A Greedy Iterative Algorithm and VLSI Implementation Strategy for Multiuser Detection

Geoff Knagge

Abstract—**Multiuser detection (MUD) strategies have the potential to significantly increase the capacity of wireless communications systems, but for these to be useful they must also be practical for implementation in VLSI circuits that cope with real world situations and process data in real time. The application of the optimal Maximum Likelihood (ML) solution is limited by its exponential computational complexity. Current near-ML algorithms such as the spherical detector offer significantly reduced complexity, but are still likely to be impractical for larger problems. Simpler MUD algorithms, such as the interference canceller (IC), are more practical but offer poor performance in some cases of interest.**

In this paper, we present a unique algorithm that shares aspects of these well known techniques, but offers higher performance than the simpler methods, while remaining practical. Our approach involves a series of iterations that that gradually converge to a decision, by choosing the estimates that produce lower values for our ML cost function. While this is more complex than well known methods, it offers higher bit error rate performance and is lower in complexity that other methods with comparable performance. We also show how to further reduce the complexity to make our algorithm suitable for VLSI implementation.

Index Terms— **Multiuser Detection, VLSI, Iterative Optimisation, Greedy, ASIC Implementation.**

I. Problem Background

The problem of multiuser detection (MUD) for Code Division Multiple Access (CDMA) system is important due to its potential to significantly increase the performance and capacity of these systems. The optimality of the maximum likelihood (ML) multiuser detector was proven by Verdú [1], and offers significantly improved bit error rate (BER) performance over conventional detection techniques. While Verdú's work provides a bound on achievable BER, the ML algorithm is a combinatorial optimisation problem that is NP-hard, and hence is impractical for all but the smallest of problems. With M users or transmitters, each transmitting from a constellation of size 2^k, the complexity of the problem becomes $O\left(2^{kM}\right)$. As the dimensions of the detection problems increase, the complexity of this approach becomes prohibitive.

Interference cancellation (IC) methods are well studied alternatives that involve estimating the multiple access interference (MAI) between users and subtracting it from the received signal. These use multiple stages to progressively remove interference and improve the estimate. If this is done correctly, the multiple access interference (MAI) seen on each user can be greatly reduced so that the original transmission can be better determined [2], [3], [4]. While they are lower in complexity, we will show that their BER performance remains well below that of ML.

Other multistage iterative algorithms include those that combine multiuser detection with advanced error correction techniques such as turbo codes, as described in [5]. Such techniques have been shown to perform very well. However, in cases where we do not want to use turbo coding, or where we cannot afford additional latency and complexity in the turbo loop, these may be unsuitable.

Lattice decoding [6], and the closely related "Spherical decoding" [7], are regarded as very promising candidates for high performance, near-ML algorithms. It has recently received attention from researchers for its potential application to space-time decoding, and multiuser detection (MUD) for MC-CDMA [8] and DS-CDMA [9] systems. However, as the size of the detection problem grows, the size of the channel matrix also increases, and the complexity of the preprocessing requirements and size of the search tree makes VLSI practicality questionable.

This paper proposes an iterative algorithm that attempts to approach an ML result while reducing complexity. The mathematical model for our problem is described in section II, and we introduce our new strategy in section III. Section IV discusses the comparative performance of our approach to other methods, and the effects of tuning its various parameters. Section V addresses the implementation aspects of building our proposed detector into a VLSI design, and Section VI discusses additional issues that need to be considered in a real world implementation. We then conclude with a summary and evaluation of our proposal.

II. Mathematical Model

A single DS-CDMA user, k, transmits a data bit s of value ± 1, which is spread by a channelisation code, $c(t)$ of length j. In addition, as the signal is transmitted across an air interface, it is also subject to additive white Gaussian noise (AWGN), $n(t)$. If we define each time interval to be equal to one chip, a discrete-time model of the received signal is then represented by

$$\mathbf{y}_k = \mathbf{c}_k s + \mathbf{n}, \tag{1}$$

where $\mathbf{y}_k$, $\mathbf{c}_k$ and $\mathbf{n}$ are jx1 vectors.

In a multiuser system such as UMTS, user transmissions are additionally scrambled with a QPSK Gold sequence, a 1xj vector $\mathbf{G}_k$, which can be combined with the spreading code to form a simplified channel model, $\mathbf{H}_k$:

The author is with the School of Electrical Engineering and Computer Science, The University of Newcastle, Callaghan NSW 2308, Australia, e-mail:gknagge@ee.newcastle.edu.au . This work was supported by Bell Laboratories Australia, Lucent Technologies, with the Australian Research Council under Linkage Grant LP0211210.

Thanks are extended to Graeme Woodward of Agere Systems (and formerly Bell Labs Australia), and Steve Weller of the University of Newcastle, for guidance during the progress of this work.

$$\mathbf{H}_k = \left(\mathbf{C}_k \mathbf{G}_k^T\right)^T . \tag{2}$$

Using this technique, M users may share a communications channel by using different scrambling codes and may be modelled by simply adding their signal contribution to the channel. If all users were controlled so that they all transmitted synchronous symbols, then the discrete time model becomes

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n}, \tag{3}$$

where $\mathbf{y}$ and $\mathbf{s}$ are jx1 vectors, and $\mathbf{H}$ is a jxM matrix of channel estimates.

The multiuser detection problem is to solve (3) for $\mathbf{s}$, with knowledge of the channel $\mathbf{H}$ and the received signal $\mathbf{y}$. To solve optimally, this is a combinatorial optimisation problem which results in a solution $\hat{s}$:

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}\in\Lambda} \|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2, \tag{4}$$

where Λ is the set of possible decisions over all users.

We will initially consider a method to solve this simplified model with hard decisions, but later introduce solutions for complications that must be considered for real systems. These include asynchronous transmission of symbols, multipath interference, and the generation of soft information.

III. Iterative Greedy Algorithm

Our approach uses a combination of features from the standard approaches to derive an alternative method to achieve high performance with low complexity. It is an iterative greedy-style algorithm that aims to converge in such a way as to minimise the cost function

$$\|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2.$$

Greedy algorithms are well known in computer science literature due to their simplicity while obtaining good, and in some cases, optimal, results [10]. Here, our use of the greedy principle simply refers to the act of, on each iteration, taking the decision that provides the best immediate solution out of the current choices.

The detection process starts with an initial estimate of $\mathbf{s}$, which may either be zero, or an intelligent guess such as a matched filter output. On each iteration, each user is individually processed and we consider incrementing or decrementing its guess by a small amount and calculating the new cost function for each alternative. The better option is then selected for the next iteration, with the intention of progressively improving the estimate of what was transmitted.

When all iterations are complete, we will obtain an estimate for each user of a value that is some multiple of the step size. A hard decision can easily be generated by taking the sign of these estimates, to produce an output that is comparable with other detection methods. The magnitude of these final estimates provide an indication to the degree of confidence of each decision, although they do not represent true soft information as is discussed in section VI-C.

This method represents a combination of the following standard detection types:

- The cost function is the same at that used in ML and sub-optimal ML detection
- The iterative approach represents a form of interference cancellation
- Decisions are made in a manner similar to greedy and standard matched filter detectors.

We consider two alternatives to how this may be implemented, which is either a serial or parallel approach. Each method has the same basic parameters that need to be tuned, including the number of iterations to perform, the size of the adjustment step to use on each iteration, and the method of determining the initial estimate.

A. Parallel Iterative Greedy (PIG) Detector

The parallel approach, outlined in Alg. III.1, allows all users to be considered at the same time, using the same knowledge of the state of the current guess. On subsequent iterations, all user's calculations are based on the new estimate calculated for all users in the previous iteration.

Algorithm III.1 Parallel Iterative Greedy Detector

```
iteration = 0
complete=false
step = 0.1
N = number of channels to detect
channelMat = Channel Matrix
estimate = vector of zeros, length N
while not complete {
    iteration = iteration + 1
    newguess = vector of zeros, length N
    for cnt=1 to N
        option1 = estimate
        option2 = estimate
        option1(cnt) = estimate(cnt) + step
        option2(cnt) = estimate(cnt) - step
        cost1 = $|y - \mathrm{channelMat} * \mathrm{option1}|^2$
        cost2 = $|y - \mathrm{channelMat} * \mathrm{option2}|^2$
        if cost1<cost2 {
            newguess(cnt)=option1(cnt)
        } else {
            newguess(cnt)=option2(cnt)
        }
    }
    **estimate = newguess**
    if (iteration>maxIterations) {
        complete = true
    }
}
```

B. Serial Iterative Greedy (SIG) Detector

The only difference between the serial and parallel approach is that the serial approach requires the current estimate to be updated after each individual user is adjusted, and so users must be processed sequentially. This is outlined in Alg. III.2.

Algorithm III.2 Serial Iterative Greedy Detector

```
iteration = 0
complete=false
step = 0.1
N = number of channels to detect
channelMat = Channel Matrix.
estimate = vector of zeros, length N
while not complete {
    iteration = iteration + 1
    newguess = vector of zeros, length N
    for cnt=1 to N
        option1 = estimate
        option2 = estimate
        option1(cnt) = estimate(cnt) + step
        option2(cnt) = estimate(cnt) - step
        cost1 = $|y - \mathrm{channelMat} * \mathrm{option1}|^2$
        cost2 = $|y - \mathrm{channelMat} * \mathrm{option2}|^2$
        if cost1<cost2 {
            newguess(cnt)=option1(cnt)
        } else {
            newguess(cnt)=option2(cnt)
        }
        estimate = newguess
    }
    if (iteration>maxIterations) {
        complete = true
    }
}
```

Successive iterative algorithms are often sensitive to the order in which users are processed, such as the case for the successive interference canceller (SIC). Simulations have shown that this is not the case for the SIG when small step sizes, of 0.125 or lower, are used. Therefore, no special ordering has been used for all results for the SIG presented in this paper.

IV. Performance

To evaluate the relative performance of the PIG and SIG detection methods, a CDMA model was simulated for synchronous users with a spreading factor of 16, and spread and scrambled, as described in our channel model. For these experiments, we used 15 iterations with a step size of 0.1, and made a hard decision based on the sign of the final estimates. The successive interference canceller (SIC) that we used for comparison was configured to provide good performance, with 10 stages where 10% of the estimated signal was cancelled for each user on the first stage, and that amount was increased by 10% on each stage. Standard ML, matched filter, and Linear MMSE detectors were also implemented as described in [1] and [11].

Fig. 1 shows the BER of PIG and SIG algorithms in an eight user system, compared with other standard MUD techniques. We can see from this that our proposed scheme is capable of outperforming the more complex linear MMSE technique. The simpler SIC suffers from BER flooring as the signal to noise ratio increases.

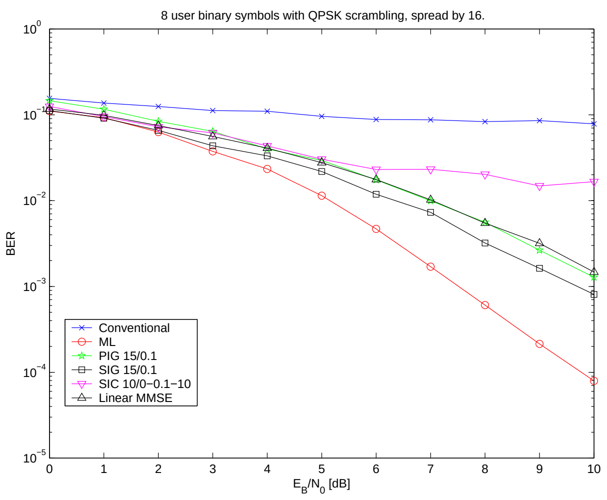


Fig. 1. Comparison of the iterative technique to other techniques.

The trend of the PIG detector's BER, as the number of users is varied, is given in Fig. 2. As is the case for linear MMSE and other MUD algorithms, simulation shows that the performance of our algorithm is degraded as the channel saturates.

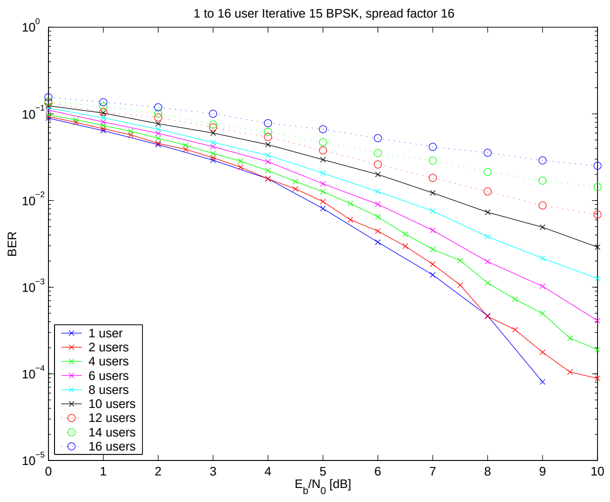


Fig. 2. The performace of the PIG technique on various loadings of a BPSK system with a spreading factor of 16.

A. Effect of number of Iterations on performance

It is interesting to track to BER of iterative detectors as the number of stages is varied, and this is done for the SIG algorithm in Fig. 3. We see that, for a step size of 0.1, the algorithm reaches its best BER at around the twelfth stage. For further stages, the BER actually worsens slightly before stabilising

B. Effect of step size on performance

The other parameter that can affect performance is the size of the step by which values will be adjusted on each iteration. Fig. 4 indicates that decreasing the step below 0.125 does not seem to make any distinguishable difference to the BER curves. If it

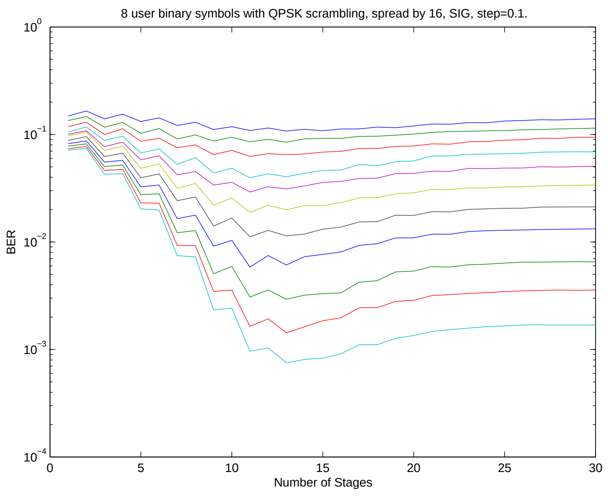


Fig. 3. The performance of the SIG detector over a variable number of iterations. The lines represents a range of $E_b/N_0 = 0dB$ (top) to $E_b/N_0 = 10dB$ (bottom) in steps of 1dB.

is set too high then the performance is degraded significantly. To allow for a VLSI friendly implementation, we also require a step size that is a power of two.

For larger step sizes in the SIG algorithm, the ordering of users becomes important, since an incorrect guess for the first user can impair the ability to correctly detect other users.

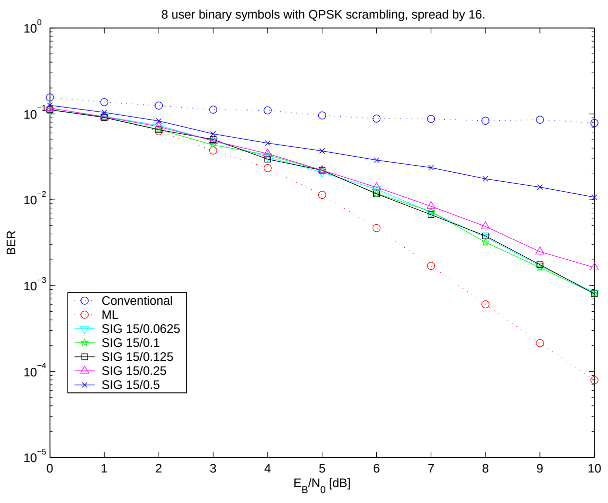


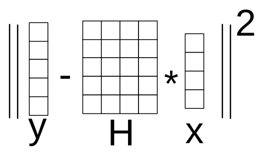
Fig. 4. The effect of varying step size on performance.

V. Complexity Analysis

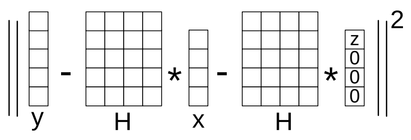
The PIG and SIG algorithms are more complex than iterative cancellers, but offer improved performance and are not too complex to be feasible for VLSI implementation. However, compared to other high performance algorithms such as linear MMSE and spherical detection, the level of complexity is much simpler. Linear MMSE requires VLSI expensive matrix inversions, and the spherical algorithm requires a large amount of calculation in the preprocessing and searching stages. For systems with continually changing scrambling sequences, such preprocessing requirements severely limit the usefulness of these methods.

The advantage of this iterative approach is that its complexity is fixed and easily adjusted, at the expense of incurring a possible performance penalty. The algorithm, as presented, may initially seem computationally expensive due to the number of repeated cost function calculations required, but this may be greatly reduced through careful selection of the step size.

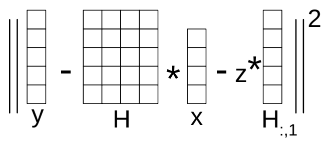
Assume we have a received signal, $\mathbf{y}$, a channel matrix $\mathbf{H}$ and an existing estimate $\mathbf{x}$. For a four user system, spread by a five chips, the cost calculation is represented by :



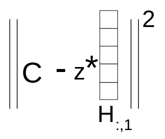
If the first user is adjusted by a step of size z, the new cost becomes :



This can be simplied to :



By representing the existing cost components as C, the simplifed new cost can be expressed with :



$$\|\mathbf{C} - z\mathbf{H}_{:,1}\|^2 = |C_1 - zH_{1,1}|^2 + \ldots + |C_5 - zH_{5,1}|^2 \quad (5)$$

In general, each of the terms $|C_n - zH_{n,1}|^2$ requires one scalar multiplication, but this is where simplifications may be made. For scalar quantities, we observe that

$$(a - b)^2 = a^2 - 2ab + b^2. \quad (6)$$

In our case, $a = C_n$, $b = zH_{n,1}$, and we assume that we already know a^2 from a previous cost calculation. In hardware, multiplications by powers of 2 are achievable at insignificant cost, and so the simplification is found by assigning $z = 0.125$, or to another power of 2. If $\mathbf{H}$ contains entries of the form ± 1 or $\pm j$, then calculating $2ab$ simply involves a shifting operation, and b^2 is always constant. Therefore, each component of the new cost may be determined from the components of the existing cost without the need for any explicit multiplications.

The assumption that $\mathbf{H}$ contains entries of the form ± 1 or $\pm j$ may initially seem highly restrictive, since it assumes that all chips in the sequence are of the same amplitude. This appears to eliminate the possibility of directly including multipath properties in the channel matrix, but we will address this in section VI-A by using a Rake approach. It also does not impact upon the effects of having differences between the amplitude of different user's signals, since this is compensated by the gradual stepping of, and different magnitudes of, user estimates. In fact, this detector does not require any knowledge of the received power of any user, which is a desirable quality because detectors that do rely on such information can be sensitive to errors in that data.

An additional complexity of the parallel version of this detector is that we need to determine the components of the existing cost value in order for the described multiplication-free decisions to occur. Since all users are performing the above operations in parallel and changing together, the above technique merely yields an estimation of the effect that a particular decision will have on the overall cost. Hence, a full cost calculation is necessary at the beginning of each iteration.

The serial version has very similar complexity to that of the parallel alternative, with one significant deviation. Since only one user decision is changed at a time, the multiplication-free calculation process will in fact generate an accurate calculation of the new overall cost of making a particular decision. In contrast, the PIG acts on each user as if it was the only user being adjusted, and so the actual cost, after all of the parallel adjustments are made, will be different to the estimates used to make those adjustments. Therefore, the PIG requires an explicit calculation of the overall cost after the completion of each stage. This is not required for the SIG after the algorithm is initialised, since it can be derived from the change in the estimate for the last user that was processed.

VI. Further Modifications

While many algorithms are theoretically promising, to be of real use they must also be able to be adapted to deal with practical difficulties that are presented by real-world systems. In particular, independent wireless transmitters can almost never be coordinated to generate synchronous transmissions. In addition, their signals will also travel over many paths, with some reflections arriving several chips later than the first.

For an algorithm to be useful in modern systems, it must also be able to provide a measure of the confidence of each decision so that high performance error correction algorithms, such as turbo decoders, may be most effectively used.

A. Multipath Interference

The problem of handling multipath interference is one that complicates the implementation of this algorithm. If the timing and strength of the various paths are known, then this information may normally be represented by applying appropriate modifications to the spreading code for each particular user.

However, for VLSI friendly implementation, we require a unit amplitude for each chip in the detector's representation of this code. A work-around for this problem is to use a Rake approach, where we separately consider each of the strongest paths as a separate user, and combine the results to reach a final decision.

B. Asynchronous and multi-rate Channels

A major problem that increases the practical complexity of most MUD algorithms is that real systems are not synchronous. It is virtually impossible to coordinate mobile devices to transmit synchronously at the data rates required, and any synchronisation that was obtained could be quite easily lost due to multipath effects.

In addition, many CDMA systems, such as those used in the 3G mobile standards, are multirate systems. This means that in the time interval needed for one user to transmit a single symbol, another user may have transmitted up to 64 symbols. Fig. 5 gives an illustration of a possible 9 user system, with the different significant combinations of user arrangements:

- Users 1 and 2 are synchronous
- Users 3 to 9 are asynchronous and of possibly different data rates
- Users 1 to 5 begin transmission of a symbol at time interval 1
- Users 3 and 7 end and begin transmission of a symbol at time interval 16
- Users 1,2, and 7 end transmission of a symbol at time interval 25

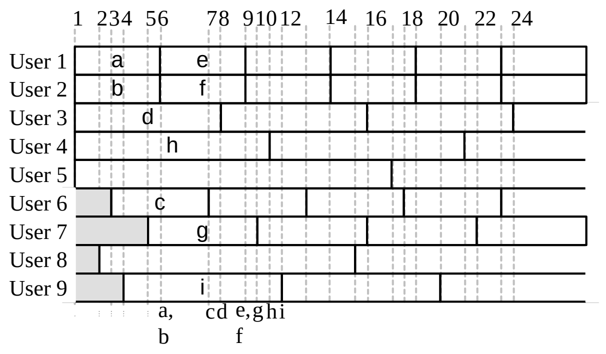


Fig. 5. A possible confi guration of asynchronous user symbols. The top scale shows time intervals where an entire symbol moves into the detection window, and the bottom scale indicates the symbols that end on each of these intervals.

The problem that this causes is that any decision on a given symbol may also affect two or more additional symbols from every other user on the system, and those may cause the decision to indirectly affect the decision made on the data that those additional symbols overlap. Since the detection must be done in real-time, it is necessary to perform any detection on "windows" of data at a time.

A sliding window could be used, with a small number of iterations occurring on each chip interval to progressively determine an estimate for each symbol. Once an entire symbol has moved into the detection window, as is the case at the numbered instants in Fig. 5, then the decision for that symbol may be finalised and its contribution cancelled from the received signal. This would mean that all preceding symbols of other users would already be known, so for this example we only need to deal with 8 other unknowns at any one time. For example, the shaded regions in Fig. 5 represent the already detected symbols at time interval 6. The size of the window would only need to be large enough to contain the longest possible symbol. It is possible to achieve this whilst maintaining the VLSI-friendly nature of the SIG implementation.

C. Soft Information

Normally, we require that a detector not only provides an estimate of what was transmitted, but also calculates a measure of the confidence in that decision. Generally, we require a likelihood ratio of probabilities:

$$\mathrm{LR}(\mathbf{y}) = \frac{P(-1|\mathbf{y})}{P(1|\mathbf{y})}. \tag{7}$$

The soft outputs can then be determined by

$$\begin{aligned} \mathrm{LLR}_k &= \ln\frac{P(s_k = 1|y)}{P(s_k = -1|y)} &(8)\\ &= \ln P(s_k = 1|y) - \ln P(s_k = -1|y) &(9) \end{aligned}$$

For a system containing AWGN, and where the cost of a particular set of decisions, $\mathbf{s}$, is $d_{\mathbf{s}}^2 = \|\mathbf{y} - \mathbf{Hs}\|^2$,

$$P(\mathbf{s}|\mathbf{y}) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{d_{\mathbf{s}}^2}{2\sigma^2}}. \tag{10}$$

The cost of one of the two possible decisions for each user is already known from the detection process, and so we need to explicitly calculate the effect of inverting that decision for each user. The magnitude of the detectors output may be viewed as either the signal strength of the user, the level of confidence in that decision, or some combination of both.

VII. Conclusions

Traditional MUD algorithms are either limited in their VLSI feasibility due to high computational complexity, or lacking in performance when compared with the optimal ML technique. Newer techniques, such as the spherical search, offer a medium between these but their practicality for large systems is still questionable.

We have proposed an iterative technique that offers relatively good performance, and proposed a VLSI implementation strategy that takes advantage of selectively chosen parameters to improve its practicality. We have also offered simple approaches to dealing with real world issues, such as multipath and asynchronous transmissions.

Our algorithm is more complex than traditional interference canceller approaches, but offers improved performance. In comparison to other detectors with similar performance, such as the linear MMSE, this algorithm is simpler in complexity.

References

[1] S. Verd´u, *Multiuser Detection*. Cambridge, 1st ed., 1998.

[2] A. Johansson, *Successive Interference Cancellation in DS-CDMA Systems*. PhD thesis, Chalmers University of Technology, School of Electrical and Computer Engineering, Oct 1998.

[3] D. Guo, L. K. Rasmussen, S. Sun, and T. J. Lim, "A matrix-algebraic approach to linear parallel interference cancellation in CDMA," *IEEE Trans. Communications*, vol. 48, pp. 152–161, Jan 2000.

[4] L. K. Rasmussen, D. Guo, and T. J. Lim, "Aspects on linear parallel interference cancellation in CDMA," in *IEEE International Symposium on Information Theory*, p. 37, 1998.

[5] M. Reed, C. B. Schlegel, P. D. Alexander, and J. A. Asenstorfer, "Iterative multiuser detection for CDMA: Near single user performance," *IEEE Transactions on Communications*, vol. 46, Dec 1998.

[6] E. Agrell, T. Eriksson, A. Vardy, and K. Zager, "Closest point search in lattices," *IEEE Transactions on Information Theory*, vol. 48, pp. 2201–2213, Aug 2002.

[7] B. Hochwald and S. ten Brink, "Achieving near-capacity on a multiple-antenna channel," *IEEE Trans. Information Theory*, vol. 51, pp. 389–399, Mar 2003.

[8] L. Brunel, "Optimum and sub-optimum multiuser detection based on sphere decoding for multi-carrier code division multiple access systems," in *IEEE ICC, vol. 3*, pp. 1526–1530, 2002.

[9] L. Brunel and J. Boutros, "Euclidean space lattice decoding for joint detection in CDMA systems," in *IEEE ITW*, pp. 20–25, June 1999.

[10] G. Brassard and P. Bratley, *Fundamentals of Algorithmics*. Prentice Hall, 1st ed., 1996.

[11] S. Moshavi, "Multi-user detection for DS-CDMA communications," *IEEE Communications Magazine*, vol. 34, pp. 124–136, Oct 1996.