Decentralized Coordinate-Descent Data Detection and Precoding for Massive MU-MIMO

Kaipeng Li[1], Oscar Castañeda[2], Charles Jeon[3], Joseph R. Cavallaro[1], and Christoph Studer[2]
[1]Department of Electrical and Computer Engineering, Rice University, Houston, TX
[2]School of Electrical and Computer Engineering, Cornell University, Ithaca, NY
[3]Intel Labs, Hillsboro, OR

***Abstract*—Massive multiuser (MU) multiple-input multiple-output (MIMO) promises significant improvements in spectral efficiency compared to small-scale MIMO. Typical massive MU-MIMO base-station (BS) designs rely on centralized linear data detectors and precoders which entail excessively high complexity, interconnect data rates, and chip input/output (I/O) bandwidth when executed on a single computing fabric. To resolve these complexity and bandwidth bottlenecks, we propose new decentralized algorithms for data detection and precoding that use coordinate descent. Our methods parallelize computations across multiple computing fabrics, while minimizing interconnect and I/O bandwidth. The proposed decentralized algorithms achieve near-optimal error-rate performance and multi-Gbps throughput at sub-1 ms latency when implemented on a multi-GPU cluster with half-precision floating-point arithmetic.**

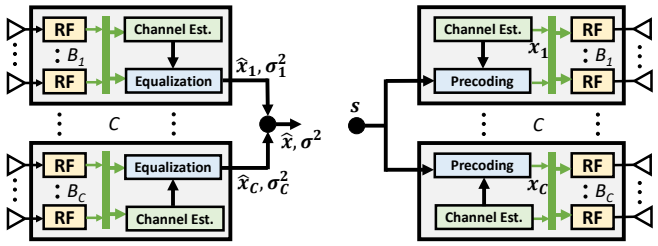
I. Introduction

Massive multi-user (MU) multiple-input multiple-output (MIMO) will be a key technology of next-generation wireless systems [1], [2]. Equipped with hundreds of antennas, a massive MU-MIMO base-station (BS) simultaneously serves tens of user equipments (UEs) in the same time-frequency resource; this yields significant spectral efficiency and power efficiency improvements compared to that of conventional, small-scale MIMO systems. As discussed in [3], centralized baseband processing in massive MU-MIMO results in excessively high complexity as well as interconnect and chip input/output (I/O) bandwidth between the baseband processors and the antenna array. For example, the raw baseband data rates for a massive MU-MIMO BS operating at 80 MHz bandwidth with 256 active antenna elements and 12-bit digital-to-analog converters (DACs) approach 1 Tbps, exceeding the capabilities of existing interconnect standards, such as the Common Public Radio Interface (CPRI) [4]. Furthermore, a single centralized computing fabric, such as a field programmable gate array (FPGA) or a graphics processing unit (GPU), has limited chip I/O bandwidth, computing, and storage resources to realize real-time baseband processing at such high rates.

A. Decentralized Baseband Processing

To resolve the interconnect and I/O bandwidth bottlenecks and reduce complexity and memory requirements per computing fabric, references [3], [5]–[7] have proposed *decentralized baseband processing* (DBP) for massive MU-MIMO systems. The key idea behind DBP is to divide the antenna array into separate antenna clusters, each associated with dedicated RF circuitry and baseband processors. At each cluster, local baseband processing, such as (de-)modulation, channel estimation, data detection, and precoding is performed. To achieve near-optimal spectral efficiency, algorithms that rely on consensus-sharing among antenna clusters [8], [9] or one-shot feedforward architectures [10]–[12] have been developed. Existing algorithms for DBP, however, require costly matrix-matrix multiplications and matrix inversion operations [11], [12] at each cluster, which causes high complexity in scenarios with short coherence times that require channel matrix preprocessing at high rates.

The work was supported in part by Xilinx, Inc. and by the US NSF under grants ECCS-1408370, CNS-1717218, CNS-1827940, ECCS-1408006, CCF-1535897, CCF-1652065, and CNS-1717559. We acknowledge the hardware support of the DGX-1 multi-GPU systems at the Nvidia Data Center.



(a) Uplink equalization (b) Downlink precoding

Fig. 1. Fully-decentralized feedforward architectures for massive MU-MIMO.

B. Contributions

We propose new coordinate descent (CD)-based data detection and precoding algorithms that leverage fully-decentralized *feedforward* architectures as illustrated in Fig 1. In the uplink (UEs transmit to BS), we perform linear minimum mean-square error (L-MMSE) equalization using CD at each cluster, and fuse the local estimates from all clusters to form a global detection result. In the downlink (BS transmits to UEs), we perform zero-forcing (ZF) precoding using CD at each cluster and allocate the power per local clusters under a global power constraint. Our decentralized CD-based data detector and precoder avoid computation of the Gram matrix and matrix inversion, which yields significant complexity savings while maintaining near-optimal error-rate performance. To showcase the efficacy of our methods in practice, we implement our algorithms on a multi-GPU system using both single-precision (32-bit) and half-precision (16-bit) floating-point formats. Our implementations outperform existing centralized and decentralized methods, and show that multi-Gbps throughput at sub-1ms latency can be achieved for realistic massive MU-MIMO systems.

II. System Model, Architecture, and Algorithms

A. Uplink System Model and Architecture

In the uplink, U single-antenna UEs transmit the vector $\mathbf{x}^{\text{ul}} \in \mathcal{O}^U$ ($\mathcal{O}$ is the constellation set) to the BS with B antennas, where $B \geq U$. The received signal $\mathbf{y}^{\text{ul}} \in \mathbb{C}^B$ at the BS can be modeled using the well-known baseband input-output relation $\mathbf{y}^{\text{ul}} = \mathbf{H}^{\text{ul}}\mathbf{x}^{\text{ul}} + \mathbf{n}^{\text{ul}}$, where $\mathbf{H}^{\text{ul}} \in \mathbb{C}^{B \times U}$ is the MIMO channel matrix, and $\mathbf{n}^{\text{ul}} \in \mathbb{C}^B$ models noise as i.i.d. complex circularly-symmetric Gaussian random vector with variance N_0 per entry.

Algorithm 1 Decentralized L-MMSE Detection using CD

Input: $\mathbf{H}_c$, $\mathbf{y}_c$, $c = 1, \ldots, C$ and N_0, E_x
Preprocessing (decentralized):
$m_{u,c} = (\|\mathbf{h}_{u,c}\|_2^2 + \frac{N_0}{E_x})^{-1}, u = 1, \ldots, U, c = 1, \ldots, C$
$n_{u,c} = m_{u,c}\|\mathbf{h}_{u,c}\|_2^2, u = 1, \ldots, U, c = 1, \ldots, C$
CD iterations (decentralized):
Init: $\mathbf{r}_c = \mathbf{y}_c, \mathbf{x}_c^{(0)} = \mathbf{0}$
for $t = 1, \ldots, T_{\text{max}}$ **do**
  for $u = 1, \ldots, U$ **do**
    $x_{u,c}^{(t)} = m_{u,c}\mathbf{h}_{u,c}^H \mathbf{r}_c + n_{u,c}x_{u,c}^{(t-1)}$
    $\delta x_{u,c}^{(t)} = x_{u,c}^{(t)} - x_{u,c}^{(t-1)}$
    $\mathbf{r}_c = \mathbf{r}_c - \mathbf{h}_{u,c}\delta x_{u,c}^{(t)}$
  end for
end for
Data fusion and averaging (centralized):
Output: $\hat{\mathbf{x}} = \Sigma_{c=1}^{C} \lambda_c \hat{\mathbf{x}}_c^{(T_{\text{max}})}$

Recently, fully-decentralized uplink architectures have been proposed in [11], [12]. As shown in Fig. 1(a), the B BS-antennas are partitioned into C antenna clusters, where the c^{th} cluster consists of B_c antennas with $B = \sum_{c=1}^{C} B_c$. Each cluster receives its own signals $\mathbf{y}_c^{\text{ul}} \in \mathbb{C}^{B_c}$ and the corresponding input-output relation for each cluster is

$$\mathbf{y}_c^{\text{ul}} = \mathbf{H}_c^{\text{ul}}\mathbf{x}^{\text{ul}} + \mathbf{n}_c^{\text{ul}}, \quad c = 1, 2, \ldots, C.$$

Here, $\mathbf{H}_c^{\text{ul}} \in \mathbb{C}^{B_c \times U}$ (a sub-matrix of $\mathbf{H}^{\text{ul}}$) is the local channel matrix and $\mathbf{n}_c^{ul} \in \mathbb{C}^{B_c}$ is the local noise vector at cluster c.

For the fully-decentralized feedforward uplink architecture shown in Fig. 1(a), each BS cluster estimates and stores its own matrix $\mathbf{H}_c^{\text{ul}}$, and performs local data detection based only on its local receive vector $\mathbf{y}_c^{\text{ul}}$ and channel matrix $\mathbf{H}_c^{\text{ul}}$ to form a local estimate. All clusters then send their local estimates $\hat{\mathbf{x}}_c^{\text{ul}}$ in a feedforward manner to a centralized processor that forms the global estimate $\hat{\mathbf{x}}^{\text{ul}}$. See Sec. III for more details.

B. Downlink System Model and Architecture

In the downlink, the BS precodes the vector $\mathbf{s} \in \mathcal{O}^U$ to form a beamforming vector $\mathbf{x}^{\text{dl}} \in \mathbb{C}^B$ that is transmitted to U UEs. The UE vector $\mathbf{y}^{\text{dl}} \in \mathbb{C}^U$ that contains the receive signal for each UE is given by $\mathbf{y}^{\text{dl}} = \mathbf{H}^{\text{dl}}\mathbf{x}^{\text{dl}} + \mathbf{n}^{\text{dl}}$, where $\mathbf{H}^{\text{dl}} \in \mathbb{C}^{U \times B}$ and $\mathbf{n}^{\text{dl}} \in \mathbb{C}^U$ are the downlink channel and noise, respectively.

In the fully decentralized downlink architecture put forward in [10] and shown in Fig. 1(b), there are B_c BS antennas at each of the C clusters. The local input-output relation is

$$\mathbf{y}_c^{\text{dl}} = \mathbf{H}_c^{\text{dl}}\mathbf{x}_c^{\text{dl}} + \mathbf{n}_c^{\text{dl}}, \quad c = 1, 2, \ldots, C.$$

Here, $\mathbf{x}_c^{\text{dl}} \in \mathbb{C}^{B_c}$ is the local beamforming vector and the local downlink channel can be obtained via reciprocity in the uplink, i.e., $\mathbf{H}_c^{\text{dl}} = {\mathbf{H}_c^{\text{ul}}}^H$. In Sec. IV, we show a new CD-based precoding algorithm for the fully-decentralized architecture to compute $\mathbf{x}_c^{\text{dl}}$ in a feedforward manner.

III. Decentralized Uplink Data Detection

In the uplink, the BS estimates the transmit signal $\hat{\mathbf{x}}^{\text{ul}}$ based on $\mathbf{y}^{\text{ul}}$ and $\mathbf{H}^{\text{ul}}$. In what follows, we omit the superscript $^{\text{ul}}$ and we focus on the L-MMSE data detector by solving the following convex optimization problem:

$$\hat{\mathbf{x}} = \underset{\mathbf{x} \in \mathbb{C}^U}{\arg\min} \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \tfrac{N_0}{E_x}\|\mathbf{x}\|_2^2. \tag{1}$$

Here, E_x represents the per-user transmit energy. To solve this problem efficiently, reference [13] proposed to use coordinate descent (CD), which avoids computation of the Gram matrix and matrix inversion, and effectively recovers $\hat{\mathbf{x}}$ with only a few iterations for massive MU-MIMO systems. We propose to decentralize this CD-based detector by leveraging the fully-decentralized feedforward architecture in Fig. 1(a). At each antenna cluster, we calculate a local estimate $\hat{\mathbf{x}}_c$ given $\mathbf{y}_c$ and $\mathbf{H}_c$ using CD as in [13]. We then fuse the local estimates $\hat{\mathbf{x}}_c$ to form a global estimate $\hat{\mathbf{x}}$ via a weighted sum: $\hat{\mathbf{x}} = \Sigma_{c=1}^{C} \lambda_c \hat{\mathbf{x}}_c$, where $\lambda_c = \frac{1}{\sigma_c^2}(\Sigma_{c'=1}^{C} \frac{1}{\sigma_{c'}^2})^{-1}$ is the optimal fusion rule proposed in [11]; σ_c^2 is the post-equalization noise variance at cluster c. The decentralized CD data detector is summarized in Algorithm 1. As seen from Algorithm 1, the CD based detector does not require computationally-intensive Gram matrix computations and matrix inversions; instead, the algorithm mostly relies on vector operations, which significantly reduces the complexity of L-MMSE data detection.

Algorithm 2 Decentralized ZF Precoding using CD

Input: $\mathbf{H}_c$, $\mathbf{s}$, ρ_c, $c = 1, \ldots, C$
Data broadcasting (centralized node → all clusters): $\mathbf{s}$
Preprocessing (decentralized):
$p_{u,c} = \|\mathbf{h}_{u,c}\|_2^{-1}, u = 1, \ldots, U, c = 1, \ldots, C$
$\bar{\mathbf{h}}_{u,c} = p_{u,c}\mathbf{h}_{u,c}, u = 1, \ldots, U, c = 1, \ldots, C$
$\bar{s}_u = p_{u,c}s_u, u = 1, \ldots, U, c = 1, \ldots, C$
CD iterations (decentralized):
Init: $\mathbf{x}_c = \mathbf{0}$
for $t = 1, \ldots, T_{\text{max}}$ **do**
  for $u = 1, \ldots, U$ **do**
    $\mathbf{x}_c = \mathbf{x}_c - (\bar{\mathbf{h}}_{u,c}\mathbf{x}_c - \bar{s}_u)\bar{\mathbf{h}}_{u,c}^H$
  end for
end for
$\mathbf{x}_c = \rho_c \mathbf{x}_c / \|\mathbf{x}_c\|_2$ ***(decentralized)***
Output: $\hat{\mathbf{x}} = [\mathbf{x}_1; \ldots; \mathbf{x}_C]$

IV. Decentralized Downlink Precoding

A. Coordinate Descent (CD)-based Precoding

In the downlink, the BS precodes the transmit signal $\mathbf{s}$ using the matrix $\mathbf{H}^{\text{dl}}$ to generate a beamforming vector $\mathbf{x}^{\text{dl}}$. In what follows, we omit the superscript $^{\text{dl}}$ and focus on ZF precoding which minimizes MU interference. ZF precoding can be formulated as the following convex optimization problem:

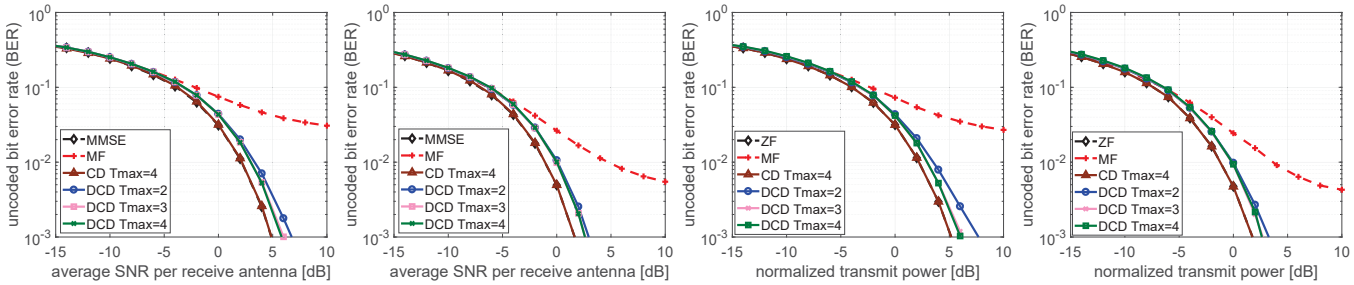
$$\hat{\mathbf{x}} = \underset{\mathbf{x} \in \mathbb{C}^B}{\arg\min} \tfrac{1}{2}\|\mathbf{x}\|_2^2 \quad \text{subject to } \mathbf{s} = \mathbf{H}\mathbf{x}, \tag{2}$$

which has a closed form solution $\hat{\mathbf{x}} = \mathbf{H}^H(\mathbf{H}\mathbf{H}^H)^{-1}\mathbf{s}$. We next derive a CD-based ZF precoder that avoids the need of matrix multiplications and inversions. To simplify the derivation, we normalize $\mathbf{H}$ so that the u^{th} row of $\mathbf{H}$ is $\bar{\mathbf{h}}_u^H = \mathbf{h}_u^H / \|\mathbf{h}_u\|_2$. We also scale $\mathbf{s}$ accordingly so that the u^{th} entry of $\bar{\mathbf{s}}$ is $\bar{s}_u = s_u / \|\mathbf{h}_u\|_2$. We note that the scaled beamforming constraint $\bar{\mathbf{s}} = \bar{\mathbf{H}}\mathbf{x}$ can be used in (2) without altering the solution. We first reformulate (2) to its Lagrangian dual

$$\hat{\mathbf{z}} = \underset{\mathbf{z} \in \mathbb{C}^U}{\arg\min} f(\mathbf{z}) := \tfrac{1}{2}\left\|\bar{\mathbf{H}}^H\mathbf{z}\right\|_2^2 - \bar{\mathbf{s}}^H\mathbf{z} \tag{3}$$

for which the primal solution $\hat{\mathbf{x}}$ is given by $\hat{\mathbf{x}} = \bar{\mathbf{H}}^H\hat{\mathbf{z}}$. We solve (3) using CD by updating $\mathbf{z}$ iteratively. Specifically, we update $\mathbf{z}$ across each of its U coordinates in the opposite direction to the gradient component of that coordinate, respectively, in round-robin fashion. For coordinate u, we update $\mathbf{z}$ as

$$\mathbf{z} \leftarrow \mathbf{z} - \nabla f(\mathbf{z})_u \mathbf{e}_u = \mathbf{z} - (\bar{\mathbf{h}}_u^H \bar{\mathbf{H}}^H \mathbf{z} - \bar{s}_u)\mathbf{e}_u, \tag{4}$$



(a) $C = 4$, $B = 128$, uplink (b) $C = 8$, $B = 256$, uplink (c) $C = 4$, $B = 128$, downlink (d) $C = 8$, $B = 256$, downlink

Fig. 2. Uncoded bit error rate (BER) performance for $U = 8$ users and $B_c = 32$ antennas per cluster. Figs. 2(a) and 2(b): Uplink data detection performance comparison between decentralized CD, MF, centralized CD and L-MMSE data detectors. Figs. 2(c) and 2(d): Downlink precoding performance comparison between decentralized CD, MF, centralized CD and ZF precoders. We scale the total BS antenna number B by increasing the cluster size C.

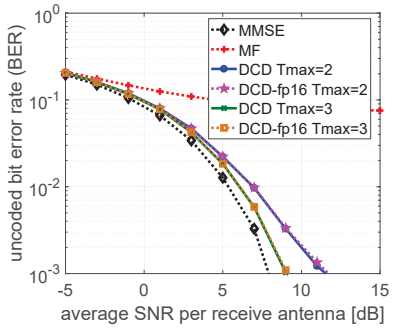


Fig. 3. BER of 16-bit vs. double precision floating-point uplink detectors.

where $\mathbf{e}_u$ is the u^{th} unit vector. Given $\hat{\mathbf{x}} = \bar{\mathbf{H}}^H \hat{\mathbf{z}}$, we have:

$$\mathbf{x} \leftarrow \mathbf{x} - (\bar{\mathbf{h}}_u^H \bar{\mathbf{H}}^H \mathbf{z} - \bar{s}_u)\bar{\mathbf{h}}_u = \mathbf{x} - (\bar{\mathbf{h}}_u^H \mathbf{x} - \bar{s}_u)\bar{\mathbf{h}}_u, \quad (5)$$

which is the update formula for the u^{th} coordinate. We iteratively update $\mathbf{x}$ with (5) across all U coordinates in a cyclic fashion until convergence. Once $\mathbf{x}$ is computed, we transmit $\mathbf{x} \leftarrow \rho\mathbf{x}/\|\mathbf{x}\|_2$ to satisfy the transmit power constraint ρ^2.

B. Decentralized CD-based Precoding

The proposed decentralized CD precoder is suitable for the fully-decentralized feedforward architecture depicted in Fig. 1(b). Given a transmit power constraint ρ^2, we first broadcast the transmit signal $\mathbf{s}$ to each cluster, and solve the local ZF precoding problem using CD as described above. We then scale the beamforming vector $\mathbf{x}_c$ at each cluster to the local power constraint $\rho_c^2 = \rho^2/C$ so that $\mathbf{x}_c \leftarrow \rho_c \mathbf{x}_c/\|\mathbf{x}_c\|_2$. The decentralized CD precoder is summarized in Algorithm 2. As for data detection, Gram matrix computation and matrix inversion are avoided, which reduces complexity.

V. Simulation Results

We now show uncoded bit error-rate (BER) results of the proposed decentralized CD-based data detection and precoding algorithms for a Rayleigh fading massive MU-MIMO system with 16-QAM. We fix the number of local BS antennas $B_c = 32$, and the numbers of UEs $U = 8$. Figs. 2(a) and 2(b) show the results of data detection for a total number of BS antennas $B = \{128, 256\}$ divided into $C = \{4, 8\}$ clusters, respectively; Figs. 2(c) and 2(d) show the results for precoding. We see that for data detection and precoding, the proposed decentralized CD-based methods effectively approach the performance of centralized methods with $T_{\text{max}} = 3$ or 4 iterations and with negligible performance loss. For example, our methods suffer only a 1 dB SNR loss at 10^{-3} BER, while outperforming the fully-decentralized matched filter (MF) by a significant margin.

To illustrate the effect of arithmetic precision on the BER, we show results using a half-precision floating-point format (fp16). Fig. 3 reveals that the decentralized CD detector for $B_c = 32$, $U = 8$, $C = 2$, and $B = 64$, with fp16 has virtually no BER performance loss when compared to the double precision. This indicates that we can implement the proposed decentralized algorithms in hardware with low precision to reduce transfer bandwidth and complexity, without sacrificing the BER. The throughput results shown in Sec. VI-C support this claim.

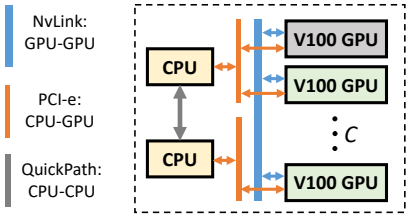


Fig. 4. Overview of the multi-GPU system architecture. The top GPU, colored in gray, is the master GPU for data broadcasting or fusion.

VI. Multi-GPU Implementations

A. System Architecture

We now present a multi-GPU implementation of the proposed decentralized CD data detector and precoder. We implemented our designs on an Nvidia DGX-1 [15] multi-GPU system whose architecture is shown in Fig. 4. The system has two 20-core Intel Xeon E5-2698 v4 CPUs and eight Tesla V100 Volta GPUs with NvLink [16] connection, a direct GPU-to-GPU communication link with 300 GBps bi-directional bandwidth. Each V100 GPU consists of 5120 CUDA cores and 16 GB high bandwidth memory (HBM). The designs are implemented with CUDA v9.2 [17] for GPU acceleration of detection and precoding computations, and the NVIDIA Collective Communications Library (NCCL) v2.2 [18], which builds on the message passing interface (MPI) library for efficient data transfer among GPUs over NvLink. We implemented our designs with floating-point data types that are natively supported by the used GPUs.

B. Implementation Details

We invoke C MPI processes from the CPU controller, each process supervising the local CD detection or precoding computations on a GPU. The fusion of local detection results $\hat{\mathbf{x}}_c$ in the uplink and broadcasting of the transmit signal $\mathbf{s}$ in the

TABLE I
LATENCY (L) AND THROUGHPUT (TP) PERFORMANCE OF DECENTRALIZED CD DATA DETECTION AND PRECODING

	Uplink data detection				Downlink precoding			
$U = 8$, $B_c = 32$	$C = 4$, $B = 128$		$C = 8$, $B = 256$		$C = 4$, $B = 128$		$C = 8$, $B = 256$	
L [ms] / TP [Gbps]	fp32	fp16	fp32	fp16	fp32	fp16	fp32	fp16
$T_{\text{max}} = 2$	0.465/1.23	0.259/2.21	0.502/1.14	0.295/1.95	0.414/1.38	0.228/2.52	0.440/1.30	0.233/2.46
$T_{\text{max}} = 3$	0.540/1.06	0.298/1.92	0.585/0.98	0.331/1.73	0.499/1.15	0.267/2.15	0.525/1.09	0.275/2.09
$T_{\text{max}} = 4$	0.617/0.93	0.341/1.68	0.665/0.86	0.370/1.55	0.577/0.99	0.305/1.88	0.601/0.95	0.311/1.84

TABLE II
PERFORMANCE COMPARISON BETWEEN DIFFERENT MASSIVE MU-MIMO DATA DETECTORS (-D) AND PRECODERS (-P)

	Neumann-D [14]	OCD-D [13]	FD-LAMA-D [12]	**DCD-D**	FD-WF-P [10]	**DCD-P**
Fabric	ASIC (fixed-point)	FPGA (fixed-point)	GPU (fp32)	**GPU (fp16)**	GPU (fp32)	**GPU (fp16)**
Archtecture	Centralized	Centralized	Decentralized	**Decentralized**	Decentralized	**Decentralized**
Antenna	128×8	128×8	128×16	$\mathbf{128 \times 8}$	128×16	$\mathbf{128 \times 8}$
Modulation	64-QAM	64-QAM	16-QAM	**16-QAM**	64-QAM	**16-QAM**
Iteration	3	3	3	**3**	N/A	**3**
Throughput (TP)	3.8 Gbps	0.38 Gbps	1.34 Gbps	**1.92 Gbps**	1.91 Gbps	**2.15 Gbps**
TP/UE @ 16-QAM	317 Mbps	31 Mbps	84 Mbps	**240 Mbps**	80 Mbps	**269 Mbps**

downlink are realized by inter-process communication over the NvLink. Specifically, we use the `ncclReduce` function in NCCL for efficient data fusion and reduction, and use the `ncclBcast` function for fast broadcasting, both leveraging direct GPU-to-GPU memory copy for low latency.

The local CD computations on each GPU are implemented by multi-threaded customized kernels. According to Algorithm 1 and Algorithm 2, the dominating operations of CD are vector operations instead of matrix operations. For vector scaling, addition, and subtraction operations, which have native parallelism, we straightforwardly generate multiple GPU threads to compute each element in parallel. Our algorithms also involve vector dot product and vector norm computations, which require aggregation, i.e., inter-thread communication, of element-wise product results in a pair of vectors. Here, we resort to the *warp shuffle* technique to realize direct *register-to-register* copy among threads within a *warp* [17], which significantly improves inter-thread communication efficiency compared to conventional solutions based on slower shared memories or global memories. To reduce high-latency global memory transactions, we combine multiple vector operations into a single kernel function, so that intermediate computation results are shared with fast on-chip memories within the kernel. To fully exploit the GPU computing resources for high throughput, we process local detection or precoding workload for a large amount of OFDM subcarriers together with thousands of threads in parallel to keep high GPU occupancy.

To further improve throughput, we explored fp16 implementations supported by the latest CUDA release. Taking advantage of single-instruction multiple-data (SIMD) operations realized by CUDA fp16 *intrinsic* functions [17], in a single thread, we can simultaneously operate on two fp16 values packed inside a 32-bit `half2` type data, which leads to higher parallelism and computation efficiency. In addition, fp16 reduces the message size of inter-GPU data transfer to half of the fp32 message, decreasing transfer latency in our decentralized designs.

C. Data-Rate Performance

Table I reports the latency and throughput of our CD data detector (DCD-D) and precoder (DCD-P) for various antenna configurations and both fp32 and fp16 implementations. We fix $U = 8$ and $B_c = 32$, and measure the performance of our decentralized designs on $C = \{4, 8\}$ GPUs, which corresponds to $B = \{128, 256\}$. To characterize the performance of our implementations in a more practical setup, we assumed an LTE scenario in which we used batched processing workload with 1200 OFDM data subcarriers. For all antenna configurations, the data rate of fp16 based designs significantly outperforms the fp32 ones. If we scale up the total number of BS antennas by increasing C, then the throughput degrades only slightly; this demonstrates the scalability of our decentralized designs in systems with a large number of BS antennas.

Table II compares the throughput performance of existing algorithms implemented on different computing fabrics, such as ASIC [14], FPGA [13], and GPU [10], [12]. The proposed decentralized fp16 CD implementations achieve $2.5\times$ to $3.5\times$ improvements compared to existing fp32 fully-decentralized detectors and precoders in terms of per-user throughput where the throughput is normalized to 16-QAM[1].

VII. CONCLUSIONS

We have presented decentralized CD-based data detection and precoding algorithms for massive MU-MIMO systems which mitigate the computation and interconnection bottlenecks in typical centralized designs. Our methods require lower complexity than existing methods without explicit matrix multiplication and inversions. We have demonstrated the hardware efficiency and design scalability with fp32 and fp16 multi-GPU implementations, and have realized a $3\times$ speed up on per-user throughput compared to previous decentralized detectors and precoders. We conclude by noting that our current GPU implementations serve as a fast prototyping solution of our algorithms, while the power dissipation of C Tesla V100 GPUs can be as high as $C \times 300$ W. To reduce power, one can resort to multi-FPGA and multi-ASIC designs; an analysis of such more efficient solutions is left for future work.

[1]Scaling the modulation scheme to 16-QAM for architectures supporting higher-order constellations may penalize these designs as they were optimized for a different target rate; the penalty, however, is expected to be rather small.

REFERENCES

[1] E. Larsson, O. Edfors, F. Tufvesson, and T. Marzetta, "Massive MIMO for next generation wireless systems," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 186–195, Feb. 2014.

[2] J. Andrews, S. Buzzi, W. Choi, S. Hanly, A. Lozano, A. Soong, and J. Zhang, "What will 5G be?" *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1065–1082, Jun. 2014.

[3] K. Li, R. R. Sharan, Y. Chen, T. Goldstein, J. R. Cavallaro, and C. Studer, "Decentralized baseband processing for massive MU-MIMO systems," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 7, no. 4, pp. 491–507, Nov. 2017.

[4] *Common public radio interface*. [Online]. Available: http://www.cpri.info/

[5] E. Bertilsson, O. Gustafsson, and E. G. Larsson, "A distributed processing architecture for modular and scalable massive MIMO base stations," *arXiv preprint: 1801.07967*, 2018.

[6] E. Bertilsson, O. Gustafsson, and E. G. Larsson, "A scalable architecture for massive MIMO base stations using distributed processing," in *Proc. Asilomar Conf. Signals, Syst., Comput.*, Nov 2016, pp. 864–868.

[7] C. Desset, E. Björnson, S. Kashyap, E. G. Larsson, C. Mollén, L. Liu, S. Malkowsky, H. Prabhu, Y. Liu, J. Vieira, O. Edfors, E. Karipidis, F. Dielacher, and D. V. Pop, "Distributed and centralized baseband processing algorithms, architectures, and platforms," MAMMOET, Tech. Rep. ICT-619086-D3.2, 2016.

[8] K. Li, Y. Chen, R. Sharan, T. Goldstein, J. R. Cavallaro, and C. Studer, "Decentralized data detection for massive MU-MIMO on a Xeon Phi cluster," in *Proc. Asilomar Conf. Signals, Syst., Comput.*, Nov. 2016, pp. 468–472.

[9] K. Li, R. Sharan, Y. Chen, J. R. Cavallaro, T. Goldstein, and C. Studer, "Decentralized beamforming for massive MU-MIMO on a GPU cluster," in *Global Conf. Sig. Inform. Process. (GlobalSIP)*, Dec. 2016, pp. 590–594.

[10] K. Li, C. Jeon, J. R. Cavallaro, and C. Studer, "Feedforward architectures for decentralized precoding in massive MU-MIMO systems," *arXiv preprint: 1804.10987*, 2018.

[11] C. Jeon, K. Li, J. R. Cavallaro, and C. Studer, "On the achievable rates of decentralized equalization in massive MU-MIMO systems," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 1102–1106.

[12] C. Jeon, K. Li, J. R. Cavallaro, and C. Studer, "Decentralized equalization with feedforward architectures for massive MU-MIMO," *arXiv preprint: 1808.04473*, 2018.

[13] M. Wu, C. Dick, J. R. Cavallaro, and C. Studer, "FPGA design of a coordinate descent data detector for large-scale MU-MIMO," in *Proc. IEEE Int. Symp. Circuits and Syst. (ISCAS)*, May 2016, pp. 1894–1897.

[14] B. Yin, M. Wu, G. Wang, C. Dick, J. R. Cavallaro, and C. Studer, "A 3.8Gb/s large-scale MIMO detector for 3GPP LTE-Advanced," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, May 2014, pp. 3879–3883.

[15] *Nvidia DGX-1 system*. [Online]. Available: https://www.nvidia.com/en-us/data-center/dgx-1

[16] *Nvidia NvLink fabric*. [Online]. Available: https://www.nvidia.com/en-us/data-center/nvlink

[17] *Nvidia CUDA programming guide*. [Online]. Available: http://docs.nvidia.com/cuda

[18] *Nvidia collective communications library*. [Online]. Available: https://developer.nvidia.com/nccl