A Fixed-Complexity Smart Candidate Adding Algorithm for Soft-Output MIMO Detection

David L. Milliner*, *Student Member, IEEE,* Ernesto Zimmermann, *Member, IEEE,*
John R. Barry, *Senior Member, IEEE,* and Gerhard Fettweis, *Fellow, IEEE*

***Abstract*—We present a soft-output MIMO detection algorithm that achieves near max-log optimal error rate performance with low- and fixed- computational complexity. The proposed SOCA algorithm combines a smart-ordered QR decomposition with smart candidate adding and a parallel layer-by-layer search of the detection tree. In contrast to prior algorithms that use smart candidate adding, the proposed algorithm has fixed computational complexity, and it never visits a node more than once. Results indicate that the SOCA algorithm has an attractive performance-complexity profile for both fast and slow fading 4×4 and 8×8 MIMO channels with QAM inputs.**

***Index Terms*—Soft-Output MIMO Detection, Parallel Smart Candidate Adding, List Detection, Smart-Ordered QR, Breadth-first, SOCA, Tree Search.**

I. Introduction

THE use of multiple antennas at the transmitter and the receiver leads to a *multiple-input multiple-output (MIMO)* channel that can significantly increase the data rate and the reliability of a wireless communication link, without the need for increased transmit power or additional bandwidth. Like any communication system, a MIMO system relies on error-control coding to ensure reliable communication in the presence of noise. While in principle a receiver could *jointly* account for the mutual interference introduced by the MIMO channel and the constraints introduced by the channel code, a practical receiver will account for them separately using first a MIMO detector, which effectively ignores the presence of the code by assuming that the code bits are independent and uniformly distributed, and then an error-control decoder. With this assumption, the best the detector can do is to compute the *a posteriori probability (APP)* for each of the coded bits, which is the conditional probability that each coded bit is 1 (or 0) given the observation of the channel output. The problem of soft-output detection, which aims to compute or approximate these APPs, is important for two fundamental reasons first, the performance of the error-control decoder depends critically on how well its inputs approximate the true APPs, and second, the high complexity of soft-output detection can easily dominate the other receiver tasks such as error-control decoding. Specifically, the complexity of exact APP computation grows exponentially with the spectral efficiency, making it prohibitively complex even for MIMO systems with moderately large antenna arrays and modulation alphabets.

Soft-output MIMO detectors have been proposed based on a variety of principles, including Monte Carlo methods [4], semidefinite programming [5], [6], interference cancelation [7], sphere projection [8] and tree search [9], [10]. Of particular interest is the *smart candidate adding (SCA)* approach of [11]–[13], where a maximum a posteriori (MAP) estimate (or an approximation thereof) is supplemented by directed searches for *counterhypotheses* to this estimate. In [14] an improvement over the SCA approaches of [11]–[13] was proposed that finds its list using a single pass through the detection tree, rather than using multiple searches. A low complexity SCA approach, motivated by the algorithm in [14], was proposed in [15]. However, [11]–[15] suffer from the problem of variable computational complexity and a potentially high worst-case complexity. Recently, a fixed complexity soft-output detection algorithm was proposed in [16].

Our contribution is a *low- and fixed- computational complexity* tree-based solution to the soft-output MIMO detection problem that never visits nodes in the detection tree more than once. We call our solution the *smart ordering and candidate adding (SOCA)* algorithm. The SOCA algorithm sacrifices max-log optimality for low and fixed computational complexity. The SOCA algorithm achieves a desirable performance-complexity profile by combining an intelligent ordering algorithm with SCA techniques and a parallel search of the detection tree.

We now briefly provide the necessary notation. Matrices are set in boldface capital letters and vectors in boldface lowercase letters. We denote the entry in the i^{th} row and v^{th} column of the matrix $\mathbf{R}$ as R_{iv}, the v^{th} column of $\mathbf{R}$ as $\mathbf{R}_v$ and the i^{th} entry of the vector $\mathbf{b} = [b_1\ b_2\ \ldots\ b_N]^T$ as b_i. The superscripts ^{T} and * stand for the transpose and conjugate transposition, respectively. $\mathbf{I}_N$ and $\mathbf{0}_N$ denote the $N \times N$ identity and zero matrices, respectively. Finally, $|\mathcal{A}|$ denotes the cardinality of the set $\mathcal{A}$.

Manuscript received September 17, 2008; revised April 6, 2009. This paper was presented in part in the 2007 ITG/IEEE Workshop on Smart Antennas (WSA'07) [1], the 7th International ITG Conference on Source and Channel Coding (SCC'08) [2] and the 10th International Symposium on Spread Spectrum Techniques and Applications (ISSSTA'08) [3]. This work was supported in part by Texas Instruments, Inc., the German Academic Exchange Service (DAAD) and National Science Foundation grant no. 0431031.

D. L. Milliner and J. R. Barry are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, 30332 USA e-mail: {dlm,barry}@ece.gatech.edu. E. Zimmermann is with RadioOpt, D-01069 Dresden, Germany, email: ernesto.zimmermann@radioopt.com. G. Fettweis is with the Vodafone Chair Mobile Communications Systems, Technische Universität Dresden, D-01062 Dresden, Germany e-mail: fettweis@ifn.et.tu-dresden.de.

*Corresponding Author

The remainder of this paper is organized as follows: after discussing the employed system model in section II, section III describes the problem of soft-output MIMO detection. This is followed by a presentation of the proposed fixed-complexity SOCA detection algorithm in section IV. Results are presented in section V and conclusions are drawn in section VI.

II. System Model

We assume the transmitter shown in Fig. 1(a) [9]. The input is a vector $\mathbf{u}$ of i.i.d. uniform information bits that is encoded and interleaved. The coded bit stream is partitioned into blocks $\mathbf{c}$ of ωN_t bits. Each block is mapped onto a vector $\mathbf{a}$ whose N_t component symbols are taken from a QAM alphabet $\mathcal{A}$ of size $q = |\mathcal{A}| = 2^\omega$ and energy $\mathbb{E}\left(|a_i|^2\right) = E/N_t$, where ω is the number of bits per symbol. We define $\mathcal{Z} = \mathcal{A}^{N_t}$ as the set of all possible symbol vectors $\mathbf{a} \in \mathcal{Z}$, one for each binary vector $\mathbf{c} \in \{\pm 1\}^{\omega N_t}$, as determined by the mapping from coded bits to transmitted symbols.

The vector of symbols $\mathbf{a}$ is transmitted through an $N_r \times N_t$ MIMO channel whose equivalent complex baseband model is:

$$\mathbf{r} = \mathbf{H}\mathbf{a} + \mathbf{n}, \tag{1}$$

where $\mathbf{r} \in \mathbb{C}^{[N_r \times 1]}$ is the received signal vector, $\mathbf{n} \in \mathbb{C}^{[N_r \times 1]}$ is additive noise, and $\mathbf{H} \in \mathbb{C}^{[N_r \times N_t]}$ is the channel matrix. We are thus assuming a single-carrier narrowband flat-fading channel. We assume additive white Gaussian noise (AWGN), so that the components of $\mathbf{n}$ are zero-mean, circularly symmetric, i.i.d. complex Gaussian random variables with variance N_0, satisfying $\mathbb{E}[\mathbf{n}\mathbf{n}^*] = N_0\mathbf{I}_{N_r}$. The entry in the i^{th} row and v^{th} column of $\mathbf{H}$ represents the complex channel gain between transmit antenna v and receive antenna i. We assume Rayleigh fading, typical of non-line-of-sight communication systems, so that the entries of $\mathbf{H}$ are i.i.d. complex Gaussian random variables of variance 1. The signal-to-noise ratio (SNR) at any receive antenna is SNR $= E/N_0$. We assume $N_r \geq N_t$ and that the receiver knows the channel perfectly.

In Fig. 1(b) we show a MIMO receiver consisting of a MIMO detector (this paper's focus), a deinterleaver, and an error-control decoder. While an iterative receiver based on the turbo principle [17] can improve performance, we limit our discussion to a non-iterative system with no feedback from the decoder to the detector. This limitation simplifies our presentation while still retaining the essential features of the problem.

III. Problem Statement

The aim of a soft-output detector is to calculate or approximate the *a posteriori* probability (APP) for each of the code bits c_j in a given signaling interval, where $j \in \{1, \ldots \omega N_t\}$ is the bit index. This probability is conveniently represented by the so-called a posteriori log-likelihood ratio (LLR):

$$L(c_j|\mathbf{r}) := \ln \frac{\Pr[c_j = +1|\mathbf{r}]}{\Pr[c_j = -1|\mathbf{r}]}. \tag{2}$$

The sign of $L(c_j|\mathbf{r})$ is the maximum a posteriori (MAP) estimate for c_j, and the magnitude represents the reliability of the estimate. Larger magnitudes correspond to higher reliability, and smaller magnitudes indicate low reliability.

After a series of manipulations that exploit (a) the application of Bayes' rule, (b) our assumption of no *a priori* information, (c) our assumption of AWGN and (d) the application of the max-log approximation, (2) reduces to [9]:

$$L(c_j|\mathbf{r}) \approx \min_{\hat{\mathbf{a}} \in \mathcal{Z}_j^{-1}} \frac{\|\mathbf{r} - \mathbf{H}\hat{\mathbf{a}}\|^2}{N_0} - \min_{\hat{\mathbf{a}} \in \mathcal{Z}_j^{+1}} \frac{\|\mathbf{r} - \mathbf{H}\hat{\mathbf{a}}\|^2}{N_0}, \tag{3}$$

where $\mathcal{Z}_j^{\pm 1}$ denotes a partitioning of $\mathcal{Z}$ depending on whether the j^{th} bit label is 1 or -1, namely:

$$\begin{aligned} \mathcal{Z}_j^{+1} &= \{\mathbf{a}(\mathbf{c}) : c_j = +1\}, \\ \mathcal{Z}_j^{-1} &= \{\mathbf{a}(\mathbf{c}) : c_j = -1\}. \end{aligned} \tag{4}$$

Although at a glance it might appear from (3) that the receiver would need to perform a pair of optimizations for each of the ωN_t bits of interest, the MAP solution will obviously always be one of each pair. The other of each pair is called its counterhypothesis. So to compute (3) exactly (i.e. max-log optimally) for each of the ωN_t bits, it is sufficient to find the MAP solution once, and then, for each of the ωN_t bits, to negate the bit of interest and solve a constrained optimization problem to find the optimal counterhypothesis.

An exact solution to (3) need only consider a *list* containing the MAP candidate along with at most ωN_t counterhypotheses[1]. *List detection* is the process of finding a list of candidates $\mathcal{L} \subseteq \mathcal{Z}$. From this list, (3) is approximated as:

$$L(c_j|\mathbf{r}) \approx \min_{\hat{\mathbf{a}} \in \mathcal{L} \cap \mathcal{Z}_j^{-1}} \frac{\|\mathbf{r} - \mathbf{H}\hat{\mathbf{a}}\|^2}{N_0} - \min_{\hat{\mathbf{a}} \in \mathcal{L} \cap \mathcal{Z}_j^{+1}} \frac{\|\mathbf{r} - \mathbf{H}\hat{\mathbf{a}}\|^2}{N_0}. \tag{5}$$

The only difference between (3) and (5) is the insertion of the list $\mathcal{L}$, whose size is typically much less than that of $|\mathcal{Z}|$. The list size $\ell = |\mathcal{L}|$ plays a critical roll in the overall complexity and performance.

The problem of soft-output MIMO detection can thus be phrased as the search for the counterhypothesis list. The contribution in this paper is a list detection algorithm that closely approximate the max-log optimal counterhypothesis list with low and fixed computational complexity.

IV. SOCA Algorithm

Our solution to the soft-output MIMO detection problem is called the *smart ordering and candidate adding* algorithm. Two stages comprise the SOCA algorithm, a preprocessing stage and a core processing stage. The preprocessing stage is used to determine the mapping between layers in the detection tree and the transmitted vector of information symbols. The core processing finds the list $\mathcal{L}$, the output of the SOCA algorithm. Because the preprocessing can be considered a performance enhancement, we begin by describing the core processing.

A. SOCA Core Processing

The SOCA algorithm finds $\mathcal{L}$ using a standard detection tree. After a QR-decomposition of the channel matrix, $\mathbf{H} =$

[1] A single vector may serve as the counterhypothesis for multiple bits.

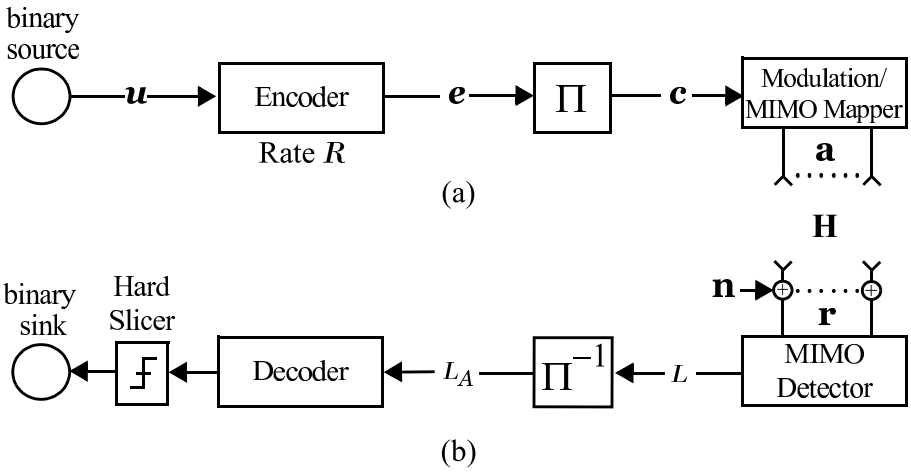

Fig. 1: System model with (a) MIMO transmitter and (b) MIMO receiver.

$\mathbf{QR}$, the squared Euclidean distance for a candidate $\hat{\mathbf{a}}$ is:

$$J(\hat{\mathbf{a}}) = \|\mathbf{r} - \mathbf{H}\hat{\mathbf{a}}\|^2 \quad (6)$$

$$= \|\mathbf{y} - \mathbf{R}\hat{\mathbf{a}}\|^2 \quad (7)$$

$$= \sum_{i=1}^{N_t} |y_i - \sum_{v=1}^{i} R_{iv}\hat{a}_v|^2, \quad (8)$$

where $\mathbf{R}$ is an $N_t \times N_t$ lower triangular matrix, $\mathbf{Q}$ is an orthogonal matrix and $\mathbf{y} = \mathbf{Q}^*\mathbf{r}$. The cost function (8) can be interpreted as the sum of N_t *branch metrics*, one for each branch in a path from the root to a leaf node, where the metric for a branch in the i-th stage of the detection tree is:

$$\left| y_i - \sum_{v=1}^{i} R_{iv} a_v \right|^2. \quad (9)$$

The cost of a node is defined as the sum of the branch metrics in the path from the root to the node.

From the root node in the tree at layer 0 there are q children, one for each value of a_1. From each child node (now parent nodes) are q children, one for each a_2. This continues until there are q^{N_T} leaf nodes at layer N_t. There is a one-to-one correspondence between a leaf node and a particular $\mathbf{a} = [a_1\ a_2\ \ldots\ a_{N_t}]^T$. The detection process can hence be interpreted as a search for leaf nodes in a tree, corresponding to elements of a list $\mathcal{L}$.

The foundation of the SOCA algorithm is a simple breadth-first strategy for searching the tree that is closely related to the M algorithm [18]. Like the M algorithm, the SOCA algorithm moves through the tree one layer at a time, discarding all but a subset of "surviving" nodes from a given layer before moving to the next. One minor difference is how many surviving nodes are retained at each layer; rather than keeping this fixed, the SOCA algorithm allows for the possibility that this number m_i may depend on the layer index i. Another minor difference is how many children from each surviving node are extended; rather than keeping this fixed, the SOCA algorithm allows for the possibility that this number b_i may also depend on the layer index i. (Similar variations of the M algorithm have appeared in [3], [19].)

The SOCA algorithm builds upon its breadth-first foundation by inserting a new step. Before pruning away all but the m_i best surviving nodes from a current set of candidate nodes at layer i, the SOCA algorithm identifies the candidate node with the best metric as the *partial MAP (PMAP)* node. Once identified, the SOCA algorithm adds new nodes to the candidate set so that each of the ω bits corresponding to the current symbol a_i has a counterhypothesis. Specifically, if $\hat{\mathbf{c}}^{\text{PMAP}}$ denotes the ωi-bit pattern corresponding to the node with the best metric, with the last ω of these bits corresponding to a_i, then the SOCA algorithm adds the ω sibling nodes of the partial-MAP node by simply flipping each of the last ω bits of $\hat{\mathbf{c}}^{\text{PMAP}}$ in turn. This bit flipping strategy was chosen because of its low complexity, despite the facts that (1) the counterhypotheses so generated may not be the ones having the best metric, and (2) a counterhypothesis for the bit in question may already be represented in the candidate set. Once added, these counterhypotheses may be immediately pruned, although our results indicate that for MIMO system sizes at least as large as 4×4, the performance benefit of protecting these added counterhypotheses combined with the increased computational complexity of a candidate sort to determining which nodes to prune mean that protecting all enumerated nodes for the SOCA algorithm is advised.

In the case of gray mapping and QAM alphabets, while the sibling nodes are not guaranteed to have small metrics, they are likely to have small metrics because at least two, and at most four, of the siblings are nearest neighbors of the transmitted symbol estimate. In fact, exactly two, three and four siblings correspond to nearest neighbors in the case of an estimated corner, border, and interior point for a gray-mapped QAM alphabet, respectively. We refer to our low and fixed complexity technique for finding counterhypotheses as parallel smart candidate adding (PSCA).

Because the SOCA algorithm is breadth-first and possesses fixed computational complexity it lends itself well to architectural implementation. For reasons we will discuss later, namely the preprocessing algorithm, the SOCA algorithm does not

need to consider the problem of missing counterhypotheses to the children extended from the root node, i.e. the SOCA algorithm does not concern itself with missing counterhypotheses at the first layer of the detection tree.

Like any other algorithm built on the foundation of the M algorithm, the tree for the SOCA algorithm can be pruned using a sort-and-select procedure, reducing the number of nodes to the m_i best nodes whenever m_i is less than the number of nodes enumerated at the current layer in the tree. When m_i is larger than the number of nodes extended at a given layer in the tree, this sort-and-select stage is omitted for reduced complexity. In section V we employ no pruning for soft-output detection of a 4×4 MIMO channel because we can obtain near max-log optimal performance with $b_i = 1 \ \forall i > 1$. Here, instead of a sort-and-select procedure, all that is required is to determine the minimum cost node at each layer in the tree. We will, however, resort to pruning for soft-output detection of an 8×8 MIMO channel, where the dimensionality of the problem leads to higher complexity to maintain near-optimal performance. When a sort-and-select is employed one option is the heapsort algorithm [20]. The heapsort algorithm, at the i^{th} layer of the tree, is achieved with computational complexity $\Theta(m_i \log m_i)$.

A concise description of the SOCA algorithm is provided in Fig. 2. In summary, the SOCA algorithm takes as input the received signal $\mathbf{r}$, the MIMO channel $\mathbf{H}$, the alphabet $\mathcal{A}$, and two vectors $\mathbf{b} = [b_1 \ b_2 \ \dots \ b_{N_t}]$ and $\mathbf{m} = [m_1 \ m_2 \ \dots \ m_{N_t}]$, where $\mathbf{b}$ grows the tree by adding nodes and $\mathbf{m}$ prunes the tree by deleting nodes. The set $\mathcal{S}$ is used to denote the surviving nodes at the current layer in the tree. We recommend keeping the elements of $\mathbf{b}$ small (1 if possible), with an exception for the first detection layer (i.e. $b_1 > 1$) due to the fact that the diversity order of the first symbol to be detected is $N_r - N_t + 1$ and a mistake here leads to error propagation. In many practically relevant system configurations $\mathbf{b} = [b_1 \ 1 \ \dots \ 1]$ with b_1 set to between 25% and 50% of q yields excellent performance at very low complexity. The reason we do not need to set b_1 equal to q is because of our use of a smart-ordered QR algorithm, i.e. the first line in Fig. 2. Without lines 6 through 11 and the assurance of a smart-ordered QR decomposition in line 1, the rest of the pseudocode is simply the M algorithm with variable $\mathbf{b}$ and $\mathbf{m}$.

Algorithm: SOCA
Input: $\mathbf{r}, \mathbf{H}, \mathbf{b}, \mathbf{m}$
Output: $\mathcal{L}$

1 $[\mathbf{Q}, \mathbf{R}, \mathbf{P}] = \texttt{SOQR}(\mathbf{H}, b_1)$
2 $\mathbf{y} = \mathbf{Q}^*\mathbf{r}$
3 $\mathcal{S} =$ root node
4 **for** $i = 1 : N_t$ **do**
5 $\quad \mathcal{S} = \cup_{\texttt{node}\in\mathcal{S}}$ $\{b_i$ best children of node$\}$
6 $\quad$ **if** $i > 1$ **then**
7 $\quad\quad$ **for** $j = (i-1)\omega + 1 : i\omega$ **do**
8 $\quad\quad\quad$ Flip bit j of $\hat{\mathbf{c}}^{\text{PMAP}}$ and add the corresponding node to $\mathcal{S}$
9 $\quad\quad$ **end**
10 $\quad$ **end**
11 $\quad \mathcal{S} = m_i$ best of $\mathcal{S}$
12 **end**
13 $\mathcal{L} = \mathbf{P}\mathcal{S}$

Fig. 2: SOCA Algorithm Description.

B. Ordering of the Channel Matrix

The mapping of layers in the detection tree to transmitted symbols is a critical factor in determining either the performance or the computational complexity (or both) for soft-output MIMO detection. This mapping, which is a direct consequence of the channel matrix ordering during the QR decomposition, deserves careful attention. Specifically, the detection order is a key factor affecting the performance of all suboptimal tree-based detectors, from simple decision-feedback detectors [21], [22] to hard-output breadth-first detectors [19], [23], [24] to, as we will show, soft-output fixed-complexity breadth-first detectors. Layer ordering can, in addition to improving the performance of suboptimal detectors, also be used to reduce the average computational complexity of variable complexity detectors [15].

An ordered QR decomposition is used to achieve the desired ordering, i.e. $\mathbf{HP} = \mathbf{QR}$, where $\mathbf{P}$ is a permutation of $\mathbf{I}_{N_t}$. The BLAST ordering [22] is the optimal detection order for hard-output decision-feedback detection, when only a single path of the tree from root to leaf is traversed, since it maximizes the SNR at each layer. An attractive alternative to the BLAST ordering is the sorted QR decomposition (SQRD) [25], which achieves nearly the same hard-output decision-feedback performance as the BLAST ordering with reduced complexity.

The BLAST ordering is not generally optimal when more than one node is enumerated at any stage in the detection tree. For example, the parallel detector (PD) of [23] enumerates all q child nodes of the root node and extends each of these nodes using decision feedback to obtain q leaf nodes. The parallel detector works best when the *weakest* received signal component is detected first, so that its contribution is completely removed from the detection problem. Intuitively, this is because there is no possibility for an error to occur in a layer where all child nodes are enumerated. Consequently, it is desirable to enumerate all child nodes in the layer with the largest noise enhancement to minimize performance loss. In [19] the parallel detector ordering was extended by employing the weakest-first parallel detector ordering for layers where all q child nodes are enumerated and the strongest-first BLAST ordering for all other layers. In [26] it was shown that the ordering of [19] maintains the diversity order of the maximum-likelihood detector with a fixed complexity and order $\Theta(q^{\sqrt{N_t}})$ if $N_r = N_t$, when all nodes in the first $\lfloor\sqrt{N_t}\rfloor$ layers are enumerated.

A detection order for cases where the number of child nodes to be enumerated from each parent is between 1 and q is given by the B-Chase detector [24]. B-Chase preprocessing has been shown to gracefully trade off between the opposing design goals of maximizing (as in the BLAST ordering) vs. minimizing (as in the PD ordering) the SNR of the first

detection layer by allowing the ordering algorithm to consider an increase in the number of child nodes enumerated from the root node as an effective SNR gain for the receiver.

We now present a particular B-Chase preprocessing configuration that we found to perform well. We call this configuration the smart-ordered QR (SOQR) decomposition. A SOQR decomposition takes as inputs $\mathbf{H}$ and b_1 and produces the outputs $\mathbf{Q}$, $\mathbf{R}$, and $\mathbf{P}$. The key step in the SOQR is to determine which layer to detect first. This decision is a function of the per-layer SNRs and b_1. As b_1 is increased from 1 to q the layer selected to be detected first moves from the one with the highest SNR to the one with the lowest. This is done so that as b_1 is increased to approach q we order the detection based on the assumption that detection errors in the first layer in the tree are unlikely. Indeed, they are impossible when $b_1 = q$.

We propose that the index k of the first layer to be detected be chosen according to (10), i.e. the B-Chase criterion [24]. The matrix $\mathbf{Y}^* = \mathbf{R}^{-1}$ is determined by a QR decomposition of the channel matrix, i.e. $\mathbf{QR} = \mathbf{H}$. Additionally, $g_{s,n} = \mathbf{Y}_s^* \mathbf{Y}_n / \|\mathbf{Y}_n\|$, where $\mathbf{Y}_n$ is the nth column of $\mathbf{Y}$. The parameter $\gamma_{b_1}^2$ is the effective SNR gain (see [24]) at the first detection layer when b_1 child nodes are enumerated from the root node. For QPSK transmission $\gamma_1^2 = 1$, $\gamma_2^2 = \gamma_3^2 = 2$ and $\gamma_4^2 = \infty$. Values of $\gamma_{b_i}^2$ for 16 and 64-QAM transmission are found in [24]. However, because the value for γ can be determined using a lookup table that is a function of the parameter b_1, the selection for b_1 does not influence the complexity of the SOQR decomposition. The complexity of (10) is dominated by computing the squared column norm $\|\mathbf{Y}_n\|^2$ a total of N_t times and the $N_t(N_t - 1)$ vector multiplications $\mathbf{Y}_s^* \mathbf{Y}_n$ to compute all $g_{s,n}$ values.

After selecting the index of the first layer to be detected, the remainder of the SOQR is essentially a SQRD [25], where the ordering of the first detection layer is forced. The SOQR can be achieved with complexity order $\Theta(N_t^3)$. Pseudocode for the `SOQR` algorithm is provided in Fig. 3. Note that the forced ordering in line 1, the initialization of k_2 in line 3, and the forced ordering of lines 5-12 that ensure the first layer detected is chosen according to (10).

Algorithm: `SOQR`
Input: $\mathbf{H}, b_1$
Output: $\mathbf{Q}, \mathbf{R}, \mathbf{P}$

1 Find k using (10): a function of $\mathbf{H}$ and b_1
2 $\mathbf{Q} = \mathbf{H}$, $\mathbf{R} = \mathbf{0}_{N_t}$, $\mathbf{P} = \mathbf{I}_{N_t}$
3 $\mathbf{d} = \mathtt{diag}(\mathbf{Q}^*\mathbf{Q})$; $k_2 = k$
4 **for** $i = N_T : -1 : 1$ **do**
5 **if** *i==1* **then**
6 $k = 1$
7 **else**
8 $k = \arg\min_{k \neq k_2} \mathbf{d}$
9 **end**
10 **if** $i == k_2$ **then**
11 $k_2 = k$
12 **end**
13 Swap columns i and k in $\mathbf{Q}$, $\mathbf{R}$, $\mathbf{P}$
14 Swap elements i and k in $\mathbf{d}$
15 $R_{i,i} = \sqrt{d_i}$
16 $\mathbf{q}_i = \mathbf{q}_i / R_{i,i}$
17 **for** $j = i - 1 : -1 : 1$ **do**
18 $R_{i,j} = \mathbf{q}_i^* \mathbf{q}_j$
19 $\mathbf{q}_j = \mathbf{q}_j - R_{i,j}\mathbf{q}_i$
20 $d_j = d_j - |R_{i,j}|^2$
21 **end**
22 $d_i = \infty$
23 **end**

Fig. 3: Smart-Ordered QR (SOQR) Decomposition.

C. Computational Complexity

The computational complexity of list MIMO detection can be measured in many ways, e.g. using the number of floating point operations, silicon area, or amenability to parallelization. However, such measures depend critically on the specific target architecture: fixed vs. floating point operation, ASIC vs. FPGA, etc. A well accepted and relatively architecture-agnostic metric for computational complexity is the number of visited nodes in the detection tree [14], [27]. This metric has gained increased popularity with the introduction of a one node per cycle hardware implementation as reported in [27].

In this paper we quantify computational complexity by the number of nodes visited, with the understanding that any node whose branch metric is computed is considered a visited node. This avoids a complexity comparison that is unbalanced in favor of schemes that calculate metrics for nodes they later discard (as, e.g. the M algorithm) as opposed to algorithms that only calculate metrics for nodes they do visit (as, e.g. the Schnorr-Euchner sphere decoder).

Observe that using the number of visited nodes omits the complexity of the preprocessing algorithm. Such an omission may not be acceptable for fast-fading scenarios where the computational complexity of the detection ordering dominates, but is more appropriate for scenarios where the coherence time of the channel is long. Additionally, the preprocessing complexity can be extracted from our discussion because the preprocessing algorithms for all detectors compared in section V possess the same complexity order $\Theta(N_t^3)$.

The SOCA algorithm has the property that when $\mathbf{m} = q^i$) and $\mathbf{b} = [b_1\ 1\ \ldots\ 1]$, as is the case for the 4×4 results presented in section V, only

$$\mu = b_1 + \sum_{i=2}^{N_t} b_1 + \omega(i-1) = N_t \left(b_1 + \frac{\omega(N_t - 1)}{2} \right) \quad (11)$$

branch metrics are computed. Multiplying out the right hand side of (11) demonstrates that the number of branch metric computations for the SOCA algorithm has order $\Theta(N_t^2)$. Additionally, the list size for the SOCA algorithm when $\mathbf{b} = [b_1\ 1\ \ldots\ 1]$ is given by:

$$\ell = \min\left(m_1, b_1 + \omega\left(N_t - 1\right)\right). \quad (12)$$

In [3] the interested reader will find general expressions for the number of branch metric computations and list sizes for

$$k = \begin{cases} \underset{n\in\{1,\dots N_t\}}{\arg\max} \|\mathbf{Y}_n\|^2, & b_1 = q \\ \underset{n\in\{1,\dots N_t\}}{\arg\max} \min\left\{ \frac{\gamma_{b_1}^2}{\|\mathbf{Y}_n\|^2}, \ \frac{1}{\min\limits_{s\neq n}\{\|\mathbf{Y}_s\|^2 - |g_{s,n}|^2\}} \right\}, & \text{otherwise} \end{cases} \tag{10}$$

fixed-complexity breadth-first tree-based detection algorithms. Consequently, for the SOCA algorithm, the list size grows linearly with the number of input streams.

V. Results and Analysis

A. Simulation Setups

In this section we present results for both fast and slow fading scenarios in order to demonstrate the value of the proposed algorithm in situations where the ability to extract either (a) the time diversity or (b) the spatial diversity from the channel is essential. For both the fast and slow scenarios the detector is run only once, i.e. we do not employ iterative detection-decoding. The proposed SOCA algorithm is compared against the list sphere detector (LSD) [9] and the list sequential detector (LISS) [28], where the LISS employs a bias parameter to perform statistical tree pruning. This LISS length bias term is set to 1 at each level of the detection tree [29] to reduce computational complexity at the cost of a small performance penalty relative to the LSD with the same list size. We also compare against the recently introduced single-tree-search LSD (STS-LSD) algorithm [15] and the list fixed-complexity sphere detector (LFSD) [16]. For completeness, we compare against a soft-output implementation of the M algorithm [18], where we form $\mathcal{L}$ from the m_{N_t} best leaf nodes at the final detection layer.

For the LSD, LISS, M, LFSD and SOCA detectors we clipped the LLRs at a magnitude of ± 6. For the STS-LSD algorithm we varied the value of the clipping parameter $L_{\max}$ to achieve a performance complexity tradeoff[2]. As $L_{\max}$ approaches infinity, the performance of the STS-LSD approaches that of the max-log optimal detector [15]. Conversely, as $L_{\max}$ approaches zero the performance of the STS-LSD approaches that of the hard-output joint maximum likelihood (JML) detector. For all algorithms we employ unbiased MMSE detection, meaning that we removed the contribution of the MMSE extended channel matrix from the node metric computations so that the node metrics are exactly (6) (see [30] for details). SQRD preprocessing is used to reduce the complexity of the variable complexity algorithms.

Because the LSD, STS-LSD and LISS algorithms have variable computational complexity, results for the average and the 99.9th percentile computational complexities, as measured in terms of the number of visited nodes, are provided. We chose 99.9% to ensure that, if we limited the maximum complexity of the tree search, the number of additional errors introduced would be substantially lower than a typical raw bit error rate at the input of the channel decoder – say 0.01 (so 99.9% would be 10% additional errors). The LFSD, M, and SOCA algorithms have fixed computational complexity and so only one number is reported. The M algorithm was realized using SQRD preprocessing, while the LFSD and SOCA employ their algorithmic specific preprocessing. Finally, all algorithms employ the best-first Schnorr-Euchner enumeration [31], rather than Fincke-Pohst enumeration [32].

We consider transmission over a spatially i.i.d. fading 4×4 MIMO channel using 16-QAM and 64-QAM modulation alphabets and an 8×8 MIMO channel using 16-QAM inputs. A random interleaver is employed and detection is performed using the complex-valued system model.

1) Fast Fading: For the fast-fading scenario we use temporally i.i.d. fading, i.e., each transmitted vector symbol sees a new channel realization. The information block size (including tail bits) is 9216 bits. We use a setup equivalent to the one in [9]: a rate 1/2 parallel concatenated convolutional code (PCCC) based on memory 2 constituent convolutional codes with generator polynomials $(7_R, 5)_{\text{octal}}$ using 8 internal iterations of logMAP decoding, where R denotes which generator is in the denominator. Fast-fading performance is measured in terms of the averaged E_b/N_0 in dB to achieve a bit error rate of 10^{-5} to match [9].

2) Slow Fading: Here we assume the channel does not change during the duration of the entire transmitted codeword and that the channel matrix entries are drawn anew with the transmission of each new codeword. A convolutional code with code polynomial $[133\ 171]$ and constraint length 7, punctured to code rate $3/4$ is employed and the information block size (including tail bits) is 3456. The convolutional decoder employed is MaxLog(MAP). Performance is measured in terms of the E_b/N_0 in dB required to achieve a frame error rate (FER) of 10^{-2}. We used the FER to measure slow-fading performance because for this scenario, where we employ a weak code and the channel offers no time diversity, BER results can often be misleading. The target FER of 10^{-2} was selected because it is common to design systems for this error rate [15].

B. Results

Fig. 4 depicts performance versus computational complexity for 16-QAM transmission in fast Rayleigh fading. The average computational complexities for the LSD, LISS, and STS-LSD are represented using dashed lines and the 99.9th percentile computational complexities are represented using solid lines. The LSD is represented by square markers, the STS-LSD by diamond markers and the LISS by circular markers. For the LSD the list size ℓ is provided for each marker. The same list sizes are represented for the LISS, although the performance results differ due to the statistical tree pruning performed by

[2]The values for $L_{\max}$ we provide are larger than reported in [15], but are equivalent, because we do not normalize by the noise variance.

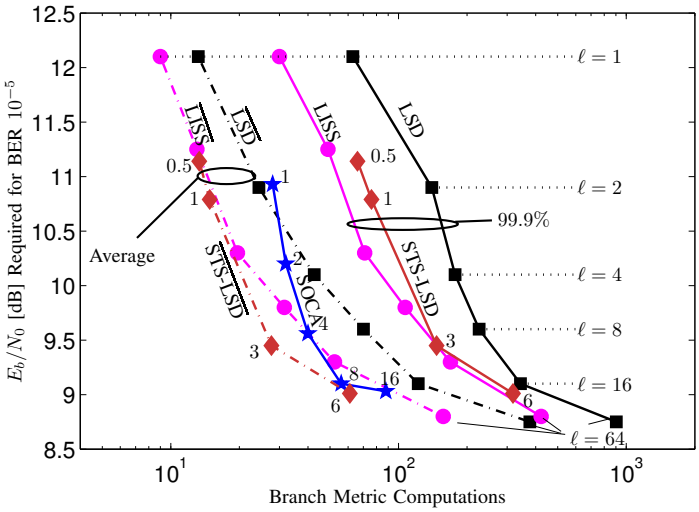

Fig. 4: Performance vs. complexity for soft-output 4×4 MIMO detection schemes using 16-QAM transmission in fast Rayleigh fading. The numbers corresponding to the SOCA curve represent the value for b_1 and the numbers corresponding to STS-LSD curves represent $L_{\max}$.

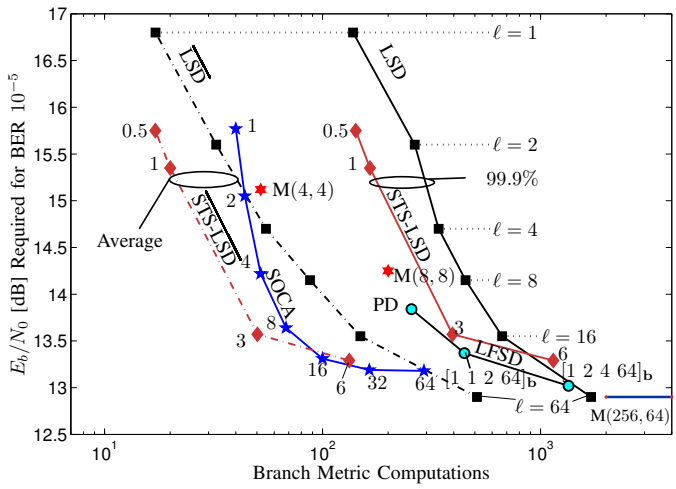

Fig. 5: Performance vs. complexity for soft-output 4×4 MIMO detection schemes using 64-QAM transmission in fast Rayleigh fading. Results for the LFSD [16] are provided for $\mathbf{b} = [64\ 1\ 1\ 1]$, $\mathbf{b} = [64\ 2\ 1\ 1]$ and $\mathbf{b} = [64\ 4\ 2\ 1]$.

the LISS [29]. For the STS-LSD the numbers next to the markers represent the value of the clipping/pruning parameter $L_{\max}$ as described in [15], instead of the list length, because this parameter determines the computational complexity for the STS-LSD. In addition to the variable complexity algorithms, the solid curve denoted with pentagram markers represents the proposed fixed complexity SOCA algorithm. The numbers corresponding to each SOCA marker denote the number of nodes enumerated at the first detection layer b_1. Finally, we note that for all 4×4 SOCA results we use the parameterization $\mathbf{b} = [b_1\ 1\ 1\ 1]$, where b_1 is the number of child nodes enumerated from the root of the tree, and $\mathbf{m} = \infty$ so that no tree pruning occurs. A consequence of omitting tree pruning is that we do not need a sort-and-select stage to determine the m_i nodes to retain at the i^{th} layer of the tree. Instead, all that is needed is the selection of the lowest cost node at any layer in the tree so that parallel smart candidate adding can be applied to this node.

Fig. 4 shows that for the fast-fading case, the averaged computational complexity for the STS-LSD (i.e. $\overline{\text{STS-LSD}}$) algorithm achieves a desirable performance-complexity tradeoff. Often the worst-case (or bounded worst-case) computational complexity is more important in terms of system design. The performance-complexity curve for the $\overline{\text{STS-LSD}}$ therefore serves as a somewhat idealized reference to which other detection algorithms should aspire. Here the fixed-complexity SOCA algorithm with $b_1 = 8$ is an attractive option because, while it performance-complexity profile is worse than the $\overline{\text{STS-LSD}}$, it significantly outperforms the 99.9th percentile STS-LSD. Fig. 4 also shows that for $b_1 > 4$ the SOCA achieves a better performance-complexity tradeoff than the $\overline{\text{LISS}}$ or $\overline{\text{LSD}}$ algorithms. In contrast to all variable complexity algorithms, the SOCA algorithm achieves its performance-complexity tradeoff with fixed computational complexity. This is a desirable property because, like other breadth-first algorithms [33], it leads to a regular design structure that lends itself well to parallelization and low latency.

Fig. 5 provides the same performance-complexity plot as Fig. 4, but for 64-QAM transmission. Once more the SOCA performance-complexity curve falls between that of the average and 99.9th percentile computational complexity for the STS-LSD, with the SOCA having fixed computational complexity. New to Fig. 5 are comparisons against two fixed complexity soft-output algorithms, namely a soft-output version of the M algorithm [18] and the recently proposed list fixed-complexity sphere detector (LFSD) [16]. The M algorithm is denoted by hexagram markers and the corresponding term $\text{M}(m, b)$ is provided, where b is the number of nodes extended and m is the number of nodes retained at each layer in the tree, respectively. When $b = |\mathcal{A}|$ the M algorithm is also known as the K-best algorithm [34]. The reference performance provided was found using the K-best algorithm and $m = 256$. Such a realization would compute over 36000 branch metrics and so the computational complexity is not shown. The LFSD is denoted by lightly shaded circular markers with dark edges. In its minimum configuration the LFSD reduces to a soft-output parallel detector, i.e. $\mathbf{b} = [64\ 1\ 1\ 1]$ with all leaf nodes in the tree used to form $\mathcal{L}$. LFSD results are also provided for $\mathbf{b} = [64\ 2\ 1\ 1]$ and $\mathbf{b} = [64\ 4\ 2\ 1]$, where the subscript $\mathbf{b}$ is used to denote that the vector to which it is attached is $\mathbf{b}$. One reason the SOCA algorithm outperforms the LFSD in terms of the performance-complexity tradeoff is because of the way it adds counterhypotheses. Specifically, rather than increasing the elements of $\mathbf{b}$ like the LFSD (i.e. $b_i > 1$), the SOCA simply bit flips around the estimate that is currently best, thereby growing the tree by addition of nodes rather than a multiplicative factor of nodes. Additionally, because of its use of the SOQR, the SOCA does not need to extend all $q = 16$ child nodes at the first layer of the tree to achieve good performance.

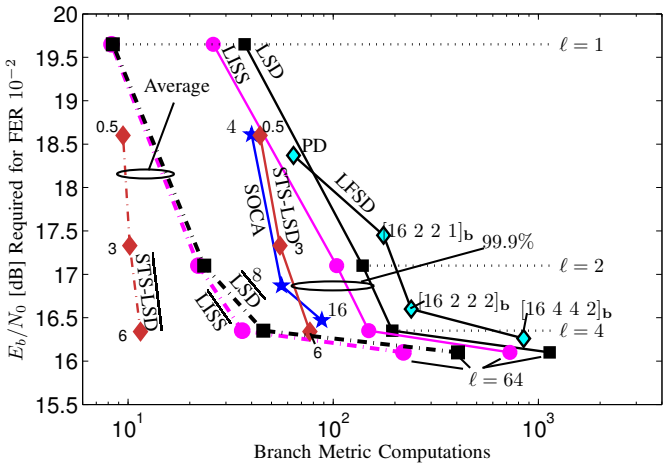

Fig. 6: Performance vs. complexity for soft-output 4×4 MIMO detection schemes using 16-QAM transmission in slow Rayleigh fading.

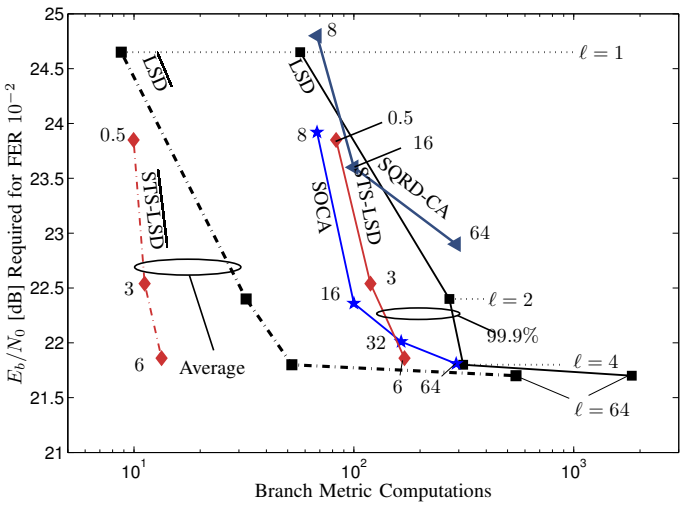

Fig. 7: Performance vs. complexity for soft-output 4×4 MIMO detection schemes using 64-QAM transmission in slow Rayleigh fading.

Fig. 6 provides a performance-complexity plot for 16-QAM transmission in slow fading, where the curves, algorithms and markers are the same as outlined previously, with LFSD results provided for $[16\ 1\ 1\ 1]_\mathbf{b}$, $[16\ 2\ 2\ 1]_\mathbf{b}$, $[16\ 2\ 2\ 2]_\mathbf{b}$ and $[16\ 4\ 2\ 2]_\mathbf{b}$. From Fig. 6 it can be observed that an increase in SNR is required for the slow-fading scenario to achieve comparable error rate performance to the fast-fading scenario. In slow fading the SOCA algorithm remains an attractive option, even though its performance-complexity profile is never superior to the average computational complexities of the LSD, LISS, or STS-LSD. However, the fixed computational complexity of the SOCA algorithm is again significantly lower than the worst-case (or bounded worst-case) computational complexity of the LSD and LISS. Finally, the 99.9th percentile computational complexity for the STS-LSD has almost the same computational complexity as the SOCA algorithm for the algorithmic realizations presented. Here, the STS-LSD employing upper bounded computational complexity is an attractive alternative to the SOCA.

Fig. 7 provides results for a 4×4 channel employing 64-QAM transmission in slow fading and is used to demonstrates the importance of the SOQR on the overall error rate performance. The solid curve with left facing triangular markers, denoted SQRD-CA, represents the SOCA algorithm except that instead of using a SOQR decomposition the algorithm employs the commonly used sorted-QR decomposition [25]. Ignoring the forced detection ordering in the first layer, the SQRD-CA and SOCA have identical computational complexities, yet the SOCA algorithm outperforms the SQRD-CA algorithm by 1.2 dB when $b_1 = 16$.

We now look at a larger 8×8 communication channel. Fig. 8 provides performance versus computational complexity results for a fast-fading 8×8 MIMO channel. The performance of the K-best algorithm with $m = 512$ is the reference performance for this system configuration. Reference is also made to the M algorithm with $m = b = 4$ and $m = b = 8$. In order to achieve a desirable performance versus computational complexity tradeoff for this larger system size, the SOCA

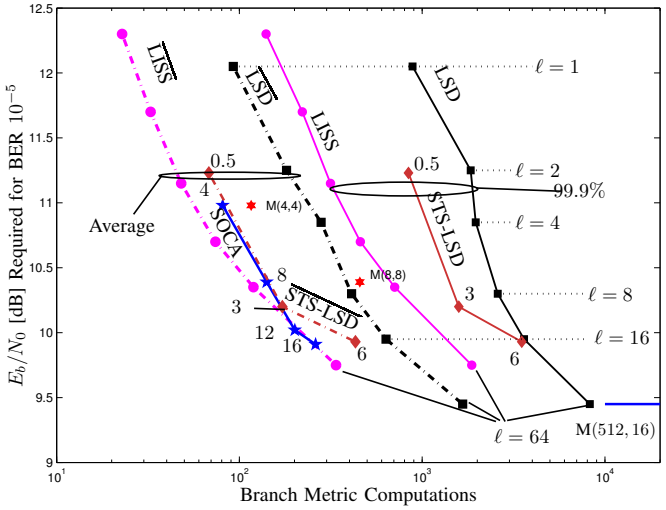

Fig. 8: Performance vs. complexity for soft-output 8×8 MIMO detection schemes using 16-QAM transmission in fast Rayleigh fading.

algorithm requires a change to $\mathbf{b}$ such that $\mathbf{b} = [b_1\ 2\ \ldots\ 2]$, where b_1 is the number of child nodes enumerated from the root of the tree. This is because the performance drops off significantly when $\mathbf{b}$ is maintained at $\mathbf{b} = [b_1\ 1\ \ldots\ 1]$. A second important change is the incorporation of tree pruning. For the results shown in Fig. 8, at each level of the tree the survivor nodes were pruned to $m_i = b_1$, i.e. the $\mathbf{m}$ vector for the SOCA algorithm was set to $\mathbf{m} = [b_1\ b_1\ \ldots\ b_1]$. This means that in Fig. 8 the corresponding value next to each marker for the SOCA represents the algorithmic realization when $b_1 = m_8 = \ell$. Without tree pruning the performance of the SOCA algorithm is slightly improved relative to the SOCA without tree pruning. However, these results are not shown because the computational complexity would increase prohibitively when tree pruning is omitted. This increase is due to the large system size which, without tree pruning, allows

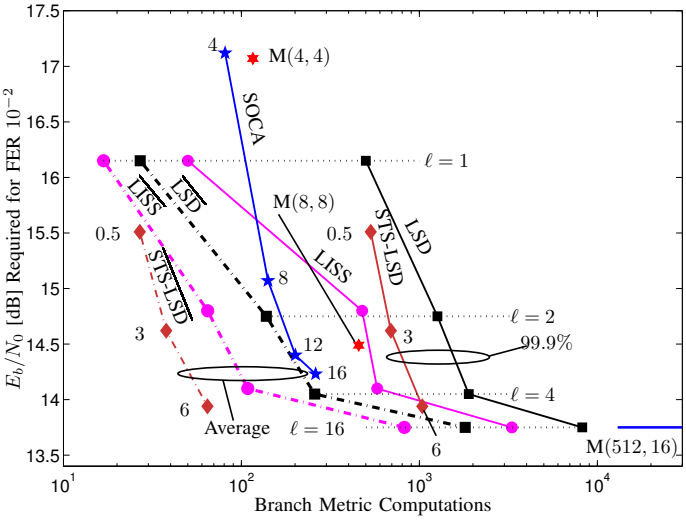

Fig. 9: Performance vs. complexity for soft-output 8×8 MIMO detection schemes using 16-QAM transmission in slow Rayleigh fading.

for extra layers of tree growth. Finally, we note that the SOCA algorithm has roughly the same performance-complexity curve as the average complexity of the STS-LSD.

Fig. 9 provides the same 8×8 16-QAM results as Fig. 8 but for the slow-fading scenario. The SOCA algorithm with $b_1 = m_8 = 16$ has roughly the same performance as the LISS with $\ell = 4$ but its computational complexity is 45% of the 99.9th percentile computational complexity. Additionally, the SOCA algorithm with $b_1 = 12$ has roughly the same performance as the M algorithm with parameterization $m = b = 8$, but with 57% of the complexity. This savings reduction come from the fact that, for layers 2 through N_t of the tree, we have a multiplier of $b_i = 2$ for the SOCA and a significantly larger $b_i = 8$ for the M algorithm. Finally, we observe that the SOCA algorithm has a fixed performance-complexity curve that sits between the average and 99.9th percentile computational complexity of the STS-LSD. Thus, even for the most challenging scenario presented (i.e. 8×8 16-QAM in slow fading) the SOCA algorithm remains a good choice for soft-output MIMO detection.

VI. Conclusion

We presented a soft-output MIMO detection algorithm that achieves near max-log optimal error rate performance with low and fixed computational complexity. The proposed algorithm combines a smart-ordered QR decomposition with candidate adding and a parallel breadth-first search of the detection tree to achieve its desirable performance-complexity tradeoff. Furthermore, the proposed algorithm visits nodes in the detection tree only once, employs a simple bit flipping to add candidates, and is able to avoid a sort-and-select operation for many practical scenarios. Results indicated that the proposed algorithm performs well in both fast and slow fading 4×4 and 8×8 MIMO channels with QAM inputs.

References

[1] E. Zimmermann and G. Fettweis, "Generalized Smart Candidate Adding for Tree Search Based MIMO Detection," in *ITG/IEEE Workshop on Smart Antennas (WSA'07)*, Vienna, Austria, Feb. 2007.

[2] E. Zimmermann, D. L. Milliner, J. R. Barry, and G. Fettweis, "A parallel smart candidate adding algorithm for soft-output MIMO detection," in *Proc. 7th International ITG Conference on Source and Channel Coding (SCC'08)*, Ulm, Germany, Jan. 2008.

[3] D. L. Milliner, E. Zimmermann, J. R. Barry, and G. Fettweis, "A Framework for Fixed Complexity Breadth-First MIMO Detection," in *10th International Symposium on Spread Spectrum Techniques and Applications (ISSSTA'08)*, Bologna, Italy, Aug. 2008, pp. 129 – 132.

[4] B. Farhang-Boroujeny, Z. Haidong, and S. Zhenning, "Markov chain Monte Carlo algorithms for CDMA and MIMO communication systems," *IEEE Transactions on Signal Processing*, vol. 54, pp. 1896–1909, May 2006.

[5] B. Steingrimsson, Z. Luo, and K. Wong, "Soft quasi-maximum likelihood detection for multipleantenna wireless channels," *IEEE Transactions on Signal Processing*, vol. 51, pp. 2710–2718, Nov. 2003.

[6] M. Nekuii and T. N. Davidson, "List-based soft demodulation of MIMO QPSK via semidefinite relaxation," in *IEEE 8th Workshop on Signal Processing Advances in Wireless Communications*, Jun. 2007, pp. 1–5.

[7] W. J. Choi, K. W. Cheong, and J. M. Cioffi, "Iterative Soft Interference Cancellation for Multiple Antenna Systems," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC00)*, vol. 1, 2000, pp. 304–309.

[8] D. Seethaler, G. Matz, and F. Hlawatsch, "Efficient Soft Demodulation in MIMO-OFDM Systems with BICM and Constant Modulus Alphabets," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 4, May 2006, pp. IV – IV.

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

[10] J. Hagenauer and C. Kuhn, "The List-Sequential (LISS) algorithm and its application," *IEEE Transactions on Communications*, vol. 55, no. 5, pp. 918–928, May 2007.

[11] P. Marsch, E. Zimmermann, and G. Fettweis, "Smart Candidate Adding: A new Low-Complexity Approach towards Near-Capacity MIMO Detection," in *13th European Signal Processing Conference (EUSIPCO'05)*, Antalya, Turkey, Sep. 2005.

[12] M. S. Yee, "Max-log-MAP sphere decoder," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'05)*, vol. 3, 18.-23. Mar. 2005, pp. 1013–1016.

[13] R. Wang and G. Giannakis, "Approaching MIMO channel capacity with Soft Detection Based on Hard Sphere Decoding," *IEEE Transactions on Communications*, vol. 54, pp. 587–590, Apr. 2006.

[14] J. Jaldén and B. Ottersten, "Parallel Implementation of a Soft Output Sphere Decoder," in *Conference Record of the 39th Asilomar Conference on Signals, Systems and Computers*, Oct. 2005, pp. 581–585.

[15] C. Studer, A. Burg, and H. Bölcskei, "Soft-output sphere decoding: Algorithms and VLSI implementation," *IEEE Journal on Selected Areas in Communications*, vol. 26, pp. 290–300, Feb. 2008.

[16] L. G. Barbero and J. S. Thompson, "Extending a Fixed-Complexity Sphere Decoder to Obtain Likelihood Information for Turbo-MIMO Systems," *IEEE Transactions on Vehicular Technology*, 2008.

[17] L. Hanzo, J. P. Woodard, and P. Robertson, "Turbo Decoding and Detection for Wireless Applications," *Proceedings of the IEEE*, vol. 95, pp. 1178 – 1200, Jun. 2007.

[18] J. Anderson and S. Mohan, "Sequential Coding Algorithms: A Survey and Cost Analysis," *IEEE Transactions on Communications*, vol. 32, pp. 169–176, Feb. 1984.

[19] L. G. Barbero and J. S. Thompson, "Fixing the Complexity of the Sphere Decoder for MIMO Detection," *IEEE Transactions on Wireless Communications*, vol. 7, pp. 2131–2142, Jun. 2008.

[20] D. E. Knuth, *The Art of Computer Programming, Vol. III: Sorting and Searching*. Reading, MA: Addison-Wesley, 1973.

[21] D. Wübben, R. Böhnke, V. Kühn, and K. D. Kammeyer, "MMSE extension of V-BLAST based on sorted QR decomposition," in *Proceedings of the IEEE Semiannual Vehicular Technology Conference (VTC2003-Fall)*, Orlando, USA, Oct. 2003, pp. 508 – 512.

[22] G. J. Foschini, G. D. Golden, R. A. Valenzuela, and P. W. Wolniansky, "Simplified processing for wireless communication at high spectral efficiency," *IEEE Journal on Selected Areas in Communications*, vol. 17, pp. 1841–1852, Nov. 1999.

[23] Y. Li and Z.-Q. Luo, "Parallel detection for V-BLAST system," in *Proc. IEEE International Conference on Communications (ICC'02)*, vol. 1, New York, NY USA, Apr. 2002, pp. 340–344.
[24] D. W. Waters and J. R. Barry, "The Chase family of Detection Algorithms for MIMO Channels," *IEEE Transactions on Signal Processing*, vol. 56, pp. 739–747, Feb. 2008.
[25] D. Wübben, R. Böhnke, V. Kühn, and K. D. Kammeyer, "Efficient algorithm for decoding layered space-time codes," *Electronic Letters*, vol. 37, pp. 1348–1350, Oct. 2001.
[26] J. Jaldén, L. G. Barbero, B. Ottersten, and J. S. Thompson, "Full Diversity Detection in MIMO Systems with a Fixed-Complexity Sphere Decoder," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'07)*, Honolulu, Hawaii, USA, Apr. 2007, pp. III–49–III–52.
[27] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Boelcskei, "VLSI Implementation of MIMO Detection Using the Sphere Decoding Algorithm," *IEEE Journal on Solid-State Circuits*, vol. 40, Jul. 2005.
[28] S. Bäro, J. Hagenauer, and M. Witzke, "Iterative detection of MIMO transmission using a list-sequential (LISS) detector," in *IEEE International Conference on Communications (ICC'03)*, vol. 4, Mar. 2003, pp. 2653 – 2657.
[29] S. Bittner, E. Zimmermann, W. Rave, and G. Fettweis, "List sequential MIMO detection: Noise bias term and partial path augmentation," in *IEEE International Conference on Communicaitons (ICC'06)*, Istanbul, Turkey, 11.-15 Jun. 2006, pp. 1300 – 1305.
[30] E. Zimmermann and G. Fettweis, "Unbiased MMSE Tree Search MIMO Detection," in *International Symposium on Wireless Personal Multimedia Communications (WPMC'06)*, San Diego, USA, Sep. 2006.
[31] C. P. Schnorr and M. Euchner, "Lattice basis reduction: Improved practical algorithms and solving subset sum problems," in *Mathematical Programming*, vol. 66, no. 1-3, Aug. 1994, pp. 181–199.
[32] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in lattice, including a complexity analysis," *Mathematics of Computation*, vol. 44, pp. 463–471, Apr. 1985.
[33] Y. X. S. Chen, T. Zhang, "Breadth-first tree search MIMO signal detector design and VLSI implementation," in *IEEE MILCOM 2005*, vol. 3, Oct. 2005, pp. 1470–1476.
[34] K. W. Wong, C. Y. Tsui, R. S. K. Cheng, and W. H. Mow, "A VLSI architecture of a K-best lattice decoding algorithm for MIMO channels," in *IEEE International Symposium on Circuits and Systems (ISCAS'02)*, vol. 3, 2002, pp. 273–276.

David L. Milliner received the S.B. and M.Eng degrees in Electrical Engineering and Computer Science from the Massachusetts Institute of Technology in 2003 and 2004, respectively and the Ph.D. degree in Electrical and Computer Engineering from the Georgia Institute of Technology in 2009. At Georgia Tech he was a Presidential Fellow recipient. He was a 2004 MIT Masterworks Finalist and a recipient of the 2001 MIT Bell Northern Research Undergraduate Laboratory Prize. His research interests are in the area of communication theory, with a focus on multiantenna systems. He is coauthor with J. R. Barry for a chapter on soft-output MIMO detection in The Digital Signal Processing Handbook (CRC Press, 2009, 2nd ed.).

Ernesto Zimmermann received his Dipl.-Ing. degree from Technische Universität Dresden in 2003, after studies at UPC Barcelona, ENST de Bretagne, and TU Dresden. He then joined the Vodafone Chair Mobile Communications Systems, focusing his research interests on information theory, channel coding, Dirty RF, cooperative relaying and receiver architectures for MIMO systems. After earning his PhD degree in 2007, he took on new responsibilities at the chair as research group leader. During this time, Dr. Zimmermann authored and co-authored more than 40 conference and journal articles, as well as several patents. He is now COO and co-founder of RadioOpt, one of the latest spin-outs of the chair.

John R. Barry received the B.S. degree in electrical engineering from the State University of New York, Buffalo, in 1986 and the M.S. and Ph.D. degrees in electrical engineering from the University of California, Berkeley, in 1987 and 1992, respectively. Since 1992, he has been with the Georgia Institute of Technology, Atlanta, where he is currently a Professor with the School of Electrical and Computer Engineering. His research interests include wireless communications, equalization, and multiuser communications. He is coauthor with E. A. Lee and D. G. Messerschmitt of Digital Communications (Norwell, MA: Kluwer, 2004, 3rd ed.) and the author of Wireless Infrared Communications (Norwell, MA: Kluwer, 1994).

Gerhard Fettweis earned his Ph.D. degree from Aachen University of Technololgy (RWTH) in 1990. From 1990 to 1991, he was Visiting Scientist at the IBM Almaden Research Center in San Jose, CA, developing signal processing innovations for IBMs disk drive products. From 1991 to 1994, he was a Scientist with TCSI Inc., Berkeley, CA, responsible for signal processor IC development projects for cellular phone chip-sets. Since 1994 he holds the Vodafone Chair at Technische Universität Dresden, Germany.

Gerhard Fettweis has (co-)authored 500 publications and more than 25 patent families. He was TPC Chair of IEEE ICC 2009 (Dresden), and has organized many other events. Among receiving other awards, as the Alcatel-Lucent Research Award, he is an IEEE Fellow. Next to producing scientific innovations, he has spun-out eight start-ups: Systemonic (now NXP and ST-NXP Wireless), Radioplan (Actix), Signalion, InCircuit, Dresden Silicon (Signalion), Freedelity, RadioOpt, and Blue Wonder Communications.

At TU Dresden he set up a team of currently 20 companies sponsoring his research, coming from Asia, Europe, and the US. In addition, an equally large set of companies have funded his Ph.D. research projects so far. A prominent project example is the German EASY-C project, researching on LTE-Advanced technology, and setting up the largest cellular testbed in downtown Dresden. In his current capacity as the Vodafone Academic Ambassador he assists Vodafone in strategic guidance of international academic research collaborations.