Generalized Smart Candidate Adding for Tree Search Based MIMO Detection

Ernesto Zimmermann and Gerhard Fettweis
Vodafone Chair Mobile Communications Systems, TU Dresden, D-01062 Dresden, Germany
zimmere@ifn.et.tu-dresden.de

***Abstract*— Tree search schemes are an efficient means of solving the detection problem in MIMO systems. There exist two fundamentally different approaches when using such techniques for soft output detection. The traditional way is to employ a single undirected search to generate a list of hypotheses on the transmit signal. Alternatively, multiple directed searches can be used – Smart Candidate Adding. This paper provides a detailed assessment of the complexity and performance of different tree search schemes, when following the single search approach. Based on these results, suitable component tree search techniques for Smart Candidate Adding are selected. It is shown that employing a breadth-first tree search scheme (more specifically, the M-Algorithm) offers several advantages over other approaches.**

I. Introduction

Future wireless communications systems will make use of multiple antennas at transmitter and receiver to increase spectral efficiency. The main challenge for such MIMO systems lies in the non-orthogonality of the transmission channel, which renders the correct separation of the transmitted data streams at the receiver a challenging task. This task can be solved effectively by using Turbo processing, i.e., exchanging probabilistic feedback (soft information) between the inner MIMO detector and the outer channel decoder. In this context, tree search based detection techniques are known to enable a performance close to channel capacity, while avoiding the prohibitive complexity of the a posteriori probability (APP) detector. Sphere [1], sequential [2] and M-algorithm based detection [3] are representative examples of such schemes.

However, the application of the Turbo principle requires the detector to generate precise information on the reliability of each of the received bits. This poses a significant challenge for the straightforward "list" extensions of the aforementioned algorithms: In order to ensure a high accuracy of the soft output, the list size has to be chosen very large – which obviously entails high detection complexity. Therefore, it was proposed in [4]–[6] to generate the soft output by using multiple instances of a Schnorr-Euchner sphere detector, each of which searches only for a single leaf node (list size 1). At first, a search for the MAP estimate is performed, followed by a set of searches for counter-hypotheses to this estimate. The term Smart Candidate Adding (SCA) has been coined for this strategy in [4]. In this contribution, we extend this proposal to other tree search based detection techniques and discuss which algorithms are best suited to achieve a favorable trade-off between performance and detection complexity.

The remainder of this paper is structured as follows: Section II discusses the system model and parameters used for performance evaluations. Section III provides an introduction to tree search based MIMO detection, as well as an assessment of the performance and complexity achievable with different schemes. This is followed by a description of Smart Candidate Adding in Section IV. Section V and VI present results for the case of non-iterative and iterative detection-decoding, respectively. We finally draw conclusions in Section VII.

II. System Model

Consider a $N_T \times N_R$ MIMO system based on a BICM transmit strategy: the vector $\mathbf{u}$ of i.i.d. information bits is encoded and interleaved. The resulting code bit stream is partitioned into blocks $\mathbf{c}$ of $N_T \cdot L$ bits and mapped onto a vector symbol $\mathbf{x}$ whose components are taken from some complex constellation $\mathcal{C}$. Here, L denotes the number of bits per symbol, allowing to distinguish between $Q = |\mathcal{C}| = 2^L$ different constellation points. We consider transmission over a flat fading channel. In the equivalent base-band model, the received signal $\mathbf{y}$ is given by:

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

where $\mathbf{H} \in \mathbb{C}^{N_R \times N_T}$ is the channel transfer matrix which is assumed to be perfectly known at the receiver. The entries of $\mathbf{H}$ are realizations of zero mean i.i.d. complex Gaussian random processes of variance 1 (passive subchannels). The average transmit energy is normalized such that $\mathcal{E}\{\mathbf{x}\mathbf{x}^{\mathrm{H}}\} = E_s/N_T\ \mathbf{I}$. The vector $\mathbf{n} \in \mathbb{C}^{N_R \times 1}$ represents the receiver noise whose components are zero mean i.i.d. complex Gaussian random variables with variance $N_0/2$ per real dimension: $\mathcal{E}\{\mathbf{n}\mathbf{n}^{\mathrm{H}}\} = N_0\ \mathbf{I}$. The signal-to-noise ratio (SNR) at each receive antenna is hence given by SNR $= E_s/N_0$.

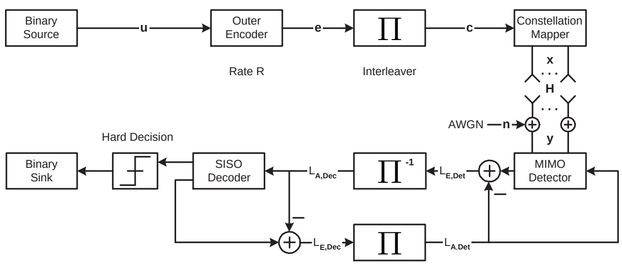

Fig. 1. System model with BICM transmitter and iterative receiver.

To ensure comparability of results, we use a setup equivalent to the one in [1], [2]: A rate 1/2 PCCC based on $(7_R, 5)$ convolutional codes is employed for transmission over a 4×4 MIMO channel which is spatially and temporally i.i.d. fading. The information block size (including tail bits) is 9216 bits. The PCCC decoder uses 8 internal iterations (logMAP decoding). In the iterative setup, 4 iterations between detector and decoder are performed.

III. Tree Search Based MIMO Detection

A. Fundamentals

The task of the detector is to calculate the a posteriori probability for each of the code bits $c_{m,l}$ in $\mathbf{x}$. Since we are dealing with binary numbers, this information is conveniently expressed in the form of log-likelihood ratios (LLRs):

$$\begin{aligned} L(c_{m,l}|\mathbf{y}) &:= \ln \frac{P\left[c_{m,l}=+1|\mathbf{y}\right]}{P\left[c_{m,l}=-1|\mathbf{y}\right]} \\ &\approx \max_{\mathbf{x}\in\mathcal{X}_{m,l}^{+1}} \left\{ \frac{-\left\|\mathbf{y}-\mathbf{H}\mathbf{x}\right\|^2}{N_0} + \sum_{i=1}^{N_T \cdot L} \ln P[c_i] \right\} \\ &\quad - \max_{\mathbf{x}\in\mathcal{X}_{m,l}^{-1}} \left\{ \dots \right\}. \end{aligned} \tag{2}$$

where the second line follows from the application of the so-called max-log approximation. Here, $\mathcal{X}_{m,l}^{\pm 1}$ denotes the set of $2^{N_T \cdot L-1}$ symbols $\mathbf{x} \in \mathcal{X}$ for which $c_{m,l} = \pm 1$. Evaluating (2) by a brute-force approach (maxLogAPP detection) is well known to require an effort growing exponentially in the number of transmitted bits per vector symbol. However, only a few hypotheses in $\mathcal{X}_{m,l}^{\pm 1}$ actually maximize each of the respective terms in (2). Several close-to-optimal detection strategies therefore construct a subset list $\mathcal{L} \subset \mathcal{X}$ from which the LLRs are determined. The subset should on the one hand include only a fraction of the elements from $\mathcal{X}$ to minimize complexity. On the other hand, it should be large enough to allow approaching the true detector LLRs as closely as possible, to maximize performance. Let the size of the list $\mathcal{L}$ be denoted as $M = |\mathcal{L}|$. Tree search based MIMO detection techniques construct $\mathcal{L}$ using a back-substitution approach. After a QR-decomposition of $\mathbf{H}$, the LLRs can be determined using the per-antenna metric increments Λ_m:

$$L(c_{m,l}|\mathbf{y}) \approx \max_{\mathbf{x}\in\mathcal{L}\cap\mathcal{X}_{m,l}^{+1}} \left\{ \sum_{1}^{N_T} \Lambda_m \right\} - \max_{\mathbf{x}\in\mathcal{L}\cap\mathcal{X}_{m,l}^{-1}} \left\{ \sum_{1}^{N_T} \Lambda_m \right\}$$

which are referred to as *branch metrics* and are given by

$$\Lambda_m = -\frac{1}{N_0} \left\| \tilde{y}_m - \sum_{j=m}^{N_T} r_{m,j} x_j \right\|^2 + \sum_{l=1}^{L} \ln \Pr[c_{m,l}] \tag{3}$$

with $\tilde{\mathbf{y}} = \mathbf{Q}^{\mathrm{H}}\mathbf{y}$. The detector starts in layer $n = N_T$ and works its way up until layer $n = 1$ is reached. For each branch in the tree, Q different choices are possible for the signal estimate x_m. The detection process can hence be interpreted as a search for leaf nodes in a tree structure. Different types of tree search based detectors can be implemented by using the *path metrics* $\sum_{m=n}^{N_T} \Lambda_m$ to control which tree nodes are added to the working stack and in which order.

A major problem for all these schemes are missing counter-hypotheses: whenever $\mathcal{L} \cap \mathcal{X}_{m,l}^{\pm 1} = \emptyset$, the magnitude of the LLR for the corresponding bit cannot be determined from the entries of $\mathcal{L}$. The standard way of addressing this issue is to simply clip the magnitude of the soft output to a certain predefined value [1]. However, the performance of the system is very sensitive to the choice of the clipping level, especially for smaller list sizes (see [3] and Section VI).

B. Classification of Tree Search Strategies

Tree search algorithms have been the subject of extensive study already in the 1960ies, in the context of sequential decoding. Based on the framework presented in [7], we discern the following three representative classes of algorithms:

- *Depth-first search* is a scheme which only considers a single tree node at a time. This node is extended until its path metric falls below a given threshold, in which case the algorithm back-tracks and extends the tree in a different direction. The sphere detector [1], [8] is an instance of this approach. The challenge lies in finding an appropriate value for the threshold (or *sphere radius*). A very attractive solution is to start with an extreme value and successively refine the threshold during the tree search, based on the path metrics of found leaf nodes [9].
- *Metric-first search* keeps track of a number of nodes simultaneously, and always extends the node which has currently the largest path metric. The list sequential (LISS) detector [2] implements this strategy. The very high storage requirements are the main disadvantage of this technique. As soon as the number of considered paths exceeds the size of the working stack, the paths with the smallest metrics have to be dropped [2], or the search must be stopped and the LISS declares an erasure [10].
- *Breadth-first search* extends the tree layer-by-layer. At each depth, the M nodes with the largest path metrics are retained and all other nodes are dropped. The classical example for this approach is the M-Algorithm [3]. The advantage of this technique is the fixed detection complexity. However, the achievable performance is limited by error propagation, particularly for low values of M.

Note that the first two schemes have a variable complexity, which might be undesirable from an implementation perspective. Furthermore, both the average and (the potentially very high) worst case complexity depend on the operating SNR.

C. Preprocessing and Enumeration Aspects

Layer ordering and MMSE preprocessing can be used to improve the performance-complexity trade-off achievable with tree search schemes. More specifically, they will enhance performance for schemes with fixed or (tightly) upper bounded complexity [11], and reduce complexity for schemes with variable complexity [9], [12]. However, the use of MMSE preprocessing introduces a bias on the calculated metrics, which should be removed to maximize performance [11].

In order to facilitate the use of efficient enumeration strategies (see below), the tree search should be performed on a real-valued system model of doubled dimensions and a real-valued constellation of size $\sqrt{Q}$ (see e.g. [9] for details).

For the case of higher order modulation, a further substantial complexity reduction is enabled by using a Schnorr-Euchner enumeration strategy [8]: the child nodes of a parent node are generated in descending order of their branch metrics Λ_m. While this technique has been mainly studied in the context of sphere detection, it is readily applicable to all tree search schemes. In fact, it was shown in [13] that the Schnorr-Euchner LISS detector without length bias term visits the least number of tree nodes among all optimal search algorithms.

For the M-Algorithm, the Schnorr-Euchner strategy can be pragmatically implemented as follows: First, for each of the M kept nodes, the child node with the largest metric is generated, resulting in a total of M new nodes. A threshold is set to the Mth largest path metric. Subsequently, the next M child nodes are generated. For each parent node where the metric of the new child node is below the threshold, the extension process can be stopped. The threshold is updated and the process continues until no further child nodes need to be generated. The achieved complexity reduction factor is around $\sqrt{Q}/3$ for 64-QAM transmission, compared to $\sqrt{Q}/2$ for the LISS and the sphere detector. Note, however, that as soon as a priori information has to be incorporated into the branch metrics, implementing a Schnorr-Euchner enumeration strategy requires the explicit calculation of the metrics for all children of a parent node and subsequent sorting.

D. Comparison of Techniques

Results for the performance of different tree search techniques are provided in Figure 2 for a non-iterative detection-decoding setup. All schemes employ Schnorr-Euchner enumeration and a sorted QR decomposition [14] for preprocessing.

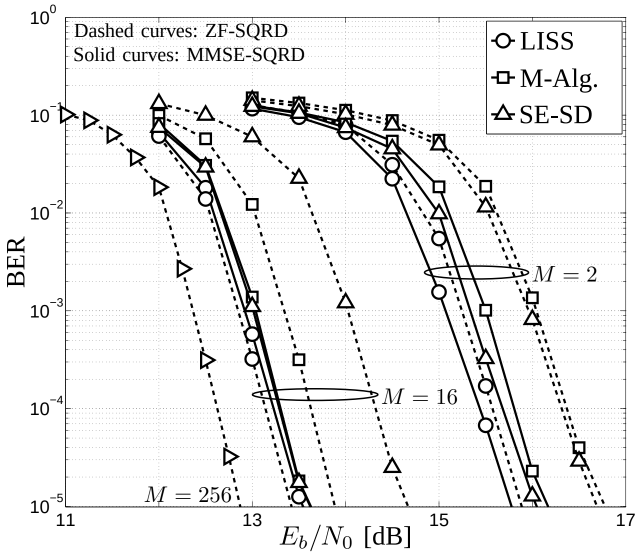

Fig. 2. Performance of tree search detection (4×4 MIMO, 64-QAM). Dashed curves: ZF preprocessing, solid curves: MMSE preprocessing.

The complexity of the Schnorr-Euchner sphere detector (SE-SD) and the LISS has been upper bounded to double the complexity of an M-Algorithm which generates a list of the same size M. This bound was chosen based on the analysis of the distribution of the SE-SD and LISS complexity for the case of hard output detection ($M = 1$, results not shown). It has been found to yield reasonable performance for a 4×4 MIMO setup with constellation sizes from 4-QAM up to 64-QAM.

For a list size of $M = 2$, the performance of all three investigated tree search schemes is enhanced by using MMSE preprocessing. The best performance is achieved by the LISS detector. For larger list sizes ($M = 16$), the gain from using MMSE preprocessing decreases for both the LISS and the M-Algorithm (this has also been noted in [11]). For the LISS, there is even a slight loss w.r.t. the ZF case. This is due to the relatively low worst case complexity of the LISS for the case of unbounded complexity. The imposed upper bound on the number of branch metric computations has therefore almost no impact and loss in performance due to the sub-optimality introduced by MMSE preprocessing [9] is the dominating effect. For the sphere detector, however, the gains from using MMSE preprocessing are higher for larger list sizes. This is a direct result of the radius determination strategy: The sphere radius can only be fixed once M leaf nodes have been found. This results in a high worst case complexity for large values of M, if the initially found leaf nodes are far away from the MAP estimate (as is the case for ZF preprocessing). It can also be seen that the performance of all investigated schemes is very similar for $M = 16$, if MMSE-SQRD preprocessing is employed.

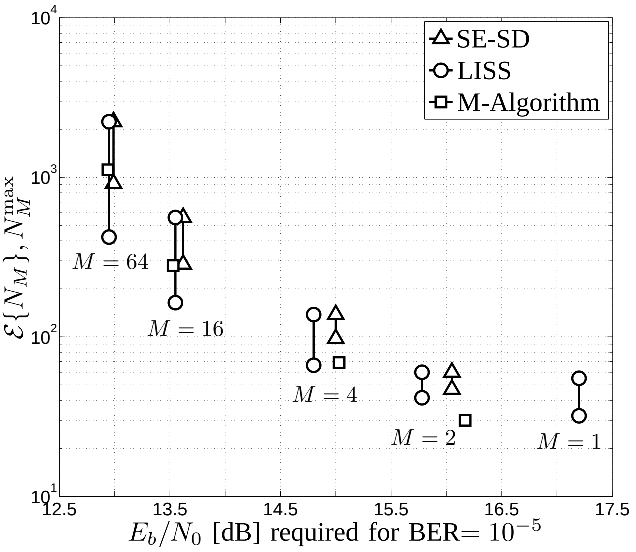

Fig. 3. Performance-complexity trade-off for different tree search techniques (4×4 MIMO, 64-QAM). All schemes use MMSE-SQRD preprocessing.

Figure 3 shows the performance and complexity (in terms of the required number of branch metric computations N_M) of the three investigated tree search schemes. For the LISS and the sphere detector, the upper markers indicate the imposed upper bound (i.e., the worst-case complexity $N_M^{\max}$). The lower markers indicate the average complexity $\mathcal{E}\{N_M\}$. Consistent with the results from [13], the complexity of the LISS is in general lower than that of the SE-SD. The complexity reduction enabled by using MMSE preprocessing is around 20% (results for the ZF case not shown). Note that the average complexity of all schemes is typically within a factor two of each other, with some advantages for the M-Algorithm for very small list sizes, and the LISS for large list sizes. In light of the high storage and memory access requirements of the LISS, the sphere detector and the M-Algorithm emerge as the most promising techniques from an implementation perspective.

Observe that for the high raw BERs (around 10%) at which powerful coding schemes operate, the average complexity of the sphere detector is comparable to that of the M-Algorithm. The often claimed lower average complexity of the sphere detector does only hold in the high SNR regime, at target BERs which might be uninteresting for practical applications. Furthermore, the M-Algorithm is typically run without Schnorr-Euchner enumeration, which renders the comparison unfair in favor of Schnorr-Euchner sphere detection.

IV. Smart Candidate Adding

A. Fundamentals

From (2) it is easily seen that the LLRs at the output of the maxLogAPP detector may also be written in the form:

$$L(c_{m,l}|\mathbf{y}) = c_{m,l}^{\text{MAP}} \left(\left\{ \frac{-\left\|\mathbf{y}-\mathbf{H}\mathbf{x}^{\text{MAP}}\right\|^2}{N_0} + \sum_{i=1}^{N_T \cdot L} \ln P[c_i^{\text{MAP}}] \right\} - \max_{\mathbf{x}\in\mathcal{X}_{m,l}^{-\text{MAP}}} \left\{ \frac{-\left\|\mathbf{y}-\mathbf{H}\mathbf{x}\right\|^2}{N_0} + \sum_{i=1}^{N_T \cdot L} \ln P[c_i] \right\} \right). \quad (4)$$

with $\mathbf{x}^{\text{MAP}}$ as the hypothesis which maximizes the a posteriori probability (the *MAP estimate*), $\mathbf{c}^{\text{MAP}}$ the corresponding bit pattern and $\mathcal{X}_{m,l}^{-\text{MAP}}$ the set of potential counter-hypotheses, for which $c_{m,l} = -c_{m,l}^{\text{MAP}}$. The maxLogAPP detection problem may hence be solved by first finding the MAP estimate and then performing $N_T \cdot L$ searches which cover only a subset of the transmitter signal set. This fact has already been observed in [15] in the context of a semi-definite relaxation approach. A direct implementation of (4) has been proposed in [5], [6], employing a radius-based Fincke-Pohst sphere detector [8] to determine the MAP estimate and the counter-hypotheses. However, this approach faces the problem of choosing an appropriate value for the sphere radius. Furthermore, the work concentrated on the case of QPSK transmission in a 4×4 MIMO setup, where the MaxLogAPP detection problem has still manageable complexity (see also results in Section VI).

In [4], it was proposed to use a Schnorr-Euchner sphere detector for the searches, thus avoiding the radius determination problem. The technique was also applied to higher order modulation (16- and 64-QAM), where it became evident that some bounds on the number of visited nodes have to be imposed in order to avoid unreasonably high detection complexity. This is a first hint that the sphere detector may not be best suited as tree search scheme for the SCA approach.

A beneficial "side effect" of Smart Candidate Adding is that it entirely avoids the problem of missing counter-hypotheses. However, bounding the tree search complexity may still lead to overestimated LLR magnitudes, which would necessitate the use of LLR clipping. Fortunately, the performance of reduced complexity Smart Candidate Adding has been found to be very robust to the choice of the LLR clipping level.

B. Choice of Component Techniques

In principle, any combination of the techniques introduced in Section III is possible for use in the first and the subsequent search stages of the SCA approach. From the available options, the following assignment of techniques is expected to achieve a favorable trade-off between performance and complexity:

- The *search for the MAP estimate* will cover the whole signal set $\mathcal{X}$ and should be done such that errors in the hard output of the MIMO detector are avoided, i.e., the MAP estimate has to be found with high probability. This can be assured by using either a sphere or LISS detector with upper bounded complexity, or an M-Algorithm with large enough list size.
- Each single *search for a counter-hypothesis* will cover only a constrained signal set $\mathcal{X}_{m,l}^{-\text{MAP}}$. This is the computationally most expensive part, since the involved effort scales with the number of transmitted bits per vector symbol. The use of a LISS detector is less attractive for solving this task, due to its high storage requirements and the fact that the algorithm may produce erasures if a too strict upper bound on the complexity is imposed.

Motivated by the above arguments, the focus in the subsequent investigations will be on using a Schnorr-Euchner sphere detector or an M-Algorithm for the tree search. It is evidently possible to obtain comparable results by using a LISS detector in the first search stage. The difference in complexity can be determined based on the results presented in Section III.

V. Non-iterative Detection-Decoding

Consider first the case of Schnorr-Euchner sphere detection (this is the original SCA proposal). Figure 4 provides results for the complexity of the search for counter-hypotheses. The challenges faced by this approach are clearly visible: while for some bits, the average tree search complexity is still acceptable, it is extremely high for others. The peaks in the complexity distribution corresponds to the most reliable bits in each layer (Gray mapping is used). This behavior is expected, as a high reliability is synonymous to a large distance of the counter-hypothesis to the MAP estimate, and thus a high number of nodes which have to be visited. The total average complexity of this scheme is on the order of 10^4 branch metric computations – a factor 5-10 higher than a standard sphere detector with list size $M = 64$ which achieves comparable performance (cf. Figures 3 and 5).

In order for the SE-SD based SCA approach to become competitive, it is thus necessary to accept some inaccuracy

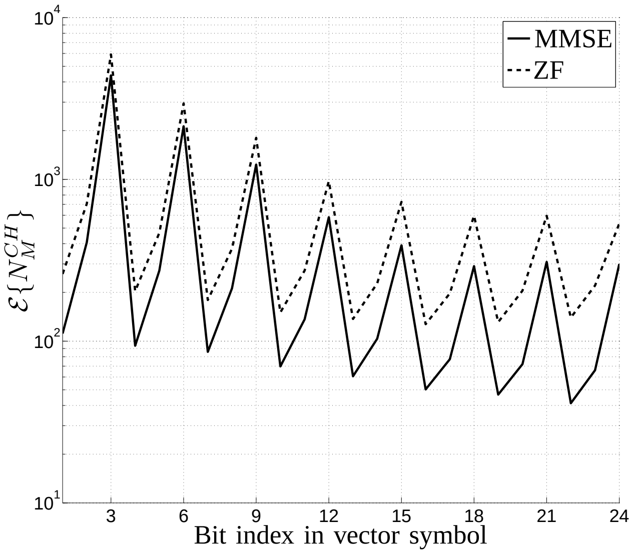

Fig. 4. Average complexity of the search for counter-hypotheses for SCA based on a Schnorr-Euchner sphere detector (4×4 MIMO, 64-QAM).

in the soft output and upper bound the complexity of the tree searches. In the following, the term N_M^{MAP} refers to the number of branch metric computations in the search for the MAP estimate while N_M^{CH} is the complexity of each of the individual searches for a counter-hypothesis. Performance results for such a setup are presented in Figure 5. A maximum of $N_M^{CH} = 64$ branch metric computations for the second stage searches suffices to achieve performance within 0.25dB of MaxLogAPP detection (results for $N_M^{MAP}, N_M^{CH} \rightarrow \infty$). It can also be seen that investing only $N_M^{MAP} = 30$ branch metric computations is enough to find the MAP estimate with high probability – increasing the complexity of the first search stage to $N_M^{MAP} = 90$ yields hardly any gain.

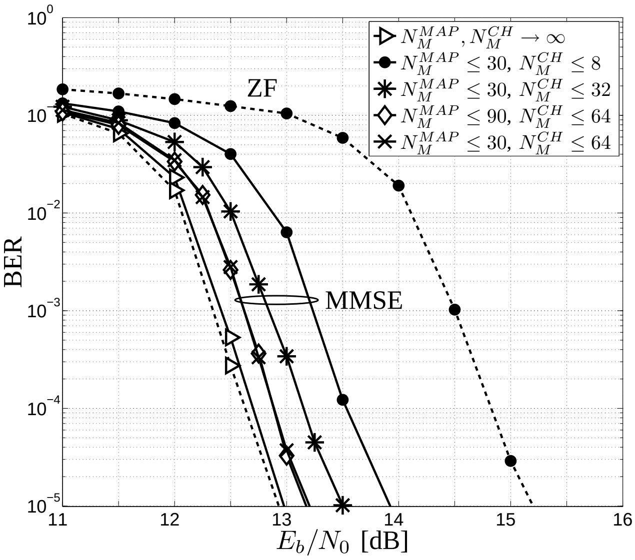

Fig. 5. Performance of sphere detection based Smart Candidate Adding (4×4 MIMO, 64-QAM). Dashed curves: ZF-SQRD, solid curves: MMSE-SQRD.

The minimum complexity setup with $N_M^{MAP} = 30$ and $N_M^{CH} = 8$ still achieves a performance within 1dB of the MaxLogAPP detector. Note that with $N_M^{CH} = 2N_T = 8$, the searches for the counter-hypotheses will only find the constrained SIC solution (or the specific Babai point, hence the term SCA-Babai used in [4]). The distinct advantage of this configuration is that imposing constraints on the value of a certain bit in layer m only affects decisions in layers which are detected later in layers $n < m$. Some of the calculated branch metrics may thus be re-used for several of the second stage searches. This is illustrated by the results in Figure 6 (diamond markers): the complexity of the SCA-SE-SD with $N_M^{CH} = 8$ is only around 150 branch metric computations, compared to around 200 which one would expect (the configuration with $N_M^{CH} = 32$ requires around 800 branch metric computations).

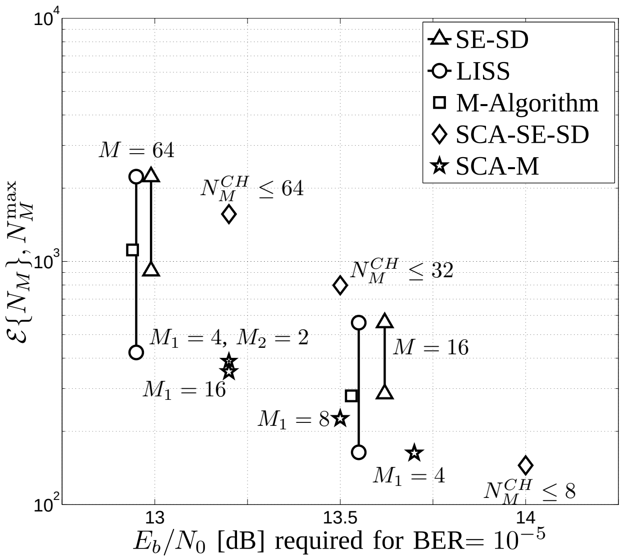

Fig. 6. Performance-complexity trade-off for conventional and SCA based tree search detection (4×4 MIMO, 64-QAM, MMSE-SQRD preprocessing).

Reducing the complexity of the search for counter-hypotheses is thus crucial to achieving a good performance-complexity trade-off. In this respect, using the M-Algorithm offers several advantages over the sphere detection based approach. Firstly, it enables to further increase the re-use factor between the first and the second tree search stage (we use the notation $M_1 = a, M_2 = b$ for the two employed list sizes in the following). In order to reduce error propagation effects, the search for the MAP estimate has to be performed with a medium value of M (say, $M_1 = 4$). In contrast to the sphere detector case, a number of counter-hypotheses will hence already be available and the second stage searches have only to be performed for a subset of the bits. It might be argued that the SCA-SE-SD approach may also use a value of $M > 1$ in the first search stage. However, we have seen that the sphere detector is unattractive for generating lists of small size. Furthermore, due to the layer-by-layer operation of the M-Algorithm, the above stated reuse strategy is also applicable in the second search stage, in contrast to the sphere detector which descends and ascends the tree structure as necessary. Finally, the detection complexity is fixed, which might be advantageous for real-time implementation.

The achieved complexity-performance trade-off is illustrated in Figure 6 ($M_2 = 1$ except where indicated). It can clearly be seen that the M-Algorithm based SCA approach (SCA-M) generally requires much less complexity to achieve performance comparable to the SCA-SE-SD approach (a factor 3-4 reduction in complexity is possible). However, it is also apparent that the required effort is not substantially lower than that of conventional tree search schemes (cf. e.g. the setup with $M_1 = 8, M_2 = 1$ compared to the M-Algorithm with $M = 16$). The main advantages of the SCA approach are hence the lower storage space requirements due to the smaller list sizes, plus the high potential for parallelization of the second stage tree searches.

VI. Iterative Detection-Decoding

Consider now the case of an iterative MIMO receiver. Four detector-decoder iterations are performed. Figure 7 illustrates the sensitivity of conventional tree search schemes to the choice of the LLR clipping level. The loss resulting from choosing the clipping level too high ($L_{clip} = 3/3.5/5$ for $M = 2/4/64$, respectively) is roughly 0.5dB for smaller list sizes, thus potentially setting off a large part of the gain obtained by increasing the list size[1]. The clipping levels which maximized performance were found to be $L_{clip} = 2/2/4$ for $M = 2/4/64$, respectively.

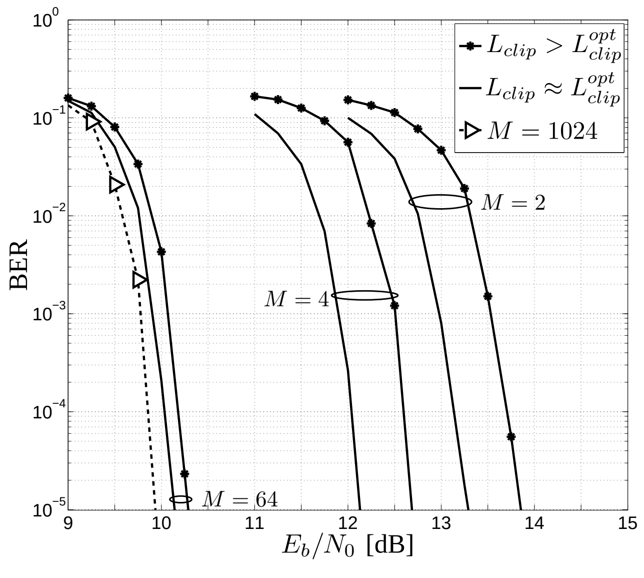

Fig. 7. Impact of LLR clipping level on the performance of M-Algorithm based detection (4×4 MIMO, 64-QAM, MMSE-SQRD preprocessing).

The performance of SCA is largely invariant to the choice of the clipping level (results not shown). It must of course be set high enough to avoid a decrease in mutual information. A value of $L_{clip} = 5$ was found to yield satisfactory performance (a higher value did not improve performance significantly). For the case of iterative detection-decoding, the tree search complexity is in general higher than for the non-iterative case, due to the fact that efficient node enumeration strategies cannot be used (cf. the M-Algorithm results with $M = 2$ in Figure 3 and Figure 8; the increase is roughly $\sqrt{Q}/2 = 4$).

For M-Algorithm based Smart Candidate Adding, however, this inefficiency can be exploited to the scheme's advantage. The initial search for the MAP estimate generates child nodes with all possible $\sqrt{Q}$ bit combinations at each layer (note that a real-valued system model is employed). Since for the considered case of an M-Algorithm, constraints on the value of a certain bit do not propagate towards the tree root, the second stage searches can be directly started from those "dead-ends" of the initial search for the MAP estimate.

Figure 8 shows performance-complexity results for the case of 64-QAM transmission. Results for conventional tree search schemes are plotted as reference. For the LISS, the fixed complexity configuration from [10] was employed: the tree search was run until the number of paths reached the stack size. Furthermore, the noise bias term proposed in [16] was used. The general trends are comparable to the non-iterative case: the average complexity of all conventional tree search schemes is within a factor of two of each other, if comparable performance is to be achieved. The SCA-M approach allows to achieve performance very close to MaxLogAPP detection already when using $M_1 = 4$ and $M_2 = 1$. The complexity is roughly the same as that of an M-Algorithm or a LISS achieving similar performance. Note that the setting with $M_1 = 16$ and $M_2 = 1$ allows to achieve the same performance as a conventional M-Algorithm with $M = 64$, but requires only half the number of branch metric computations. The complexity of the search for counter-hypotheses is still substantial, such that the application of the SCA approach is mainly attractive if close-to-optimal performance is targeted.

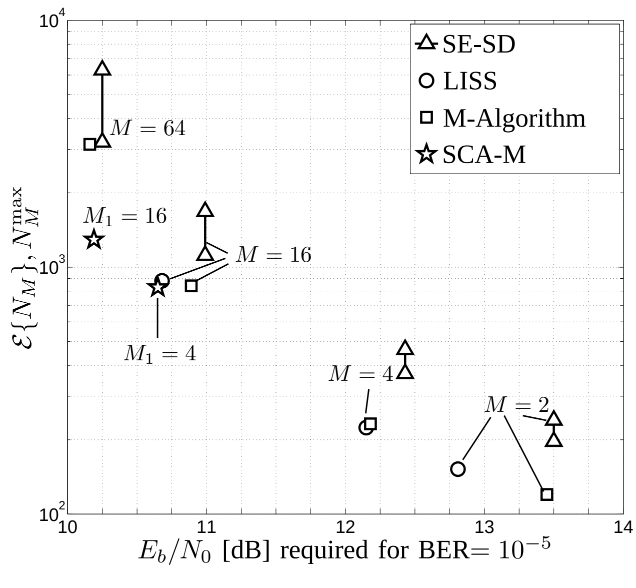

Fig. 8. Performance-complexity trade-off for conventional and SCA based tree search detection (4×4 MIMO, 64-QAM, MMSE-SQRD preprocessing).

[1]Note that a constant clipping level of 8 has been used in [1], vs. a constant level of 3 in [3]. Both solutions are clearly suboptimal.

Results for the case of 4-QAM transmission are provided in Figure 9 for the sake of completeness. Note that the given complexity figures for the conventional tree search schemes will be comparable also for the non-iterative case, as all schemes enumerate (at least) two child nodes per parent, even when using Schnorr-Euchner enumeration. Again, the average complexity of all investigated schemes is in the same order of magnitude. The LISS with fixed complexity and noise bias term is a very attractive solution, if only the required number of branch metric computations is considered. Smart Candidate Adding based on the M-Algorithm essentially achieves MaxLogAPP performance already when using $M_1 = 2$ and $M_2 = 1$. The complexity is roughly 50% lower than that of the LISS which achieves the same performance. Observe that the complexity reduction compared to a MaxLogAPP detector is "only" on the order of factor 5. In light of the additional overheads involved in the presented tree search schemes, a brute force approach might still be attractive for a practical implementation, see [17].

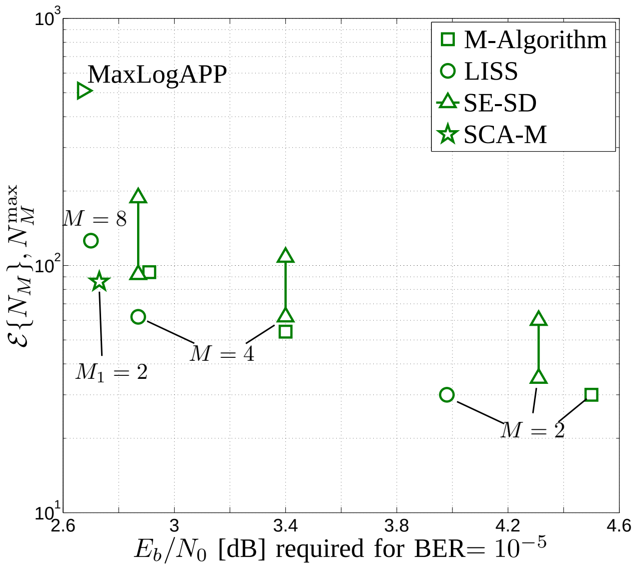

Fig. 9. Performance-complexity trade-off for conventional and SCA based tree search detection (4×4 MIMO, 4-QAM, MMSE-SQRD preprocessing).

VII. Conclusions

In this contribution, we showed that Smart Candidate Adding is a flexible and efficient way of achieving near-capacity performance in MIMO systems. Performance and complexity results were provided for SCA based on the M-Algorithm and a Schnorr-Euchner sphere detector. Based on the obtained results, M-Algorithm based tree search detection is judged to be the most attractive solution for a practical implementation of the Smart Candidate Adding approach. Compared to conventional tree search schemes, the complexity in terms of the number of branch metric computations can be reduced by a factor of 1.5-2 in an iterative detection-decoding setup. Furthermore, storage space requirements are smaller, and the search for counter-hypotheses offers a high potential for parallelization. Regarding the relative merits of the "list" versions of the sphere detector, the LISS, and the M-Algorithm, it was shown that the average complexity of all schemes is typically within a factor of two of each other (for the considered 4×4 MIMO setup; appropriate upper bounds on the tree search complexity have to be imposed for the LISS and the sphere detector). Due to the lower storage requirements and sorting effort, the M-Algorithm and the sphere detector appear to be the most attractive techniques for a real-time implementation.

References

[1] 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.

[2] S. Baero, J. Hagenauer, and M. Witzke, "Iterative detection of MIMO transmission using a list-sequential (LISS) detector," in *Proceedings of the IEEE International Conference on Communications (ICC'03)*, vol. 4, 11.-15. May 2003, pp. 2653– 2657.

[3] S. Haykin, M. Sellathurai, Y. de Jong, and T. Willink, "Turbo-MIMO for wireless communications," *IEEE Communications Magazine*, vol. 42, pp. 48– 53, Oct. 2004.

[4] 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, 04.-08. Sept. 2005.

[5] 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.

[6] 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.

[7] J. Anderson and S. Mohan, "Sequential coding algorithms: A survey and cost analysis," *IEEE Transactions on Communications*, vol. 32, pp. 169–176, Feb. 1984.

[8] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closest point search in lattices," *IEEE Transactions on Information Theory*, vol. 48, pp. 2201–2214, Aug. 2002.

[9] M. Damen, H. E. Gamal, and G. Caire, "On maximum-likelihood detection and the search for the closest lattice point," *IEEE Transactions on Information Theory*, vol. 49, pp. 2389–2402, Oct. 2003.

[10] J. Hagenauer and C. Kuhn, "The List Sequential (LISS) Algorithm and Its Application," *IEEE Transactions on Communications*, 2006, accepted for publication.

[11] E. Zimmermann and G. Fettweis, "Unbiased MMSE Tree Search MIMO Detection," in *International Symposium on Wireless Personal Multimedia Communications (WPMC'06)*, San Diego, USA, 17.-20. Sept. 2006.

[12] E. Zimmermann, W. Rave, and G. Fettweis, "On the Complexity of Sphere Decoding," in *7th International Symposium on Wireless Personal Multimedia Communications (WPMC'04)*, Abano Terme, Italy, 12.-15. Sept. 2004.

[13] W. Xu, Y. Wang, Z. Zhou, and J. Wang, "A computationally efficient exact ML sphere decoder," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM'04)*, vol. 4, Dec. 2004, pp. 2594–2598.

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

[15] B. Steingrimsson, Z.-Q. Luo, and K. M. Wong, "Soft quasi-maximum-likelihood detection for multiple-antenna wireless channels," *IEEE Transactions on Signal Processing*, vol. 51, pp. 2710–2719, Nov. 2003.

[16] 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 Communications (ICC'06)*, Istanbul, Turkey, 11.-15. June 2006.

[17] D. Garrett, L. Davis, S. ten Brink, and B. Hochwald, "APP processing for high performance MIMO systems," in *Proceedings of the IEEE Custom Integrated Circuits Conference (CICC'03)*, 21.-24. Sept. 2003, pp. 271– 274.