A Simplified Sphere Decoding-Based Detector for Generalized SCMA Codebooks

Xiaotian Fu, Didier Le Ruyet, Bruno Fontana da Silva, Bartolomeu Uchoa-Filho

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 1, 2021, accepted December 10, 2021, date of publication December 16, 2021,
date of current version January 4, 2022.

Digital Object Identifier 10.1109/ACCESS.2021.3136303

A Simplified Sphere Decoding-Based Detector for Generalized SCMA Codebooks

XIAOTIAN FU[1], (Member, IEEE), DIDIER LE RUYET[1], (Senior Member, IEEE), BRUNO FONTANA DA SILVA[2], (Member, IEEE), AND BARTOLOMEU F. UCHÔA-FILHO[3], (Senior Member, IEEE)

[1]CNAM CEDRIC/LAETITIA, 75141 Paris, France
[2]Advanced Analytics, Magazine Luiza, São Paulo 93804-870, Brazil
[3]GPqCom/LCS/EEL, Federal University of Santa Catarina, Florianópolis 88040-900, Brazil

Corresponding author: Xiaotian Fu (xiaotian.fu@lecnam.net)

ABSTRACT Sparse code multiple access (SCMA) is one of the promising schemes to meet high connectivity and spectral efficiency in the future wireless networks. The iterative detectors, for example message passing algorithm (MPA), can provide near optimal multiuser detection (MUD) performance but becomes infeasible when the codebook size is large or the overloading factor is high. Recently, sphere decoding (SD) has been considered in the MUD of SCMA by rewriting the generalized transmission into a linear system. In this work, we first review the state-of-the-art SD-based detectors for SCMA: sphere decoding for SCMA (SD-SCMA) and generalized SD-SCMA (GSD-SCMA). We not only explain the state-of-the-art in a comprehensive way, but also exploit the sorted QR decomposition and Schnorr-Euchner enumeration to accelerate the tree search. Although GSD-SCMA overcomes the codebook constraint of SD-SCMA, its computational complexity is extremely sensitive to the overloading factor. To satisfy the trade-off between complexity and MUD performance, we propose two pruning algorithms, PRUN1 and PRUN2, and introduce the simplified GSD-SCMA (SGSD-SCMA). In the paper, error probabilities of the proposed pruning algorithms are derived. Simulation results show that the proposed detector outperforms the iterative detectors and SD-based state-of-the-art when the overloading factor is moderate and the codebook size is large.

INDEX TERMS SCMA, sphere decoding, pruning algorithm, float-point operations.

I. INTRODUCTION

With the development of Internet of Things (IoT), not only people but also ubiquitous devices are going to be connected in the wireless networks. International Telecommunication Union (ITU) has pinpointed three usage scenarios for 5G and beyond which are enhanced mobile broadband (eMBB), ultra-reliable and low-latency communications (URLLC) and massive machine type communications (mMTC) [1]. For the mMTC, massive connections and high spectral efficiency are two major demands. Non-orthogonal multiple access (NOMA) techniques have been proposed as potential enablers to meet these demands [2]. Compared with conventional orthogonal multiple access (OMA), NOMA enables more users than radio (time/frequency) resources to transmit their signals by leveraging degrees of freedom in the power or code domain. Accordingly, NOMA techniques are categorized into power-domain NOMA (PD-NOMA) and code-domain NOMA (CD-NOMA).

CD-NOMA is motivated by the idea of non-orthogonal spreading sequences. Among all the CD-NOMA techniques, sparse code multiple access (SCMA) has figured prominently. In SCMA, users transmit signals over orthogonal resource elements (REs) and the sparse spreading reduces the number of interfering users at the same RE. SCMA can be regarded as generalized low-density signature (LDS) [3] which combines binary mapping and constellation spreading jointly at the encoder. In other words, SCMA maps users' incoming bits directly to multidimensional codewords according to a pre-designed codebook set [4]. At the receiver, efficient algorithms take charge of multiuser detection (MUD), generally by exploiting the intrinsic sparsity of the signature design.

The pre-designed multidimensional codebooks play an essential role in SCMA system performance [5]. In one of the first works on SCMA codebook design [4], the authors

The associate editor coordinating the review of this manuscript and approving it for publication was Olutayo O. Oyerinde.

introduce a multistage method in order to simplify the design steps of the set of users' codebooks. Several approaches are based on mother codebooks. In [6], [7], the authors focus on optimization of the mother constellation by maximizing important figures of merit, such as minimum Euclidean distance and minimum product distance. By combining probabilistic shaping and geometric shaping in uplink SCMA codebook design, [8] shows superiority of the designed codebooks in terms of block error rate (BLER) performance and average mutual information.

At the receiver of SCMA, the maximum likelihood (ML) algorithm can provide optimal performance for MUD by joint decoding, at the cost of exhaustively testing all combinations of the transmitted codewords. However, it becomes infeasible when the number of users increases or the codebook size gets large. Due to the codeword spreading sparsity, the SCMA receiver can take advantage of the message passing algorithm (MPA) which is motivated by the sum-product algorithm [9], [10]. After iteratively updating the probability messages between the users and REs, the MPA outputs the decoded signals which have the maximum a posteriori probability. The MUD performance of the MPA is determined by the number of iterations, i.e., the higher number of iterations is, the better BER performance it has. Still, MPA complexity increases dramatically with the codebook size, the number of users and the overloading factor (ratio of the number of users to the number of REs). To further reduce the complexity of MPA, some researchers propose Log-MPA and Max-Log-MPA calculating the probability messages in the logarithm domain [11]. Other works propose edge selection criteria to decrease the number of interfering users considered in the calculation of the messages [12], [13]. Other iterative detectors based on expectation propagation algorithm and Gaussian approximation show near-MPA performance and substantial complexity decrease in coded SCMA systems [14], [15].

Meanwhile, sphere decoding (SD) has recently been adopted in the MUD of SCMA systems. SD, firstly proposed as a resolution of the closest lattice point search problem [16], [17], has been regarded as an effective scheme in MIMO detection [18]. The principle of SD is to search the nearest lattice point to the received signal within a predefined hypersphere. By leveraging matrix operation, e.g. QR decomposition, SD obtains an upper triangular matrix and transforms the detection into a sequential decoding by the tree search process. The complexity of SD is proportional to the number of nodes visited in the tree search. Authors in [19] apply list SD in the MPA in order to prune the candidates at each RE that are outside of the search space when calculating probability messages. To employ SD to the entire SCMA MUD, authors in [20], [21] reformulate some multidimensional codebooks with special structures and transform the SCMA MUD into a binary lattice point detection. Although the proposed detectors in [20], [21] show better performance and lower complexity than the MPA, they are not applicable to generalized SCMA multidimensional codebooks. The work proposed in [22] introduces the SD for SCMA (SD-SCMA) which requires no specific codebook reformulation. Limited by the shape of the channel matrix, SD-SCMA constrains the transmitted codewords to have constant modulus. To overcome this constraint, authors in [23] propose the generalized SD for SCMA (GSD-SCMA) which is composed of two parts. As GSD-SCMA performs a brute-force search on partial transmitted symbols, its computational complexity may surge when the codebook size or overloading factor is large.

Hence, in this work we focus on further reducing complexity of GSD-SCMA detector while still reaching good MUD performance. We propose some algorithms to remedy the brute-force search issue in GSD-SCMA detector. The new detector scheme with lower complexity is called simplified GSD for SCMA (SGSD-SCMA). The main contributions of this work are exhibited as follows:

- A clear description of the generalized SCMA system model in the real domain is provided. It is used to facilitate the employment of SD-based detectors in SCMA systems. The system model is expressed to be linear, applicable to all types of SCMA codebooks.
- Based on [22], [23], we present the SD-SCMA and GSD-SCMA detectors in a comprehensive way. As a novel contribution, we leverage the sorted QR decomposition (SQRD) [24] and Schnorr-Euchner (SE) enumeration [25] in those detectors to further enhance the decoding efficiency and raise the tree search speed.
- We introduce the SGSD-SCMA detector which consists in the proposed pruning algorithms based on the GSD-SCMA detector. The SGSD-SCMA detector can also be regarded as the parameterized GSD-SCMA detector based on the error detection probability p. There is a trade-off between the MUD performance and the computational complexity when selecting p value. Thus, we derive the error probability expressions of the proposed pruning algorithms and analyze the codeword error rate (CER) for the proposed detector.
- In the Monte-Carlo simulation, we first prove the accuracy of the theoretical CER analysis of the proposed SGSD-SCMA detector. Secondly, we compare the proposed detector with the SD-SCMA, GSD-SCMA and Max-Log-MPA detectors in terms of bit error rate (BER) and computational complexity. The proposed algorithms significantly decrease the complexity for the SGSD-SCMA detector in comparison to the GSD-SCMA detector. By simulating with three SCMA codebooks designed by different approaches, it is found that the SGSD-SCMA detector outperforms other detectors the most when the overloading factor is moderate and the codebook size is large.

The rest of this paper is structured as follows. Section II reviews the uplink SCMA transmission system and provides the complex to real system model conversion. A comprehensive introduction of the SD-based SCMA state-of-the-art detectors is presented in Section III. In Section IV, we give a toy example, real-valued multiuser uplink transmission, to explain the general idea of pruning algorithm

in an overloaded transmission. The two proposed pruning algorithms and theoretical CER analysis of the SGSD-SCMA detector are introduced in Section V. Section VI analyzes the computational complexity of different detectors. The evaluation of simulated numerical results is presented in Section VII. Finally, Section VIII summarizes the paper.

Notations: Lowercase letters (e.g., x) denote scalars, bold lowercase letters (e.g., $\mathbf{x}$) denote column vectors, bold uppercase letters (e.g., $\mathbf{X}$) denote matrices and calligraphic uppercase letters (e.g., $\mathcal{X}$) denote sets and sequences. Complex Gaussian distribution function with mean τ and variance σ^2 is denoted by $\mathcal{N}_c(\tau, \sigma^2)$. $(\cdot)^\intercal$ is transpose operator of matrices. $\mathbf{I}_q$ denotes the $q \times q$ identity matrix and $\mathbf{0}_{m\times n}$ denotes the $m \times n$ zero matrix. The important notations used in this paper are listed in the Table 1.

II. SCMA UPLINK SYSTEM MODEL

A. GENERAL SCMA TRANSMISSION SYSTEM

In this paper, we consider a general uplink SCMA system where J users share K orthogonal REs for the signal transmission. For the remainder of this paper, it is always assumed an overload condition on the system, meaning that $J > K$. Each user occupies d_v ($d_v < K$) REs. We assume that the number of users linked to each RE is the same, denoted as $d_f = Jd_v/K$, and ideally $d_f \ll J$ to maintain the sparsity feature. In this work, we assume that K is an integer multiple of d_v. The encoding procedure for user $j \in \{1, 2, \cdots, J\}$ is expressed as $f_j : \mathbf{b}_j \to \mathbf{x}_j$, where $\mathbf{b}_j \in \{0, 1\}^{\log_2(M)\times 1}$ is the binary message and $\mathbf{x}_j \in \mathbb{C}^{d_v\times 1}$ is the transmitted codeword selected from codebook $\mathcal{X}_j$ which has size M. Then a $K \times d_v$ binary mapping matrix $\mathbf{S}_j$ spreads user j's d_v-dimensional codeword over the K REs. The complex-valued received signal $\mathbf{y} \in \mathbb{C}^{K\times 1}$ is written as

$$\mathbf{y} = \sum_{j=1}^{J} diag(\mathbf{h}_j)\mathbf{S}_j\mathbf{x}_j + \mathbf{n} \tag{1}$$

$$= \mathbf{G}\mathbf{x}_{\mathrm{T}} + \mathbf{n} \tag{2}$$

where $\mathbf{h}_j \in \mathbb{C}^{K\times 1}$ is the channel vector from user j to the receiver, and $\mathbf{n} \in \mathbb{C}^{K\times 1}$ is the additive white Gaussian noise (AWGN) vector whose entries follow $\mathcal{N}_c(0, \sigma^2)$. $\mathbf{G} = [\mathbf{G}_1, \cdots, \mathbf{G}_J] \in \mathbb{C}^{K\times Jd_v}$ is called effective channel gain matrix, where $\mathbf{G}_j = diag(\mathbf{h}_j)\mathbf{S}_j$ is user j's effective channel gain matrix. $\mathbf{x}_{\mathrm{T}} = [\mathbf{x}_1^\intercal, \cdots, \mathbf{x}_J^\intercal]^\intercal \in \mathbb{C}^{Jd_v\times 1}$ is the transmitted codeword vector consisting of the codewords of all the users.

Every column of the binary mapping matrix $\mathbf{S}_j$ has only one non-zero element. Usually, the mapping relation of $\mathbf{S}_j$ can also be represented by the j-th column of the factor graph matrix $\mathbf{F} = [\mathbf{f}_1, \cdots, \mathbf{f}_J] \in \{0, 1\}^{K\times J}$ which indicates the resource allocation of the transmission. The relation between $\mathbf{f}_j$ and $\mathbf{S}_j$ can be expressed as $diag(\mathbf{f}_j) = \mathbf{S}_j\mathbf{S}_j^\intercal$. A benchmark example of $\mathbf{F}$ is given below:

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix} \tag{3}$$

which has $J = 6$, $K = 4$, $d_v = 2$ and $d_f = 3$. Thus the mapping matrix of the first user is

$$\mathbf{S}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{4}$$

By concatenating the mapping matrix of all users, we can obtain the mapping matrix of the transmission system

$$\mathbf{S} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{5}$$

B. THE REWRITTEN SYSTEM MODEL IN THE REAL DOMAIN

In order to facilitate the application of SD-based detectors for SCMA, we rewrite the complex-valued system model in the real domain. Please note that this rewriting is generalized for any type of SCMA codebooks.

To express the SCMA transmission Eq. (2) in the real domain, we separate the real and imaginary parts. The received signal in the real domain is defined as

$$\bar{\mathbf{y}} = \bar{\mathbf{G}}\bar{\mathbf{x}}_{\mathrm{T}} + \bar{\mathbf{n}} \tag{6}$$

where $\bar{\mathbf{y}} \in \mathbb{R}^{2K\times 1}$, $\bar{\mathbf{G}} \in \mathbb{R}^{2K\times 2Jd_v}$, $\bar{\mathbf{x}}_{\mathrm{T}} \in \mathbb{R}^{2Jd_v\times 1}$ and $\bar{\mathbf{n}} \in \mathbb{R}^{2K\times 1}$ are real-valued received signal vector, effective channel matrix, effective transmitted symbol vector and AWGN vector, respectively. For simplicity, in the remainder of this paper we will refer to $\bar{\mathbf{G}}$ as the real-valued channel matrix and to $\bar{\mathbf{x}}_T$ as the real-valued transmitted symbol vector. Matrix $\bar{\mathbf{G}}$ is built from the complex matrix $\mathbf{G}$ by replacing each of its complex entries g_{il} ($1 \leqslant i \leqslant K$, $1 \leqslant l \leqslant Jd_v$) with $\begin{bmatrix} \Re\{g_{il}\} & -\Im\{g_{il}\} \\ \Im\{g_{il}\} & \Re\{g_{il}\} \end{bmatrix}$. As a column vector, $\bar{\mathbf{x}}_{\mathrm{T}}$ is constructed from its associated complex vector $\mathbf{x}_{\mathrm{T}}$ by substituting every complex element x_n ($1 \leqslant n \leqslant Jd_v$) with $\begin{bmatrix} \Re\{x_n\} & \Im\{x_n\} \end{bmatrix}^\intercal$. The same real-valued column vector conversion is also applied to vectors $\mathbf{y}$ and $\mathbf{n}$ for obtaining vectors $\bar{\mathbf{y}}$ and $\bar{\mathbf{n}}$, respectively.

Assume that the channel coefficients are estimated perfectly at the receiver. The optimal maximum likelihood (ML) detection for SCMA is given by

$$\hat{\bar{\mathbf{x}}}_{\mathrm{T}} = \arg\min_{\bar{\mathbf{x}}\in\bar{\mathcal{X}}} \|\bar{\mathbf{y}} - \bar{\mathbf{G}}\bar{\mathbf{x}}\|^2 \tag{7}$$

where $\bar{\mathcal{X}} = \bar{\mathcal{X}}_1 \times \bar{\mathcal{X}}_2 \times \cdots \bar{\mathcal{X}}_J$ is the codebook collection of J users in the real domain. Please note that $\bar{\mathcal{X}}_j$ is a sequence of which the elements are $2d_v \times 1$ vectors. For example, $\bar{\mathcal{X}}_j(m)$ represents the m-th real-valued codeword of user j. As SCMA is an overloaded multiuser transmission technique, the computational complexity of ML receiver is proportional to $|\bar{\mathcal{X}}| = M^J$, which surges rapidly for even small choices of M and J s.t. $M \geqslant 2$ and $J > K$. Instead of performing exhaustive search, sphere decoding (SD) can be deemed as

TABLE 1. Notations used in the paper.

Symbol	Meaning
$\bar{\mathbf{x}}_{\mathrm{T}} \in \mathbb{C}^{2Jd_v \times 1}$	Real-valued transmitted codeword vector comprising codewords of all the users
$\bar{\mathbf{G}} \in \mathbb{C}^{2K \times 2Jd_v}$	Real-valued effective channel gain matrix
$\bar{\mathbf{n}} \in \mathbb{C}^{2K \times 1}$	Real-valued AWGN vector
$\bar{\mathbf{y}} \in \mathbb{C}^{2K \times 1}$	Real-valued received signal vector
$\bar{\mathbf{G}}^{(1)}$	Matrix containing the first $2K$ columns of the matrix $\bar{\mathbf{G}}$
$\bar{\mathbf{G}}^{(2)}$	Matrix containing the last $(2Jd_v - 2K)$ columns of the matrix $\bar{\mathbf{G}}$
$\bar{\mathbf{x}}_{\mathrm{T}}^{(1)}$	Vector containing the first $2K$ elements of the vector $\bar{\mathbf{x}}$
$\bar{\mathbf{x}}_{\mathrm{T}}^{(2)}$	Vector containing the last $(2Jd_v - 2K)$ elements of the vector $\bar{\mathbf{x}}$
$\tilde{\mathbf{G}} \in \mathbb{C}^{2Jd_v \times 2Jd_v}$	Real-valued modified channel gain matrix in the SD-SCMA detector
d^2	Initial squared sphere radius
p	Error detection probability in the proposed pruning algorithms
M_1	Average number of distinct projections at each real dimension

ML with the conditions which is expressed as

$$\hat{\bar{\mathbf{x}}}_{\mathrm{T}} = \arg\min_{\bar{\mathbf{x}} \in \bar{\mathcal{X}}} \|\bar{\mathbf{y}} - \bar{\mathbf{G}}\bar{\mathbf{x}}\|^2 \tag{8}$$

$$\text{subject to } \|\bar{\mathbf{y}} - \bar{\mathbf{G}}\bar{\mathbf{x}}\|^2 \leqslant d^2 \tag{9}$$

where d^2 is the squared sphere radius which defines the hyper-sphere for the tree search in SD. It is taken as an alternative to ML detector which can guarantee good detection performance and significant complexity decrease. More details on that will be presented in the following part of this paper.

III. STATE-OF-THE-ART SD-BASED SCMA DETECTORS

In this section we provide a comprehensive description of SD-based SCMA detector state-of-the-art, which are SD-SCMA [22] and GSD-SCMA [23]. We not only elaborate on the state-of-the-art by illustrating with flowcharts and detailed pseudo-code algorithms, but also enhance the original scheme by adopting SQRD.

A. SD-SCMA

As revealed in Eq. (6), the channel matrix $\bar{\mathbf{G}}$ is a "flat" matrix since its number of rows is less than number of columns. The fact that $\bar{\mathbf{G}}$ is not a square matrix makes it difficult to solve Eqs. (7) and (9) in an efficient way. To resolve this problem and properly implement SD in SCMA MUD, SD-SCMA takes advantage of the method inspired by the Tikhonov regularization [26]. We define the real-valued modified channel matrix as

$$\tilde{\mathbf{G}} = \begin{bmatrix} \bar{\mathbf{G}}^{(1)} & \bar{\mathbf{G}}^{(2)} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{10}$$

where matrix $\bar{\mathbf{G}}^{(1)}$ contains the first $2K$ columns of the matrix $\bar{\mathbf{G}}$ having size $2K \times 2K$ and matrix $\bar{\mathbf{G}}^{(2)}$ contains the matrix $\bar{\mathbf{G}}$ from the $(2K+1)$-th column to the last column having size $2K \times (2Jd_v - 2K)$. Thus, it can be expressed as $\bar{\mathbf{G}} = \begin{bmatrix} \bar{\mathbf{G}}^{(1)} & \bar{\mathbf{G}}^{(2)} \end{bmatrix}$. The received signal in the real domain can be modified as

$$\tilde{\mathbf{y}} = \tilde{\mathbf{G}}\bar{\mathbf{x}}_{\mathrm{T}} + \tilde{\mathbf{n}} \tag{11}$$

$$\begin{bmatrix} \bar{\mathbf{y}} \\ \mathbf{0} \end{bmatrix} = \tilde{\mathbf{G}} \begin{bmatrix} \bar{\mathbf{x}}_{\mathrm{T}}^{(1)} \\ \bar{\mathbf{x}}_{\mathrm{T}}^{(2)} \end{bmatrix} + \begin{bmatrix} \mathbf{n} \\ -\bar{\mathbf{x}}_{\mathrm{T}}^{(2)} \end{bmatrix} \tag{12}$$

where $\bar{\mathbf{x}}_{\mathrm{T}} = \begin{bmatrix} \bar{\mathbf{x}}_{\mathrm{T}}^{(1)} \\ \bar{\mathbf{x}}_{\mathrm{T}}^{(2)} \end{bmatrix}$. We remark that the vector $\bar{\mathbf{x}}_{\mathrm{T}}^{(1)} \in \bar{\mathcal{X}}^{(1)}$ has size $2K \times 1$ and vector $\bar{\mathbf{x}}_{\mathrm{T}}^{(2)} \in \bar{\mathcal{X}}^{(2)}$ has size $(2Jd_v - 2K) \times 1$, where $\bar{\mathcal{X}}^{(1)} = \bar{\mathcal{X}}_1 \times \cdots \times \bar{\mathcal{X}}_{\frac{K}{d_v}}$ and $\bar{\mathcal{X}}^{(2)} = \bar{\mathcal{X}}_{\frac{K}{d_v}+1} \times \cdots \times \bar{\mathcal{X}}_J$ while $\mathbf{0}$ in this case has size $(2Jd_v - 2K) \times 1$. Thus, the SD is rewritten as

$$\hat{\bar{\mathbf{x}}}_{\mathrm{T}} = \arg\min_{\bar{\mathbf{x}} \in \bar{\mathcal{X}}} \|\tilde{\mathbf{y}} - \tilde{\mathbf{G}}\bar{\mathbf{x}}\|^2 - \bar{\mathbf{x}}^{(2)\mathsf{T}}\bar{\mathbf{x}}^{(2)} \tag{13}$$

$$\text{subject to } \|\tilde{\mathbf{y}} - \tilde{\mathbf{G}}\bar{\mathbf{x}}\|^2 - \bar{\mathbf{x}}^{(2)\mathsf{T}}\bar{\mathbf{x}}^{(2)} \leqslant d^2 \tag{14}$$

The prerequisite of applying the Tikhonov regularization in SD-SCMA is $\bar{\mathbf{x}}^{(2)}$ of constant modulus, which guarantees the equivalence between Eqs. (8) and (9) and Eqs. (13) and (14). As $\bar{\mathbf{x}}^{(2)}$ has constant modulus, Eqs. (13) and (14) can be simplified to

$$\hat{\bar{\mathbf{x}}}_{\mathrm{T}} = \arg\min_{\bar{\mathbf{x}} \in \bar{\mathcal{X}}} \|\tilde{\mathbf{y}} - \tilde{\mathbf{G}}\bar{\mathbf{x}}\|^2 \tag{15}$$

$$\text{subject to } \|\tilde{\mathbf{y}} - \tilde{\mathbf{G}}\bar{\mathbf{x}}\|^2 \leqslant d'^2 \tag{16}$$

where $d'^2 = d^2 - \|\bar{\mathbf{x}}^{(2)}\|^2$. This prerequisite can be regarded as a constraint of the SD-SCMA detector. This means that the SD-SCMA can be only applied to SCMA codebooks which guarantee $\bar{\mathbf{x}}^{(2)}$ to have constant modulus.

We adopt SQRD [24] to obtain an upper triangular matrix for the tree search in SD-SCMA. SQRD can be seen as a modified Gram-Schmidt algorithm which is famous for efficiently improving MIMO detection [27]. In the MIMO scenario, it minimizes the diagonal element modulus of the output upper triangular matrix from upper left to bottom right by reordering the columns of the input matrix, usually the channel matrix. As a result, SQRD is able to enhance the efficiency of the earlier layers in the tree search, which further reduces the complexity by decreasing the number of visited nodes. Moreover, SQRD guarantees output upper triangular matrix to have positive diagonal elements. The SQRD for SD-SCMA is written as

$$\text{sqrd}(\tilde{\mathbf{G}}) : \mathbf{QR} = \tilde{\mathbf{G}}\mathbf{P} \tag{17}$$

where $\mathbf{Q} \in \mathbb{R}^{2Jd_v \times 2Jd_v}$ is an orthogonal matrix, $\mathbf{R} \in \mathbb{R}^{2Jd_v \times 2Jd_v}$ is an upper triangular matrix and $\mathbf{P} \in \mathbb{R}^{2Jd_v \times 2Jd_v}$

Algorithm 1 SQRD for SCMA

Input: $\tilde{\mathbf{G}}$
Output: $\mathbf{R}$, $\mathbf{Q}$ and $\mathbf{P}$
Initialization: $\mathbf{Q} = \tilde{\mathbf{G}}$, $\mathbf{R} = \mathbf{0}^{2Jd_v \times 2Jd_v}$, $\mathbf{P} = \mathbf{I}_{2Jd_v}$ and $\mathbf{m} = \mathbf{0}^{J\times 1}$
for $j = 1, 2, \cdots, J$ **do**
  $m_j = \sum_{l=0}^{2d_v-1} \|\mathbf{q}_{j+l}\|_2^2$
end for
for $j = 1, 2, \cdots, J$ **do**
  $i_j = \arg\min_{l\in\{j,j+1\cdots,J\}} m_l$
  Exchange columns $2(j-1)d_v+1$ to $2jd_v$ and $2(i_j-1)d_v+1$ to $2i_jd_v$ one-by-one in $\mathbf{R}$, $\mathbf{Q}$ and $\mathbf{P}$
  for $i_1 = j, j+1, \cdots, j+(2d_v-1)$ **do**
    $r_{i_1,i_1} = \|\mathbf{q}_{i_1}\|_2^2$
    $\mathbf{q}_{i_1} = \mathbf{q}_{i_1}/\sqrt{r_{i_1,i_1}}$
    for $i_2 = i_1+1, i_1+2, \cdots, 2Jd_v$ **do**
      $r_{i_1,i_2} = \mathbf{q}_{i_1}^{\mathsf{T}}\mathbf{q}_{i_2}$
      $\mathbf{q}_{i_2} = \mathbf{q}_{i_2} - r_{i_1,i_2}\mathbf{q}_{i_1}$
      $m_{\lceil i_2/(2d_v)\rceil} = m_{\lceil i_2/(2d_v)\rceil} - r_{i_1,i_2}$
    end for
  end for
end for

is a permutation matrix. Considering that the columns with the indices $[2d_vj-(2d_v-1), \cdots, 2d_vj]$ $(j \in \{1, 2, \cdots, J\})$ belong to the same user, the SQRD for SCMA should reorder the columns of $\tilde{\mathbf{G}}$ in groups of $2d_v$. To achieve this, SQRD for SCMA calculates the squared modulus sum of the $2d_v$ columns that are associated with each user. To better understand the SQRD for SCMA, please refer to Algorithm 1.

As SQRD introduces permutation among columns of $\tilde{\mathbf{G}}$, the decoding problem in Eqs. (15) and (16) becomes

$$\hat{\tilde{\mathbf{x}}}_{\mathrm{T}} = \arg\min_{\tilde{\mathbf{x}}\in\tilde{\mathcal{X}}} \|\mathbf{Q}^{\mathsf{T}}\tilde{\mathbf{y}} - \mathbf{R}\mathbf{P}^{-1}\tilde{\mathbf{x}}\|^2 \tag{18}$$

$$\text{subject to } \|\mathbf{Q}^{\mathsf{T}}\tilde{\mathbf{y}} - \mathbf{R}\mathbf{P}^{-1}\tilde{\mathbf{x}}\|^2 \leqslant d'^2 \tag{19}$$

By defining $\tilde{\mathbf{y}}' = \mathbf{Q}^{\mathsf{T}}\tilde{\mathbf{y}}$ and $\bar{\mathbf{x}}^{\mathrm{p}} = \mathbf{P}^{-1}\tilde{\mathbf{x}}$, the above decoding problem is simplified to

$$\hat{\tilde{\mathbf{x}}}_{\mathrm{T}} = \mathbf{P}\arg\min_{\bar{\mathbf{x}}^{\mathrm{p}}\in\bar{\mathcal{X}}^{\mathrm{p}}} \|\tilde{\mathbf{y}}' - \mathbf{R}\bar{\mathbf{x}}^{\mathrm{p}}\|^2 \tag{20}$$

$$\text{subject to } \|\tilde{\mathbf{y}}' - \mathbf{R}\bar{\mathbf{x}}^{\mathrm{p}}\|^2 \leqslant d'^2 \tag{21}$$

where $\bar{\mathcal{X}}^{\mathrm{p}}$ represents the permuted real-valued codebook collection. Since $\mathbf{R}$ is an upper triangular matrix, Eq. (21) can be rewritten as

$$d'^2 \geqslant \sum_{i=1}^{2Jd_v}\left(\tilde{y}'_i - \sum_{l=i}^{2Jd_v} r_{i,l}\bar{x}_l^{\mathrm{p}}\right)^2 \tag{22}$$

$$= (\tilde{y}'_{2Jd_v} - r_{2Jd_v,2Jd_v}\bar{x}_{2Jd_v}^{\mathrm{p}})^2 + \left(\tilde{y}'_{2Jd_v-1} - \sum_{l=2Jd_v-1}^{2Jd_v} r_{2Jd_v-1,l}\bar{x}_l^{\mathrm{p}}\right)^2 + \cdots \tag{23}$$

Therefore, it is natural to start the tree search from the bottom and trace upwards. The layers with the indices $i \in \{2d_vj-(2d_v-1), \cdots, 2d_vj\}$ $(j \in \{1, 2, \cdots, J\})$ correspond to user j. Because each user spreads the same information bits over d_v REs, $2d_v$ layers should be correlated in the tree search of SD-SCMA. The layers with indices multiple of $2d_v$ have M branches and the rest has only one branch. In other words, the layers having indices $2d_vj$ $(j \in \{1, 2, \cdots, J\})$ determine the layers with indices from $2d_vj-(2d_v-1)$ to $2d_vj-1$. As a result, branch selection calculation is performed only in J layers. Details of the SD-SCMA tree search is illustrated by a flow chart as shown in Fig. 1.

At the $2Jd_v$-th layer, for the first branch selection calculation, the necessary condition of Eq. (22) is

$$(\tilde{y}'_{2Jd_v} - r_{2Jd_v,2Jd_v}\bar{x}_{2Jd_v}^{\mathrm{p}})^2 \leqslant d'^2 \tag{24}$$

The corresponding range of $\bar{x}_{2Jd_v}^{\mathrm{p}}$ is

$$\frac{\tilde{y}'_{2Jd_v} - d'}{r_{2Jd_v,2Jd_v}} \leqslant \bar{x}_{2Jd_v}^{\mathrm{p}} \leqslant \frac{\tilde{y}'_{2Jd_v} + d'}{r_{2Jd_v,2Jd_v}} \tag{25}$$

To extend this to the i-th $(i = 2d_vj,\ j \in \{1, 2, \cdots, J\})$ layer, we define the lower and upper bounds of $\bar{x}_i^{\mathrm{p}}$ as

$$l_i = \frac{1}{r_{i,i}}\left(\tilde{y}'_i - \sum_{l=i+1}^{2Jd_v} r_{i,l}\bar{x}_l^{\mathrm{p}} - \sqrt{d'^2 - p_{i+2d_v}}\right) \tag{26}$$

and

$$u_i = \frac{1}{r_{i,i}}\left(\tilde{y}'_i - \sum_{l=i+1}^{2Jd_v} r_{i,l}\bar{x}_l^{\mathrm{p}} + \sqrt{d'^2 - p_{i+2d_v}}\right) \tag{27}$$

respectively, where p_i is the path metric in the i-th layer which will be introduced later. Thus, the corresponding range of $\bar{x}_i^{\mathrm{p}}$ is expressed as

$$l_i \leqslant \bar{x}_i^{\mathrm{p}} \leqslant u_i \tag{28}$$

We use vector $\mathbf{a}_i$ to record the indices of the candidate codewords, i.e., the codewords that satisfy the condition Eq. (28), which can be expressed as

$$\mathbf{a}_i = \mathrm{enum}\big(l_i, u_i, \bar{\mathcal{X}}_j^{2d_v}\big) \tag{29}$$

where $j = \lceil \frac{i}{2d_v} \rceil$, $\mathrm{enum}(v_1, v_2, \mathcal{S})$ denotes the function enumerating all the positions of elements in the real sequence $\mathcal{S}$ having values between v_1 and v_2 and $\bar{\mathcal{X}}_l^k$ represents the real sequence consisting of all the k-th entries of the elements in the sequence $\bar{\mathcal{X}}_l$. s_i is the number of valid branches associated with the candidate codewords at the i-th layer, obtained by

$$s_i = \mathrm{length}(\mathbf{a}_i) \tag{30}$$

According to the tree search rules for SCMA, $\mathbf{a}_i$ and s_i with $i = 2d_vj,\ j \in \{1, 2, \cdots, J\}$ are the vector of candidate codewords indices and the number of candidate codewords of user j, respectively. Hence, to visit the next branch selection layer, the layer index is updated by $i = i - 2d_v$ as shown in Fig. 1.

To accelerate the tree search in SD-SCMA, we adopt SE strategy [25] to order the codeword candidates. In SE enumeration, the candidates are examined and sorted based on their path metric values. As the path metric is identical for all child

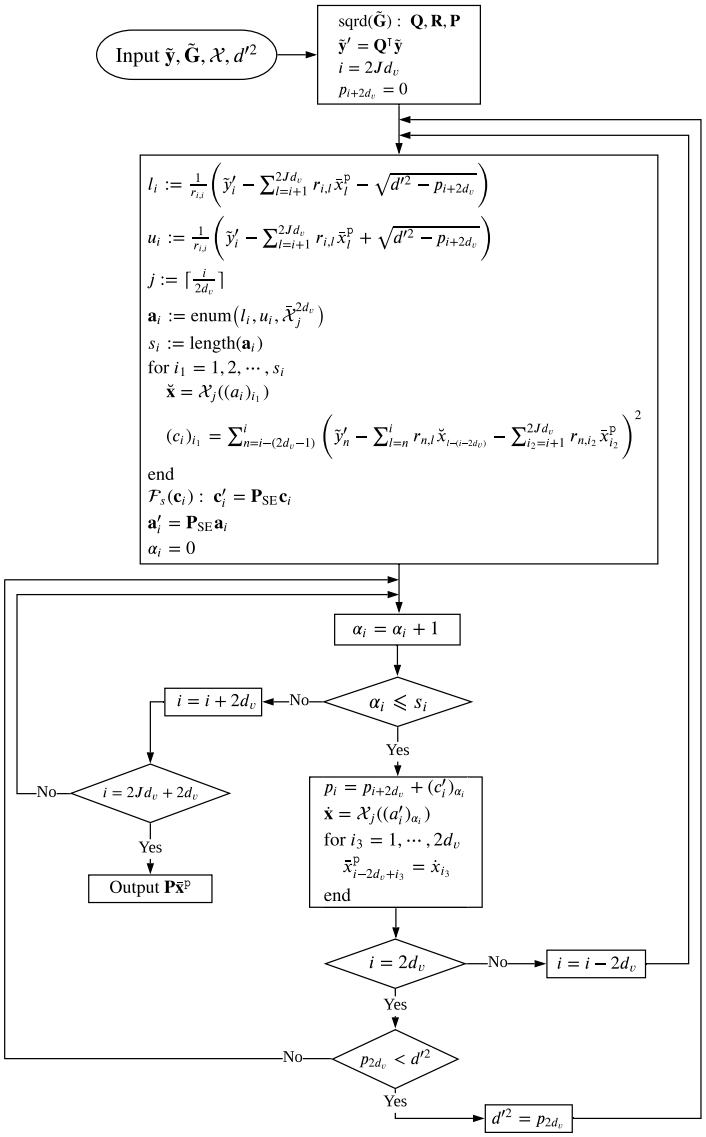


FIGURE 1. Flow chart of SD-SCMA.

nodes having the same parent in tree search, the ordering is based on the branch metrics of child nodes. At the i-th layer ($i = 2d_v j,\ j \in \{1, 2, \cdots, J\}$), all the candidate codewords

$$\check{\mathbf{x}} = \mathcal{X}_j((a_i)_{i_1}), \quad 1 \leqslant i_1 \leqslant s_i \tag{31}$$

are considered, where $(a_i)_{i_1}$ refers to the i_1 elements of vector $\mathbf{a}_i$. For a given candidate codeword $\check{\mathbf{x}}$, its branch metric is calculated as [25]

$$(c_i)_{i_1} = \sum_{n=i-(2d_v-1)}^{i}\left(\tilde{y}'_n - \sum_{l=n}^{i} r_{n,l}\check{x}_{l-(i-2d_v)} - \sum_{i_2=i+1}^{2Jd_v} r_{n,i_2}\bar{x}^{\mathrm{p}}_{i_2}\right)^2 \tag{32}$$

where $\check{x}_k$ is the k-th entry of vector $\check{\mathbf{x}}$. Consequently, vector $\mathbf{c}_i = [(c_i)_1, (c_i)_2, \cdots, (c_i)_{s_i}]^{\mathsf{T}}$ stores all the branch metrics of the candidate codewords at i-th layer. Then, we sort the elements of vector $\mathbf{c}_i$ to satisfy the condition $(c_i)_n \leqslant (c_i)_{n+1}$ where $1 \leqslant n \leqslant s_i - 1$. This sorting process can be formulated as

$$\mathcal{F}_s(\mathbf{c}_i) : \ \mathbf{c}'_i = \mathbf{P}_{\mathrm{SE}}\mathbf{c}_i \tag{33}$$

where function $\mathcal{F}_s(\cdot)$ refers to the aforementioned sorting criterion, $\mathbf{c}'_i$ denotes the sorted vector and $\mathbf{P}_{\mathrm{SE}}$ is the permutation matrix. Then, the same element-wise permutation is applied to the candidate codewords indices vector $\mathbf{a}_i$, given in Eq. (29), to obtain the sorted codewords indices vector $\mathbf{a}'_i$, denoted as

$$\mathbf{a}'_i = \mathbf{P}_{\mathrm{SE}}\mathbf{a}_i \tag{34}$$

α_i is used to refer to the index of the chosen valid branch in the i-th layer, which follows $1 \leqslant \alpha_i \leqslant s_i$. If the branch index α_i is chosen, the cumulative path metric p_i ($i = 2d_v j,\ j \in \{1, 2, \cdots, J\}$) is computed by

$$p_i = p_{i+2d_v} + (c'_i)_{\alpha_i} \tag{35}$$

where $(c_i')_{\alpha_i}$ is the α_i-th element of vector $\mathbf{c}_i'$. At the beginning of the tree search, $p_{2Jd_v+2d_v} = 0$ is initialized. The decoding results are partially updated by the chosen branch, expressed as

$$\bar{x}^{\mathrm{p}}_{i-2d_v+i_3} = \dot{x}_{i_3}, \quad 1 \leqslant i_3 \leqslant 2d_v \tag{36}$$

where vector $\dot{\mathbf{x}} = \mathcal{X}_j((a_i')_{\alpha_i})$ denotes the chosen codeword. Every time, when a temporary best solution is found, the squared radius is updated by

$$d'^2 = p_{2d_v} \tag{37}$$

The tree search stops when no better solution can be found; it outputs the final result $\mathbf{P}\bar{\mathbf{x}}^{\mathrm{p}}$.

B. GSD-SCMA

As discussed above, SD-SCMA is only compatible with codebooks having $\bar{\mathbf{x}}^{(2)}$ of constant modulus. To circumvent this codebook power constraint, GSD-SCMA is proposed for generalized SCMA codebooks [23].

Since the channel matrix can be expressed as $\bar{\mathbf{G}} = \left[\begin{array}{cc}\bar{\mathbf{G}}^{(1)} & \bar{\mathbf{G}}^{(2)}\end{array}\right]$, it can be rewritten as

$$\begin{aligned}\bar{\mathbf{G}} &= \left[\begin{array}{cc}\bar{\mathbf{G}}^{(1)} & \bar{\mathbf{G}}^{(2)}\end{array}\right] &(38)\\ &= \mathbf{Q}_1\left[\begin{array}{cc}\mathbf{R}_1\mathbf{P}_1^{-1} & \mathbf{Q}_1^{-1}\bar{\mathbf{G}}^{(2)}\end{array}\right] &(39)\\ &= \mathbf{Q}_1\left[\begin{array}{cc}\mathbf{R}_1\mathbf{P}_1^{-1} & \mathbf{R}_2\end{array}\right] &(40)\end{aligned}$$

where $\mathbf{Q}_1$ and $\mathbf{R}_1$ are outputs of SQRD of $\bar{\mathbf{G}}^{(1)}$ expressed as

$$\mathrm{sqrd}(\bar{\mathbf{G}}^{(1)}) : \mathbf{Q}_1\mathbf{R}_1 = \bar{\mathbf{G}}^{(1)}\mathbf{P}_1 \tag{41}$$

and $\mathbf{R}_2 \in \mathbb{R}^{2K\times(2Jd_v-2K)}$ is the multiplication of matrix $\mathbf{Q}_1^{-1}$ and $\bar{\mathbf{G}}^{(2)}$. Please note that because the first $2K$ columns of the transmission mapping matrix $\mathbf{S}$ are orthogonal, $\mathbf{R}_1$ is a positive diagonal matrix which simplifies the calculation in the tree search of GSD-SCMA.

Based on the channel matrix expression, Eq. (40), the decoding problem in Eqs. (8) and (9) becomes

$$\hat{\bar{\mathbf{x}}}_{\mathrm{T}} = \arg\min_{\substack{\bar{\mathbf{x}}^{(1)}\in\bar{\mathcal{X}}^{(1)}\\ \bar{\mathbf{x}}^{(2)}\in\bar{\mathcal{X}}^{(2)}}} \|\mathbf{Q}_1^{\mathsf{T}}\bar{\mathbf{y}} - \mathbf{R}_1\mathbf{P}_1^{-1}\bar{\mathbf{x}}^{(1)} - \mathbf{R}_2\bar{\mathbf{x}}^{(2)}\|^2 \tag{42}$$

$$\text{subject to } \|\mathbf{Q}_1^{\mathsf{T}}\bar{\mathbf{y}} - \mathbf{R}_1\mathbf{P}_1^{-1}\bar{\mathbf{x}}^{(1)} - \mathbf{R}_2\bar{\mathbf{x}}^{(2)}\|^2 \leqslant d^2 \tag{43}$$

which is known as the GSD-SCMA decoding problem [23]. Defining $\bar{\mathbf{y}}' = \mathbf{Q}_1^{\mathsf{T}}\bar{\mathbf{y}} - \mathbf{R}_2\bar{\mathbf{x}}^{(2)}$ and $\bar{\mathbf{x}}^{(1)\mathrm{p}} = \mathbf{P}_1^{-1}\bar{\mathbf{x}}^{(1)}$, Eqs. (42) and (43) become

$$\hat{\bar{\mathbf{x}}}_{\mathrm{T}} = \arg\min_{\bar{\mathbf{x}}^{(2)}\in\bar{\mathcal{X}}^{(2)}}\ \min_{\bar{\mathbf{x}}^{(1)\mathrm{p}}\in\bar{\mathcal{X}}^{(1)\mathrm{p}}} \|\bar{\mathbf{y}}' - \mathbf{R}_1\bar{\mathbf{x}}^{(1)\mathrm{p}}\|^2 \tag{44}$$

$$\text{subject to } \|\bar{\mathbf{y}}' - \mathbf{R}_1\bar{\mathbf{x}}^{(1)\mathrm{p}}\|^2 \leqslant d^2 \tag{45}$$

where $\bar{\mathcal{X}}^{(1)\mathrm{p}}$ represents the codebook collection $\bar{\mathcal{X}}^{(1)}$ permuted by $\mathbf{P}_1$. Since $\bar{\mathbf{y}}'$ is dependent on $\bar{\mathbf{x}}^{(2)}$, the latter impacts on the decoding results of $\bar{\mathbf{x}}^{(1)\mathrm{p}}$. Every choice of $\bar{\mathbf{x}}^{(2)}$ must be considered in the detection problem in Eq. (44) because of no codebook power constraint. As K is an integer multiple of d_v, vector $\bar{\mathbf{x}}^{(2)}$ encompasses the real-valued codewords of the last $J' = J - \frac{K}{d_v} = J(1 - 1/d_f)$ users. Vector $\bar{\mathbf{x}}^{(2)}$ can take on $M^{J'}$ different values all of which will be tested during the decoding. It is noticeable that as the overloading factor increases (hence, d_f increases), J' gets larger and larger. For heavy overloading scenarios, the exhaustive search in GSD-SCMA will approach that in ML detector.

Fig. 2 shows the flowchart of GSD-SCMA. We initialize a sequence $\mathcal{L}_2 = \left(\mathbf{l}_1, \mathbf{l}_2, \cdots, \mathbf{l}_{M^{J'}}\right)$ which contains all the $M^{J'}$ possible values of vector $\bar{\mathbf{x}}^{(2)}$. GSD-SCMA performs SD for $\bar{\mathbf{x}}^{(1)\mathrm{p}}$ based on every element in the sequence $\mathcal{L}_2$. For a given element in the sequence $\mathcal{L}_2$, namely $\mathbf{l}_t$ ($1 \leqslant t \leqslant M^{J'}$), if a solution of $\bar{\mathbf{x}}^{(1)\mathrm{p}}$ can be found, the value of $\bar{\mathbf{x}}^{(1)\mathrm{p}}$ and $\bar{\mathbf{x}}^{(2)}$ are updated as

$$\bar{\mathbf{x}}^{(1)\mathrm{p}} = \mathbf{v} \tag{46}$$

and

$$\bar{\mathbf{x}}^{(2)} = \mathbf{l}_t \tag{47}$$

respectively, where $\mathbf{v}$ is an intermediate vector in the tree search storing the updated solution. As the squared radius d^2 gets smaller every time a solution of $\bar{\mathbf{x}}^{(1)\mathrm{p}}$ is found, the current values of $\bar{\mathbf{x}}^{(1)\mathrm{p}}$ and $\bar{\mathbf{x}}^{(2)}$ are better than the previous ones. Therefore, after testing on all $M^{J'}$ values of vector $\bar{\mathbf{x}}^{(2)}$, the best decoding results according to Eqs. (44) and (45) can be found.

Specifically, for a given value $\mathbf{l}_t$ ($1 \leqslant t \leqslant M^{J'}$), since we have $\bar{\mathbf{y}}' = \mathbf{Q}_1^{\mathsf{T}}\bar{\mathbf{y}} - \mathbf{R}_2\mathbf{l}_t$ and the fact that $\mathbf{R}_1$ is diagonal, Eq. (45) can be rewritten as

$$\begin{aligned}d^2 &\geqslant \sum_{i=1}^{2K}\left(\bar{y}_i' - (r_1)_{i,i}\bar{x}_i^{(1)\mathrm{p}}\right)^2 &(48)\\ &= \left(\bar{y}_{2K}' - (r_1)_{2K,2K}\bar{x}_{2K}^{(1)\mathrm{p}}\right)^2\\ &\quad + \left(\bar{y}_{2K-1}' - (r_1)_{2K-1,2K-1}\bar{x}_{2K}^{(1)\mathrm{p}}\right)^2 + \cdots &(49)\end{aligned}$$

where $(r_1)_{i,l}$ is the element on the i-th row and l-th column of matrix $\mathbf{R}_1$. The tree search for $\bar{\mathbf{x}}^{(1)\mathrm{p}}$ is similar to what have explained in the previous subsection. Similarly, at the i-th ($i = 2d_v j,\ j \in \{1, \cdots, \lceil\frac{K}{2d_v}\rceil\}$) layer, the lower and upper bounds of $\bar{x}_i^{(1)\mathrm{p}}$ are computed as

$$l_i = \frac{1}{r_{i,i}}\left(\bar{y}_i' - \sqrt{d^2 - p_{i+2d_v}}\right) \tag{50}$$

and

$$u_i = \frac{1}{r_{i,i}}\left(\bar{y}_i' + \sqrt{d^2 - p_{i+2d_v}}\right). \tag{51}$$

respectively. Accordingly, in GSD-SCMA the branch metric is computed by

$$(c_i)_{i_1} = \sum_{n=i-(2d_v-1)}^{i}\left(\bar{y}_n' - (r_1)_{n,n}\breve{x}_{n-(i-2d_v)}\right)^2 \tag{52}$$

where $\breve{x}_k$ is the k-th entry of vector $\breve{\mathbf{x}}$ defined in Eq. (31). Considering the permutation in SQRD of $\bar{\mathbf{G}}^{(1)}$, the final decoding results of GSD-SCMA is obtained by

$$\hat{\bar{\mathbf{x}}}_{\mathrm{T}} = \left[\begin{array}{c}\mathbf{P}_1\bar{\mathbf{x}}^{(1)\mathrm{p}}\\ \bar{\mathbf{x}}^{(2)}\end{array}\right] \tag{53}$$

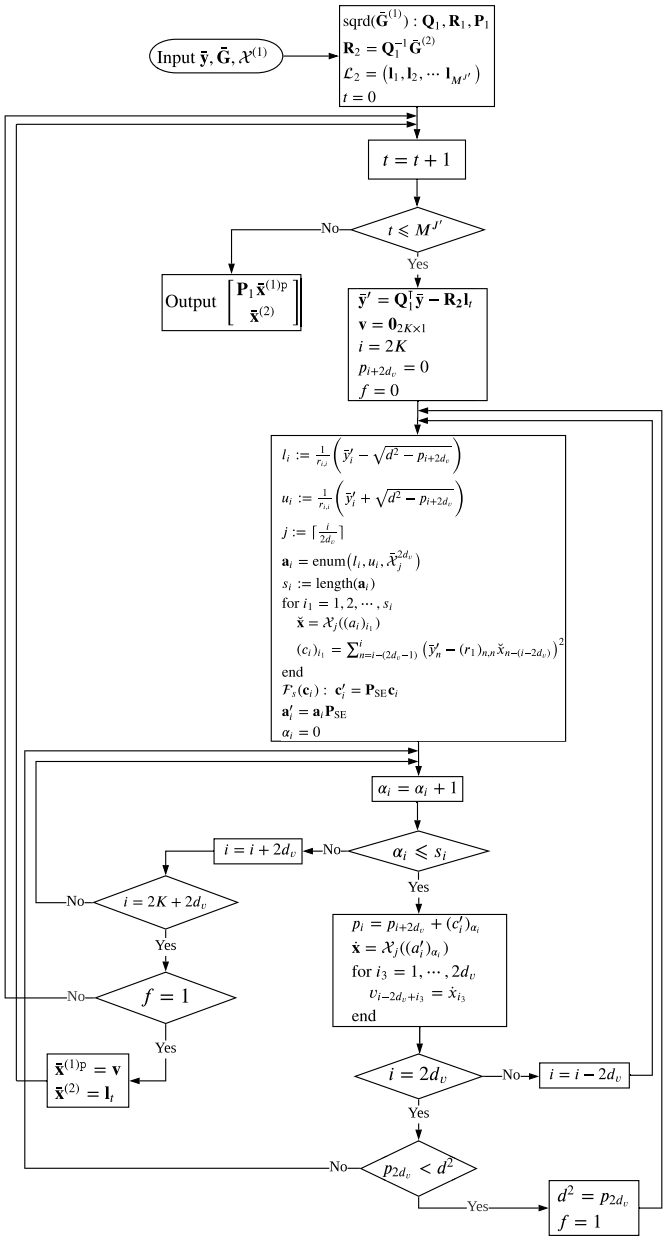


FIGURE 2. Flow chart of GSD-SCMA.

IV. INTRODUCING PRUNING ALGORITHM: A SIMPLE EXAMPLE

In this section, we take a toy example to explain the idea of pruning algorithm, which helps to introduce the proposed pruning algorithms for SCMA in the next section. In this toy example, we consider a real-valued multiuser uplink overloading transmission. Specifically, there are two users transmitting real-valued signals to the same base station, where each end has single antenna. Thus, the received signal is expressed as

$$y = \mathbf{h}\mathbf{x}_{\mathrm{T}} + n = [h_1\ h_2]\begin{bmatrix} x_{\mathrm{T1}} \\ x_{\mathrm{T2}} \end{bmatrix} + n \tag{54}$$

where $\mathbf{h} \in \mathbb{R}^{1\times 2}$ is the channel vector, $\mathbf{x}_{\mathrm{T}} \in \mathbb{R}^{2\times 1}$ is the transmitted signal vector and $n \in \mathcal{N}(0, \sigma_1^2)$ is AWGN. The transmitted symbols x_{T1} and x_{T2} are drawn independently from an identical codebook $\mathcal{C}$ whose cardinality is M_1. We assume that the channel state information (CSI) is known perfectly at the receiver. By applying the GSD rules, the decoding problem of this example system is written as

$$\hat{\mathbf{x}}_{\mathrm{T}} = \arg\min_{x_2\in\mathcal{C}} \arg\min_{x_1\in\mathcal{C}} \|y' - h_1x_1\|^2 \tag{55}$$

$$\text{subject to } \|y' - h_1x_1\|^2 \leqslant d^2 \tag{56}$$

where $y' = y - h_2x_2$, which is equivalent to an SD problem of x_1 based on an exhaustive test of x_2. To reduce the decoding complexity, we adopt a pruning algorithm aiming at reducing the number of tested x_2. We denote $\mathcal{C}^*$ as the reduced candidate sequence of x_2. Without considering the additive noise, the pruning algorithm builds the sequence $\mathcal{C}^*$ in an efficient

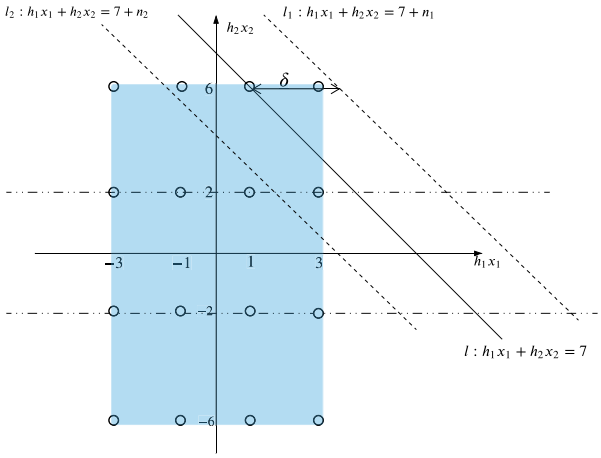


FIGURE 3. Decoding diagram of the transmission example.

way, given as

$$x_2 \in \mathcal{C} \tag{57}$$

$$\text{subject to} \quad \min(h_1x_1) \leqslant |y - h_2\hat{x}_2| \leqslant \max(h_1x_1) \tag{58}$$

Considering the sign of h_1, Eq. (58) can be rewritten as

$$\begin{cases} h_1 \min(x_1) \leqslant |y - h_2\hat{x}_2| \leqslant h_1 \max(x_1), \ h_1 > 0 \\ h_1 \max(x_1) \leqslant |y - h_2\hat{x}_2| \leqslant h_1 \min(x_1), \ h_1 \leqslant 0 \end{cases} \tag{59}$$

After applying the pruning algorithm, the decoding problem becomes

$$\hat{\mathbf{x}}_\mathrm{T} = \arg \min_{x_2 \in \mathcal{C}^*} \arg \min_{x_1 \in \mathcal{C}} \|y' - h_1x_1\|^2 \tag{60}$$

$$\text{subject to } \|y' - h_1x_1\|^2 \leqslant d^2 \tag{61}$$

In the following, we explain the pruning algorithm geometrically. As an example, consider that the channel vector is $\mathbf{h} = \begin{bmatrix} 1 & -2 \end{bmatrix}$ and the transmitted symbol vector is $\mathbf{x}_\mathrm{T} = \begin{bmatrix} 1 & -3 \end{bmatrix}^\mathsf{T}$, whose entries are drawn independently from the PAM-4 codebook $\mathcal{C}$. The received constellation points transformed by the channel vector are illustrated in Fig. 3. The constellation region is marked blue where all points lie. The received signal without any noise is $\mathbf{h}\mathbf{x}_\mathrm{T} = 7$ depicted as line l in the figure. Ideally, there is no noise and the pruning criterion is that the value of h_2x_2 whose geometric representation does not have an intersection point with the line l in the constellation region is eliminated. For example, as the line of $h_2x_2 = 2$ has an intersection point with the line l that is outside of the constellation region, $x_2 = -1$ is pruned from the candidate sequence $\mathcal{C}^*$.

Because of the existence of noise n, the received signal in practice can be represented by line l_1 (l_2) which is horizontally shifted by the noise n_1 (n_2) as shown in Fig. 3. If the received signal can be represented as l_1, it has no intersection point with the line $h_2x_2 = 6$ in the constellation region, while line l meets the line $h_2x_2 = 6$ in the constellation region. This implies that when the original transmission signal, line l, is impaired by noise becoming l_1, the value $x_2 = -3$, which is transmitted, will be pruned by accident. Considering the additive noise in practice, the sequence $\mathcal{C}^*$ built by the pruning algorithm is refined to

$$x_2 \in \mathcal{C} \tag{62}$$

$$\text{subject to} \quad \begin{cases} h_1 \min(x_1) - \delta \leqslant |y - h_2\hat{x}_2| \leqslant h_1 \max(x_1) + \delta, \ h_1 > 0 \\ h_1 \max(x_1) - \delta \leqslant |y - h_2\hat{x}_2| \leqslant h_1 \min(x_1) + \delta, \ h_1 \leqslant 0 \end{cases} \tag{63}$$

where δ is a positive constant that is calculated based on the additive noise level. As the additive noise n follows $\mathcal{N}(0, \sigma_1^2)$, we define the error detection probability p guaranteeing that

$$p = Q(\frac{\delta}{\sigma_1}) \tag{64}$$

where $Q(\cdot)$ is the Q-function. Consequently, δ is computed by

$$\delta = \mathcal{F}^{-1}(1 - p, 0, \sigma_1) \tag{65}$$

where $\mathcal{F}^{-1}(x, \mu, \sigma')$ denotes the inverse cumulative distribution function (CDF) of the Gaussian distribution with mean μ and standard deviation σ'. Since function $\mathcal{F}^{-1}(\cdot)$ is monotonically increasing, the smaller p is, the larger δ is. The increase of δ can bring decoding performance gain but high computational complexity. Thus, there is a trade-off between the decoding performance and complexity when choosing the value of p. Normally, p is set to a decimal that is very close to 0.

In the following, we will discuss the impact of p on the symbol error rate (SER) performance of the transmission example. The SER is defined as $Pr\{\hat{\mathbf{x}}_\mathrm{T} \neq \mathbf{x}_\mathrm{T}\}$. We define an error probability $Pr\{x_{\mathrm{T}2} \notin \mathcal{C}^*\}$ describing the probability that the transmitted codeword $x_{\mathrm{T}2}$ is not included in the reduced sequence $\mathcal{C}^*$. SER of the proposed pruning detector can be written as

$$Pr\{\hat{\mathbf{x}}_\mathrm{T} \neq \mathbf{x}_\mathrm{T}\} = P_\mathrm{ML}\big(1 - Pr\{x_{\mathrm{T}2} \notin \mathcal{C}^*\}\big) + Pr\{x_{\mathrm{T}2} \notin \mathcal{C}^*\} \tag{66}$$

where P_ML is the SER of the ML detector. If the term $Pr\{x_{\mathrm{T}2} \notin \mathcal{C}^*\}$ is very close to zero, the above equation can be approximated to

$$Pr\{\hat{\mathbf{x}}_\mathrm{T} \neq \mathbf{x}_\mathrm{T}\} \approx P_\mathrm{ML} + Pr\{x_{\mathrm{T}2} \notin \mathcal{C}^*\} \tag{67}$$

In high SNR regime, as P_ML approaches to 0, the SER can be formulated as

$$\lim_{\mathrm{SNR} \to \infty} Pr\{\hat{\mathbf{x}}_\mathrm{T} \neq \mathbf{x}_\mathrm{T}\} = Pr\{x_{\mathrm{T}2} \notin \mathcal{C}^*\} \tag{68}$$

In this transmission example, each real dimension has M_1 distinct projections, half positive and half negative. For the $M_1/2$ positive projections, we assume that only the decoding accuracy of the projection on the edge, the maximum one, is affected by the pruning algorithm due to the additive noise. So do the negative projections. Since the pruning is performed only once in the example, $Pr\{x_{\mathrm{T}2} \notin \mathcal{C}^*\}$ is calculated by

$$Pr\{x_{\mathrm{T}2} \notin \mathcal{L}_2^*\} = \frac{p}{M_1/2} = \frac{2p}{M_1} \tag{69}$$

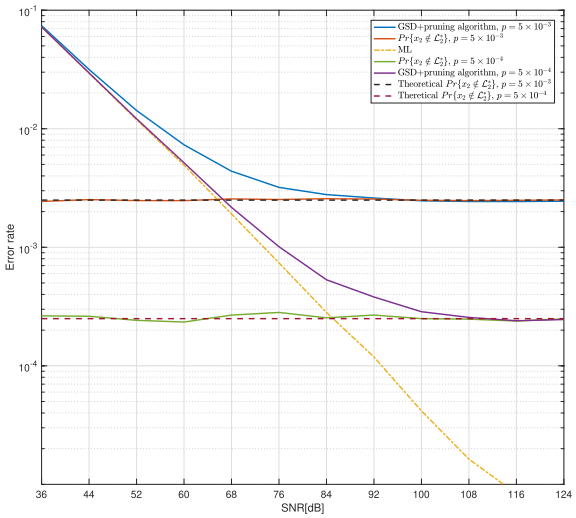


FIGURE 4. SER comparison of the example transmission.

which has positive correlation with p and negative correlation with M_1.

To verify the accuracy of Eq. (69), we simulate the SER performance of the transmission example with codebook PAM-4. Fig. 4 shows the error rate of different detectors in the transmission example. The Monte Carlo simulation of SER and $Pr\{x_{\mathrm{T}2} \notin \mathcal{C}^*\}$ is illustrated in the figure. In high SNR regime, the SER performance of the proposed pruning detector follows Eq. (68). That is, $Pr\{x_{\mathrm{T}2} \notin \mathcal{C}^*\}$ is the error floor of the GSD detector with the proposed pruning algorithm. To verify the effectiveness of Eq. (69), we draw the theoretical performance of $Pr\{x_{\mathrm{T}2} \notin \mathcal{C}^*\}$ in the same figure. As shown in Fig. 4, the simulation performance fits the theoretical analysis in Eq. (69) perfectly.

V. PROPOSED SGSD-SCMA AND PRUNING ALGORITHMS

Recall that, since the GSD-SCMA detector performs an exhaustive test of $M^{J'}$ different values of $\bar{\mathbf{x}}^{(2)}$, where $J' = J(1 - 1/d_f)$, the computational complexity of GSD-SCMA approaches that of ML when overloading factor is large. To efficiently decrease the computational complexity of GSD-SCMA, it is sensible to consider a reduced sequence $\mathcal{L}_2^*$ for $\bar{\mathbf{x}}^{(2)}$ values. Motivated by the pruning methodology demonstrated in Section IV, we propose SGSD-SCMA, the GSD-SCMA detector simplified by applying a pruning algorithm for $\bar{\mathbf{x}}^{(2)}$ values.

The proposed SGSD-SCMA consists of two steps described as follows:

- Step1: A pruning algorithm obtains the reduced sequence $\mathcal{L}_2^*$, which contains potentially good values of vector $\bar{\mathbf{x}}^{(2)}$. The pruning algorithm for SGSD-SCMA follows the same principle of the pruning criterion illustrated by the transmission example in Section IV, but with the particularities of the SCMA transmission.
- Step2: The SD of $\bar{\mathbf{x}}^{(1)}$ is performed based on the sequence $\mathcal{L}_2^*$, which is similar with that of GSD-SCMA. The flowchart of SGSD-SCMA is the same as in Fig. 2, except for $\mathcal{L}_2$ substituted with $\mathcal{L}_2^*$ and the condition $t \leqslant M^{J'}$ replaced by $t \leqslant L_1$ where L_1 is the size of $\mathcal{L}_2^*$ with $L_1 < M^{J'}$.

Therefore, the decoding problem of SGSD-SCMA becomes

$$\hat{\bar{\mathbf{x}}}_{\mathrm{T}} = \arg \min_{\bar{\mathbf{x}}^{(2)} \in \mathcal{L}_2^*} \min_{\bar{\mathbf{x}}^{(1)\mathrm{p}} \in \bar{\mathcal{X}}^{(1)\mathrm{p}}} \|\bar{\mathbf{y}}' - \mathbf{R}_1 \bar{\mathbf{x}}^{(1)\mathrm{p}}\|^2 \tag{70}$$

$$\text{subject to } \|\bar{\mathbf{y}}' - \mathbf{R}_1 \bar{\mathbf{x}}^{(1)\mathrm{p}}\|^2 \leqslant d^2 \tag{71}$$

where $\bar{\mathbf{y}}' = \mathbf{Q}_1^{\mathsf{T}} \bar{\mathbf{y}} - \mathbf{R}_2 \bar{\mathbf{x}}^{(2)}$.

In the following, we propose two pruning algorithms, namely PRUN1 and PRUN2, for SGSD-SCMA. PRUN1 is a general algorithm applicable to any kind of SCMA codebooks, while PRUN2 algorithm requires a constraint on the codebook factor graph to work. In the sequel, SGSD-SCMA detector with the PRUN1 algorithm is referred as SGSD1-SCMA detector and the SGSD-SCMA detector with the PRUN2 algorithm is named SGSD2-SCMA. Besides, a theoretical analysis of the CER of the SGSD-SCMA is presented. Please note that the two proposed pruning algorithms are applicable to both regular and irregular SCMA codebooks. For simplicity, the following description of the proposed pruning algorithms are based on the assumption of the regular SCMA codebooks.

A. THE PROPOSED PRUN1 ALGORITHM

As SCMA is a multiple access transmission, a similar pruning process explained in Section IV can be employed in every dimension of the received signal $\bar{\mathbf{y}}$. Recalling that matrix $\mathbf{R}_1$ has positive diagonal, the pruning algorithm for SGSD-SCMA builds the sequence $\mathcal{L}_2^*$ given as

$$\bar{\mathbf{x}}^{(2)} \in \bar{\mathcal{X}}^{(2)} \tag{72}$$

$$\text{subject to}$$

$$(r_1)_{i,i} \min\left((\bar{\mathcal{X}}_{j_1}^{(1)\mathrm{p}})^{i-2d_v(j_1-1)}\right) - \delta \leqslant \|\bar{y}_i - (\mathbf{r}_2)_i \bar{\mathbf{x}}^{(2)}\| \leqslant (r_1)_{i,i} \max\left((\bar{\mathcal{X}}_{j_1}^{(1)\mathrm{p}})^{i-2d_v(j_1-1)}\right) + \delta, \ \forall i \in \{1, \cdots, 2K\} \tag{73}$$

where $j_1 = \lceil \frac{i}{2d_v} \rceil$, $(r_1)_{i,i}$ denotes the i-th diagonal element of square matrix $\mathbf{R}_1$, $(\bar{\mathcal{X}}_j^{(1)\mathrm{p}})^k$ is the real sequence consisting of the k-th entry of all the elements of sequence $\bar{\mathcal{X}}_j^{(1)\mathrm{p}}$ and $(\mathbf{r}_2)_i$ denotes the i-th row vector of matrix $\mathbf{R}_2$. As explained in Section IV, δ is determined by the additive noise and predefined error detection probability p. In this case, δ is calculated by

$$\delta = \mathcal{F}^{-1}(1 - p, 0, \sigma) \tag{74}$$

Before diving into the details of PRUN1 algorithm, we want to illustrate the sparse nature of matrix $\mathbf{R}_2$ which helps to simplify the algorithm. Because $\mathbf{Q}$ is diagonal and $\bar{\mathbf{G}}^{(2)}$ is sparse due to the factor graph, $\mathbf{R}_2$ is also sparse according to Eq. (39). For instance, when the codebook factor graph matrix is Eq. (3), the matrix $\mathbf{R}_2$ has

the following structure

$$\mathbf{R}_2 = \begin{bmatrix} 0\,0\,0\,0\,\times\,\times\,0\,0\,\times\,\times\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,\times\,\times\,0\,0\,\times\,\times\,0\,0\,0\,0\,0\,0 \\ \times\,\times\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,\times\,\times\,0\,0 \\ \times\,\times\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,\times\,\times\,0\,0 \\ 0\,0\,0\,0\,0\,0\,\times\,\times\,0\,0\,0\,0\,0\,0\,\times\,\times \\ 0\,0\,0\,0\,0\,0\,\times\,\times\,0\,0\,0\,0\,0\,0\,\times\,\times \\ 0\,0\,\times\,\times\,0\,0\,0\,0\,0\,0\,\times\,\times\,0\,0\,0\,0 \\ 0\,0\,\times\,\times\,0\,0\,0\,0\,0\,0\,\times\,\times\,0\,0\,0\,0 \end{bmatrix} \tag{75}$$

where $\times$ represents a non-zero element in the matrix. Thanks to the sparsity of $\mathbf{R}_2$ illustrated above, for a given row, we only consider the elements of $\bar{\mathbf{x}}^{(2)}$ that are associated with the non-zero elements of $\mathbf{R}_2$. There are always $2(d_f-1)$ non-zero elements in each row vector $(\mathbf{r}_2)_i$ ($i \in \{1, \cdots, 2K\}$) which correspond to the same user in pairs. Therefore, instead of testing every value of $\bar{\mathbf{x}}^{(2)}$ ($\bar{\mathbf{x}}^{(2)} \in \bar{\mathcal{X}}^{(2)}$), we can test M^{d_f-1} combinations of those elements of $\bar{\mathbf{x}}^{(2)}$ that are associated with the non-zero elements of $\mathbf{R}_2$ at each row. Moreover, another property of matrix $\mathbf{R}_2$ is that the $(2k-1)$-th and $2k$-th ($1 \leqslant k \leqslant K$) rows of $\mathbf{R}_2$ have the same indices of non-zero elements. Thus, we perform the pruning operation of two rows, the $(2k-1)$-th and $2k$-th, at once. Accordingly, there will be K pruning operations for the PRUN1 algorithm in total. As $\bar{\mathbf{x}}^{(2)}$ contains the real-valued codewords of J' users, we define vector $\mathbf{v} \in \{1, \cdots, M\}^{J'\times 1}$ to efficiently represent the values of vector $\bar{\mathbf{x}}^{(2)}$. Every entry of $\mathbf{v}$ is the codeword index of the corresponding user. To simplify the process, PRUN1 algorithm outputs the reduced sequence Υ of values of vector $\mathbf{v}$. The sequence $\mathcal{L}_2^*$ can be constructed based on the sequence Υ after the PRUN1 algorithm and the size of the two sequences are the same. The PRUN1 algorithm is summarized in Algorithm 2.

In the following, we elaborate on the PRUN1 algorithm. The set $\mathcal{V}$ stores the indices of users that have been visited. The sequence Υ of size L_1 is the output of PRUN1 algorithm. Similar to the tree search, the PRUN1 algorithm starts by performing pruning from the last two rows of $\mathbf{R}_2$ and moves upwards. At each pruning operation, d_f-1 users associated with the non-zero elements are visited. These users can be divided into the revisited users and newly visited users. For simplicity, the newly visited users are referred to as new users in sequel. At the k-th ($1 \leqslant k \leqslant K$) pruning operation, function UserPart$((\mathbf{r}_2)_{2k}, \mathcal{V})$ outputs the partitioned users and their corresponding indices. Specifically, the sequence $\mathcal{A}$ stores the indices of non-zero elements of $(\mathbf{r}_2)_{2k}$ and the sequence $\mathcal{B}$ stores the indices of users associated with the elements in sequence $\mathcal{A}$. We define the set $\mathcal{C}$ to store the currently visited users. Thus, the set $\mathcal{C}$ consists of the d_f-1 distinct elements in sequence $\mathcal{B}$. Sets $\mathcal{U}$ and $\mathcal{W}$ store the indices of new users and revisited users, respectively. The sequence $\mathcal{A}_1$ stores the indices in the sequence $\mathcal{A}$ that are associated with the new users and the sequence $\mathcal{A}_2$ stores the indices in $\mathcal{A}$ that associated with the revisited users. Besides, sequence $\mathcal{B}_1$ and $\mathcal{B}_2$ contain the indices of users associated to the elements in $\mathcal{A}_1$ and $\mathcal{A}_2$, respectively. Afterwards, the number of new users l_1 and the number of revisited users l_2 are obtained.

Algorithm 2 The PRUN1 Algorithm

Input: $\bar{\mathbf{y}}$, $\mathbf{R}_1$, $\mathbf{R}_2$, $\mathcal{X}^{(1)}$ and $\mathcal{X}^{(2)}$
Output: Υ

Initialization: $\mathcal{V}=\emptyset$, $\Upsilon=(\mathbf{0}_{J'\times 1})$ and $L_1=1$.
for $k = K, K-1, \cdots, 1$ **do**
  $\mathcal{U}, \mathcal{W}, \mathcal{A}_1, \mathcal{A}_2, \mathcal{B}_1, \mathcal{B}_2 = \text{UserPart}((\mathbf{r}_2)_{2k}, \mathcal{V})$
  $l_1 = \text{size}(\mathcal{U})$
  $l_2 = d_f - 1 - l_1$
  if $\mathcal{W} \neq \emptyset$ **then**
    for $i_1 = 1, 2, \cdots, L_1$ **do**
      Obtain $\mathbf{q}$ using Eq. (76).
      Calculate $t_1(i_1)$ and $t_2(i_1)$ based on Eq. (78) and Eq. (77), respectively.
    end for
  else
    $t_1 = t_2 = 0$
  end if
  $\Upsilon' = \emptyset$
  $\mathbf{P} = \mathcal{F}_1(M, l_1)$
  $c_1 = 0$
  for $i_2 = 1, 2, \cdots, M^{l_1}$ **do**
    Calculate $s_1(i_2)$ and $s_2(i_2)$ using Eq. (80) and Eq. (81), respectively.
    for $i_1 = 1, 2, \cdots, L_1$ **do**
      Calculate ξ_1 and ξ_2 using Eqs. (85) and (86), respectively.
      if Eq. (82) and Eq. (83) are satisfied **then**
        $\mathbf{v} = \Upsilon(i_1)$
        for $i = 1, 2, \cdots, l_1$ **do**
          $v_{\mathcal{B}_1(i)} = (p_{i_2})_i$
        end for
        $c_1 = c_1 + 1$
        $\Upsilon'_i(c_1) = \mathbf{v}$
      end if
    end for
  end for
  $\Upsilon = \Upsilon'_i$
  $L_1 = c_1$,
  $\mathcal{V} = \mathcal{V} \bigcup \mathcal{U}$
end for

For the sake of reducing computational complexity, PRUN1 algorithm performs the computations concerning the two kinds of users separately. Because PRUN1 carries out the K pruning operations in sequence, the indices combinations of the revisited users are those stored in the sequence Υ. At the k-th ($1 \leqslant k \leqslant K$) pruning operation, if set $\mathcal{W}$ is not empty, for the i_1-th ($1 \leqslant i_1 \leqslant L_1$) element in sequence Υ, the codeword indices of the revisited users constitute the vector $\mathbf{q}$, given as

$$\mathbf{q} = \Upsilon^{\mathcal{W}}(i_1) \tag{76}$$

where $\Upsilon^{\mathcal{W}}(i_1)$ denotes the entries with indices in set $\mathcal{W}$ of the i_1-th element of sequence Υ. Then, we calculate two

Function: UserPart $((\mathbf{r}_2)_{2k}, \mathcal{V})$
$\mathcal{C} = \emptyset$
$i = 0$
for $j = 1, \cdots, 2Jdv - 2K$ **do**
  if $(r_2)_{2k,j} = 1$ **then**
    $i = i + 1$
    $\mathcal{A}(i) = j$
    $\mathcal{B}(i) = \lceil \frac{\mathcal{A}(i)}{2d_v} \rceil$
    $\mathcal{C} = \mathcal{C} \cup \{\mathcal{B}(i)\}$
  end if
end for
$\mathcal{U} = \mathcal{C} \setminus \mathcal{V}$
$\mathcal{W} = \mathcal{C} \setminus \mathcal{U}$
$i_1 = i_2 = 0$
for $i = 1, 2, \cdots, 2(d_f - 1)$ **do**
  if $\mathcal{B}(i) \in \mathcal{U}$ **then**
    $i_1 = i_1 + 1$
    $\mathcal{B}_1(i_1) = \mathcal{B}(i)$
    $\mathcal{A}_1(i_1) = \mathcal{A}(i)$
  else
    $i_2 = i_2 + 1$
    $\mathcal{B}_2(i_2) = \mathcal{B}(i)$
    $\mathcal{A}_2(i_2) = \mathcal{A}(i)$
  end if
end for
return $\mathcal{U}, \mathcal{W}, \mathcal{A}_1, \mathcal{A}_2, \mathcal{B}_1, \mathcal{B}_2$.

intermediates regarding the revisited users, $t_1(i_1)$ and $t_2(i_1)$, as

$$t_1(i_1) = \sum_{l=1}^{2l_2} (r_2)_{2k-1,\mathcal{A}_2(l)} \bar{\mathcal{X}}_{\frac{K}{d_v}+\mathcal{B}_2(l)}^{\mathcal{A}_2(l)-2d_v(\mathcal{B}_2(l)-1)}(q_l) \quad (77)$$

and

$$t_2(i_1) = \sum_{l=1}^{2l_2} (r_2)_{2k,\mathcal{A}_2(l)} \bar{\mathcal{X}}_{\frac{K}{d_v}+\mathcal{B}_2(l)}^{\mathcal{A}_2(l)-2d_v(\mathcal{B}_2(l)-1)}(q_l) \quad (78)$$

respectively, where $(r_2)_{i,l}$ denotes the element in the i-th row and the l-th column of matrix $\mathbf{R}_2$, $\bar{\mathcal{X}}_j^a(b)$ refers to the a-th entry of the b-th element of the sequence $\bar{\mathcal{X}}_j$, which is a scalar. For the new users, all M^{l_1} combinations of the associated codewords are tested. To enable the codeword enumeration of l_1 new users, we utilize the function $\mathcal{F}_1(M, l_1)$ to generate the matrix $\mathbf{P} = [\mathbf{p}_1, \cdots, \mathbf{p}_{M^{l_1}}] \in \{1, 2, \cdots, M\}^{l_1 \times M^{l_1}}$. If $M = 4$ and $l_1 = 2$, the matrix $\mathbf{P}$ is written as

$$\begin{aligned}\mathbf{P} &= \mathcal{F}_1(4, 2) \\ &= \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 4 & 4 & 4 & 4 \\ 1 & 2 & 3 & 4 & \cdots & 1 & 2 & 3 & 4 \end{bmatrix}\end{aligned} \quad (79)$$

Each column of $\mathbf{P}$ is a codeword indices combination. For the i_2-th $(1 \leqslant i_2 \leqslant M^{l_1})$ combination, $\mathbf{p}_{i_2}$ is used to calculate two intermediates, $s_1(i_2)$ and $s_2(i_2)$, as

$$s_1(i_2) = \bar{y}_{2k-1} - \sum_{l=1}^{2l_1} (r_2)_{2k-1,\mathcal{A}_1(l)} \times \bar{\mathcal{X}}_{\frac{K}{d_v}+\mathcal{B}_1(l)}^{\mathcal{A}_1(l)-2d_v(\mathcal{B}_1(l)-1)}((p_{i_2})_{\lceil \frac{l}{2} \rceil}) \quad (80)$$

and

$$s_2(i_2) = \bar{y}_{2k} - \sum_{l=1}^{2l_1} (r_2)_{2k,\mathcal{A}_1(l)} \times \bar{\mathcal{X}}_{\frac{K}{d_v}+\mathcal{B}_1(l)}^{\mathcal{A}_1(l)-2d_v(\mathcal{B}_1(l)-1)}((p_{i_2})_{\lceil \frac{l}{2} \rceil}) \quad (81)$$

respectively. Based on Eqs. (72) and (73), at the k-th pruning operation, for a given i_1-th element in sequence Υ, the pruning criterion keeps the vectors $\mathbf{p}_{i_2}$ $(1 \leqslant i_2 \leqslant M^{l_1})$ that jointly satisfy the conditions

$$\begin{aligned}(r_1)_{2k-1,2k-1} \min\left((\bar{\mathcal{X}}_j^{(1)\mathrm{p}})^{2k-1-2d_v(j-1)}\right) - \delta \leqslant \xi_1 \\ \leqslant (r_1)_{2k-1,2k-1} \max\left((\bar{\mathcal{X}}_j^{(1)\mathrm{p}})^{2k-1-2d_v(j-1)}\right) + \delta\end{aligned} \quad (82)$$

and

$$\begin{aligned}(r_1)_{2k,2k} \min\left((\bar{\mathcal{X}}_j^{(1)\mathrm{p}})^{2k-2d_v(j-1)}\right) - \delta \leqslant \xi_2 \\ \leqslant (r_1)_{2k,2k} \max\left((\bar{\mathcal{X}}_j^{(1)\mathrm{p}})^{2k-2d_v(j-1)}\right) + \delta\end{aligned} \quad (83)$$

where

$$j = \lceil \frac{k}{d_v} \rceil \quad (84)$$

$$\xi_1 = s_1(i_2) - t_1(i_1) \quad (85)$$

$$\xi_2 = s_2(i_2) - t_2(i_1) \quad (86)$$

and δ is calculated according to Eq. (74). If vectors $\mathbf{p}_{i_2}$ and $\Upsilon(i_1)$ satisfy the conditions Eq. (82) and Eq. (83) jointly, they are used to construct the elements in sequence Υ.

We provide a performance analysis of the SGSD1-SCMA detector considering the error detection probability p as a factor. Similar to Eqs. (66) to (68), CER of SGSD1-SCMA can be approximated to

$$Pr\{\hat{\bar{\mathbf{x}}}_{\mathrm{T}} \neq \bar{\mathbf{x}}_{\mathrm{T}}\} \approx P_{\mathrm{ML}} + Pr\{\bar{\mathbf{x}}_{\mathrm{T}}^{(2)} \notin \mathcal{L}_2^*\} \quad (87)$$

where P_{ML} is the CER of the ML detector Eq. (7) and $Pr\{\bar{\mathbf{x}}_{\mathrm{T}}^{(2)} \notin \mathcal{L}_2^*\}$ is the error probability of the PRUN1 algorithm. At high SNR values, since P_{ML} approaches to zero, CER can be written as

$$\lim_{\mathrm{SNR}\to\infty} Pr\{\hat{\bar{\mathbf{x}}}_{\mathrm{T}} \neq \bar{\mathbf{x}}_{\mathrm{T}}\} = Pr\{\bar{\mathbf{x}}_{\mathrm{T}}^{(2)} \notin \mathcal{L}_2^*\} \quad (88)$$

At the k-th pruning operation, the error probability is computed as

$$p_k = \frac{2p}{M_1}\left(1 - \frac{2p}{M_1}\right)^{k-1} \quad (89)$$

where M_1 is the average number of distinct projections in a real dimension of the SCMA codebook. In this work, M_1 is approximated to $\sqrt{M}$. Because there are always K sequential pruning operations, the error probability of PRUN1 algorithm is calculated by

$$Pr\{\bar{\mathbf{x}}_{\mathrm{T}}^{(2)} \notin \mathcal{L}_2^*\}_1 = \sum_{k=1}^{K} p_k \quad (90)$$

$$= \sum_{k=1}^{K} \frac{2p}{M_1}\left(1-\frac{2p}{M_1}\right)^{k-1} \quad (91)$$

$$= 1-\left(1-\frac{2p}{M_1}\right)^{K} \quad (92)$$

B. THE PROPOSED PRUN2 ALGORITHM

In this subsection, we introduce another pruning algorithm called PRUN2 which simplifies the process by considering certain factor graph constraint of the codebook. The PRUN2 algorithm can be regarded as a simplified version of the PRUN1 algorithm.

A major difference between the two pruning algorithms for SCMA is that there is no revisited user at a given pruning operation in PRUN2 algorithm. To satisfy this condition, at any pruning operation, the set $\mathcal{C}$ containing the currently visiting users and the set $\mathcal{V}$ storing the users that have been visited must be disjoint. Besides, the PRUN2 algorithm stops the pruning operation once all J' users associated with $\mathbf{R}_2$ have been visited. Therefore, the PRUN2 algorithm only carries out $K' = \frac{J'_i}{d_f-1} = \frac{K}{d_v}$ $(K' < K)$ pruning operations. If we can rewrite the factor graph matrix $\mathbf{F}$ as

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}^{(1)} & \mathbf{F}^{(2)} \end{bmatrix} \quad (93)$$

where $\mathbf{F}^{(1)} = [\mathbf{f}_1, \cdots, \mathbf{f}_{\frac{K}{d_v}}]$ and $\mathbf{F}^{(2)} = [\mathbf{f}_{\frac{K}{d_v}+1}, \cdots, \mathbf{f}_J]$, the requirement of the PRUN2 algorithm can be met by the fact that the last K' rows of matrix $\mathbf{F}^{(2)}$ are mutually orthogonal. Defining matrix $\mathbf{F}_s^{(2)}$ containing the last K' rows of matrix $\mathbf{F}^{(2)}$, the constraint of the PRUN2 algorithm can be expressed as

$$\mathbf{F}_s^{(2)}\mathbf{F}_s^{(2)\mathsf{T}} = \mathbf{I}_{K'} \quad (94)$$

Algorithm 3 concludes the PRUN2 algorithm and we will introduce the details in the following. At a pruning operation, the algorithm obtains sequence $\mathcal{A}$, sequence $\mathcal{B}$ and set $\mathcal{C}$ same as the PRUN1 algorithm. In the PRUN2 algorithm, as the set $\mathcal{C}$ only contains new users, we need to test M^{d_f-1} codeword combinations of the d_f-1 new users. Thus, we use function $\mathcal{F}_1$ to generate indices matrix $\mathbf{P}$ as

$$\mathbf{P} = \mathcal{F}_1(M, d_f - 1) \quad (95)$$

The pruning criterion retains the vectors $\mathbf{p}_i$ $(1 \leqslant i \leqslant M^{d_f-1})$ satisfying the condition in Eqs. (82) and (83) jointly. As $\mathcal{C}$ has only new users, the intermediates ξ_1 and ξ_2 in the PRUN2 algorithm are computed by

$$\xi_1 = \bar{y}_{2k-1} - \sum_{l=1}^{2(d_f-1)} (r_2)_{2k-1,\mathcal{A}(l)} \times \bar{\mathcal{X}}^{\mathcal{A}(l)-2d_v(\mathcal{B}(l)-1)}_{\frac{K}{d_v}+\mathcal{B}(l)}((p_i)_{\lceil \frac{l}{2} \rceil}) \quad (96)$$

and

$$\xi_2 = \bar{y}_{2k} - \sum_{l=1}^{2(d_f-1)} (r_2)_{2k,\mathcal{A}(l)} \times \bar{\mathcal{X}}^{\mathcal{A}(l)-2d_v(\mathcal{B}(l)-1)}_{\frac{K}{d_v}+\mathcal{B}(l)}((p_i)_{\lceil \frac{l}{2} \rceil}) \quad (97)$$

respectively. The qualified vectors $\mathbf{p}_i$ are used to create the elements of sequence Υ.

Algorithm 3 The PRUN2 Algorithm

Input: $\bar{\mathbf{y}}$, $\mathbf{R}_1$, $\mathbf{R}_2$, $\mathcal{X}^{(1)}$ and $\mathcal{X}^{(2)}$
Output: Υ
Initialization: $\mathcal{V} = \emptyset$, $\Upsilon = (\mathbf{0}_{J'\times 1})$ and $L_1 = 1$.
for $k = K, K-1, \cdots, K-K'+1$ **do**
  $\mathcal{C} = \emptyset$
  $i = 0$
  for $j = 1, \cdots, 2Jdv - 2K$ **do**
    if $(r_2)_{2k,j} = 1$ **then**
      $i = i + 1$
      $\mathcal{A}(i) = j$
      $\mathcal{B}(i) = \lceil \frac{\mathcal{A}(i)}{2d_v} \rceil$
      $\mathcal{C} = \mathcal{C} \cup \{\mathcal{B}(i)\}$
    end if
  end for
  if $\mathcal{C} \cap \mathcal{V} = \emptyset$ **then**
    $c_1 = 0$
    $\Phi = \emptyset$
    $\mathbf{P} = \mathcal{F}_1(M, d_f - 1)$
    for $i = 1, \cdots, M^{d_f-1}$ **do**
      Calculate ξ_1 and ξ_2 using Eq. (96) and Eq. (97), respectively.
      if Eq. (82) and Eq. (83) are satisfied **then**
        $\mathbf{v} = \mathbf{0}_{J'\times 1}$
        for $n = 1, \cdots, d_f - 1$ **do**
          $v_{\mathcal{B}(2n)} = (p_i)_n$
        end for
        $c_1 = c_1 + 1$
        $\Phi(c_1) = \mathbf{v}$
      end if
    end for
    $\Upsilon' = \Upsilon$
    for $i_1 = 1, 2, \cdots, L_1$ **do**
      for $i_2 = 1, 2, \cdots, c_1$ **do**
        $\mathbf{v} = \Upsilon'(i_1) + \Phi(i_2)$
        $\Upsilon(i_1(L_1 - 1) + i_2) = \mathbf{v}$
      end for
    end for
    $L_1 = L_1 c_1$
    $\mathcal{V} = \mathcal{V} \cup \mathcal{C}$
  end if
end for

The CER of SGSD2-SCMA has the same expression as shown in Eq. (87) expect the error probability of the PRUN2 algorithm is computed as

$$Pr\{\bar{\mathbf{x}}_{\mathrm{T}}^{(2)} \notin \mathcal{L}_2^*\}_2 = \sum_{k'=1}^{K'} \frac{2p}{M_1}\left(1-\frac{2p}{M_1}\right)^{k'-1} \quad (98)$$

$$= 1-\left(1-\frac{2p}{M_1}\right)^{K'} \quad (99)$$

Compared with the PRUN1 algorithm carrying out K pruning operations, the PRUN2 algorithm performs only K' pruning operations. For a codebook compatible with both algorithms, some values of $\mathbf{v}$ satisfying the conditions in the PRUN2 algorithm might be eliminated by the PRUN1

algorithm. As a result, the PRUN1 algorithm leads to a smaller size of $\mathcal{L}_2^*$ than the PREUN2 algorithm reducing the complexity of SGSD-SCMA but has higher error probability, $Pr\{\bar{\mathbf{x}}_{\mathrm{T}}^{(2)} \notin \mathcal{L}_2^*\}_1 > Pr\{\bar{\mathbf{x}}_{\mathrm{T}}^{(2)} \notin \mathcal{L}_2^*\}_2$. This property of the PRUN1 algorithm will be verified by the simulation performance in Section VII.

VI. COMPLEXITY ANALYSIS

In this section, we study the computational complexity of different detectors, categorized to iterative detectors and SD-based detectors. For the first category, only the Max-Log-MPA [11] is considered. Three SD-based detectors are considered in the second category, which are SD-SCMA, GSD-SCMA and SGSD-SCMA. The computational complexity is investigated in terms of floating point operations (FLOPs).

A. ITERATIVE DETECTOR

The Log-MPA and Max-Log-MPA detectors [11] are approximations of the conventional MPA detector, which compute the probability messages in the logarithm domain. Considering that the Max-Log-MPA detector has lower computational complexity than the Log-MPA and MPA detectors while having almost the same BER performance as the MPA detector, we consider Max-Log-MPA detector as the only iterative detector in the complexity analysis and also the simulation comparison in the following section. It follows the message-update rules and exchanges messages from REs (function nodes) to users (variable nodes) and vice versa. The number of iterations of the Max-Log-MPA detector not only impacts the MUD performance but also the computational complexity. Larger number of iterations improves the MUD performance, but brings about higher computational complexity. The number of FLOPs of the Max-Log-MPA detector is $T\left[Kd_f\left(M^{d_f}(9d_f+4)-M\right)+JMd_v^2\right]+J\left(M(d_v+1)-2\right)\log_2(M)$, where T $(T \geqslant 1)$ is the number of iterations.

B. SD-BASED DETECTORS

In contrary to iterative detectors, the numbers of FLOPs of SD-based detectors are stochastic and depend on the transmitted signal, the additive noise and the squared radius. As the complexity of SD-based detectors is relevant to the number of visited nodes in the tree search which is unpredictable, we provide the FLOPs expression of different constituting components of the detectors. The number of FLOPs of these detector are obtained by the Monte-Carlo simulation and will be presented in Section VII.

1) NUMBER OF FLOPs OF SD-SCMA

As demonstrated in Fig. 1, the complexity of SD-SCMA detector comes from the SQRD of the channel matrix $\tilde{\mathbf{G}}$, the matrix multiplication $\mathbf{Q}^{\mathsf{T}}\tilde{\mathbf{y}}$, the tree search and the re-permutation at the output. The number of FLOPs of sqrd($\tilde{\mathbf{G}}$) are $2(2Jd_v)^3+2Jd_v(6Jd_v-1)-J$. The matrix multiplication needs $2Jd_v(4Jd_v-1)$ FLOPs. For the tree search, we list the intermediate calculation complexity in Table 2. For the re-permutation process, it requires $2Jd_v(4çJd_v-1)$ FLOPs.

2) NUMBER OF FLOPs OF GSD-SCMA

For GSD-SCMA, complexity of the calculation of Eq. (40), the computation of $\bar{\mathbf{y}}'=\mathbf{Q}_1^{\mathsf{T}}\bar{\mathbf{y}}-\mathbf{R}_2\bar{\mathbf{x}}^{(2)}$, the tree search process and the re-permutation of the output constitute its computational complexity. The computation of sqrd$\left(\tilde{\mathbf{G}}_1\right)$ needs $16K^3+2K(6K-1)-J'$ FLOPs. The calculation of matrix $\mathbf{R}_2$ takes $(2K)^3+2K(4K-1)(2Jd_v-2K)$ FLOPs. Consequently, the number of FLOPs of the calculation of Eq. (40) is $(2K)^3+2K(8K-1)(2Jd_v+1)-J'$. To avoid repeated computation and decrease complexity, $\mathbf{Q}_1^{\mathsf{T}}\bar{\mathbf{y}}$ is calculated once and stored for reuse. The number of FLOPs of $\mathbf{Q}_1^{\mathsf{T}}\bar{\mathbf{y}}$ are $2K(4K-1)$. For a given value of $\bar{\mathbf{x}}^{(2)}$, the computation of $\bar{\mathbf{y}}'$ needs $2K(4Jd_v-4K)$ FLOPs. Since all $M^{J'}$ values of $\bar{\mathbf{x}}^{(2)}$ are considered in GSD-SCMA, computation of $\bar{\mathbf{y}}'=\mathbf{Q}_1^{\mathsf{T}}\bar{\mathbf{y}}-\mathbf{R}_2\bar{\mathbf{x}}^{(2)}$ needs $2K\left[M^{J'}(4Jd_v-4K)+(4K-1)\right]$ FLOPs in total. Because the matrix $\mathbf{R}_1$ is diagonal, the calculation of u_i and l_i is simplified. Therefore, the number of FLOPs of the intermediate calculations in tree search process for GSD-SCMA is shown in Table 3. For the re-permutation $\mathbf{P}_1\bar{\mathbf{x}}^{(1)\mathrm{p}}$, the complexity is $2K(4K-1)$ FLOPs.

3) NUMBER OF FLOPs OF SGSD-SCMA

The components of SGSD-SCMA's complexity are the same as those of GSD-SCMA. For SGSD-SCMA, the complexity of calculation in Eq. (40) is the same as that in GSD-SCMA. The complexity of $\bar{\mathbf{y}}'=\mathbf{Q}_1^{\mathsf{T}}\bar{\mathbf{y}}-\mathbf{R}_2\bar{\mathbf{x}}^{(2)}$ is affected by the size of $\mathcal{L}_2^*$, which is $2K\left[L_1(4Jd_v-4K)+(4K-1)\right]$ FLOPs. The numbers of FLOPs required for the intermediate calculations of the PRUN1 and PRUN2 algorithm are listed in Table 4 and Table 5, respectively. Because the intermediate computations in tree search for SGSD-SCMA are the same as those for GSD-SCMA, their complexity are presented in Table 3.

VII. NUMERICAL RESULTS

This section investigates the performance of different SCMA detectors in terms of CER, BER and the computational complexity. The performance of the proposed SGSD-SCMA detector is compared with the Max-Log-MPA detector and with SD-based state-of-the-art detectors, i.e. SD-SCMA and GSD-SCMA. In the simulation, the squared radius is initialized as $d^2=50$. To better understand the capability of various detectors, we consider three SCMA codebooks with different settings of the system parameters in the simulation. First, the codebook proposed in [28] with typical benchmark setting $J=6$, $K=4$ and $M=4$ is considered, denoted as CB1. The second codebook [4] has also the typical factor graph setting $J=6$ and $K=4$ but has the codebook size $M=16$, denoted as CB2. Since only the design approach is presented in [4], interested readers can find the CB2 details on this web page [29]. The last one has $J=15$, $K=6$ and $M=4$ introduced in [30] with an overloading factor of 250%, denoted as CB3. The first two codebooks meet the constraint of the PRUN2 algorithm and are compatible with

TABLE 2. The number of FLOPs of the calculations in tree search for SD-SCMA.

Variables	u_i	l_i	$\mathbf{c}_i$	p_i
FLOPs	$2(2Jd_v - i) + 4$	$2(2Jd_v - i) + 4$	$s_i\left[\left(4(2Jd_v - i) + 4d_v + 6\right) - 1\right]$	1

TABLE 3. The number of FLOPs of the calculations in tree search for GSD-SCMA.

Variables	u_i	l_i	$\mathbf{c}_i$
FLOPs	4	4	$s_i(8d_v - 1)$

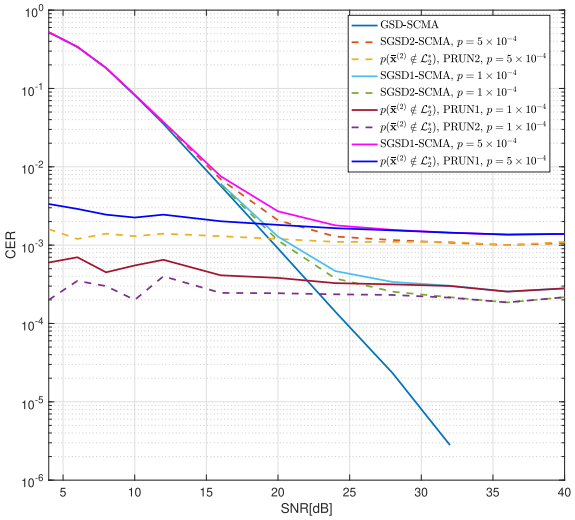


FIGURE 5. CER performance of different detectors when simulating with the CB1.

both two proposed pruning algorithms. As the factor graph of the CB3 does not satisfy the constraint in Eq. (94), only the PRUN1 algorithm is applicable to it. As the compatibility of the investigated algorithms is different, the following results are organized by the considered codebooks. It is noticeable that the three codebooks are designed by different approaches and there is no common structure of them.

A. CB1 ($J = 6, K = 4, M = 4$)

We first study the CER performance of the CB1. As this codebook has constant modulus, all the studied detectors are compatible with it. The SD-SCMA and GSD-SCMA detectors are alternatives to each other. Knowing that SD has the same detection performance as the ML detector, both SD-SCMA and GSD-SCMA detectors have the optimal ML MUD performance. For the sake of simplicity, we only show the performance of the GSD-SCMA detector. Fig. 5 shows the CER performance of various detectors. It is obvious that, the simulated CER follows the limit demonstrated in Eq. (68). Table 6 compares the simulation error rate and the error probability of the CB1. Apart from the case of the PRUN1 algorithm with $p = 1 \times 10^{-4}$, the rest of the simulation performance fits well the theoretical expressions Eqs. (92) and (99) for the proposed pruning algorithms.

Fig. 6 shows the BER performance comparison of different detectors when applying the CB1. The GSD-SCMA and Max-Log-MPA with 3 iterations has almost the same BER performance. It is obvious that for a given pruning algorithm,

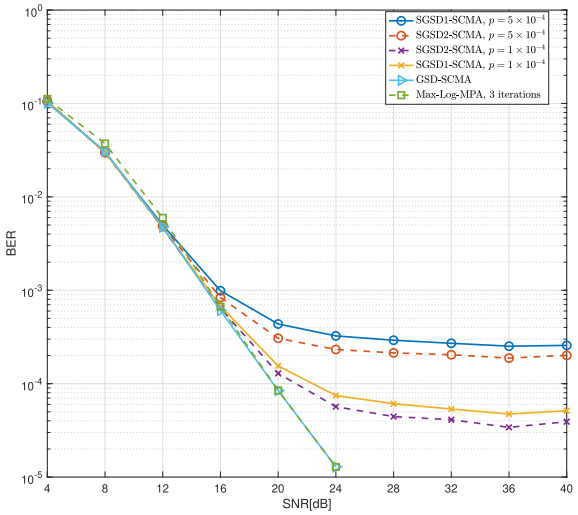


FIGURE 6. BER performance of different detectors when applying the CB1.

the lower p is, the better BER performance SGSD-SCMA has. For a given value of p, the SGSD2-SCMA detector has slightly better performance than the SGSD1-SCMA detector. The proposed SGSD-SCMA has performance gap with GSD-SCMA and Max-Log-MPA when SNR $\geqslant$ 12dB for $p = 5 \times 10^{-4}$ and SNR $\geqslant$ 16dB for $p = 1 \times 10^{-4}$ respectively.

Lastly, we evaluate the computational complexity. Fig. 7 illustrates the number of FLOPs of the Max-Log-MPA with 3 iterations and the GSD-SCMA detector. Please note that the number of FLOPs of the Max-Log-MPA does not vary with SNR values. We can see that the Max-Log-MPA with 3 iterations has lower complexity than the GSD-SCMA detector. There is no obvious reduction of the number of FLOPs for GSD-SCMA with the increase of SNR. Fig. 8 compares the FLOPs evolution of different steps in SGSD-SCMA with different values of p. In contrast to GSD-SCMA, the SGSD-SCMA detector shows significant decrement of FLOPs with the increment of SNR in Fig. 8. The number of FLOPs of the SGSD-SCMA detector is inversely proportional to the error detection probability p at low-to-moderate SNR values. In high SNR regime, the numbers of FLOPs of the SGSD-SCMA detector with different p values converge to the same level. Fig. 8 also compares the complexity of the two steps of the SGSD-SCMA detector. The number of FLOPs of the PRUN1 algorithm is greater than that of the PRUN2 algorithm. However, the SGSD1-SCMA detector has a significantly lower overall number of FLOPs in comparison to the SGSD2-SCMA detector because the PRUN1 algorithm outputs a smaller size of $\mathcal{L}_2^*$ as shown in Table 7. It implies that the size of $\mathcal{L}_2^*$ plays a significant role in the overall complexity of SGSD-SCMA.

For the CB1, when 4dB $\leqslant$ SNR $\leqslant$ 12dB, SGSD-SCMA with $p = 5 \times 10^{-4}$ has the same BER performance and lower

TABLE 4. The number of FLOPs of the PRUN1 algorithm.

Variables	USERPART $((\mathbf{r}_2)_{2k}, \mathcal{V})$	l_2	$t_1(i_1)(t_2(i_1))$	$s_1(i_1)(s_2(i_1))$	j	$\xi_1(\xi_2)$
FLOPs	$6(d_f-1)$	2	$4l_2-1$	$2l_1$	1	1

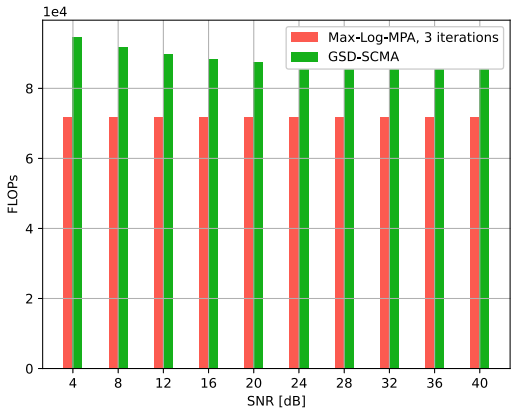


FIGURE 7. FLOPs of Max-Log-MPA and GSD-SCMA when applying the CB1.

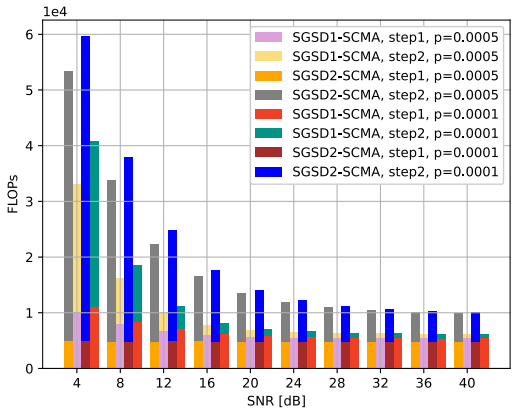


FIGURE 8. FLOPs of two steps in SGSD-SCMA when applying the CB1.

TABLE 5. The number of FLOPs of the PRUN2 algorithm.

Variables	$\mathcal{B}$	$\xi_1(\xi_2)$
FLOPs	$4(d_f-1)$	$4(d_f-1)$

complexity than the Max-Log-MPA and GSD-SCMA. When 4dB $\leqslant$ SNR $\leqslant$ 16dB, SGSD-SCMA with $p = 1 \times 10^{-5}$ is more efficient than the Max-Log-MPA and GSD-SCMA in terms of complexity. Overall, SGSD-SCMA has a better trade-off than the max-Log-MPA and GSD-SCMA at low to moderate SNR values for the CB1.

B. CB2 ($J = 6, K = 4, M = 16$)

Fig. 9 shows the simulation CER performance of the CB2. Table 8 compares the simulation error rate and error probability of the proposed pruning algorithms regarding the CB2. There is a mismatch of the PRUN2 algorithm between the simulation and theoretical performance. The mismatch might be brought by the value of $M_1 = \sqrt{M}$ because it is a coarse approximation derived from a classical modulation. In fact, the number of projections of the CB2 in each real dimension is not necessarily identical.

Because SD-SCMA is not compatible with the CB2, Fig. 10 only compares BER performance of Max-Log-MPA and SGSD-SCMA. The SGSD-SCMA detector with $p = 5 \times 10^{-3}$ outperforms the Max-Log-MPA at the range of 14dB $\leqslant$ SNR $\leqslant$ 22dB and it with $p = 5 \times 10^{-4}$ outperforms the Max-Log-MPA at the range of 14dB $\leqslant$ SNR $\leqslant$ 24dB.

Fig. 11 compares the number of FLOPs of two steps in SGSD-SCMA with different values of p. For the Max-Log-MPA detector with 5 iterations, the number of FLOPs is

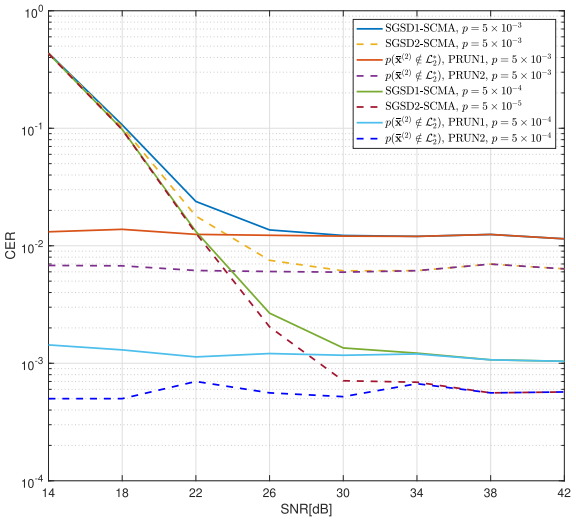


FIGURE 9. CER performance comparison of different detectors when applying the CB2.

TABLE 6. Simulation error rate and error probability of the CB1.

p	Algorithm	Simulation	Theoretical
5×10^{-4}	PRUN1	1.5×10^{-3}	2.0×10^{-3}
	PRUN2	1.1×10^{-3}	1.0×10^{-3}
1×10^{-4}	PRUN1	3.0×10^{-4}	4.0×10^{-4}
	PRUN2	2.0×10^{-4}	2.0×10^{-4}

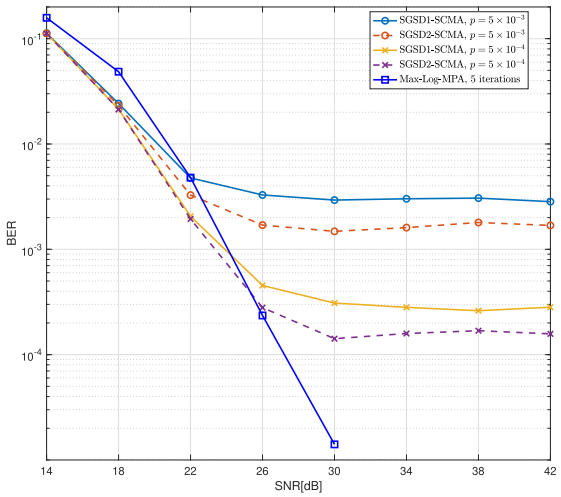


FIGURE 10. BER performance of different detectors when applying the CB2.

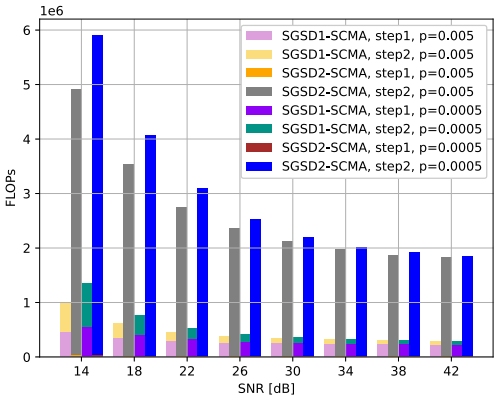


FIGURE 11. FLOPs of two steps in SGSD-SCMA when applying the CB2.

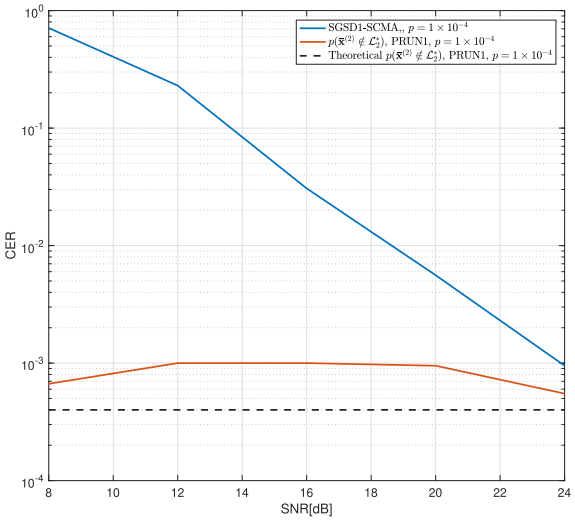


FIGURE 12. CER performance of SGSD-SCMA when applying the CB3.

7.62×10^6. When SNR $=$ 14dB, the number of FLOPs of the SGSD2-SCMA with $p = 1 \times 10^{-4}$ is about 77.50% of that of the Max-Log-MPA with 5 iterations. For a given value of p, the number of FLOPs of the SGSD1-SCMA is always lower than one fifth of that of the SGSD2-SCMA. The PRUN1 algorithm shows further substantial complexity advantage over the PRUN2 algorithm for the CB2 than for the CB1, which implies that the complexity of the tree search (the step2) of the SGSD-SCMA detector is sensitive to the codebook size.

C. CB3 ($J = 15, K = 6, M = 4$)

Since the CB3 is only compatible with the Max-Log-MPA and SGSD1-SCMA detectors, Fig. 12 solely illustrates the CER of the SGSD1-SCMA detector along with the simulation error rate and error probability of the PRUN1 algorithm when simulating with the CB3. The simulation error rate shows considerable performance gap with the error probability which is caused by the overlapping of the projections in each real dimension of the CB3. This reveals that a low value of M_1 deteriorates the error probability of the proposed pruning algorithm.

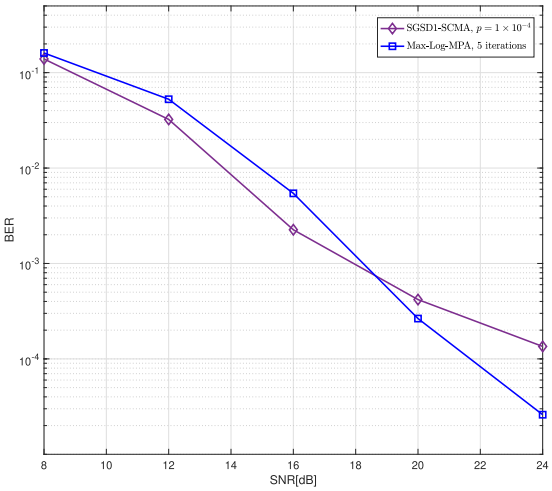


FIGURE 13. BER performance of different detectors when applying the CB3.

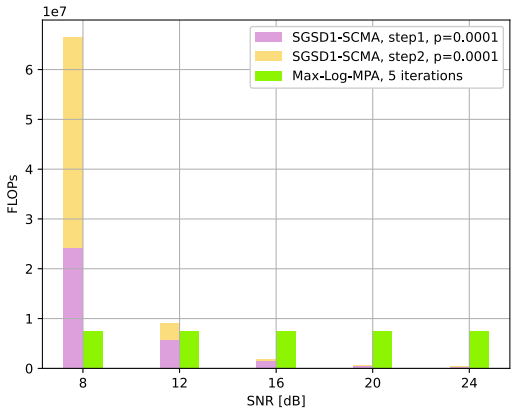


FIGURE 14. FLOPs comparison of different detector when applying the CB3.

Fig. 13 illustrates the BER performance of the two detectors when simulated with the CB3. The SGSD1-SCMA detector outperforms the Max-Log-MPA with 5 iterations at 8dB $\leqslant$ SNR $\leqslant$ 18.5dB when $p = 1 \times 10^{-4}$. Fig. 14 compares the number of FLOPs of the two detectors. The SGSD1-SCMA detector shows substantial complexity diminution compared with the Max-Log-MPA when SNR $\geqslant$ 16dB. For such a high overloading factor and small size codebook, the complexity of the SGSD1-SCMA detector majorly comes from the PRUN1 algorithm, which is the opposite for the CB1 and CB2.

Overall, the proposed SGSD-SCMA detector is significantly less complex than GSD-SCMA and Max-Log-MPA while reaching good MUD performance at low and moderate SNRs. As the three simulated codebooks are designed by different approaches and the proposed detector does not take advantage of any particular codebook structure, it can be applicable to any SCMA codebooks. The two proposed pruning algorithms, PRUN1 and PRUN2, have their own pros and cons. The PRUN1 algorithm is compatible with any SCMA codebooks and has lower complexity cost than the PRUN2 algorithm. However, the PRUN1 algorithm shows a slight performance degradation compared with the PRUN2 algorithm at moderate to high SNR values. On the other

TABLE 7. Size of sequence $\mathcal{L}_2^*$ of the pruning algorithms when applying the CB1.

p	Algorithm	4dB	8dB	12dB	16dB	20dB	24dB	28dB	32dB	36dB	40dB
5×10^{-4}	PRUN1	63	23	9	4	3	2	2	2	2	2
	PRUN2	130	79	48	32	24	19	17	16	15	14
1×10^{-4}	PRUN1	82	31	11	5	3	2	2	2	2	2
	PRUN2	148	91	55	35	25	20	17	16	15	14

TABLE 8. Simulation error rate and error probability of the CB2.

p	Algorithm	Simulation	Theoretical
5×10^{-3}	PRUN1	1.1×10^{-2}	1.0×10^{-2}
	PRUN2	6.0×10^{-3}	5.0×10^{-3}
5×10^{-4}	PRUN1	1.1×10^{-3}	1.0×10^{-3}
	PRUN2	6.0×10^{-4}	5.0×10^{-4}

hand, the PRUN2 algorithm is subject to the factor graph constraint in Eq. (94). Furthermore, among the three investigated codebook settings, SGSD-SCMA detector is the most efficient when simulating with the CB2. This implies that the proposed SGSD-SCMA detector is especially advantageous to the codebooks with moderate overloading factor and large codebook size in terms of both the complexity and MUD performance. Regarding the error probability increase of the proposed pruning algorithms brought by the low number of codeword projection, the proposed detector performs better with full diversity SCMA codebooks.

VIII. CONCLUSION

In this paper, we explore the SD-based detectors for SCMA systems. First, we review some of the SD-based state-of-the-art detectors which have less or no constraint on the codebook, namely SD-SCMA and GSD-SCMA detectors. A comprehensive introduction of the state-of-the-art including flowchart and pseudo-code is presented. Moreover, as a novelty, we leverage the SQRD and SE enumeration to accelerate the tree search process. Secondly, to remedy the computational complexity issue of the GSD-SCMA detector, we propose two pruning algorithms, named PRUN1 and PRUN2, and introduce the SGSD-SCMA detector. The PRUN1 algorithm is compatible to any SCMA codebooks while the PRUN2 algorithm restricts the codebook factor graph. Beside, we derive error probability expressions of the two proposed pruning algorithms. In the performance evaluation, codebooks with three parameter settings are investigated. The simulation error rates of the pruning algorithms match the derived error probabilities when the average number of projections in a real domain approximates the square root of the codebook size. Thus, the diversity of the codebook influence the error probability of the proposed pruning algorithms. The BER and computational complexity performance of the proposed SGSD-SCMA detector is compared with SD-based state-of-the-art detectors and the Max-Log-MPA detector. Although the efficiency of the proposed detector varies with the codebook, generally it shows substantial BER performance gains and complexity reduction than other detectors. By comparing the performance of different codebooks, we see that the SGSD-SCMA detector is especially advantageous to codebooks with moderate overloading factor and large size. Furthermore, the proposed low-complexity detector has the potential to be adopted in coded SCMA systems. The proposed SGSD-SCMA with soft output will be considered in a future work.

REFERENCES

[1] M. Series, "Minimum requirements related to technical performance for IMT-2020 radio interface (s)," ITU, Geneva, Switzerland, Tech. Rep. 2410-0, 2017.

[2] Y. Chen, A. Bayesteh, Y. Wu, B. Ren, S. Kang, S. Sun, Q. Xiong, C. Qian, B. Yu, Z. Ding, S. Wang, S. Han, X. Hou, H. Lin, R. Visoz, and R. Razavi, "Toward the standardization of non-orthogonal multiple access for next generation wireless networks," *IEEE Commun. Mag.*, vol. 56, no. 3, pp. 19–27, Mar. 2018.

[3] R. Hoshyar, F. P. Wathan, and R. Tafazolli, "Novel low-density signature for synchronous CDMA systems over AWGN channel," *IEEE Trans. Signal Process.*, vol. 56, no. 4, pp. 1616–1626, Apr. 2008.

[4] M. Taherzadeh, H. Nikopour, A. Bayesteh, and H. Baligh, "SCMA codebook design," in *Proc. IEEE 80th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2014, pp. 1–5.

[5] M. Vameghestahbanati, I. D. Marsland, R. H. Gohary, and H. Yanikomeroglu, "Multidimensional constellations for uplink SCMA systems—A comparative study," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2169–2194, 3rd Quart., 2019.

[6] M. Gao, W. Ge, P. Zhang, and Y. Zhang, "An efficient codebook design for uplink SCMA," *IEEE Access*, vol. 8, pp. 211665–211675, 2020.

[7] L. Yu, P. Fan, D. Cai, and Z. Ma, "Design and analysis of SCMA codebook based on star-QAM signaling constellations," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 10543–10553, Nov. 2018.

[8] C. Jiang and Y. Wang, "An uplink SCMA codebook design combining probabilistic shaping and geometric shaping," *IEEE Access*, vol. 8, pp. 76726–76736, 2020.

[9] H. Nikopour and H. Baligh, "Sparse code multiple access," in *Proc. IEEE 24th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, Sep. 2013, pp. 332–336.

[10] J. Dai, K. Niu, C. Dong, and J. Lin, "Improved message passing algorithms for sparse code multiple access," *IEEE Trans. Veh. Technol.*, vol. 66, no. 11, pp. 9986–9999, Nov. 2017.

[11] W. B. Ameur, P. Mary, M. Dumay, J.-F. Helard, and J. Schwoerer, "Performance study of MPA, log-MPA and MAX-Log-MPA for an uplink SCMA scenario," in *Proc. 26th Int. Conf. Telecommun. (ICT)*, Apr. 2019, pp. 411–416.

[12] B. F. D. Silva, D. Le Ruyet, and B. F. Uchoa-Filho, "Threshold-based edge selection MPA for SCMA," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 2957–2966, Mar. 2020.

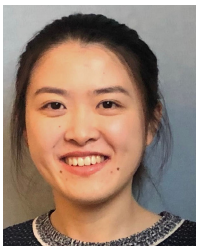
[13] Y. Wang and L. Qiu, "Edge selection-based low complexity detection scheme for SCMA system," in *Proc. IEEE 84th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2016, pp. 1–5.

[14] X. Meng, Y. Wu, Y. Chen, and M. Cheng, "Low complexity receiver for uplink SCMA system via expectation propagation," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Mar. 2017, pp. 1–5.

[15] X. Fu, M. Pischella, and D. L. Ruyet, "On Gaussian approximation algorithms for SCMA," in *Proc. 16th Int. Symp. Wireless Commun. Syst. (ISWCS)*, Aug. 2019, pp. 155–160.

[16] E. Viterbo and J. Boutros, "A universal lattice code decoder for fading channels," *IEEE Trans. Inf. Theory*, vol. 45, no. 5, pp. 1639–1642, Jul. 1999.

[17] A. M. Chan and I. Lee, "A new reduced-complexity sphere decoder for multiple antenna systems," in *Proc. IEEE Int. Conf. Communications. Conf.)*, vol. 1, Apr. 2002, pp. 460–464.
[18] S. Yang and L. Hanzo, "Fifty years of MIMO detection: The road to large-scale MIMOs," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 1941–1988, 4th Quart., 2015.
[19] F. Wei and W. Chen, "Low complexity iterative receiver design for sparse code multiple access," *IEEE Trans. Commun.*, vol. 65, no. 2, pp. 621–634, Feb. 2017.
[20] L. Li, J. Wen, X. Tang, and C. Tellambura, "Modified sphere decoding for sparse code multiple access," *IEEE Commun. Lett.*, vol. 22, no. 8, pp. 1544–1547, Aug. 2018.
[21] G. Chen, J. Dai, K. Niu, and C. Dong, "Optimal receiver design for SCMA system," in *Proc. IEEE 28th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, Oct. 2017, pp. 1–6.
[22] M. Vameghestahbanati, E. Bedeer, I. Marsland, R. H. Gohary, and H. Yanikomeroglu, "Enabling sphere decoding for SCMA," *IEEE Commun. Lett.*, vol. 21, no. 12, pp. 2750–2753, Dec. 2017.
[23] M. Vameghestahbanati, I. Marsland, R. H. Gohary, and H. Yanikomeroglu, "A novel SD-based detection for generalized SCMA constellations," *IEEE Trans. Veh. Technol.*, vol. 68, no. 10, pp. 10278–10282, Aug. 2019.
[24] D. Wübben, R. Böhnke, J. Rinas, V. Kühn, and K. D. Kammeyer, "Efficient algorithm for decoding layered space-time codes," *Electron. Lett.*, vol. 37, no. 22, pp. 1348–1350, Oct. 2001.
[25] B. Shim and I. Kang, "Sphere decoding with a probabilistic tree pruning," *IEEE Trans. Signal Process.*, vol. 56, no. 10, pp. 4867–4878, Oct. 2008.
[26] T. Cui and C. Tellambura, "An efficient generalized sphere decoder for rank-deficient MIMO systems," *IEEE Commun. Lett.*, vol. 9, no. 5, pp. 423–425, May 2005.
[27] D. Wubben, R. Bohnke, V. Kuhn, and K.-D. Kammeyer, "MMSE extension of V-BLAST based on sorted QR decomposition," in *Proc. IEEE Veh. Technol. Conf. (VTC-Fall)*, vol. 1, Oct. 2003, pp. 508–512.
[28] Z. Li, W. Chen, F. Wei, F. Wang, X. Xu, and Y. Chen, "Joint codebook assignment and power allocation for SCMA based on capacity with Gaussian input," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Jul. 2016, pp. 1–6.
[29] *Reproduced Codebook With J=6, K=4 and M=16*. Accessed: 2021. [Online]. Available: https://github.com/fuxtina/Reproduced-SCMA-codebook-with-J-6-K-4-and-M-16
[30] S. Zhang, K. Xiao, B. Xiao, Z. Chen, B. Xia, D. Chen, and S. Ma, "A capacity-based codebook design method for sparse code multiple access systems," in *Proc. 8th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Oct. 2016, pp. 1–5.



XIAOTIAN FU (Member, IEEE) received the B.Sc. degree in automation from Southeast University, Nanjing, China, in 2017, and the M.Sc. degree in telecommunications and networks from Conservatoire National des Arts et Métiers (CNAM), Paris, France, in 2018, where she is currently pursuing the Ph.D. degree with the CEDRIC Laboratory. Her research interests include wireless communications and signal processing including multiple access techniques, low-complexity detection algorithms, multi-antenna transmission, and machine learning for future wireless networks.

DIDIER LE RUYET (Senior Member, IEEE) received the B.E. degree in electrical engineering, the M.Sc. degree in physical systems and metrolology and the Ph.D. degree in communication from the Conservatoire National des Arts et Métiers (CNAM), Paris, in 1994 and 2001, respectively, and the Habilitation à diriger des recherches degree from Paris XIII University, in 2009. From 1988 to 1996 he was a Senior Member of the Technical Staff at SAGEM Defence and Telecommunication, France. He joined the Signal and Systems Laboratory, CNAM, as a Research Assistant, in 1996. From 2002 to 2010, he was an Assistant Professor at CNAM Paris. Since 2010, he has been a Full Professor at CNAM, CEDRIC Research Laboratory. From 2016 to 2019, he was the Deputy Director of the CEDRIC. He has published about 200 papers in refereed journals and conference proceedings and published five books in the area of digital communication. He has been involved in several National and European projects dealing with multicarrier transmission techniques and multi-antenna transmission. He served as a Technical Program Committee Member in major IEEE ComSoc and VTS conferences, and as the General Chair of the ninth edition of ISWCS 2012 conference and the Co-Chair of the ISWCS 2013 and 2019 edition. His research interests include digital communications and signal processing, including channel coding, detection and estimation algorithms, filter bank-based multi-carrier communication, and multi-antenna transmission.

BRUNO FONTANA DA SILVA (Member, IEEE) received the B.Sc. degree in electrical engineering from the Federal University of Santa Maria (UFSM), Santa Maria, Rio Grande do Sul, Brazil, in 2013, and the M.Sc. and Ph.D. degrees from the Federal University of Santa Catarina (UFSC), Florianópolis, Santa Catarina, Brazil, in 2015 and 2019, respectively. During his education, Mr. da Silva has been funded by National Council for Scientific and Technological Development (CNPq) with scholarships 132030/2013-6, 141161/2016-7, and 204676/2018-5. He has worked as an Assistant Professor with the Instituto Federal Sul-rio-grandense (IFSul), Sapiranga (RS), Brazil, from 2017 to 2021. He is currently working as a Data Scientist for the Advanced Analytics Team of Magazine Luiza, São Paulo, Brazil. His research interests include channel coding, MIMO communications, low-complexity receivers, and multiple access.

BARTOLOMEU F. UCHÔA-FILHO (Senior Member, IEEE) was born in Recife, Brazil, in 1965. He received the B.S.E.E. degree from the Federal University of Pernambuco (UFPE), Recife, Brazil, in 1989, the M.S.E.E. degree from the State University of Campinas (UNICAMP), Campinas, Brazil, in 1992, and the Ph.D. degree in electrical engineering from the University of Notre Dame, Notre Dame, Indiana, USA, in 1996.

He has held a postdoctoral/a visiting scholar positions at UNICAMP, (1977–1999), University of Sydney, Australia (2009–2010), *Centrale-Supélec* and CNAM, France, in 2017. He is currently a Professor with the Electrical and Electronics Engeneering Department, Federal University of Santa Catarina, Florianópolis, Brazil, and a Research Productivity Fellow (Level 1C) of the CNPq (the Brazilian National Council for Scientific and Technological Development, Ministry of Science and Technology). He served as the Editor-in-Chief for the *Journal of the Brazilian Telecommunications Society* (2003–2004), an Associate Editor of Information and Coding Theory for the *Journal of Communication and Information Systems* (JCIS) (2011–2014), and an Associate Editor of *Physical Communication* (Elsevier) and *Digital Signal Processing* (Elsevier) (2016–2018). He was the General Chair of the 38th Brazilian Telecommunications Symposium (SBrT'20), the Technical Program Committee Co-Chair of the 27th Brazilian Telecommunications Symposium (SBrT'09), the Track-Chair of the Information Theory and Coding Track of the 2014 IEEE/SBrT International Telecommunications Symposium (ITS'14), and the Co-Chair of "Track 1: Fundamentals and PHY" of IEEE PIMRC'2019. He has also served as a TPC member of several national and international symposia. His research interests include coding and information theory, with applications to digital communications systems.