List Detection for Symmetric Multi-Access Channels

Aaron Reid[1], Alex Grant
Institute for Telecommunications Research
University of South Australia
Mawson Lakes SA 5095 Australia
e-mail: `areid@spri.levels.unisa.edu.au`
e-mail: `alex.grant@unisa.edu.au`

Paul Alexander
Southern Poro Communications
6A Nelson Street
Annandale NSW 2038 Australia
e-mail: `palex@spri.levels.unisa.edu.au`

Abstract — **We propose a low complexity multiuser iterative decoder. We perform the multiuser a-posteriori probability calculation based on the marginalization of probabilities over a subset of the P highest probable sequences received over a K user symmetric channel which may be approximated closely with per-bit computational complexity $O(K + P + 2K \log K)$.**

I. Introduction

A code-division multiple-access (CDMA) system may be viewed as a serially-concatenated system and many iterative decoders have been proposed for these systems. The main contribution of this paper is an algorithm that approximates *a-posteriori* probability (APP) calculation with low complexity for symmetric CDMA channels (in which all the cross-correlations are identical). This algorithm reduces the complexity required by finding a list of P sequences with high *a-posteriori* probability and marginalizing only over this list. Numerical investigations have shown that typically the size of the list required is very small compared to the total number of sequences. For the symmetric channel, we give a polynomial complexity algorithm for finding such this list. We additionally show how this idea may be applied to any system for which there exists a polynomially complex optimal detection algorithm.

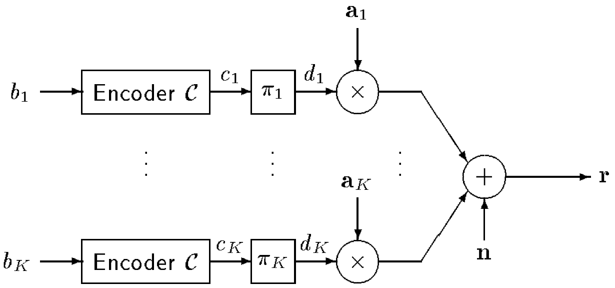

Figure 1: CDMA Channel Model

With reference to Figure 1, each user $k = 1, 2, \ldots, K$ encodes their binary information sequence b_k using a rate R code $\mathcal{C}$. Each user independently permutes their encoded sequence c_k with an interleaver π_k. We denote the binary sequence output from the interleaver of user k as $d_k[i]$, where i is the symbol time. Transmission is assumed to be symbol-synchronous. The received signal in the i-th symbol duration is given by $\mathbf{r}[i] = \mathbf{A}\mathbf{d}[i] + \mathbf{n}[i]$, where $\mathbf{A}$ is an $N \times K$ real or complex matrix whose unit energy columns are the discrete signature signals $\mathbf{a}_k[i]$ of the K users, $\mathbf{d}[i]$ is a length K column vector with elements $d_j[i] \in \{-1, +1\}$ (for binary phase-shift keying) being the transmitted binary symbol for user j, and $\mathbf{n}[i]$ is a sampled circularly symmetric complex noise vector with covariance matrix $E[\mathbf{n}\mathbf{n}^*] = \sigma^2 \mathbf{I}_K$.

Figure 2 shows an iterative decoder for this system. The multiuser APP operates on a per-bit basis (ignoring constraints imposed by $\mathcal{C}$) and we therefore drop the symbol index. Consider half-iteration n. Given the received vector $\mathbf{r}$ and prior probabilities $p_k^{(n)}(d)$ on each user's bits the multiuser APP produces extrinsic probabilities for each user according to

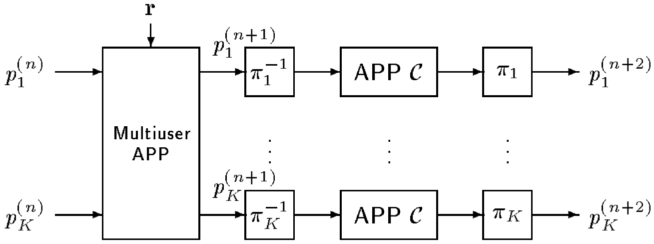

Figure 2: Iterative Receiver Structure

$$p_k^{(n+1)}(d) = \frac{1}{p(\mathbf{r})p_k^{(n)}(d)} \sum_{\mathbf{d}:d_k=d} p(\mathbf{r} \mid \mathbf{d}) \prod_{j=1}^{K} p_j^{(n)}(d_j) \qquad (1)$$

where the product term is the contribution of the a-priori information received from the previous iteration (at the first iteration the priors are all uniform) and $p(\mathbf{r} \mid \mathbf{d})$ is the probability of receiving $\mathbf{r}$ given $\mathbf{d}$ (specified by the channel model). For the second half of the iteration, user k performs single-user symbol-wise APP decoding of $\mathcal{C}$ using $p_k^{(n+1)}$ as priors, and outputs extrinsic probabilities $p_k^{(n+2)}(d_k)$ which serve as priors for the next half-iteration.

---

[1] This work was supported by Southern-Poro Communications and the Australian Government under ARC SPIRT C00002232.

The number of $\mathbf{d}$ with $d_i = d$ is 2^{K-1} and hence the marginalization (1) is intractable for large K. We will approximate (1) using the key empirical observation that the vast majority of the $\mathbf{d}$ sequences contribute a negligible amount to the total probability when performing the marginalization of each coded bit.

II. APPROXIMATE JOINT APP

We propose to decrease the complexity of the calculation of (1) by summing over a high-probability subset of the sequences $\mathbf{d}$. Suppose we can find a *prior list* of the P sequences $\mathbf{d}$ with the largest prior probability, $\prod_{k=1}^{K} p_k(d_k)$ (dropping the iteration index) and another *channel list* of the P sequences $\mathbf{d}$ with the highest channel probability, $p(\mathbf{r}|\mathbf{d})$. Once both lists have been found, we may then merge them into a single list of size P by taking the intersection of the sequences from both lists and then filling the remaining list positions with the next highest probable $\mathbf{d}$ from either list. Note that this procedure does not guarantee that we obtain even the most probable sequence according to $p(\mathbf{r}, \mathbf{d})$.

Clearly, the channel list needs to be calculated only once. The prior list must be updated every iteration. For the first iteration we calculate the overall list based purely on the channel list, since the priors give no additional information at this stage.

The prior list may be calculated very easily. Create a graph with $(K + 1)$ nodes and place parallel directed edges from the first node to the second and so forth to the node $(K + 1)$. Label the edges from node k to node $k + 1$ with $-\log p_k(+1)$ and $-\log p_k(-1)$, $k = 1, \ldots, K$. With this representation we may find the P most probable sequences using the P Shortest Paths Algorithm [1] which has complexity $O(K + P + (K + 1) \log K)$.

Finding the channel list is in general NP-hard [2]. We now describe a special case for which we give a polynomially complex algorithm.

III. FINDING THE P MOST PROBABLE SEQUENCES

Consider the cross-correlation matrix $\mathbf{R} = \mathbf{A}^t \mathbf{A}$ such that

$$R_{ij} = \begin{cases} 1 & i = j, \\ \rho & i \neq j, \end{cases} \tag{2}$$

for $1 \leq i, j \leq K$. This is what we mean by the symmetric channel. Such $\mathbf{R}$ occurs with use of different shifts of m-sequences by the users. It also occurs in certain multibeam narrowband satellite models.

A polynomial complexity optimal detection algorithm was presented in [3, 4] for $\mathbf{R}$ of the form (2). The optimal detection problem for this channel is

$$\hat{\mathbf{d}} = \arg \min_{\mathbf{d} \in \{-1,+1\}^K} \rho \sum_{i=1}^{K} \sum_{j=1}^{K} d_i d_j - 2 \sum_{i=1}^{K} d_i y_i$$

where $\mathbf{y} = \mathbf{A}^t \mathbf{r}$ is the matched filter output. The polynomial complexity algorithm depended upon the fact that the first term on the RHS depends only upon the number of negative elements in $\mathbf{d}$, denoted $n(\mathbf{d})$.

Part of the development in [3, 4] shows that the most probable $\mathbf{d}$ conditioned upon $n(\mathbf{d}) = 0, 1, \ldots, K$ can be found easily (optimal detection simply involves picking the best of these). For fixed $n(\mathbf{d})$ the most probable $\mathbf{d}$ is found as follows. Let $\mathbf{y}_\pi$ be $\mathbf{y}$ ordered such that the elements are non-decreasing. Let $\mathbf{d}_\pi$ be $\mathbf{d}$ in the same order. The optimum $\mathbf{d}_\pi$ is obtained by making the first $n(d)$ elements in $\mathbf{d}_\pi$ negative.

The key to finding channel list of size P is that beginning with the optimal $\mathbf{d}_\pi$ with $n = n(\mathbf{d})$ negatives, we may systematically obtain in order of decreasing probability, every other $\mathbf{d}_\pi$ with $n(\mathbf{d})$ by a series of swaps between the negative and positive elements in $\mathbf{d}_\pi$. This is Algorithm 1 in the Appendix. We can show that this process of swapping has complexity $O(n(K - n) + P \log P)$. This proof is based on the fact that for each new sequence found, there are at most two candidate sequences created, and sequences are selected only by choosing among these candidates.

Given that we can determine in order the sequences of highest probability with fixed $n(\mathbf{d})$, we now have the problem of determining the overall highest probability $\mathbf{d}_\pi$. This problem is easily solved by growing $K + 1$ subtrees from a root node corresponding to $n(\mathbf{d}) = 0, 1, \ldots, K$. Each subtree is grown according to Algorithm 1 (with $n(\mathbf{d})$-specific variables local to each sub-tree), with the modification that we only extend the best node from the entire tree at each stage.

IV. PERFORMANCE RESULTS

We now present performance results for the proposed system. The codes used were the maximal free distance rate 1/2, 4 state convolutional codes. We use information sequences of length 100, resulting in an interleaver size of 200.

Figure 3 ($K = 10$ users and $\rho = 0.6$), shows that after 4 iterations single user performance is almost achieved for all users using the proposed receiver with a list size of only $P = 80$ (compared to the full marginalization, which would require $P = 1024$).

Figure 4 shows the performance of the receiver versus P for $\rho = 0.6$ and $\rho = 0.7$. We see that the performance of the full-complexity ($P = 1024$) system is obtained with $P = 70$ when $\rho = 0.6$ and $P = 90$ when $\rho = 0.7$.

Figure 5 shows results for $K = 30$ and $\rho = 0.5$. There we see that if we take only $P = 3500$ out of the possible 2^{30} sequences, we obtain performance within 1 dB from single user performance at $E_b/N_0 = 6$ dB after 4 iterations. This (and other similar numerical results that we have obtained) indicates that the required P does not need to increase exponentially with K in order to obtain near single-user performance. If we decrease the value of ρ (i.e decrease the amount of multiuser inteference), the required list size P also decreases dramatically and single user performance is achieved at lower values of E_b/N_0.

V. DISCUSSION AND CONCLUSION

We have proposed a low complexity multiuser iterative decoder. It was shown that the marginalization of probabilities in (1) may be approximated closely with per-bit

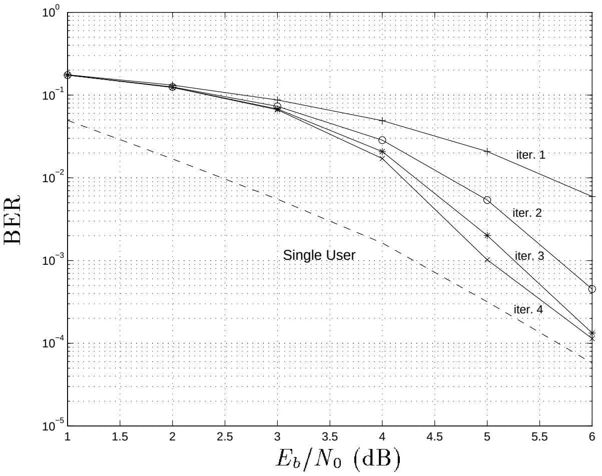

Figure 3: $K = 10, \rho = 0.6, P = 80$

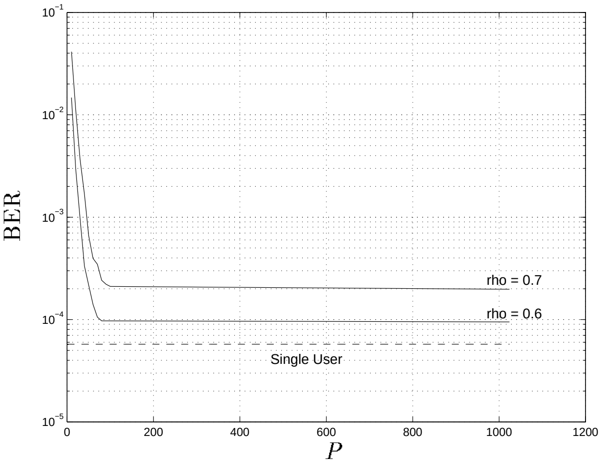

Figure 4: $K = 10, E_b/N_0 = 6$

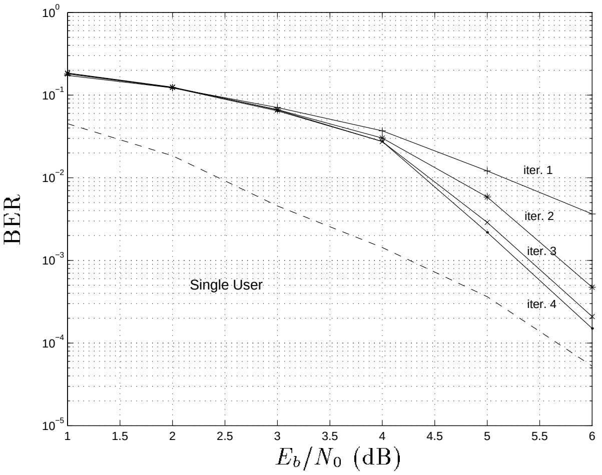

Figure 5: $K = 30, \rho = 0.5, P = 3500$

computational complexity $O(K+P+2K\log K)$. An integral part of the good performance of the system is due to Algorithm 1 which gives the P highest probable sequences when we have a symmteric channel as in (2). However a generic procedure is given in [5] which shows that if the number of computations to find an optimal solution to an integer optimization programming problem with n binary variables is $c(n)$, then the amount of computation required to obtain the P best solutions is $O(Pnc(n))$. Hence if we had a channel of the type in [6, 7] where all off-diagonal elements in the cross-correlation matrix R are negative we can find the P sequences with the highest channel probability (a method for finding the P best sequences for this type of cross-correlation matrix is outlined in [8] which can be used by applying the formulae defined in [6, 7]). We may then utilise the principle of merging the *prior list* and the *channel list* into one overall list to perform the marginalization (1). In generic terms, we can use the procedure in [5] to obtain a method to find the P most likely sequences with polynomial complexity for any system in which optimal detection is possible with polynomial detection.

VI. Acknowledgment

Thanks to C. Schlegel who contributed to early discussion of this work.

VII. Appendix

We need some definitions prior to stating Algorithm 1. For convenience, the algorithm shall operate using the metric

$$M(\mathbf{d}) = \mathbf{r}^*\mathbf{r} + \mathbf{d}^t\mathbf{R}\mathbf{d} - 2\mathbf{d}^t\mathbf{y} \tag{3}$$

rather than probabilities (which may be easily calculated based on the metrics later). Let $\mathbf{d}_{\text{opt}}(n)$ be the best sequence with $n = n(\mathbf{d})$ negatives (easily found using the algorithm in [3, 4]). Calculate the cost (change to (3)) of exchanging the signs of each pair of non-equal bits d_i, d_j in $\mathbf{d}_{\text{opt}}(n)$. Note that because we retain $n(\mathbf{d})$ and we have channel symmetry the cost is simply $m_{ij} = 4d_{\pi(i)}y_{\pi(i)} + 4d_{\pi(j)}y_{\pi(j)}$ where $1 \leq i \leq n$ and $n+1 \leq j \leq K$. Consecutively number these exchanges (swaps) in non-decreasing order to get an array $\mathbf{\Delta} = [\delta(1), \delta(2), \ldots, \delta(n(K-n))]$, where the i-th best swap $\delta(i)$ has members $\delta.m$ (cost of the swap) and $\delta.\mathsf{swapbits}$, a pair of numbers identifying which bits were swapped.

The algorithm will operate on a tree. Each node n of the tree (in addition to the child-parent relationships) has of the following members, $n.\mathbf{d}_\pi$, the permuted $\mathbf{d}$ sequence; $n.\mathsf{metric}$ according to (3); $n.\mathsf{swapbits}$, a list of bits which have been swapped from $\mathbf{d}_{\text{opt}}(n)$ to obtain $n.\mathbf{d}_\pi$; $n.\mathsf{largestswap}$, where $\delta(n.\mathsf{largestswap})$ is the most costly swap which has been performed to reach n; and $n.\mathsf{state}$ $\in \{\mathsf{ACTIVE}, \mathsf{POTENTIAL}, \mathsf{COMPLETE}\}$.

References

[1] D. Eppstein, "Finding the k shortest paths," Tech. Rep., Dep. of Info. and Comp. Sci., University of California, 1994.

[2] S. Verdú, *Multiuser Detection*, Cambridge University Press, 1998.

[3] C. Schlegel and A. J. Grant, "Polynomial complexity optimal detection of certain multiple access systems," *IEEE Trans. Inform. Theory*, vol. 46, no. 6, pp. 2246 –2248, Sep. 2000.

[4] A. B. Reid, A. J. Grant, and P. D. Alexander, "Direct proof of a polynomial complexity optimum detection algorithm," *IEE Electron. Lett.*, vol. 37, no. 19, pp. 1203–1205, Sep. 2001.

[5] E. L. Lawler, "A procedure for computing the k best solutions to discrete optimization problems and its application to the shortest path problem," *Management Science*, vol. 18, no. 7, pp. 401–405, Mar. 1972.

[6] S. Ulukus and R. Yates, "Optimum multiuser detection is tractable for synchronous CDMA systems using m-sequences," *IEEE Comm. Lett.*, vol. 2, no. 4, pp. 89–91, Apr. 1998.

[7] C. Sankaran and A. Ephremides, "Solving a class of optimum multiuser detection problems with polynomial complexity," *IEEE Trans. Inform. Theory*, vol. 44, no. 5, pp. 1958–1961, Sep. 1998.

[8] H. Hamacher, "An $O(K \cdot n^4)$ algorithm for finding the K best cuts in a network," *Operations Research Letters*, vol. 1, no. 5, pp. 186–189, Nov. 1982.

Algorithm 1 P best sequences for given $n = n(\mathbf{d})$

```
Create node $n$ with members $n.\mathbf{d}_\pi \leftarrow \mathbf{d}_{\text{opt}}(n)$, $n.\mathsf{metric} \leftarrow M(\mathbf{d}_{\text{opt}}(n))$, $n.\mathsf{state} \leftarrow$POTENTIAL, $n.\mathsf{swapbits} \leftarrow \emptyset$, $n.\mathsf{largestswap} \leftarrow 0$.
sequences_found $\leftarrow 0$
POTENTIAL_LIST $\leftarrow n$
PATH_LIST $\leftarrow \emptyset$
while sequences_found $\neq$num_sequences_desired do
  // Choose Best Potential
  if POTENTIAL_LIST $= \emptyset$ then
    All paths found for $n(d)$ negatives. Terminate Algorithm.
  end if
  $p \leftarrow$ node $p' \in$POTENTIAL_LIST with smallest $p'$.metric.
  Append $p$ to PATH_LIST
  Remove $p$ from POTENTIAL_LIST
  $p$.state$\leftarrow$ACTIVE
  sequences_found $\leftarrow$ sequences_found $+1$
  // Extend Active Nodes
  for all $n \in \{p, p.\mathsf{parent}\}$ such that $n$ does not have a POTENTIAL child do
    $s \leftarrow n.\mathsf{largestswap} + 1$
    FIND_s:
    if there is such an $s$ then
      $n.\mathsf{largestswap} \leftarrow s$
      if $\delta(s).\mathsf{swapbits} \cap n.\mathsf{swapbits} = \emptyset$ then
        $\mathbf{d} \leftarrow n.\mathbf{d}_\pi$ with the bits $\delta(s)$.swapbits swapped
        if there is no node $m \in$POTENTIAL_LIST with $m.\mathbf{d}_\pi = \mathbf{d}$ then
          Create child node $n'$ of $n$ with members
          $n'.\mathsf{metric} \leftarrow n.\mathsf{metric} + \delta(s).\mathsf{metric}$
          $n'.\mathbf{d}_\pi \leftarrow \mathbf{d}$
          $n'.\mathsf{swapbits} \leftarrow n.\mathsf{swapbits} \cup s.\mathsf{swapbits}$
          $n'.\mathsf{largestswap} \leftarrow s$
          $n'$.state$\leftarrow$POTENTIAL
          Append $n'$ to POTENTIAL_LIST
        else
          Increment s
          goto FIND_s:
        end if
      else
        Increment s
        goto FIND_s:
      end if
    end if
  end for
end while
```