Efficient Coordination of Radio Frames to Mitigate Cross-Link Interference in 5G D-TDD Systems

You-Chiun Wang and Chia-Wei Chou

Abstract—*Dynamic time division duplexing (D-TDD)* is widely used in 5G to support various services and short-latency communications, where uplink (UL) and downlink (DL) traffic may be highly asymmetric and fast-varying. In particular, D-TDD allows each cell to arrange subframes in terms of selecting link directions, which betters the utilization of spectrum resources based on instantaneous traffic demand. However, the inter-cell subframe misalignment, which is caused by different link directions on the same frequency band in neighboring cells, leads to notable *cross-link interference (CLI)*. Especially, CLI has a negative impact on UL performance. Hence, this paper proposes a *cluster-based radio frame coordination using codebooks (CRCC)* framework to mitigate CLI efficiently. CRCC first clusters cells and formulates a codebook of radio frame configuration for each cluster, where cells pick their configuration of subframes. Then, the cluster is divided into subclusters, where the cells in each subcluster adjust subframes to reduce misalignments. Finally, CRCC combines subclusters by deciding the subframes of those cells located on the boundaries between subclusters. Simulation results reveal that CRCC greatly increases UL throughput and keeps high DL throughput, thereby improving the performance of a 5G D-TDD system.

Index Terms—cross-link interference, D-TDD, radio frame configuration.

1 Introduction

The *time division duplexing (TDD)* technology has been highly regarded in the development of cellular networks. There are two types of TDD systems, namely *static TDD (S-TDD)* and *dynamic TDD (D-TDD)*. In S-TDD, all cells adopt the same arrangement of subframes for uplink/downlink (UL/DL) communications, which is usually configured based on the long-term average of UL/DL traffic loads. Since UL/DL slots are synchronized with a common frame structure in every cell, *inter-cell interference* is ineluctable. More concretely, a *user equipment (UE)* in a UL cell disturbs the *base station (BS)* in another UL cell obtaining data, called *UE-to-BS (U2B) interference*, as shown in Fig. 1(a). Besides, the BS in a DL cell interferes with the UEs in an adjoining DL cell receiving data, also known as *BS-to-UE (B2U) interference*, as Fig. 1(b) shows. Since a BS has much larger transmitted power[1] than a UE, B2U interference is more severe than U2B interference. Hence, many studies aim to mitigate B2U interference [1]–[3].

Owing to the random nature of traffic loads, S-TDD systems may encounter long packet delays [4]. Hence, 3GPP proposes the technique of *enhanced interference mitigation and traffic adaptation (eIMTA)* to support D-TDD [5]. Specifically, eIMTA provides different *radio frame configuration (RFC)* patterns for TDD with various UL to DL traffic ratios. Each cell can arrange radio frames based on its criteria for selecting the link direction, which helps raise throughput and reduce packet delays. In addition, 5G supports flexible TDD slot format and variable *transmission time interval (TTI)* [6], so the periodicity of link direction switching can be slot-based (i.e., shorter than 1 ms). In this way, 5G D-TDD systems improve the spectrum utilization even with fast-varying and asymmetric UL/DL traffic, as compared with S-TDD systems [7].

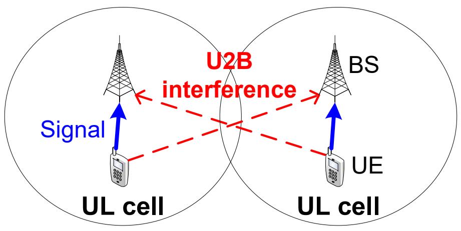
The authors are with the Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung 80424, Taiwan (e-mail: ycwang@cse.nsysu.edu.tw; gino8534@gmail.com).

1. Below, we call it *power* for short.

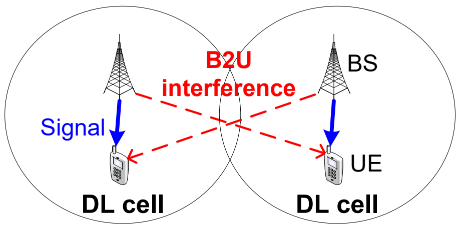
However, D-TDD has the side effect of *cross-link interference (CLI)*, which is caused by different link directions on the same frequency band in neighboring cells (also known as the *inter-cell subframe misalignment*). There are two types of CLI, as Fig. 1(c) shows. When a UE in a UL cell disturbs another UE in a DL cell receiving data, it is called *UE-to-UE (U2U) interference*. On the other hand, since the BS of the DL cell also interferes with the BS of the UL cell receiving data, *BS-to-BS (B2B) interference* will occur. Due to the power imbalance between the BS and UEs, B2B interference may significantly degrade UL throughput and becomes a critical issue in D-TDD [8]. Therefore, how to efficiently arrange the subframes of cells to mitigate CLI plays a key role in improving 5G D-TDD performance.

In this paper, we propose a *cluster-based radio frame coordination using codebooks (CRCC)* framework to address the subframe arrangement problem. First, CRCC partitions cells into clusters, such that the cells in a cluster would impose non-neglected CLI on each other. In each cluster, we select one cell to be the master, where the distance between the master and the other cells in the cluster is the smallest. The master formulates a shared codebook of RFCs as a reference, and other cells choose RFCs from the codebook according to their UL and DL demand. Moreover, the master also takes charge of coordinating RFCs among cells. With the *divide-and-conquer* concept, the cluster is divided into subclusters, and the master performs RFC coordination for the cells in each subcluster to minimize inter-cell subframe misalignments. Finally, CRCC merges subclusters by adjusting the subframes of cells on the boundaries between subclusters. Simulation results display that the CRCC framework greatly increases UL throughput and also keeps DL throughput high. Hence, CRCC can efficiently improve the performance of a 5G D-TDD system, as compared with the existing methods.

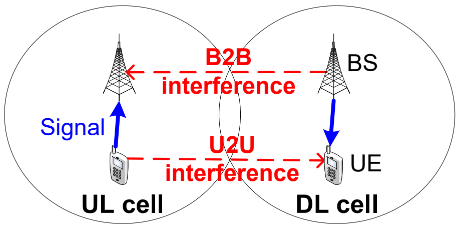
The rest of this paper is organized as follows: Section 2 discusses the related work. Section 3 presents the system model. We elaborate on the CRCC framework in Section 4 and



(a) U2B interference



(b) B2U interference



(c) CLI

Fig. 1: Inter-cell interference and CLI.

then evaluate system performance by simulation in Section 5. Finally, Section 6 concludes this paper.

2 Related Work

In the literature, the CLI issue has received considerable attention. Given A_{B} antennas equipped on a BS, the work [9] analyzes CLI and transceiver noise by using the random matrix theory, and proves that their effects can scale down to $O(A_{\text{B}}^{-1})$. The study [7] takes account of the CLI effect when allocating spectrum resources to UEs. In [10], the orthogonal spatial projection is used to suppress CLI to satisfy the outage latency demand of URLLC (ultra-reliable and low latency communications) flows. Pedersen et al. [11] discuss how to use UEs as sensors for detecting CLI. In our work, we aim to mitigate CLI by arranging the subframes of cells. Evidently, the above studies have different objectives from ours.

Many studies [12]–[15] mitigate B2B interference (i.e., a type of CLI) by power control to raise UL performance. Since the BS in a DL cell causes B2B interference (as shown in Fig. 1(c)), these studies lower the BS's power. Moreover, they boost the power of UEs in a UL cell to reduce the effect of B2B interference. However, lowering the BS's power degrades DL channel quality, and boosting UEs' power worsens U2U interference. By using coordinated beamforming, the work [16] adjusts the direction of a receiver's antenna to diminish the CLI effect. Kim et al. [17] combine both scheduling and beamforming to handle CLI. However, it incurs a high cost to implement the above beamforming methods because of UE mobility. Ericsson proposes CLI mitigation strategies based on hybrid TDD for the scenarios of dense urban [18] and indoor hotspot [19]. Specifically, Ericsson suggests using S-TDD only when there exists traffic in opposite directions to be scheduled in that cell or in any co-located cell; otherwise, D-TDD can be adopted. The work [20] divides a cell into center and edge regions. Cell-center UEs choose D-TDD to improve performance, and cell-edge UEs employ S-TDD to avoid CLI. Nevertheless, hybrid TDD may complicate TDD management, especially when UEs move across different cells or regions.

Lee et al. [21] consider a *multi-input multi-output (MIMO)* system with a 2D antenna array, which is given as a single panel with multiple horizontal and vertical BS antenna elements. They propose a decentralized slot assignment method to reduce CLI based on the location-dependent information and 3D spatial angle for each UE in a cell. However, the proposed method could not be applied to other MIMO systems. Chen et al. [22] develop a two-phase approach to eliminate B2B interference. In phase 1, the interfering BS and the UEs in the interfered cell send signal vectors at the same time. In phase 2, only the interfering BS sends signal vectors, which is gotten by the interfered BS for mitigating interference caused in phase 1. However, this approach may result in a high signal overhead.

How to select RFC patterns for cells is also discussed. In [23], each cell picks one of four RFC patterns in eIMTA (specifically, pattern numbers 0, 1, 2, and 6) based on its ratio of UL to DL traffic. Li et al. [24] group the cells with severe B2B/U2U interference and similar transmission demand together, and choose an eIMTA RFC pattern for each group to meet the demand of its cells. The work [25] defines a common codebook of RFCs, where each cell selects an RFC fit for its traffic load. As compared with the traditional eIMTA scheme, using a codebook of RFCs is more flexible. However, the work [25] does not consider that the strength of CLI will be different depending on the distance between two BSs and treats each cell in the same way. Thus, some nearby cells may still cause non-neglected CLI. To this end, we develop the CRCC framework based on a regional concept, which recursively divides cells into subclusters and carefully arranges their subframes to mitigate CLI. Doing so can efficiently improve UL performance without significantly degrading DL performance.

3 System Model

In this section, we introduce the network model, discuss how to estimate the *signal-to-interference-plus-noise ratio (SINR)*, and formulate the subframe arrangement problem.

3.1 Network Model

Let us consider a service area covered by 5G NR macrocells, where D-TDD and MIMO techniques are used. Each BS has A_{B} antennas and each UE is equipped with A_{U} antennas, where $A_{\text{B}} > A_{\text{U}}$. In a subframe, UEs may send data to the BS (i.e., UL UEs) or receive data from the BS (i.e., DL UEs). Depending on their traffic demand, there are three types of cells. Specifically, let D_{UL} and D_{DL} be the amount of UL and DL demand in a cell, respectively. A cell is called *UL-biased*, *DL-biased*, or *neutral* cell if $D_{\text{UL}} > \alpha D_{\text{DL}}$, $D_{\text{DL}} > \alpha D_{\text{UL}}$, or otherwise, respectively, where $1 \leq \alpha < 1.2$. For example, we can set $\alpha = 1.1$. In this case, when the amount of UL demand is 10% more than that of DL demand, the cell is UL-biased. Conversely, if the amount of DL demand is 10% more than that of UL demand, the cell is DL-biased.

For management, cells are grouped into clusters, where the cells in each cluster may impose significant CLI on each other (how to group cells will be discussed later in Section 4.1). In

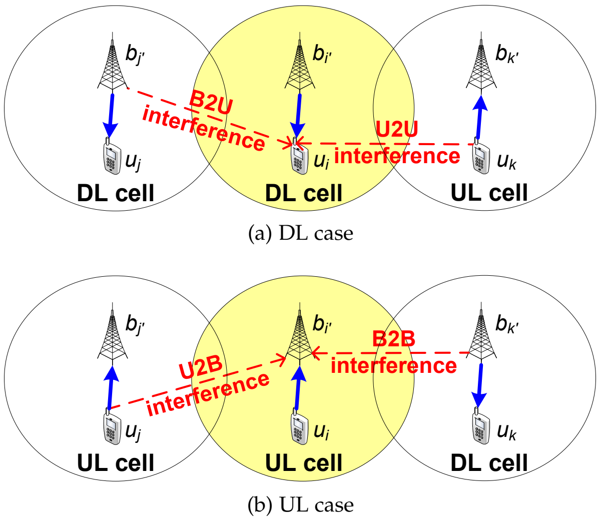


Fig. 2: Two cases for SINR estimation.

each cluster, a cell is selected as the *master* to take charge of the coordination of radio frames. It works out a codebook of RFCs and shares the codebook with other cells (called *slaves*). Each slave chooses its preferred RFC from the codebook based on its UL and DL demand and sends a request to the master. With these requests, the master performs RFC coordination for slaves to mitigate CLI. Here, the master's BS can communicate with a slave's BS through the 5G Xn interface [26].

3GPP defines seven numerologies to support different kinds of subcarrier spacing [27]. Each numerology is labeled by a parameter μ, where $\mu = 0, 1, \cdots, 6$. Take the numerology ($\mu = 0$) as an example. A frame is composed of 10 subframes. Each subframe is 1 TTI (= 1 ms) in length and can be one of UL, DL, and special subframes. The smallest unit of spectrum resources for scheduling and allocation is known as a *physical resource block*, which contains 12 subcarriers in the frequency domain. Each subcarrier has 15 kHz bandwidth. If the normal cyclic prefix is adopted, there are 14 *orthogonal frequency division multiplexing (OFDM)* symbols in a TTI. Since we aim at the arrangement of subframes, our CRCC framework can still perform well when using other numerologies.

3.2 SINR Estimation

Let $\hat{\mathcal{B}}_{\text{UL}}$ and $\hat{\mathcal{B}}_{\text{DL}}$ be the sets of BSs in UL and DL cells of a cluster, respectively. Besides, $\hat{\mathcal{U}}_{\text{UL}}$ and $\hat{\mathcal{U}}_{\text{DL}}$ denote the sets of UEs conducting UL and DL communications, respectively. For ease of explanation, we assume that a UE u_i is communicating with a BS $b_{i'}$. Fig. 2(a) shows the DL case (i.e., $u_i \in \hat{\mathcal{U}}_{\text{DL}}$ and $b_{i'} \in \hat{\mathcal{B}}_{\text{DL}}$). In particular, the compound signal received by UE u_i is expressed by

$$\mathbf{H}^{\text{DL}}_{i,i'}\mathbf{x}_{i'}s_i + \sum_{u_j \in \hat{\mathcal{U}}_{\text{DL}} \setminus u_i} \mathbf{H}^{\text{DL}}_{i,j'}\mathbf{x}_{j'}s_j + \sum_{u_k \in \hat{\mathcal{U}}_{\text{UL}}} \mathbf{G}_{i,k}\mathbf{y}_k s_k + \epsilon^{\text{DL}}_i, \quad (1)$$

where $\mathbf{H}^{\text{DL}}_{i,i'}$ denotes the DL channel coefficient matrix between u_i and $b_{i'}$ (size: $A_{\text{U}} \times A_{\text{B}}$), $\mathbf{x}_{i'}$ is the single-stream precoding vector of $b_{i'}$ (size: $A_{\text{B}} \times 1$), s_i is the data symbol of u_i, $\mathbf{G}_{i,k}$ is the channel coefficient matrix between UEs u_i and u_k (size: $A_{\text{U}} \times A_{\text{U}}$), and $\mathbf{y}_k$ is the single-stream precoding vector of u_k (size: $A_{\text{U}} \times 1$). Here, only the 1st term $\mathbf{H}^{\text{DL}}_{i,i'}\mathbf{x}_{i'}s_i$ is the useful signal for u_i, and other terms are interference or noise. Specifically, the 2nd term $\sum_{u_j \in \hat{\mathcal{U}}_{\text{DL}} \setminus u_i} \mathbf{H}^{\text{DL}}_{i,j'}\mathbf{x}_{j'}s_j$ estimates B2U interference from other BSs (e.g., the left cell in Fig. 2(a)), the 3rd term $\sum_{u_k \in \hat{\mathcal{U}}_{\text{UL}}} \mathbf{G}_{i,k}\mathbf{y}_k s_k$ calculates U2U CLI from other cells (e.g., the right cell in Fig. 2(a)), and the 4th term ϵ^{DL}_i is the *additive white Gaussian noise (AWGN)* at u_i. Then, we can compute u_i's SINR by

$$\gamma^{\text{DL}}_i = \frac{\|\mathbf{H}^{\text{DL}}_{i,i'}\mathbf{x}_{i'}\|^2 p^{\text{DL}}_{i'}}{\sum\limits_{u_j \in \hat{\mathcal{U}}_{\text{DL}} \setminus u_i} \|\mathbf{H}^{\text{DL}}_{i,j'}\mathbf{x}_{j'}\|^2 p^{\text{DL}}_{j'} + \sum\limits_{u_k \in \hat{\mathcal{U}}_{\text{UL}}} \|\mathbf{G}_{i,k}\mathbf{y}_k\|^2 p^{\text{UL}}_k + \sigma^2}, \quad (2)$$

where $p^{\text{DL}}_{i'}$ is the power of BS $b_{i'}$, $p^{\text{DL}}_{j'}$ is the power of BS $b_{j'}$ (which communicates with UE u_j), and p^{UL}_k is the power of UE u_k. Besides, σ is the standard deviation of AWGN. The notation $\|\mathbf{Z}\|$ represents the Frobenius norm of a matrix $\mathbf{Z} = [z_{i,j}]_{I \times J}$, which is defined by [28]

$$\|\mathbf{Z}\| = \sqrt{\sum\nolimits_{i=1}^{I} \sum\nolimits_{j=1}^{J} |z_{i,j}|^2}. \quad (3)$$

Fig. 2(b) gives the UL case (i.e., $u_i \in \hat{\mathcal{U}}_{\text{UL}}$ and $b_{i'} \in \hat{\mathcal{B}}_{\text{UL}}$). The compound signal gotten by BS $b_{i'}$ can be expressed by

$$\mathbf{H}^{\text{UL}}_{i',i}\mathbf{y}_i s_i + \sum_{u_j \in \hat{\mathcal{U}}_{\text{UL}} \setminus u_i} \mathbf{H}^{\text{UL}}_{i',j}\mathbf{y}_j s_j + \sum_{u_k \in \hat{\mathcal{U}}_{\text{DL}}} \mathbf{F}_{i',k'}\mathbf{x}_{k'}s_k + \epsilon^{\text{UL}}_{i'}, \quad (4)$$

where $\mathbf{H}^{\text{UL}}_{i',i}$ is the UL channel coefficient matrix between $b_{i'}$ and u_i (size: $A_{\text{B}} \times A_{\text{U}}$), and $\mathbf{F}_{i',k'}$ is the channel coefficient matrix between BSs $b_{i'}$ and $b_{k'}$ (size: $A_{\text{B}} \times A_{\text{B}}$). Here, the 1st term $\mathbf{H}^{\text{UL}}_{i',i}\mathbf{y}_i s_i$ shows the useful signal for $b_{i'}$. Then, the 2nd term $\sum_{u_j \in \hat{\mathcal{U}}_{\text{UL}} \setminus u_i} \mathbf{H}^{\text{UL}}_{i',j}\mathbf{y}_j s_j$ is U2B interference from other UEs in UL cells (e.g., the left cell in Fig. 2(b)), the 3rd term $\sum_{u_k \in \hat{\mathcal{U}}_{\text{DL}}} \mathbf{F}_{i',k'}\mathbf{x}_{k'}s_k$ gives B2B CLI from other cells (e.g., the right cell in Fig. 2(b)), and the last term $\epsilon^{\text{UL}}_{i'}$ is the AWGN at $b_{i'}$. Thus, we can estimate the SINR of BS $b_{i'}$ with relation to u_i's signal as follows:

$$\gamma^{\text{UL}}_i = \frac{\|\mathbf{H}^{\text{UL}}_{i',i}\mathbf{y}_i\|^2 p^{\text{UL}}_i}{\sum\limits_{u_j \in \hat{\mathcal{U}}_{\text{UL}} \setminus u_i} \|\mathbf{H}^{\text{UL}}_{i',j}\mathbf{y}_j\|^2 p^{\text{UL}}_j + \sum\limits_{u_k \in \hat{\mathcal{U}}_{\text{DL}}} \|\mathbf{F}_{i',k'}\mathbf{x}_{k'}\|^2 p^{\text{DL}}_{k'} + \sigma^2}. \quad (5)$$

In Eq. (5), since a BS has much larger power than a UE (i.e., $p^{\text{DL}}_{k'} \gg p^{\text{UL}}_i$), B2B CLI evidently decreases the SINR of a UE conducting UL communication. This explains why CLI is in particular harmful to UL performance.

3.3 Subframe Arrangement Problem

Since cells are grouped into clusters such that the cells in different clusters will not impose significant CLI on each other, our discussion aims at a cluster $\hat{\mathcal{C}}$ of cells. Suppose that Θ is a solution set of RFCs for $\hat{\mathcal{C}}$. Then, each cell $c_i \in \hat{\mathcal{C}}$ is assigned an RFC $\bar{\theta}_i = (\theta^1_i, \theta^2_i, \cdots, \theta^m_i)$, where $\bar{\theta}_i \in \Theta$ and m is the number of subframes in one RFC. Each field θ^k_i indicates the type of the k-th subframe in $\bar{\theta}_i$; specifically, $\theta^k_i = 0$, $\theta^k_i = 1$, and $\theta^k_i = -2$ mean UL, DL, and special subframes, respectively, where $k = 1, \cdots, m$. Moreover, we denote by $N_{\text{UL}}(\bar{\theta}_i)$ and $N_{\text{DL}}(\bar{\theta}_i)$ the numbers of UL and DL subframes in $\bar{\theta}_i$, respectively.

Based on the Shannon capacity theorem, the maximum UL and DL data rates in a cell $c_i \in \hat{\mathcal{C}}$ can be estimated by

$$R^{\text{UL}}_i = \sum\nolimits_{u_a \in \hat{\mathcal{U}}_i \cap \hat{\mathcal{U}}_{\text{UL}}} \xi \log_2(1 + \gamma^{\text{UL}}_a), \quad (6)$$

$$R^{\text{DL}}_i = \sum\nolimits_{u_a \in \hat{\mathcal{U}}_i \cap \hat{\mathcal{U}}_{\text{DL}}} \xi \log_2(1 + \gamma^{\text{DL}}_a), \quad (7)$$

TABLE 1: Summary of acronyms.

acronym	full name
AWGN	additive white Gaussian noise
B2B	BS-to-BS (BS: base station)
B2U	BS-to-UE (UE: user equipment)
CDF	cumulative distribution function
CLI	cross-link interference
CQI	channel quality indicator
CRCC	cluster-based radio frame coordination with codebooks
eAHC	enhanced agglomerative hierarchical clustering
eIMTA	enhanced interference mitigation and traffic adaptation
MCS/MCT	majority choice of subcodebook/type
MIMO	multi-input multi-output
OFDM	orthogonal frequency-division multiplexing
RFC	radio frame configuration
RFCbCB	RFC-based sliding codebook
SINR	signal-to-interference-plus-noise ratio
TDD	time division duplexing (S/D-TDD: static/dynamic TDD)
TORS	traffic-oriented RFC selection
TTI	transmission time interval
U2U/U2B	UE-to-UE/UE-to-BS
UL/DL	uplink/downlink

where $\hat{\mathcal{U}}_i$ is the set of UEs in c_i and ξ is the channel bandwidth. Furthermore, the number of subframe misalignments between two cells c_i and c_j in $\hat{\mathcal{C}}$ is calculated by

$$\vartheta_{i,j} = \sum\nolimits_{k=1}^{m} f(\theta_i^k, \theta_j^k), \tag{8}$$

where

$$f(\theta_i^k, \theta_j^k) = \begin{cases} 0 & \text{if } \theta_i^k + \theta_j^k < 0 \\ \theta_i^k \oplus \theta_j^k & \text{otherwise.} \end{cases} \tag{9}$$

In Eq. (9), $\oplus$ is the exclusive-or (XOR) operator. Note that if any of θ_i^k and θ_j^k is a special subframe (in this case, we have $\theta_i^k + \theta_j^k < 0$), there will be no subframe misalignment.

Then, the subframe arrangement problem asks how to find Θ to satisfy two objective functions:

$$\text{maxmize} \sum\nolimits_{c_i \in \hat{\mathcal{C}}} R_i^{\text{UL}} \times N_{\text{UL}}(\bar{\theta}_i)t + R_i^{\text{DL}} \times N_{\text{DL}}(\bar{\theta}_i)t, \tag{10}$$

$$\text{minimize} \frac{\sum_{c_i \in \hat{\mathcal{C}}} \sum_{c_j \in \hat{\mathcal{C}}, c_j \neq c_i} \vartheta_{i,j}}{\tilde{\text{P}}\binom{|\hat{\mathcal{C}}|}{2}}, \tag{11}$$

where t is the length of a subframe (in seconds) and $\tilde{\text{P}}\binom{|\hat{\mathcal{C}}|}{2} = \frac{|\hat{\mathcal{C}}|!}{(|\hat{\mathcal{C}}|-2)!}$ (i.e., permutation). Specifically, Eq. (10) is to maximize the overall throughput in $\hat{\mathcal{C}}$. However, one may find a solution to maximize DL throughput with poor UL throughput, which fulfills the objective function in Eq. (10) but is actually a bad solution. To avoid this situation and assure UL performance, we additionally use the objective function in Eq. (11) to minimize the average number of subframe misalignments between any two cells in $\hat{\mathcal{C}}$, so as to prevent CLI from degrading UL performance. Table 1 lists acronyms, and Table 2 summarizes notations.

4 The Proposed CRCC Framework

Algorithm 1 gives the pseudocode of our CRCC framework, which contains five modules. First of all, the *cell grouping module* divides all cells in the service area into clusters, where the cells in each cluster would impose significant CLI on each other, as shown in line 1. Then, for each cluster $\hat{\mathcal{C}}$, the *codebook formulating module* works out a codebook of RFCs (denoted by Γ) for the cells in $\hat{\mathcal{C}}$ to select their RFCs, as shown in line 3. In line 4, we recursively partition $\hat{\mathcal{C}}$ into subclusters by the

TABLE 2: Summary of notations.

notation	definition
$\hat{\mathcal{C}}$	set of cells in a cluster
$\hat{\mathcal{C}}_{i,j}$	j-th subcluster to be divided in iteration i
$T_{i,j}$	dividing line to partition $\hat{\mathcal{C}}_{i,j}$
Γ	a codebook of RFCs ($\Gamma = \{\Gamma_1, \Gamma_2, \cdots, \Gamma_h\}$)
$\bar{\theta}_{x,y}$	y-th RFC in subcodebook Γ_x
φ	the maximum number of cells in a basic subcluster
$\tilde{d}(\cdot,\cdot)$	distance between two clusters or BSs
m	the number of subframes in an RFC
$\beta_x^{\text{UL}}, \beta_x^{\text{DL}}, \beta_x^{\text{S}}$	the numbers of UL, DL, special subframes in an RFC
$\vartheta_{l,\text{A}}$	the number of subframe misalignments between the RFC chosen by a cell and the anchor RFC $\bar{\theta}_{\text{A}}$
$\delta_{\text{SDM}}, \delta_{\text{SMM}}$	thresholds on subframe misalignments used in the subframe deciding and subcluster merging modules

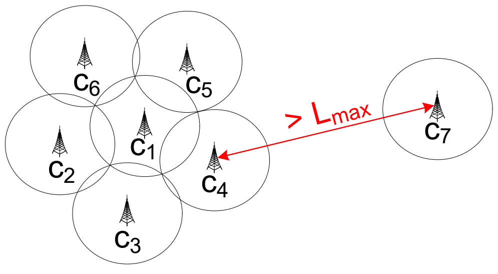
Algorithm 1: The CRCC framework

```
1 Divide cells into clusters by the cell grouping module;
2 foreach cluster $\hat{\mathcal{C}}$ do
3     Find a codebook $\Gamma$ of RFCs by the codebook formulating module;
4     Recursively partition $\hat{\mathcal{C}}$ until to basic subclusters by the cluster dividing module;
5     foreach basic subcluster $\hat{\mathcal{C}}_{i,j}$ in $\hat{\mathcal{C}}$ do
6         Arrange the subframes of cells in $\hat{\mathcal{C}}_{i,j}$ by the subframe deciding module;
7     Recursively combine subclusters until to $\hat{\mathcal{C}}$ by the subcluster merging module;
```

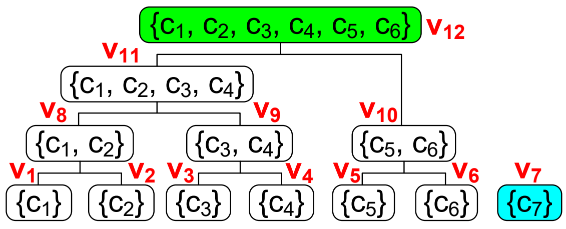
cluster dividing module, until to basic subclusters. Afterward, the *subframe deciding module* helps the cells in each basic subcluster arrange subframes to reduce misalignments. The code is given in lines 5 and 6. Finally, the *subcluster merging module* recursively combines subclusters until to $\hat{\mathcal{C}}$, as shown in line 7. When combining two subclusters, the subcluster merging module may tune the subframes of those cells located on the boundaries between these two subclusters.

The objective of each module and how it contributes to the overall solution is given as follows:

- Cell grouping module: Perform preliminary clustering of cells. Hence, the remaining modules can focus on handling the cells in a cluster $\hat{\mathcal{C}}$ without considering the effect from other clusters. The cell grouping module is meant to facilitate the operations of other modules.
- Codebook formulating module: Define a shared codebook Γ of RFCs for cluster $\hat{\mathcal{C}}$. Each cell in $\hat{\mathcal{C}}$ takes codebook Γ as a reference to arrange its subframes (which is done by the subframe deciding module). This helps each cell meet its traffic demand while reducing the effect of CLI.
- Cluster dividing module: Apply the divide-and-conquer strategy (i.e., the part of dividing the problem into subproblems). This module recursively breaks down $\hat{\mathcal{C}}$ into two or more subclusters, until each subcluster has just a few cells (i.e., basic subclusters). In this way, we can easily solve the subframe arrangement problem in each basic subcluster.
- Subframe deciding module: Arrange the subframes of cells. The module cooperates with the codebook formulating module by consulting its generated codebook Γ to configure subframes. The objective is to satisfy both



(a) distribution of cells



(b) forest formed by eAHC

Fig. 3: Example of grouping cells by the eAHC scheme.

UL and DL demand of each cell and also minimize subframe misalignments to mitigate CLI.

- Subcluster merging module: Also apply the divide-and-conquer strategy (i.e., the part of combining sub-problems into the original problem). Since the cells in adjacent subclusters may still impose CLI on each other, this module adjusts their subframes to decrease sub-frame misalignments. The subcluster merging module together with the cluster dividing module address the weaknesses encountered in the study [25][2].

Below, we elaborate on these five modules, followed by a discussion on CRCC's rationale.

4.1 Cell Grouping Module

As mentioned in Section 1, CLI is caused by different link directions in nearby cells. In other words, the effect of CLI from far cells can be actually ignored. In view of this, the cell grouping module partitions cells into clusters, so that any two clusters are far enough apart and their cells will not impose significant CLI on each other. Hence, we can perform the remaining modules on each cluster separately, thereby reducing unnecessary computation.

To do so, we adopt the *enhanced agglomerative hierarchical clustering (eAHC)* scheme [29] to group cells in the service area. This scheme starts by treating each cell as one singleton cluster. Then, pairs of clusters are recursively merged until the condition of merging cannot be met. The result will be a forest of binary trees. More concretely, eAHC comprises the following steps:

1. In the beginning, each cell c_l is added to an individual cluster. The cluster is assigned a tree node v_l.
2. Let $\tilde{d}(\hat{\mathcal{C}}_i, \hat{\mathcal{C}}_j)$ be the *inter-cluster distance (ICD)* between two clusters $\hat{\mathcal{C}}_i$ and $\hat{\mathcal{C}}_j$, as defined by the Euclidean distance between the two closest BSs b_x and b_y, where b_x and b_y belong to $\hat{\mathcal{C}}_i$ and $\hat{\mathcal{C}}_j$, respectively. Then, we combine two clusters $\hat{\mathcal{C}}_i$ and $\hat{\mathcal{C}}_j$ such that 1) neither $\hat{\mathcal{C}}_i$ nor $\hat{\mathcal{C}}_j$ has been combined with a cluster yet, 2) they have the minimum ICD, and 3) $\tilde{d}(\hat{\mathcal{C}}_i, \hat{\mathcal{C}}_j) \leq L_{\max}$. In this case, we generate a new cluster $\hat{\mathcal{C}}_k = \hat{\mathcal{C}}_i \cup \hat{\mathcal{C}}_j$ and assign a tree node v_k to $\hat{\mathcal{C}}_k$. Here, v_k will be the parent node of v_i and v_j (i.e., the tree nodes of $\hat{\mathcal{C}}_i$ and $\hat{\mathcal{C}}_j$).
3. Repeat step 2 until no two clusters can be combined.
4. The solution set contains the root node of each tree in the forest.

In step 2, when the ICD between two clusters exceeds $L_{\max}$, the cells in these two clusters are too far to impose significant CLI on each other. Fig. 3 gives an example with seven cells, where each cell c_l is assigned a tree node v_l ($l = 1..7$). For instance, the ICD between two clusters $\{c_1\}$ and $\{c_2\}$ is below $L_{\max}$, so they can be combined to a cluster $\{c_1, c_2\}$. In this case, we create a tree node v_8 for the new cluster, which is the parent node of both v_1 and v_2 (i.e., the tree nodes of clusters $\{c_1\}$ and $\{c_2\}$). From Fig. 3(b), two binary trees are formed, whose roots are v_{12} and v_7, so the solution set contains clusters $\{c_1, c_2, c_3, c_4, c_5, c_6\}$ and $\{c_7\}$. Theorem 1 analyzes the time complexity of the cell grouping module.

Theorem 1. *Given ζ_n cells, the worst-case time complexity of the cell grouping module is $O(\zeta_n^2)$.*

Proof: The cell grouping module uses the eAHC scheme to group cells, which forms a forest of binary trees, as shown in Fig. 3(b). Each leaf node of the trees corresponds to a cell. The worst case occurs when merely one binary tree is formed. In this case, all cells will be eventually grouped into a cluster (i.e., the same with the original AHC scheme). It has been shown in [30] that the time complexity of AHC is $O(\zeta_n^2)$ by using some optimizations. With these optimizations, the time complexity of the cell grouping module can be thus $O(\zeta_n^2)$. □

As we will carry out the subsequent modules on each cluster individually, our discussion below aims at one cluster, which is denoted by $\hat{\mathcal{C}}$.

4.2 Codebook Formulating Module

In cluster $\hat{\mathcal{C}}$, one cell acts as the master and other cells are slaves. The master works out a codebook of RFCs. Based on the amount of traffic demand, slaves choose RFCs from the codebook. Afterward, the master performs RFC coordination for slaves (the detail of coordination will be discussed in Section 4.4). Specifically, there are two issues to be addressed: 1) selecting the master and 2) formulating the codebook.

For the first issue, we select a cell c_k to be the master such that the sum of the distance between c_k's BS and the BS of each other cell in $\hat{\mathcal{C}}$ is the minimum. More concretely, let $\hat{\mathcal{B}}$ be the set of BSs located in $\hat{\mathcal{C}}$. The master selection procedure can be expressed by

$$b_k = \arg\min_{b_i \in \hat{\mathcal{B}}} \sum\nolimits_{b_j \in \hat{\mathcal{B}}, b_j \neq b_i} \tilde{d}(b_i, b_j), \qquad (12)$$

where $\tilde{d}(b_i, b_j)$ is the Euclidean distance between two BSs b_i and b_j. Then, the cell whose BS is b_k is the master. The idea behind Eq. (12) is that b_k would be close to each other BS in $\hat{\mathcal{B}}$. As a result, the transmission latency of messages (for RFC coordination) between BSs could decrease. Generally speaking, the master is located at the geometric center of all cells in $\hat{\mathcal{C}}$.

2. As mentioned in Section 2, the study [25] also formulates a codebook of RFCs for cells to select their subframes, but it does not consider that CLI's strength would be different with the distances between two BSs. Hence, some cells may still impose non-neglected CLI on each other. By applying the divide-and-conquer strategy, both cluster dividing and subcluster merging modules can implement the regional concept and thus deal with this issue.

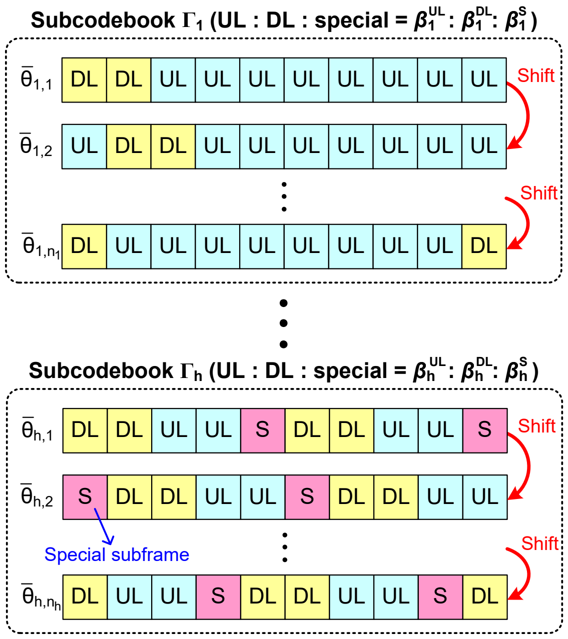

Subcodebook Γ1 (UL : DL : special = β1UL : β1DL : β1S)
θ̄1,1 DL DL UL UL UL UL UL UL UL UL
Shift
θ̄1,2 UL DL DL UL UL UL UL UL UL UL
Shift
θ̄1,n1 DL UL UL UL UL UL UL UL UL DL
Subcodebook Γh (UL : DL : special = βhUL : βhDL : βhS)
θ̄h,1 DL DL UL UL S DL DL UL UL S
Shift
θ̄h,2 S DL DL UL UL S DL DL UL UL
Special subframe
Shift
θ̄h,nh DL UL UL S DL DL UL UL S DL

Fig. 4: The concept of sliding codebook.

Hence, we find the center position of all BSs in $\hat{\mathcal{B}}$, which can be done by computing the average of the coordinates of these BSs. Then, c_k is the cell where the center position locates. In case that the center position is located in multiple cells, we pick the cell whose BS is the closest to the center position. Theorem 2 shows the time complexity of the master selection procedure.

Theorem 2. *Suppose that $\hat{\mathcal{C}}$ has ζ_c cells. The master selection procedure requires $O(\zeta_c)$ time.*

Proof: In this procedure, we find the master based on the center position of BSs in $\hat{\mathcal{B}}$. To do so, we calculate the average of their coordinates, which takes $O(|\hat{\mathcal{B}}|)$ time. Since each cell contains exactly one BS, we have $|\hat{\mathcal{B}}| = |\hat{\mathcal{C}}| = \zeta_c$. Hence, the time complexity is $O(\zeta_c)$. □

To address the second issue, we borrow the notion of sliding codebook in [25], as shown in Fig. 4. Let Γ be the codebook, which comprises h subcodebooks $\Gamma_1, \Gamma_2, \cdots$, and Γ_h. Each subcodebook Γ_x is dedicated to a different ratio $\beta_x^{\mathtt{UL}} : \beta_x^{\mathtt{DL}} : \beta_x^{\mathtt{S}}$ of UL, DL, and special subframes and contains n_x RFCs $\{\bar{\theta}_{x,1}, \bar{\theta}_{x,2}, \cdots, \bar{\theta}_{x,n_x}\}$. In other words, each RFC $\bar{\theta}_{x,y} \in \Gamma_x$ has $\beta_x^{\mathtt{UL}}$ UL subframes, $\beta_x^{\mathtt{DL}}$ DL subframes, and $\beta_x^{\mathtt{S}}$ special subframes. To compute the RFCs in Γ_x, the cyclic shift method is used. In particular, the subframe placement of an RFC $\bar{\theta}_{x,y}$ ($y = 2, \cdots, n_x$) is derived by shifting the subframe placement of its previous RFC $\bar{\theta}_{x,y-1}$ right by one subframe, as shown in Fig. 4. In this way, after deciding $\bar{\theta}_{x,1}$, all other RFCs in Γ_x can be simply calculated.

However, the study [25] neither discusses the relationship between parameters (specifically, m, n_x, $\beta_x^{\mathtt{UL}}$, $\beta_x^{\mathtt{DL}}$, and $\beta_x^{\mathtt{S}}$) nor gives their necessary constraints, which may lead to duplicate RFCs. Let us consider an example, where $m = 3$ (i.e., each RFC has three subframes), $n_x = 4$, and $\beta_x^{\mathtt{UL}} : \beta_x^{\mathtt{DL}} : \beta_x^{\mathtt{S}} = 1 : 2 : 0$. Suppose that the first RFC is [UL, DL, DL]. By the cyclic shifting method, the remaining three RFCs in a subcodebook will be [DL, UL, DL], [DL, DL, UL], and [UL, DL, DL]. Obviously, the last RFC is identical to the first RFC. In fact, no matter how we adjust subframes in this example, some RFCs will be inevitably duplicate.

TABLE 3: An instance of the sliding codebook with rules R1 and R2.

subcodebooks	$\beta_x^{\mathtt{UL}} : \beta_x^{\mathtt{DL}} : \beta_x^{\mathtt{S}}$	applicable cells
$\Gamma_1, \Gamma_2, \Gamma_3, \Gamma_4$	6:3:1, 7:2:1, 7:3:0, 8:2:0	UL-biased cells
$\Gamma_5, \Gamma_6, \Gamma_7, \Gamma_8$	2:8:0, 3:7:0, 2:7:1, 3:6:1	DL-biased cells
Γ_9, Γ_{10}	4:4:2, 5:5:0	neutral cells

To solve the above problem, we add two rules to ameliorate the sliding codebook:

R1. The number of RFCs in each subcodebook Γ_x (i.e., n_x) shall be set equal to the number of subframes in one RFC (i.e., m), that is,

$$n_x = m, \quad \forall \Gamma_x \text{ in } \Gamma. \tag{13}$$

R2. The subframes in each RFC are heterogeneous. To do so, one of the two conditions must hold for each subcodebook Γ_x: 1) $\min\{\beta_x^{\mathtt{UL}}, \beta_x^{\mathtt{DL}}\} \geq 1$ and 2) $1 \leq \beta_x^{\mathtt{S}} < m$. The first condition means that an RFC contains at least one UL subframe and at least one DL subframe, and the second condition implies that some (but not all) subframes of an RFC are special subframes.

Table 3 gives an instance of the sliding codebook by applying both rules. Codebook Γ contains ten subcodebooks (i.e., $h = 10$), where each subcodebook has ten RFCs (i.e., $n_x = 10$).

In Theorem 3, we show that each subcodebook Γ_x has no duplicate RFCs. Since Γ_x is dedicated to a unique ratio $\beta_x^{\mathtt{UL}} : \beta_x^{\mathtt{DL}} : \beta_x^{\mathtt{S}}$ of UL, DL, and special subframes, no two subcodebooks in Γ have the same numbers of UL, DL, and special subframes. As a result, each RFC in Γ will be unique by using rules R1 and R2.

Theorem 3. *By applying rules R1 and R2 to the cyclic shift method, we ensure that any two RFCs in a subcodebook Γ_x will be different.*

Proof: Let $\mathrm{P}_{\mathtt{RFC}}$ denote the set of all possible RFC permutations based on a subcodebook Γ_x's parameters (i.e., $\beta_x^{\mathtt{UL}}$ UL subframes, $\beta_x^{\mathtt{DL}}$ DL subframes, and $\beta_x^{\mathtt{S}}$ specifical subframes). One necessary condition to make any two RFCs in Γ_x be different is that the number of RFC permutations (i.e., $|\mathrm{P}_{\mathtt{RFC}}|$) shall be no fewer than the number of RFCs in Γ_x (i.e., n_x):

$$|\mathrm{P}_{\mathtt{RFC}}| = \frac{m!}{\beta_x^{\mathtt{UL}}!\beta_x^{\mathtt{DL}}!\beta_x^{\mathtt{S}}!} \geq n_x, \tag{14}$$

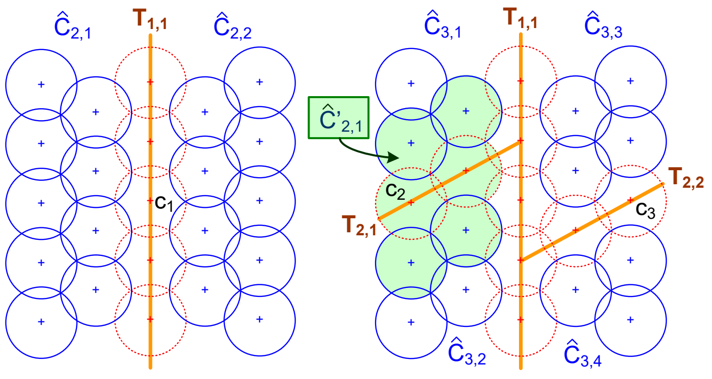
where an RFC has m subframes. Otherwise, no matter how we choose RFCs from $\mathrm{P}_{\mathtt{RFC}}$ for subcodebook Γ_x, some RFCs in Γ_x must be identical.

Due to rule R1, Γ_x contains m RFCs (i.e., $n_x = m$ by Eq. (13)). With rule R2, the subframes of each RFC are heterogeneous. The worst case (i.e., the minimum value of $|\mathrm{P}_{\mathtt{RFC}}|$) occurs when there exists only one heterogeneous subframe in an RFC. In this case, we have $|\mathrm{P}_{\mathtt{RFC}}| \geq \frac{m!}{0!1!(m-1)!} = m$. Therefore, the condition of Eq. (14) must hold.

Given an RFC $\bar{\theta}_{x,1}$, the cyclic shift method generates the remaining $(m-1)$ RFCs by iteratively shifting one subframe. This is equivalent to picking m RFCs from $\mathrm{P}_{\mathtt{RFC}}$ for Γ_x. Since $\mathrm{P}_{\mathtt{RFC}}$ has m or more RFC permutations, these m RFCs in Γ_x must be different, thereby proving this theorem. □

4.3 Cluster Dividing Module

This module recursively divides cluster $\hat{\mathcal{C}}$ into subclusters, until each subcluster contains no more than φ cells (below, we call it a *basic subcluster*). Hence, we can carry out the subframe deciding module in Section 4.4 on each basic subcluster individually, thereby facilitating RFC coordination for cells.



(a) iteration 1 (b) iteration 2

Fig. 5: Dividing a cluster into subclusters recursively.

Our idea is to iteratively find a dividing line to partition each subcluster into two smaller subclusters that contain a similar number of cells. For convenience, we denote by $\hat{\mathcal{C}}_{i,j}$ the j-th subcluster to be divided in iteration i. Besides, the dividing line for $\hat{\mathcal{C}}_{i,j}$ is denoted by $T_{i,j}$. Then, the cluster dividing module is composed of three steps:

1. Initially, cluster $\hat{\mathcal{C}}$ is viewed as a subcluster $\hat{\mathcal{C}}_{1,1}$.
2. In a subcluster $\hat{\mathcal{C}}_{i,j}$, we find its *central cell* c_k. Let $\hat{\mathcal{N}}_k$ be the set of c_k's adjacent cells. For each cell $c_a \in \hat{\mathcal{N}}_k$, we draw a line to link the BSs of c_k and c_a and extend this line to cut $\hat{\mathcal{C}}_{i,j}$ into two halves (i.e., $T_{i,j}$). In this way, we can obtain two smaller subclusters $\hat{\mathcal{C}}_{i+1,2(j-1)+1}$ and $\hat{\mathcal{C}}_{i+1,2(j-1)+2}$. Note that neither $\hat{\mathcal{C}}_{i+1,2(j-1)+1}$ nor $\hat{\mathcal{C}}_{i+1,2(j-1)+2}$ contains the cells located on the dividing line $T_{i,j}$. Among all division for $\hat{\mathcal{C}}_{i,j}$, we select the one such that the difference between the numbers of cells in both $\hat{\mathcal{C}}_{i+1,2(j-1)+1}$ and $\hat{\mathcal{C}}_{i+1,2(j-1)+2}$ is the minimum.
3. Repeat step 2 until each subcluster has no more than φ cells (that is, all subclusters are basic subclusters).

In step 2, the central cell of subcluster $\hat{\mathcal{C}}_{i,j}$ is the cell located at the geometric center of all cells in $\hat{\mathcal{C}}_{i,j}$. We can use the master selection procedure in Section 4.2 to find the central cell.

Fig. 5 gives an example, where $\varphi = 4$. In iteration 1, we find the central cell c_1 of subcluster $\hat{\mathcal{C}}_{1,1}$ and decide its dividing line $T_{1,1}$. Thus, subclusters $\hat{\mathcal{C}}_{2,1}$ and $\hat{\mathcal{C}}_{2,2}$ have 9 cells (excluding th cells on $T_{1,1}$). Then, we find the dividing lines $T_{2,1}$ and $T_{2,2}$ to split $\hat{\mathcal{C}}_{2,1}$ and $\hat{\mathcal{C}}_{2,2}$, respectively. As none of subclusters $\hat{\mathcal{C}}_{3,1}$, $\hat{\mathcal{C}}_{3,2}$, $\hat{\mathcal{C}}_{3,3}$, and $\hat{\mathcal{C}}_{3,4}$ has more than 4 cells, the cluster dividing module finishes, and the final result is shown in Fig. 5(b).

Let us discuss the range of φ's value. As mentioned later in Section 4.4, the subframe deciding module finds an anchor RFC for the cells in a basic subcluster to be the reference, which depends on the majority choice of subcodebook type by these cells. Since there are three types of subcodebooks (i.e, UL-biased, DL-biased, and neutral), the minimum number of cells required to determine the majority in a basic subcluster is obviously 4. Thus, the lower bound of φ is 4. On the other hand, the cluster dividing module seeks to partition a subcluster into two "equal" smaller subclusters. Moreover, each subcluster will not include those cells located on the dividing line. Evidently, there are at least two cells on a dividing line (i.e., c_k and c_a). To let cluster $\hat{\mathcal{C}}$ be at least divided into two subclusters, the upper bound of φ is set to $\frac{|\hat{\mathcal{C}}|-2}{2}$. To sum up, we can obtain that $4 \leq \varphi \leq \frac{|\hat{\mathcal{C}}|-2}{2}$. Theorem 4 analyzes the time complexity of the cluster dividing module.

Theorem 4. *Given ζ_c cells in $\hat{\mathcal{C}}$, the cluster dividing module spends $O(\zeta_c \log_2 \frac{\zeta_c}{\varphi})$ time in the worst case.*

Proof: As the cluster dividing module recursively cuts $\hat{\mathcal{C}}$ into two halves, until each subcluster has at most φ cells, the number of iterations is $\log_2 \zeta_c - \log_2 \varphi = \log_2 \frac{\zeta_c}{\varphi}$. In iteration 1, it takes $O(\zeta_c)$ time to find the central cell in $\hat{\mathcal{C}}$ (i.e., by Theorem 2) and $O(\min\{\zeta_c, \zeta_N\})$ time to find dividing lines, where ζ_N is the maximum number of neighbors of each cell. Thus, iteration 1 spends time of $O(\zeta_c) + O(\min\{\zeta_c, \zeta_N\}) = O(\zeta_c)$. In iteration 2, there are two subclusters, each with no more than $\frac{\zeta_c}{2}$ cells. In each subcluster, it takes $O(\frac{\zeta_c}{2})$ time to find the central cell and $O(\min\{\frac{\zeta_c}{2}, \zeta_N\})$ time to decide the dividing line. Thus, iteration 2 requires time of $2(O(\frac{\zeta_c}{2}) + O(\min\{\frac{\zeta_c}{2}, \zeta_N\})) = O(\zeta_c)$. Similarly, each of the remaining iterations spends $O(\zeta_c)$ time. Hence, the total time complexity is $\log_2 \frac{\zeta_c}{\varphi} \times O(\zeta_c) = O(\zeta_c \log_2 \frac{\zeta_c}{\varphi})$. □

4.4 Subframe Deciding Module

After obtaining codebook Γ by the module in Section 4.2, the master can disseminate Γ to other cells in $\hat{\mathcal{C}}$. Then, each cell $c_i \in \hat{\mathcal{C}}$ picks a subcodebook Γ_x from Γ based on its type and the amount of UL and DL demand (i.e., D_{UL} and D_{DL}). More concretely, there are three cases for c_i to choose Γ_x:

A1. When c_i is a UL-biased cell (i.e., $D_{\text{UL}} > \alpha D_{\text{DL}}$), it selects a *UL-biased subcodebook* Γ_x by

$$\Gamma_x = \arg \min_{\Gamma_x \in \Gamma_{\text{UB}}} \left| \frac{D_{\text{UL}}}{D_{\text{UL}} + D_{\text{DL}}} - \frac{\beta_x^{\text{UL}}}{\beta_x^{\text{UL}} + \beta_x^{\text{DL}}} \right|, \tag{15}$$

where Γ_{UB} is a subset of Γ whose RFCs are applicable to UL-biased cells (i.e., Γ_1–Γ_4 in Table 3). The idea behind Eq. (15) is to find a subcodebook Γ_x whose $\beta_x^{\text{UL}} : \beta_x^{\text{DL}}$ ratio is the closest to the $D_{\text{UL}} : D_{\text{DL}}$ ratio. In other words, Γ_x is the fittest for the need of c_i's traffic demand.

A2. If c_i is a DL-biased cell (that is, $D_{\text{DL}} > \alpha D_{\text{UL}}$), it chooses a *DL-biased subcodebook* Γ_x by

$$\Gamma_x = \arg \min_{\Gamma_x \in \Gamma_{\text{DB}}} \left| \frac{D_{\text{DL}}}{D_{\text{UL}} + D_{\text{DL}}} - \frac{\beta_x^{\text{DL}}}{\beta_x^{\text{UL}} + \beta_x^{\text{DL}}} \right|, \tag{16}$$

where Γ_{DB} is a subset of Γ whose RFCs are applicable to DL-biased cells (i.e., Γ_5–Γ_8 in Table 3).

A3. Otherwise, c_i is a neutral cell, so it picks a *neutral subcodebook* Γ_x (i.e., Γ_9 and Γ_{10} in Table 3) such that c_i's traffic demand can be met. If there are multiple choices, we check whether there is a need for special subframes (e.g., Γ_{10}).

Through cases A1–A3, each cell can select a subcodebook based on its traffic demand. Then, we need to coordinate the RFCs of cells (which would be picked from their selected subcodebooks) to minimize subframe misalignments and reduce the effect of CLI. Specifically, we are given a subcluster $\hat{\mathcal{C}}_{i,j}$ to perform RFC coordination. Let $\widetilde{\Gamma}$ be the collection of subcodebooks selected by the cells in $\hat{\mathcal{C}}_{i,j}$. If the number of cells in $\hat{\mathcal{C}}_{i,j}$ that choose a subcodebook Γ_x is the maximum and exceeds one, we say that Γ_x is a *majority choice of subcodebook (MCS)* in $\widetilde{\Gamma}$. For example, suppose that $\hat{\mathcal{C}}_{i,j}$ has four cells c_1, c_2, c_3, and c_4, which select subcodebooks Γ_1, Γ_1, Γ_5, and Γ_9, respectively. Then, Γ_1 is an MCS. On the other hand, when the number of cells in $\hat{\mathcal{C}}_{i,j}$ that choose a type Υ of subcodebook is the maximum and overtakes one, where Υ can be UL-biased (i.e., Γ_1–Γ_4), DL-biased (i.e., Γ_5–Γ_8), or neutral (i.e., Γ_9–Γ_{10}), Υ is said to be a *majority choice of type (MCT)* in $\widetilde{\Gamma}$. For example, suppose that cells c_1, c_2, c_3, and c_4 choose subcodebooks Γ_1, Γ_2, Γ_3, and Γ_8, respectively. Then, the MCT will be UL-biased (as Γ_1, Γ_2, and Γ_3 are UL-biased subcodebooks).

To facilitate the coordination of RFCs, we pick an *anchor RFC* $\bar{\theta}_{\texttt{A}}$ to be the reference for the cells in $\hat{\mathcal{C}}_{i,j}$. In particular, there are three cases used to find $\bar{\theta}_{\texttt{A}}$:

B1. When there is an MCS in $\widetilde{\Gamma}$, say, Γ_x, we pick an RFC from Γ_x to be $\bar{\theta}_{\texttt{A}}$. In case of a tie, we arbitrarily select one subcodebook from these MCSs.

B2. There is no MCS in $\widetilde{\Gamma}$, but there exists an MCT Υ. Let $\widetilde{\Gamma}'$ be a subset of $\widetilde{\Gamma}$ whose subcodebooks belong to type Υ. Then, we randomly select a subcodebook Γ_x from $\widetilde{\Gamma}'$ and pick one RFC from Γ_x to be $\bar{\theta}_{\texttt{A}}$.

B3. Otherwise, there is neither MCS nor MCT in $\widetilde{\Gamma}$. This case occurs only when $\hat{\mathcal{C}}_{i,j}$ contains three or fewer cells. Hence, we arbitrarily pick a subcodebook from $\widetilde{\Gamma}$ and select one RFC from that subcodebook to be $\bar{\theta}_{\texttt{A}}$.

Afterward, for each cell $c_k \in \hat{\mathcal{C}}_{i,j}$, we decide its RFC by the following method. Suppose that c_k chooses a subcodebook Γ_x. Then, we find an RFC $\bar{\theta}_l$ from Γ_x such that the number of subframe misalignments between $\bar{\theta}_l$ and the anchor RFC $\bar{\theta}_{\texttt{A}}$ (as denoted by $\vartheta_{l,\texttt{A}}$) is the minimum. Here, the number of subframe misalignments can be calculated by Eq. (8). However, if $\vartheta_{l,\texttt{A}} > \delta_{\texttt{SDM}}$, where $\delta_{\texttt{SDM}} \in \mathbb{Z}^+$ is a predefined threshold, it implies that there is no suitable RFC in Γ_x for c_k to avoid CLI (since the number of subframe misalignments is too large). In this case, c_k will select another subcodebook $\Gamma_{x'}$ such that Γ_x and $\Gamma_{x'}$ have the same type, and pick one RFC from $\Gamma_{x'}$ with the minimum $\vartheta_{l,\texttt{A}}$ value and $\vartheta_{l,\texttt{A}} \leq \delta_{\texttt{SDM}}$. Nevertheless, if all subcodebooks (of the same type with Γ_x) have been checked, but no RFC can fulfil the requirement of $\vartheta_{l,\texttt{A}} \leq \delta_{\texttt{SDM}}$, then among all RFCs ever checked, we choose the one with the minimum $\vartheta_{l,\texttt{A}}$ value for c_k. Theorem 5 analyzes the time complexity of the subframe deciding module.

Theorem 5. *Suppose that a subcluster $\hat{\mathcal{C}}_{i,j}$ contains ζ_s cells and the maximum number of subcodebooks of the same type in Γ is ζ_t. Then, it takes $O(\zeta_t(\zeta_s + m))$ time for the subframe deciding module to handle $\hat{\mathcal{C}}_{i,j}$ in the worst case.*

Proof: The subframe deciding module has three parts. In the first part, we find a subcodebook from Γ for each cell in $\hat{\mathcal{C}}_{i,j}$, which consumes time of $\zeta_s \times O(\zeta_t) = O(\zeta_s\zeta_t)$. Finding the anchor RFC involves in checking a collection $\widetilde{\Gamma}$ of subcodebooks chosen by all cells in $\hat{\mathcal{C}}_{i,j}$. This requires $O(\zeta_s)$ time. In the third part, we decide the RFC for each cell in $\hat{\mathcal{C}}_{i,j}$. The worst case occurs when every RFC in all subcodebooks of the same type in Γ needs to be checked. Based on rule R1 in Section 4.2, each subcodebook has m RFCs. Thus, this part takes time of $\zeta_t \times O(m) = O(\zeta_t m)$. To sum up, the total time complexity is $O(\zeta_s\zeta_t + \zeta_s + \zeta_t m) = O(\zeta_t(\zeta_s + m))$. □

4.5 Subcluster Merging Module

This module is inverse to the cluster dividing module in Section 4.3, which merges two adjacent subclusters recursively, until all subclusters are merged together to revert to cluster $\hat{\mathcal{C}}$. In Section 4.3, a subcluster $\hat{\mathcal{C}}_{i,j}$ is split into two smaller subclusters $\hat{\mathcal{C}}_{i+1,2(j-1)+1}$ and $\hat{\mathcal{C}}_{i+1,2(j-1)+2}$ by a dividing line $T_{i,j}$. Inversely, to merge $\hat{\mathcal{C}}_{i+1,2(j-1)+1}$ and $\hat{\mathcal{C}}_{i+1,2(j-1)+2}$ (i.e., reverting to $\hat{\mathcal{C}}_{i,j}$), we perform RFC coordination for some cells in both $\hat{\mathcal{C}}_{i+1,2(j-1)+1}$ and $\hat{\mathcal{C}}_{i+1,2(j-1)+2}$ and those cells located on $T_{i,j}$. More concretely, let $\hat{\mathcal{C}}_{\tau_1}$ be the set of cells on $T_{i,j}$. Besides, $\hat{\mathcal{C}}_{\tau_2}$ and $\hat{\mathcal{C}}_{\tau_3}$ are the subsets of cells in $\hat{\mathcal{C}}_{i+1,2(j-1)+1}$ and $\hat{\mathcal{C}}_{i+1,2(j-1)+2}$ such that these cells are the neighbors of those cells in $\hat{\mathcal{C}}_{\tau_1}$, respectively. Then, we create a set $\hat{\mathcal{C}}'_{i,j} = \hat{\mathcal{C}}_{\tau_1} \cup \hat{\mathcal{C}}_{\tau_2} \cup \hat{\mathcal{C}}_{\tau_3}$. Fig. 5(b) shows an example, where a set $\hat{\mathcal{C}}'_{2,1}$ is used to merge subclusters $\hat{\mathcal{C}}_{3,1}$ and $\hat{\mathcal{C}}_{3,2}$.

By using the RFC coordination method in Section 4.4, we adjust the subframes of cells in $\hat{\mathcal{C}}'_{i,j}$ to mitigate CLI. This can be done by finding an anchor RFC and tuning the subframes of each cell in $\hat{\mathcal{C}}'_{i,j}$ to reduce subframe misalignments. Here, we adopt a more relaxed threshold $\delta_{\texttt{SMM}}$, where the value of $\delta_{\texttt{SMM}}$ is slightly larger than $\delta_{\texttt{SDM}}$ (e.g., $\delta_{\texttt{SMM}} = \delta_{\texttt{SDM}} + 1$). The reason is that $\hat{\mathcal{C}}'_{i,j}$ contains some cells of two subclusters $\hat{\mathcal{C}}_{i+1,2(j-1)+1}$ and $\hat{\mathcal{C}}_{i+1,2(j-1)+2}$, where their RFC arrangement may be quite different. Hence, we employ a more relaxed threshold $\delta_{\texttt{SMM}}$ to facilitate the combination of these two subclusters.

4.6 Rationale

Let us discuss the rationale of the CRCC framework. By using the eAHC scheme, CRCC partitions cells into clusters to make any two clusters far enough apart, so the amount of CLI caused by their cells can be neglected. Hence, we can focus on handling CLI elimination in a cluster $\hat{\mathcal{C}}$ and reduce superfluous computation. Then, CRCC employs the sliding codebook for each cell in $\hat{\mathcal{C}}$ to choose a suitable RFC. As compared with the original sliding codebook in [25], CRCC adds two significant improvements. First, some RFCs in a codebook Γ generated by [25] could be duplicate. This obviously wastes the codebook's space. Even worse, some cells with different conditions might select duplicate RFCs, thereby degrading system performance. Hence, we propose rules R1 and R2 in the codebook formulating module to ensure that every RFC in Γ is unique. In this way, we can not only find a compact codebook for the master to save the message cost when disseminating the codebook to slaves, but also help cells choose appropriate RFCs.

Second, unlike the study [25] that directly coordinates the RFCs of all cells in $\hat{\mathcal{C}}$, CRCC employs the divide-and-conquer strategy. In particular, the cluster dividing module splits $\hat{\mathcal{C}}$ into basic subclusters. Then, the subframe deciding module transacts RFC coordination in each basic subcluster. Finally, the subcluster merging module combines subclusters by adjusting the subframes of those cells located on subcluster boundaries. Doing so brings three benefits:

- As each basic subcluster has no more than φ cells, it is easy to perform RFC coordination for these cells.
- Since a basic subcluster contains cells in a small region, the coordination of RFCs by the subframe deciding module can better reflect the need and difference of cells in each small region.
- By carefully adjusting the subframes of those cells located on subcluster boundaries, we can mitigate CLI caused by the division of subclusters.

The above designs distinguish our CRCC framework from the existing solution [25] and mitigate CLI more efficiently, thereby improving the performance of a 5G D-TDD system.

5 Performance Evaluation

We build our simulation by MATLAB for performance evaluation, where Table 4 lists the simulation parameters. Following

TABLE 4: Simulation parameters.

BS-related parameters:	
type and number	23 5G NR macrocells, topology: Fig. 5(a)
transmitted power	46 dBm (i.e., 40 watt)
cell range	500 meters
channel bandwidth	10 MHz
antenna (A_{B})	8
modulation	QPSK, 16QAM, 64QAM
TTI	1 ms (with 14 OFDM symbols)
UE-related parameters:	
number	230 (around 10 UEs per cell)
distribution	uniform
transmitted power	23 dBm (i.e., 1 watt)
antenna (A_{U})	2
traffic demand	[UL-biased cell] D_{UL} = 20 Mbps, D_{DL} = 10 Mbps
	[DL-biased cell] D_{UL} = 10 Mbps, D_{DL} = 20 Mbps
	[neutral cell] D_{UL} = 15 Mbps, D_{DL} = 15 Mbps
Channel-related parameters:	
propagation loss	urban macrocell model
path loss	referring to Eq. (17)
shadowing fading	zero-mean log-normal distribution
multipath fading	Rayleigh fading model
AWGN (σ)	spectral density: -174dBm/Hz

the 3GPP specification (for 5G NR) [31], the amount of path loss for a UE u_i is estimated by

$$\text{PL} = \begin{cases} \text{PL}_1 & 10\,\text{m} \le d_{\text{2D}} < d_{\text{BP}} \\ \text{PL}_2 & d_{\text{BP}} \le d_{\text{2D}} \le 10\,\text{km} \end{cases} \tag{17}$$

where d_{2D} denotes the Euclidean distance between u_i and the BS. Besides, d_{BP} is the distance of break point, which is estimated by

$$d_{\text{BP}} = (2\pi h_{\text{BS}} h_{u_i} f_o)/\tilde{c}, \tag{18}$$

where h_{BS} and h_{u_i} are the heights of the BS and u_i (in meter), respectively, f_o signifies the operating frequency (in Hz), and $\tilde{c}$ is the light speed ($\tilde{c} \approx 3 \times 10^8$ m/s). Given the distance from the top of BS's antenna to the top of u_i's antenna (as denoted by d_{3D}) and the average building height h, both PL_1 and PL_2 can be calculated as follows:

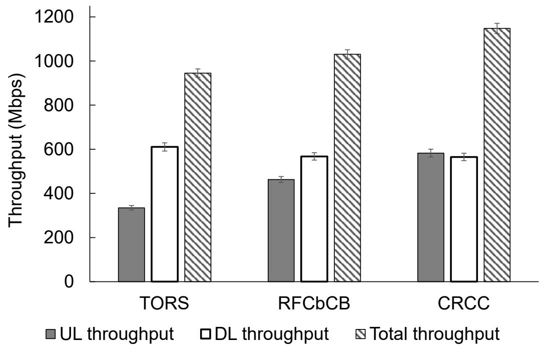
$$\text{PL}_1 = 20\log_{10}\left(\frac{40\pi d_{\text{3D}} f_o}{3}\right) + \min\{0.03h^{1.72}, 10\}\log_{10}(d_{\text{3D}}) - \min\{0.044h^{1.72}, 14.77\} + 0.002\log_{10}(h)d_{\text{3D}}, \tag{19}$$

$$\text{PL}_2 = \text{PL}_1(d_{\text{BP}}) + 40\log_{10}(d_{\text{3D}}/d_{\text{BP}}). \tag{20}$$

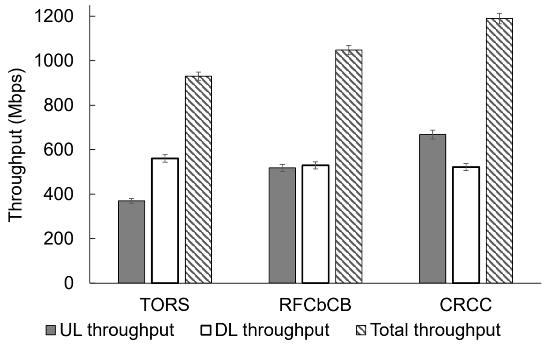
The simulation takes account of two scenarios for traffic demand. In scenario 1, we consider a general situation, where the numbers of UL-biased, DL-biased, and neutral cells are nearly equal. On the other hand, scenario 2 considers a UL-dominating situation, where the UL-biased, DL-biased, and neutral cells occupy 50%, 25%, and 25% of total cells in the service area, respectively. Scenario 2 could be used in user-centric vehicle-to-infrastructure communication applications, where many cars send their information to BSs for cooperative intelligent transportation service [32], or Internet-of-Things applications, where devices collect environmental data and report their sensing data to BSs [33].

We compare our CRCC framework with two methods:

- Traffic-oriented RFC selection (TORS): Each cell c_i picks an RFC fit for its UL and DL demand. More concretely, c_i uses cases A1, A2, or A3 mentioned in Section 4.4 to find a subcodebook if c_i is UL-biased, DL-biased, or neutral. Then, c_i randomly picks an RFC from the selected subcodebook to arrange its subframes.
- RFC-based sliding codebook (RFCbCB) [25]: Each cell c_i also picks a subcodebook Γ_x and selects an RFC $\bar{\theta}_j$ from Γ_x. Based on a common RFC $\bar{\theta}_{\text{O}}$, if the number of subframe misalignments between $\bar{\theta}_j$ and $\bar{\theta}_{\text{O}}$ is below a threshold[3], c_i can use RFC $\bar{\theta}_j$. Otherwise, c_i finds an RFC $\bar{\theta}_k$ from Γ_x such that the number of subframe misalignments between $\bar{\theta}_k$ and $\bar{\theta}_{\text{O}}$ is the smallest.



(a) scenario 1 (general situation)



(b) scenario 2 (UL-dominating situation)

Fig. 6: Comparison of throughput.

To make a fair comparison, TORS, RFCbCB, and CRCC all employ the same codebook in Table 3 for cells to select RFCs. In CRCC, we set $\delta_{\text{SDM}} = 2$ and $\delta_{\text{SMM}} = 3$.

Regarding performance metrics, we adopt throughput, UL and DL SINRs, and CLI. As mentioned in Section 3, Eqs. (6) and (7) give the amount of UL and DL throughput of a cell, and Eq. (10) calculates the overall throughput of cells in the service area. Besides, Eq. (2) estimates the SINR of a UE for DL communication, while Eq. (5) figures out the SINR of BS for UL communication. On the other hand, the calculation in Eqs. (1) and (4) take account of U2U and B2B CLI, respectively. Below, for each experiment, we repeat the simulation 100 times (each with a different random seed) and take their average.

5.1 Comparison of Throughput

Fig. 6(a) shows the amount of UL, DL, and total throughput in scenario 1, where the numbers of UL-biased, DL-biased, and neutral cells are similar. The TORS method makes each cell pick RFC based on its traffic demand, without considering the effect of CLI. As mentioned in Section 1, due to power imbalance between the BS and UEs, doing so could benefit DL performance but greatly degrade UL performance. That explains why TORS has the lowest UL throughput and the highest DL throughput. However, the loss of UL performance caused by CLI cannot be compensated by the improvement of

3. According to [25], the threshold is set to 3 subframes.

TABLE 5: Comparison of UL CQI, where SINR is the average

(a) Scenario 1 (general situation)

method	SINR	CQI	bits/symbol
TORS	0.26 dB	5	0.8770
RFCbCB	6.78 dB	8	1.9141
CRCC	8.93 dB	9	2.4063

(b) Scenario 2 (UL-dominating situation)

method	SINR	CQI	bits/symbol
TORS	2.45 dB	6	1.1758
RFCbCB	8.60 dB	9	2.4063
CRCC	12.91 dB	11	3.3223

DL performance. Hence, TORS has the lowest total throughput among all methods.

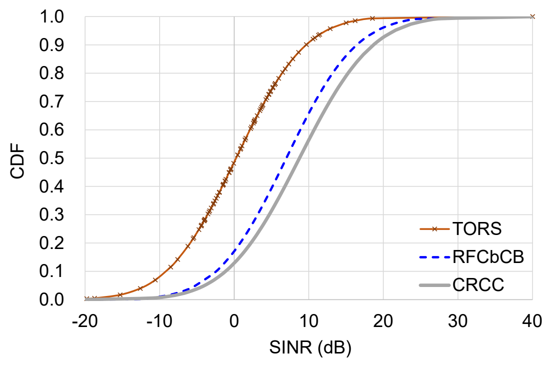
The RFCbCB method uses a common RFC for adjusting the RFCs of cells to diminish subframe misalignments. Though DL throughput decreases slightly (as compared with TORS), UL throughput can improve significantly. On the other hand, our CRCC framework divides cells into basic subclusters and then performs RFC coordination in each basic subcluster. In this way, CRCC can better reflect the requirement and dissimilitude of cells in different small regions. As compared with RFCbCB, CRCC further improves UL throughput almost without decreasing DL throughput, thereby raising the total throughput.

Based on Fig. 6(a), CRCC improves 74.16% and 25.91% of UL throughput but loses merely 7.43% and 0.44% of DL throughput than TORS and RFCbCB, respectively. Moreover, CRCC increases 21.46% and 11.4% of total throughput, as compared with TORS and RFCbCB, respectively. The result shows that our CRCC framework outperforms other methods in the general situation (i.e., scenario 1).

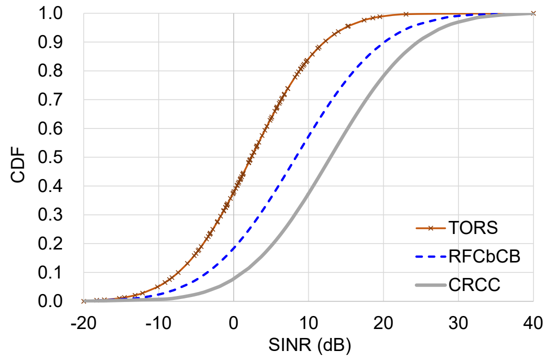
Fig. 6(b) presents the amount of UL, DL, and total throughput in scenario 2. As can be seen, the trend is similar to that in Fig. 6(a). Since UL-biased cells are more than DL-biased cells, each method has higher UL throughput and lower DL throughput, as compared with Fig. 6(a). In particular, CRCC raises 80.51% and 28.87% of UL throughput but reduces only 6.93% and 1.49% of DL throughput than TORS and RFCbCB, respectively. Moreover, CRCC improves 27.84% and 13.53% of total throughput, as compared with TORS and RFCbCB, respectively. This result displays the superiority of the CRCC framework over others in scenario 2 (i.e., the UL-dominating situation).

5.2 Comparison of UL SINR

The difference in throughput performance of each method mainly comes from the difference in channel quality of UEs. Therefore, we further investigate the SINRs of UEs. Fig. 7(a) shows the *cumulative distribution function (CDF)* of SINRs of UL UEs in scenario 1. Generally speaking, if the CDF line is closer to the right, it means that most UEs have higher SINRs (i.e., better channel quality). Since the TORS method does not consider adjusting subframes to mitigate CLI, UL UEs will be significantly affected by B2B interference from DL cells. That is why the CDF line of TORS is on the far left (that is, many UL UEs have relatively bad channel quality). Both RFCbCB and CRCC perform RFC coordination to diminish subframe misalignments, so their CDF lines will shift to the right. By applying the divide-and-conquer strategy to subframe adjustment, our CRCC framework can substantially improve the channel quality of UL UEs, so its CDF line will be further to the right than the RFCbCB method.



(a) scenario 1 (general situation)



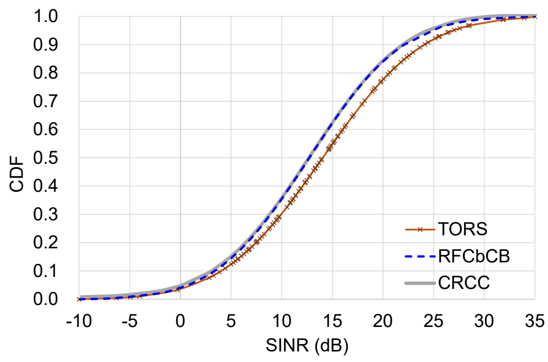
(b) scenario 2 (UL-dominating situation)

Fig. 7: Comparison of the CDF of UL SINR.

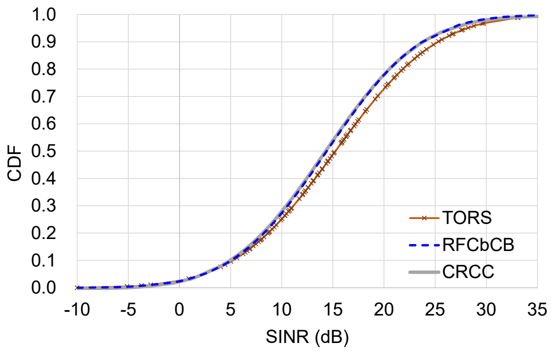
Fig. 7(b) gives the CDF of SINRs of UL UEs in scenario 2. Since the number of DL-biased cells decreases (as compared with scenario 1), the amount of B2B interference from DL cells reduces accordingly. In this case, the channel quality of UL UEs could improve. Hence, the CDF line of each method shifts to the right, as compared with Fig. 7(a). Obviously, the gap between the CDF lines of RFCbCB and CRCC becomes larger. This phenomenon reveals that CRCC performs much better (in terms of the improvement of UL performance) in scenario 2, where UL-biased cells account for the majority.

By taking the data of the middle 80% UL UEs, Table 5(a) shows the average SINR, *channel quality index (CQI)*, and the average number of bits carried by OFDM symbols in scenario 1. For TORS, the UL SINR range is [-5.52, 9.67] dB, so the average SINR is 0.26 dB and the corresponding CQI is 5. In this case, each OFDM symbol can send only 0.8770 bits. For RFCbCB, the UL SINR range is [-8.52, 16.18] dB, so the average SINR is 6.78 dB and the CQI increases to 8. Hence, each OFDM symbol can transmit 1.9141 bits. On the other hand, the UL SINR range of our CRCC framework is [-0.57, 18.85] dB. The average SINR can be improved to 8.93 dB and the CQI rises to 9. Since more complex modulation and higher code rates will be used, each OFDM symbol can carry up to 2.4063 bits. That explains why CRCC greatly improves UL throughput.

Table 5(b) gives the statistics of UL UEs in scenario 2. For TORS, the UL SINR range is [-7.40, 12.28] dB, so the average SINR is 2.45 dB and the CQI is 6. Each OFDM symbol can send 1.1758 bits. For RFCbCB, the UL SINR range is [-3.34, 20.26] dB, so the average SINR is 8.60 dB and the CQI is 9. An OFDM symbol can transmit 2.4063 bits. The UL SINR range of CRCC is [1.01, 23.79] dB. The average SINR is 12.91 dB and the CQI increases to 11. In this case, each OFDM symbol can carry



(a) scenario 1 (general situation)



(b) scenario 2 (UL-dominating situation)

Fig. 8: Comparison of the CDF of DL SINR, where the CDF lines of RFCbCB and CRCC overlap.

TABLE 6: Comparison of DL CQI, where SINR is the average

(a) Scenario 1 (general situation)

method	SINR	CQI	bits/symbol
TORS	14.24 dB	12	3.9023
RFCbCB	12.72 dB	11	3.3223
CRCC	12.71 dB	11	3.3223

(b) Scenario 2 (UL-dominating situation)

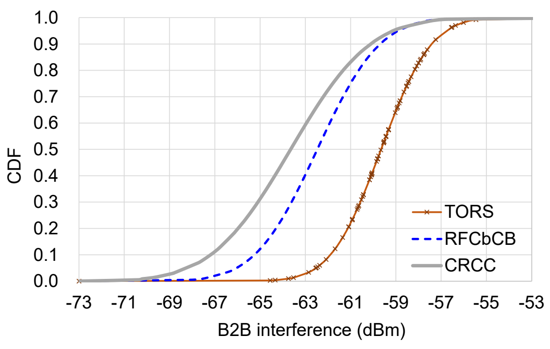
method	SINR	CQI	bits/symbol
TORS	15.23 dB	12	3.9023
RFCbCB	14.39 dB	12	3.9023
CRCC	14.27 dB	12	3.9023

3.3223 bits, thereby increasing UL performance.

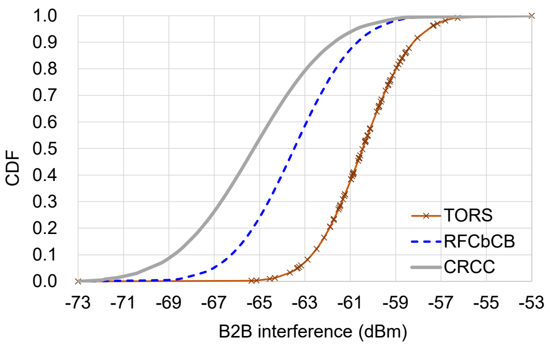
5.3 Comparison of DL SINR

Fig. 8(a) and (b) present the CDFs of SINRs of DL UEs in scenarios 1 and 2, respectively. Evidently, the CDF line of TORS is on the far right, which means that DL UEs have better channel quality. However, the gap between the CDF lines of TORS and other two methods is small. This appearance proves that RFCbCB and CRCC just slightly degrade DL performance, as compared with TORS. The result together with the discussion in Section 5.2 demonstrate that our CRCC framework can efficiently and significantly improve UL throughput by sacrificing just a little DL throughput.

Table 6(a) gives the average SINR, CQI, and the average number of bits sent by OFDM symbols in scenario 1, where we take the data of the middle 80% DL UEs. For TORS, the DL SINR range is [4.41, 24.14] dB. The average SINR is 14.24 dB and the CQI is 12. Each OFDM symbol can transmit 3.9023 bits. For RFCbCB, the DL SINR range is [3.24, 22.01] dB. The average SINR is 12.72 dB and the CQI decreases to 11. Thus, each OFDM symbol can send 3.3223 bits. The DL SINR range



(a) scenario 1 (general situation)



(b) scenario 2 (UL-dominating situation)

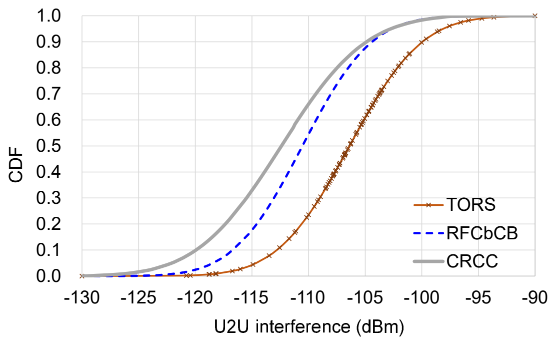
Fig. 9: Comparison of the CDF of B2B interference.

of CRCC is [3.20, 22.04] dB. The average SINR is 12.71 dB and the CQI is 11. Similarly, an OFDM symbol carries 3.3223 bits. On the other hand, Table 6(b) shows the statistics of DL UEs in scenario 2. Regarding TORS, the DL SINR range is [5.61, 25.33] dB. The average SINR is 15.23 dB and the CQI is 12. Each OFDM symbol carries 3.9023 bits. For RFCbCB, the DL SINR range is [4.91, 23.68] dB. The average SINR is 14.39 dB and the CQI is 12. An OFDM symbol can also send 3.9023 bits. In CRCC, the DL SINR range is [4.87, 23.54] dB. Hence, the average SINR is 14.27 dB, the CQI is 12, and each OFDM symbol carries 3.9023 bits. As can be seen, the DL performance loss of both RFCbCB and CRCC (as compared with TORS) is not significant, especially in scenario 2.

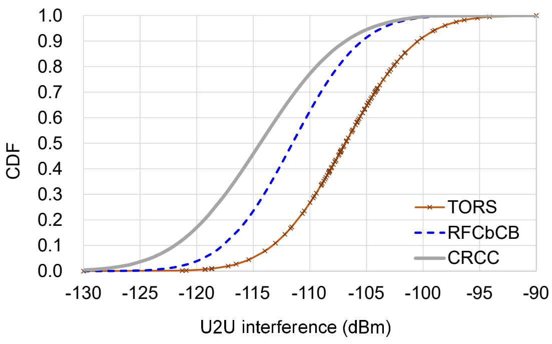
5.4 Comparison of CLI

Then, we evaluate the strength of CLI. Recall that there are two types of CLI, namely B2B and U2U interference, as shown in Fig. 1(c). B2B interference occurs when the BS of a DL cell interferes with the BS of a UL cell receiving data. On the other hand, U2U interference occurs when a UE in one UL cell disturbs another UE in a DL cell receiving data.

Fig. 9(a) compares the CDF of B2B interference strength in scenario 1. When the CDF line is closer to the right, it implies that B2B interference becomes more severe. Since TORS lets cells choose their preferred RFCs forthright, TORS will inevitably incur the most severe B2B interference. RFCbCB uses a common RFC to align the subframes of each cell, which helps mitigate B2B interference. CRCC partitions cells into subclusters and performs RFC coordination in each subcluster to minimize subframe misalignments. Hence, our CRCC framework further lessens the effect of B2B interference, as compared with the RFCbCB method. Then, Fig. 9(b) presents the CDF of B2B interference strength in scenario 2. Overall, the



(a) scenario 1 (general situation)



(b) scenario 2 (UL-dominating situation)

Fig. 10: Comparison of the CDF of U2U interference.

TABLE 7: Performance metric ranking of different methods

metric	TORS	RFCbCB	CRCC
UL SINR	3	2	1
DL SINR	1	2	2
B2B CLI	3	2	1
U2U CLI	3	2	1
UL throughput	3	2	1
DL throughput	1	2	2
total throughput	3	2	1

trend is similar to that in Fig. 9(a), except that the CDF lines of all methods shift to the left. Since there exist fewer DL-biased cells in scenario 2, B2B interference can be mitigated.

Fig. 10 gives the CDF of U2U interference strength. The trend is similar to that in Fig. 9, where TORS encounters the most severe U2U interference, followed by RFCbCB and CRCC. Because a UE has much less power than a BS, U2U interference is predictably weaker than B2B interference in strength. According to the results in Figs. 9 and 10, we can conclude that the CRCC framework efficiently mitigates CLI (including both B2B and U2U interference), as compared with the TORS and RFCbCB methods.

5.5 Discussion

According to the experimental results in Sections 5.1–5.4, Table 7 gives the performance metric ranking of the TORS, RFCbCB, and CRCC methods. In a performance metric, if the difference between two methods is less than 2%, they have the same ranking. As can be seen, no single method can rank first in all performance metrics. Instead, there is a tradeoff between UL and DL performance for each method.

In the TORS method, cells choose RFCs that meet their traffic demand without considering interference between each other. Since a BS has much larger power than a UE, TORS will benefit DL communication. Hence, TORS has the worst performance in UL SINR but the best performance in DL SINR. However, it is vulnerable to CLI, including both B2B and U2U interference. As a result, TORS has the highest DL throughput and the lowest UL throughput. As discussed in Section 5.1, the loss of UL throughput caused by CLI cannot be compensated by the improvement of DL throughput in TORS. Hence, TORS has the worst performance in total throughput.

The RFCbCB method uses a common RFC to adjust the subframes of cells to reduce their misalignments, which sacrifices a small amount of DL performance in exchange for the improvement of UL performance. As compared with TORS, RFCbCB can get a better tradeoff between UL and DL performance, where RFCbCB ranks second in all performance metrics. By comparing TORS with RFCbCB, we can see how CLI affects UL performance and show how important it is to have RFC coordination among cells to mitigate CLI.

Like RFCbCB, our CRCC framework also strikes a good tradeoff between UL and DL performance. Thanks to the divide-and-conquer strategy, CRCC performs the best on performance metrics of UL SINR, B2B CLI, and U2U CLI. Though CRCC's DL throughput is similar to that of RFCbCB (which is lower than TORS's DL throughput), CRCC can significantly raise UL throughput. Consequently, the CRCC framework achieves the highest total throughput among all methods.

6 Conclusion

D-TDD lets a 5G system flexibly handle asymmetric and fast-varying UL/DL traffic and raise the spectrum utilization, but it brings CLI that makes a significant impact on UL performance. To this end, the paper proposes the CRCC framework to efficiently solve the subframe arrangement problem in 5G D-TDD systems, which not only improves network throughput but also reduces subframe misalignments to mitigate the effect of CLI. CRCC groups cells into clusters through the eAHC scheme, and assigns each cluster one codebook of RFCs to be the reference for subframe arrangement. Based on the divide-and-conquer idea, the cluster is partitioned into basic subclusters, and the master performs RFC coordination for the cells in each basic subcluster to minimize subframe misalignments. Then, CRCC merges subclusters by adjusting the subframes of those cells located on their boundaries. Simulation results reveal that the CRCC framework can efficiently mitigate B2B and U2U CLI, thereby bettering UL throughput and keeping high DL throughput, as compared with both TORS and RFCbCB.

References

[1] J. Li, X. Wang, Z. Li, H. Wang, and L. Li, "Energy efficiency optimization based on eICIC for wireless heterogeneous networks," *IEEE Internet of Things J.*, vol. 6, no. 6, pp. 10166–10176, 2019.

[2] Y.C. Wang and B.J. Huang, "Efficient coordination of almost blank subframes with coupling macro cells in heterogeneous networks," *Int'l J. Comm. Systems*, vol. 33, no. 4, pp. 1–19, 2020.

[3] J. Zheng, L. Gao, H. Zhang, D. Niyato, J. Ren, H. Wang, H. Guo, and Z. Wang, "eICIC configuration of downlink and uplink decoupling with SWIPT in 5G dense IoT HetNets," *IEEE Trans. Wireless Comm.*, vol. 20, no. 12, pp. 8274–8287, 2021.

[4] M.S. Elbamby, M. Bennis, and M. Latva-aho, "UL/DL decoupled user association in dynamic TDD small cell networks," *Proc. IEEE Int'l Symp. Wireless Comm. Systems*, 2015, pp. 456–460.

[5] J. Rachad, R. Nasri, and L. Decreusefond, "Interference analysis in dynamic TDD system combined or not with cell clustering scheme," *Proc. IEEE Vehicular Technology Conf.*, 2018, pp. 1–5.

[6] F. Rinaldi, A. Raschella, and S. Pizzi, "5G NR system design: a concise survey of key features and capabilities," *Wireless Networks*, vol. 27, pp. 5173–5188, 2021.

[7] A. Lukowa and V. Venkatasubramanian, "Centralized UL/DL resource allocation for flexible TDD systems with interference cancellation," *IEEE Trans. Vehicular Technology*, vol. 68, no. 3, pp. 2443–2458, 2019.

[8] 3GPP, "Cross Link Interference (CLI) handling and Remote Interference Management (RIM) for NR," TR 38.828 V16.1.0, Sept. 2019.

[9] Y. Long and Z. Chen, "Interference-cancelled asymmetric traffic cellular networks: dynamic TDD meets massive MIMO," *IEEE Trans. Vehicular Technology*, vol. 67, no. 10, pp. 9785–9800, 2018.

[10] A.A. Esswie and K.I. Pedersen, "Cross-link interference suppression by orthogonal projector for 5G dynamic TDD URLLC systems," *Proc. IEEE Wireless Comm. and Networking Conf.*, 2020, pp. 1–6.

[11] K. Pedersen, A. Esswie, D. Lei, J. Harrebek, Y. Yuk, S. Selvaganapathy, and H. Helmers, "Advancements in 5G New Radio TDD cross link interference mitigation," *IEEE Wireless Comm.*, vol. 28, no. 4, pp. 106–112, 2021.

[12] H. Takahashi, K. Yokomakura, and K. Imamura, "An interference mitigation technique for dynamic TDD based frequency-separated small cell network in LTE-Advanced based future wireless access," *IEICE Trans. Comm.*, vol. E98-B, no. 8, pp. 1436–1446, 2015.

[13] ZTE, "Discussion on duplexing flexibility and cross-link interference mitigation schemes," 3GPP TSG RAN WG1 Meeting#88: R1-1701616, Feb. 2017.

[14] Nokia, "UL interference coordination and power control," 3GPP TSG-RAN WG1 Meeting#88: R1-1703111, Feb. 2017.

[15] M. Hao, H. Zhao, and L. Zhang, "A modified power control algorithm for coordinating CLI in massive MIMO system," *Proc. IEEE Int'l Conf. Electronic Information and Comm. Technology*, 2019, pp. 1–5.

[16] Samsung, "Cross-link interference management based on coordinated beamforming," 3GPP TSG RAN WG1 Meeting#89: R1-1708056, May 2017.

[17] H. Kim, K. Lee, H. Wang, and D. Hong, "Cross link interference mitigation schemes in dynamic TDD systems," *Proc. IEEE Vehicular Technology Conf.*, 2019, pp. 1–5.

[18] Ericsson, "Cross-link interference mitigation for dynamic TDD in dense urban environments," 3GPP TSG-RAN WG1 Meeting#88: R1-1703302, Feb. 2017.

[19] Ericsson, "Cross-link interference mitigation for dynamic TDD in indoor hotspot environments," 3GPP TSG-RAN WG1 Meeting#88: R1-1703303, Feb. 2017.

[20] J.W. Lee, C.G. Kang, and M.J. Rim, "SINR-ordered cross link interference control scheme for dynamic TDD in 5G system," *Proc. Int'l Conf. Information Networking*, 2018, pp. 359–361.

[21] J. Lee, M. Rim, and C.G. Kang, "Decentralized slot-ordered cross link interference control scheme for dynamic time division duplexing (TDD) in 5G cellular system," *IEEE Access*, vol. 9, pp. 63567–63579, 2021.

[22] Y. Chen, L. Wu, Z. Zhang, J. Dang, B. Zhu, and L. Wang, "Blind interference alignment scheme for dynamic TDD systems," *Proc. Information Comm. Technologies Conf.*, 2022, pp. 135–140.

[23] MediaTek, "Interference management in NR," 3GPP TSG RAN WG1 Meeting #88: R1-1702719, Feb. 2017.

[24] X. Li, N. Ma, Q. Tang, and B. Liu, "Buffered DL/UL traffic ratio sensing cell clustering for interference mitigation in LTE TDD system," *Proc. IEEE Wireless Comm. and Networking Conf.*, 2018, pp. 1–6.

[25] A.A. Esswie and K.I. Pedersen, "Inter-cell radio frame coordination scheme based on sliding codebook for 5G TDD systems," *Proc. IEEE Vehicular Technology Conf.*, 2019, pp. 1–6.

[26] 3GPP, "Xn general aspects and principles," TS 38.420 V16.0.0, Jul. 2020.

[27] 3GPP, "Physical channels and modulation (Release 17)," TS 38.211 V17.4.0, Dec. 2022.

[28] B. Clerckx and C. Oestges, *MIMO Wireless Networks: Channels, Techniques and Standards for Multi-Antenna, Multi-User and Multi-Cell Systems*. Amsterdam: Elsevier, 2013.

[29] Y.C. Wang and S.J. Liu, "Minimum-cost deployment of adjustable readers to provide complete coverage of tags in RFID systems," *J. Systems and Software*, vol. 134, pp. 228–241, 2017.

[30] W.H.E. Day and H. Edelsbrunner, "Efficient algorithms for agglomerative hierarchical clustering methods," *J. Classification*, vol. 1, pp. 7–24, 1984.

[31] ETSI, "Study on channel model for frequencies from 0.5 to 100 GHz," 3GPP TR 38.901 version 16.1.0, Nov. 2020.

[32] P. Wu, L. Ding, Y. Wang, L. Li, H. Zheng, and J. Zhang, "Performance analysis for uplink transmission in user-centric ultra-dense V2I networks," *IEEE Trans. Vehicular Technology*, vol. 69, no. 9, pp. 9342–9355, 2020.

[33] M. Emara, H. ElSawy, and G. Bauch, "Prioritized multistream traffic in uplink IoT networks: spatially interacting vacation queues," *IEEE Internet of Things J.*, vol. 8, no. 3, pp. 1477–1491, 2021.