Fast Optimal and Suboptimal Any-Time Algorithms for CDMA Multiuser Detection Based on Branch and Bound

Jie Luo, *Member, IEEE*, Krishna R. Pattipati, *Fellow, IEEE*, Peter Willett, *Fellow, IEEE*, and Georgiy M. Levchuk, *Member, IEEE*

***Abstract*—A fast optimal algorithm based on the branch-and-bound (BBD) method is proposed for the joint detection of binary symbols of K users in a synchronous code-division multiple-access channel with Gaussian noise. Relationships between the proposed algorithms (depth-first BBD and fast BBD) and both the decorrelating decision-feedback (DF) detector and sphere-decoding algorithm are clearly drawn. It turns out that decorrelating DF detector corresponds to a "one-pass" depth-first BBD; sphere decoding is, in fact, a type of depth-first BBD, but one that can be improved considerably via tight upper bounds and user ordering, as in the fast BBD. A fast "any-time" suboptimal algorithm is also available by simply picking the "current-best" solution in the BBD method. Theoretical results are given on the computational complexity and the performance of the "current-best" suboptimal solution.**

***Index Terms*—Branch and bound (BBD), code-division multiple access (CDMA), multiuser detection, optimal algorithm.**

I. Introduction

DUE TO THE problem of multiple-access interference (MAI) in many multiuser communication systems, multiuser detection for the symbol-synchronous Gaussian code-division multiple access (CDMA) channel has received considerable attention over the past 15 years. When the source signals are binary- or integer-valued, the resulting integer programming problem is generally NP-hard [4], unless the signature waveform autocorrelation matrix has a special structure [24], [25]. Consequently, prior research has focused on designing suboptimal receivers with low computational complexity and better performance than a conventional detector. Popular suboptimal detectors include the linear detectors, such as the decorrelator [4] and the minimum mean-square error (MMSE) detector [5]; the decision-driven detectors, such as the multistage detector [6], [7], the group detector [8], [9] and the decision-feedback (DF) detector [10]–[12]. The DF detector, one of the most efficient methods, has $O(K^2)$ complexity (K is the number of users), and its performance is significantly better than those of the linear detectors. Other advanced detectors, such as the semidefinite relaxation method [13], [14] and the probabilistic data association (PDA) method [15], [16] were proposed recently to achieve close-to-optimal performance at the expense of somewhat increased computational cost. A comparison of the performances of different detectors can be found in [17].

Since optimal multiuser detection is generally NP-hard, and the worst-case computational cost grows exponentially in the number of users [18], [19], it is unlikely to be implemented in a practical system. However, the optimal algorithm serves as a benchmark against which to evaluate suboptimal algorithms. In such an environment, average computational cost is much more important than the worst-case one. Since the multiuser detection problem can be viewed as a binary quadratic programming problem, smart search techniques, such as a branch-and-bound (BBD) method based on tight lower and upper bounds and user ordering, can speed up the solution process significantly. Prior research on using the BBD algorithm to multiuser detection includes [22] and [23]. An optimal algorithm based on sphere decoding (SD) was also proposed recently in [26]. These results show that the average computational cost can be significantly less than that of the worst-case one for an optimal multiuser detector.

Prior research on optimal multiuser detection used only the quadratic cost function and the binary constraints on user signals. Problem-domain information in the form of the matched-filter outputs being generated from a known statistical model is essentially ignored. In this paper, we propose a fast optimal BBD algorithm, and show that using the statistical information in the matched-filter outputs significantly reduces the average computational cost of the optimal multiuser detector. Compared with the breadth-first BBD algorithm in [23] and the SD procedure of [26], the key speed-up mechanisms of the proposed optimal algorithm are the following.

1) The use of user ordering proposed in [12]. We show that the first feasible solution of the BBD algorithm is, in fact, the solution of the decorrelating DF detector. Therefore, the user ordering proposed in [12] for the DF detector maximizes the probability that the first feasible solution is optimal.

Paper approved by M. Brandt-Pearce, the Editor for Modulation and Signal Design of the IEEE Communications Society. Manuscript received September 21, 2000; revised August 2, 2002. This work was supported by the Office of Naval Research under Contract N00014-98-1-0465 and Contract N00014-00-1-0101, and by the Naval Undersea Warfare Center under Contract N66604-1-99-5021. This paper was presented in part at the IEEE International Symposium on Information Theory, Washington, DC, June 2001, and in part at the IEEE International Conference on Communications, Anchorage, AK, May 2003.

J. Luo is with the Institute for Systems Research, University of Maryland, College Park, MD 20742 USA.

K. R. Pattipati and P. Willett are with the Electrical Engineering Department, University of Connecticut, Storrs, CT 06269 USA (e-mail: krishna@engr.uconn.edu; willett@engr.uconn.edu).

G. M. Levchuck is with Aptima Inc., Woburn, MA 01801 USA.

Digital Object Identifier 10.1109/TCOMM.2004.826349

2) A search strategy that maximizes the probability that the "current-best" feasible solution is optimal.
3) A computational enhancement that minimizes redundant computations in the lower bound computation.

When strict computational limits exist, a suboptimal solution can be obtained by simply picking the "current-best" solution in the BBD search. Since the decorrelating DF method is a first-order approximation to the optimal algorithm, a "current-best" solution of the second- or the third-order approximation will generally outperform the decorrelating DF method with a marginal increase in computation. Theoretical analysis of the performance and computational cost on the "current-best" solution is given and verified by the simulation results.

The rest of the paper is organized as follows. The synchronous multiuser detection problem formulation and existing techniques are discussed in Section II. In Section III, a depth-first BBD-based optimal algorithm (which can also be treated as a simplified version of the fast BBD algorithm) is presented. Analysis is given to show that the decorrelating DF solution is, in fact, the first feasible solution in the depth-first BBD approach. The relationship between the depth-first BBD and the SD method proposed in [26] and [27] is pointed out. In Section IV, we present a user ordering, a search strategy, and a computational method to use the statistical information in the matched-filter output. The fast optimal BBD algorithm is presented and is also extended to nonbinary systems. In Section V, suboptimal algorithms are introduced by picking the "current-best" solution in the BBD search. Theoretical analysis on the performances and the computational costs are given. Simulation results and comparative analysis are provided in Section VI. The paper concludes with a summary in Section VII.

II. Problem Formulation and Existing Methods

A discrete-time equivalent model for the matched-filter outputs at the receiver of a CDMA channel is given by the K-length vector [4]

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{b} + \boldsymbol{n} \tag{1}$$

where $\boldsymbol{b} \in \{-1, +1\}^K$ denotes the K-length vector of bits transmitted by the K active users. Here $\boldsymbol{H} = \boldsymbol{W}\boldsymbol{R}\boldsymbol{W}$ is a nonnegative definite signature waveform correlation matrix, $\boldsymbol{R}$ is the symmetric normalized correlation matrix with unit diagonal elements, $\boldsymbol{W}$ is a diagonal matrix whose kth diagonal element, w_{kk}, is the square root of the received signal energy per bit of the kth user, and $\boldsymbol{n}$ is a real-valued zero-mean Gaussian random vector with a covariance matrix $\sigma^2\boldsymbol{H}$. It has been shown that this model holds for both baseband [4] and passband [12] channels with additive Gaussian noise.

Letting $\boldsymbol{H} = \boldsymbol{L}^T\boldsymbol{L}$ be the Cholesky decomposition of $\boldsymbol{H}$, the system can also be represented by a white noise model

$$\tilde{\boldsymbol{y}} = \boldsymbol{L}^{-T}\boldsymbol{y} = \boldsymbol{L}\boldsymbol{b} + \boldsymbol{v} \tag{2}$$

where $\boldsymbol{v} = \boldsymbol{L}^{-T}\boldsymbol{n}$ is a white Gaussian noise with zero mean and covariance $\sigma^2\boldsymbol{I}$.

When all the user signals are equally probable, the optimal solution of (1) is the output of a maximum-likelihood (ML) detector [4]

$$\phi_{\mathrm{ML}} : \hat{\boldsymbol{b}} = \arg \min_{\boldsymbol{b}\in\{-1,+1\}^K} (\boldsymbol{b}^T\boldsymbol{H}\boldsymbol{b} - 2\boldsymbol{y}^T\boldsymbol{b}). \tag{3}$$

The ML detector has the property that it minimizes, among all detectors, the probability that not all users' decisions are correct.

The solution of the decorrelating detector [4]

$$\phi_{\mathrm{D}} : \hat{\boldsymbol{b}} = \arg \min_{\boldsymbol{b}\in\{-1,+1\}^K} \|\boldsymbol{b} - \boldsymbol{H}^{-1}\boldsymbol{y}\|_2^2 \tag{4}$$

is found in two steps. First, the unconstrained solution $\tilde{\boldsymbol{b}} = \boldsymbol{H}^{-1}\boldsymbol{y}$ is computed. This is then projected onto the constraint set via $\hat{b}_i = \mathrm{sign}(\tilde{b}_i)$.

The decorrelating DF method is described in [12]. If we denote the ith component of a vector $\boldsymbol{y}$ by y_i, and denote the (i,j)th component of a matrix $\boldsymbol{A}$ by a_{ij}, the decorrelating DF detector can be characterized by

$$\phi_{\mathrm{DF}} : \hat{\boldsymbol{b}} = \boldsymbol{P}\tilde{\boldsymbol{b}}, \tilde{b}_i = \mathrm{sign}\left(\sum_{j=1}^{K} f_{ij}[\boldsymbol{P}\boldsymbol{y}]_j - \sum_{j=1}^{i-1} a_{ij}\tilde{b}_j\right) \tag{5}$$

where $\boldsymbol{F} = \mathrm{U}([\boldsymbol{P}\boldsymbol{H}\boldsymbol{P}^T]^{-1})$, $\boldsymbol{A} = \mathrm{L}(\boldsymbol{F}\boldsymbol{P}\boldsymbol{H}\boldsymbol{P}^T)$. Here, $\mathrm{U}(\cdot)$ represents the upper triangular part of a matrix, $\mathrm{L}(\cdot)$ represents the strictly lower triangular part of a matrix, and $\boldsymbol{P}$ is a permutation matrix. The choice of $\boldsymbol{P}$ has been discussed in [12, Th. 1].

For multiuser detectors, symmetric energy (SE) is an important performance measure in the high signal-to-noise ratio (SNR) regime that characterizes the probability that all user signals are detected correctly. The SE of the ML detector, decorrelator, and the decorrelating DF detector can be expressed, respectively, by [12]

$$\begin{aligned} E(\phi_{\mathrm{ML}}) &= \min_{\boldsymbol{e}\in\{-1,0,1\}^K - \{\boldsymbol{0}\}} \boldsymbol{e}^T\boldsymbol{H}\boldsymbol{e} \\ E(\phi_{\mathrm{D}}) &= \min_{i=1,\ldots,K} \frac{1}{[\boldsymbol{H}^{-1}]_{ii}} \\ E(\phi_{\mathrm{DF}}) &= \min_{i=1,\ldots,K} l_{ii}^2. \end{aligned} \tag{6}$$

It has been shown in [12] that $E(\phi_{\mathrm{DF}}) \geq E(\phi_{\mathrm{D}})$ (assuming that the users are properly ordered). Usually, the DF detector can provide two to three orders of improvement in the magnitude of probability of error when compared with a linear detector. However, the output of the DF detector is still a suboptimal solution. Simulation results show that, in most cases, there still exists a substantial gap in performance between the DF detector and the optimal solution.

III. Optimal Algorithm Based on Depth-First BBD (Simplified Version of Fast BBD)

The idea of using a BBD method in solving binary or integer programming problems is already well known [20], [21]. BBD divides the decision regions into several parts and assigns each part to a branch in the BBD tree. For each branch, the decision region is further divided and assigned to subbranches. The node

on each leaf of the tree represents a feasible solution of the optimization problem. In order to avoid searching the whole tree to find the optimal solution, BBD associates a lower bound on the objective cost to each of the branches. The algorithm also keeps an upper bound, which is the cost of the "current-best" feasible solution. When the lower bound associated with a branch is greater than the upper bound, the whole branch is discarded, since no better solution can be found. For BBD methods, the tradeoff between a tight lower bound and a lower bound with fewer computational requirements is common to most of the problems.

In multiuser detection, a BBD method with breadth-first search has been used in [23] to find the minimum distance, which is defined by

$$d_{\min} = \sqrt{\min_{\boldsymbol{e}\in\{-1,0,1\}^K-\{0\}} \boldsymbol{e}^T\boldsymbol{H}\boldsymbol{e}}. \tag{7}$$

In this section, we present an optimal algorithm based on BBD with depth-first search. We point out the relationship between the proposed depth-first BBD method, the decorrelating DF detector, and the SD detector [26], [27].

For the convenience of the readers, the algorithm presented in this section is a simplified version of the fast BBD algorithm. The fast optimal BBD that fully uses the statistical information in (1) is proposed and studied in Section IV.

A. Depth-First BBD Algorithm

Since $\boldsymbol{H}^{-1} = \boldsymbol{L}^{-1}\boldsymbol{L}^{-T}$, the objective function in (3) can be equivalently written as

$$\begin{aligned}\phi_{\mathrm{ML}}: \hat{\boldsymbol{b}} &= \arg \min_{\boldsymbol{b}\in\{-1,+1\}^K} (\boldsymbol{b}-\boldsymbol{H}^{-1}\boldsymbol{y})^T\boldsymbol{H}(\boldsymbol{b}-\boldsymbol{H}^{-1}\boldsymbol{y}) \\ &= \arg \min_{\boldsymbol{b}\in\{-1,+1\}^K} \left\|\boldsymbol{L}(\boldsymbol{b}-\boldsymbol{H}^{-1}\boldsymbol{y})\right\|_2^2 \\ &= \arg \min_{\boldsymbol{b}\in\{-1,+1\}^K} \|\boldsymbol{L}\boldsymbol{b}-\tilde{\boldsymbol{y}}\|_2^2. \end{aligned} \tag{8}$$

Define $\boldsymbol{D} = \boldsymbol{L}\boldsymbol{b}$. We have

$$\begin{aligned}\phi_{\mathrm{ML}}: \hat{\boldsymbol{b}} &= \arg \min_{\boldsymbol{b}\in\{-1,+1\}^K} \|\boldsymbol{D}-\tilde{\boldsymbol{y}}\|_2^2 \\ &= \arg \min_{\boldsymbol{b}\in\{-1,+1\}^K} \sum_{k=1}^{K}(D_k-\tilde{y}_k)^2. \end{aligned} \tag{9}$$

Here, since $\boldsymbol{L}$ is a lower triangular matrix, D_k depends only on $(b_1, b_2, \ldots, b_k)$. When the decisions for the first k users are fixed, the term

$$\xi_k = \sum_{i=1}^{k}(D_i-\tilde{y}_i)^2 \tag{10}$$

can serve as a lower bound of (9). It can be easily seen that the lower bound is achievable when the binary constraints on $(b_{k+1}, \ldots, b_K)$ are disregarded. The BBD tree search to find the minimum value of $\|\boldsymbol{D}-\tilde{\boldsymbol{y}}\|_2^2$ is described below.

Similar to a general BBD method [20], [21], the algorithm maintains a node stack called OPEN, and a scalar called UPPER, which is equal to the minimum feasible cost found so far, i.e., the "current-best" solution. Define k to be the level of a node (the virtual root node has level 0). Label the branch which connects the two nodes $(b_1, \ldots, b_{k-1})$ and $(b_1, \ldots, b_k)$ with $D_k(b_1, b_2, \ldots, b_k)$. The node $(b_1, \ldots, b_k)$ is labeled with the lower bound ξ_k. Also, define $\boldsymbol{z}_k = \tilde{\boldsymbol{y}} - \sum_{i=1}^{k} b_i\boldsymbol{l}_i$, where $\boldsymbol{l}_i$ denotes the ith column of $\boldsymbol{L}$. Denote $[z_k]_j$ as the jth component of vector $\boldsymbol{z}_k$. The depth-first BBD algorithm proceeds as follows.

Depth-first BBD Algorithm (Simplified Version of the Fast BBD)

1) Order users according to [12, Th. 1], which is also presented in *Proposition 2* of Section IV below. Compute $\boldsymbol{y}$, $\boldsymbol{H}$, and $\boldsymbol{L}$ matrices for the ordered system.
2) Precompute $\tilde{\boldsymbol{y}} = \boldsymbol{L}^{-T}\boldsymbol{y}$.
3) Initialize $k = 0$. $\boldsymbol{z}_k = \tilde{\boldsymbol{y}}$, $\xi_k = 0$, UPPER $= +\infty$ and OPEN $=$ NULL.
4) Set $k = k + 1$. For both nodes, let $\boldsymbol{z}_k = \boldsymbol{z}_{k-1}$, $\xi_k = \xi_{k-1}$. Choose the node in level k such that $b_k = \mathrm{sign}([z_k]_k)$. Set flag $f = 1$.
5) Compute $[z_k]_k = [z_k]_k - b_k l_{kk}$.
6) Compute $\xi_k = \xi_k + (D_k - \tilde{y}_k)^2 = \xi_k + [z_k]_k^2$.
7) If $\xi_k \geq$ UPPER, drop this node. Go to step 10).
8) If $\xi_k <$ UPPER and $k < K$, $\forall j > k$ precompute $[z_k]_j = [z_k]_j - b_k l_{jk}$. If $f = 1$, append the other node with $b_k = -\mathrm{sign}([z_k]_k)$ to the end of the OPEN list, and store the associated k, ξ, $\boldsymbol{z}_k$ together with this node. Go to step 4).
9) If $\xi_k <$ UPPER, $k = K$, update the "current-best" solution and UPPER $= \xi_k$. Go to step 10).
10) If the OPEN list is not empty, pick the node from the end of the OPEN list, set k, ξ, and $\boldsymbol{z}_k$ equal to the stored values associated with this node. Set flag $f = 0$ and go to step 5).
11) Stop and report the "current-best" solution.

Example 1: The following three-user example illustrates the procedure. The system is given by

$$\begin{aligned}\boldsymbol{y} &= \boldsymbol{H}\boldsymbol{b} + \boldsymbol{n} \\ \boldsymbol{H} &= \begin{bmatrix} 4.25 & 0.85 & 0.57 \\ 0.85 & 2.2 & 1.6 \\ 0.57 & 1.6 & 2.0 \end{bmatrix} = \boldsymbol{L}^T\boldsymbol{L} \\ \boldsymbol{L} &= \begin{bmatrix} 1.980 & 0 & 0 \\ 0.411 & 0.960 & 0 \\ 0.403 & 1.131 & 1.414 \end{bmatrix}. \end{aligned} \tag{11}$$

Assume that the source signal is $\boldsymbol{b} = [1, 1, 1]^T$ and the noise vector is $\boldsymbol{n} = [-0.313, -0.781, 0.585]^T$, hence, $\boldsymbol{y} = [5.357, 3.869, 4.755]^T$. Fig. 1 shows the BBD tree structure.

In step 1), we precompute $\tilde{\boldsymbol{y}} = \boldsymbol{L}^{-T}\boldsymbol{y} = [2.007, 0.068, 3.363]^T$. Then, initialize $k = 0$, $\boldsymbol{z}_0 = [2.007, 0.068, 3.363]^T$, $\xi_0 = 0$, UPPER $= +\infty$, OPEN $=$ NULL. In step 4), let $k = 1$, choose the node with $b_1 = \mathrm{sign}(2.007) = 1$ (node 1 in Fig. 1). Add node 6 to the OPEN list. Update $\boldsymbol{z}_1 = [0.028, -0.343, 2.960]^T$, $\xi_1 = 0.0008$. Since $\xi_1 <$ UPPER and $k < 3$, go to step 4). This leads us to node 2. Add node 4 to the end of the OPEN list. Go back to step 4), which leads us to node 3 (which is the first feasible solution and, as shown later, it also corresponds to the decorrelating DF solution). Since this is the bottom level, we know that node 3 gives a better result than node $(1, -1, -1)$. Therefore, without

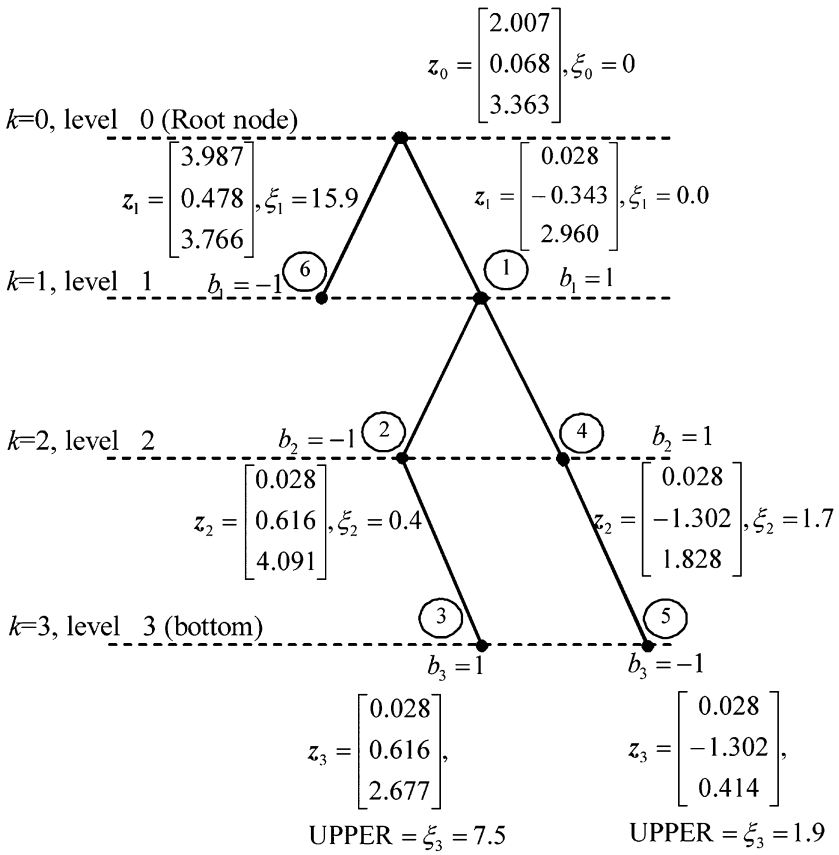


Fig. 1. Example of the depth-first BBD algorithm.

changing the OPEN list, update UPPER $= \xi_3 = 7.545$. In step 9), we pick node 4 from the end of the OPEN list. Go to node 5, and obtain $\xi_3 = 1.868 <$ UPPER, which means that node 5 is a better solution. Update UPPER $= 1.868$ and pick node 6 from the OPEN list. For node 6, since $\xi_1 = 15.896 >$ UPPER, we drop this node. Now since the OPEN list is empty, the algorithm stops and reports node 5 as the optimal solution.

The above algorithm is a BBD method with depth-first search. The computational cost for step 1) is $(K(K+1))/2$ multiplications and $(K(K-1))/2$ additions. Steps 5) and 6) need two additions and one multiplication. Notice that step 1) is outside the BBD search. In step 8), since b_k can only take known discrete values, $b_k \boldsymbol{l}_k$ can be precomputed and stored; hence, only $K-k$ additions are needed to obtain $\boldsymbol{z}_k$. To update the lower bound for a node on level $K-k+1 (k=1,\ldots,K)$, at most $k+1$ additions and one multiplication are needed. In addition, the computational requirements for finding the first feasible solution (also the optimal solution in the noise-free case) are $(K(K+3))/2$ multiplications and $K(K+1)$ additions.

B. Relationship Between the Depth-First BBD and the Decorrelating DF Detector

Proposition 1: The first feasible solution obtained from the above depth-first BBD search is the solution of decorrelating DF method.

Proof: From step 3), when we branch, we first go to the node with a smaller lower bound value. In the above BBD method, suppose $(b_1,\ldots,b_{k-1})$ has already been fixed by the branch. The choice of b_k for the BBD method can be described by

$$\tilde{\boldsymbol{b}} = \arg \min_{\substack{b_k \in \{-1,+1\} \\ b_{j>k} \in (-\infty,\infty)}} (\boldsymbol{b} - \boldsymbol{H}^{-1}\boldsymbol{y})^T \boldsymbol{H} (\boldsymbol{b} - \boldsymbol{H}^{-1}\boldsymbol{y})$$
$$b_k = \tilde{b}_k. \tag{12}$$

Notice that in (12), $(b_1,\ldots,b_{k-1})$ is fixed and we only have a binary constraint on b_k. The choice of b_k for the decorrelating DF method, however, is given by

$$\tilde{\boldsymbol{b}} = \arg \min_{b_{j\geq k} \in (-\infty,\infty)} (\boldsymbol{b} - \boldsymbol{H}^{-1}\boldsymbol{y})^T \boldsymbol{H} (\boldsymbol{b} - \boldsymbol{H}^{-1}\boldsymbol{y})$$
$$b_k = \operatorname{sign}(\tilde{b}_k). \tag{13}$$

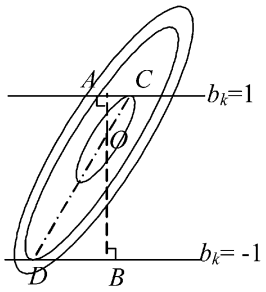


Fig. 2. Comparison of decorrelating DF and BBD decisions on b_k.

Fig. 2 shows the difference between the above two choices. The ellipses here represent the level curves of the objective

function in the user-expurgated channel that contains users $k, \ldots, K$. For the DF method, the soft solution $\tilde{\boldsymbol{b}}$ corresponds to point O. The decision on b_k is made by comparing the lengths $|AO|$ and $|BO|$. While for the proposed depth-first BBD method, the decision on b_k is made by comparing the values of the cost function on points C and D, which corresponds to comparing the lengths of $|CO|$ and $|DO|$. Since the triangles AOC and BOD are similar, (12) and (13) are equivalent.

Example 1—Continued: In the above example, at node 1, the user-expurgated channel for the decorrelating DF method is represented by

$$\begin{bmatrix} y_2 \\ y_3 \end{bmatrix} - \begin{bmatrix} 0.85 \\ 0.57 \end{bmatrix} = \begin{bmatrix} 2.2 & 1.6 \\ 1.6 & 2.0 \end{bmatrix} \begin{bmatrix} b_2 \\ b_3 \end{bmatrix} + \begin{bmatrix} n_2 \\ n_3 \end{bmatrix}. \tag{14}$$

According to (13), the decision on b_2 for DF detector is given by

$$\begin{aligned} \tilde{b} &= \begin{bmatrix} 2.2 & 1.6 \\ 1.6 & 2.0 \end{bmatrix}^{-1} \left\{ \begin{bmatrix} 3.87 \\ 4.76 \end{bmatrix} - \begin{bmatrix} 0.85 \\ 0.57 \end{bmatrix} \right\} \\ &= \begin{bmatrix} -0.358 \\ 2.379 \end{bmatrix} \\ b_2 &= \operatorname{sign}(\tilde{b}_2) = -1 \end{aligned} \tag{15}$$

which is consistent with the depth-first BBD algorithm. However, as shown in the example, the DF method failed to find the optimal solution.

Recall that in the BBD algorithm, the computational cost to obtain the first feasible solution (also the solution of DF detector) is much less than the computational cost of a conventional linear detector. Evidently, any further computations will result in better accuracy than the decorrelating DF solution (unless the DF solution is already optimal).

C. Relationship Between the Depth-First BBD and the SD

The SD, originally proposed in [28], is a well-known efficient lattice decoding algorithm [26], [27], and was introduced to the multiuser detection community recently in [26]. We first rewrite the SD algorithm as follows.

SD Algorithm

1) Compute the Cholesky decomposition matrix $\boldsymbol{H} = \boldsymbol{L}^T \boldsymbol{L}$.
2) Precompute $\tilde{\boldsymbol{y}} = \boldsymbol{L}^{-T} \boldsymbol{y}$, $C = \alpha K \sigma^2$, where α is chosen so that [27]
$$\int_0^{\alpha K} \frac{\lambda^{\frac{K}{2}-1}}{\Gamma\left(\frac{K}{2}\right)} e^{-\lambda} d\lambda = 0.99. \tag{16}$$
3) Initialize $k = 0$, $\boldsymbol{z}_k = \tilde{\boldsymbol{y}}$, $\xi_k = 0$. Initialize UPPER $= C$. Initialize OPEN $=$ NULL.
4) Set $k = k + 1$. For both nodes, let $\boldsymbol{z}_k = \boldsymbol{z}_{k-1}$, $\xi_k = \xi_k$. Choose the node in level k such that $b_k = -1$. Append the node with $b_k = +1$ to the end of the OPEN list, and store the associated k, ξ, and $\boldsymbol{z}_k$ together with this node.
5) Compute $[z_k]_k = [z_k]_k - b_k l_{kk}$.
6) Compute $\xi_k = \xi_k + (D_k - \tilde{y}_k)^2 = \xi_k + [z_k]_k^2$.
7) If $\xi_k \geq$ UPPER, drop this node. Go to step 10).
8) If $\xi_k <$ UPPER and $k < K$, for $j = k+1$, precompute $[z_k]_j = [z_k]_j - \sum_{i=1}^{k} b_i l_{ji}$. Go to step 4).
9) If $\xi_k <$ UPPER, $k = K$, update the "current-best" solution and UPPER $= \xi_k$. Go to step 10).
10) If the OPEN list is not empty, pick the node from the end of the OPEN list, set k, ξ, and $\boldsymbol{z}_k$ equal to the stored values associated with this node, and go to step 5).
11) If no solution is available yet, let $C = 2C$ and go to step 3). Otherwise, stop and report the "current-best" solution.

Although written in a different form with a different notation, it is easy to show that the above algorithm is indeed identical to the SD methods proposed in [26] and [27].[1] Apparently, the SD method can be categorized as a depth-first BBD algorithm. The major differences between the SD and the proposed depth-first BBD algorithm, however, are in steps 1), 3), 4), and 8). The lower-bound update is also identical to the breadth-first BBD algorithm proposed in [23].

- As we have shown before, the choice of b_k in step 4) of the proposed depth-first BBD algorithm corresponds to the solution of the DF detector (which uses the statistical information embedded the system model). In the noise-free case, the first feasible solution of the fast optimal algorithm is optimal. This guarantees a minimum computational cost when the system is noise free. It is also a key step that allows the fast BBD algorithm (proposed in the next section) to further use the statistical information and perform user ordering, smart search, smart computing, etc. However, these enhancements are not exploited in the SD algorithm because statistical information in the model is ignored in step 4).
- The user ordering in step 1) of the simplified fast optimal algorithm is the single most important step that reduces the average computational cost. This is further studied in Section IV.
- Step 8) in the two algorithms represent different ways of computing the lower bounds. Since $D_k = \sum_{j=1}^{k} b_j l_{kj}$ for the sibling nodes share most of the computations, step 8) in the SD method precomputes the common terms for both nodes in level $k+1$. However, notice that the computations of D_k for nodes in different levels also involve partially common terms. In the depth-first BBD, each node precomputes part of the lower bounds for the subnodes in the branch and removes the redundant computations. However, if the branch is discarded later in the BBD search, such precomputing itself is a waste of computational resources. Unfortunately, it is impossible to completely avoid the redundant computations. Nevertheless, different computational methods may result in different average computational costs, especially when the statistical information embedded in (1) is considered. In the proposed depth-first BBD, the computational cost for user k to obtain the lower bound is $K-k+2$ additions and one multiplication. Hence, the computational cost

[1]Two different upper bound initializations are proposed for the SD algorithm in [26] and [27]. According to computer simulations, the average computational complexity of the SD in [26] is higher than the one in [27], both in low and high SNR regimes. Bounding and sphere-enlargement parameters, respectively, in steps 2) and 11) may be better coordinated, but no suggestions of such tuning have appeared to date.

for the nodes near the bottom of the tree is relatively small. When the signal powers are not close to each other, the user ordering in step 1) puts the weak users at the bottom of the tree. In such situations, since branching and searching happen mostly on the weak users, the computational method in the depth-first BBD results in a lower average computational cost than that of the SD. We will study this aspect of computation further in Section IV.

- The initialization of UPPER in step 3) of the sphere decoder is certainly a step that uses the statistical information embedded in (1). In spite of the drawback of requiring σ, this key step in the SD algorithm ensures that the order of the asymptotic average computational cost is $O(K^2)$ [27]. Such an initialization technique can also be easily applied to the depth-first BBD as well as the fast BBD algorithm (described in the next section). However, since the proposed BBD algorithm ensures a minimum asymptotic computational cost, the effect of upper bound initialization in the high SNR regime is limited. Nevertheless, a proper initialization of the upper bound does reduce the average computational cost when SNR is moderate. This is further studied toward the end of Section IV.

IV. Fast Optimal BBD Algorithm Using the Statistical Information (Full Version)

A key feature of multiuser detection is that the matched-filter output $\boldsymbol{y}$ is generated from a statistical model given by (1). Typically, the variance of the noise is not very large, which means that a significant fraction of optimal multiuser detection problems can be solved easily. The statistical information helps suboptimal algorithms, such as the DF detector [12] and the PDA detector [15], to achieve outstanding bit-error performance with low computational costs. However, this information is essentially ignored in most of the existing optimal multiuser detectors. In this section, we present the full version of the fast optimal BBD algorithm. The key ideas of using the statistical information are: the user ordering, the search strategy, and the lower-bound computation.

The BBD search can be separated into two stages. We term the first stage the "search" stage, where the "current-best" solution is not the optimal solution. The second stage is termed the "confirm" stage, where the "current-best" solution is optimal, but the algorithm needs to confirm that it is indeed better than any other solution.

Assume that the true solution $\bar{\boldsymbol{b}}$ is also the ML solution. In the "confirm" stage, we have

$$\text{UPPER} = \|\boldsymbol{L}\bar{\boldsymbol{b}} - \boldsymbol{L}^{-T}\boldsymbol{y}\|_2^2 = \|\boldsymbol{v}\|_2^2. \tag{17}$$

Asymptotically, $\|\boldsymbol{v}\|_2^2 \to 0$.

Now, consider any other branch associated with vector $(b_1, b_2, \ldots, b_k)$. Without loss of generality, suppose $\forall j < k$, $b_j = \bar{b}_j$, and $b_k \neq \bar{b}_k$. The lower bound is

$$\xi_k = \sum_{i=1}^{k}(D_i - \tilde{y}_i)^2 = \sum_{i=1}^{k-1}(v_i)^2 + (v_k \pm 2l_{kk})^2. \tag{18}$$

Apparently, when $\sigma \to 0$, $\xi_k \to 4l_{kk}^2$. This shows that, asymptotically, whenever the algorithm enters the "confirm" stage, all the branches will be discarded with a high probability.

A. User Ordering

According to the above intuitive analysis, the major task in the "search" stage is to maximize the probability that the "current-best" solution is optimal, so that the algorithm can enter the "confirm" stage as soon as possible. As we have shown in the previous section, the first feasible solution in the proposed algorithm is the decorrelating DF solution. For the DF detector, define $P_e(k)$ to be the probability of error on user k, given that all the decisions on users $1, \ldots, k-1$ are correct. We have from [12]

$$P_e(k) = Q\left(\frac{l_{kk}}{\sigma}\right). \tag{19}$$

In the high SNR regime, the probability of error of the DF solution is dominated by the user corresponding to the minimum diagonal element of $\boldsymbol{L}$.

Proposition 2: The following user-ordering algorithm, presented first in [29] and then in [12, Th. 1], maximizes the SE of the decorrelating DF detector, i.e., it maximizes the probability that the first feasible solution in the fast BBD algorithm is optimal.

User-Ordering Algorithm: Select the first user of the new order (denote this user's index as i_1) as the one that corresponds to the minimum diagonal element of $\boldsymbol{H}^{-1}$. For $k = 2, \ldots, K$, select the kth user of the new order (denote this user's index as i_k) as the user that corresponds to the minimum diagonal element of $\tilde{\boldsymbol{H}}^{-1}$, and $\tilde{\boldsymbol{H}}$ is the $\boldsymbol{H}$ matrix of the user-expurgated channel that contains only the remaining $K-k+1$ users. Then the optimal user order that maximizes SE is $\{i_1, i_2, \ldots, i_K\}$.

Proof: See [9, Prop. 1] for the proof.

B. Search Strategy

Although the user ordering maximizes the probability that the first feasible solution is optimal, there is still a small probability that it is not optimal. In the high SNR regime, defining $P_e(1^{\text{st}})$ to be the probability of error of the first feasible solution, we have

$$P_e(1^{\text{st}}) = \sum_{i=1}^{K} Q\left(\frac{l_{ii}}{\sigma}\right). \tag{20}$$

Define

$$\begin{aligned} m_1 &= \arg\min_j l_{jj} \\ m_i &= \arg\min_{j \neq m_1, \ldots, m_{i-1}} l_{jj}. \end{aligned} \tag{21}$$

Given that the first feasible solution is not optimal, user m_1 has a high probability of being the erroneous user, since $Q(l_{m_1 m_1}/\sigma)$ dominates $P_e(1^{\text{st}})$. Consequently, swapping the decision on user m_1 and applying DF detection to find the second feasible solution is the best choice. The probability that

neither the first nor the second feasible solutions are optimal is given by

$$P_e(2^{\rm nd}) = \sum_{i \neq m_1} Q\left(\frac{l_{ii}}{\sigma}\right). \tag{22}$$

Similarly, m_2 is the next user we should search. And the probability that the "current-best" solution is still not optimal after searching users m_1 and m_2 is

$$P_e(3^{\rm rd}) = \sum_{i \neq m_1, m_2} Q\left(\frac{l_{ii}}{\sigma}\right). \tag{23}$$

Apparently, unlike the search strategy in the depth-first BBD which searches nodes in descending order of their levels in the tree, the optimal search strategy visits nodes in ascending order according to the values of the diagonal elements of the $\bar{\boldsymbol{L}}$ matrix.

Due to the dynamic choice of which node to explore next, the worst-case storage requirement (i.e., of previously visited-node data) is exponential. The fixed search strategy of the depth-first BBD, including the SD, obviates this; but likewise, so do other versions of BBD that perform a smart search only on certain users. Extreme demands on memory are rare, but if they are of concern, it is worthwhile to recall that there is a continuum of tradeoffs between memory and speed.

C. Computational Enhancement

Step 8) of the SD algorithm precomputes part of the lower bounds for the sibling nodes. However, it does not take advantage of the fact that other nodes may also share part of the computations. The depth-first BBD method does precomputing for all the nodes under the same branch. However, if the branch is discarded, the precomputing itself is a waste of computational resources.

In the high SNR regime, since the error performance of the DF detector is characterized by the diagonal elements of $\boldsymbol{L}$, it is reasonable to make the following assumption in the BBD search.

Assumption: If a branch on level k is accepted (not discarded), then the subbranches on levels $k+1, \ldots, k+m$ may also be accepted with a high probability, as long as $\forall k < j \leq k+m, l_{jj} \leq l_{kk}$.

Based on this assumption, suppose for user k, u_k and d_k are defined as

$$u_k = \arg \min_{0<i<k} (\forall i \leq j < k, l_{jj} < l_{kk}) \tag{24}$$

or $u_k = k$ if no solution can be found in (24)

$$d_k = \arg \max_{k<i\leq K} (\forall k < j \leq i, l_{jj} \leq l_{kk}) \tag{25}$$

or $d_k = k$ if no solution can be found in (25).

Similar to the simplified fast optimal algorithm, we also keep a vector $\boldsymbol{z}$. When computing the lower bound ξ_k, we precompute for users $k+1, \ldots, d_k$

$$\forall k < j \leq d_k, [z_k]_j = [z_{k-1}]_j - \sum_{i=u_k}^{k} b_i l_{ji} \tag{26}$$

i.e., precomputing involves only the block in $\boldsymbol{L}$ with rows $[k+1, d_k]$ and columns $[u_k, k]$.

D. Fast Optimal BBD Algorithm (Full Version)

Similar to the depth-first BBD, the user ordering is precomputed offline, and we assume that all matrices are properly precomputed for the ordered system. In order to implement the new search strategy, instead of using the OPEN stack, in the full version, we have K queues, termed $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_K$. Queue $\boldsymbol{q}_k$ is associated with user k. The nodes in a queue follow the "first-in, first-out" rule, i.e., nodes enter from the tail and are taken from the head of the queue. In addition, we order queues according to the values of the diagonal elements of $\boldsymbol{L}$, i.e., in the BBD search, we take nodes from the queues in the order $[\boldsymbol{q}_{m_1}, \ldots, \boldsymbol{q}_{m_K}]$, where $m_1, \ldots, m_K$ are defined in (21).

To implement the proposed method, for each user k, the block margins u_k and d_k are precomputed and stored in vectors $\boldsymbol{u}$ and $\boldsymbol{d}$.

The full version of the fast optimal BBD algorithm proceeds as follows.

Fast Optimal BBD Algorithm

1) Order users according to [12, Th. 1], which is also presented in *Proposition 2*. Compute $\boldsymbol{y}$, $\bar{\boldsymbol{H}}$, and $\bar{\boldsymbol{L}}$ matrices for the ordered system; precompute the vectors $\boldsymbol{u}$ and $\boldsymbol{d}$, the components of which are defined by (24) and (25).
2) Precompute $\tilde{\boldsymbol{y}} = \boldsymbol{L}^{-T}\boldsymbol{y}$.
3) Initialize $k = 0$. $\boldsymbol{z}_k = \tilde{\boldsymbol{y}}$, $\xi_k = 0$, $\mathrm{UPPER} = +\infty$ and initialize K queues by $\forall k$, $\boldsymbol{q}_k = \mathrm{NULL}$.
4) Set $k = k+1$. For both nodes, let $\boldsymbol{z}_k = \boldsymbol{z}_{k-1}$, $\xi_k = \xi_{k-1}$. Choose the node in level k such that $b_k = \mathrm{sign}([z_k]_k)$. Set flag $f = 1$.
5) Compute $[z_k]_k = [z_k]_k - b_k l_{kk}$.
6) Compute $\xi_k = \xi_k + (D_k - \tilde{y}_k)^2 = \xi_k + (z_k)_k^2$.
7) If $\xi_k \geq \mathrm{UPPER}$, drop this node. Go to step 10).
8) If $\xi_k < \mathrm{UPPER}$ and $k < K$, do

8.1) If $f = 1$, for both nodes in level k:

8.1.1) If $d_k > k$, precompute $\forall k < j \leq d_k$, $[z_k]_j = [z_k]_j - \sum_{i=u_k}^{k-1} b_i l_{ji}$.

8.1.2) If $d_k = k$, precompute $j = k+1$, $[z_k]_j = [z_k]_j - \sum_{i=u_{k+1}}^{k-1} b_i l_{ji}$.

8.1.3) Append the node $b_k = -\mathrm{sign}([z_k]_k)$ to the tail of queue $\boldsymbol{q}_{m_k}$, and store the associated k, ξ, and $\boldsymbol{z}_k$ together with this node.

8.2) If $d_k > k$, precompute $\forall k < j \leq d_k$, $[z_k]_j = [z_k]_j - b_k l_{jk}$.

8.3) If $d_k = k$, precompute $j = k+1$, $[z_k]_j = [z_k]_j - b_k l_{jk}$.

8.4) Go to step 4).

9) If $\xi_k < \mathrm{UPPER}$, $k = K$, update the "current-best" solution and $\mathrm{UPPER} = \xi_k$. Go to step 10).
10) If not all the queues are empty, pick one node from the queues (note that we should check queues in the order of $\boldsymbol{q}_{m_1}, \ldots, \boldsymbol{q}_{m_K}$). Set k, ξ, and $\boldsymbol{z}_k$ equal to the stored values associated with this node. Set $f = 0$, go to step 5).
11) Stop and report the "current-best" solution.

In the ideal case, the first feasible solution is optimal, and the algorithm does not search on any other branches. The computational cost of the ideal situation is the same for both the depth-first BBD and the fast optimal BBD algorithm, and is

$O(K^2)$. When $\sigma \to 0$, it is expected that the average computational cost converges to the ideal one. For moderate SNRs, however, the average computational cost is affected by the correlation coefficients in the $\boldsymbol{H}$ matrix. Generally speaking, when the signal powers are not similar, or when user signature sequences are not highly correlated, the optimal detection problems are relatively easier to solve. On the other hand, when the correlations between user signatures are high and when the signal powers are similar, the optimal detection problem is deemed "hard" and the average computational cost will be high.

E. Nonbinary BBD

In a system where user signals are taken from a finite alphabet, the fast optimal BBD algorithm can be directly applied. In step 5), we should choose the node associated with b_k that minimizes $|[z_k]_k - b_k l_{kk}|$. The steps 8.1), 8.2), and 8.3) should be applied to the rest of the nodes, which should be sorted in an ascending order according to the values of $|[z_k]_k - b_k l_{kk}|$.

Alternatively, one could choose to treat an M-ary user as several binary users (e.g., $M = 8$, equivalent to three binary users). Both philosophies are applicable to any BBD method, including SD.

F. Upper Bound Initialization

As shown in the SD method, the statistical information can also be used in the initialization of the upper bound in BBD. Since the fast BBD algorithm ensures a minimum asymptotic average computational cost, the positive effect of the upper bound initialization in the high SNR regime is limited. In the moderate SNR regime, however, a proper initialization on the upper bound does reduce the average computational cost and speeds up the BBD process.

To further reduce the average computational cost, we recommend modifying steps 2), 3), and 11) in the fast BBD algorithm as follows.

2) Precompute $\tilde{\boldsymbol{y}} = \boldsymbol{L}^{-T}\boldsymbol{y}$. Precompute C (The choices for C are recommended below).
3) Initialize $k = 0$. $\boldsymbol{z}_k = \tilde{\boldsymbol{y}}$, $\xi_k = 0$, UPPER $= C$ and initialize K queues by $\forall k$, $\boldsymbol{q}_k = \text{NULL}$.
11) If no solution is available so far, let $C = C + \Delta C$ and go to step 3). Otherwise, stop and report the "current-best" solution.

Assume that the true solution $\bar{\boldsymbol{b}}$ is also the ML. The optimal cost is given by (17). Approximating $\|\boldsymbol{v}\|_2^2$ by a Gaussian random variable, we have

$$\|\boldsymbol{v}\|_2^2 \sim N(K\sigma^2, 2K\sigma^4). \tag{27}$$

When σ is available, we recommend

$$\begin{aligned} C &= K\sigma^2 + \sqrt{2K} l_{m_1 m_1}\sigma \\ \Delta C &= \sqrt{2K} l_{m_1 m_1}\sigma \end{aligned} \tag{28}$$

where $l_{m_1 m_1}$ is the minimum diagonal element of $\boldsymbol{L}$.

When σ is not available, we recommend the upper bound initialization proposed in [26]

$$\begin{aligned} C &= (K\log K + K\log\log K + 5K)\frac{|\det(\boldsymbol{L})|}{V_k} \\ \Delta C &= C \end{aligned} \tag{29}$$

where V_k is the volume of a sphere of radius one in the real space R^K.

When SNR is moderate, the recommended upper bound initializations save up to 1/3 of the average computational cost of the fast BBD algorithm (speedup 33%). However, for some SNRs, upper bound initialization (29) may result in a higher average computational cost, compared with the $C = +\infty$ initialization.

V. "Any-Time" Suboptimal Algorithm

Although the average computational costs may not be very high, the computational costs for the worst case of the proposed algorithms are still exponential in the number of users, since the ML solution is generally NP hard. In practical systems, when a strict limitation on computational cost exists, the "current-best" solution in the above BBD method can serve as a suboptimal alternative to the NP-hard optimal solution.

For the depth-first BBD algorithm, define the suboptimal detector that explores the subtree under and including level $K - k + 1$ to be $\phi_{BB-k}^{(D)}(k = 1, \ldots, K)$. From the above analysis of the computational cost, the worst-case computation for $\phi_{BB-k}^{(D)}$ is given by

$$\begin{aligned} \text{Multiplications} &\le \frac{K(K+3)}{2} + 3 * 2^{k-1} - k - 2 \\ \text{Additions} &\le K(K+1) + 5 * 2^k \\ &\quad - \frac{(k+3)(k+4)}{2}. \end{aligned} \tag{30}$$

Similar to (7), define the minimum distance among users $K - k + 1, \ldots, K$ by

$$d_{\min -k} = \sqrt{\min_{\substack{\boldsymbol{e}\in\{-1,0,1\}^K - \{\boldsymbol{0}\} \\ e_1,\ldots,e_{K-k}=0}} \boldsymbol{e}^T\boldsymbol{H}\boldsymbol{e}}. \tag{31}$$

Consequently, the SE measure for $\phi_{BB-k}^{(D)}$ is given by

$$E\left(\phi_{BB-k}^{(D)}\right) = \min\left(\min_{i=1,\ldots,K-k} l_{ii}^2, d_{\min -k}^2\right). \tag{32}$$

Furthermore, from the definitions of (31) and (7), we have

$$d_{\min -k}^2 \ge d_{\min}^2 = E(\phi_{\text{ML}}) \ge E\left(\phi_{BB-k}^{(D)}\right). \tag{33}$$

$E(\phi_{BB-k}^{(S)})$ can then be denoted by

$$E\left(\phi_{BB-k}^{(S)}\right) = \min\left(d_{\min}^2, \min_{i=1,\ldots,K-k} l_{ii}^2\right). \tag{34}$$

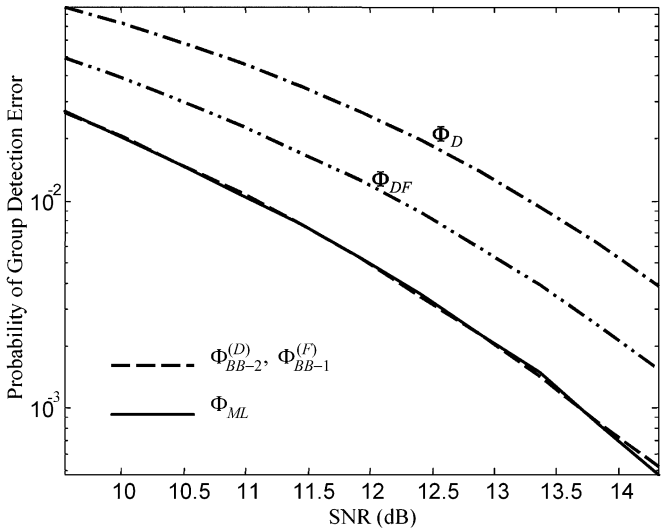


Fig. 3. Performance of various methods. (Three users, 10^6 Monte Carlo runs.)

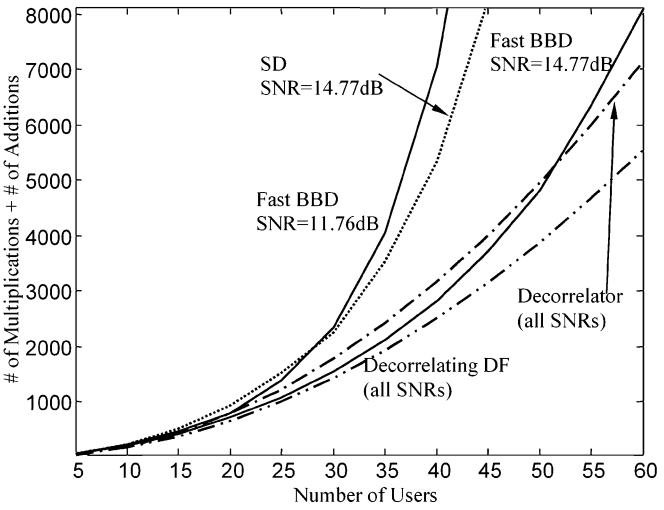


Fig. 4. Average computational costs versus number of users. (10^4 Monte Carlo runs.)

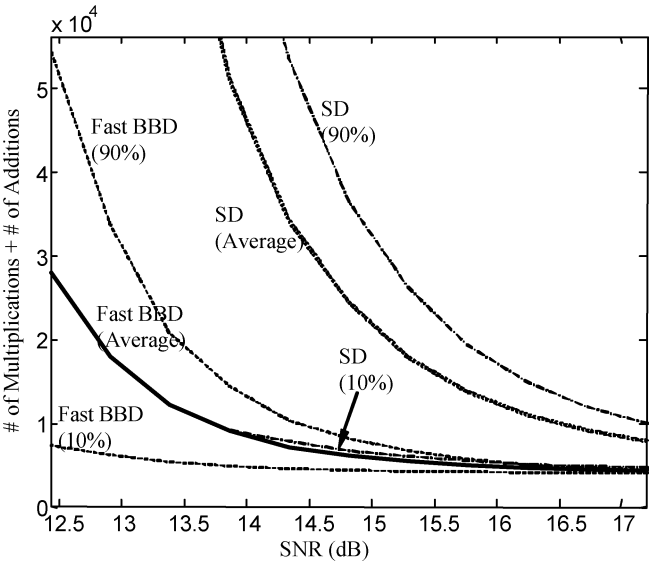


Fig. 5. Average computational cost versus SNR. (10^4 Monte Carlo runs.)

Based on the performance analysis, we have the following.

Proposition 3: When ordering users by the ordering algorithm, the SE $E(\phi^{(D)}_{BB-k})$ of all $k = 1, \ldots, K$ are maximized simultaneously.

The proof can be easily derived from [9, Prop. 2].

For the fast optimal BBD algorithm, we define the suboptimal detector that searches users $m_1, \ldots, m_k$ as $\phi^{(F)}_{BB-k}$. Similarly, the SE of $\phi^{(F)}_{BB-k}$ is given by

$$E\left(\phi^{(F)}_{BB-k}\right) = \min\left(d^2_{\min}, \min_{i=m_{k+1},\ldots,m_K} l^2_{ii}\right). \qquad (35)$$

The worst-case computational cost for $\phi^{(F)}_{BB-k}$, although tractable, is problem dependent, and therefore does not have a useful closed-form expression.

VI. Simulation Results

According to (30), the computational complexities for the suboptimal detector $\phi^{(D)}_{BB-k}$, as well as $\phi^{(F)}_{BB-k}$, are exponential in k. However, since we assume H to be known, the SE of the proposed "current-best" suboptimal solutions can be found offline by (34) and (35). The following simulation results show that, in some cases, a small amount of extra computation can significantly improve the performance of the detection system, when compared with the $\phi_{\rm DF}$ (which is the same as $\phi^{(D)}_{BB-1}$).

Example 1—Continued: In the previous example, since users 2 and 3 are strongly correlated, we expect that $E(\phi^{(D)}_{BB-2}) = E(\phi^{(F)}_{BB-1})$ will be a significant improvement over $E(\phi_{\rm DF})$. The SE for different detectors can be obtained via (6) and (34): $E(\phi_{\rm D}) = 0.836$, $E(\phi_{\rm DF}) = 0.92$, $E(\phi_{\rm ML}) = 1$, $E(\phi^{(D)}_{BB-2}) = E(\phi^{(F)}_{BB-1}) = 1$. The simulation result is given in Fig. 3, which is consistent with the theoretical analysis.

Example 2: In this example, we vary the number of users from 5 to 60. The ratio between the number of users and the signature length is fixed at 5/6. The binary signature sequences are randomly generated, and the user signal powers are set to be equal. For the suboptimal detectors, the SNR $= (w_{kk}/\sigma^2)$ is fixed at 14.77 dB. The upper bound initialization (28) is used by the fast BBD algorithm. Fig. 4 shows a comparison on the average computational costs of the fast optimal BBD algorithm, the SD method, the decorrelator, and the decorrelating DF method, at different SNRs. When SNR $= 14.77$ dB, the average computational cost of the fast BBD algorithm is comparable to that of the decorrelator for up to 60 users, and is significantly better than that of the SD method with the same SNR.

Example 3: In this example, we have 50 equal-powered users. The 53-length binary signature sequences are randomly generated. Fig. 5 shows the average computational costs versus SNR, together with the $10^{\rm th}$ and the $90^{\rm th}$ percentile curves, for the fast BBD and the SD algorithms. Upper bound initialization (28) is used in the fast BBD. The fast BBD is shown to significantly outperform the SD over all SNRs.

Example 4: In the last example, we have 50 equal-powered users, again with randomly generated 53-length binary signature sequences. We set SNR so that the probability of error of the ML detector is around 10^{-3}. This is considered "hard" for all

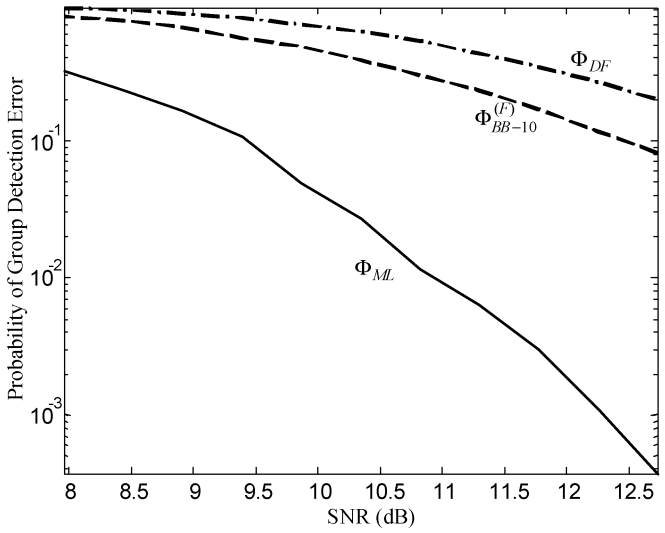


Fig. 6. Performance comparisons (50 users, 53-length random signatures, 10^5 Monte Carlo runs for the suboptimal algorithms, dynamic Monte Carlo runs for the ML detector).

TABLE I
ACTUAL NUMBER OF MONTE-CARLO RUNS FOR ML DETECTOR (FOR 100 ERRORS)

SNR (dB)	8.0	8.9	9.9	10.8	11.8	12.7
# of runs	311	602	2036	8675	3e+4	3e+5

optimal multiuser detection algorithms. However, the fast BBD algorithm is able to obtain the performance of the ML detector with reasonable computation. In computer simulation, we use upper bound initialization (28) for the fast BBD algorithm. We also use the following dynamic Monte Carlo simulation technique in order to avoid unnecessary simulations. In the simulation, instead of fixing the number of Monte Carlo runs, we stop the simulation whenever a certain number of errors are observed. (In this example, for each simulation point, we stop the fast BBD whenever 100 errors are obtained.) Fig. 6 shows the performances of the decorrelating DF detector, the ML detector, and the suboptimal detector $\phi^{(F)}_{BB-10}$. The actual numbers of Monte Carlo simulations for different SNRs are given in Table I.

VII. CONCLUSION

A fast BBD-based optimal algorithm for the multiuser detection of symbol-synchronous CDMA is proposed. Due to the use of statistical information embedded in the system model, the average computational cost has been significantly reduced. The proposed fast BBD optimal algorithm is able to simulate a 50-user bandwidth-efficient system with equal user powers. It is found to outperform the SD method for all SNRs and all numbers of users in the binary signaling case. Comparisons for the M-ary modulation cases are in progress. A suboptimal "any-time" algorithm is also proposed by simply picking the "current-best" solution in the BBD search. Theoretical analysis on both the asymptotic performance and the computational complexity are given.

ACKNOWLEDGMENT

The authors would like to thank J. Durand and Dr. L. Brunel for their help on the SD method; and would also like to thank Dr. L. Brunel for pointing out the memory requirements of the proposed search strategy in Section IV-B.

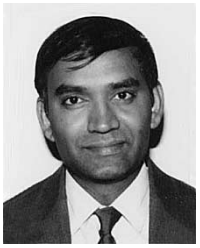
REFERENCES

[1] J. Luo, K. Pattipati, P. Willett, and G. Levchuk, "Fast optimal and suboptimal any-time algorithms for CWMA multiuser detection," in *Proc. IEEE Int. Symp. Information Theory*, Washington, DC, June 2001, p. 11.
[2] J. Luo, K. Pattipati, P. Willett, and L. Brunel, "Branch-and-bound-based fast optimal algorithm for multiuser detection in synchronous CDMA," in *Proc. IEEE Int. Conf. Communications*, Anchorage, AK, May 2003, pp. 3336–3340.
[3] S. Verdu, "Minimum probability of error for asynchronous Gaussian multiple-access channel," *IEEE Trans. Inform. Theory*, vol. IT-32, pp. 85–96, Jan. 1986.
[4] R. Lupas and S. Verdu, "Linear multiuser detectors for synchronous code-division multiple-access channels," *IEEE Trans. Inform. Theory*, vol. 35, pp. 123–136, Jan. 1989.
[5] Z. Xie, R. Short, and C. Rushforth, "A family of suboptimum detectors for coherent multiuser communications," *IEEE Trans. Commun.*, vol. 8, pp. 683–690, May 1990.
[6] M. K. Varanasi, "Multistage detection for asynchronous code-division multiple-access communications," *IEEE Trans. Commun.*, vol. 38, pp. 509–519, Apr. 1990.
[7] M. K. Varanasi and B. Aazhang, "Near-optimum detection in synchronous code-division multiple-access systems," *IEEE Trans. Commun.*, vol. 39, pp. 725–736, May 1991.
[8] M. K. Varanasi, "Group detection for synchronous Gaussian code-division multiple-access channels," *IEEE Trans. Inform. Theory*, vol. 41, pp. 1083–1096, July 1995.
[9] J. Luo, K. Pattipati, and P. Willett, "Optimal grouping algorithm for a group decision feedback detector in synchronous CDMA communications," *IEEE Trans. Commun.*, vol. 51, pp. 341–346, Mar. 2003.
[10] A. Duel-Hallen, "Decorrelating decision-feedback multiuser detector for synchronous code-division multiple-access channel," *IEEE Trans. Commun.*, vol. 41, pp. 285–290, Feb. 1993.
[11] ——, "A family of multiuser decision-feedback detectors for asynchronous code-division multiple-access channels," *IEEE Trans. Commun.*, vol. 43, pp. 421–432, Feb.-Apr. 1995.
[12] M. K. Varanasi, "Decision feedback multiuser detection: a systematic approach," *IEEE Trans. Inform. Theory*, vol. 45, pp. 219–240, Jan. 1999.
[13] P. Tan and L. Rasmussen, "The application of semidefinite programming for detection in CDMA," *IEEE J. Select. Areas Commun.*, vol. 19, pp. 1442–1449, Aug. 2001.
[14] W. Ma, T. Davison, K. Wong, Z. Luo, and P. Ching, "Quasi-maximum-likelihood multiuser detection using semidefinite relaxation with application to synchronous CDMA," *IEEE Trans. Signal Processing*, vol. 50, pp. 912–922, Apr. 2002.
[15] J. Luo, K. Pattipati, P. Willett, and F. Hasegawa, "Near-optimal multiuser detection in synchronous CDMA using probabilistic data association," *IEEE Commun. Lett.*, vol. 5, pp. 361–363, Sept. 2001.
[16] J. Luo, K. Pattipati, and P. Willett, "A sliding window PDA for asynchronous CDMA, and a proposal for deliberate asynchronicity," *IEEE Trans. Commun.*, vol. 51, pp. 1970–1974, Dec. 2003.
[17] F. Hasegawa, J. Luo, K. Pattipati, P. Willett, and D. Pham, "Speed and accuracy comparison of techniques for multiuser detection in synchronous CDMA," *IEEE Trans. Commun.*, vol. 52, pp. 540–545, Apr. 2004.
[18] S. Verdu, "Computational complexity of optimum multiuser detection," *Algorithmica*, vol. 4, pp. 303–312, 1989.
[19] S. Verdu and H. Poor, "Abstract dynamic programming models under commutativity conditions," *SIAM J. Control Optim.*, vol. 25, pp. 990–1006, July 1987.
[20] C. Papadimitriou and K. Steiglitz, *Combinatorial Optimization—Algorithms and Complexity*. Englewood Cliffs, NJ: Prentice-Hall, 1982.
[21] D. Bertsekas, *Network Optimization, Continuous and Discrete Models*. Belmont, MA: Athena Scientific, 1998, ch. 10, pp. 483–492.
[22] B. Paris, "Finite-precision decorrelating receivers for multi-user CDMA communication systems," *IEEE Trans. Commun.*, vol. 44, pp. 496–507, Apr. 1996.
[23] C. Schlegel and L. Wei, "A simple way to compute the minimum distance in multiuser CDMA systems," *IEEE Trans. Commun.*, vol. 45, pp. 532–535, May 1997.

[24] C. SanKaran and A. Ephremides, "Solving a class of optimum multiuser detection problems with polynomial complexity," *IEEE Trans. Inform. Theory*, vol. 44, pp. 1958–1961, Sept. 1998.
[25] S. Ulukus and R. Yates, "Optimum multiuser detection is tractable for synchronous CDMA systems using M-sequences," *IEEE Commun. Lett.*, vol. 2, pp. 89–91, Feb. 1998.
[26] L. Brunel and J. Boutros, "Lattice decoding for joint detection in direct sequence CDMA systems," *IEEE Trans. Inform. Theory*, to be published.
[27] B. Hassibi and H. Vikalo, "On the expected complexity of integer least-squares problems," in *Proc. IEEE ICASSP*, vol. 2, Orlando, FL, May 2002, pp. 1497–1500.
[28] U. Fincke and M. Pohst, "Improved algorithms on integer programming and related lattice problems," in *Proc. 15th Annu. ACM Symp. Theory of Computing*, 1983, pp. 193–206.
[29] P. Wolniansky, G. Foschini, G. Golden, and R. Valenzuela, "V-blast: an architecture for realizing very high data rates over the rich-scattering wireless channel," in *Proc. ISSE*, Pisa, Italy, Sept. 1998, pp. 295–300.
[30] J. Luo, "Improved multiuser detection in code-division multiple access communications," Ph.D. dissertation, Elect. Comput. Eng. Dept., Univ. Connecticut, Storrs, CT, May 2002.

Jie Luo (S'00–M'03) received the B.S. and M.S. degrees in electrical engineering from Fudan University, Shanghai, China, in 1995 and 1998, respectively, and the Ph.D. degree in electrical engineering from University of Connecticut, Storrs, in 2002.

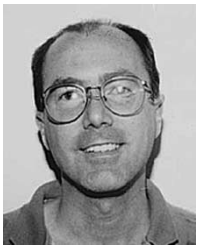
Currently, he is a Research Associate at the Institute for Systems Research, University of Maryland, College Park. His current research interests are in ad hoc networking and wireless communication.



Krishna R. Pattipati (S'77–M'80–SM'91–F'95) received the B.Tech. degree in electrical engineering with highest honors from the Indian Institute of Technology, Kharagpur, India, in 1975, and the M.S. and Ph.D. degrees in systems engineering from the University of Connecticut, Storrs, in 1977 and 1980, respectively.

From 1980 to 1986, he was with Alphatech, Inc., Burlington, MA. Since 1986, he has been with the University of Connecticut, Storrs, where he is a Professor of Electrical and Computer Engineering. His current research interests are in the areas of adaptive organizations for dynamic and uncertain environments, multiuser detection in wireless communications, signal processing and diagnosis techniques for power quality monitoring, multi-object tracking, and scheduling of parallelizable tasks on multiprocessor systems. He has published over 270 articles, primarily in the application of systems theory and optimization (continuous and discrete) techniques to large-scale systems. He has served as a consultant to Alphatech, Inc., and IBM Research and Development, and is a cofounder of Qualtech Systems, Inc., a small business specializing in advanced integrated diagnostics software tools.

Dr. Pattipati was selected by the IEEE Systems, Man, and Cybernetics (SMC) Society as the Outstanding Young Engineer of 1984, and received the Centennial Key to the Future award. He has served as the Editor-in-Chief of the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: PART B–CYBERNETICS during 1998–2001, Vice-President for Technical Activities of the IEEE SMC Society (1998–1999), and as Vice-President for Conferences and Meetings of the IEEE SMC Society (2000–2001). He was co-recipient of the Andrew P. Sage Award for the Best SMC Transactions Paper for 1999, the Barry Carlton Award for the Best AES Transactions Paper for 2000, the 2002 NASA Space Act Award for "A Comprehensive Toolset for Model-Based Health Monitoring and Diagnosis," and the 2003 AAUP Research Excellence Award at the University of Connecticut. He also won the best technical paper awards at the 1985, 1990, 1994, and 2002 IEEE AUTOTEST Conferences, and at the 1997 Command and Control Conference.



Peter Willett (S'83–M'86–SM'97–F'03) received the B.A.Sc. degree in 1982 from the University of Toronto, Toronto, ON, Canada, and the Ph.D. degree in 1986 from Princeton University, Princeton, NJ.

He is a Professor of Electrical and Computer Engineering at the University of Connecticut, Storrs. He has written, among other topics, about the processing of signals from volumetric arrays, decentralized detection, information theory, CDMA, learning from data, target tracking, and transient detection.

Dr. Willett is a member of the IEEE AES Society's Board of Governors, and is a member of the IEEE Signal Processing Society's SAM technical committee. He is an Associate Editor both for the IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS and for the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS. He is a track organizer for Remote Sensing at the IEEE Aerospace Conference (2001–2003), and was Co-Chair of the Diagnostics, Prognosis, and System Health Management SPIE Conference in Orlando. He also served as Program Co-Chair for the 2003 IEEE Systems, Man, and Cybernetics Conference in Washington, DC.

Georgiy M. Levchuk (S'00–M'03) was born on May 27, 1973 in Kiev, Ukraine. He received the B.S. and M.S. degrees in mathematics with highest honors from the Kiev Taras Shevchenko University, Kiev, Ukraine, in 1995, and the Ph.D. degree in electrical engineering from University of Connecticut, Storrs, in 2003.

He is currently a Simulation and Optimization Engineer with Aptima, Inc., Woburn, MA. His research interests include global, multi-objective optimization and its applications in the areas of organizational design and adaptation, and network optimization. Prior to joining Aptima, he held a Research Assistant position with the Institute of Mathematics, Kiev, Ukraine, a Teaching Assistantship at Northeastern University, Boston, MA, and a Research Assistantship at the University of Connecticut, working on projects sponsored by the Office of Naval Research.

Dr. Levchuk received Best Student Paper Awards at both the 2002 and 2003 International Command and Control Research and Technology Symposia.