Hierarchical Tree Search for Runtime-Constrained Soft-Output MIMO Detection

Kyung Jun Choi, *Student Member, IEEE*, and
Kwang Soon Kim, *Senior Member, IEEE*

***Abstract*—In this paper, a novel low-complexity tree-search (TS) algorithm, which is referred to as a hierarchical tree search (HTS), is proposed for soft-output multiple-input–multiple-output (MIMO) detection to mitigate the performance loss caused by an early termination. The proposed HTS algorithm is realized by the following two components: the hierarchical set partitioning to find all hypotheses with reasonable quality as fast as possible and to fairly improve them and the new cost metric to determine the search order by considering the quality of the hypotheses found so far. Through simulation, it is shown that the performance–complexity tradeoff of the proposed HTS algorithm surpasses those of the existing algorithms in various MIMO configurations.**

***Index Terms*—Multiple-input–multiple-output (MIMO) detection, soft-output MIMO detection, tree-search (TS) algorithm.**

I. Introduction

Recently, due to the demands for both higher spectral efficiency and reliability, multiple-input–multiple-output (MIMO) systems incorporating with error-correcting codes (ECCs) have been adopted in wireless standards, such as IEEE 802.11n, IEEE 802.16e, and Third-Generation Partnership Project Long-Term Evolution, and it has been shown that a soft-output MIMO detector can dramatically improve the error performance compared with a hard-output MIMO detector [1]. Because the *maximum a posteriori* (MAP) detector for soft-output MIMO detection requires tremendous complexity, tree-search (TS) algorithms have been widely considered for practical soft-output MIMO receivers [2], which can fall into one of the following three categories: 1) a depth-first search (DFS); 2) a breadth-first search (BFS); and 3) a metric-first search (MFS). The K-best [3] and M-algorithm [4] belong to BFS, which expands the K best nodes in each level of the tree by forward-only directional search. Although BFS has fixed complexity and memory requirements so that it can be implemented by a parallel pipelined structure [3], it suffers from a poor performance–complexity tradeoff. Sphere decoding (SD) [5] and its variants, including list SD (LSD) [1] and single TS (STS) [6], belong to DFS, in which a bidirectional search is performed within a predetermined sphere. It is known that DFS can provide near-optimal performance with reasonable average complexity [6]. Dijkstra's algorithm [7] and its variants [8] belong to MFS, in which the search order is determined by the metric of nodes so that it can minimize the average complexity. However, although DFS or MFS requires relatively low average complexity, the worst-case complexity is the same as that of an exhaustive search. In a practical system, the maximum allowed runtime is typically constrained to guarantee a constant throughput, and such algorithms may suffer from a severe performance loss caused by an early termination [9], [10].

There have been a number of approaches to mitigate such a performance loss. One approach is to make the complexity fixed by searching the nodes with a predetermined order, including fixed versions of SD [11], [12]. However, the performance becomes poor because a substantial amount of complexity is wasted due to a fixed search order regardless of the channel and noise realizations. More reasonable approach is to design a TS algorithm as robust to an early termination as possible by adaptively adjusting the search order or the search space. In [13], a multistack (MS) algorithm was proposed to remedy the disadvantages of MFS by considering the level of nodes. In [14], a probabilistic SD (P-SD) algorithm was proposed to remedy the disadvantages of DFS, in which the number of nodes within a sphere is kept constant by adaptively changing the radius. In [15], a modified best-first with fast descent (MBF-FD) algorithm was proposed, in which two strategies, namely, DFS and MFS, are used in turn. However, such algorithms focus only on finding reliable leaf nodes without considering the quality of the hypotheses found so far. As a result, the constrained runtime may be wasted to find less important nodes, and the quality of each *tentative hypothesis*[1] may be highly deviated when an early termination occurs.

In this paper, a novel low-complexity TS algorithm, which is referred to as a hierarchical tree search (HTS), is proposed for soft-output MIMO detection to reduce the performance loss caused by an early termination due to a runtime constraint. The goal of the proposed HTS is to find all hypotheses with reasonable quality from an early stage of the algorithm and to improve them fairly during the constrained runtime. We first divide the whole nodes in the tree into disjoint sets, called *hierarchical sets* (HSs), and utilize them sequentially as the candidate node set to find all hypotheses with reasonable quality as fast as possible and then fairly improve the quality of each hypothesis. In addition, a new cost metric is proposed to determine the search order so that the search for poor hypothesis is prioritized. Thus, the proposed HTS can provide fairly good quality for each hypothesis at any stage of the TS so that it is expected to provide a uniformly better performance–complexity tradeoff compared with the conventional ones.

Notation: Matrices and vectors are set in boldface uppercase and lowercase characters, respectively. Sets are denoted by mathematical script (e.g., $\mathcal{A}$). Superscript $(\cdot)^T$ and $(\cdot)^H$ denote transpose and conjugate transpose, respectively, and $|\mathcal{A}|$ stands for the cardinality of a set $\mathcal{A}$.

II. Soft-Output Multiple-Input–Multiple-Output Detection and Tree-Search Algorithms

A. Soft-Output MIMO Detection

Consider a coded MIMO system with N_T transmit and $N_R(\geq N_T)$ receive antennas. A sequence of binary information bits is encoded by an ECC and interleaved. The B-coded bits are mapped into a symbol among the alphabet $\mathcal{O}$ with $|\mathcal{O}| = Q = 2^B$. Denoting $\mathbf{s}^{\mathrm{TX}} = [s_1^{\mathrm{TX}}, \ldots, s_{N_T}^{\mathrm{TX}}]^T$ as the transmitted symbol vector, the received signal vector can be written as

$$\mathbf{y} = \mathbf{H}\mathbf{s}^{\mathrm{TX}} + \mathbf{n} \tag{1}$$

Manuscript received February 13, 2012; revised October 30, 2012; accepted November 4, 2012. Date of publication November 15, 2012; date of current version February 12, 2013. This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea funded by the Ministry of Education, Science, and Technology under Grant 2012-0001571 and in part by the Korea Communications Commission under Grant KCA-2012-12-911-04-004. The review of this paper was coordinated by Prof. H.-F. Lu.

The authors are with the Department of Electrical and Electronic Engineering, Yonsei University, Seoul 120-749, Korea (e-mail: kyungjun.choi@yonsei.ac.kr; ks.kim@yonsei.ac.kr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2012.2227868

[1]The currently updated version of the hypothesis is called the tentative hypothesis.

where $\mathbf{H} \in \mathbb{C}^{N_R \times N_T}$ is the channel matrix, and $\mathbf{n} \in \mathbb{C}^{N_R \times 1}$ is the noise vector, which is composed of independent and identically distributed (i.i.d.) $\mathcal{CN}(0,1)$ and $\mathcal{CN}(0, 2\sigma_n^2)$, respectively.

The optimal MAP detector computes *a posteriori* log-likelihood ratio (LLR) for each bit. By applying the max-log approximation [1], the max-log LLR can be written as

$$\mathrm{LLR}_{j,b} = \frac{1}{2\sigma_n^2}\left(\min_{\substack{\mathbf{s}\in\mathcal{O}^{N_T}:\\ x_{j,b}=-1}} \|\mathbf{y}-\mathbf{H}\mathbf{s}\|^2 - \min_{\substack{\mathbf{s}\in\mathcal{O}^{N_T}:\\ x_{j,b}=+1}} \|\mathbf{y}-\mathbf{H}\mathbf{s}\|^2\right). \quad (2)$$

One of the two minimum values in (2) is determined by the ML hypothesis as

$$\mathbf{s}^{\mathrm{ML}} = \arg\min_{\mathbf{s}\in\mathcal{O}^{N_T}} \|\mathbf{y}-\mathbf{H}\mathbf{s}\|^2 \quad (3)$$

and the other minimum value is determined by its (j,b)th counterhypothesis as

$$\mathbf{s}_{j,b}^{\mathrm{CH}} = \arg\min_{\mathbf{s}\in\mathcal{O}^{N_T}: x_{j,b}=\overline{x_{j,b}^{\mathrm{ML}}}} \|\mathbf{y}-\mathbf{H}\mathbf{s}\|^2 \quad (4)$$

where $x_{j,b}$ and $x_{j,b}^{\mathrm{ML}}$ denote the bth bit of the jth symbol in $\mathbf{s}$ and $\mathbf{s}^{\mathrm{ML}}$, respectively, and $\overline{x}$ denotes the binary complement of x. Then, the max-log LLR in (2) can be rewritten as

$$\mathrm{LLR}_{j,b} = \frac{1}{2\sigma_n^2}\left(\Lambda_{j,b}^{\mathrm{CH}} - \Lambda^{\mathrm{ML}}\right) x_{j,b}^{\mathrm{ML}} \quad (5)$$

where $\Lambda^{\mathrm{ML}} = \|\mathbf{y}-\mathbf{H}\mathbf{s}^{\mathrm{ML}}\|^2$ and $\Lambda_{j,b}^{\mathrm{CH}} = \|\mathbf{y}-\mathbf{H}\mathbf{s}_{j,b}^{\mathrm{CH}}\|^2$ are the distance metrics for the ML hypothesis and its (j,b)th counterhypothesis, respectively.

B. Tree Structure and Unified TS Algorithm

Consider the QL-decomposition $\mathbf{H} = \mathbf{Q}\mathbf{L}$, where $\mathbf{Q}$ is an $N_R \times N_T$ matrix with orthonormal columns and $\mathbf{L}$ is an $N_T \times N_T$ lower triangular matrix with real positive diagonal entries. Then, the transformed received signal vector $\tilde{\mathbf{y}}$ is obtained as $\tilde{\mathbf{y}} = \mathbf{Q}^H\mathbf{y} = \mathbf{L}\mathbf{s} + \tilde{\mathbf{n}}$, where $\tilde{\mathbf{n}} = \mathbf{Q}^H\mathbf{n}$. Denoting $\mathbf{s}_{1:m} = [s_1, s_2, \ldots, s_m]^T$ as a partial symbol vector (PSV), its distance metric $\lambda_m(\mathbf{s}_{1:m})$, called the partial distance metric (PDM) of $\mathbf{s}_{1:m}$, is given by

$$\lambda_m(\mathbf{s}_{1:m}) = \sum_{j=1}^{m}\left|\tilde{y}_j - \sum_{i=1}^{j} l_{ji}s_i\right|^2 \quad (6)$$

where $\tilde{y}_j$ is the jth entry of $\tilde{\mathbf{y}}$, and l_{ji} is the (j,i)th entry of $\mathbf{L}$. Since $\lambda_m(\mathbf{s}_{1:m})$ depends only on the PSV $\mathbf{s}_{1:m}$, it can be calculated by the recursive formula $\lambda_m(\mathbf{s}_{1:m}) = \lambda_{m-1}(\mathbf{s}_{1:m-1}) + \phi_m(\mathbf{s}_{1:m})$, for $m = 1, \ldots, N_T$, where $\phi_m(\mathbf{s}_{1:m})$ is the partial increment, given by $\phi_m(\mathbf{s}_{1:m}) = |\tilde{y}_m - \sum_{i=1}^{m} l_{mi}s_i|^2$. Note that although $\mathbf{s}_{1:0}$ is not defined, we set $\lambda_0(\mathbf{s}_{1:0}) = 0$ for initialization. By the recursive formula, a PSV and its PDM can be mapped into the corresponding node of the tree, as shown in Fig. 1. The tree consists of $N_T + 1$ levels (each level is indexed by $0, \ldots, N_T$ from top to bottom), the set of nodes in the mth level is denoted by $\mathcal{N}_m$, and the whole node is denoted by $\mathcal{N} = \bigcup_{m=0}^{N_T}\mathcal{N}_m$. In particular, the node in $\mathcal{N}_0$ is called the root node, and each node in $\mathcal{N}_{N_T}$ is called a leaf node. Each node except leaf nodes has Q outgoing branches, and the qth branch corresponds to the qth symbol of the alphabet, i.e., $o_q \in \mathcal{O}$. Thus, every node in the tree is uniquely mapped to a particular PSV by collecting the corresponding symbols along the branches from the root node to a particular node. For brevity, we use "node $\mathbf{s}_{1:m}$" to refer to the node corresponding to the PSV $\mathbf{s}_{1:m}$, and $L(\mathbf{s}_{1:m})$ represents the level of the node $\mathbf{s}_{1:m}$.

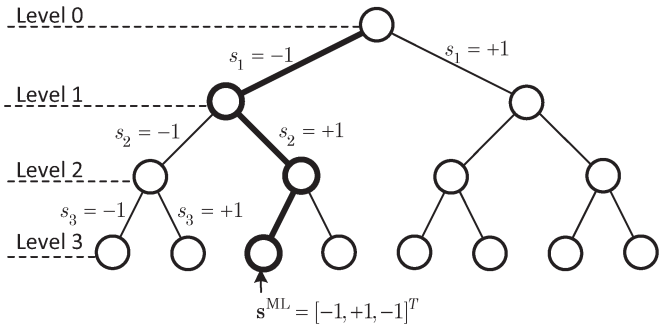

Fig. 1. Tree of a 3×3 MIMO system with BPSK modulation $\mathcal{O} = \{-1, +1\}$.

1: Initialization step:
$k \leftarrow 0$, $\mathcal{S} \leftarrow \phi$, $\mathcal{L} \leftarrow \phi$ $\gamma \leftarrow 0$, $\Lambda^{\mathrm{ML}} \leftarrow \Theta^{\max}$, $\Lambda_{j,b}^{\mathrm{CH}} \leftarrow \Theta^{\max}$ for $\forall(j,b)$
2: Add the root node in $\mathcal{S}$
3: **while** $|\mathcal{S}| > 0$ and $\gamma < \gamma_{\max}$ **do**
4: Determine the k-th candidate set, $\mathcal{C}_k \leftarrow \mathrm{SearchSpace}(k)$
5: For each node $\mathrm{Node} \in \mathcal{S}$, if $\mathrm{Node} \in \mathcal{C}_k$, $\mathrm{Node} \in \mathcal{B}$ and $\mathcal{S} \leftarrow \mathcal{S}\backslash\mathcal{B}$
6: **if** $|\mathcal{B}| = 0$ **then**
7: Go to next stage, $k \leftarrow k + 1$
8: **else**
9: **while** $\gamma < \gamma_{\max}$ **do**
10: Arrange the nodes in $\mathcal{B}$ according to SearchOrder
11: Select the first node in $\mathcal{B}$, denoted as S-Node
12: Expand the unexpanded child node, denoted as C-Node, having the smallest PDM
13: Cumulate the number of expanded nodes, $\gamma \leftarrow \gamma + 1$
14: **if** $L(\text{C-Node}) < N_{\mathrm{T}}$ **then**
15: If $\text{C-Node} \in \mathcal{C}_k$, $\mathcal{B} \leftarrow \mathcal{B} \cup \{\text{C-Node}\}$, else $\mathcal{S} \leftarrow \mathcal{S} \cup \{\text{C-Node}\}$
16: **else**
17: $\mathcal{L} \leftarrow \mathcal{L} \cup \{\text{C-Node}\}$ and update $\hat{\mathbf{s}}^{\mathrm{ML}}$ and $\hat{\mathbf{s}}_{j,b}^{\mathrm{CH}}$ for $\forall(j,b)$ by (8) and (9)
18: Remove S-Node from $\mathcal{B}$
19: Update the k-th candidate set, $\mathcal{C}_k \leftarrow \mathrm{SearchSpace}(k)$
20: **end if**
21: Remove S-Node from $\mathcal{B}$ if all child nodes of S-Node are expanded
22: **end while**
23: **end if**
24: **end while**
25: Compute max-log LLR $\mathrm{LLR}_{j,b}$ using (10) and the TS algorithm is terminated

Fig. 2. Unified TS algorithm for soft-output MIMO detection.

Soft-output MIMO detection can then be transformed into a TS problem, in which at most $N_T + 1$ leaf nodes corresponding to the ML hypothesis and all its counterhypotheses need to be found. A unified TS algorithm[2] for soft-output MIMO detection is summarized in Fig. 2, and two algorithm-dependent operations, namely, SearchSpace and SearchOrder, are listed in Table I. The operation SearchSpace determines the set of candidate nodes $\mathcal{C}_k$ in the kth stage, and the operation SearchOrder determines the search order of the nodes. Although the search space and order update is efficiently performed

[2]In the unified TS algorithm, $\mathcal{S}$ and $\mathcal{L}$ are nonordered sets, but $\mathcal{B}$ is an ordered set in which nodes are arranged according to the ascending order of its cost metric, and $\Theta^{\max}$ is a positive large value for initialization.

TABLE I
ALGORITHM-DEPENDENT OPERATIONS, SEARCHSPACE, AND SEARCHORDER FOR EACH ALGORITHM

		SearchSpace(k)	SearchOrder
DFS [6]		$\mathcal{C}_k = \{\mathbf{s}_{1:m} \in \mathcal{N} \vert \lambda_m(\mathbf{s}_{1:m}) \leq R(\mathbf{s}_{1:m})\}$	$-L(\mathbf{s}_{1:m})$
MFS [7]		$\mathcal{C}_k = \mathcal{N}$	$\lambda_m(\mathbf{s}_{1:m})$
MS [13]		$\mathcal{C}_k = \mathcal{N}_k$	$\lambda_m(\mathbf{s}_{1:m})$
P-SD [14]		$\mathcal{C}_k = \{\mathbf{s}_{1:m} \in \mathcal{N} \vert \lambda_m(\mathbf{s}_{1:m}) \leq m + \delta_\epsilon \log N_{\mathrm{T}}\}$	$-L(\mathbf{s}_{1:m})$
MBF-FD [15]		$\mathcal{C}_k = \mathcal{N}$	$\lambda_m(\mathbf{s}_{1:m})$, if $L(\text{C-Node}) = N_{\mathrm{T}}$ $-L(\mathbf{s}_{1:m})$, otherwise
HTS	$k = 0$	$\mathcal{C}_k = \{\mathbf{s}_{1:m} \in \mathcal{N} \vert \lambda_m(\mathbf{s}_{1:m}) \leq \widehat{\Lambda}^{\mathrm{ML}}\}$	$\lambda_m(\mathbf{s}_{1:m}) - 2\sigma_n^2 L(\mathbf{s}_{1:m}) - \Theta(\mathbf{s}_{1:m})$
	$k > 0$	$\mathcal{C}_k = \{\mathbf{s}_{1:m} \in \mathcal{H}_k \vert \lambda_m(\mathbf{s}_{1:m}) \leq \Theta(\mathbf{s}_{1:m})\}$	

by using the boundary set $\mathcal{B}$, the results are the same as the case where the search space and order update is performed at the beginning of each stage or whenever any hypothesis is updated. In DFS (LSD [1] and STS [6]), SearchOrder uses the descending order of the level of nodes, and SearchSpace determines the candidate node set as all nodes with its PDM less than the sphere radius $R(\mathbf{s}_{1:m})$, which is defined by [6]

$$R(\mathbf{s}_{1:m}) = \max_{\forall (j,b): x_{j,b} \neq x_{j,b}^{\mathrm{ML}}} \Lambda_{j,b}^{\mathrm{CH}} \tag{7}$$

where $x_{j,b}^{\mathrm{ML}}$ is the bth bit of the jth symbol in $\mathbf{s}^{\mathrm{ML}}$. In MFS [7], SearchOrder uses the ascending order of the PDM of nodes, and SearchSpace includes all nodes in the candidate node set. The MS algorithm [13] is similar to MFS, except that SearchSpace uses the nodes in the kth level as $\mathcal{C}_k$. The P-SD algorithm [14] is similar to DFS, except that SearchSpace adjusts the parameter δ_ϵ to make $|\mathcal{C}_k|$ constant. The MBF-FD algorithm [15] uses SeachOrder of DFS and MFS, in turn, to take advantage of both algorithms.

C. Runtime Constraint

To guarantee a constant throughput, a TS algorithm should be early terminated whenever the runtime γ for detecting a specific received signal exceeds the maximum allowed runtime $\gamma_{\max}$. An early termination does not occur in BFS due to its fixed complexity, but the complexity of DFS and MFS is a random variable depending on the instantaneous channel and noise realizations so that, when an early termination occurs, such an algorithm should return the LLR computed by the tentative ML hypothesis $\widehat{\mathbf{s}}^{\mathrm{ML}}$ and its (j,b)th tentative counterhypothesis $\widehat{\mathbf{s}}_{j,b}^{\mathrm{CH}}$, which are defined by

$$\widehat{\mathbf{s}}^{\mathrm{ML}} = \arg\min_{\mathbf{s} \in \mathcal{L}} \|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2 \tag{8}$$

$$\widehat{\mathbf{s}}_{j,b}^{\mathrm{CH}} = \arg\min_{\mathbf{s} \in \mathcal{L}: x_{j,b} = \overline{\widehat{x}_{j,b}^{\mathrm{ML}}}} \|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2 \tag{9}$$

where $\mathcal{L}$ is the set of leaf nodes expanded so far, and $\widehat{x}_{j,b}^{\mathrm{ML}}$ is the bth bit of the jth symbol in $\widehat{\mathbf{s}}^{\mathrm{ML}}$. Similarly, the distance metric of the tentative ML hypothesis and its (j,b)th tentative counterhypothesis are denoted by $\widehat{\Lambda}^{\mathrm{ML}}$ and $\widehat{\Lambda}_{j,b}^{\mathrm{CH}}$, respectively, and the corresponding max-log LLR is computed by

$$\widehat{\mathrm{LLR}}_{j,b} = \frac{1}{2\sigma_n^2}\left(\widehat{\Lambda}_{j,b}^{\mathrm{CH}} - \widehat{\Lambda}^{\mathrm{ML}}\right)\widehat{x}_{j,b}^{\mathrm{ML}}. \tag{10}$$

Clearly, the sign of $\widehat{\mathrm{LLR}}_{j,b}$ is solely determined by the tentative ML hypothesis because $\widehat{\Lambda}_{j,b}^{\mathrm{CH}} - \widehat{\Lambda}^{\mathrm{ML}} \geq 0$ for any (j,b), and the magnitude of $\widehat{\mathrm{LLR}}_{j,b}$ depends on both the tentative ML hypothesis and its (j,b)th tentative counterhypothesis.

The runtime of TS algorithms can be measured by the number of floating-point operations or silicon complexity [2]. We measure the runtime as the number of expanded nodes, which is the most popularly accepted way in the literature because it can be mapped into the real runtime [6], [16], is independent of target architecture [2], and is mathematically tractable [8]. Note that, since our measure does not capture the total runtime, we additionally compare the CPU runtime to clarify the effect of additional complexity independent of the number of expanded nodes.

III. HIERARCHICAL SEARCH TREE

Here, we describe two components of the proposed HTS algorithm. First, the whole tree is partitioned into the disjoint sets, called the HS, and then, the new cost metric, called the HTS cost metric, is derived. Finally, the proposed HTS algorithm is described in the framework of the unified TS algorithm.

1) HS Partitioning: The constrained runtime may be too short to find the ML hypothesis and all its counterhypotheses in a bad channel or noise realization [18], or it may be entirely wasted to search only a few hypotheses [10]. As a result, the quality of some tentative hypotheses becomes very poor, which mainly degrades the overall performance. To remedy such a problem, we first find the ML hypothesis and then partition the tree into N_T HSs according to the ML hypothesis. To improve each counterhypothesis fairly, each HS should have the same number of candidate nodes for each counterhypothesis. For the ML hypothesis $\mathbf{s}^{\mathrm{ML}}$, let $\{\mathcal{H}_1, \ldots, \mathcal{H}_{N_T}\}$ be the hierarchical partition, where the kth HS is defined as

$$\mathcal{H}_k = \left\{ \mathbf{s}_{1:m} \left| \sum_{j=1}^{m} \mathbf{1}_{\{s_j \neq s_j^{\mathrm{ML}}\}} = k, \ \forall \mathbf{s}_{1:m} \in \mathcal{N} \right. \right\} \tag{11}$$

and $\mathbf{1}_{\{\cdot\}}$ denotes the indicator function. In other words, every node $\mathbf{s}_{1:m}$ in $\mathcal{H}_k$ has k different symbols compared with the ML hypothesis. Define the (j,b)th counterhypothesis found in $\mathcal{H}_k$ as

$$\mathbf{s}_{j,b,k}^{\mathrm{CH}} = \arg\min_{\mathbf{s} \in \mathcal{H}_k \cap \mathcal{N}_{N_T}} \|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2.$$

Then, the following theorem suggests the order of HS utilization as a candidate node set.

Theorem 1: Assume that all entries of matrix $\mathbf{H}$ are finite. Then, for $1 \leq n < m \leq N_T$

$$\mathbb{E}\left\|\mathbf{y} - \mathbf{H}\mathbf{s}_{j,b,n}^{\mathrm{CH}}\right\|^2 < \mathbb{E}\left\|\mathbf{y} - \mathbf{H}\mathbf{s}_{j,b,m}^{\mathrm{CH}}\right\|^2 + \epsilon \tag{12}$$

for an arbitrarily small $\epsilon > 0$, provided that the SNR is sufficiently large.

Proof: See the Appendix. ■

Theorem 1 implies that the hypotheses obtained from $\mathcal{H}_n$ is more reliable than those from $\mathcal{H}_m$ in an average sense, and it is reasonable to use $\mathcal{H}_1$ first and utilize the next HSs sequentially.

2) Cost Metric: The search order for the candidate nodes has a significant impact on the performance when an early termination occurs. A reasonable way for determining the search order is to select the node providing the largest improvement. To quantify the improvement, we first define the worst hypothesis quality of a node as follows. When a subtree originated from a node is searched, its leaf nodes can update the tentative counterhypothesis $\widehat{\mathbf{s}}_{j,b}^{\mathrm{CH}}$ only if $x_{j,b} \neq \widehat{x}_{j,b}^{\mathrm{ML}}$. Thus, the worst hypothesis quality of node $\mathbf{s}_{1:m}$ can be defined as the maximum distance metric among the tentative counterhypotheses affected by expanding the node $\mathbf{s}_{1:m}$, given by

$$\Theta(\mathbf{s}_{1:m}) = \max_{\forall (j,b), x_{j,b} \neq x_{j,b}^{\mathrm{ML}}} \widehat{\Lambda}_{j,b}^{\mathrm{CH}}.$$

Note that $\Theta(\mathbf{s}_{1:m})$ is identical to the sphere radius $R(\mathbf{s}_{1:m})$ in (7), so that no additional complexity is required. Now, define the *improvement* of a node $\mathbf{s}_{1:m}$ as the difference between the worst hypothesis quality and the quality of the node $\mathbf{s}_{1:m}$ as

$$I(\mathbf{s}_{1:m}) = \Theta(\mathbf{s}_{1:m}) - \min_{\mathbf{s}_{m+1:N_T}} \lambda_{N_T}(\mathbf{s}_{1:N_T}) \tag{13}$$

where the second term denotes the quality of the node $\mathbf{s}_{1:m}$, which is defined by the minimum distance metric among the leaf nodes originated from $\mathbf{s}_{1:m}$. To alleviate the complexity for computing (13), the subvector of the ML hypothesis, i.e., $\mathbf{s}_{m+1:N_T}^{\mathrm{ML}}$, is used instead. Assuming that the subvector is correct and the distance is averaged over noise realizations as in [19], the improvement of a node $\mathbf{s}_{1:m}$ can be approximated as

$$I(\mathbf{s}_{1:m}) \approx \Theta(\mathbf{s}_{1:m}) - \left(\lambda_m(\mathbf{s}_{1:m}) + 2\sigma_n^2 \left(N_T - L(\mathbf{s}_{1:m})\right)\right). \tag{14}$$

Then, by ignoring a constant, the proposed HTS cost metric of a node $\mathbf{s}_{1:m}$ is given by

$$C^{\mathrm{HTS}}(\mathbf{s}_{1:m}) = \lambda_m(\mathbf{s}_{1:m}) - 2\sigma_n^2 L(\mathbf{s}_{1:m}) - \Theta(\mathbf{s}_{1:m}). \tag{15}$$

Note that the additional complexity for computing the proposed HTS cost metric is only two additions per node, which is ignorable.

3) Proposed HTS Algorithm: The proposed HTS algorithm can be described by the framework of the unified TS algorithm by replacing two algorithm-dependent operations, which are also listed in Table I. The operation SearchSpace of the HTS algorithm in the 0th stage determines the candidate node set as $\mathcal{C}_0 = \{\mathbf{s}_{1:m} \in \mathcal{N} | \lambda_m(\mathbf{s}_{1:m}) \leq \widehat{\Lambda}^{\mathrm{ML}})\}$. In the kth stage $(k = 1, \ldots, N_T)$, the operation SearchSpace determines the candidate node set as the nodes in $\mathcal{H}_k$ with its PDM less than the worst hypothesis quality, and the operation SearchOrder uses the ascending order of the HTS cost metric in (15).

Here, an illustrative example of the proposed HTS algorithm is shown in Fig. 3, and the search result is listed in Table II. In Fig. 3, we assume that the PDM of each node is calculated and indicated inside the nodes and $\sigma_n^2 = 1$. At the beginning of the 0th stage, the candidate node set is $\mathcal{C}_0 = \mathcal{N}$ due to $\widehat{\Lambda}^{\mathrm{ML}} = \Theta^{\max}$; thus, $\mathcal{B} = \{N_0\}$. Then, the PDM $\lambda_m(\cdot)$, the worst hypothesis quality $\Theta(\cdot)$, and its HTS cost metric $C^{\mathrm{HTS}}(\cdot)$ of the child node N_1 originated from N_0 are computed and inserted into $\mathcal{B}$ since $N_1 \in \mathcal{C}_0$. Two nodes N_0 and N_1 stored in $\mathcal{B}$ are sorted by the ascending order of the HTS cost metric so that N_1 becomes the first node and N_0 becomes the second node. The next search begins from the first node N_1. After the fifth expansion, the leaf node N_9 is expanded and inserted into the leaf node set $\mathcal{L}$, and the new tentative ML hypothesis is found, i.e., $\widehat{\mathbf{s}}^{\mathrm{ML}} = N_9$, and $\widehat{\Lambda}^{\mathrm{ML}}$ is updated to 7. At the sixth, seventh, and eighth expansions, N_{10}, N_7, and N_8 are expanded, respectively, and the tentative hypotheses and their distance metrics are updated ($\mathbf{s}_{1,1}^{\mathrm{CH}} = N_{10}$, $\widehat{\Lambda}_{1,1}^{\mathrm{CH}} = 13$, $\mathbf{s}_{2,1}^{\mathrm{CH}} = N_7$, $\widehat{\Lambda}_{2,1}^{\mathrm{CH}} = 14$, $\mathbf{s}_{3,1}^{\mathrm{CH}} = N_8$, and $\widehat{\Lambda}_{3,1}^{\mathrm{CH}} = 16$). This procedure is iteratively repeated until $\mathcal{B}$ becomes empty, and the HTS algorithm proceeds to the next stage. At the beginning of the first stage, the HSs are first determined so that $\mathcal{H}_1 = \{N_2, N_3, N_6, N_7, N_{10}, N_{13}\}$, $\mathcal{H}_2 = \{N_5, N_8, N_{11}, N_{14}\}$, and $\mathcal{H}_3 = \{N_{12}\}$ are obtained, as shown in Fig. 3, because $\mathbf{s}^{\mathrm{ML}} = N_9$ and the candidate node set is determined as $\mathcal{C}_1 = \{N_6, N_{13}\}$, which is the subset of $\mathcal{H}_1$. Similar to the 0th stage, the child node is recursively expanded until $\mathcal{B}$ becomes empty. After the 13th expansion, the second stage begins, and the candidate node set is determined as $\mathcal{C}_2 = \{N_5, N_{11}\}$, which is the subset of $\mathcal{H}_2$. After the 14th expansion, all nodes of the tree are searched; thus, the HTS algorithm is finished.

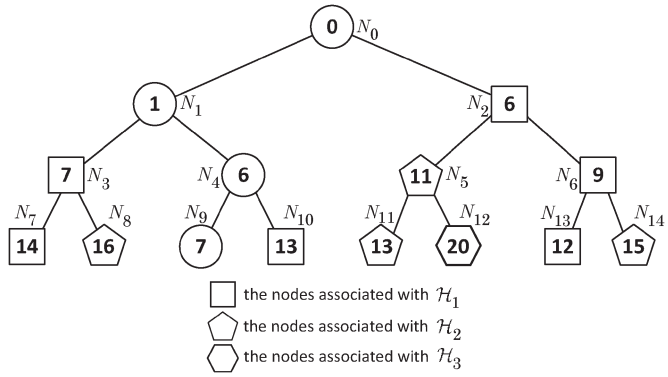

Fig. 3. Illustrative example of the HS partition for the 3 × 3 MIMO system with BPSK modulation, where $\mathbf{s}^{\mathrm{ML}} = N_9$.

IV. Simulation Results

Simulation results are obtained by using a 64-tone MIMO orthogonal frequency-division multiplexing (OFDM) system with $N_T = 4$, $N_R = 4$, and 16-quadrature amplitude modulation (QAM) modulation alphabet (referred to as 4 × 4 16-QAM) or with $N_T = 6$, $N_R = 6$, and 64-QAM modulation alphabet (referred to as 6 × 6 64-QAM). The convolutional code (CC) with the rate R_c of 1/2 using the generator polynomials of $[133_o \quad 171_o]$ and the 5/6-rate punctured CC using the 1/2-rate CC with the puncturing pattern $([1 \quad 0 \quad 1 \quad 0 \quad 1], [1 \quad 1 \quad 0 \quad 1 \quad 0])$ are used together with a random interleaver. For each simulation, 10^4 frames (four OFDM symbols per frame) are decoded by using each TS algorithm and soft-input Viterbi decoder.

In the simulation, the following algorithms are compared.

1) As the representative of DFS, the STS algorithm [6] with MMSE-sorted QR decomposition (QRD) preprocessing is considered. To obtain the performance–complexity tradeoff, the clipping level of the max-log LLR is set to the (nearly) optimal value obtained from intensive computer simulation.
2) As the representative of BFS, the smart ordering and candidate adding (SOCA) algorithm [17] with smart-ordered QRD preprocessing is considered.
3) As the representative of MFS, Dijkstra's algorithm [7] with QRD preprocessing is considered.

TABLE II
EXAMPLE OF THE PROPOSED HTS ALGORITHM

	γ	$\mathcal{B}=\{\mathrm{Node}(\Theta, C^{\mathrm{HTS}})\}$	$\mathcal{S}=\{\mathrm{Node}(\Theta, C^{\mathrm{HTS}})\}$	$\widehat{\mathbf{s}}^{\mathrm{ML}}(\widehat{\Lambda}^{\mathrm{ML}})$	$\widehat{\mathbf{s}}^{\mathrm{CH}}_{1,1}(\widehat{\Lambda}^{\mathrm{CH}}_{1,1})$	$\widehat{\mathbf{s}}^{\mathrm{CH}}_{2,1}(\widehat{\Lambda}^{\mathrm{CH}}_{2,1})$	$\widehat{\mathbf{s}}^{\mathrm{CH}}_{3,1}(\widehat{\Lambda}^{\mathrm{CH}}_{3,1})$
Initialization	0	-	$N_0(\Theta^{\max}, -\Theta^{\max})$	-	-	-	-
The 0th stage	1	$N_1(\Theta^{\max}, -1-\Theta^{\max})$ $N_0(\Theta^{\max}, -\Theta^{\max})$	-	-	-	-	-
	2	$N_1(\Theta^{\max}, -1-\Theta^{\max})$ $N_0(\Theta^{\max}, -\Theta^{\max})$ $N_4(\Theta^{\max}, 2-\Theta^{\max})$	-	-	-	-	-
	3	$N_0(\Theta^{\max}, -\Theta^{\max})$ $N_4(\Theta^{\max}, 2-\Theta^{\max})$ $N_3(\Theta^{\max}, 3-\Theta^{\max})$	-	-	-	-	-
	4	$N_4(\Theta^{\max}, 2-\Theta^{\max})$ $N_3(\Theta^{\max}, 3-\Theta^{\max})$ $N_2(\Theta^{\max}, 4-\Theta^{\max})$	-	-	-	-	-
	5	$N_9(\Theta^{\max}, 1-\Theta^{\max})$ $N_4(\Theta^{\max}, 2-\Theta^{\max})$ $N_3(\Theta^{\max}, 3-\Theta^{\max})$ $N_2(\Theta^{\max}, 4-\Theta^{\max})$	-	$N_9(7)$	-	-	-
	6	$N_3(\Theta^{\max}, 3-\Theta^{\max})$ $N_2(\Theta^{\max}, 4-\Theta^{\max})$	-	$N_9(7)$	$N_{10}(13)$	-	-
	7	$N_3(\Theta^{\max}, 3-\Theta^{\max})$ $N_2(\Theta^{\max}, 4-\Theta^{\max})$	-	$N_9(7)$	$N_{10}(13)$	$N_7(14)$	-
	8	$N_2(16, -12)$	-	$N_9(7)$	$N_{10}(13)$	$N_7(14)$	$N_8(16)$
	9	$N_2(16, -12)$	$N_6(16, -11)$	$N_9(7)$	$N_{10}(13)$	$N_7(14)$	$N_8(16)$
	10	-	$N_6(16, -11)$ $N_5(16, -9)$	$N_9(7)$	$N_{10}(13)$	$N_7(14)$	$N_8(16)$
The 1st stage	11	$N_6(16, -11)$	$N_5(16, -9)$	$N_9(7)$	$N_{10}(13)$	$N_7(14)$	$N_8(16)$
	12	$N_6(16, -11)$	$N_5(16, -9)$	$N_9(7)$	$N_{13}(12)$	$N_7(14)$	$N_8(16)$
	13	-	$N_5(15, -10)$	$N_9(7)$	$N_{13}(12)$	$N_7(14)$	$N_{14}(15)$
The 2nd stage	14	$N_5(15, -10)$	-	$N_9(7)$	$N_{13}(12)$	$N_{11}(13)$	$N_{14}(15)$
The 3rd stage	14	-	-	$N_9(7)$	$N_{13}(12)$	$N_{11}(13)$	$N_{14}(15)$

4) As the representative of the multistage algorithm, the MS algorithm [13] with V-BLAST QRD preprocessing is considered. The natural restart ordering is applied.
5) As the representative of a non-TS algorithm, the algorithm proposed in [20] is considered. The number of expanded full-length symbol vectors is set to $\gamma_{\max}/N_T$.[3]

For fair comparison, the maximum size of the list $\mathcal{B}$ for each algorithm is set to 32 for the 4×4 16-QAM system and 64 for the 6×6 64-QAM system. The clipping level of the max-log LLR is set to 5 for all except the STS algorithm to prevent over- or underestimation of the max-log LLR.

Fig. 4 compares the performance–complexity tradeoffs of the STS algorithm [6], the SOCA algorithm [17], Dijkstra's algorithm [7], the MS algorithm [13], the non-TS algorithm [20], and the proposed HTS algorithm when the 4×4 16-QAM is used. The frame length is 4096 bits, and the minimum SNR required to achieve the frame error rate (FER) of 10^{-2} is obtained according to the runtime constraint $\gamma_{\max}$. As a reference, the performance using the exact max-log LLR is displayed by the horizontal lines. From the results, it is shown that SOCA provides a poor performance–complexity tradeoff in both code rates and that Dijkstra's algorithm suffers from severe performance degradation since it may not find any leaf node at all under a tight runtime constraint. Although the MS algorithm may mitigate the performance degradation of Dijkstra's algorithm and STS provides a relatively good performance–complexity tradeoff among conventional algorithms, HTS outperforms MS and STS algorithms over the wide range of the runtime constraint.[4] In addition, the performance gap increases as the code rate increases. Precisely, when $\gamma_{\max} = 12$, the SNR gain of the proposed HTS over STS is 2.3 dB for the 1/2 code rate and 6.2 dB for the 5/6 code rate. Furthermore, it is shown that the proposed HTS algorithm outperforms a non-TS algorithm such as in [20], which is slightly worse than the MS algorithm.

Fig. 5 provides the performance–complexity tradeoff when the 6×6 64-QAM is used. Here, the frame length is increased to 9216 bits. We can observe by comparing with Fig. 4 that the gain of the HTS algorithm over the STS algorithm is slightly increased (3.4 dB for the 1/2 code rate and 9.1 dB for the 5/6 code rate at $\gamma_{\max} = 48$) and that the proposed HTS algorithm becomes more attractive as the tree size increases.

To reflect the total runtime induced by any additional operations that are not captured by the number of expanded nodes, such as sorting or preprocessing, we compare the normalized CPU runtime[5] of the

[3]We assume the complexity to find one full-length symbol vector is similar to the complexity expanding N_T nodes.

[4]Although the slope of the tradeoff curve for Dijkstra's algorithm or the SOCA algorithm looks much sharper than that for the proposed HTS, they are similar if drawn with a linear-scale x-axis.

[5]The CPU runtime is measured by a personal computer (Intel E7300 with a 2.66-GHz clock) and MATLAB R2011a version, and the normalized CPU runtime of an algorithm A under a runtime constraint $\gamma_{\max}$ is defined as $\mu_A(\gamma_{\max}) = T_A/T_{\mathrm{HTS}}$, where T_A is the CPU runtime of an algorithm A.

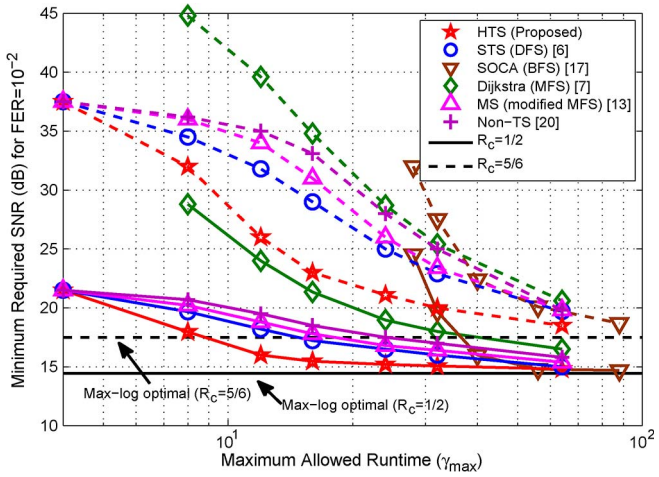

Fig. 4. Performance–complexity tradeoff curves when the 4 × 4 16-QAM is used.

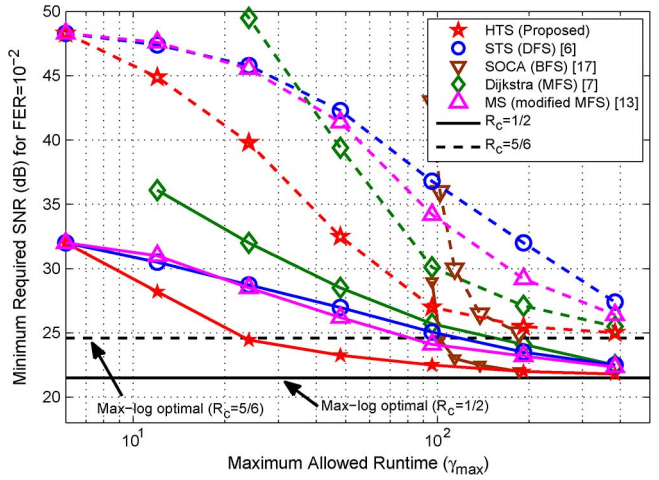

Fig. 5. Performance–complexity tradeoff curves when the 6 × 6 64-QAM is used.

various TS algorithms. As a result, the normalized runtime of the STS algorithm is $\mu_{\mathrm{STS}}(16) = 0.94$ and $\mu_{\mathrm{STS}}(64) = 0.89$, and that of the MS algorithm is $\mu_{\mathrm{MS}}(16) = 1.12$ and $\mu_{\mathrm{MS}}(64) = 1.09$. Therefore, the normalized CPU runtime per expanded node of the proposed HTS algorithm is not much different to those of the STS and MS algorithms (less than 11% when $\gamma_{\max} \leq 64$). Note that Dijkstra's algorithm consumes much higher normalized CPU runtime per expanded node since it requires a sorting operation in a large stack. If Fig. 5 is redrawn by including the additional complexity converted into the number of expanded nodes, the minimum required SNRs at $\gamma_{\max} = 12, 24, 48$ are changed from 30.5, 28.7, and 27.0 dB to 31.1, 29.3, and 27.4 dB, respectively, for the STS algorithm and from 28.2, 24.5, and 23.3 dB to 30.2, 26.5, and 23.9 dB, respectively, for the proposed HTS algorithm. Therefore, it is clearly confirmed that the practical performance–complexity tradeoff of the propose HTS algorithm still outperforms conventional algorithms.

V. Conclusion

In this paper, a novel multistage TS algorithm, which is referred to as the HTS, has been proposed for soft-output MIMO detection to mitigate the performance loss caused by an early termination due to a runtime constraint. To efficiently use the constrained runtime, the whole search space is divided into small disjoint sets, called the HS, and in each stage, one HS is utilized as the candidate node set to find all hypotheses with reasonable quality as fast as possible and then fairly improve the quality of each hypothesis. In addition, the proposed HTS cost metric determines the search order in the candidate node set and further improves the performance–complexity tradeoff.

Simulation results showed that, in a wide range of practical interest, the proposed HTS algorithm provides a significantly improved performance–complexity tradeoff and the gain increases as the number of antennas, the modulation order, and the code rate increase. Thus, the proposed HTS can be considered as promising soft-output MIMO detection for future wireless communication systems, and it would be fruitful to investigate an efficient hardware architecture for the proposed HTS algorithm as future work.

Appendix
Proof of Theorem 1

Define the difference vector between the ML hypothesis and the (j,b)th counterhypothesis found in $\mathcal{H}_n$ as $\boldsymbol{\delta}^n \triangleq \mathbf{s}^{\mathrm{CH}}_{j,b,n} - \mathbf{s}^{\mathrm{ML}} = \alpha_j \mathbf{e}_j + \sum_{i=1}^{n-1} \alpha_{k_i} \mathbf{e}_{k_i}$, where $\mathbf{e}_i$ is the $N_T \times 1$ vector having all zero entries except that the ith entry is 1 and $\alpha_{k_i} \triangleq (\mathbf{s}^{\mathrm{CH}}_{j,b,n})_{k_i} - (\mathbf{s}^{\mathrm{ML}})_{k_i}$, where $(\mathbf{x})_a$ is the ath entry of the vector $\mathbf{x}$. Similarly, the difference vector $\boldsymbol{\delta}^m$ for $m > n$ can be written as $\boldsymbol{\delta}^m \triangleq \mathbf{s}^{\mathrm{CH}}_{j,b,m} - \mathbf{s}^{\mathrm{ML}} = \beta_j \mathbf{e}_j + \sum_{i=1}^{m-1} \beta_{p_i} \mathbf{e}_{p_i}$. Since $m > n$, $\boldsymbol{\delta}^m$ can be rewritten as $\boldsymbol{\delta}^m = \boldsymbol{\zeta}^m_n + \boldsymbol{\Delta}^m_n$, where $\boldsymbol{\zeta}^m_n \triangleq \beta_j \mathbf{e}_j + \sum_{i=1}^{n-1} \beta_{p_i} \mathbf{e}_{p_i}$ and $\boldsymbol{\Delta}^m_n \triangleq \sum_{i=n}^{m-1} \beta_{p_i} \mathbf{e}_{p_i}$ are orthogonal, then we obtain

$$\begin{aligned} &\mathbb{E}\left\|\mathbf{y} - \mathbf{H}\mathbf{s}^{\mathrm{CH}}_{j,b,m}\right\|^2 - \mathbb{E}\left\|\mathbf{y} - \mathbf{H}\mathbf{s}^{\mathrm{CH}}_{j,b,n}\right\|^2 \\ &= \mathbb{E}\left\{\left\|\mathbf{n}^{\mathrm{ML}} - \mathbf{H}\boldsymbol{\zeta}^m_n\right\|^2 - \left\|\mathbf{n}^{\mathrm{ML}} - \mathbf{H}\boldsymbol{\delta}^n\right\|^2\right\} + \mathbb{E}\left\|\mathbf{H}\boldsymbol{\Delta}^m_n\right\|^2 \\ &\quad + 2\Re\left(\mathbb{E}\left\{\left(\mathbf{H}\boldsymbol{\zeta}^m_n\right)^H \mathbf{H}\boldsymbol{\Delta}^m_n\right\} - \mathbb{E}\left\{\left(\mathbf{n}^{\mathrm{ML}}\right)^H \mathbf{H}\boldsymbol{\Delta}^m_n\right\}\right) \end{aligned} \quad (16)$$

where $\mathbf{n}^{\mathrm{ML}} \triangleq \mathbf{y} - \mathbf{H}\mathbf{s}^{\mathrm{ML}}$, and $\Re(\cdot)$ denotes the real part. Since $\mathbf{s}^{\mathrm{ML}} + \boldsymbol{\zeta}^m_n \in \mathcal{H}_n$ and $\mathbf{s}^{\mathrm{ML}} + \boldsymbol{\delta}^n \in \mathcal{H}_n$, $\|\mathbf{n}^{\mathrm{ML}} - \mathbf{H}\boldsymbol{\zeta}^m_n\|^2 - \|\mathbf{n}^{\mathrm{ML}} - \mathbf{H}\boldsymbol{\delta}^n\|^2 \geq 0$. In addition, $\|\mathbf{H}\boldsymbol{\Delta}^m_n\|^2 \geq 0$ and $\mathbb{E}\{(\mathbf{H}\boldsymbol{\delta}^m_n)^H \mathbf{H}\boldsymbol{\Delta}^m_n\} = (\boldsymbol{\zeta}^m_n)^H \mathbb{E}\{\mathbf{H}^H\mathbf{H}\}\boldsymbol{\Delta}^m_n = (N_R/N_T)(\boldsymbol{\zeta}^m_n)^H \boldsymbol{\Delta}^m_n = 0$ since $\boldsymbol{\delta}^m_n$ and $\boldsymbol{\Delta}^m_n$ are orthogonal. Finally, the magnitude of the last term is bounded as

$$\begin{aligned} \left|\mathbb{E}\left\{(\mathbf{n}^{\mathrm{ML}})^H \mathbf{H}\boldsymbol{\Delta}^m_n\right\}\right| &= \left|\mathbb{E}\left\{(\mathbf{s}^{\mathrm{TX}} - \mathbf{s}^{\mathrm{ML}})^H \mathbf{H}^H \mathbf{H}\boldsymbol{\Delta}^m_n\right\}\right| \\ &\leq cN_T \Pr(\mathbf{s}^{\mathrm{TX}} \neq \mathbf{s}^{\mathrm{ML}}) \mathbb{E}\|\mathbf{H}^H \mathbf{H}\boldsymbol{\Delta}^m_n\| \end{aligned} \quad (17)$$

where $\mathbf{s}^{\mathrm{TX}}$ is the transmitted symbol vector, and $\mathbf{n}$ is the noise vector defined in (1). Here, $|\mathbb{E}\{x\}| \leq \mathbb{E}|x|$, and the Cauchy–Schwarz inequality is applied, and $\mathbb{E}\|\mathbf{s}^{\mathrm{TX}} - \mathbf{s}^{\mathrm{ML}}\| \leq cN_T \Pr(\mathbf{s}^{\mathrm{TX}} \neq \mathbf{s}^{\mathrm{ML}})$ is used with any finite number $c \geq \max_{a,b \in \mathcal{O}} |a - b|$. Thus, for a sufficiently large SNR, $\Pr(\mathbf{s}^{\mathrm{TX}} \neq \mathbf{s}^{\mathrm{ML}})$ can be made arbitrarily small so that the right-hand side of (17) is greater than $-\epsilon$ for any given $\epsilon > 0$, which concludes the proof.

References

[1] B. Hochwald and S. ten Brink, "Achieving near-capacity on a multiple-antenna channel," *IEEE Trans. Commun.*, vol. 51, no. 3, pp. 389–399, Mar. 2003.

[2] D. Garrett, L. Davis, S. ten Brink, B. Hochwald, and G. Knagge, "Silicon complexity for maximum likelihood MIMO detection using spherical decoding," *IEEE J. Solid-State Circuits*, vol. 39, no. 9, pp. 1544–1552, Sep. 2004.
[3] Z. Guo and P. Nilsson, "Algorithm and implementation of the K-best sphere decoding for MIMO detection," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 3, pp. 491–503, Mar. 2006.
[4] Y. L. C. de Jong and T. J. Willink, "Iterative tree search detection for MIMO wireless systems," *IEEE Trans. Commun.*, vol. 53, no. 6, pp. 930–935, Jun. 2005.
[5] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closest point search in lattices," *IEEE Trans. Inf. Theory*, vol. 48, no. 8, pp. 2201–2214, Aug. 2002.
[6] C. Studer, A. Burg, and H. Bölcskei, "Soft-output sphere decoding: Algorithm and VLSI implementation," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 2, pp. 290–300, Feb. 2008.
[7] M. Myllylä, M. Juntti, and J. R. Cavallaro, "A list sphere detector based on Dijkstra's algorithm for MIMO-OFDM systems," in *Proc. IEEE Int. Symp. Pers., Indoor, Mobile Radio Commun.*, Athens, Greece, Sep. 3–7, 2007, pp. 1–5.
[8] K. Su, "Efficient maximum likelihood detection for communication over multiple input multiple output channels," Ph.D. dissertation, Dept. Eng., Univ. Cambridge, Cambridge, U.K., 2005.
[9] A. Burg, M. Borgmann, M. Wenk, C. Studer, and H. Bölcskei, "Advanced receiver algorithms for MIMO wireless communications," in *Proc. Des. Autom. Test Eur. Conf.*, Mar. 6–10, 2006, vol. 1, pp. 593–598.
[10] M. Rachid and B. Daneshrad, "Iterative MIMO sphere decoding throughput guarantees under realistic channel conditions," *IEEE Commun. Lett.*, vol. 14, no. 4, pp. 342–344, Apr. 2010.
[11] L. G. Barbero and J. S. Thompson, "Fixing the complexity of the sphere decoder for MIMO detection," *IEEE Trans. Wireless Commun.*, vol. 7, no. 6, pp. 2131–2142, Jun. 2008.
[12] L. G. Barbero and J. S. Thompson, "Extending a fixed-complexity sphere decoder to obtain likelihood information for turbo-MIMO systems," *IEEE Trans. Veh. Technol.*, vol. 57, no. 5, pp. 2804–2814, Sep. 2008.
[13] M. Nekuii and T. Davidson, "A multistack algorithm for soft MIMO demodulation," *IEEE Trans. Veh. Technol.*, vol. 58, no. 5, pp. 2592–2597, Jun. 2009.
[14] C. S. Park, K. K. Parhi, and S. C. Park, "Probabilistic spherical detection and VLSI implementation for multiple-antenna systems," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 3, pp. 685–698, Mar. 2009.
[15] L. Chun-Hao, W. To-Ping, and C. Tzi-Dar, "A 74.8 mW soft-output detector IC for 8 × 8 spatial-multiplexing MIMO communications," *IEEE J. Solid-State Circuits*, vol. 45, no. 2, pp. 411–421, Feb. 2010.
[16] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bölcskei, "VLSI implementation of MIMO detection using the sphere decoding algorithm," *IEEE J. Solid-State Circuits*, vol. 40, no. 7, pp. 1566–1577, Jul. 2005.
[17] D. L. Milliner, E. Zimmermann, J. R. Barry, and G. Fettweis, "A fixed complexity smart candidate adding algorithm for soft-output MIMO detection," *IEEE. J. Sel. Topics Signal Process.*, vol. 3, no. 6, pp. 1016–1025, Dec. 2009.
[18] A. D. Murugan, H. E. Gamel, M. O. Damen, and G. Caire, "A unified framework for tree search decoding: Rediscovering the sequential decoder," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 933–953, Mar. 2006.
[19] R. Gowaikar and B. Hassibi, "Statistical pruning for near-maximum likelihood decoding," *IEEE Trans. Signal Process.*, vol. 55, no. 6, pp. 2661–2675, Jun. 2007.
[20] J.-S. Kim, S.-H. Moon, and I. Lee, "A new reduced complexity ML detection scheme for MIMO systems," *IEEE Trans. Commun.*, vol. 58, no. 4, pp. 1302–1310, Apr. 2010.

Regenerative Cooperative Diversity Networks With Co-channel Interference: Performance Analysis and Optimal Energy Allocation

Salama S. Ikki, *Member, IEEE*, P. Ubaidulla, *Member, IEEE*, and Sonia Aïssa, *Senior Member, IEEE*

***Abstract*—A study of the effects of co-channel interference on a multi-relay system with decode-and-forward (DF) protocol is presented. Orthogonal relaying is considered, and all relays that correctly decode the message in the broadcasting phase participate in the adaptive relaying phase. First, the effective signal-to-interference-plus-noise ratio (SINR) at the receiver is derived. Then, considering outage as the performance metric, we obtain exact closed-form expression for the outage probability. Simple and general asymptotic expressions for the outage probability, which explicitly show the coding and the diversity gains, are also derived and discussed. Furthermore, we present optimal energy-allocation schemes for minimizing outage under different resource constraints. Monte Carlo simulations are further provided to confirm the analytical results and illustrate the outage performance for different interference conditions and optimization schemes.**

***Index Terms*—Co-channel interference, cooperative diversity, decode-and-forward (DF) relaying, optimal energy allocation, outage probability, Rayleigh fading channels.**

I. Introduction

The performance of cooperative diversity networks has been widely examined with respect to the number, selection, and placement of relays, and various resource-allocation schemes have been proposed. These studies have been mostly based on the assumption of additive white Gaussian noise (AWGN) [1], [2]. However, in wireless networks with dense frequency reuse, co-channel interference usually dominates the AWGN. Furthermore, since all the relays may use the same carrier frequency, co-channel interference can potentially exist in every link in the network.

Some of the works on performance analysis of cooperative diversity networks have focused on the effect of co-channel interference in terms of signal-to-interference ratio (SIR) while neglecting the effect of AWGN [3]. However, the performance analysis in terms of SIR and the asymptotic analysis based on high signal-to-noise ratio (SNR) are not accurate in the low signal-to-interference-plus-noise ratio (SINR) regime, where AWGN dominates the interference.

In [4], Ikki and Aïssa examined the effect of co-channel interference in dual-hop amplify-and-forward relaying networks with nonidentical Rayleigh fading. The outage probability of dual-hop decode-and-forward (DF) relaying was also analyzed in [5], assuming the specific scenario in which the destination is corrupted by co-channel interference while the relay is only perturbed by AWGN. A more general scenario, where all nodes are affected by co-channel interference and all channels undergo Rayleigh fading, was considered in

Manuscript received September 12, 2011; revised June 8, 2012; accepted September 19, 2012. Date of publication October 12, 2012; date of current version February 12, 2013. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada and in part by King Abdullah University of Science and Technology. The review of this paper was coordinated by Prof. M. Uysal.

S. S. Ikki and S. Aïssa are with Institut National de la Recherche Scientifique (INRS), University of Quebec, Montreal QC H5A 1K6, Canada (e-mail: ikki@emt.inrs.ca; aissa@emt.inrs.ca).

P. Ubaidulla is with the Computer, Electrical and Mathematical Sciences and Engineering Division, King Abdullah University of Science and Technology, Thuwal 239955, Saudi Arabia (e-mail: ubaidulla@ieee.org).

Digital Object Identifier 10.1109/TVT.2012.2224679