Layered Tabu Search Algorithm for Large-MIMO Detection and a Lower Bound on ML Performance

N. Srinidhi, *Student Member, IEEE,* Tanumay Datta, *Student Member, IEEE,*
A. Chockalingam, *Senior Member, IEEE,* and B. Sundar Rajan, *Senior Member, IEEE*

***Abstract*—In this letter, we are concerned with low-complexity detection in large multiple-input multiple-output (MIMO) systems with tens of transmit/receive antennas. Our new contributions in this letter are two-fold. First, we propose a low-complexity algorithm for large-MIMO detection based on a *layered low-complexity local neighborhood search*. Second, we obtain a lower bound on the maximum-likelihood (ML) bit error performance using the local neighborhood search. The advantages of the proposed ML lower bound are i) it is easily obtained for MIMO systems with large number of antennas because of the inherent low complexity of the search algorithm, ii) it is tight at moderate-to-high SNRs, and iii) it can be tightened at low SNRs by increasing the number of symbols in the neighborhood definition. The proposed detection algorithm based on the layered local neighborhood search achieves bit error performances which are quite close to this lower bound for large number of antennas and higher-order QAM.**

***Index Terms*—Large-MIMO detection, local neighborhood search, QR decomposition, ML lower bound, higher-order QAM, high spectral efficiency.**

I. Introduction

LARGE-MIMO systems with tens of transmit and receive antennas are of interest because of the high capacities theoretically predicted in them [1],[2],[3]. Such large number of antennas can be employed in large/medium sized communication terminals like set top boxes, laptops and TVs for spectrally efficient wireless delivery of high data rate applications (e.g., wireless HDTV/3DTV distribution, broadcast/multicast video, interactive gaming). Evolution of WiFi standards to IEEE 802.11ac, which aims to achieve multi-gigabit rate transmissions in 5 GHz band, considers 16×16 MIMO operation (e.g., 16×16 MIMO indoor channel sounding measurements at 5 GHz have been reported in [4] for consideration in WiFi standards). Also, there is growing maturity of compact antennas/RF design suited for large-MIMO systems. For example, 24 and 36 antennas mounted in cubes of dimensions 8cm$\times$8cm$\times$8cm and 12cm$\times$12cm$\times$12cm, respectively, for MIMO applications have been reported in [5]. Low-complexity near-optimal processing at the receiver for large-MIMO systems is challenging. Research in low-complexity receive processing (e.g., MIMO detection, channel estimation) techniques that can lead to practical realization of large-MIMO systems is both nascent as well as promising. Recently, certain algorithms from machine learning/artificial intelligence have been shown to achieve near-optimal performance in large-MIMO systems with tens of antennas at low complexities [6]-[11][1]. In [8], near-optimal detection in a 50×50 MIMO system with BPSK was reported using a Gibbs sampling based detection algorithm. In [9], near-optimal detection performance in a 64×64 MIMO system, again with BPSK modulation, was reported using a factor graph based belief propagation (BP) algorithm that employed a Gaussian approximation of the interference. In [10],[11], a *tabu search algorithm*, which is a local neighborhood search algorithm, was shown to achieve near-optimal performance in large-MIMO systems for 4-QAM modulation, but its performance was far from optimum for higher-order QAM. Tabu search [12],[13] has been proposed previously for multiuser detection [14],[15],[16] and MIMO detection [17]. The tabu search in [17] uses fixed tabu period (fixed tabu search), whereas we employ reactive tabu search where tabu period is adapted. List decoding in [18] has shown good potential for large MIMO systems.

Our first new contribution in this letter is that we propose a *layering approach in conjunction with the low-complexity tabu search* which significantly improves higher-order QAM performance (e.g., 16-QAM, 64-QAM) in large-MIMO systems, bringing its performance much closer to the maximum-likelihood (ML) performance compared to the basic tabu search without layering [19]. The layered structure is inspired by previously suggested approaches based on successive cancellation (or decision feedback) systems [20],[21],[22], along with the use of QR decomposition for detection and detection ordering [21]-[25]. However, as opposed to cancellation, the proposed scheme can update the solution vector for all symbols under consideration within the specific layer.

In order to assess how well the proposed layered tabu search algorithm performs w.r.t. to the true ML performance in large-MIMO systems (i.e., for large n_t, where n_t denotes the number of transmit antennas), we resort to obtaining bounds on the ML performance which are computable at low complexities. This is because predicting the ML performance either through a brute-force search or by using sphere decoding (SD) is prohibitively complex for large-MIMO systems (like 32×32, 64×64 V-BLAST MIMO systems). Bounding of performance and minimum distances for breadth-first tree search algorithms in multiuser CDMA systems has been addressed in [26]. Upper bounds on the ML performance based on union bounding

Paper approved by S. Ariyavisitakul, the Editor for Wireless Communications of the IEEE Communications Society. Manuscript received March 9, 2011; revised April 18, 2011.

This work in part was presented in the IEEE International Symposium on Information Theory (ISIT'2009), Seoul, July 2009, and in the IEEE Global Communications Conference (GLOBECOM'2010), Miami, December 2010.

The authors are with the Department of ECE, Indian Institute of Science, Bangalore 560012, India (e-mail: tan.swapnil@gmail.com, {srinidhi, achockal, bsrajan}@ece.iisc.ernet.in).

Digital Object Identifier 10.1109/TCOMM.2011.070511.110058

[1]Similar algorithms have been reported earlier in the context of multiuser detection in large CDMA systems.

are known [27]. However, the complexity of computing this bound is M^{n_t} [27], which is prohibitive for large n_t and M (M is the modulation alphabet size). Also, the tightness of such upper bounds for large n_t and M for a given SNR/BER is difficult to predict because of the lack of knowledge of true ML performance for those n_t, M, and SNR/BER values. This then brings the need for good low-complexity lower bounds on the ML performance for large n_t, so that nearness to the ML performance of large-MIMO detection algorithms (like the one proposed in this letter) can be predicted.

Consequently, our second new contribution in this letter is that we obtain a lower bound on the ML bit error performance using the neighborhood search in tabu search algorithm. The advantages of the proposed bound are i) it is easily obtained for MIMO systems with large n_t because of the inherent low complexity of the search algorithm, ii) it is tight at moderate to high SNRs, and iii) it can be tightened at low SNRs by increasing the number of symbols in the neighborhood definition. Interestingly, the proposed layered search algorithm for detection, termed as *layered tabu search (LTS)* algorithm, achieves bit error performances which are quite close to this lower bound for large n_t and higher-order QAM. For e.g., in a 32×32 V-BLAST MIMO system, the proposed LTS algorithm performs close to within 1.7 dB of the proposed ML lower bound at 10^{-3} BER for 16-QAM (128 bps/Hz).

II. System Model

Consider a V-BLAST MIMO system with n_t transmit and n_r receive antennas. The transmitted symbols take values from a modulation alphabet $\mathbb{A}$ (e.g., M-QAM/M-PSK). Let $\mathbf{x} \in \mathbb{A}^{n_t}$ denote the transmitted vector. Let $\mathbf{H} \in \mathbb{C}^{n_r \times n_t}$ denote the channel gain matrix, whose entries are assumed to be i.i.d. Gaussian with zero mean and unit variance. The received vector $\mathbf{y}$ is given by

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

where $\mathbf{n}$ is the noise vector whose entries are are modeled as i.i.d. $\mathbb{C}\mathcal{N}(0, \sigma^2)$. The ML detection rule is given by

$$\widehat{\mathbf{x}}_{ML} \;=\; \arg \min_{\mathbf{x} \in \mathbb{A}^{n_t}} \|\mathbf{y} - \mathbf{H}\mathbf{x}\|^2 \;=\; \arg \min_{\mathbf{x} \in \mathbb{A}^{n_t}} \phi(\mathbf{x}), \tag{2}$$

where $\phi(\mathbf{x}) \stackrel{\triangle}{=} \mathbf{x}^H \mathbf{H}^H \mathbf{H}\mathbf{x} - 2\Re\left(\mathbf{y}^H \mathbf{H}\mathbf{x}\right)$ is the ML cost. The computational complexity in (2) is exponential in n_t, which is prohibitive for large n_t. Our interest is to achieve near-ML performance for large n_t at low complexities for modulation alphabets including higher-order QAM. The proposed LTS algorithm, which is presented in the next section, is aimed at achieving these performance and complexity objectives for large n_t and higher-order modulations.

III. Proposed Layered Tabu Search Algorithm

In this section, we present the proposed LTS algorithm for large-MIMO detection. The proposed algorithm involves a strategy of detecting symbols in a layered manner, where in each layer a low-complexity local neighborhood search, namely, tabu search (TS) [10],[11], detects a sub-vector of the transmitted symbol vector. The sub-vector size is increased from one layer to the next layer. In addition, the detected sub-vector in a given layer is used to form the initializing solution for the search in the next layer. We first present the TS algorithm without layering in Sec. III-A, and present layered TS (LTS) algorithm in Sec. III-B.

A. Tabu Search Algorithm Without Layering

The TS algorithm [10],[11] starts with an initial solution vector, defines a neighborhood around it (i.e., defines a set of neighboring vectors based on a neighborhood criterion), and moves to the best vector among the neighboring vectors (even if the best neighboring vector is worse, in terms of ML cost, than the current solution vector; this allows the algorithm to escape from local minima). This process is continued for a certain number of iterations, after which the algorithm is terminated and the best among the solution vectors in all the iterations is declared as the final solution vector. In defining the neighborhood of the solution vector in a given iteration, the algorithm attempts to avoid cycling by marking the moves to solution vectors of the past few iterations as 'tabu' (i.e., prohibits these moves), which ensures efficient search of the solution space. The number of these past iterations is parametrized as the 'tabu period,' which is dynamically changed depending on the number of repetitions of the solution vectors that are observed in the search path.

Neighborhood Definition: Let M denote the cardinality of $\mathbb{A} = \{a_1, a_2, \cdots, a_M\}$. Define a set $\mathcal{N}(a_q)$, $q \in \{1, \cdots, M\}$, as a fixed subset of $\mathbb{A} \backslash a_q$, which we refer to as the *symbol-neighborhood* of a_q. We choose the cardinality of this set to be the same for all a_q, $q = 1, \cdots, M$; i.e., we take $|\mathcal{N}(a_q)| = N$, $\forall q$. Note that the maximum and minimum values of N are $M-1$ and 1, respectively. We choose the symbol neighborhood based on *Euclidean distance*, i.e., for a given symbol, those N symbols which are the nearest will form its neighborhood; the nearest symbol will be the first neighbor, the next nearest symbol will be the second neighbor, and so on. For e.g., $\mathbb{A} = \{-3, -1, 1, 3\}$ for 4-PAM, and choosing N to be 2, $\mathcal{N}(-3) = \{-1, 1\}$, $\mathcal{N}(-1) = \{-3, 1\}$, $\mathcal{N}(1) = \{-1, 3\}$, $\mathcal{N}(3) = \{1, -1\}$ are possible symbol-neighborhoods. Let $w_v(a_q)$, $v = 1, \cdots, N$ denote the vth element in $\mathcal{N}(a_q)$; i.e., we say $w_v(a_q)$ is the vth symbol-neighbor of a_q.

Let $\mathbf{x}^{(m)} = [x_1^{(m)}\, x_2^{(m)} \cdots x_{n_t}^{(m)}]$ denote the data vector belonging to the solution space in the mth iteration, where $x_i^{(m)} \in \mathbb{A}$. We refer to the vector

$$\mathbf{z}^{(m)}(u, v) \;=\; \left[z_1^{(m)}(u, v) \;\; z_2^{(m)}(u, v) \cdots z_{n_t}^{(m)}(u, v)\right], \tag{3}$$

as (u, v)th *vector-neighbor* (or simply the (u, v)th neighbor) of $\mathbf{x}^{(m)}$, $u = 1, \cdots, n_t$, $v = 1, \cdots, N$, if i) $\mathbf{x}^{(m)}$ differs from $\mathbf{z}^{(m)}(u, v)$ in the uth coordinate only, and ii) the uth element of $\mathbf{z}^{(m)}(u, v)$ is the vth symbol-neighbor of $x_u^{(m)}$. That is,

$$z_i^{(m)}(u, v) \;=\; \begin{cases} x_i^{(m)} & \text{for } i \neq u \\ w_v(x_u^{(m)}) & \text{for } i = u. \end{cases} \tag{4}$$

So we will have $n_t N$ vectors which differ from a given vector in the solution space in only one coordinate. These $n_t N$ vectors form the neighborhood of the given vector. An operation on $\mathbf{x}^{(m)}$ which gives $\mathbf{x}^{(m+1)}$ belonging to the

vector-neighborhood of $\mathbf{x}^{(m)}$ is called a *move*. The algorithm is said to execute a move (u, v) if $\mathbf{x}^{(m+1)} = \mathbf{z}^{(m)}(u, v)$. We note that the number of candidates to be considered for a move in any one iteration is $n_t N$. Also, the overall number of 'distinct' moves possible is $n_t MN$, which is the cardinality of the union of all moves from all M^{n_t} possible solution vectors. The tabu value of a move, which is a non-negative integer, means that the move cannot be considered for that many number of subsequent iterations.

Tabu Matrix: A *tabu_matrix* $\mathbf{T}$ of size $n_t M \times N$ is the matrix whose entries denote the tabu values of moves. For each coordinate of the solution vector (there are n_t coordinates), there are M rows in $\mathbf{T}$, where each row corresponds to one symbol in the modulation alphabet $\mathbb{A}$; the indices of the rows corresponding to the uth coordinate are from $(u-1)M+1$ to uM, $u \in \{1, \cdots, n_t\}$. The N columns of the $\mathbf{T}$ matrix correspond to the N symbol-neighbors of the symbol corresponding to each row. In other words, the (r, s)th entry of the *tabu_matrix*, $r = 1, \cdots, n_t M$, $s = 1, \cdots, N$, corresponds to the move (u, v) from $\mathbf{x}^{(m)}$ when $u = \lfloor \frac{r-1}{M} \rfloor + 1$, $v = s$ and $x_u^{(m)} = a_q$, where $q = mod(r-1, M) + 1$. The entries of the tabu matrix, which are non-negative integers, are updated in each iteration, and they are used to decide the direction in which the search proceeds (as described in the algorithm description below).

Tabu Search Algorithm: Let $\mathbf{g}^{(m)}$ be the vector which has the least ML cost found till the mth iteration of the algorithm. Let l_{rep} be the average length (in number of iterations) between two successive occurrences of a solution vector (repetitions). Tabu period, P, a dynamic non-negative integer parameter, is defined as follows: if a move is marked as tabu in an iteration, it will remain as tabu for P subsequent iterations unless the move results in a better solution. A binary flag, $lflag \in \{0, 1\}$, is used to indicate whether the algorithm has reached a local minimum in a given iteration or not; this flag is used in the evaluation of the stopping criterion of the algorithm. The algorithm starts with an initial solution vector $\mathbf{x}^{(0)}$. Set $\mathbf{g}^{(0)} = \mathbf{x}^{(0)}$, $l_{rep} = 0$, and $P = P_0$. All the entries of the *tabu_matrix* are set to zero. The following steps 1) to 3) are performed in each iteration. Consider mth iteration in the algorithm, $m \geq 0$.

Step 1): Initialize $lflag = 0$. The ML costs of the $n_t N$ neighbors of $\mathbf{x}^{(m)}$, $\phi(\mathbf{z}^{(m)}(u, v))$, $u = 1, \cdots, n_t$, $v = 1, \cdots, N$, are computed. Let

$$(u_1, v_1) \quad = \quad \arg \min_{u,v} \phi(\mathbf{z}^{(m)}(u, v)). \tag{5}$$

The move (u_1, v_1) is accepted if any one of the following two conditions is satisfied:

$$\phi(\mathbf{z}^{(m)}(u_1, v_1)) \quad < \quad \phi(\mathbf{g}^{(m)}) \tag{6}$$

$$\mathbf{T}((u_1 - 1)M + q, v_1) \quad = \quad 0, \tag{7}$$

where q is such that $a_q = x_{u_1}^{(m)}, a_q \in \mathbb{A}$. If move (u_1, v_1) is not accepted (i.e., neither of the conditions in (6) and (7) is satisfied), find (u_2, v_2) such that

$$(u_2, v_2) = \arg \min_{u,v: u \neq u_1, v \neq v_1} \phi(\mathbf{z}^{(m)}(u, v)), \tag{8}$$

and check for acceptance of the move (u_2, v_2). If this also cannot be accepted, repeat the procedure for (u_3, v_3), and so on. If all the $n_t N$ moves are tabu, then all the *tabu_matrix* entries are decremented by the minimum value in the *tabu_matrix*; this goes on till one of the moves becomes acceptable. Let (u', v') be the index of the neighbor with the minimum cost for which the move is permitted. Make

$$\mathbf{x}^{(m+1)} \quad = \quad \mathbf{z}^{(m)}(u', v'). \tag{9}$$

The variables q', q'', v'' are implicitly defined by $a_{q'} = x_{u'}^{(m)} = w_{v''}(x_{u'}^{(m+1)})$, and $a_{q''} = x_{u'}^{(m+1)}$, where $a_{q'}, a_{q''} \in \mathbb{A}$. It is noted that in this *Step 1* of the algorithm, essentially the best permissible vector-neighbor is chosen as the solution vector for the next iteration.

Step 2): The new solution vector obtained from *Step 1* is checked for repetition. For the linear vector channel model in (1), repetition can be checked by comparing the ML costs of the solutions in the previous iterations. If there is a repetition, the length of the repetition from the previous occurrence is found, the average length, l_{rep}, is updated, and the tabu period P is modified as $P = P+1$. If the number of iterations elapsed since the last change of the value of P exceeds βl_{rep}, for a fixed $\beta > 0$, make $P = \max(1, P-1)$. After a move (u', v') is accepted, if $\phi(\mathbf{x}^{(m+1)}) < \phi(\mathbf{g}^{(m)})$, make

$$\mathbf{T}((u'-1)M + q', v') \; = \; \mathbf{T}((u'-1)M + q'', v'') \; = \; 0, \qquad \mathbf{g}^{(m+1)} \; = \; \mathbf{x}^{(m+1)},$$

else

$$\mathbf{T}((u'-1)M + q', v') \; = \; \mathbf{T}((u'-1)M + q'', v'') \; = \; P+1, \qquad lflag = 1, \quad \mathbf{g}^{(m+1)} = \mathbf{g}^{(m)}.$$

It is noted that this *Step 2* of the algorithm implements the 'reactive' part in the search, by dynamically changing P.

Step 3): Update the entries of the *tabu_matrix* as

$$\mathbf{T}(r, s) \quad = \quad \max\{\mathbf{T}(r, s) - 1, 0\}, \tag{10}$$

for $r = 1, \cdots, n_t M$, $s = 1, \cdots, N$. The algorithm terminates in *Step 3* if the following stopping criterion is satisfied, else it goes back to *Step 1*.

Stopping criterion: The search algorithm described above is stopped if maximum number of iterations max_iter is reached. Also, if the current solution is a local minimum ($lflag = 1$) and the total number of repetitions of solutions is greater than max_rep, the algorithm is stopped. The solution of the algorithm would then be the vector with the least ML cost which has been found before the algorithm was stopped. The average per-symbol complexity of the TS algorithm is $O(n_t n_r)$, which is attractive for large-MIMO signal detection [10],[11]. Though this TS algorithm without layering has been shown to achieve near-ML performance in large n_t for 4-QAM, its performance in higher-order QAM is not as good [10],[11]. The LTS algorithm proposed in the following subsection improves higher-order QAM performance significantly for large n_t.

B. LTS Algorithm

In this subsection, we present the proposed LTS algorithm which performs detection in a layered manner, where the TS algorithm (presented in the previous subsection) is applied

in each layer. Let $\mathbf{U}$ denote the upper triangular matrix obtained from the QR decomposition of $\mathbf{H}$. Then, the objective equivalent to (2) will be to find the transmitted vector $\mathbf{x}$ which minimizes $\|\mathbf{U}(\mathbf{x}-\bar{\mathbf{x}})\|^2$, where

$$\bar{\mathbf{x}} \;=\; \mathbf{H}^{\dagger}\mathbf{y}, \tag{11}$$

and $\mathbf{H}^{\dagger}$ is the Moore-Penrose pseudo inverse of $\mathbf{H}$. Let u_{ij} denote the element in the ith row and jth column of the $\mathbf{U}$ matrix, and x_i denote the ith element of the vector $\mathbf{x}$.

The algorithm processes one layer at a time. It starts with the n_tth layer first. In the kth layer, $k = n_t, (n_t-1), (n_t-2), \cdots, 1$, the algorithm detects the (n_t-k+1)-sized sub-vector $[x_k, x_{k+1}, \cdots, x_{n_t}]$. We detect the symbols of this sub-vector jointly because they interfere with each other due to the structure of the $\mathbf{U}$ matrix. For e.g., since $\mathbf{U}$ is upper triangular, there will be no interference to the symbol x_{n_t} in the n_tth layer. In the (n_t-1)th layer, there will be one interferer x_{n_t}. In the (n_t-2)th layer there will be two interferers x_{n_t-1} and x_{n_t}, and so on in the subsequent layers. The joint detection method employed in each layer is the TS algorithm described in Sec. III-A. We propose to reduce the complexity further by skipping the joint detection search in a layer if a simple cancellation of interference due to the already detected symbols in the previous layer results in a good quality output. The resulting LTS algorithm is stated below.

Let $\check{\mathbf{x}}$ be the quantized version of $\bar{\mathbf{x}}$, i.e., each element in $\bar{\mathbf{x}}$ is rounded-off to its nearest symbol in the alphabet to get $\check{\mathbf{x}}$, so that $\check{\mathbf{x}} \in \mathbb{A}^{n_t}$. Let d_{min} be the minimum Euclidean distance between any two symbols in the alphabet $\mathbb{A}$. The steps performed in the kth layer, $k = n_t, (n_t-1), \cdots, 1$, are as follows:

Step 1): Calculate

$$r_k \;=\; \bar{x}_k - \sum_{l=k+1}^{n_t} \frac{u_{kl}}{u_{kk}}(\hat{x}_l - \bar{x}_l), \tag{12}$$

which is a cancellation operation that removes the interference due to the symbols detected in the previous layer (i.e., $\hat{x}_l$'s). Note that for $k = n_t$ (i.e., for the n_tth layer, which is processed first), there will be no 2nd term on the RHS in (12).

Step 2): Find the symbol in the alphabet $\mathbb{A}$ which is closest to r_k in Euclidean distance. Let this symbol be a_q.

i) If $|r_k - a_q| < \delta d_{min}$, $0 < \delta \leq 0.5$, then $\hat{x}_k = a_q$ ($\hat{x}_k$ is the detected symbol corresponding to x_k). Make $k = k-1$ and return to Step 1)[2].

ii) If $|r_k - a_q| \geq \delta d_{min}$, then set $\hat{x}_k = \check{x}_k$. Run the TS algorithm in Sec. III-A, by replacing $\mathbf{x}^{(0)}$ with $\tilde{\mathbf{x}}^{(0)}$, $\mathbf{H}$ with $\tilde{\mathbf{H}}$, $\mathbf{y}$ with $\tilde{\mathbf{y}}$, where $\tilde{\mathbf{x}}^{(0)}$, $\tilde{\mathbf{H}}$, $\tilde{\mathbf{y}}$ for the kth layer are taken as

$$\tilde{\mathbf{x}}^{(0)} \;=\; [\hat{x}_k, \hat{x}_{k+1}, \cdots, \hat{x}_{n_t}], \tag{13}$$

$$\tilde{\mathbf{H}} = \begin{bmatrix} u_{kk} & u_{k(k+1)} & \cdots & u_{kn_t} \\ 0 & u_{(k-1)(k-1)} & \cdots & u_{(k-1)n_t} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & u_{n_tn_t} \end{bmatrix}, \tag{14}$$

$$\tilde{\mathbf{y}} \;=\; \tilde{\mathbf{H}}\,[\bar{x}_k \;\; \bar{x}_{k+1} \;\; \cdots \;\; \bar{x}_{n_t}]^T. \tag{15}$$

The output vector from the TS algorithm is made as the updated $[\hat{x}_k, \hat{x}_{k+1}, \cdots, \hat{x}_{n_t}]$ sub-vector. Make $k = k-1$ and return to Step 1).

After processing all the n_t layers, the vector $\hat{\mathbf{x}} = [\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_{n_t}]$ is declared as the final detected data vector. We note that in the TS algorithm without layering in Sec. III-A, TS is carried out once on the full $n_t \times n_r$ system model. Whereas, in the LTS algorithm TS is performed multiple times, once on each layer (depending on the effectiveness of the interference cancellation performed in that layer as per eqn. (12)). The dimension of the problem gets increased by 1 from one layer to the next.

C. Detection with Ordering

A way to improve performance is to follow an optimum order while detecting the symbols. We need to find an optimum order $(p_1, p_2, \cdots, p_{n_t})$ which is a permutation of $(1, 2, \cdots, n_t)$. We obtain the optimum ordering based on the post-detection SNR of the symbols as follows.

Perform the following steps for $i = n_t, \cdots, 1$ with $\mathbf{H}_{n_t} = \mathbf{H}$: *i*) Find $\mathbf{H}_i^{\dagger}$, the Moore-Penrose pseudo-inverse of $\mathbf{H}_i$, where $\mathbf{H}_i$ is obtained by zeroing $(p_{i+1}, p_{i+2}, \cdots, p_{n_t})$ columns of $\mathbf{H}$; *ii*) Find p_i, the index that corresponds to the row with the least norm among all rows of $\mathbf{H}_i^{\dagger}$. Detection is then carried out in the following order: p_{n_t}, p_{n_t-1}, p_{n_t-2}, and so on.

IV. A Lower Bound on ML Performance

In this section, we obtain a lower bound on the ML bit error performance using the neighborhood search in the TS algorithm in Sec. III-A. To find the lower bound, we will use the actually transmitted vector $\mathbf{x}$ as the initial vector in the TS algorithm. Define n-symbol neighborhood of a certain vector to be the set of all vectors which differ from that vector in i coordinates, $1 \leq i \leq n$.

With the transmitted vector $\mathbf{x}$ as the initial vector, TS algorithm is run and the output solution vector is obtained. Let $\mathbf{x}_{TS}$ denote the output solution vector obtained from the TS algorithm, and let e_{TS} denote the number of symbol errors in $\mathbf{x}_{TS}$ compared to $\mathbf{x}$. For each realization in the simulations, $\mathbf{x}$, $\mathbf{x}_{TS}$, and hence e_{TS} are known. Let $\mathcal{N}_{\mathbf{x}}$ denote the n-symbol neighborhood of $\mathbf{x}$ (defined in the previous paragraph). Also, let $\mathbf{x}_{ML}$ denote the true ML vector, and e_{ML} denote the number of symbol errors in $\mathbf{x}_{ML}$ (which we do not know, and seek to get a lower bound on). Note that the solution vector $\mathbf{x}_{TS}$ may or may not lie in the n-symbol neighborhood of $\mathbf{x}$, $\mathcal{N}_{\mathbf{x}}$. Since the TS algorithm chooses $\mathbf{x}_{TS}$ to be the vector with the least ML cost among all the tested vectors, if $\mathbf{x}_{TS} \notin \mathcal{N}_{\mathbf{x}}$, then $\mathbf{x}_{ML} \notin \mathcal{N}_{\mathbf{x}}$. Also, by the definition of $\mathcal{N}_{\mathbf{x}}$, the number of errors in $\mathbf{x}_{TS}$ and $\mathbf{x}_{ML}$ are lower bounded by $n+1$, i.e., $e_{TS}, e_{ML} \geq n+1$. So, in the simulations, if $e_{TS} \geq n+1$ in a given realization, then take e_{ML} as $n+1$ as a lower bound

[2]Execution of this part of the step essentially skips the joint detection using TS. Nearness of r_k to an element in $\mathbb{A}$ to within δd_{min}, $0 < \delta \leq 0.5$ is used as the criterion to decide to carry out or skip TS in layer k. Figure 3 shows the BER performance and complexity (in average number of real operations per symbol) of the LTS algorithm as a function of δ for 16×16 V-BLAST MIMO with 16-QAM at an SNR of 19 dB. It can be seen that the BER improves as δ is decreased from 0.5 towards 0. This is because a smaller δ means increased chance of carrying out joint detection using TS in Step 2-*ii*, which results in improved performance while incurring increase in complexity. We have used $\delta = \frac{1}{4}$ in the LTS simulations.

on the number symbol errors in the ML vector. On the other hand, if $\mathbf{x}_{TS} \in \mathcal{N}_{\mathbf{x}}$ which implies that $e_{TS} = \kappa$, $1 \leq \kappa \leq n$, then two cases are possible: 1) $\mathbf{x}_{TS}$ is the ML vector, and 2) $\mathbf{x}_{TS}$ is not the ML vector. In case 1) $e_{TS} = e_{ML} = \kappa$, and in case 2) $e_{TS} = \kappa$ and $\mathbf{x}_{ML}$ being outside $\mathcal{N}_{\mathbf{x}}$, $e_{ML} \geq n+1$. So, in the simulations, if $e_{TS} = \kappa$, $\kappa \leq n$, then take e_{ML} as κ as a lower bound. Lastly, if $e_{TS} = 0$, then $\mathbf{x}_{TS} = \mathbf{x}$ which may or may not be the ML vector; in such a realization, take $e_{ML} = 0$ as a lower bound. In summary, in the simulations,

- if $e_{TS} = \kappa$, $\kappa \leq n$, then take e_{ML} as κ, and
- if $e_{TS} \geq n+1$, then take e_{ML} as n+1,

which results in a lower bound on the ML symbol error performance. Since the number of symbol errors is a lower bound on the number of bit errors, it is a bit error bound as well.

A. Results and Discussions on the Lower Bound

We simulated the tabu search algorithm for a 16×16 V-BLAST MIMO system and obtained the proposed lower bounds for 4-QAM, 16-QAM, and 64-QAM. In the simulations, we have applied the algorithm on the real-valued system model corresponding to (1), i.e., on the system model $\mathbf{y}_r = \mathbf{H}_r \mathbf{x}_r + \mathbf{n}_r$, where

$$\mathbf{H}_r = \begin{bmatrix} \Re(\mathbf{H}) & -\Im(\mathbf{H}) \\ \Im(\mathbf{H}) & \Re(\mathbf{H}) \end{bmatrix}, \quad \mathbf{y}_r = \begin{bmatrix} \Re(\mathbf{y}) \\ \Im(\mathbf{y}) \end{bmatrix},$$
$$\mathbf{x}_r = \begin{bmatrix} \Re(\mathbf{x}) \\ \Im(\mathbf{x}) \end{bmatrix}, \quad \mathbf{n}_r = \begin{bmatrix} \Re(\mathbf{n}) \\ \Im(\mathbf{n}) \end{bmatrix}.$$

The following parameters are used in the tabu search algorithm simulations: $max\_rep = 75$, $max\_iter = 300$, $\beta = 0.1$, $P_0 = 2$ for 4-QAM; $max\_rep = 250$, $max\_iter = 1000$, $\beta = 0.01$, $P_0 = 2$ for 16-QAM; and $max\_rep = 1000$, $max\_iter = 3000$, $\beta = 0.01$, $P_0 = 2$ for 64-QAM. Perfect channel state information at the receiver (CSIR) is assumed. In Fig. 1, we plot the ML lower bounds for $n = 1, 2, 3, 4$ and compare them with the actual ML performance obtained by sphere decoding. We note that sphere decoding simulations for the considered 16×16 system took long simulation run times. From Fig. 1, it can be observed that the proposed lower bound is quite tight (within just 0.5 dB) for BERs less than 10^{-2}, and gets increasingly tighter for lesser BERs. We note that because of the low-complexity of the tabu search algorithm, these bounds are easily computed for large n_t. Also, even at low SNRs the bound gets increasingly tighter with increasing n.

An Approximate Prediction of ML Performance: The improved tightness of the proposed lower bound for increasing n is observed to be quite significant at low SNRs in Fig. 1. However, a large n means increased complexity. As a low-complexity alternative, we approximate the true ML error performance to be the error performance of the tabu search solution when the transmitted vector $\mathbf{x}$ is used as the initial vector, i.e., we assume $e_{ML} = e_{TS}$. From the previous discussion on the lower bound, we note that e_{TS} indeed corresponds to an upper bound to the proposed ML lower bound. But this upper bound need not be a lower or upper bound to true ML performance. So we refer to the performance obtained by equating e_{TS} to e_{ML} as an 'approximate ML performance.' It can be noted that, complexity-wise, like the proposed lower bound, the approximate ML performance is also easily obtained for large n_t. In Fig. 2, we compare the lower bound, approximate ML, and the sphere decoder performances for 16×16 V-BLAST MIMO with 4-, 16- and 64-QAM. It is seen that the proposed approximate ML performance is quite close to the actual ML performance even at low SNRs.

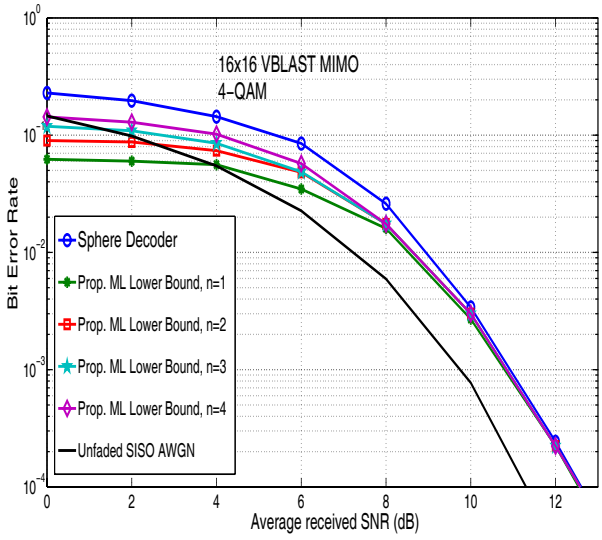

Fig. 1. Comparison of the proposed lower bound on ML performance for $n = 1, 2, 3, 4$ with the ML performance predicted by sphere decoder for 16×16 V-BLAST MIMO with 4-QAM.

V. Proposed LTS Performance in Large-MIMO

Large-System Behavior of LTS: Figure 4 shows the performance of the LTS algorithm with ordering in $n_t \times n_r$ V-BLAST MIMO systems with $n_t = n_r = 4, 8, 32$ and 16-QAM. The following parameters are used in all LTS simulations in each layer: $max\_rep = 10$, $max\_iter = \beta = 20$ for 4-QAM, $max\_rep = 10$, $max\_iter = \beta = 100$ for 16-QAM, $max\_rep = 20$, $max\_iter = \beta = 200$ for 64-QAM, $P_0 = 1$, and $\delta = 1/4$. In Fig. 4, we see that the LTS algorithm exhibits large-system behavior, where the achieved BER performance improves with increasing $n_t = n_r$. Figure 4 also shows the BER performance of LTS in 32×32 MIMO with 8-PSK, where the following parameters are used: $max\_rep = 10$, $max\_iter = 50$, $\beta = 50$, $P_0 = 1$, $\delta = 1/8$. It is seen that LTS performance in 32×32 MIMO with 8-PSK is quite close to 8-PSK performance in unfaded SISO AWGN.

BER/Complexity Comparison with TS with No Layering: Figure 5 shows a comparison between the BER performances of the proposed LTS algorithm without and with ordering, and the TS algorithm without layering in a 32×32 V-BLAST MIMO system with 16-QAM and 64-QAM. It can be seen that compared to TS without layering, the proposed layered TS approach significantly improves the BER performance. For e.g., TS without layering needs 24 dB SNR to achieve 10^{-3} BER for 16-QAM, whereas the proposed LTS algorithm with ordering achieves the same BER at 19 dB, which amounts to an SNR gain of 5 dB. For 64-QAM, this SNR gain is even higher. In Fig. 6, we show a complexity comparison between

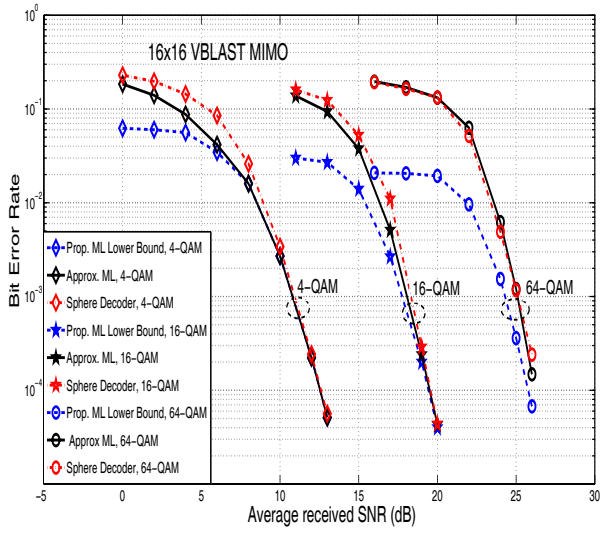

Fig. 2. Comparison of the proposed lower bound on ML performance for $n = 1$ and the 'approximate ML' performance with the ML performance predicted by sphere decoder for 16×16 V-BLAST MIMO with 4-, 16- and 64-QAM.

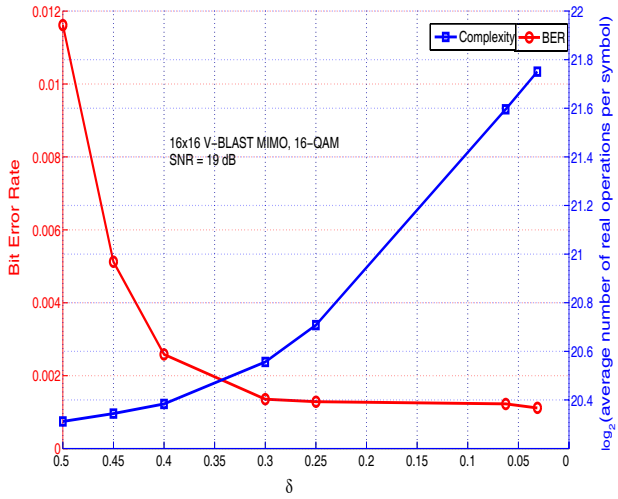

Fig. 3. BER performance and complexity of the LTS algorithm as a function of δ for 16×16 V-BLAST MIMO with 16-QAM at an SNR of 19 dB.

the algorithms, where we have plotted the average number of real operations as a function $n_t = n_r$ for 16-QAM at 10^{-2} BER. Though the order of complexity for TS without layering is less, the constant is high; at $n_t = n_r = 16$ the proposed LTS with ordering has almost similar complexity as that of TS without layering. Also, LTS without ordering has about the same complexity as TS without layering for $n_t = n_r = 32$; LTS without ordering, however, achieves better performance than that of TS without layering. Figure 5 also presents a BER comparison with a tree structured detector in [28], namely, SIC via breadth-first search (SIC-BFS). In Fig. 5, the performance of LTS with layering is better than the performance of SIC-BFS in [28].

Comparison with Other Detectors: In Table-I, we present a comparison of the performance (in terms of SNR required to achieve 10^{-2} BER) and complexity (in terms of per-symbol complexity in number of real operations at 10^{-2} BER) comparison of the LTS algorithm with those of two low-complexity variants of sphere decoding. The first one is the sphere decoder (SD) algorithm in [29], which uses lattice boundary awareness to reduce complexity. The second one is the fixed complexity sphere decoder (FSD) in [30], which restricts the search in such a way that the complexity of the algorithm is fixed irrespective of the SNR. The FSD in [30] can result in sub-optimal performance because of limiting the search to achieve fixed complexity at all SNRs. In [30], FSD has been shown to achieve almost the same performance as that of the SD performance for 4×4 V-BLAST MIMO with 4-, 16- and 64-QAM, and for 8×8 V-BLAST MIMO with 4- and 16-QAM, at lower complexities than SD at low to moderate SNRs. We first simulated the FSD algorithm and repeated its performance in [30] for 4×4 and 8×8 V-BLAST MIMO with 4- and 16-QAM. In addition, we ran FSD simulations for 16×16 and 32×32 V-BLAST MIMO with 16-QAM. The FSD performance results for 8×8, 16×16 and 32×32 V-BLAST MIMO with 16-QAM obtained from the simulations and the corresponding complexities are shown in Table-I. The corresponding results for TS (without layering), proposed LTS (without and with ordering), and SD in [29] are also shown. From Table-I, it is observed that $i)$ in 8×8 MIMO, all detectors achieve almost the same performance, with FSD having the least complexity among them and LTS having lesser complexity than TS (without layering), $ii)$ in 16×16 MIMO, the FSD, LTS, and TS (without layering) detectors perform worse than SD in [29] by about 0.5 dB, with LTS without ordering having the least complexity and SD in [29] having the highest complexity, and $iii)$ in 32×32 MIMO, LTS without ordering achieves better performance than TS (without layering) and FSD, at significantly lesser complexity than FSD and at similar complexity of TS (without layering). We do not give the 32×32 MIMO results for SD in [29] because of its prohibitively high complexity to simulate in such large dimensions (64 real dimensions in 32×32 MIMO with QAM). So, LTS without ordering is quite attractive in performance and complexity for large n_t (e.g., 16×16 and 32×32 MIMO with 16-QAM).

Channel Estimation and Turbo Coded Performance: In this subsection, we relax the perfect CSIR assumption made in the previous simulations, by considering a training based iterative channel estimation/LTS detection scheme. Transmission is carried out in frames (similar to the transmission scheme in [7]), where each frame consists of a pilot part followed by a data part. The pilot part consists of n_t pilot channel uses, where one $n_t \times n_t$ identity matrix is transmitted as pilot for channel estimation purposes. The data part consists of n_d data channel uses, where n_d number of n_t-sized data symbol vectors are transmitted. One frame length, T (taken to be the channel coherence time), is $T = n_t + n_d$ channel uses [31]. The iterative channel estimation/detection scheme works as follows: $i)$ obtain an MMSE estimate of the channel matrix during the pilot part, $ii)$ use the estimated channel matrix to detect the data vectors using LTS in the data part, $iii)$ use the detected data vectors to estimate the channel matrix again, and $iv)$ iterate between channel estimation and LTS detection for a certain number of times.

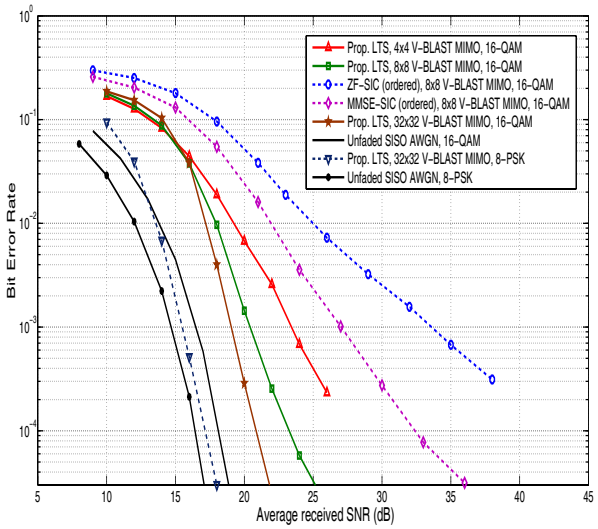

Fig. 4. BER performance of the proposed LTS algorithm with ordering in V-BLAST MIMO for $n_t = n_r = 4, 8, 32$ with 16-QAM, and $n_t = n_r = 32$ with 8-PSK.

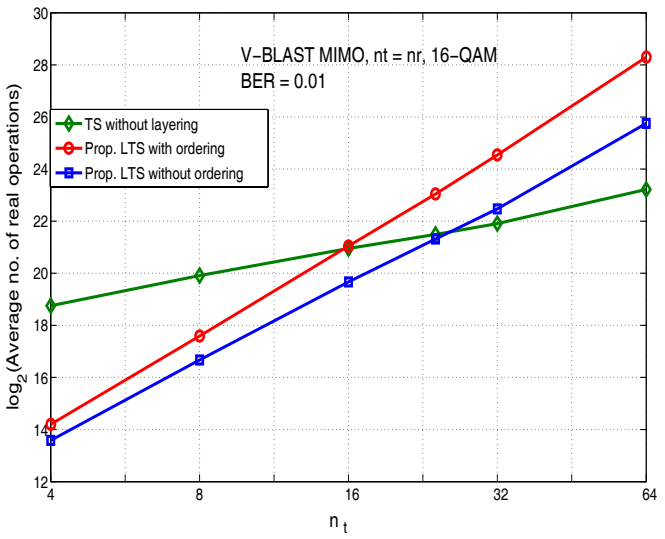

Fig. 6. Complexity comparison between LTS with ordering, LTS without ordering, and TS without layering in terms of average number of real operations for different $n_t = n_r$ with 16-QAM at 10^{-2} BER.

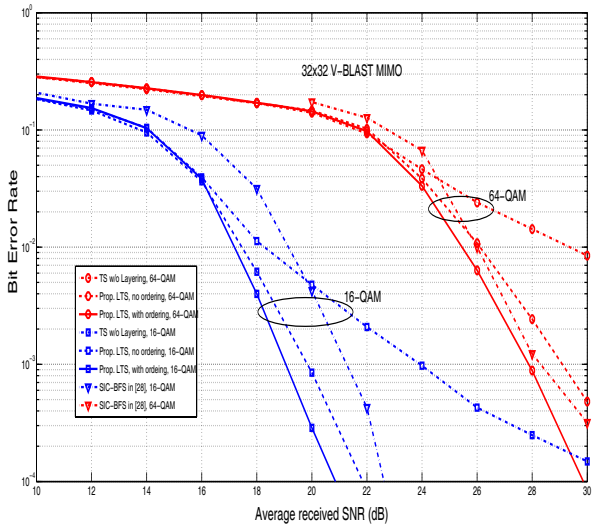

Fig. 5. BER comparison between the proposed LTS algorithm without and with ordering, TS without layering, and SIC-BFS in [28] for 32×32 V-BLAST MIMO with 16-QAM and 64-QAM.

We generated soft decision outputs from LTS output as follows. Let $\mathbf{d} = [\widehat{x}_1, \widehat{x}_2, \cdots, \widehat{x}_{n_t}]$, $\widehat{x}_i \in \mathbb{A}$ denote the detected output symbol vector from the LTS algorithm. Let the symbol $\widehat{x}_i$ map to the bit vector $\mathbf{b}_i = [b_{i,1}, b_{i,2}, \cdots, b_{i,K}]^T$, where $K = \log_2 |\mathbb{A}|$, and $b_{i,j} \in \{+1, -1\}$, $i = 1, 2, \cdots, n_t$ and $j = 1, 2, \cdots, K$. Let $\tilde{b}_{i,j} \in \mathbb{R}$ denote the soft value for the jth bit of the ith symbol. Given $\mathbf{d}$, we need to find $\tilde{b}_{i,j}$, $\forall\,(i,j)$. Note that the quantity $\|\mathbf{y} - \mathbf{Hd}\|^2$ is inversely related to the likelihood that $\mathbf{d}$ is indeed the transmitted symbol vector. Let the $\mathbf{d}$ vector with its jth bit of the ith symbol forced to +1 be denoted as vector $\mathbf{d}_i^{j+}$. Likewise, let $\mathbf{d}_i^{j-}$ be the vector $\mathbf{d}$ with its jth bit of the ith symbol forced to -1. Then the quantities $\|\mathbf{y} - \mathbf{Hd}_i^{j+}\|^2$ and $\|\mathbf{y} - \mathbf{Hd}_i^{j-}\|^2$ are inversely related to the likelihoods that the jth bit of the ith transmitted symbol is +1 and -1, respectively. So, if $\|\mathbf{y} - \mathbf{Hd}_i^{j-}\|^2 - \|\mathbf{y} - \mathbf{Hd}_i^{j+}\|^2$ is +ve (or -ve), it indicates that the jth bit of the ith transmitted symbol has a higher likelihood of being +1 (or -1). So, the quantity $\tilde{b}_{i,j} \stackrel{\triangle}{=} \|\mathbf{y} - \mathbf{Hd}_i^{j-}\|^2 - \|\mathbf{y} - \mathbf{Hd}_i^{j+}\|^2$, appropriately normalized to avoid unbounded increase for increasing n_t, can be a good soft value for the jth bit of the ith symbol. A normalization factor of $\left(\frac{\lambda_{i,j}}{2}\right)^2 \|\mathbf{h}_i\|^2$ is found to give good performance, where $\lambda_{i,j}$ is given by $\mathbf{d}_i^{j-} = \mathbf{d}_i^{j+} + \lambda_{i,j}\mathbf{e}_i$ (since $\mathbf{d}_i^{j-}$ and $\mathbf{d}_i^{j+}$ differ in only one location), and $\mathbf{e}_i$ is the vector with 1 in the ith coordinate and zeroes in all the other coordinates. In coded BER simulations, we have used $\tilde{b}_{i,j}$ normalized by the above factor as soft inputs to the decoder.

In Fig. 7, we present the simulated turbo coded BER performance of LTS detection, with perfect CSIR and estimated CSIR (using 2 iterations between MMSE channel estimation and LTS detection), for 18×18 V-BLAST MIMO, 16-QAM, and rate-1/2 turbo code. Coded BER results for the estimated CSIR scheme with $n_d = 54, 108, 162$ (corresponding to coherence times of $T = 72, 126, 180$, and spectral efficiencies of 27 bps/Hz, 30.8 bps/Hz, 32.4 bps/Hz, respectively) are shown. From Fig. 7, it is seen that, as expected, performance with estimated CSIR degrades compared to that with perfect CSIR. For e.g., with $T = 72$, the performance degrades by about 2.5 dB at 10^{-3} coded BER compared to perfect CSIR performance. However, this performance degradation reduces for increasing coherence times (i.e., slow fading); e.g., for $T = 126, 180$, the degradation compared to perfect CSIR performance is about 1.5 dB. In other words, the BER and bps/Hz with estimated CSIR gets increasing closer to those with perfect CSIR for increasing channel coherence times, T, which are typical in low-/no-mobility applications.

VI. Conclusion

We made two new contributions in this letter. First, we presented a layered detection approach in conjunction with a low-complexity local neighborhood tabu search and showed

TABLE I
COMPLEXITY AND PERFORMANCE COMPARISON OF THE LTS ALGORITHM WITH OTHER ALGORITHMS. $\star$: 32×32 MIMO RESULTS FOR SD IN [29] ARE NOT GIVEN BECAUSE OF ITS PROHIBITIVELY HIGH COMPLEXITY TO SIMULATE IN SUCH LARGE DIMENSIONS (64 REAL DIMENSIONS IN 32×32 MIMO WITH QAM).

Algorithm	Per-symbol-complexity (PSC) in number of real operations$\times 10^3$ and SNR required to achieve 10^{-2} BER in 16-QAM					
	8×8		16×16		32×32	
	PSC	SNR	PSC	SNR	PSC	SNR
TS (without layering)	123.432	18 dB	126.832	17.5 dB	128.227	18.2 dB
LTS (without ordering)	14.263	18 dB	57.052	17.4 dB	179.049	17.4 dB
LTS (with ordering)	24.644	18 dB	135.082	17.4 dB	764.297	17.2 dB
FSD in [30]	11.341	17.9 dB	302.277	17.6 dB	143735.349	17.8 dB
SD in [29]	11.433	17.9 dB	6227.990	17 dB	$\star$	$\star$

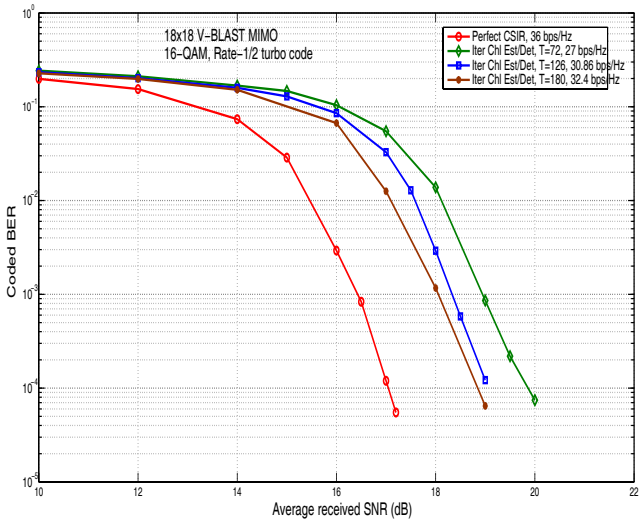

Fig. 7. Turbo coded performance of LTS with perfect CSIR and estimated CSIR for 18×18 V-BLAST MIMO, 16-QAM, rate-1/2 turbo code. $T = 72, 126, 180$ for estimated CSIR.

that it indeed works very well in terms of both performance as well as complexity in MIMO systems with large number of antennas. Performance-wise, we showed that it achieves close to ML performance, and complexity-wise it scales well for large number of antennas. Such good performance and complexity features of the proposed algorithm are quite attractive for large-MIMO systems. Second, we proposed a lower bound on ML bit error performance based on the neighborhood search of the tabu search algorithm, which is a novel and effective approach. The proposed bound is easy to obtain for large-MIMO systems, and it can serve as a benchmark for evaluating the nearness to ML performance achieved by different large-MIMO detection algorithms.

REFERENCES

[1] I. E. Telatar, "Capacity of multi-antenna Gaussian channels," *European Trans. Telecommun.,* vol. 10, no. 6, pp. 585-595, Nov. 1999.

[2] A. Paulraj, R. Nabar, and D. Gore, *Introduction to Space-Time Wireless Communications*. Cambridge University Press, 2003.

[3] H. Bölcskei, D. Gesbert, C. B. Papadias, and Alle-Jan van der Veen, editors. *Space-Time Wireless Systems: From Array Processing to MIMO Communications*. Cambridge University Press, 2006.

[4] G. Breit *et al.*, 802.11ac Channel Modeling, doc. IEEE 802.11-09/0088r0, submission to Task Group TGac, 19 Jan. 2009.

[5] C.-Y. Chiu, J.-B. Yan, and R. D. Murch, "24-port and 36-port antenna cubes suitable for MIMO wireless communications," *IEEE Trans. Antennas Propagation*, vol. 56, no. 4, pp. 1170-1176, Apr. 2008.

[6] K. Vishnu Vardhan, S. K. Mohammed, A. Chockalingam, and B. Sundar Rajan, "A low-complexity detector for large MIMO systems and multicarrier CDMA systems," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 3, pp. 473-485, Apr. 2008.

[7] S. K. Mohammed, A. Zaki, A. Chockalingam, and B. Sundar Rajan, "High-rate space-time coded large-MIMO systems: low-complexity detection and channel estimation," *IEEE J. Sel. Topics Signal Process.*, vol. 3, no. 6, pp. 958-974, Dec. 2009.

[8] M. Hansen, B. Hassibi, A. G. Dimakis, and W. Xu, "Near-optimal detection in MIMO systems using Gibbs sampling," in *Proc. IEEE ICC*, Dec. 2009.

[9] P. Som, T. Datta, A. Chockalingam, and S. Sundar Rajan, "Improved large-MIMO detection based on damped belief propagation," in *Proc. IEEE ITW*, Jan. 2010.

[10] N. Srinidhi, S. K. Mohammed, A. Chockalingam, and B. Sundar Rajan, "Low-complexity near-ML decoding of large non-orthogonal STBCs using reactive tabu search," in *Proc. IEEE ISIT*, July 2009.

[11] N. Srinidhi, S. K. Mohammed, A. Chockalingam, and B. Sundar Rajan, "Near-ML signal detection in large-dimension linear vector channels using reactive tabu search," arXiv:0911.4640v1 [cs.IT], 24 Nov. 2009.

[12] F. Glover, "Tabu search—part I," *ORSA J. Comput.,* vol. 1, no. 3, Summer 1989, pp. 190-206.

[13] F. Glover, "Tabu search—part II," *ORSA J. Comput.,* vol. 2, no. 1, Winter 1990, pp. 4-32.

[14] P. H. Tan and L. K. Rasmussen, "A reactive tabu search heuristic for multiuser detection in CDMA," in *Proc. IEEE ISIT*, pp. 472, 2002.

[15] P. H. Tan and L. K. Rasmussen, "Multiuser detection in CDMA—a comparison of relaxations, exact, and heuristic search methods," *IEEE Trans. Wireless Commun.*, vol. 3, no. 5, pp. 1802-1809, Sep. 2004.

[16] Y. Wang, Z. Hu, and J. Yu, "A hybrid multiuser detection approach based on tabu search algorithm and multistage detector," in *Proc. IEEE Intl. Conf. Commun., Circuits Syst. West Sino Expositions*, vol. 1, pp. 294-298, June-July 2002.

[17] H. Zhao, H. Long, and W. Wang, "Tabu search detection for MIMO systems," in *Proc. IEEE PIMRC*, pp. 1-5, Sep. 2007.

[18] A. B. Reid, A. J. Grant, and A. P. Kind, "Low complexity list detection for high-rate multiple-antenna channels," in *Proc. IEEE ISIT*, pp. 273, June-July 2003.

[19] N. Srinidhi, T. Datta, A. Chockalingam, and B. Sundar Rajan, "Layered tabu search algorithm for large-MIMO detection and a lower bound on ML performance," in *Proc. IEEE GLOBECOM*, pp. 1-5, Dec. 2010.

[20] G. J. Foschini, "Layered space-time architecture for wireless communication in a fading environment when using multiple antennas," *Bell Lab. Tech. J.*, vol. 1, no. 2, pp. 41-59, Autumn 1996.

[21] A. Duel-Hallen, "Decorrelating decision-feedback multiuser detector for synchronous code-division multiple-access channel," *IEEE Trans. Commun.*, vol. 41, pp. 285-290, Feb. 1993.

[22] L. Wei and C. Schlegel, "Synchronous DS-SSMA with improved decorrelating decision-feedback multiuser detector," *IEEE Trans. Veh. Technol.*, vol. 43, pp. 767-772, Aug. 1994.

[23] D. K. C. So and R. S. Cheng, "Layered maximum likelihood detection for MIMO systems in frequency selective fading channels," *IEEE Trans. Wireless Commun.*, vol. 5, no. 4, pp. 752-762, Apr. 2006.

[24] M. Siti and M. P. Fitz, "On layer ordering techniques for near-optimal MIMO detectors," in *Proc. IEEE WCNC*, pp. 1199-1204, Mar. 2007.

[25] P. H. Tan, L. K. Rasmussen, and T. J. Lim, "Constrained maximum-likelihood detection in CDMA," *IEEE Trans. Commun.*, vol. 49, pp. 142-153, Jan. 2001.

[26] C. Schlegel and L. Wei, "A simple way to compute the minimum distance in multiuser CDMA systems," *IEEE Trans. Commun.*, vol. 45, no. 5, pp. 532-535, May 1997.

[27] X. Zhu and R. D. Murch, "Performance analysis of maximum-likelihood detection in a MIMO antenna system," *IEEE Trans. Commun.,* vol. 50, no. 2, pp. 187-191, Feb. 2002.

[28] Y. Su, X.-D. Zhang, and X. Wang, "A successive interference cancellation algorithm in MIMO systems via breadth-first search," in *Proc. IEEE ICASSP*, pp. 2709-2712, Mar.-Apr. 2008.

[29] Y. Wang and K. Roy, "A new reduced complexity sphere decoder with true lattice boundary awareness for multi-antenna systems," *IEEE ISCAS*, vol. 5, pp. 4963-4966, May 2005.

[30] 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, June 2008.

[31] B. Hassibi and B. M. Hochwald, "How much training is needed in multiple-antenna wireless links?" *IEEE Trans. Inf. Theory*, vol. 49, no. 4, pp. 951-963, Apr. 2003.