A Pipelined Scalable High-throughput Implementation of a Near-ML K-Best Complex Lattice Decoder

Mahdi Shabany, Karen Su, P. Glenn Gulak
The Edward S. Rogers Sr. Department of Electrical & Computer Engineering, University of Toronto

***Abstract*— In this paper, a practical pipelined K-best lattice decoder featuring efficient operation over infinite complex lattices is proposed. This feature is a key element that enables it to operate at a significantly lower complexity than currently reported schemes. The main innovation is a simple means of expanding/visiting the intermediate nodes of the search tree on-demand, rather than exhaustively or approximately, and also directly within the complex-domain framework. In addition, a new distributed sorting scheme is developed to keep track of the best candidates at each search phase; the combined expansion and sorting cores are able to find the K best candidates in just K clock cycles. Its support of unbounded infinite lattice decoding distinguishes our work from previous K-best strategies and also allows its complexity to scale sub-linearly with modulation order. Since the expansion and sorting cores cooperate on a data-driven basis, the architecture is well-suited for a pipelined parallel VLSI implementation of the proposed K-best lattice decoder. Comparative results demonstrating the promising performance, complexity and latency profiles of our proposal are provided in the context of the 4x4 MIMO detection problem.**

***Index Terms*— Breadth-first search, K-Best algorithm, sub-optimal detection, MIMO detection, IEEE802.11m.**

I. Introduction

Multiple-input-multiple-output (MIMO) systems have recently attracted a lot of attention for achieving large spectral efficiency, which makes it the technology of choice in many standards such as IEEE802.11n, IEEE802.16m, and IEEE 802.20. One of the main challenges in exploiting the potential of MIMO systems is to design low-complexity high-throughput detection schemes, which are suitable for efficient VLSI realization, to implement low-power MIMO receivers with near-maximum-likelihood (ML) performance.

However, the complexity of the exhaustive-search optimal ML detection scheme grows exponentially with the number of transmit antennas and the constellation order. Therefore, suboptimal lower-complexity approaches need to be developed in real applications. Depending on how they carry out the non-exhaustive search, detector methods generally fall into two main categories, namely the *depth*-first search, and *breadth*-first search. Sphere decoding (SD) is the most attractive depth-first approach whose performance is ML under the assumption of unlimited execution time, [1]. However, the actual runtime of the algorithm is dependent not only on the channel realization, but also on the operating SNR. Thus leading to a variable throughput rate, which results in an extra overhead in the hardware due to the extra required I/O buffers and lower hardware utilization.

Among the breadth-first search methods, the most well-known approach is the K-Best algorithm (a.k.a M-Algorithm), [2], which is the focus of this paper. The K-Best detector guarantees a SNR-independent fixed-throughput with a performance close to ML. Being fixed-throughput in nature along with the fact that the breadth-first approaches are feed-forward detection schemes with no feedback, makes them especially attractive for VLSI implementation.

II. System Model

Consider a MIMO system with N_T transmit and N_R receive antennas. The equivalent baseband model of the Rayleigh fading channel between the transmitter and the receiver is described by a complex-valued $N_R \times N_T$ channel matrix H. There are two models for such MIMO system namely the *complex* equivalent model and the *real* equivalent model. In this paper, we consider the complex-domain framework. However, the proposed scheme can be easily tailored for the real equivalent model. The complex baseband equivalent model can be expressed as

$$Y = HS + V, \tag{1}$$

where $S = [s_1, s_2, \cdots, s_{N_T}]^T$ is the N_T-dimensional complex transmit signal vector, in which each element is independently drawn from a complex constellation $\mathcal{O}$ orthogonally separable M-QAM schemes with $\log_2 M$ bits per symbol, i.e., $|\mathcal{O}| = M$), $Y = [y_1, y_2, \cdots, y_{N_R}]^T$ is the N_R-dimensional received symbol vector, and $V = [v_1, v_2, \cdots, v_{N_R}]^T$ represents the N_R-dimensional independent identically distributed (i.i.d) circularly symmetric, complex, zero-mean Gaussian noise vector with variance σ^2, i.e., $v_i \sim \mathcal{N}_c(0, \sigma^2)$. The real equivalent model can also be derived using a simple decomposition [3]. The objective of the MIMO detection method is to find the closest lattice point $\hat{S}$ for a given received signal Y, i.e.,

$$\hat{S} = \arg\min_{S \in \mathcal{O}^{N_T}} \| Y - HS \|^2 . \tag{2}$$

The K-Best algorithm is a linear complexity technique that returns near-optimal solutions to the above problem.

A. K-Best Algorithm

Consider the problem in (2), and let's denote the QR-decomposition of the channel matrix as $H = QR$, where Q is a unitary matrix of size $N_R \times N_T$ and R is an upper triangular $N_T \times N_T$ matrix. Applying the nulling operation Q^H results in $Z = Q^H Y = RS + W$, where $W = Q^H V$. Since the nulling matrix is unitary, the noise, W, remains spatially white after nulling. Exploiting the triangular nature of R, (2) can be expanded as follows.

$$\hat{S} = \arg\min_{S \in \mathcal{O}^{N_T}} \sum_{i=1}^{N_T} \left| z_i - \sum_{j=i}^{N_T} R_{ij} s_j \right|^2. \tag{3}$$

The above problem can be thought of as a tree search problem with N_T levels, where starting from the last row, one symbol is detected and based on the detected symbol the next symbol in the upper row is detected and so on. Thus starting from $i = N_T$, (3) can be evaluated in a recursive manner as follows.

$$T_i\big(S^{(i)}\big) = T_{i+1}\big(S^{(i+1)}\big) + \big|e_i\big(S^{(i)}\big)\big|^2 \tag{4}$$

$$e_i\big(S^{(i)}\big) = z_i - \sum_{j=i}^{N_T} R_{ij} s_j = L_i\big(S^{(i)}\big) - R_{ii} s_i, \tag{5}$$

where $S^{(i)} = [s_i \; s_{i+1} \cdots s_{N_T}]^T$, $T_i\big(S^{(i)}\big)$ is the accumulated partial Euclidean distance (PED) where $T_{N_T+1}\big(S^{(N_T+1)}\big) = 0$, $\big|e_i\big(S^{(i)}\big)\big|^2$ denotes the distance increment between two successive nodes/levels in the tree, and $L_i\big(S^{(i)}\big) = z_i - \sum_{j=i+1}^{N_T} R_{ij} s_j$. Note that the same equations can be derived for the real-valued equivalent. Based on the above model, the K-Best algorithm can be described as follows.

1) **Initialization**: Set one path at level $N_T + 1$ with PED = 0.
2) **Expansion**: Expand each of the surviving paths from the previous step to M new possible children in $\mathcal{O}$ and calculate the updated PED for each resulting path.

3) **Sorting**: Sort all the existing paths according to their accumulated PEDs and select the best K paths and discard the others.
4) If not at the last level of the tree, go to step 2) otherwise stop.

The path with the lowest PED at the last level of the tree is the hard decision output of the detector. There are two main computation cores in the above algorithm that play critical roles in the total computational complexity, which are discussed in the following.

1) ***Expansion***: The K-Best algorithm enumerates all the possible children of a parent node, which is M children per parent. Therefore, there are KM total children to be computed and updated in each level. This incurs a large computational complexity overhead to the detectors. The PSK enumeration scheme [4], or its simplified version for M-QAM systems, [5] are proposed to enumerate the children more efficiently. Although PSK is more efficient, it does not scale with the level of the constellation, and has performance loss compared to true K-Best. Although a simplified version of PSK enumeration is proposed in [6], still the number of comparisons increases for higher order constellations.

2) ***Sorting***: In each level of the tree there are KM children. In general, sorting a list of KM numbers has a complexity of $O(K^2M^2)$. Using bubble sorting, which distributes the sorting over multiple cycles, [2], it takes KM cycles to obtain the sorted list, which is still complicated to implement. In [7] a distributed sorting method is proposed based on the Schnorr-Euchner (SE) ordered search technique [1]. However, the proposed approach requires all the children of a parent node to be calculated by the metric computation unit (MCU) and is applicable only for $K \leq \sqrt{M}$. Furthermore, it can not be extended to the complex equivalent model as the distributed sorting scheme uses the natural ordering of the integer numbers on the real axis. Moreover, for higher values of K, the proposed single-cycle merge core becomes increasingly complex resulting in a long critical path. Therefore, the proposed architecture is not suitable for higher order modulations like 64-QAM and 256-QAM as the value of K is large. An improved K-Best approach was also proposed in [8], where the number of children to be sorted is reduced and the sorting is also distributed such that for $K < \sqrt{M}$ the number of children to be sorted is reduced from KM to K^2 (still complex to implement). However, there is no simplification when $K > \sqrt{M}$.

III. Proposed Scheme

A. Novel sorting scheme

Let's consider a $N_R \times N_T$ MIMO system. After the QR-decomposition, the complex equivalent model of this system can be thought of as a detection problem in a tree with N_T levels and M children per parent[1]. The algorithm operates backward from level N_T (corresponding to the last row of R) going up to the first level. Let's consider level l of the tree and assume that the set of K-Best candidates in level $l+1$ (denoted by $\mathcal{K}_{l+1}$) are already determined. Each node in level $l+1$ has M possible children resulting in KM children in level l, from which K best nodes should be selected.

The key idea of our proposed distributed K-Best scheme is to find the first child[2] of each node in $\mathcal{K}_{l+1}$. Among these first children the one with the lowest PED is definitely one of the K-Best candidates in $\mathcal{K}_l$. That child is selected and is replaced by its next best sibling[3]. This process is repeated K times to find the K-Best candidates in level l ($\mathcal{K}_l$). Thus the proposed algorithm can be described as follows:

Distributed K-Best algorithm in K cycles:

1) `Find the K-Best children of level` N_T ($\mathcal{K}_{N_T}$).
2) **`For`** $l = N_T - 1 : -1 : 1$
3) `Find the first child of all candidates in` $\mathcal{K}_{l+1}$. `Call this set` $\mathcal{K}_l$.
4) **`For`** $k = 1 : K$
   a) `Select the child in` $\mathcal{K}_l$ `with lowest PED.`
   b) `Announce this child as the next K-Best candidate in level` l.
   c) `Replace it with its next best sibling.`

   `End`
   `End`

[1]The sorting scheme is described for the complex version. However, the proposed scheme can be easily extended to the real equivalent model. In that case there are $2N_T$ levels in the tree and $\sqrt{M}$ children per parent.

[2]The first child refers to the child with the lowest *local* PED.

[3]The next sibling refers to the child with the next lowest *local* PED.

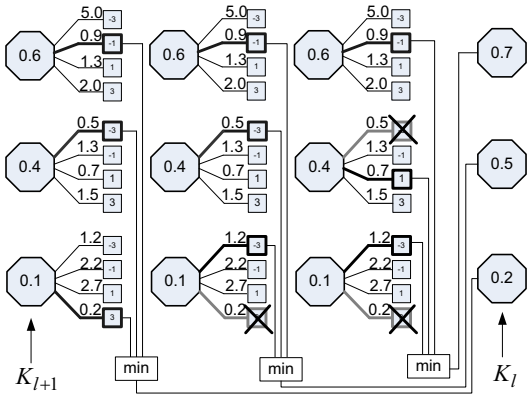


Fig. 1. The proposed scheme for $M = 4$ and $K = 3$.

The proposed scheme is pictorially depicted in Fig. 1 for level l where $M = 4$ and $K = 3$. It basically shows the way that $\mathcal{K}_l$ is derived from $\mathcal{K}_{l+1}$. This scheme takes exactly K clock cycles to find the K best nodes in each level independent of the constellation scheme. Therefore, it is a promising method for the implementation of the K-Best algorithm in high-order constellation schemes such as 64-QAM and 256-QAM. Note that by increasing the order of the constellation, the value of K should be increased. However, in the simulations we show that the value of K scales sub-linearly. Next we propose an efficient technique to find each node's first child (`Step 3`) and the next child (`Step 4.c`) in the complex domain.

B. On-demand expansion scheme

The reason to use the ordered expansion scheme is to find the first child and next child of each parent (i) on-demand and (ii) without visiting all the children. The proposed scheme has two key properties. First, it efficiently expands the tree with the minimum computational complexity and secondly it provides a framework to implement the sorting in a pipelined fashion as was addressed in Section III-A.

Let's consider the level l of the tree. There are K surviving nodes from level $l+1$, (by $\mathcal{K}_{l+1}$). For each node in $\mathcal{K}_{l+1}$, the first child needs to be found (`Step 3`). Since the calculation of the first/next child of all nodes in $\mathcal{K}_{l+1}$ are performed totally independently, hereafter, without loss of generality we focus on one node for the brevity of the the presentation.

1) First Child Selection: Based on the system model in (4), the first child of a node in $\mathcal{K}_{l+1}$ ($s_l^{[1]}$) is the one that minimizes $e_l\big(S^{(l)}\big)$, i.e.[4],

$$
\begin{aligned}
s_l^{[1]} &= \arg\min_{s_l \in \mathcal{O}} \big|e_l\big(S^{(l)}\big)\big|^2 = \arg\min_{s_l \in \mathcal{O}} \big|L_l\big(S^{(l)}\big) - R_{ll}s_l\big|^2 \\
&= \arg\min_{\Re(s_l)\in\Omega} \big| \underbrace{\Re\big[L_l\big(S^{(l)}\big)\big]/R_{ll}}_{u_l^R} - \Re(s_l)\big|^2 \qquad (6) \\
&+ \arg\min_{\Im(s_l)\in\Omega} \big| \underbrace{\Im\big[L_l\big(S^{(l)}\big)\big]/R_{ll}}_{u_l^I} - \Im(s_l)\big|^2, \qquad (7)
\end{aligned}
$$

[4]Because $T_{l+1}\big(S^{(l+1)}\big)$ is in common for all the children of a parent node.

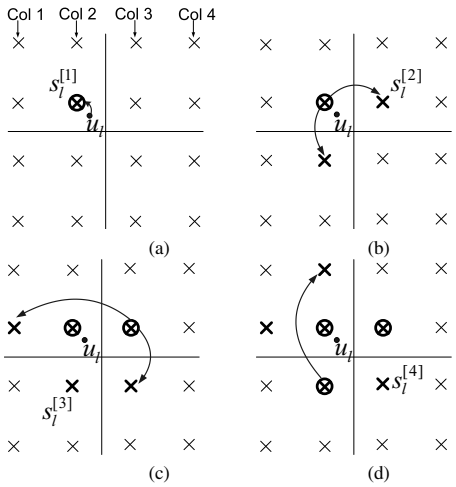


Fig. 2. The first four best children using complex SE enumeration in a 16-QAM Constellation scheme: (a) $\mathcal{L} = \{-1+j\}$, (b) $\mathcal{L} = \{-1-j, +1+j\}$, (c) $\mathcal{L} = \{-1-j, 1-j, -3+j\}$ and (d) $\mathcal{L} = \{-1+3j, 1-j, -3+j\}$.

where $\Omega = \{-\sqrt{M}+1, \cdots, -1, 1, \cdots, +\sqrt{M}-1\}$ represents all the possible values of the real/imaginary part of the constellation points, and (6)-(7) are derived based on the fact that R_{ll} is a real number. Considering the square symmetric M-QAM constellation schemes, there are $|\Omega| = \sqrt{M}$ possible integers on both real and imaginary axis. Thus the optimization in (6)-(7) are computationally cheap to implement as u_l^R, and u_l^I can be easily rounded to the nearest integers in Ω to find the optimized value for $\Re(s_l)$ and $\Im(s_l)$, respectively, as is shown in Fig. 2(a), ($u_l = u_l^R + ju_l^I$). Since this process is implemented for all of the K-Best nodes in level $l+1$, the output of `Step 3` are the K first children of the parent nodes in $\mathcal{K}_{l+1}$.

2) Next Best Child: Upon announcing a child as the next K-Best candidate (`Step 4.b`), its next best sibling needs to be determined (`Step 4.c`). Recall that the first child corresponds to the $s_l \in \mathcal{O}$ that minimizes $e_l(S^{(l)})$. By definition, its next best sibling ($s_l^{[2]}$) is the one that has the next smallest incremental distance $e_l(S^{(l)})$, i.e., it is the (one in) $s_l \in \{\mathcal{O} - s_l^{[1]}\}$ that minimizes $e_l(S^{(l)})$. The selection of the next best sibling can be achieved using our proposed complex SE enumeration scheme, described briefly as follows.

Let $\mathcal{L}$ denote the set of all visited points in the constellation, which have not yet been announced as the next best sibling. As a new point is enumerated, it is added to $\mathcal{L}$ so initially $\mathcal{L} = \{s_l^{[1]}\}$. At each step, the point in $\mathcal{L}$ with the lowest PED is announced to be the next best sibling. That point is removed from $\mathcal{L}$ and replaced by one or two new points that are generated by column or row and column one-dimensional (real) SE enumerations performed on it. As shown in Fig. 2, the row and column enumerations enable coverage of the possible sets of values for $\Re[s_l]$ and $\Im[s_l]$, respectively. These new point(s) are said to be visited and are then added back into $\mathcal{L}$. Note that in Fig. 2, the bold crosses represent the visited points while the bold circled crosses represent the points announced as the next best siblings. Observe also in Fig. 2(c) that when the removed node has the same imaginary component as the first child (in this example, +j), two new row and column nodes are enumerated, whereas in Fig. 2(d), only one new column node is enumerated. Detailed consideration reveals that this simple strategy prevents a node from being added into $\mathcal{L}$ more than once. Repeated application of this procedure ensures on-demand enumeration of the complex constellation points in order of increasing local PED.

The sequence in Fig. 2 shows the process of finding the first 4-best children of a particular parent node using the proposed scheme with the elements of $\mathcal{L}$ listed in each stage in the caption.

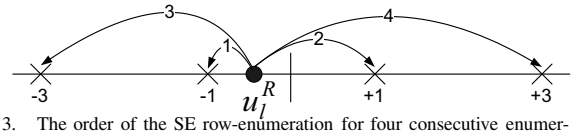


Fig. 3. The order of the SE row-enumeration for four consecutive enumerations.

One key property of the proposed enumeration scheme is that because it is based on the SE enumeration, it does not require that the lattice search space under consideration be bounded. Another feature is that the best children of each parent are generated one-by-one and on-demand. Therefore, the complexity of this approach and the search complexity is independent of the constellation order. This makes our approach a promising one especially for higher order modulations like 64-QAM or 256-QAM.

Note that the above complex version can be easily applied to the real equivalent model. In fact the real domain implementation is a special case of the above scheme where only a one-dimensional row SE enumeration on the real numbers is implemented. (Therefore, there is no need to develop a new approach for the real model.) Note also that due to the pipelined nature of the proposed scheme, the minimization in `Step 4.a` to find the next K-Best candidate is implemented in one clock cycle as the first children in `Step 3` and the next siblings in `Step 4.c` are sorted as being generated in a pipelined fashion. The fine-grained VLSI implementation of the above architecture is an ongoing research project.

IV. Complexity Analysis

In this section we study the complexity of the proposed approach and compare it with the other K-Best schemes in the literature. The focus would be on the detection part excluding the QR-decomposition and channel preprocessing as they are the common blocks between all the schemes. Note that some of the K-Best schemes in the literature consider radius control, as is done in SD, along with the K-Best implementation, which might result in some complexity saving for high SNR regimes. Since this technique can apply to any implementation of K-Best, we do not consider radius control in this paper and focus purely on the K-Best algorithm itself.

Table I shows the complexity comparison between different schemes. For comparison metrics, the number of visited children (called *expand* in Table I), the required number of clock cycles to do the sorting (called *sort*), and the total latency are considered. The latency shows the number of clock cycles it takes to detect a symbol vector, S. The sorting for [3], and [7] are calculated with the assumption that it can be implemented with a linear complexity using bubble sort, [2]. The value listed in the expand row refers to the number of visited children or equivalently the number of Euclidean distances required to be calculated. This directly translates to more area and power when it comes to the silicon implementation. However, if all the distance calculations are done in parallel there is no significant throughput loss. The bit-error-rate (BER) performance results of our proposed approach is the same as that of in [3], [7], and [8] as they all implement the true K-Best method. Thus the major difference between all these schemes including this work is the way the K-Best method is implemented, which translates to different throughput and/or hardware complexity. As can be seen, using our scheme both in real and complex modes, significant complexity savings are achieved. More specifically, the complex mode is clearly beneficial as none of the proposed approaches support the complex mode. The table shows the total number of distance calculations, sorting period as well as the total latency of the search throughout the tree in a 4×4 MIMO configuration for different constellation schemes. The numbers in parentheses show the value of K for real and complex modes in each case, respectively. Note that both our real and complex mode versions outperform [7] in terms of the latency, total distance calculation as well as the sorting time. Moreover, our

scheme is more efficient in the sense that only $2K-1$ children need to be calculated whereas in [7] the PED of all the children of all parent nodes should be calculated by MCU. Thus our complex scheme results in a faster architecture with almost half silicon area due to the lower depth of the tree and the number of visited children.

The interesting point is that by deploying the complex mode, the computational complexity is even lower than the real case. This is because of the fact that the depth of the tree is halved, however, the value of K resulting in the same performance is not doubled. For instance, in the case of 4×4 16-QAM, as we will see in the simulation results, $K=6$ in the complex mode gives the same performance result as $K=5$ in the real mode (saving of 30% in total area and latency). The saving in the sorting is also considerable in 64 and 256-QAM cases (35%, and 30%, respectively). To make this happen, each parent node can calculate its next best child while the distributed sorting is performed. This means each parent always knows its current and the next best child so they are ready to be used by the sorting core.

Another promising aspect of our approach is that the number of visited nodes is NOT a function of the constellation order. The fact that makes this feature possible is the ordered expansion (with on-demand basis) along with the pipelined sorting scheme proposed in this paper. Therefore, the complex version is the preferred scheme to apply although in order to obtain the same performance, the value of K should increase. It is worth noting that the nice features of the complex version requires an extra circuitry to implement the above proposed expansion scheme in the complex domain. However, since it is implemented in an on-demand basis, and the fact that $\mathcal{L}$ does not populate proportionally, this extra circuitry is negligible. This is because based on the numerical results, $|\mathcal{L}|\leq 7$ independent of the constellation scheme.

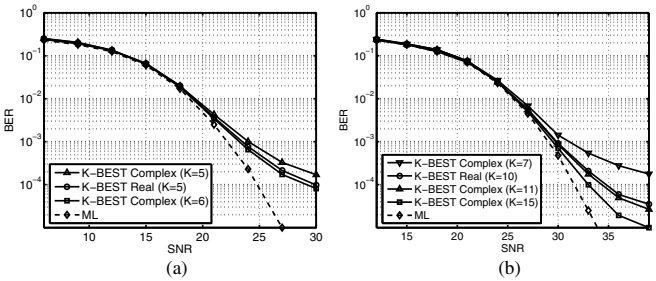
V. Simulations

In theory, the K-Best algorithm might miss the hard-ML point and might have performance loss as a result. However, for a proper choice of K, the BER performance of the K-Best method approaches the optimal case for a reasonable range of signal-to-noise-ratio (SNR) values. In the following, the simulation results for a single-carrier 4×4 MIMO system is presented for both 16-QAM and 64-QAM schemes. Fig. 4(a) shows the BER performance of a 16-QAM scheme for different values of K for real and complex equivalent models as well as the optimal ML result. The SNR value is defined as the signal-to-noise ratio per transmitted symbol. As shown, for the normal range of interest (0-23 dB) the performance loss is small. Based on the figure, the real model implementation outperforms the complex model for the same value of K (K=5 in this Fig. 4(a)). However, by increasing the value of K in the complex mode to 6, the performance result of the complex and real models are almost the same. Increasing the K from 5 to 6 increases complexity by 20% for the calculation of the extra children. However, since the number of levels in the complex mode is 4 vs 8 in the real case, the total complexity saving is 40% in this case.

Fig. 4(b) shows the BER performance for 64-QAM scheme. Again for the normal range of interest (0-30 dB) the performance loss is small. Based on the figure, the performance of the real model with $K=10$ is the same as that of the complex model for $K=11$. This would result in a saving of 45% in the sorting core compared to the best real model implementation in the literature (see Table. I). The performance curve for $K=15$, and $K=7$ are also shown. The loss associated to $K=7$ is considerable whereas in the case of $K=15$, there is 0.4 dB improvement at 30 dB.

VI. Conclusions

An efficient K-best lattice decoder featuring efficient operation over infinite complex lattices was proposed. The key idea was a simple means of expanding/visiting the intermediate nodes of the search tree on-demand, rather than exhaustively, and also directly within the complex-domain framework. In addition, a new distributed sorting scheme was developed, which finds the K best candidates in just K clock cycles. As opposed to the other schemes in the literature, the proposed scheme can be applied to an infinite lattice, finds the children on-demand, is scalable both in terms of the number of transmit antenna and the constellation level, and can be easily applied to both real and complex mode. The architecture is well-suited for a pipelined parallel VLSI implementation of a K-best lattice decoder.



(a) (b)

Fig. 4. K-Best vs ML result for 4×4 with different values of K in both real and complex mode: (a) 16-QAM (b) 64-QAM.

TABLE I
Comparison between different K-Best implementations.

		[3] (real)	[7] (real)	[8] (real)	This work (real)	This work (complex)
4×4 16QAM (5,6)	expand	160	160	160	72	44
	sort	160	40	280	37	24
	latency	240	49	440	40	35
4×4 64QAM (10,11)	expand	640	640	640	152	84
	sort	640	80	900	74	44
	latency	720	99	1540	80	65
4×4 256QAM (15,17)	expand	1920	1920	1920	232	140
	sort	1920	120	2700	113	72
	latency	2000	149	4620	120	107

References

[1] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closest point search in lattices," *IEEE Trans. on Inf. Theory*, vol. 48, no. 8, pp. 2201–2214, Aug. 2002.

[2] K. W. Wong, C. Y. Tsui, R. S. K. Cheng, and W. H. Mow, "A VLSI architecture of a K-best lattice decoding algorithm for MIMO channels," *IEEE Int. Symp. Circuits Syst.*, vol. 3, pp. 273–276, May 2002.

[3] Z. Guo and P. Nilsson, "Algorithm and implementation of the K-Best sphere decoding for MIMO detection," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 3, pp. 491–503, March 2006.

[4] B. M. Hochwald and S. ten Brink, "Achieving near-capacity on a multiple-antenna channel," *IEEE Trans. Commun.*, vol. 51, pp. 389–399, March 2003.

[5] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bolcskei, "VLSI implementation of MIMO detection using the sphere decoding algorithm," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 7, pp. 1566–1577, July 2005.

[6] S. Chen, T. Zhang, and Y. Xin, "Relaxed K-best MIMO signal detector design and VLSI implementation," *IEEE Transactions on very large scale integration VLSI systems*, vol. 15, no. 3, pp. 328–337, 2007.

[7] M. Wenk, M. Zellweger, A. Burg, N. Felber, and W. Fichtner, "K-Best MIMO detection VLSI architectures achieving up to 424 Mbps," *In proceedings of IEEE International Symposium on Circuits and Systems (ISCAS'06)*, pp. 1151–1154, 2006.

[8] Q. Li and Z. Wang, "An improved K-Best sphere decoding architecture for MIMO systems," *Fortieth Asilomar Conference on Signals, Systems and Computers ACSSC'06*, pp. 2190–2194, 2006.