. Further, we found it is sufficient to set $L = 650$ for our setting.

For the wireless channel, we consider the path-loss model and fast fading. The path-loss between UE i and the BS is modeled by $PL_i = 38 + 30 \times \log_{10}(d_i)$ (in dB) [24]. For fast fading, we employ Rician fading with a 10 dB Rician factor, which is a common model for correlated RBs. We also employ a truncated Gaussian distribution to simulate the CSI estimation errors [9]. Note that ReDBeam only relies on the CSI data samples and does not assume any channel models.

B. Choosing N and $\theta_{(g,i)}$

We first discuss how to choose N and $\theta_{(g,i)}$. Intuitively, we would like to choose a small N to reduce complexity and a small $\theta_{(g,i)}$ that meets the probabilistic data rate guarantees with lower BS power consumption. For a given N, $\theta_{(g,i)}$ is related to how uncertain the N CSI data samples are. We propose a fast heuristic based on these samples to calculate $\theta_{(g,i)}$ before executing ReDBeam. Specifically, we resort to a constant factor and the estimated standard deviation from N CSI data samples, i.e., for any $i \in \mathcal{K}_g,\ g \in \mathcal{G}$,

$$\theta_{(g,i)} = \frac{\rho}{N} \cdot \sqrt{\frac{1}{N-1} \sum_{n=1}^{N} \left(||\hat{\mathbf{h}}_{(g,i)}(n) - \frac{\sum_{n=1}^{N} \hat{\mathbf{h}}_{(g,i)}(n)}{N}||_2^2 \right)}.$$

ρ is the constant factor we need to choose. We use ρ/N in the above expression because the more CSI data samples we have, the closer the empirical distribution is to the true distribution.

To find a proper ρ and N, we set $\epsilon_i = 0.1$ and run ReDBeam under $0.25 \le \rho \le 3$ and $24 \le N \le 88$. The actual violation probabilities and achieved objectives are shown in Fig. 6. As shown in Fig. 6(a), it is sufficient to choose $\rho \in [1.5, 2.5]$ in all cases of N. Then we zoom into these ρ values and study the achieved objectives as shown in Fig. 6(b). As shown in Fig. 6(b), the objective only increases slightly w.r.t. ρ. Taking both Fig. 6(a) and Fig. 6(b) into account, we conclude that it is prudent to choose $N = 40$ and $\rho = 2$ to calculate $\theta_{(g,i)}$.

The above process shows how to set N and $\theta_{(g,i)}$ for a given network setting. They can be dynamically adjusted during runtime through continuous tracking of the violation probabilities at the UEs. As for time consumption, in each window, before executing ReDBeam in Fig. 4, we will use $G|\mathcal{K}_g|$ thread blocks to calculate these $\theta_{(g,i)}$'s in parallel and then use these $\theta_{(g,i)}$'s in ReDBeam to derive a beamforming solution. Note that ρ's have no impact on the overall time consumption while a larger N will slightly increase time consumption.

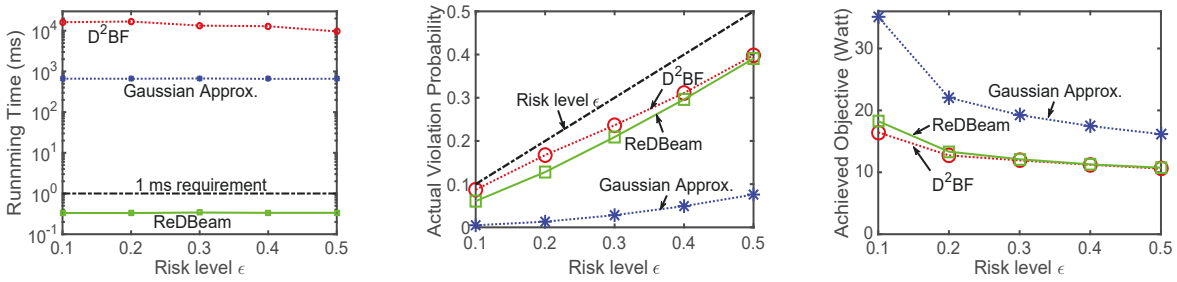
C. Results

We evaluate ReDBeam with two benchmarks. The first one is D^{2}BF [16], which solves each P2 through convex approximation and Semidefinite Programming (SDP). The second one is Gaussian Approximation [8], which assumes CSI follows a complex Gaussian distribution. We implement them with MOSEK 9.2.38 using MATLAB R2017b on Intel Xeon E5-2687w v4. All results are the average of 50 runs.

Fig. 7 shows ReDBeam's performance w.r.t. ϵ, including running time (a), threshold violation probabilities (b), and the achieved objective (c). As shown in Fig. 7(a), ReDBeam meets the 1 ms timing requirement under all risk level ϵ's. Further, it is rather independent of ϵ because the running time depends on the number of steps for each thread. On the other hand, none of the other two solutions (D^{2}BF and Gaussian Approximation) can meet the 1 ms timing requirement. Specifically, Gaussian Approximation requires $\sim 10^2$ ms while D^{2}BF requires $\sim 10^4$ ms. One may argue that employing a C API optimizer may yield a reduced execution time compared to the results of using MATLAB API (as shown in Fig. 7(a)). But this change still cannot offer a reduction required to meet the 1 ms requirement for D^{2}BF and Gaussian Approximation.

As shown in Fig. 7(b), ReDBeam can meet the target risk level ϵ. Further, Fig. 7(c) shows that the objective value achieved by ReDBeam is very close to that of D^{2}BF. This demonstrates the superb performance of ReDBeam. Gaussian Approximation offers the worst performance (as it uses the most transmission power), which is consistent with its conservativeness demonstrated in Fig. 7(b). In general, the closer the actual violation probabilities to the risk level ϵ (in Fig. 7(b)), the less power is needed (in Fig. 7(c)).

We also conducted experiments with varying M and $|\mathcal{K}_g|$ and found that our ReDBeam can meet the 1 ms real-time requirement for a network with up to $M = 18$, $|\mathcal{K}_g| = 4$, and $G = 12$ (i.e., serving 48 UEs simultaneously), which



(a) Running time (b) Violation probability (c) Objective value

Fig. 7. Performance of ReDBeam w.r.t ϵ.

is sufficient for real-world scenarios. All observations are consistent with the above discussions.

VI. Conclusions

We presented ReDBeam—a real-time MU-MIMO beamforming solution that offers performance guarantees (in terms of UEs' probabilistic data rate requirements) and minimizes BS power consumption with limited CSI data samples. The key idea is to employ GPU's massive parallel computing capability to solve the beamforming problem on each RBG in parallel and combine them as the final solution. For each RBG, ReDBeam generates initial solutions from a promising subspace, employs local search to ensure feasibility and improve objective, and finds the best feasible solution. Further, we optimized GPU implementation for ReDBeam on thread allocations and memory management. Experiment results showed that ReDBeam can deliver an MU-MIMO beamforming solution within 1 ms while meeting the UEs' probabilistic data rate requirements and minimizing the BS's power consumption.

References

[1] J. Laneman, D. Tse, and G. Wornell, "Cooperative diversity in wireless networks: Efficient protocols and outage behavior," *IEEE Trans. Information Theory*, vol. 50, no. 12, pp. 3062–3080, Dec. 2004.

[2] M. Shafi, A. F. Molisch, P. J. Smith, T. Haustein, P. Zhu, P. De Silva, F. Tufvesson, A. Benjebbour, and G. Wunder, "5G: A tutorial overview of standards, trials, challenges, deployment, and practice," *IEEE J. Selected Areas in Commun.*, vol. 35, no. 6, pp. 1201–1221, June 2017.

[3] M. Schubert and H. Boche, "Solution of the multiuser downlink beamforming problem with individual SINR constraints," *IEEE Trans. Vehicular Technology*, vol. 53, no. 1, pp. 18–28, Jan. 2004.

[4] K. Zheng, L. Zhao, J. Mei, B. Shao, W. Xiang, and L. Hanzo, "Survey of large-scale MIMO systems," *IEEE Commun. Surveys & Tutorials*, vol. 17, no. 3, pp. 1738–1760, Third Quarter 2015.

[5] Y. Wu, R. H. Louie, and M. R. McKay, "Analysis and design of wireless ad hoc networks with channel estimation errors," *IEEE Trans. Signal Processing*, vol. 61, no. 6, pp. 1447–1459, Mar. 2013.

[6] X. Rao and V. K. Lau, "Distributed compressive CSIT estimation and feedback for FDD multi-user massive MIMO systems," *IEEE Trans. Signal Processing*, vol. 62, no. 12, pp. 3261–3271, June 2014.

[7] X. Jiang and F. Kaltenberger, "Channel reciprocity calibration in TDD hybrid beamforming massive MIMO systems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 3, pp. 422–431, June 2018.

[8] K.-Y. Wang, A. M.-C. So, T.-H. Chang, W.-K. Ma, and C.-Y. Chi, "Outage constrained robust transmit optimization for multiuser MISO downlinks: Tractable approximations by conic optimization," *IEEE Trans. Signal Processing*, vol. 62, no. 21, pp. 5690–5705, Nov. 2014.

[9] D. Mi, M. Dianati, L. Zhang, S. Muhaidat, and R. Tafazolli, "Massive MIMO performance with imperfect channel reciprocity and channel estimation error," *IEEE Trans. Communications*, vol. 65, no. 9, pp. 3734–3749, Sept. 2017.

[10] H. Liu, X. Yuan, and Y. J. Zhang, "Statistical beamforming for FDD downlink massive MIMO via spatial information extraction and beam selection," *IEEE Trans. Wireless Communications*, vol. 19, no. 7, pp. 4617–4631, July 2020.

[11] B. K. Chalise, S. Shahbazpanahi, A. Czylwik, and A. B. Gershman, "Robust downlink beamforming based on outage probability specifications," *IEEE Trans. Wireless Communications*, vol. 6, no. 10, pp. 3498–3503, Oct. 2007.

[12] E. Song, Q. Shi, M. Sanjabi, R.-Y. Sun, and Z.-Q. Luo, "Robust SINR-constrained MISO downlink beamforming: When is semidefinite programming relaxation tight?" *EURASIP J. Wireless Commun. and Networking*, vol. 2012, no. 1, pp. 1–11, Aug. 2012.

[13] M. B. Shenouda and T. N. Davidson, "Nonlinear and linear broadcasting with QoS requirements: Tractable approaches for bounded channel uncertainties," *IEEE Trans. Signal Processing*, vol. 57, no. 5, pp. 1936–1947, May 2009.

[14] J. Zhang, M. You, G. Zheng, I. Krikidis, and L. Zhao, "Model-driven learning for generic MIMO downlink beamforming with uplink channel information," *IEEE Trans. Wireless Communications*, vol. 21, no. 4, pp. 2368–2382, Apr. 2022.

[15] H. Zhu, Q. Wu, X.-J. Wu, Q. Fan, P. Fan, and J. Wang, "Decentralized power allocation for MIMO-NOMA vehicular edge computing based on deep reinforcement learning," *IEEE IoT Journal*, vol. 9, no. 14, pp. 12 770–12 782, June 2022.

[16] S. Li, N. Jiang, Y. Chen, Y. T. Hou, W. Lou, and W. Xie, "D^{2}BF—Data-driven beamforming in MU-MIMO with channel estimation uncertainty," in *Proc. IEEE INFOCOM*, pp. 120–129, Virtual Conference, May 2–5, 2022.

[17] 3GPP, *TS 38.211: 5G; NR; Physical channels and modulation*, Jan. 2023, version 17.4.0. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3213 (Last Accessed: Feb. 2024).

[18] I. Ahmed, H. Khammari, A. Shahid, A. Musa, K. S. Kim, E. De Poorter, and I. Moerman, "A survey on hybrid beamforming techniques in 5G: Architecture and system model perspectives," *IEEE Commun. Surveys & Tutorials*, vol. 20, no. 4, pp. 3060–3097, Fourth Quarter 2018.

[19] 3GPP, *TS 38.214: 5G; NR; Physical layer procedures for data*, Jan. 2023, version 17.4.0. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3216 (Last Accessed: Feb. 2024).

[20] W. Xie, "On distributionally robust chance constrained programs with Wasserstein distance," *Mathematical Programming*, vol. 186, no. 1, pp. 115–155, Mar. 2021.

[21] N. Jiang and W. Xie, "ALSO-X and ALSO-X+: Better convex approximations for chance constrained programs," *Operations Research*, vol. 70, no. 6, pp. 3581–3600, Feb. 2022.

[22] M. Harris. (2012, Dec.) How to overlap data transfers in CUDA C/C++. Available: https://devblogs.nvidia.com/how-overlap-data-transfers-cuda-cc/ (Last Accessed: Feb. 2024).

[23] ——. (2007) Optimizing parallel reduction in CUDA. Available: https://developer.download.nvidia.com/assets/cuda/files/reduction.pdf (Last Accessed: Feb. 2024).

[24] 3GPP, *TR 36.931: Radio Frequency (RF) requirements for LTE Pico Node B*, Apr. 2022, version 17.0.0. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2589 (Last Accessed: Feb. 2024).