SMART MAXIMUM-LIKELIHOOD CDMA MULTIUSER DETECTION

Saeed Fouladi Fard and Chintha Tellambura

Department of Electrical and Computer Engineering
University of Alberta, Edmonton, AB, Canada, T6G 2V4
Email:{saeed, chintha}@ece.ualberta.ca

ABSTRACT

A fast-low-complexity maximum-likelihood multiuser detector for code division multiple access systems is proposed. It utilizes a best-first branch and bound approach and a constrained memory search. The memory constraint makes the proposed detector realizable, but has negligible performance loss in low signal-to-noise ratio (SNR) and virtually no performance loss in medium-to-high SNR. The results show that the proposed algorithm is much less complex, and that it executes faster than existing maximum likelihood detectors. In addition, the proposed method is suitable for parallel implementation.

1. INTRODUCTION

The complexity of optimum multiuser detection for code division multiple access (CDMA) systems increases exponentially with the number of users in general [1]. Even if the bit error rate (BER) of a particular detector is good, computational complexity can render it unsuitable for practical implementation (e.g., optimal detection with exhaustive search). Previous research over almost two decades has focused on suboptimal receivers with low computational complexity. Such suboptimal detectors include decorrelator, minimum mean square error (MMSE), generalized MMSE, soft interference canceller [2], decision-feedback (DF) detector [3], probabilistic data association (PDA) detector [4], and multistage detector [5].

While optimal multiuser detection is generally NP-hard (i.e., there is unlikely to be a polynomial time algorithm for all problem instances), there can be many problem instances that are solvable with polynomial complexity. Branch-&-Bound (B&B) is a divide and conquer framework for such hard combinatorial optimization problems [6]. The basic idea is to partition the solution set of a discrete optimization problem into successively smaller subsets (branch), bound the cost function value over each subset and use the bounds to eliminate some subsets (bound). The procedure ends when the whole solution set has been implicitly searched. Since a B&B algorithm effectively searches the entire solution space, the best solution of B&B is a global optimum. This approach is most efficient when it is possible to discard many subsets as early as possible during the branching process without actually evaluating them. For example, the n-city travelling salesman problem, a well-known classical combinatorial problem with worst-case exponential complexity in n, can be solved exactly in time $O(n^{\alpha})$ with $\alpha < 3$ (on average) using B&B.

This work has been supported in part by the Natural Sciences and Engineering Research Council of Canada and the Alberta Ingenuity Fund.

Fairly recently, B&B detectors have been developed for optimal multiuser detection. Luo et al. [7] propose a detector based on depth-first B&B [8]. Brunel and Boutros [9] propose a sphere decoder (SD) for optimal multiuser detection. Luo et al. [10] show that the SD is a form of depth-first B&B. These studies show that significant computational savings can be had by the use of B&B for multiuser detection.

B&B can also be viewed as a tree search algorithm where each subset is represented by a node in a tree, and the root of the tree represents the entire solution space. Each node is associated with a cost that is a lower bound to the global optimum. As a result, if a node cost exceeds the current best solution, the node can be pruned – i.e., the children of the node can be discarded without a loss (assuming a minimization problem). The algorithm keeps a list (queue) of nodes to be processed. When a node is expanded, its children are generated and their costs are evaluated, and those child nodes whose cost are less than the current best solution are added to the list.

A B&B algorithm can explore the search tree in several ways. Depth-first expands a node with the largest level in the search tree. The memory required by this method is proportional to the product of the number of levels in the search tree and the maximum number of children of any node. So for a finite-depth tree, this results in fairly low memory requirements. Most detection problems fall into this category. However, this algorithm may have to perform large amounts of unnecessary computations before it finds a global optimum. On the other hand, best-first expands nodes in least-cost first order from the priority queue [11]. Therefore, it searches fewer nodes than depth-first [12]. Unfortunately, this algorithm may have to store the entire tree in a worst case scenario. Its memory requirements are exponential in terms of the number of levels in the search tree.

In this paper, we propose a modified version of best-first B&B for optimal multiuser detection. All existing B&B multiuser detectors use a depth-first type. Best-first always expands fewer nodes than depth-first and has less computational complexity. As a result, our proposed detector has less time complexity than pervious B&B detectors, and the variance of its execution time remains fixed over a wide range of SNR. This behavior differs sharply from previous B&B detectors whose time complexity varies rapidly with SNR. In addition, best-first B&B is a parallel method, making it more suitable for real time implementation. However, the memory requirements for the best-first B&B can be very large [13], which is a main weakness of this approach. In this paper, we also propose a technique to reduce the memory requirements of the new detection algorithm. The memory constraint results in almost no performance loss in medium-to-high SNR and a negligible loss in low SNR.

Algorithm 1 Proposed Best-First BnB for OMUD

```
function BestFirstBnB(){
    $\hat{\mathbf{b}} = \mathbf{0}_{K\times 1}$; $UpperBound = +\infty$;
    $k = 0$; $\tilde{\mathbf{b}} = \mathbf{0}_{K\times 1}$; $\mathbf{z} = \tilde{\mathbf{y}}$; $\xi = 0$;
    put $(k, \tilde{\mathbf{b}}, \mathbf{z}, \xi)$ in the queue
    while(queue is not empty) {
        Remove $(k, \tilde{\mathbf{b}}, \mathbf{z}, \xi)$ from the first place of queue
        $k = k + 1$; $cf = 0$; $\bar{\mathbf{z}} = \mathbf{z}$;
        if $(k > 1)$ $z_k = z_k - \tilde{b}_{k-1} l_{k,k-1}$;
        if $(k == K)$ {
            $\tilde{b}_k = \mathrm{sign}(z_k)$; $\bar{\xi} = \xi + \left|\tilde{b}_k l_{k,k} - z_k\right|^2$;
            if($\bar{\xi} < UpperBound$){
                $UpperBound = \bar{\xi}$; $\hat{\mathbf{b}} = \tilde{\mathbf{b}}$;
                Clean the queue from out-of-bound nodes
            } $/ * if * /$
        } else {
            for $\tilde{b}_k = -1, +1$ {
                $\bar{z}_k = z_k - \tilde{b}_k l_{k,k}$; $\bar{\xi} = \xi + \left|\tilde{b}_k l_{k,k} - z_k\right|^2$;
                if $(\bar{\xi} < UpperBound)$ {
                    if $((k > 1)$ and $(cf == 0))$ {
                        for $(j = k + 1 : K)$ $\bar{z}_j = z_j - \tilde{b}_{k-1} l_{j,k-1}$;
                        $cf = 1$;
                    } $/ * if * /$
                    add2Queue($k, \tilde{\mathbf{b}}, \bar{\mathbf{z}}, \bar{\xi}$);
                } $/ * if * /$
            } $/ * for * /$
        } $/ * else * /$
    } $/ * while * /$
    return $\hat{\mathbf{b}}$;
} $/ * function * /$
```

2. PROBLEM STATEMENT

We consider a synchronous DS-CDMA system with K active users. The discrete-time matched filter output of a CDMA system is [1]

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

where $\mathbf{b} \in \{-1, +1\}^K$ is the $K \times 1$ vector of user transmitted bits. The positive definite signature waveform correlation matrix is $\mathbf{H} = \mathbf{W}\mathbf{R}\mathbf{W} = \mathbf{L}^T\mathbf{L}$, where $\mathbf{W}$ is a diagonal matrix with the square roots of received signal energies, $\mathbf{R}$ is the symmetric normalized correlation matrix of K user signature waveforms of length N, $\mathbf{L}$ is the lower triangular Cholesky factorization of $\mathbf{H}$, and $\mathbf{n}$ is a real zero-mean Gaussian random vector with covariance matrix $E\{\mathbf{n}\mathbf{n}^T\} = \sigma_n^2\mathbf{H}$.

The maximum Likelihood (ML) detector solves the combinational optimization problem

$$\hat{\mathbf{b}}_{ML} = \underset{\mathbf{x}\in\{-1,+1\}^K}{\arg\min} \ ||\mathbf{L}\mathbf{x} - \tilde{\mathbf{y}}||_2^2 \tag{2}$$

where $\tilde{\mathbf{y}} = \mathbf{L}^{-T}\mathbf{y} = \mathbf{L}\mathbf{b} + \mathbf{v}$, and $\mathbf{v} = \mathbf{L}^{-T}\mathbf{n}$ contains white real Gaussian noise samples with covariance matrix $\sigma_n^2\mathbf{I}_K$. For exhaustive search the computational complexity of (2) grows exponentially with the number of active users, which becomes prohibitive for moderate to large number of users.

3. SEARCH ALGORITHMS

Since $\mathbf{L}$ is lower triangular, (2) can be rearranged as

$$\begin{aligned}
\hat{\mathbf{b}}_{ML} &= \underset{\mathbf{x}\in\{-1,+1\}^K}{\arg\min} \ ||\mathbf{L}\mathbf{x} - \tilde{\mathbf{y}}||_2^2 \\
&= \underset{\mathbf{x}\in\{-1,+1\}^K}{\arg\min} \sum_{i=1}^{K} |x_i l_{i,i} + \sum_{j=1}^{i-1} x_j l_{i,j} - \tilde{y}_i|^2 \\
&= \underset{\mathbf{x}\in\{-1,+1\}^K}{\arg\min} \sum_{i=1}^{K} |d_i - \tilde{y}_i|^2 \qquad (3)
\end{aligned}$$

where $\mathbf{d} = \mathbf{L}\mathbf{x}$. Note that d_i only depends on $x_{j\leq i}$. A level-k node is associated with estimates $\hat{b}_1, \hat{b}_2, ..., \hat{b}_k$ and has the partial cost

$$\xi_k = \sum_{i=1}^{k} (d_i - \tilde{y}_i)^2 \,. \tag{4}$$

Using (3) and (4), it is possible to find the solution of (2) on a tree-search basis. In this tree, a node is represented by the bits estimated up to that node (i.e., path) and the lower bound of cost corresponding to the estimated bits. The final solution is the leaf node of the tree with the lowest cost. Different B&B techniques exist for scanning the tree to find the minimum cost node [11]. Depth-first methods try to get to the depth of the tree as fast as possible to get a feasible solution. The main property of a depth-first search is that it completely explores one subtree before going to the next subtree. Depth-first has the advantage of getting to the bottom of the tree fastest, and therefore it uncovers good solutions (but not necessarily optimal solutions) fairly quickly.

Best-first search, on the other hand, always expands the best node first [11]. The nodes with lower costs are considered as the most promising ones. Best-first is much more selective about which part of the tree it explores. If an attractive subtree becomes less promising, the best first algorithm leaves it and hops to another part of the tree. The best-first search never searches more nodes than depth-first and will always search fewer nodes [12].

Best-first search is implemented using a priority queue. The unexpanded tree nodes are stored in a priority queue, and are sorted so that the most promising node is at the head (first place) of the queue. A record of the "best solution so far" and the cost upper bound is kept updated. Algorithm 1 shows the pseudo-code of the proposed best-first B&B algorithm for optimal multiuser detection.

The function BestFirstBnB() is the entry point of Algorithm 1, and the final search result returns in $\hat{\mathbf{b}}$. In this algorithm, after initializations (line 2), the root node is inserted in the priority queue as the first node to be explored (lines 3-4). The priority of each node is determined according to its cost (ξ) and the node with the lowest cost is processed first. The search continues as long as there are nodes left in the priority queue. The number of nodes in the queue increases whenever the algorithm expands a node and puts its children in the priority queue using a function add2Queue(). Also the queue is cleaned of out of bound nodes each time the algorithm reaches a new bound at the top nodes of the tree (line 13).

While they are the most efficient in terms of computational complexity, the best-first search methods may require prohibitively large amounts of memory for the priority queue. As well, sorting of the priority queue requires computations. To overcome these problems, we propose to use a limited amount of memory and keep the nodes in a sorted linked list structure, avoiding the need to sort the queue every time a new node is added to it. The new nodes are added to the queue as long as there is enough memory available. In case there is not enough memory left, the cost to depth ratio of the new node is compared to that of the last node of the queue (the node with the lowest priority). The new node replaces the last node if it has a lower cost to depth ratio. This memory constraint results in a negligible performance loss in low SNR and virtually no loss otherwise. If the queue size is large enough, the proposed algorithm results in ML solution over the entire range of SNR, but for the systems with a large number of users especially in the low SNR region where detection requires large amounts of memory, the proposed method is a good aid to find an acceptable solution.

4. SIMULATION RESULTS

We simulated a CDMA system over a Rayleigh flat fading channel with $K = 30$ active users each using a random signature sequence of length $N = 36$. For the proposed detector, users are ordered such that the norms of the columns of $\mathbf{H}$ are of descending order, and the size of the priority queue is set to $3 \times K$. Fig. 1 shows the BER performance of the proposed detector and that of the ML detector (exhaustive search). Both of the performance curves are virtually indistinguishable. Extremely small differences can be observed in low SNR (e.g., around 5dB). We can readily conclude that the memory constraint of three times the number users has negligible impact on the detector performance.

In our simulations, all of the detector algorithms are implemented in the C language (as time measurements in MATLAB are not reliable) and run on a dedicated Pentium-IV processor. Fig. 2 compares the average detection time (in Seconds) of the detectors. Our proposed detector is about 100 times faster than the depth-first B&B detector in low SNR. All the detectors have roughly the same search time for high SNR. Of course, computational time can be influenced by many factors. So perhaps a better measure of complexity is given by the average number of floating point operations (FLOPs). Fig. 3 compares the computational complexity of the proposed method with the Sphere Decoder, depth-first B&B and the Fast B&B proposed in [10]. In this case, our detector provides significant computational savings. The average complexity of our detector is 10% to 1% of those of other detectors in low SNR. The average complexity of our detector is 10% of those of the other detectors in high SNR.

5. CONCLUSION

We have proposed a new B&B algorithm for optimal multiuser detection in CDMA. It uses the best-first strategy and imposes a limit on the size of the priority queue. This memory constraint results in a negligible performance loss in low SNR. In terms of computational complexity and execution time, our proposed detector outperforms the sphere decoder, depth-first B&B and fast B&B [10].

6. REFERENCES

[1] S. Verdú, *Multiuser Detection*, Cambridge University Press, Cambridge, U.K., 1998.

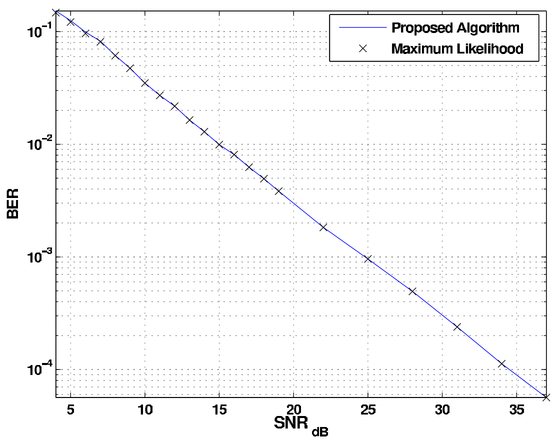

Fig. 1. BER of the proposed algorithm.

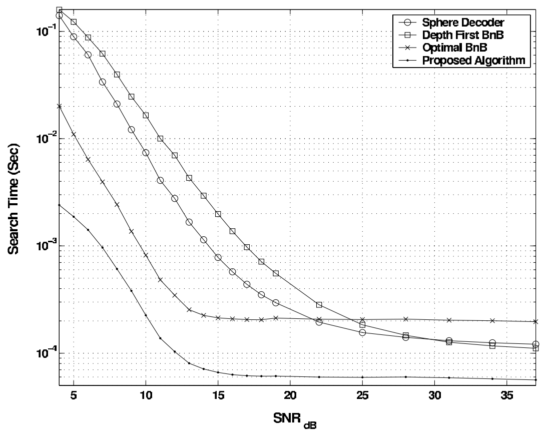

Fig. 2. Average tree search time of different algorithms.

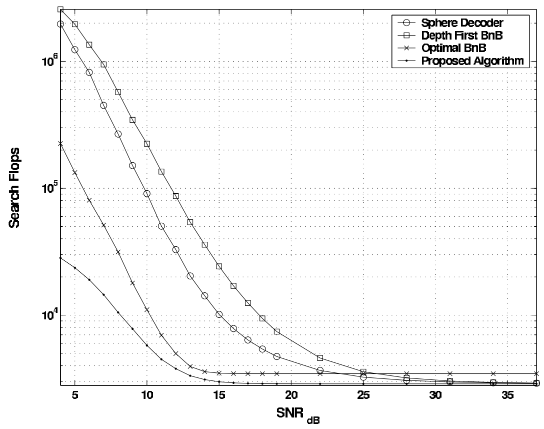

Fig. 3. Average number of FLOPs for multiuser detection.

[2] A. Yener, R.D. Yates, and S. Ulukus, "CDMA multiuser detection: A nonlinear approach," *IEEE Trans. Commun.*, vol. 50, pp. 1016–1024, June 2002.

[3] A. Duel Hallen, "Decorrelating decision-feedback multiuser detector for synchoronous code-division multiple-access channel," *IEEE Trans. Commun.*, vol. 41, pp. 285–290, Feb. 1993.

[4] J. Luo, K. Pattipati, P. Willet, and F. Hassegawa, "Near optimal multiuser detection in synchronous CDMA using probabilistic data association," *IEEE Commun. Lett*, vol. 5, pp. 361–363, Sept. 2001.

[5] M. K. Varanasi and B. Aazhang, "Near optimal detection in synchronous code-division multiple-access systems," *IEEE Trans. Commun.*, vol. 39, pp. 725–736, May 1991.

[6] C. H. Papadimitriou and K. Stieglitz, *Combinatorial Optimization: Algorithms and Complexity*, Prentice Hall, Englewood Cliffs, NJ, 1982.

[7] J. Luo, K. Pattipati, P. Willett, and L. Brunel, "Branch-and-bound-based fast optimal algorithm for multiuser detection in synchronous CDMA," in *Communications, 2003. ICC '03. IEEE International Conference on*, May 2003, vol. 5, pp. 3336–3340.

[8] D. Bertsekas, *Network optimization, Continuous and Discrete Models*, Athena Scientific, Belmont, MA, 1998.

[9] L. Brunel and J. J. Boutros, "Lattice decoding for joint detection in direct sequence CDMA systems," *IEEE Trans. Inform. Theory*, vol. 49, pp. 1030–1037, Apr. 2003.

[10] J. Luo, K. R. Pattipati, P. Willett, and G.M. Levchuk, "Fast optimal and suboptimal any-time algorithms for CDMA multiuser detection based on branch and bound," *IEEE Trans. Commun.*, vol. 52, pp. 632–642, Apr. 2004.

[11] N. J. Nilsson, *Artificial Intelligence: A New Synthesis*, Morgan Kaufmann Publishers, San Francisco, CA, 1998.

[12] E. W. Felten, "Best-first branch-and-bound on a hypercube," in *The Third Conference on Hypercube Concurrent Computers and Applications*, New York, N.Y., 1988, Association for Computing Machinery.

[13] N. R. Mahapatra and S. Dutt, "Sequential and parallel branch-and-bound search under limited-memory constraints," in *The IMA Volumes in Mathematics and its Applications—Parallel Processing of Discrete Problems, P. Pardalos (ed.)*, New York, 1999, Springer-Verlag.