Reduced Complexity Software Receivers for TD-SCDMA Downlink

Sanyogita Shamsunder and John Glossner
Sandbridge Technologies
White Plains, NY 10601
Email: {sshamsunder, glossner}@sandbridgetech.com

***Abstract*— Evolving 3G standards such as TD-SCDMA use multi-user detection (MUD) at the base station to enhance the link bugdet in the uplink. Several sub-optimal block-based techniques have been proposed for detecting the user signals in multipath. These methods have been prohibitively expensive for implementation at the mobile. In this paper, we examine some of the most promising algorithms and consider their suitability for implementation in software on the Sandblaster platform. It is shown that joint detection at the mobile is well within the capabilities of the Sandbridge processor.**

I. Introduction

TD-SCDMA belongs to the family of 3G wireless standards as defined by the 3GPP and will be deployed by many carriers in their TDD mode. The 3GPP has adopted 2 different chip rates for this mode, high (3.84 Mcps) and low (1.28 Mcps). The former is compatible with WCDMA FDD. Both versions of TD-SCDMA include sophisticated physical layer techniques such as smart antennas and joint detection to increase system capacity even with inter-chip and multi-user interference.

Joint or multi-user detection at the baseband tremendously improves performance over a conventional Rake receiver. The latter deteriorates in performance when multipath reduces the orthogonality of the spreading codes. Multi-user detection techniques for the uplink have been advocated even during the TD-SCDMA standardization process, e.g., [2] and [3]. However, the complexity of these detection techniques has prevented their widespread use especially on the downlink at the mobile. Because of the slotted TDMA scheme with short PN sequences and a small number of simultaneous users, and the availibility of powerful baseband processors, there is a possibility for using MUD even in the downlink.

II. TD-SCDMA System and Signal Model

The TD-SCDMA system as defined in the 3GPP standard combines aspects of TDMA with CDMA. It allows for the existence of multiple users in a given uplink or downlink time-slot. The structure of a time-slot is shown in Figure 1. In the low chip rate mode, TD-SCDMA defines 7 slots (5 ms) with the first slot reserved for downlink and multiple switching points for reversing the direction of transmission. Each user data vector is spread and the composite downlink data vector is scrambled using a scrambling code of length 16. A downlink burst consists of two data blocks with a user-specific mid-amble that is used for various tasks by the receiver. There is also a guard interval between neighbouring bursts. Based on the supported data rates, the modulation can be either QPSK, 8PSK or 16QAM.

Let K be the number of synchronous downlink codes in a time-slot. A user can have multiple codes assigned to it. Let N be the number of symbols/block and $1 \leq Q \leq 16$ be the spreading factor. It is assumed that there is a single receive antenna at the mobile. Let $\mathbf{c}_k$, $d_k(i)$ and Q_k be the spreading code, data symbols and the spreading factor for the $k^{th}-$user. The discrete-time signal transmitted by the base station in TD-SCDMA is

$$y(n) = z(n) \sum_{k=1}^{K} [\sum_{i} d_k(i) c_k(n - iQ_k)], \tag{1}$$

where $z(n)$ the base-station specific scrambling code will be dropped in the remainder of the paper as it is not relevant to the algorithm comparison. Let $\mathbf{h}$ be the downlink channel impulse response with duration W chips. The filtered and sampled signal at the receiver is:

$$\mathbf{x} = \mathbf{A}\mathbf{d} + \mathbf{v}. \tag{2}$$

The vector $\mathbf{x}$ denotes the received chips (1 sample/chip) in a slot and the columns of $(NQ + W - 1) \times NK$ sized matrix $\mathbf{A}$ are the convolution of $\mathbf{h}$ and $\mathbf{c}_k$, see e.g., [4]. The block-banded $\mathbf{A}$ is as shown in Figure 2. The noise vector $\mathbf{v}$ is approximately additive, white Gaussian (AWGN) and includes interference. In a typical TD-SCDMA deployment, for the downlink $K \leq 8$, $Q \in \{1, 2, 4, 8, 16\}$, $2N = 704/Q$ and $W \approx 20$.

Without loss of generaility, let $d_1(n)$ be the desired user symbols. A receiver matched to this user treats the interfering user signals as additive noise and extracts the desired signal based on the user-specific spreading or channelization code. Therefore, a Rake receiver, which is based on the matched-filter bank concept, is susceptible to near-far effects and deteriorates in performance when stronger undesired users are present. However, due to its simplicity, this is the method of choice in most downlink receivers for cellular CDMA.

[0]The channel here includes the transmit and recive filters as well

On the other hand, a joint-detector exploits the structure of the interfering signals to jointly estimate all the user symbols $\mathbf{d}$. MUDs are near-far resistant but are not practical for most of the cellular CDMA receivers. The time-slotted nature of TD-SCDMA enables the use of MUDs at the base-station side. Several sub-optimal and computationally efficient techniques for inverting the system matrix $\mathbf{A}$ in (2) have been proposed ([2], [3] and references therein). The resulting algorithms are amenable to hardware/software implementation at the base-station where complexity and power constraints are not critical; they are still considered complex for use at the hand-set. In addition, most commercial applications usually employ dedicated hardware for such complex receiver blocks.

Traditional communications systems have typically been implemented using custom hardware solutions. Chip rate, symbol rate, and bit rate co-processors are often coordinated by programmable DSPs but the DSP processor does not typically participate in computationally intensive tasks. As it typically happens in a modern day receiver, when multiple systems requirements are considered, both silicon area and design validation are major inhibitors to commercial success. A software-based platform capable of dynamically reconfiguring communications systems enables elegant reuse of silicon area and dramatically reduces time to market through software modifications instead of time consuming hardware redesigns. SDR solutions based on the SandBlaster platform have been proposed for WCDMA, 802.11 and other wireless baseband receivers [6] and [8].

In this paper, we will examine three different reduced complexity techniques that have been proposed for joint detection. We will consider their suitability for implemention on a fixed-point, multi-threaded Sandblaster platform [7]. The algorithms are briefly described in the next Section and the implementation issues are discussed in Section IV.

III. Multi-user Detection Algorithms

To solve for $\mathbf{d}$ in (2), we need the system matrix $\mathbf{A}$ which in turn can be easily computed based on estimates of the channel coefficients $\mathbf{h}$ and spreading codes. In this paper, we assume that he channel coefficients are known. The Maximum Likelihood solution for (2) involves searching for $\mathbf{d}$ over a multi-dimensional space and is thus impractical. The sub-optimal least-squares estimate of the NK data symbols is given by:

$$\hat{\mathbf{d}} = \mathbf{T}^{-1}\mathbf{y}, \quad \text{where,} \quad \mathbf{T} = (\mathbf{A}^H\mathbf{A}), \mathbf{y} = \mathbf{A}^H\mathbf{x}. \tag{3}$$

There are a number of approaches for implementing joint detection in a CDMA system. The zero-forcing (ZF) equalizer or decorrelating detector applies the inverse of the system matrix to separate the user signals and eliminate multi-access interference (MAI). This scheme is very popular and was considered, for the early TD-SCDMA demos [1]. A Minimum-Mean Square Error (MMSE) detector minimizes the error between the weighted received signal and the desired bits and can result in lower BER at high SNR levels. The computational costs of the ZF equalizer are smaller than that for the MMSE detector because the latter requires an estimate of the noise covariance matrix; however, the implementation issues are very similar. This paper only addresses the zero-forcing detector.

A. Complexity of Cholesky Decomposition

In theory, the pseudo-inverse of $\mathbf{A}$ can be computed via the Singular Value Decomposition (SVD). However, due to its complexity, SVD is not a practical approach. For Hermitian matrices, the Cholesky factorization which results in a lower triangular matrix requires fewer computations. The $NK \times NK$ matrix $\mathbf{T} = \mathbf{A}^H\mathbf{A}$ in (3), is Hermitian. The matrices $\mathbf{A}$ and $\mathbf{T}$ are block diagonal with $2\nu - 1$, $\nu = \lceil (Q + W - 1)/Q \rceil - 1$ non-zero diagonal blocks. Direct computation of the Cholesky factors of $\mathbf{T}$ requires $\mathcal{O}(N^3K^3)$ operations [10], which results in exponential complexity for a typical TD-SCDMA handset. Let $\mathbf{R}$ denote the Cholesky factor of $\mathbf{T}$. The computations can be reduced by a factor of N^2 by exploiting the block-banded property of $\mathbf{T}$. Also, as $N \to \infty$, the Cholesky factor $\mathbf{R}$ is approximately block Toeplitz for $N >> \nu$ [9]. Only the first few block rows $a_r \geq \nu$ of $\mathbf{R}$ need to be computed; the remaining blocks are simply copies of the last computed block. This approximation error is acceptable as long as $N >> \nu$ and $a_r \geq \nu$. Additional details on complexity reduction for this approach can be found in [2] and [5].

B. Complexity of Schur Decomposition

Since $\mathbf{T}$ is sparse and block Toeplitz, its Schur decomposition is an efficient way to Cholesky factorization. By working with a low redundancy representation based on the generators of $\mathbf{T}$ rather than $\mathbf{T}$, leads to a more efficient algorithm. Generator computation involves multiplications, square-roots and reciprocals, for each of the NK elements of the generator vectors $\alpha_i, 1 \leq i \leq K$. The algorithm proceeds by computing the lower triangular the Cholesky factor of $\mathbf{G}$, e.g., [11] and [4]. The resulting factor is also the Cholesky factor of $\mathbf{T}$.

For example, if Given rotations is used to reduce $\mathbf{G}$, then as elements gets zero-ed out and rows are eliminated, $\mathbf{G}$ shrinks progressively as the lower triangular matrix $\mathbf{R}$ is built. The algorithm may be terminated as soon as $a_r \geq \nu$ block rows are computed. The algorithm as applied to TD-SCDMA uplink is described in detail in [4].

C. Complexity of Block Fourier Algorithm

Another approach to solving the least-squares solution (2) is uses the fact that $\mathbf{A}$ is approximately block circulant. The eigen vectors of circulant matrices are the columns of

Discrete Fourier Transform (DFT) matrix [10]. Thus, systems of equations can be solved via a diagonalization using the Fourier Transform. In [3] a frequency domain approach was suggested for performing joint detection. The block Toeplitz matrix $\mathbf{T}$ is made block circulant (and thus square) by padding it with rows and columns. This approach to joint detection requires multiple DFTs, reciprocal and square-root operations. As NK increases, the complexity can be managed by applying the FFT to smaller overlapping blocks of data. As long as the block size is greater than $N+\nu-1$, this approximation leads to acceptable error. It was shown in [3], that the block Fourier approach requires fewer real multiplications than either the approximate Cholesky or block Schur techniques.

IV. Software-Defined Radio Implementation

The increasing need for the support of multiple wireless standards and eternally evolving standards has led to the adoption of software-defined radios in mobile terminals. Further, DSPs are increasingly powerful providing billions of operations per second and with power efficiency levels that are appropriate for handset deployment.

Sandbridge Technologies has designed a multi-threaded processor capable of executing DSP, Control, and Java code in a single compound instruction set optimized for handset radio applications. The Sandbridge design overcomes the deficiencies of previous approaches by providing substantial parallelism and throughput for high-performance DSP applications while maintaining fast interrupt response, high-level language programmability, and very low power dissipation.

As shown in Figure 9, the design includes a unique combination of modern techniques such as a SIMD Vector/DSP unit, a parallel reduction unit, and RISC-based integer unit. Instruction space is conserved through the use of compounded instructions that are grouped into packets for execution. The resulting combination provides for efficient Control Code, DSP, and Java processing execution.

The Sandblaster$^{\mathrm{TM}}$ platform consists of the fixed- point Sandblaster multi-threaded DSP processor, see Figure 9, which does the base band processing. The software tool chain is primarily dedicated towards generating and simulating efficient code for this processor. The Sandbridge compiler analyzes the C code, automatically extracts the DSP operations and synthesizes optimized DSP code without the excess operations required to specify DSP arithmetic in C code. This technique has a significant software productivity gain over intrinsic functions.

The Sandbridge vectorizing compiler is efficient at extracting this parallelism using Vectorizing optimizations. The Sandbridge compiler also handles the difficult problem of outer loop vectorization which is often a requirement for inner loop optimizations.

The common steps in all the three algorithms are the matrix multiplication involving the block diagonal $\mathbf{A}$ required to generate $\mathbf{T}$ and $\mathbf{y}$. These operations can be easily vectorized and implemented in parallel on the multi-threaded platform. For the Schur decompsition, the generator matrix computation and some steps in the Given rotations involve matrix multiplies; these are again implemented efficiently as vector operations on the Sandbridge platform.

The Fourier domain technique operates on chunks of received samples, thus it lends itself well to low latency applications. The block Fourier algorithm can be split up so that multiple threads implement the FFT on the data blocks. Note also that the FFT block size is dictated by the number of data symbols in a TD-SCDMA burst. Small block sizes also lead to larger implementation overheads, while larger block sizes lead to greater FFT complexity and latency. The current FFT implementation requires $N\log_2(N)$ MACs for the typical block sizes encountered in TD-SCDMA. Finally, all the algorithms involve several scalar inversions (reciprocals) and square-root operations which are implemented via iterative techniques requiring several cycles.

Figures 3- 8 compare the estimated processing power required for a software implementation of the joint detector on the Sandblaster platform. Other implementation overheads such as those due to synchronization, memory access are not included here. The techniques used by Vollmer *et al*, [3] are employed to estimate the number of operations in each case. In all cases, the block size for the Fourier algorithm is constant at 32. Reference [3] compared the algorithms taking into account only the number of real multiplications, while here other compute intensive operations such as reciprocals and square-roots are also taking into account. It was also shown that the Fourier-based approach required the least number of multiplications. However, as is evident from the results here, the gains in a practical implementation are smaller (or even non-existent). This is primarily due to the greater number of reciprocal and square-root operations needed in the block Fourier method when compared to either the approximate Cholesky or Schur. These operations consume many more cycles in a typical processor. Also, we are currently optimizing our FFT performance, however, the Fourier method will still be inefficient in a few cases.

Note that the complexity of the approximate Cholesky and Schur methods depends on the number of rows a_r that are computed. The approximation error is small as long as $a_r > \nu$. Thus the complexity goes up with the delay spread. For example, in Figure 5 with $a_r = 6$, the Schur and Fourier are comparable in complexity. But if $a_r = 4$, the complexity of the Schur algorithm drops (a_r is not relevant to the Fourier method).

The Cholesky and Schur decompositions require

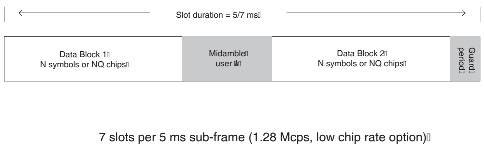

Fig. 1. Structure of a TD-SCDMA burst

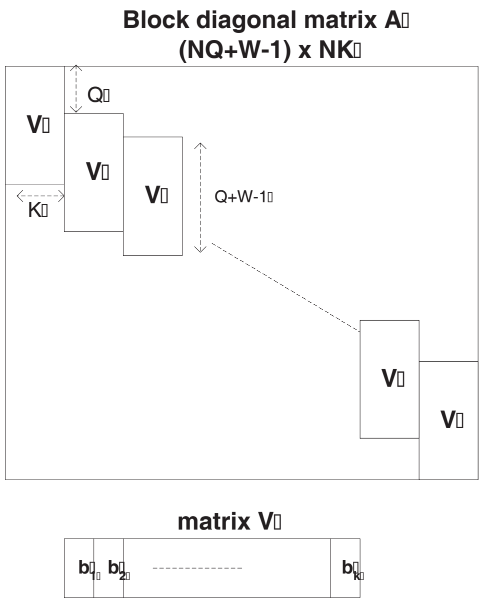

Fig. 2. Structure of the system matrix $\mathbf{A}$. The columns $\mathbf{b}_k = \mathbf{h}_k * \mathbf{c}_k$.

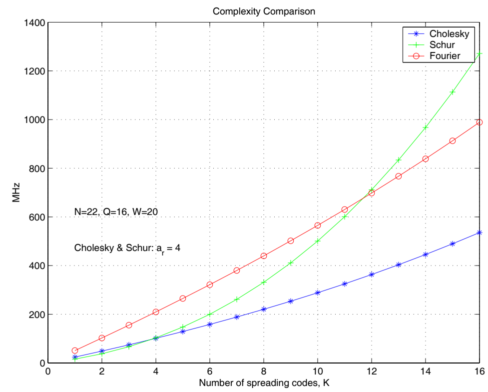

Fig. 3. Complexity as a function of number of spreading codes, $N = 22$

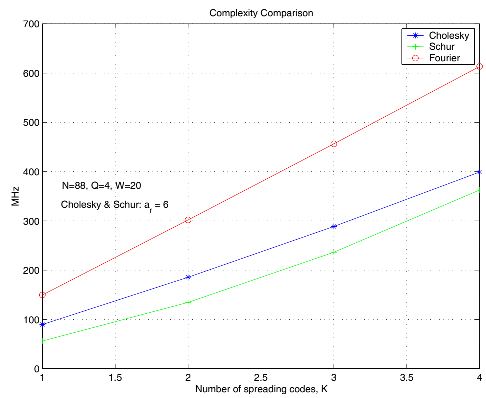

Fig. 4. Complexity as a function of number of spreading codes, $N = 88$

$\mathcal{O}(\nu^2 NK^3)$ and $\mathcal{O}(\nu NK^3)$ respectively. However, the constants involved in the Schur are much larger than those in Cholesky. Therefore, unless ν is large, the complexity of the two algorithms is comparable. However, the orthogonal operations used in the Schur algorithm are less susceptible to numerical errors than are the row operations used in the Cholesky decomposition [11]. Thus, depending on the given scenario, one of the three algorithms may be used for joint detection. For example, for longer channel delay spreads, the Fourier method, since it is a frequency domain approach offers lower complexity. While support of multiple algorithms is expensive to implement in hardware, it is certainly feasible in a software platform such as the Sandblaster.

V. Conclusions

We have shown that it is possible to implement a sub-optimal TD-SCDMA joint or multi-user detector in a software-based downlink receiver. The candidate algorithms considered each offer different advantages under varying operating conditions. Since the implementation is in software, it is possible, without additional costs, to switch to the algorithm that best suits the given operating scenario.

References

[1] "Zero-forcing and minimum-mean square error equalization for multiuser detection in code-division multiple access channels," *IEEE Trans. on Veh. Tech.*, pp. 276-287, May 1996.

[2] N. W. Anderson, H. R. Karimi, and P. Mangold,"Software-Definable Implementation of a TDMA/CDMA Transciever," *Proc. of ICSPAT*, 1998.

[3] M. Vollmer, M. Haardt, J. Gotze, "Comparative Study of Joint-Detection Techniques for TD-CDMA Based Mobile Radio Systems", *IEEE J. Selected Areas of Communications*, pp. 1461-1475, Aug. 2001.

[4] M. Vollmer, M. Haardt, J. Gotze, "Schur Algorithms for joint-detection in TD-CDMA based mobile radio systems", *Annals of Telecommunications*, pp. 365-378, 1999.

[5] M. Beretta, A. Colamonico, M. Nicoli, V. Rampa, U. Spagnolini, "Space-Time multi-user detectors for TDD-UTRA: design and optimization", *Proc. IEEE of VTC*, pp. 375-379, 2001.

[6] J. Glossner, D. Iancu, J. Lu, E. Hokenek, and M. Moudgill, "Software Defined Communications Baseband Design", *IEEE Communications Magazine*, Vol. 41, No. 1, pages 120-128, January, 2003.

[7] S. Jinturkar, J. Glossner, E. Hokenek, and M. Moudgill, "Programming the Sandbridge Multithreaded Processor", *Proceedings of the 2003 Global Signal Processing Expo (GSPx) and International Signal Processing Conference (ISPC)*, March 31-April 3, 2003, Dallas, Texas.

[8] J. Glossner, D. Iancu, G. Nacer, S. Stanley, E. Hokenek, and M. Moudgill, "Multiple Communication Protocols for Software Defined Radio", *IEE Colloquium on DSP Enable Radio*, pp. 227-236, September 22-23, 2003, ISIL, Livingston, Scotland.

[9] J. Rissanen, "Algorithms for Triangular Decomposition of Block Hankel and Toeplitz Matrices with Application to Factoring Positive Matrix Polynomials," *Math. Computations*, Vol. 27, pp. 147-154, Jan 1973.

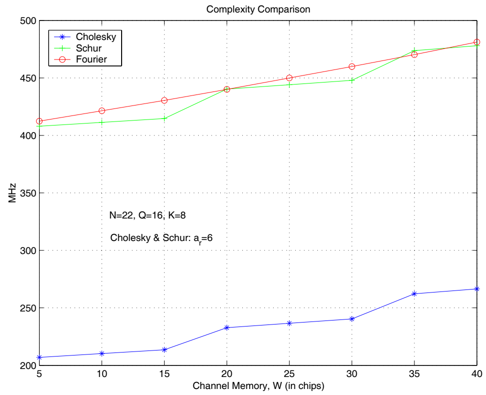

Fig. 5. Complexity as a function of channel memory, $N = 22$

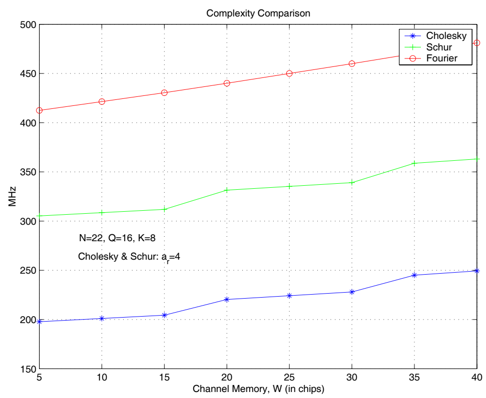

Fig. 6. Complexity as a function of channel memory,$N = 88$

[10] G. H. Golub and C. F. VanLoan, *Matrix Computations*, The Johns Hopkins University Press, 1991.

[11] G. Golub and I. Mitchell, “ Matrix Factorizations in Fixed Point on the C6x VLIW Architecture”, TI report, 1998, http://sccm.stanford.edu/students/mitchell/reportTI.ps

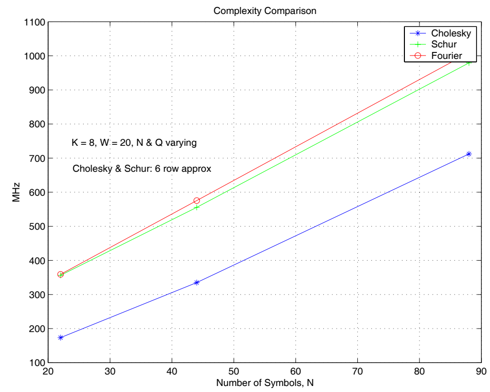

Fig. 7. Complexity as a function of number of user symbols/block, 8 parallel users

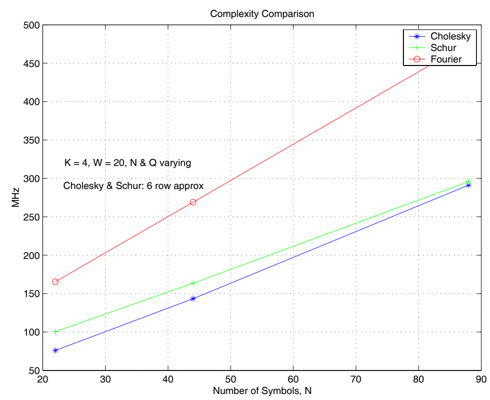

Fig. 8. Complexity as a function of number of user symbols/block, 4 parallel users

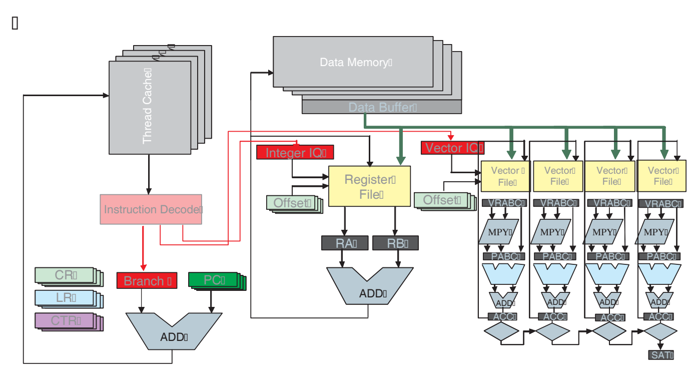

Fig. 9. Sandblaster$^{\mathrm{TM}}$ Multithreaded Processor