Received XX Month, XXXX; revised XX Month, XXXX; accepted XX Month, XXXX; Date of publication XX Month, XXXX; date of current version XX Month, XXXX.

Digital Object Identifier 10.1109/XXXX.2022.1234567

An Efficient K-best MIMO Detector for Large Modulation Constellations

Yu-Xin Liu[1], Shih-Jie Jihang[2] , and Yeong-Luh Ueng[3]

[1] Institute of Communications Engineering, National Tsing Hua University, Hsinchu 300, Taiwan
[2]Department of Electrical Engineering, National Tsing Hua University, Hsinchu 300, Taiwan
[3]Department of Electrical Engineering and the Institute of Communications Engineering, National Tsing Hua University, Hsinchu 300, Taiwan

Corresponding author: Yeong-Luh Ueng (email: ylueng@ee.nthu.edu.tw).

ABSTRACT For K-best multiple-input multiple-output (MIMO) detection using real-valued decomposition (RVD), we need to obtain the K surviving candidates from $K\sqrt{M}$ candidates, where M is the modulation order. This paper presents a sorter-free detection algorithm, where the K surviving nodes can be obtained in $\log_2 K$ iterations, which is independent of modulation size. The $K\sqrt{M}$ candidates are arranged into a multiple-layer table using the proposed path metric discretization. A bisection-based search algorithm is used to obtain the locations of the K surviving candidates. A low-complexity fully-pipelined architecture is devised in order to implement the proposed MIMO detection without the need to use any dividers. In addition, an efficient method for storing information from child nodes is proposed, which requires significantly less storage space compared to the conventional Schnorr Euchner (SE) enumeration approach. Implementation results show that the proposed K-best MIMO detector supports a 6.4Gb/s throughput that has a 0.32 μs latency in a 90 nm process for a 256-quadrature amplitude modulation (QAM) 4×4 MIMO system. In addition, compared to the sorter-based baseline detector, the proposed detector improves the hardware efficiency by 77%.

INDEX TERMS K-best detection, multiple-input multiple-output (MIMO), very large-scale integration

I. INTRODUCTION

THE use of multiple-input multiple-output (MIMO) systems is common in both wireless local area networks and mobile communication systems. In order to support the requirements of dramatically-increased throughput, the scale of the antennas and the modulation order should be increased. Although massive MIMO [1] has been considered to be one of the key techniques for future wireless communications, small-scale MIMO based on a large modulation size has been specified in the current standards. In the WiFi 6 (IEEE 802.11ax) standard [2], small-scale MIMO employing 1024 quadrature amplitude modulation (QAM) is specified. In the 5G standard [3], small-scale MIMO utilizing 256 QAM is specified. For such high-order modulations, an efficient hardware design for a high-throughput MIMO detector is challenging, even for a small-scale MIMO case. MIMO detection using the maximum likelihood (ML) algorithm provides optimal performance. However, the high complexity, which grows exponentially with the number of antennas and the modulation size, leads to difficulties in hardware implementation, especially when considering the 1024-QAM modulation used in the IEEE 802.11ax standard. Linear detection methods, such as zero-forcing detection and minimum mean-square error detection, provide a low-complexity method [4]. However, when considering the high-order modulation applied in the latest MIMO systems, the performance of linear detection methods is unsatisfactory [5]–[7].

In contrast, nonlinear detection methods, such as sphere detection (SD) [8]–[10] and K-best detection [11]–[13] have been widely considered. The SD algorithm is able to achieve a near-ML performance, where its complexity grows in conjunction with the number of transmit antennas and the modulation orders based on lower-bound analysis [14]. K-best detection, which is equivalent to the sequential M-algorithm [15], provides a flexible solution between performance and complexity, by selecting a suitable number of surviving nodes through the tree-structure detection process. To either improve the performance or reduce the complexity, several K-best MIMO detection techniques have been proposed in recent years. For the pre-processing component of K-best detection, which refers to the QR decomposition of the channel matrix, sorted-QR decomposition (SQRD) [16]–[18] and lattice reduction (LR) [19]–[24] are the two main

techniques that achieve a better conditioned R matrix and improve the error-rate performance.

Conversely, for the tree-structure detection elements, reducing the number of candidates is an effective way of reducing the complexity. In [25], cluster-based detection is proposed, where the constellation points are divided into several clusters, and only the candidates in the selected clusters will then be considered. In addition, a fixed sphere detector (FSD) [26]–[28], which also reduces the number of sorting candidates by limiting the number of expanded child nodes from each parent node, is also introduced. Furthermore, if real-valued decomposition (RVD) is applied to an M-QAM MIMO system, the number of candidates for a single detection level can be reduced from KM to $K\sqrt{M}$.

Besides reducing the number of candidates, several algorithms and architectures have been proposed in order to obtain the K surviving nodes in an efficient manner. The Schnorr Euchner (SE) enumeration [29] approach can be applied to MIMO systems using RVD, where the child nodes from each parent node can be arranged in order without sorting. In this case, merge sorters, which can be realized using an odd-even sorter and a bitonic sorter [30], are widely utilized to efficiently sort these locally ordered candidates from K parent nodes. Furthermore, by combining the odd-even sorter together with the bitonic sorter, hybrid sorters have been devised to further reduce the hardware complexity [31]–[33]. In contrast, the winner path expand (WPE) algorithm [34]–[36] replaces the sorting process by utilizing a sequential minimum search procedure to obtain the K surviving nodes. In this case, the hardware complexity for the overall MIMO detector is greatly reduced, since the minimum search approach provides a much lower hardware complexity than the sorter.

However, techniques that reduce the number of candidates, such as FSD, suffer a performance penalty when high-order modulations are considered. In addition, the WPE algorithm, which obtains the K surviving nodes through a sequential process suffers from a long latency when K is relatively large. For the improved sorters, although the number of comparison units in the sorting network is reduced in the hybrid version, it is still large in MIMO systems using large modulation constellations. For instance, the number of comparison units is increased by 70% in [33] when the modulation changes from 64-QAM to 256-QAM where $K = 32$. To address these problems, this investigation proposes a sorter-free K-best MIMO detection scheme, in which K surviving nodes can be obtained based on a short latency while retaining a comparable performance. The contributions and results of this study are summarized as follows.

1) A path metric discretization (PMD) approach is proposed that arranges the candidates into a multi-layer table. Therefore, we only need to determine the locations of K surviving nodes in the table, rather than sorting all candidates. The bisection-based search algorithm is then applied to obtain K surviving nodes using $\log_2 K$ iterations, which is independent of the modulation size. In contrast, the WPE needs K iterations in order to obtain the K-best surviving nodes. Compared to the WPE, a less number of iterations and comparison operations are used to locate the K surviving nodes.
2) Directly implementing the PMD by using look-up tables (LUT) and row-based enumeration (RBE) is able to improve the hardware efficiency by 27% compared to the sorter-based baseline detectors, which benefits from a sorter-free search process.
3) To further improve the hardware efficiency, we replace the LUT with the proposed increment calculation unit (ICU) for which we only need to determine the location of the first candidate. The locations of the remaining candidates can be determined automatically. The results show that the method improves the hardware efficiency by 39% compared to the baseline detector.
4) Finally, RBE, having a long execution cycle is replaced by column-based enumeration (CBE) in the layer-search module in order to reduce the hardware latency. Compared to the sorter-based baseline detector, the hardware efficiency is increased by 77%. For a 4×4 256-QAM MIMO detector where $K = 16$, a 6.4Gb/s data throughput and a 0.32-μs latency can be achieved in a 90 nm process.

The remainder of this paper is organized as follows. Section II presents the system model and the conventional K-best detection algorithm. The proposed sorter-free detection method is presented in Section III. In Section IV, the fully-pipelined memory-efficient VLSI architecture is illustrated. In addition, a complexity and performance evaluation compared with the state-of-the-art works are demonstrated in Section V. Finally, a conclusion is presented in Section VI.

II. Backgrounds

A. System Model

In an $N_t \times N_r$ MIMO system, where N_t is the number of transmitting antennas and N_r is the number of receiving antennas, the system model [31]- [36] can be illustrated as

$$\tilde{\mathbf{y}}_\mathbf{c} = \mathbf{H}_\mathbf{c}\mathbf{x}_\mathbf{c} + \tilde{\mathbf{n}}_\mathbf{c}, \tag{1}$$

where $\mathbf{H}_\mathbf{c}$ is the $N_r \times N_t$ Rayleigh fading channel matrix, $\mathbf{x}_\mathbf{c}$ is the $N_t \times 1$ complex transmitted vector and $\tilde{\mathbf{y}}_\mathbf{c}$ denotes the $N_r \times 1$ complex received vector. In addition, $\tilde{\mathbf{n}}_\mathbf{c}$ is an $N_r \times 1$ complex noise vector that has a complex Gaussian distribution with zero-mean and variance σ^2. After the RVD, (1) becomes

$$\begin{aligned}\tilde{\mathbf{y}} &\equiv \begin{bmatrix}\Re(\tilde{\mathbf{y}}_\mathbf{c})\\ \Im(\tilde{\mathbf{y}}_\mathbf{c})\end{bmatrix}\\ &= \begin{bmatrix}\Re(\mathbf{H}_\mathbf{c}) & -\Im(\mathbf{H}_\mathbf{c})\\ \Im(\mathbf{H}_\mathbf{c}) & \Re(\mathbf{H}_\mathbf{c})\end{bmatrix}\begin{bmatrix}\Re(\mathbf{x}_\mathbf{c})\\ \Im(\mathbf{x}_\mathbf{c})\end{bmatrix} + \begin{bmatrix}\Re(\tilde{\mathbf{n}}_\mathbf{c})\\ \Im(\tilde{\mathbf{n}}_\mathbf{c})\end{bmatrix}\\ &= \mathbf{H}\mathbf{x} + \tilde{\mathbf{n}},\end{aligned} \tag{2}$$

where $\Re(.)$ and $\Im(.)$ denote the real and the imaginary components, respectively. In addition, $\tilde{\mathbf{y}}$ is the $2N_r \times 1$ real-valued received vector, $\mathbf{H}$ is the $2N_r \times 2N_t$ channel matrix, $\mathbf{x}$ is the $2N_t \times 1$ real-valued transmitted vector and $\tilde{\mathbf{n}}$ is the $2N_r \times 1$ real-valued noise vector.

In the real-valued system, the real and the imaginary component of the QAM points are considered as two separate pulse-amplitude modulation (PAM) points. For the M-QAM considered in this work, let $\mathbf{\Omega} = \{-\sqrt{M} + 1 + 2i \mid 0 \leq i \leq \sqrt{M} - 1\}$ denotes the set of all $\sqrt{M}$ possible PAM points. For instance, $\mathbf{\Omega}$ for the 64-QAM system is $\{-7, -5, -3, -1, 1, 3, 5, 7\}$. After the RVD, the channel matrix passes through QR decomposition, denoted by $\mathbf{H} = \mathbf{QR}$, where $\mathbf{Q}$ is a unitary $2N_r \times 2N_t$ matrix and $\mathbf{R}$ is a $2N_t \times 2N_t$ real-valued upper-triangular matrix. Subsequently, the system model can be described as $\mathbf{y} = \mathbf{Rx} + \mathbf{n}$, where $\mathbf{y} \equiv \mathbf{Q}^H \tilde{\mathbf{y}}$ and $\mathbf{n} \equiv \mathbf{Q}^H \tilde{\mathbf{n}}$ are $2N_t \times 1$ vectors.

B. K-best Detection Algorithms

The tree-search based K-best detection approach comprises $2N_t$ levels corresponding to each element of $\mathbf{x}$. In each level, each parent node expands its $\sqrt{M}$ child nodes, with all child nodes from different parent nodes as candidates. Subsequently, the candidates with the K smallest Partial Euclidean Distances (PED) are regarded as the surviving nodes of the current level and also the K parent nodes for the next level. The PED in the i-th level denoted as Γ_i can be calculated according to

$$\begin{aligned} \Gamma_i &= \Gamma_{i+1} + \left\| y_i - \sum_{j=i}^{2N_t} r_{ij} x_j \right\|^2 \\ &= \Gamma_{i+1} + \gamma_i \quad i = 2N_t, 2N_t - 1, \cdots, 1, \end{aligned} \tag{3}$$

where y_i is the i-th element of $\mathbf{y}$, x_j is the j-th element of $\mathbf{x}$, γ_i is the increment in the i-th level, r_{ij} is the element in the i-th row and the j-th column of $\mathbf{R}$, and $\Gamma_{2N_t+1} = 0$. As a result, the K surviving nodes are determined by the K smallest PEDs from the $K\sqrt{M}$ candidates. Note that since the $\mathbf{R}$ matrix is an upper triangular matrix, the detection begins from detecting the $2N_t$-th element in the $2N_t$-th level and ends after detecting the first element in the first level.

If we first obtain y_i' according to

$$y_i' = y_i - \sum_{j=i+1}^{2N_t} r_{ij} x_j, \tag{4}$$

the increment term, i.e., γ_i, in (3) can be calculated from

$$\gamma_i = \left\| y_i - \sum_{j=i}^{2N_t} r_{ij} x_j \right\|^2 = \left\| y_i' - r_{ii} x_i \right\|^2. \tag{5}$$

Based on the form presented in (5), the SE enumeration [29] arranges the child nodes from a single parent node in order. By dividing y_i' by r_{ii}, we can label the closest PAM point as the first child node. Next, the SE enumeration process traverses the PAM points in a zig-zag pattern to obtain the remaining child nodes in order. In this case, the SE enumeration process arranges the child nodes in order without sorting.

C. Challenges for K-best Detection for Large Constellations

Using the SE enumeration strategy, the child nodes from each parent node can be arranged in order. All the $K\sqrt{M}$ child nodes from K different parent nodes can be considered as locally ordered. The sorting complexity can be greatly reduced by dealing with the locally-ordered candidates rather than fully-random candidates. In the conventional K-best detection approach, the merge sorter, such as the odd-even sorter together with the bitonic sorter, are ideal for obtaining the K-best surviving nodes from these locally ordered child nodes. Besides the conventional merge sorters, the hybrid sorter [31] combines the two previously mentioned merge sorters, reducing the number of comparison units and, hence, the hardware complexity, as it only preserves the comparison units for sorting the K surviving nodes rather than sorting all the candidates. Since the ordering information is not necessary for the K surviving nodes, the hybrid sorters presented in [32] and [33] remove the comparison units in order to obtain the ordering information for the K surviving nodes to further reduce the hardware complexity.

In contrast, the WPE algorithm presented in [34]–[36] replaces the sorting process by implementing the minimum search for K iterations. For the first iteration, the best child node from the SE enumeration list for each parent node is selected as a candidate. Using the minimum search, the child node that has the smallest PED can be obtained from these K candidates and is regarded as the first surviving node. In addition, the parent node that generates the first surviving node is the winner, so it is allowed to generate its second child node according to the SE enumeration list. The new child node from the winner together with the $K-1$ non-surviving child nodes in the first iteration, are fed into the minimum search again so as to determine the second surviving node and the winner in the second iteration. The process is repeated for K iterations in order to obtain the K-best surviving nodes. As a result, the huge number of comparison units required for sorting are avoided, which can help to greatly reduce the hardware complexity.

However, when large constellations are considered, the hybrid-sorter approaches and the WPE algorithm still encounter difficulties in either hardware complexity or latency since a relatively large value for K should be applied to maintain a satisfactory error-rate performance. Considering the hybrid-sorter approaches discussed in [32], the number of comparison units increases by 70% when the modulation changes from 64-QAM to 1024-QAM where $K = 32$. In the WPE algorithm, the sequential K-iteration minimum search leads to a longer latency. To overcome the existing difficulties, we propose a sorter-free detection algorithm and an architecture that can obtain the K surviving nodes using $\log_2 K$ iterations.

III. Proposed Sorter-free K-best Detection

The proposed detection scheme is divided into two stages, the table-construction stage and the candidate-update stage. In the table-construction stage, child nodes from a single parent node are allocated in order into two columns using the proposed path metric discretization technique. Therefore, there are a total of $2K$ columns in the multi-layer table. Based on this type of multi-layer candidate table, we can obtain the K surviving candidates using a search approach rather than a sorting approach. During the candidate-update stage, a bisection-based search process is applied to determine where the K surviving nodes are located in the table. Since the detection process is described generally and no specific detection level is assumed, the notations y'_i and r_{ii} in (5) are simplified to y' and r, respectively, in this section.

A. Table Construction Using Path Metric Discretization

In contrast to the zig-zag SE enumeration strategy, the proposed method divides the child nodes from a single parent node into two ordered sets, where the ordering is based on the distance between the child node and y'. The first set is formed with the child nodes on the right-hand side of y', and the second is formed with the child nodes on the left-hand side of y'. To further classify these two sets, the set containing the first closest child node to y' is denoted as $\mathbf{F}$, where $\mathbf{F} = \{v_1, v_2, ..., v_{|\mathbf{F}|}\}$, and v_1 to $v_{|\mathbf{F}|}$ are the nodes arranged in an ascending order according to its distance to y'. In contrast, the set containing the second closest node to y' is denoted as $\mathbf{S}$, where $\mathbf{S} = \{u_1, u_2, ..., u_{|\mathbf{S}|}\}$, and u_1 to $u_{|\mathbf{S}|}$ are the nodes arranged in an ascending order according to its distance to y'. Fig. 1 shows an example, where v_1 and u_1 are solid nodes. In addition, v_2 and u_2 are represented using dotted nodes since they are not actually recorded, which will be evident from the discussion presented in Section IV.

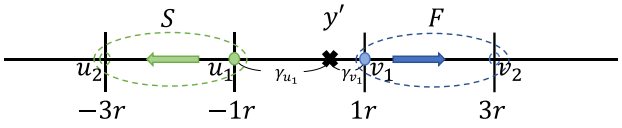


FIGURE 1. An example for sets $\mathbf{F}$ and $\mathbf{S}$, where $\mathbf{F}$ ($\mathbf{S}$) contains the first (second) closest child node to y'.

In the following, we will consider the j-th parent node and, hence, a superscript j is added to its child nodes v_k and u_k and its $\mathbf{F}$ and $\mathbf{S}$ sets. If the PED for the j-th parent node is denoted as Γ_{p_j} and the PEDs for the offspring child nodes v_k^j and u_k^j are denoted as $\Gamma_{v_k}^j$ and $\Gamma_{u_k}^j$, respectively, which can be calculated according to

$$\begin{aligned} \Gamma_{v_k}^j &= \gamma_{v_1}^j + \Gamma_{p_j} + 2|r| \times (k-1), \quad 1 \le k \le |\mathbf{F}^j|, \\ \Gamma_{u_k}^j &= \gamma_{u_1}^j + \Gamma_{p_j} + 2|r| \times (k-1), \quad 1 \le k \le |\mathbf{S}^j|, \end{aligned} \tag{6}$$

where $\gamma_{v_1}^j = \left\| y' - r v_1^j \right\|^2$ and $\gamma_{u_1}^j = \left\| y' - r u_1^j \right\|^2$. The derivation of (6) is based on (3) and (5), where r_{ii} in (5) is simplified as r in (6). It can be observed from (6) that two adjacent nodes in each set have a fixed difference in distance, which is $2|r|$.

Writing $\Gamma_{\min} \equiv \min\{\Gamma_{v_1}^0, \Gamma_{v_1}^1, ..., \Gamma_{v_1}^{K-1}, \Gamma_{u_1}^0, \Gamma_{u_1}^1, ..., \Gamma_{u_1}^{K-1}\}$, which can be realized using the minimum finder rather the sorter. The PMD can be executed according to

$$\begin{aligned} D_{v_k}^j &= \left\lfloor \frac{\Gamma_{v_k}^j - \Gamma_{\min}}{|r|} \right\rfloor, \ 1 \le k \le |\mathbf{F}^j|, \quad 0 \le j \le K-1, \\ D_{u_k}^j &= \left\lfloor \frac{\Gamma_{u_k}^j - \Gamma_{\min}}{|r|} \right\rfloor, \ 1 \le k \le |\mathbf{S}^j|, \quad 0 \le j \le K-1, \end{aligned} \tag{7}$$

where $D_{v_k}^j$ and $D_{u_k}^j$ are respectively the discretization results for $\Gamma_{v_k}^j$ and $\Gamma_{u_k}^j$. Since both $D_{v_k}^j$ and $D_{u_k}^j$ are natural numbers, we can allocate v_k^j and u_k^j into layer $D_{v_k}^j$, ($2j$-1)-th column, and layer $D_{u_k}^j$, $2j$-th columns in the candidate table, respectively.

An example of the table construction is illustrated in Fig. 2, where a 16-QAM 2x2 MIMO system and $K = 3$ are considered, and the corresponding three parent nodes are respectively denoted as p_0, p_1 and p_2. Using (6) and (7), we have $(D_{v_1}^0, D_{v_2}^0, D_{v_3}^0) = (0, 2, 4)$, $D_{u_1}^0 = 1$, $(D_{v_1}^1, D_{v_2}^1) = (1, 3)$, $(D_{u_1}^1, D_{u_2}^1) = (2, 4)$ and $(D_{v_1}^2, D_{v_2}^2, D_{v_3}^2, D_{v_4}^2) = (1, 3, 5, 7)$, as shown in Fig. 2(a). It is worth noting that for p_2, all child nodes are on the same side of y', thus classifying these nodes into the $\mathbf{F}$ set, and there is no node in set $\mathbf{S}$. In Fig. 2(b), these nodes are allocated to the corresponding columns, where the layer located is determined according to their discretized results.

B. Candidate Update Using Bisection Layer Search

After the PMD, the PEDs for the candidates located in layer $\ell + 1$ are not less than those candidates located in layer ℓ. This means that the PEDs for the candidates in the candidate table are ordered based on its layer index. As a result, it is possible to determine the K surviving candidates by sequentially accumulating the candidates from layer 0 to the following layers until the K best candidates are obtained. Mathematically, n_ℓ denotes the number of candidates in layer ℓ and $f(W)$ denotes the total number of candidates from layer 0 to layer W, i.e., $f(W) \equiv \sum_{\ell=0}^{W} n_\ell$. It is obvious that $f(W)$ is a non-decreasing function. When $f(L_m) \ge K$ and $f(L_m - 1) < K$, which means that the best K candidates will definitely be included from layer 0 to layer $L_m - 1$, and some of them will still be in layer L_m. The sequential accumulating process can be used to determine the value of L_m using $L_m + 1$ iterations. However, this approach will result in a non-deterministic throughput since L_m is a random variable.

To address this problem, we first determine a value of $L_m = 2^Q - 1$, where $Q \in \mathbb{N}$, such that $P_r(f(L_m) \ge K)$ is large enough. This means that the best K candidates will be included from layer 0 to layer L_m with a high probability. We then apply a bisection-based search algorithm on this 2^Q-by-$2K$ table to obtain the value of L_m using Q iterations. Let

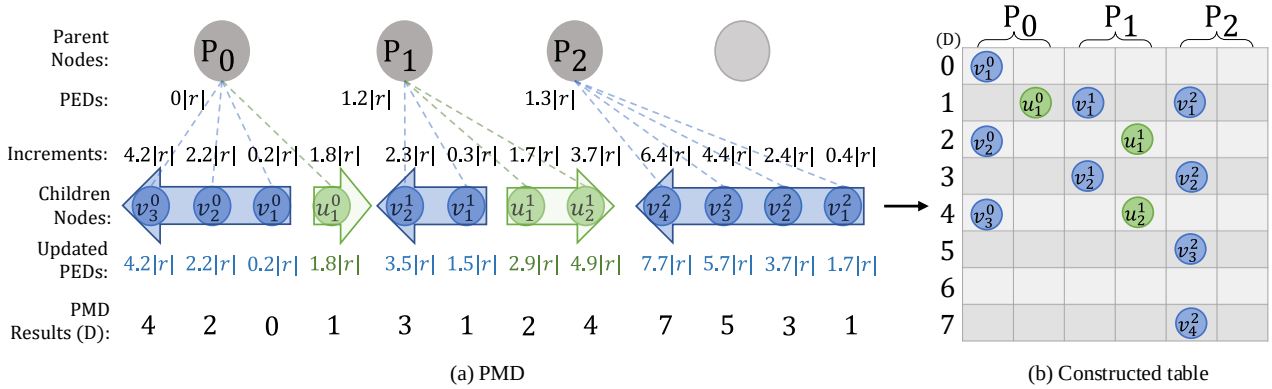


FIGURE 2. **Table construction based on the discretization results from all the child nodes where $K = 3$, where (a) is the path metric discretization (PMD) results for the candidates, and (b) is the constructed table based on the PMD results for the candidates.**

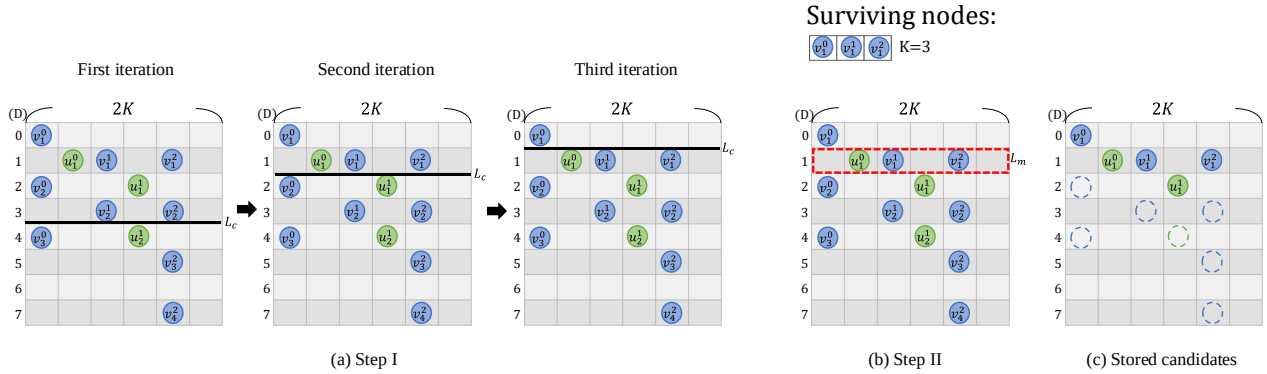


FIGURE 3. **Proposed K-best detection, where (a) is the bisection-based layer search for L_m, (b) is the candidate selection for the surviving nodes, and (c) shows the stored candidates used in the hardware implementation.**

$L_l(k)$ and $L_u(k)$ denote the indices for both the lower layer and the upper layer considered at iteration k, respectively. In addition, define another layer index $L_c(k)$ according to

$$L_c(k) = \lfloor \frac{L_l(k) + L_u(k)}{2} \rfloor. \tag{8}$$

Initially, $L_l(0) = 0$, $L_u(0) = L_m$. Write $f(L_c(k-1))$ as β. At iteration k, where $1 \le k \le Q$, the values of $L_l(k)$ and $L_u(k)$ are recursively updated according to

$$\begin{cases} L_l(k) = L_c(k-1) \quad L_u(k) = L_u(k-1), & \beta < K \\ L_l(k) = L_l(k-1) \quad L_u(k) = L_c(k-1), & \beta > K \\ L_m = L_c(k-1) \quad Terminate, & \beta = K. \end{cases} \tag{9}$$

Finally, L_m can be obtained from

$$\begin{cases} L_m = L_c(Q-1)\ , & f(L_c(Q-1)) \ge K \\ L_m = L_c(Q-1) + 1\ , & o.w.. \end{cases} \tag{10}$$

Once we have obtained L_m, we can select the best candidates from layer 0 to layer $L_m - 1$ since there are definitely going to be $f(L_m - 1)$ surviving candidates. The remaining $K - f(L_m - 1)$ surviving candidates are selected from all the candidates in layer L_m. Since the candidates in layer L_m have the largest PEDs among the K-best candidates and do not affect the detected paths most of the time, we apply a sorter-free approach, which first selects the candidates in set $\mathbf{F}$, and then set $\mathbf{S}$ to satisfy the vacancies in $K - f(L_m - 1)$ undetermined surviving candidates. It can be seen from the simulation results not shown here that the proposed sorter-free approach has almost no degradation in error-rate performance compared to the conventional K-best detection approach.

An example of the proposed searching process is illustrated in Fig. 3, where a 16-QAM 2x2 MIMO system and $Q = 3$ are considered. This means that eight layers and three iterations are needed in order to obtain L_m. For the first iteration, $L_l(0) = 0$ and $L_u(0) = 7$. Hence, $L_c(0) = 3$ and $f(L_c(0))$ is 8, as shown in Fig. 3(a). After three iterations, we can determine $L_m = 1$. Therefore, the candidates from layer 0 to layer $L_m - 1$ are directly identified as surviving candidates. As shown in Fig. 3(b),

$f(L_m - 1) = 1$ and the first surviving candidate is v_1^0. Additionally, $K - f(L_m - 1) = 2$ undetermined surviving candidates are selected using the proposed method. As a result, the node u_1^0 is discarded, while v_1^1 and v_1^2 are selected accordingly, to fill any vacancies in the surviving candidates.

C. Summary and Parameter Selection

The proposed sorter-free K-best detection method for $N_t \times N_r$ MIMO systems is summarized in Algorithm 1. Since a total of N_t transmit antennas and the RVD are used, we have a total of $2N_t$ detection levels. For each detection level, we must execute both the table-construction stage and the candidate-update stage. In the table-construction stage, we divide the child nodes into two sets for each parent node and then arrange these child nodes into a 2^Q-by-$2K$ table based on the discretization results. For the candidate-update stage, we only need to identify where the K surviving candidates are allocated in the first L_m layers rather than directly sorting $K\sqrt{M}$ candidates. We apply the bisection-based search algorithm to determine L_m using Q iterations. After that, the candidates from layer 0 to layer $L_m - 1$ are directly selected as the $f(L_m - 1)$ surviving candidates. The remaining $K - f(L_m - 1)$ surviving candidates are selected using the non-sorting candidate-selection method so as to avoid any additional latency and complexity.

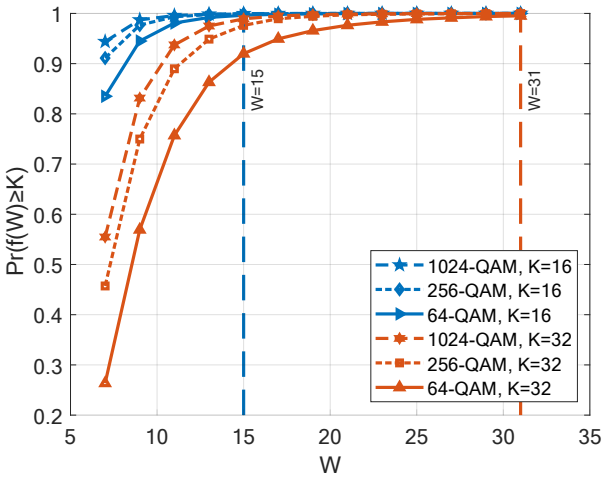


FIGURE 4. $Pr(f(W) \geq K)$ **for different values of** W**, where the SNRs considered are 30dB, 38dB and 44dB for 64-QAM, 256-QAM, and 1024-QAM systems, respectively.**

To further analyze the maximum layer index L_M after discretization, we can use the probability $Pr(f(W) \geq K)$ to ensure that there are sufficient candidates in the table. Fig. 4 shows the values for $Pr(f(W) \geq K)$ for 64-QAM, 256-QAM and 1024-QAM MIMO systems, where $K = 16$, and $K = 32$ are considered. It can be seen that if $L_M = K - 1$, $Pr(f(L_M) \geq K) \approx 0.99$ for all the considered cases. This means that the first K layers contain the K best candidates with a high probability. To further verify that $L_M = K - 1$ is suitable for obtaining the K surviving nodes, the BER performances for different values of L_M are shown in Fig. 5. It can be seem that $L_M = K-1$ is large enough to achieve almost no degradation in BER performance. In addition, the proposed sorter-free method provides a similar performance compared to the conventional K-best algorithm, since the candidates that exist in layer L_m do not form the best path and affect the detection results most of the time.

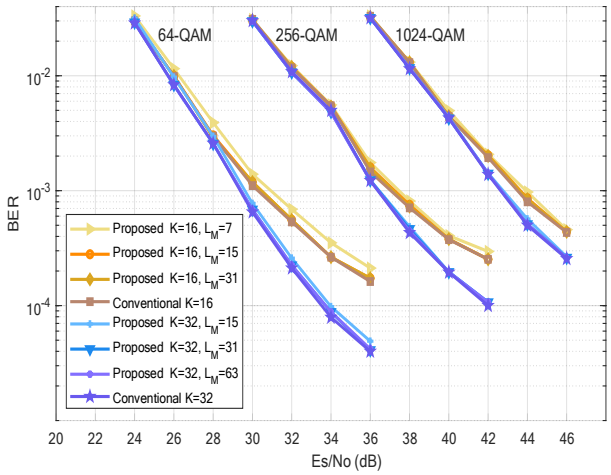


FIGURE 5. BER performance for 4×4 MIMO systems when using a variety of values for L_M**, where the Rayleigh fading channel is considered.**

Using the proposed method, the K surviving nodes can be determined using $\log_2(K)$ iterations, which provides a fixed throughput and a shorter latency compared to the K iterations needed for obtaining the K surviving nodes when using the WPE algorithm. For the proposed detector, $K(\log_2\sqrt{M} + 1)$ comparisons are used to determine the $2K$ leading elements of the constructed table based on the binary search algorithm and $\log_2(K)$ comparisons are used to determine the value of L_m. The proposed detector needs a total of $K(\log_2\sqrt{M}+1)+\log_2(K)$ comparisons to locate the K best nodes. In contrast, the WPE algorithm requires $K\sqrt{M}$ comparisons to generate the K SE lists and additional $K(K - 1)$ comparisons to determine the K best nodes. Compared to the WPE algorithm, the proposed methods can reduce the comparison operations obviously. Compared to the sorter-based algorithms, the proposed search process avoids the usage of a huge amount of comparison units and hence can reduce the hardware complexity significantly.

D. Consideration for Practical Applications

For practical communication systems, the inner MIMO detector needs to generate the log-likelihood ratio (LLR) estimates of transmitted bits for the outer channel decoder. The parallel candidate adding (PCA) algorithm presented in [37] is a hardware-friendly solution for generating soft output from the K-best detector since it can efficiently handle the bit vacancy by directly bit-flipping the hard decision path from the detector to generate the LLR estimates of transmitted

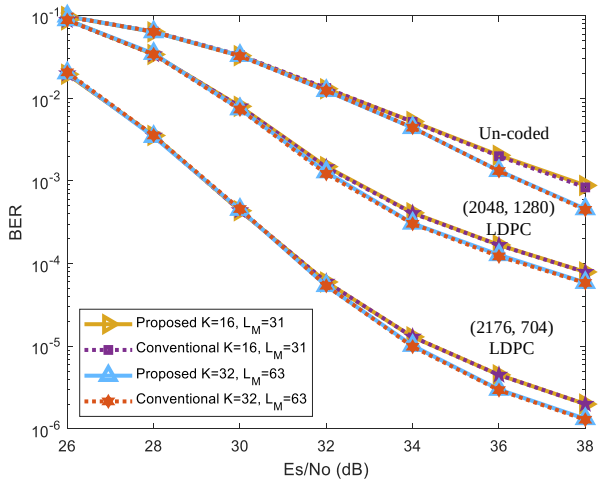


FIGURE 6. **BER performance for 4×4 MIMO systems for both the coded and un-coded cases, where the Rayleigh fading channel and the 4×4 256-QAM MIMO detector are considered.**

bits. Fig. 6 shows the error-rate performance for the coded case, where both the (2176, 704) low-density parity-check (LDPC) code and the (2048, 1280) LDPC code specified in the 5G standard [3] are considered. It can be seen that both the proposed sorter-free K-best detector and the conventional K-best detector have almost the same performance for both the coded and un-coded cases.

IV. A Fully-pipeline Detector Architecture

A. Overall Architecture

Fig. 7 shows the block diagram for the proposed high-throughput K-best MIMO detector, where a fully-pipelined architecture is used. Each level unit (LU) is used to determine the K surviving nodes and forms the paths for the corresponding detection level. Since there are $2N_t$ detection levels, $2N_t$ LUs are used. For the i-th LU, input y_i is the received element and input $\mathbf{r}_i = (r_{i,i}, r_{i,i+1}, ..., r_{i,2N_t})$ is the vector consisting of the non-zero component of the i-th row in the upper triangular matrix $\mathbf{R}$. In addition, the $i+1$-th

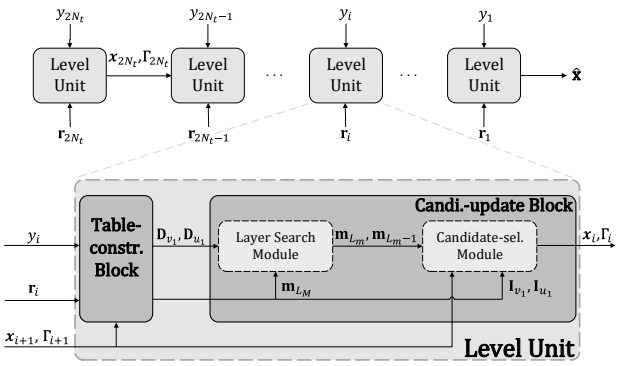


FIGURE 7. **Block diagram for the proposed K-best MIMO detector.**

Algorithm 1 Proposed sorter-free K-best detection

Output: $\mathbf{x}$: Detected transmitted vector.

```
for level == 2N_t to 1 do
    if level == 2N_t then
        expand all child nodes;
    else
        // a) table-construction stage
        for 0 ≤ j ≤ K − 1 do
            for 1 ≤ k ≤ |F^j| do
                calculate Γ^j_{v_k} using (6);
                calculate D^j_{v_k} using (7);
            end
            for 1 ≤ k ≤ |S^j| do
                calculate Γ^j_{u_k} using (6);
                calculate D^j_{u_k} using (7);
            end
        end
        // b) candidate-update stage
        initialize L_l(0) = 0 and L_u(0) = L_M;
        for 1 ≤ k ≤ log_2 K do
            calculate L_c(k − 1) using (8);
            obtain f(L_c(k − 1));
            update L_u(k) and L_l(k) using (9);
        end
        determine L_m and L_m − 1 using (10);
        select the f(L_m − 1) candidates in [L_0, L_m − 1];
        select the K − f(L_m − 1) candidates in L_m;
        if level == 1 then
            identify the smallest PED;
            x = the path that has the smallest PED;
            break;
        else
            record K paths and the K PEDs;
        end
    end
end
```

LU delivers $\mathbf{x}_{i+1}$ and $\mathbf{\Gamma}_{i+1}$ which are the detected paths and the K PEDs from the $i+1$-th detection level, respectively, to the i-th LU.

Each LU consists of two main blocks, namely, the table-construction block and the candidate-update block. In the table-construction block, the candidates are arranged into a

table using the proposed PMD. Since there is a fixed spacing between the candidates in each column, rather than recording all the candidates, we only record the PAM point together with the PED for the leading candidate in each column so as to reduce the hardware complexity, as illustrated in Fig. 3(c). As a result, we only record $\mathbf{D}_{v_1} = (D_{v_1}^0, D_{v_1}^1, ..., D_{v_1}^{K-1})$ and $\mathbf{D}_{u_1} = (D_{u_1}^0, D_{u_1}^1, ..., D_{u_1}^{K-1})$. In addition, the information for v_1 and u_1 used when calculating the PEDs and the PAM points for the remaining candidates are also recorded, which are respectively denoted as vectors $\mathbf{I}_{v_1}$ and $\mathbf{I}_{u_1}$. The details of $\mathbf{I}_{v_1}$, $\mathbf{I}_{u_1}$ and the table-construction block will be presented in Section IV.B.

The candidate-update block contains two modules, namely, the layer-search module and the candidate-selection module. In the layer-search module, L_m is determined using CBE rather than the RBE described in Section III in order to reduce the hardware latency. Letting m_L^{2j} and m_L^{2j+1} respectively denote the number of candidates in the $2j$-th and $2j+1$-th column of the table formed from layer 0 to layer L, and writing $\mathbf{m}_L = (m_L^0, m_L^1, ..., m_L^{2K-1})$, the layer-search module uses $\mathbf{m}_{L_m} = (m_{L_m}^0, m_{L_m}^1, ..., m_{L_m}^{2K-1})$ together with $\mathbf{D}_{v_1}$ and $\mathbf{D}_{u_1}$ as inputs to execute the bisection-based search that determines the value of L_m, and then outputs $\mathbf{m}_{L_m}$ and $\mathbf{m}_{L_{m-1}}$. With $\mathbf{m}_{L_m-1}$, $\mathbf{m}_{L_m}$, $\mathbf{I}_{v_1}$ and $\mathbf{I}_{u_1}$, the K surviving nodes can be generated and form the paths up to the current level i, which is denoted as $\mathbf{x}_i$. In addition, the related PEDs up to level i are also calculated and denoted as $\mathbf{\Gamma}_i$. The candidate-update block will be detailed in Section IV.C. Subsequently, $\mathbf{x}_i$ and $\mathbf{\Gamma}_i$ are passed to the following LU. After completing the detection for the final LU, the path that has the smallest PED is determined as the detected transmitted vector, and is denoted as $\hat{\mathbf{x}}$.

B. Table-Construction Block

Fig. 8 shows the table-construction block, where the detection is currently at the i-th detection level. Firstly, in each interference cancellation unit, y_i' is obtained by subtracting all the determined terms from y_i, as described in (4). All the y_i' values from K parent nodes are denoted as a length-K vector $\mathbf{y}_i'$. Secondly, each increment calculation unit (ICU) determines the PAM points and the increments from the first candidates for the corresponding two columns. The assistant information needed for calculating the PAM points and the increments for the remaining candidates are also determined. The details of this unit will be described in the following paragraphs. Thirdly, the increments for the $2K$ candidates, $\boldsymbol{\gamma}$, are added to the PEDs from the previous level, $\mathbf{\Gamma}_{i+1}$, to obtain the $2K$ PEDs for the leading candidates, $\mathbf{\Gamma}$. Lastly, each PMD unit calculates the discretization results for the leading candidates, which are respectively denoted as $\mathbf{D}_{v_1}$ and $\mathbf{D}_{u_1}$. The PEDs for v_1 and u_1 from the K parent nodes subtracted using the smallest PED, as described in (7) where $k = 1$, are respectively denoted as $\mathbf{\Gamma}_{v_1}$ and $\mathbf{\Gamma}_{u_1}$.

It is worth noting that the division and floor operations described in (7) and executed in the PMD units can be

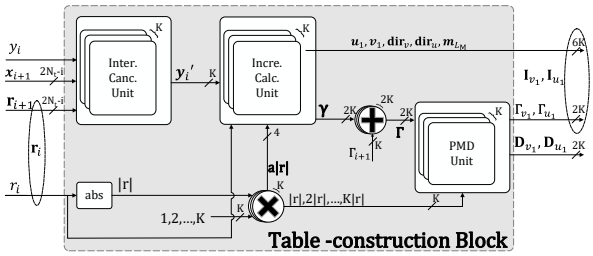


FIGURE 8. Block diagram for the table-construction block.

realized using a divider and by selecting the bits from the integer component. However, such a direct implementation approach results in high hardware complexity. In contrast, since flooring the division result for $\Gamma_{v_1}^j - \Gamma_{\min}$ in (7) determines the v_1^j layer, and there are only K layers in the table as $L_m = K-1$, the divider and the bit-selection can be replaced by K comparison units together with an adder. For instance, if $L_m = 3$, from layer 0 to layer 3, we compare $\Gamma_{v_1}^j - \Gamma_{\min}$ with $\{1|r_{ii}|, 2|r_{ii}|, 3|r_{ii}|, 4|r_{ii}|\}$ to obtain $D_{v_1}^j$, the discretization result for v_1^j. If the comparison result is $\{1, 1, 0, 0\}$, which means $2|r_{ii}| \le \Gamma_{v_1}^j - \Gamma_{\min} < 3|r_{ii}|$, then $D_{v_1}^j$ is $1+1+0+0 = 2$. Similarly, if the comparison result is $\{1, 1, 1, 1\}$ indicating $4|r_{ii}| \le \Gamma_{v_1}^j - \Gamma_{\min}$, then $D_{v_1}^j$ is $1+1+1+1 = 4$, which means $D_{v_1}^j$ is out of range from $[0, 3]$ and is set as 4. As a result, we can execute the division and the floor function in (7) using the K comparison units and an adder in order to reduce the hardware complexity.

We now detail the core concept used to devise the ICU. Since the fixed-spacing property exists in each column, we only record the PAM point and the PED for the first candidate, and the assistant information needed to obtain the remaining candidates in the same column. Since recording the PAM points and PEDs for all candidates can be avoided, the hardware complexity can be significantly reduced. The candidates from parent node j are divided into two sets, $\mathbf{F}^j$ and $\mathbf{S}^j$, and are allocated to two corresponding columns. The k-th PAM points in sets $\mathbf{F}^j$ and $\mathbf{S}^j$, which are respectively denoted as v_k^j and u_k^j, can be obtained from

$$\begin{aligned} v_k^j &= v_1^j + 2 \times (k-1) \times \mathrm{dir}_v^j, \quad 1 \le k \le |\mathbf{F}^j|, \\ u_k^j &= u_1^j + 2 \times (k-1) \times \mathrm{dir}_u^j, \quad 1 \le k \le |\mathbf{S}^j|, \end{aligned} \tag{11}$$

where $\mathrm{dir}_v^j = +1$ if the number of PAM points contained in $\mathbf{F}^j$ is increasing, e.g., $\{1, 3, 5, 7\}$, otherwise, $\mathrm{dir}_v^j = -1$ if the number of PAM points contained in $\mathbf{F}^j$ is decreasing, e.g., $\{-1, -3, -5, -7\}$. Then, the value of dir_u^j can be determined from $\mathrm{dir}_u^j = -\mathrm{dir}_v^j$. The increments for v_k^j and u_k^j, which are respectively denoted as $\gamma_{v_k}^j$ and $\gamma_{u_k}^j$, can be respectively obtained from

$$\begin{aligned} \gamma_{v_k}^j &= \gamma_{v_1}^j + 2 \times (k-1) \times |r_{ii}|, \quad 1 \le k \le |\mathbf{F}^j| \\ \gamma_{u_k}^j &= \gamma_{u_1}^j + 2 \times (k-1) \times |r_{ii}|, \quad 1 \le k \le |\mathbf{S}^j|. \end{aligned} \tag{12}$$

In summary, when using (11) and (12) to calculate the increments for v_k^j and u_k^j, a total of eight items of information are needed, which are v_1^j, u_1^j, $\gamma_{v_1}^j$, $\gamma_{u_1}^j$, $|\mathbf{F}^j|$, $|\mathbf{S}^j|$, dir_v^j and dir_u^j. In Fig. 8, the information for v_1 from the K parent nodes is denoted as $\mathbf{I}_{v_1} = (I_{v_1}^0, I_{v_1}^1, ..., I_{v_1}^{K-1})$, where $I_{v_1}^j$ includes v_1^j, $\Gamma_{v_1}^j$, $\mathrm{dir}_{v_1}^j$, $|\mathbf{F}^j|$. Similarly, the information for u_1 from the K parent nodes is denoted as $\mathbf{I}_{u_1} = (I_{u_1}^0, I_{u_1}^1, ..., I_{u_1}^{K-1})$. The aforementioned $\mathbf{m}_{L_M}$ can be expressed as $(|\mathbf{F}|^0, |\mathbf{S}|^0, |\mathbf{F}|^1, |\mathbf{S}|^1 ..., |\mathbf{F}|^{K-1}, |\mathbf{S}|^{K-1})$ if L_M is large enough.

Fig. 9 shows the hardware architecture for the ICU that obtains the eight items of information mentioned based on simple comparisons, where the 64-QAM case is considered. In order to simplify the notation, we remove the superscript j in the following discussion. Fig. 9(a) shows the circuit used to identify the first candidate, v_1. Since a symmetry exists between the positive and the negative PAM points, rather than comparing y_i' with all the combinations of $r_{ii}x_i$, where $x_i \in \mathbf{\Omega}$ is the constellation point, we compare $|y_i'|$ with $|r_{ii}x_i|$ to determine the $|x_i|$ such that $|r_{ii}x_i|$ is the closest to $|y_i'|$. In other words, we first determine $|v_1|$ by comparing $|y_i'|$ with $|r_{ii}x_i|$. After that, we use the signs of y_i' and r_{ii} to determine either $v_1 = |v_1|$ or $v_1 = -|v_1|$. In this case, we only need to compare $|y_i'|$ with the $\frac{\sqrt{M}}{2}$ combinations of $|r_{ii}x_i|$ rather than the $\sqrt{M}$ combinations of $r_{ii}x_i$ for an M-QAM MIMO system using RVD. In addition, to further reduce the number of comparisons, rather than directly comparing $|y_i'|$ with the $\frac{\sqrt{M}}{2}$ combinations of $|r_{ii}x_i|$ at the same time, a binary search method is applied in order to sequentially compare $|y_i'|$ with $|r_{ii}x_i|$. Therefore, the number of comparisons can be further reduced from $\frac{\sqrt{M}}{2}$ to $\log_2 \frac{\sqrt{M}}{2}$.

For the 64-QAM case considered in Fig. 9(a), $|r_{ii}x_i| \in \{1|r_{ii}|, 3|r_{ii}|, 5|r_{ii}|, 7|r_{ii}|\}$. Firstly, we compare $|y_i'|$ with the middle value, $4|r_{ii}|$, to identify whether $|v_1|$ is located at $\{1, 3\}$ or $\{5, 7\}$. Based on the comparison results, we secondly compare $|y_i'|$ with either $2|r_{ii}|$ or $6|r_{ii}|$. After these two comparisons have been made, $|v_1|$ is obtained. In contrast, $|u_1|$ requires us to make one more comparison, which considers whether $|y_i'|$ is larger than $|r_{ii}v_1|$, so as to determine whether $|u_1|$ is either on the left-hand side or the right-hand side of $|v_1|$. Finally, if y_i' and r_{ii} have the same signs, $v_1 = |v_1|$ and $u_1 = |u_1|$, otherwise $v_1 = -|v_1|$ and $u_1 = -|u_1|$. In a similar manner, γ_{v_1} and γ_{u_1} can be obtained, as shown in Fig. 9(b). The values for $\mathrm{dir}_{\mathrm{v_1}}$ and $\mathrm{dir}_{\mathrm{u_1}}$ are obtained after a comparison of $|y_i'|$ with $|r_{ii}v_1|$ and a comparison of the signs of y_i' and r_{ii}, as illustrated in Fig. 9(c). Lastly, to determine $|\mathbf{F}|$ and $|\mathbf{S}|$, we utilize $|v_1|$ and the comparison of $|y_i'|$ and $|r_{ii}v_1|$, as illustrated in Fig. 9(d).

In conclusion, for an M-QAM MIMO system, $\log_2\sqrt{M}+1$ comparisons are used to determine $|v_1|$ and $|u_1|$ based on the binary search algorithm. Subsequently, one more comparison is applied in order to determine v_1 and u_1 based on the signs of y_i' and r_{ii}. The other six items of the information are also obtained when the $\log_2\sqrt{M}+1$ comparisons are executed. It is worth noting that $|v_1|$, which includes a single right-shifting and the value of $\frac{\sqrt{M}}{2}$ are utilized to determine $|\mathbf{F}|$ and $|\mathbf{S}|$.

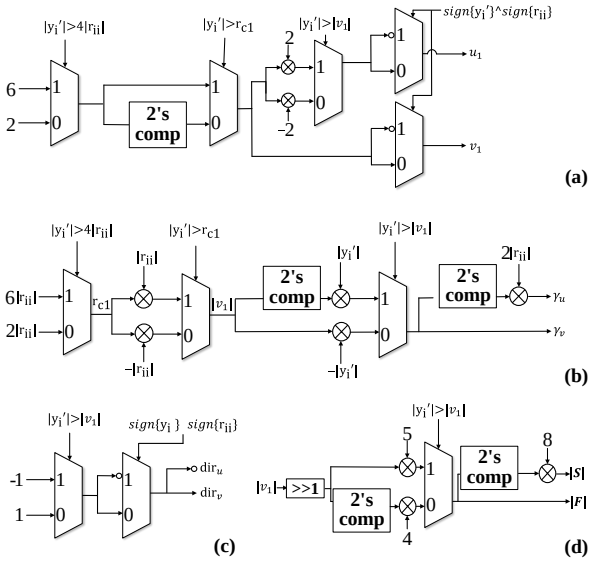


FIGURE 9. Architecture for the increment calculation unit (ICU), where 64-QAM is considered.

Compared to the conventional architecture using SE enumeration, where a LUT is used to obtain the PAM points and the related PEDs, the proposed method utilizes fewer comparisons to obtain the information needed. In general, for an M-QAM MIMO system, the LUT applies $\sqrt{M}-1$ comparisons to obtain the absolute value for the first PAM point. In addition, one more comparison unit is used in order to determine the sign of the first PAM point. From the results, a total of $\sqrt{M}$ comparison units are applied in order to generate the SE list. In contrast, the proposed method requires only $\log_2\sqrt{M}+1$ comparisons to obtain the information needed. In addition, the eight items of information is independent of the modulation size, while the $\sqrt{M}$ PAM points and the $\sqrt{M}$ related PEDs and, hence, a total of $2\sqrt{M}$ items of information have to be recorded for the conventional SE enumeration.

A summary between the proposed ICU and the SE enumeration approach is given in Table 1, where the number of comparisons used is denoted as N_c, and the items of information recorded are denoted as N_i. Compared to the conventional SE enumeration technique, the number of comparisons can be reduced when using the proposed method. In addition, the reduction in storage complexity increases as the modulation size increases.

C. Candidate-update Block

The candidate-update block consists of two modules, the layer search module, which is used to determine the value

TABLE 1. Comparison between SE enumeration and the proposed method.

Modulation (QAM)		64	256	1024
SE enumeration	N_c	8	16	32
	N_i	16	32	64
Proposed increment unit	N_c	4	5	6
	N_i	8	8	8
Reduction (%)	N_c	50	68.7	81.3
	N_i	50	75	87.5

of L_m, and the candidate-selection module, which is responsible for storing the paths and related PEDs for the surviving nodes. Fig. 10 shows the block diagram for the layer search module, where each search unit (SU) executes all the steps in an iteration round of the proposed bisection-based layer search, as described in (8) and (9). Firstly, L_c is calculated using (8). After that, the SU computes the number of candidates from layer 0 to layer L_c, i.e., $f(L_c)$. It is worth noting that if $f(L_c)$ is obtained using RBE, a long latency is induced, since there is a need to check the columns individually in order to determine whether there are any candidates located in these $2K$ columns so as to obtain the number of candidates in a single layer. Furthermore, a total of L_c+1 layers need to be checked for a single iteration. As a result, there are a total of $2K \times (L_c + 1)$ checking operations that need to be executed in order to obtain $f(L_c)$, resulting in a long latency.

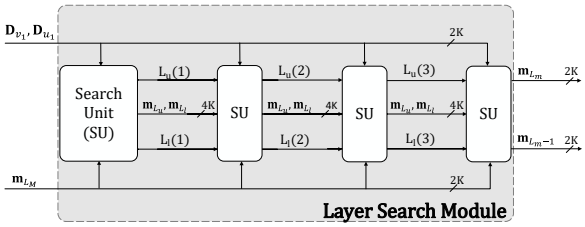


FIGURE 10. Block diagram for the layer search module, where $K = 16$.

To address this problem, rather than using RBE, we apply column-based enumeration, which provides a high parallelism and simple calculation so as to obtain $f(L_c)$. Since $f(L_c) = \sum_{j=0}^{K-1}(m_{L_c}^{2j} + m_{L_c}^{2j+1})$, we can first calculate the number of candidates in each column by utilizing the fixed-spacing property and then compute the number of candidates in these $2K$ columns to obtain $f(L_c)$. The example shown in Fig. 3 is used to clearly describe the process. For the first layer search iteration, $L_c = 3$ and $D_{v_1}^0 = 0$, we can simply obtain the number of candidates in the first column using $\lfloor \frac{3-0}{2} \rfloor + 1$, since the fixed spacing is 2. The number of candidates in the remaining columns can be obtained in a similar manner. However, two exceptions need to be considered. The first is when the calculation result is larger than the number of candidates in the column. In the first iteration, the calculation result for the second column illustrated in Fig. 3 is 2 but there is only a single candidate in the column. In this case, we directly assign the number of candidates in the column as the result. The other exception occurs when L_c is smaller than the discretization result for the first candidate, as the calculation in the fourth column for the second layer search iteration, where $D_{u_1}^1 = 2$ but $L_c = 1$. As a result, there are no candidates in the corresponding column and the result is assigned as 0.

In summary, $m_{L_c}^{2j}$ and $m_{L_c}^{2j+1}$ can be respectively obtained from

$$m_{L_c}^{2j} = \begin{cases} 0, & L_c < D_{v_1}^j \\ \lfloor \frac{L_c - D_{v_1}^j}{2} \rfloor + 1, & \lfloor \frac{L_c - D_{v_1}^j}{2} \rfloor + 1 < |\mathbf{F}^j| \\ |\mathbf{F}^j|, & \lfloor \frac{L_c - D_{v_1}^j}{2} \rfloor + 1 \geq |\mathbf{F}^j| \end{cases} \tag{13}$$

and

$$m_{L_c}^{2j+1} = \begin{cases} 0, & L_c < D_{u_1}^j \\ \lfloor \frac{L_c - D_{u_1}^j}{2} \rfloor + 1, & \lfloor \frac{L_c - D_{u_1}^j}{2} \rfloor + 1 < |\mathbf{S}^j| \\ |\mathbf{S}^j|, & \lfloor \frac{L_c - D_{u_1}^j}{2} \rfloor + 1 \geq |\mathbf{S}^j|. \end{cases} \tag{14}$$

After obtaining the number of candidates in each column, we sum the results for these $2K$ columns to obtain $f(L_c)$. Using $f(L_c)$, L_l and L_u are updated by comparing $f(L_c)$ with K, as described in (9), which are then passed to the next SU to begin the next iteration.

Since $K = 16$ is considered in Fig. 10, four SUs are used to execute four iterations. In the final iteration, we have $\mathbf{m}_{L_m-1}$ and $\mathbf{m}_{L_m}$. Note that the divisions present in (13) and (14) can be implemented using right shifters, since they are divided by 2. In addition, the operation of adding 1 can also be implemented using a low-complexity incrementer. As a result, the calculations in the SUs can be implemented by employing a low-complexity division-free architecture.

The candidate selection module uses the information recorded in the table construction block, $\mathbf{I}_{v_1}$, $\mathbf{I}_{u_1}$, $\mathbf{m}_{L_m-1}$, and $\mathbf{m}_{L_m}$, to generate the PAM points and the related PEDs for the K surviving nodes, as illustrated in Fig. 11. Instead of directly applying the inputs to generate the surviving nodes, these inputs are first passed through a $2K$-to-K mapper unit. Since the K surviving nodes are allocated, at most, in K different columns, the $2K$-to-K mapper unit only preserves the values in $\mathbf{m}_{L_m-1}$ that do not equal 0 as $\mathbf{m}'_{L_m-1}$ and saves the corresponding information as $\mathbf{I}_p$. Similarly, $\mathbf{m}_{L_m}$ is downsized as $\mathbf{m}'_{L_m}$ and the corresponding information is saved as $\mathbf{I}_f$. In this case, the hardware complexity is reduced by implementing the process of generating surviving nodes for K times rather than $2K$ times.

Based on the proposed detection method, the pick-up unit manages the surviving nodes from layer 0 to layer $L_m - 1$ based on $\mathbf{m}'_{L_m-1}$ and $\mathbf{I}_p$. The pick-up unit implements a sequential process that generates the surviving nodes and forms the paths according to the address, $\mathbf{addr}_p$. The calculation of the address is basically achieved by individually summing the values of $\mathbf{m}'_{L_m-1}$ in the address calculation unit. For instance, if $\mathbf{m}_{L_m} = (1, 2, 3, 4)$ then $\mathbf{addr}_{\mathrm{p}} = (0, 1, 3, 6)$,

TABLE 2. Summary of the proposed 4×4 MIMO detectors

Clock frequency (MHz)	200					
Process (nm)	90					
Supply voltage (V)	1.1V					
K	16			32		
Modulation (QAM)	64	256	1024	64	256	1024
Combinational EGC (KG)	684.77	1408.24	1960.16	930.40	1866.78	2530.22
Non-combinational EGC (KG)	168.53	434.04	736.44	337.06	868.08	1472.88
Total EGC (KG)	853.3	1842.28	2696.6	1267.46	2734.86	4003.1
Latency (μs)	0.29	0.32	0.35	0.34	0.38	0.4
Throughput (Mbps)	4800	6400	8000	4800	6400	8000
Power (mW)	224	483	707	333	717	1050
Power Efficiency (Gb/J)	21.42	13.25	11.31	14.41	8.92	7.62
Hardware Efficiency (Mbps/KG)	5.63	3.47	2.97	3.79	2.34	5.43

considering the address starts at 0. Alternatively, the pick-up unit utilizes $\mathbf{I}_p$ to generate the PEDs and PAM points for the candidates and then individually places them in a single column, as described in (6) and (11). The PEDs and PAM points for the candidates are then sequentially placed in the registers based on $\mathbf{addr}_{\mathrm{p}}$.

Similarly, the fill-up unit uses $\mathbf{m}'_{L_m}$ and $\mathbf{I}_f$ to complete any vacancies for the undetermined surviving nodes. Note that the candidates that exist in layer L_m are selected immediately after those from layer 0 to layer $L_m - 1$. As a result, the address unit forwards the sum of $\mathbf{m}'_{L_m-1}$, namely sum, to the fill-up unit as the first address in order to select the candidate. The fill-up unit then uses a similar sequential process to complete any vacancies for the surviving nodes. Finally, the results from the pick-up unit and the results from the fill-up unit are combined through the OR gate in order to obtain the K surviving nodes and the related PEDs.

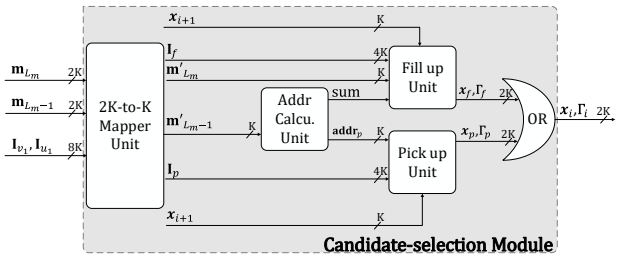


FIGURE 11. Block diagram for the candidate-selection module.

In this work, we apply the sequential process to generate the surviving nodes in both the pick-up unit and the fill-up unit. We then combine the results from these two units, which can be regarded as a method of parallelizing these two sequential processes. To improve the throughput, the sequential process that generates the surviving nodes in both the pick-up unit and the fill-up unit can be replaced with parallel generating processes at the expense of a higher hardware complexity for storage. It is worth noting that, since we employ the pick-up unit and the fill-up unit to simultaneously generate the surviving nodes, the storage size is $2M_s$ in this work, where M_s is the storage size for the paths and the PEDs for the K candidates. However, to further improve throughput, the sequential process that generates the surviving nodes in the pick-up unit and the fill-up unit can be replaced with the fully-parallel process, which also results in a larger storage size as $2K \times M_s$,

V. Performance Evaluation

A. Implementation Results

The proposed sorter-free fully-pipelined MIMO detector is designed for 4×4 MIMO systems, where 64-QAM, 256-QAM and 1024-QAM are considered. In addition, both $K = 16$ and $K = 32$ are used. The hardware is designed based on a 90 nm CMOS process, and is synthesized using Synopsys Design Compile. For the proposed 64-QAM, 256-QAM and 1024-QAM MIMO detectors, the quantization schemes used are $[6, 7]$, $[8, 10]$ and $[9, 12]$, respectively, where the form $[n,m]$ represents that there are n bits for the integer component, including the sign bit, and m bits for the fraction component. It can be seen from Fig. 12 that the performance loss due to quantization is less than 0.1dB for all the cases considered.

Table 2 summarizes the hardware results for the proposed MIMO detectors, where the equivalent gate count (EGC) [32] is measured by counting a two-input NAND gate as a single element. As the modulation size increases, the number of comparisons used in the ICU increases, since the sequential binary search method is adopted to determine the positive PAM points when obtaining the leading candidates. In addition, the sequential process that generates the surviving nodes and forms the paths also creates a greater latency as the modulation size increases. In this case, to retain the same operating clock frequency, additional pipelined registers are used. As a result, both the EGC for the non-combinational circuit and the total latency increase as the modulation size increases. However, the architecture for the sequential binary search process when obtaining the

TABLE 3. Comparison When Obtaining K Survivor Nodes

K		16			32		
QAM		64	256	1024	64	256	1024
Hybrid [32]	CASU	608	920	N/A	1832	3104	4368
	EGC (KG)	115.77	276.53	N/A	339.33	865.75	1768.67
	NL (ns)	8.48	9.26	N/A	12.79	14.44	15.05
Proposed	SU	4	4	4	5	5	5
	EGC (KG)	85.33	97.87	178.9	103.02	158.31	216.20
	NL (ns)	7.94	8.16	8.42	11.43	11.86	12.32
EGC reduction (%)		26.29	64.61	N/A	69.64	81.71	87.77
NL reduction (%)		6.36	11.87	N/A	10.6	17.86	18.13

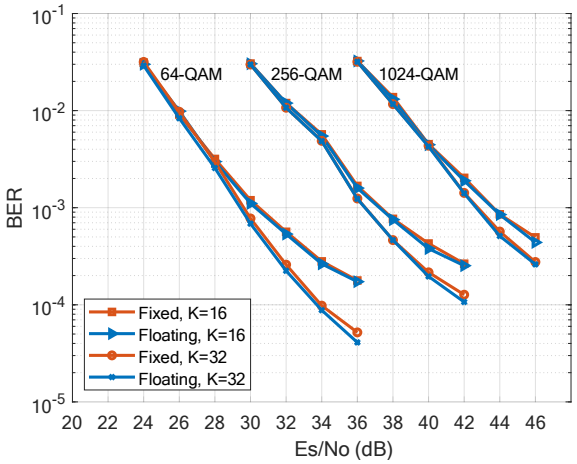


FIGURE 12. Floating-point and fixed-point BER performances for the proposed 4×4 MIMO detectors.

leading candidates and generating the surviving nodes can be paralleled in order to prevent any increase in latency.

Fig. 13 shows the progressive improvement to a baseline design when using the proposed techniques, where both the proposed detector and the baseline detector can support the 4×4 256-QAM MIMO systems. The baseline is designed based on the hybrid sorter presented in [32], where a fully-pipelined detector architecture and the techniques mentioned in the previous sections, including the SE enumeration and the simplified sorting-network, are used. Applying the proposed bisection-based layer search that includes RBE and a LUT used to obtain the eight information items, the hardware efficiency increases by 27%, since the bisection-based layer search avoids the need to employ the sorting-network used in the baseline detector. However, the RBE causes an increase in the number of cycles needed in order to obtain $f(L_c)$ under the same clock frequency, since $2K \times (L_c + 1)$ checking operations are needed in order to determine $f(L_c)$. To further improve the hardware efficiency, we replace the LUT with the proposed ICU to obtain the information for the leading candidates from the K parent nodes. Once the leading candidates are obtained, all the other information items can be obtained using simple calculations, which are independent of the modulation size. Compared to the LUT based method, the proposed method that is based on the ICU reduces the storage complexity and, hence, the hardware efficiency can be further increased by 12%. Finally, the RBE is replaced with the CBE to obtain $f(L_c)$. It is worth noting that the calculation in each column can be simultaneously executed and, hence, the cycle counts can be reduced and the related pipelined registers are also removed. As a result, the hardware efficiency for the proposed detector based on both the ICU and CBE is improved by 77% compared to the sorter-based baseline design.

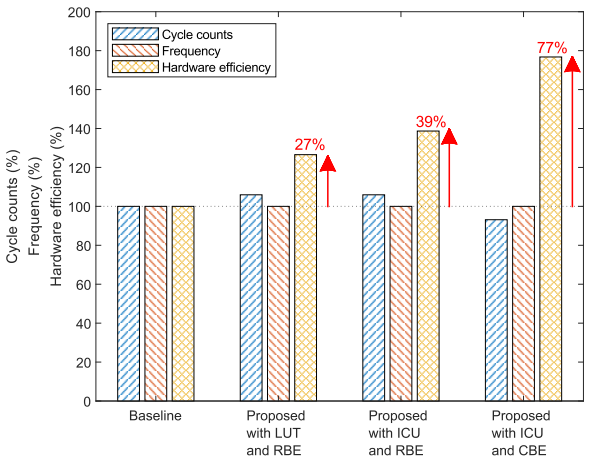


FIGURE 13. Progressive improvement when replacing RBE with CBE for the proposed architecture, where the hardware performance for the baseline design is normalized to 100%.

B. Comparison When Obtaining K Survivor Nodes

Table 3 summarizes the hardware complexity when generating the surviving nodes, where both the proposed design and the hybrid sorter described in [32] are considered. Since the hardware result provided in [32] only includes cases up to the 64-QAM MIMO system, we estimate the hardware complexity and the latency for 256-QAM and 1024-QAM cases by calculating the number of compare-and-swap units (CASU) as described in [32]. Since the formula used to calculate the number of CASUs is only available for $K \leq \sqrt{M}$,

TABLE 4. Comparison of 4×4 MIMO Detectors

	[34] Synthesis			[35] Post layout	[36] Post layout	[33] Post layout	This work Synthesis
Modulation (QAM)	64			64	64	64	256
Bits/MIMO symbol	24			24	24	24	32
K	5	10	64	10	8∼1[a]	16	16
Algorithm	WPE			WPE	WPE	hybrid	sorter-free
Process (nm)	65			130	90	65	90
Supply voltage (V)	1V			1.3V	1.3V	1.3V	1.1V
EGC (KG)	1760			114	232	285	1842.28
Frequency (MHz)	158			282	170	137	200
Throughput (Mbps)	732	463	100	655	4080	109	6400
NL (μs)	N/A			0.58	0.22	N/A	0.32
Power (mW)	165			135	234	34	483
NPE (Gb/J)	4.44	2.80	0.61	8.20	29.47	5.42	16.03
NHE (Mbps/KG)	0.41	0.26	0.05	5.74	17.58	0.38	3.47

[a] The values for K from the $2N_t$-th level to the first level are [8,8,8,8,4,2,2,1].

the estimated results for the hybrid sorter for 1024-QAM where $K = 16$ are not shown in Table 3. To fairly compare the hardware results for the different CMOS processes, the normalized latency (NL), the normalized power efficiency (NPE) and normalized hardware efficiency (NHE) are also considered [36], which are calculated using

$$\mathrm{NL} = \mathrm{latency} \times (\frac{90}{\mathrm{process}}), \tag{15}$$

$$\mathrm{NPE} = \frac{\mathrm{throughput} \times (\frac{90}{\mathrm{process}})}{\mathrm{power} \times (\frac{1.0}{\mathrm{voltage}})^2}, \tag{16}$$

$$\mathrm{NHE} = \frac{\mathrm{throughput} \times (\frac{90}{\mathrm{process}})}{\mathrm{EGC}}. \tag{17}$$

It can be seen from Table 3 that, as the modulation size increases, the hybrid sorter requires additional CASUs in order to sort the best K surviving nodes from all the candidates. In addition, the number of CASUs also increases when K becomes larger. As a result, the equivalent gate count grows when either the modulation size or K increases. In contrast, the proposed bisection method retains a similar complexity when the modulation size increases, since the number of SUs remains the same. Moreover, when K changes from 2^p to 2^{p+t}, only t additional SUs are needed, since there are only t additional iterations in the search process. In this case, the proposed method greatly reduces the hardware complexity when large modulation constellations and large K values are used, in comparison to the hybrid sorter. In conclusion, the EGCs for the proposed detector remain similar for high-order modulation systems, since the hardware complexity is mainly related to K, not the modulation size. The proposed method also contains a shorter NL compared to the hybrid-sorter based detectors, especially for high-order modulation situations. As a result, when $K = 32$ and 1024-QAM are considered, the proposed method reduces the required number of EGCs by 87.77% and reduces the amount of NL by 18.13%.

C. Comparison between Existing MIMO Detectors

A comparison with the state-of-the-art 4×4 64-QAM MIMO detectors is provided in Table 4. It can be seen from the table that a lower hardware efficiency is observed as the value of K increases for WPE-based MIMO detectors [34]. The reason is that the WPE requires a K-iteration sequential process in order to obtain a single surviving node each time, and, hence, the increase in the value of K increases the latency when obtaining the K surviving nodes. To further reduce the hardware complexity, several techniques were proposed in [35]. In [36], a variety of K values were used. Specifically, the value of K decreases as the detection level continues, which reduces the latency of the sequential process for the WPE in the last few detection levels. In addition, the values of K are not greater than $\sqrt{M} = 8$ for the 64-QAM MIMO detector presented in [36]. As a result, the fully-pipelined detector presented in [36] is able to efficiently complete the search process when obtaining the K surviving nodes based on a short latency. Therefore, the hardware efficiency is further improved with the adaptive and relatively small values for K.

Since the values of K used in these works are different, which leads to different BER performances, we adjust the value of K for the proposed architecture, as illustrated in Fig. 14, in order to compare the hardware performance based on the same target BER of 10^{-3}. Compared to the WPE-based MIMO detector using a relatively small value of K, e.g., the detector presented in [36], the proposed MIMO detector provides a comparable hardware efficiency. However, as the value of K increases, our proposed MIMO detector is able to further improve the hardware efficiency, since the proposed bisection-based search process requires $\log_2 K$ iterations rather than K iterations in order to determine the K

surviving nodes. As a result, the proposed detector is able to achieve a much higher hardware efficiency compared to the WPE-based MIMO detector presented in [35]. Compared to the MIMO detector using a hybrid sorter [33], the normalized hardware efficieny (NHE) for the proposed detector is much better, since the proposed detection method greatly reduces the hardware complexity by replacing the sorting-network with the bisection-based search process.

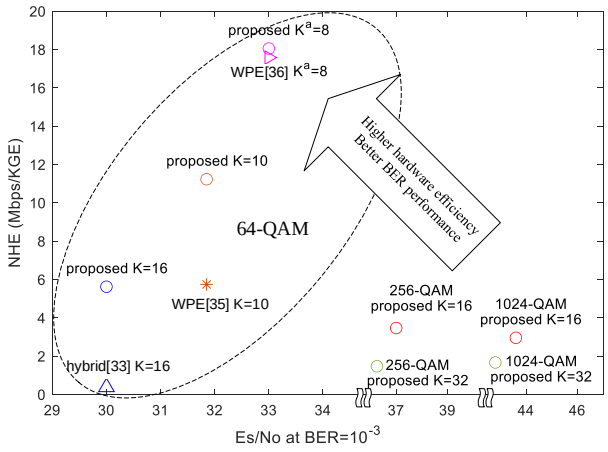


FIGURE 14. Hardware performance for the 4×4 MIMO detectors, where the values of K from the $2N_t$-th level to the first level of K^a is [8,8,8,8,4,2,2,1].

Also included in both Table 4 and Fig. 14 are the results for the proposed detectors based on 4×4 256-QAM MIMO and 4×4 1024-QAM MIMO systems. In conclusion, when high-order modulation, such as 256-QAM is used in the MIMO system, our proposed MIMO detector yields a lower hardware complexity than the sorter-based detectors, since the sorter-free bisection-based algorithm is applied. In addition, as the value of K increases in order to retain a comparable performance for high-order modulations, the improvement in NHE for the proposed MIMO detector compared to the WPE-based and the sorter-based MIMO detectors also increases. In other words, the significant improvement in hardware efficiency for the proposed MIMO detector is revealed when a relatively large value for K is applied in the high-order modulation MIMO system.

We have demonstrated the advantage in hardware efficiency of the proposed scheme over the other works presented in Tables 3 and 4 based on the 4×4 MIMO case. Our proposed method is able to efficiently select the best K candidates from $K\sqrt{M}$ nodes without using sorters. It is worth noting that the proposed K-best detector performs the same operation at each layer of the detection tree, i.e., computes the Euclidean distance, selects the best K candidates, and expands the selected nodes for the next layer. Based on such regularity, we can extend our scheme from a 4×4 MIMO system to large-scale MIMO systems using the same structure.

VI. Conclusion

We have presented an efficient sorter-free K-best MIMO detection algorithm and its associated memory-efficient architecture. The $K\sqrt{M}$ candidates are first arranged into a multi-layer table using path metric discretization and then the bisection-based method is used to obtain the locations of the K surviving candidates. The K surviving nodes can be obtained from $\log_2 K$ rounds, which is independent of the modulation order M. For MIMO systems using high-order modulation, the proposed detector provides a better hardware efficiency in comparison with existing sorter-based and WPE-based MIMO detectors. Moreover, the proposed low-complexity and fully-pipelined architecture allows us to achieve a gigabit throughput in a realizable hardware complexity.

REFERENCES

[1] S. Yang and L. Hanzo, "Fifty Years of MIMO Detection: The Road to Large-Scale MIMOs," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 1941-1988, 2015.

[2] E. Khorov, A. Kiryanov, A. Lyakhov and G. Bianchi, "A Tutorial on IEEE 802.11ax High Efficiency WLANs," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 197–216, Mar. 2019.

[3] "3GPP TS 38.214, "Physical layer procedures for data (3GPP TS 38.214 version 16.2.0 Release 16)," 2020.

[4] A. Klein, G. K. Kaleh, and P. W. Baier, "Zero forcing and minimum mean-square-error equalization for multiuser detection in code-division multiple-access channels," *IEEE Trans. Veh. Technol.*, vol. 45, no. 2, pp. 276–287, 1996.

[5] A. Trimeche, N. Boukid, A. Sakly, and A. Mtibaa, "Performance analysis of ZF and MMSE equalizers for MIMO systems," in *Proc. IEEE DTIS*, 2012, pp. 1–6.

[6] M. Wu, B. Yin, G. Wang, C. Dick, J. R. Cavallaro, and C. Studer, "Large-scale MIMO detection for 3GPP LTE: Algorithms and FPGA implementations," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 5, pp. 916–929, 2014.

[7] Y. Jiang, M. K. Varanasi, and J. Li, "Performance analysis of ZF and MMSE equalizers for MIMO systems: An in-depth study of the high SNR regime," *IEEE Trans. Inf. Theory*, vol. 57, no. 4, pp. 2008–2026, 2011.

[8] E. Viterbo and J. Boutros, "A universal lattice code decoder for fading channels," *IEEE Trans. Inf. Theory*, vol. 45, no. 5, pp. 1639–1642, 1999.

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

[10] A.D. Murugan, H. El Gamal, 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, 2006.

[11] 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, 2006.

[12] K.W. Wong, C.Y. Tsui, R.S. Cheng, and W.H. Mow, "A VLSI architecture of a K-best lattice decoding algorithm for MIMO channels," in *Proc. IEEE ISCAS*, 2002, vol. 3, pp. III–III.

[13] Z. Guo and P. Nilsson, "A VLSI architecture of the Schnorr-Euchner decoder for MIMO systems," in *Proc. IEEE 6th Circuits Syst. Symp. Emerging Technol., Frontiers Mobile Wireless Commun.*, 2004, vol. 1, pp. 65–68 Vol.1.

[14] J. Jaldén and B. Ottersten, "On the complexity of sphere decoding in digital communications," *IEEE Trans. Signal Process.*, vol. 53, no. 4, pp. 1474–1484, Apr. 2005.

[15] J. Anderson and S. Mohan, "Sequential coding algorithms: A survey and cost analysis," *IEEE Trans. Commun.*, vol. COM: 32, no. 2, pp. 169–176, Feb. 1984.

[16] D. Wübben, R. Bohnke, J. Rinas, V. Kuehn, and K.D. Kammeyer, "Efficient algorithm for decoding layered space-time codes," *Electron. Lett.*, vol. 37, pp. 1348 – 1350, 11 2001.

[17] D. Wubben, R. Bohnke, V. Kuhn, and K. . Kammeyer, "MMSE extension of V-BLAST based on sorted QR decomposition," in *Proc. IEEE VTC – Fall*, 2003, vol. 1, pp. 508–512 Vol.1.

[18] T. Kim, "Low-complexity sorted QR decomposition for MIMO systems based on pairwise column symmetrization," *IEEE Trans. Wireless Commun.*, vol. 13, no. 3, pp. 1388–1396, 2014.

[19] M. A. Albreem, M. Juntti, and S. Shahabuddin, "Massive MIMO detection techniques: A survey," *IEEE Commun. Surv. Tutor.*, vol. 21, no. 4, pp. 3109–3132, 2019.

[20] Y. H. Gan, C. Ling, and W. H. Mow, "Complex lattice reduction algorithm for low-complexity full-diversity MIMO detection," *IEEE Trans. Signal Process.*, vol. 57, no. 7, pp. 2701–2710, 2009.

[21] Q. Wen, Q. Zhou, and X. Ma, "An enhanced fixed-complexity LLL algorithm for MIMO detection," in *Proc. IEEE Global Commun. Conf.*, 2014, pp. 3231–3236.

[22] C. F. Liao, J. Y. Wang, and Y. H. Huang, "A 3.1 Gb/s 8 × 8 sorting reduced K-Best detector with lattice reduction and QR decomposition," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 12, pp. 2675–2688, 2014.

[23] G. Peng, L. Liu, S. Zhou, Y. Xue, S. Yin, and S. Wei, "Algorithm and architecture of a low-complexity and high-parallelism preprocessing-based K-best detector for large-scale MIMO systems," *IEEE Trans. Signal Process.*, vol. 66, no. 7, pp. 1860–1875, 2018.

[24] Z. Liang, D. Lv, C. Cui, H. B. Chen, W. He, W. Sheng, N. Jing, Z. Mao, and G. He, "A 3.85-Gb/s 8 × 8 soft-output MIMO detector with lattice-reduction-aided channel preprocessing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 29, no. 2, pp. 307–320, 2021.

[25] Y. Liao and T. Hsu, "A cost-effective adaptive overlapped cluster-based MIMO detector in a frequency domain reconfigurable modem," *IEEE Access*, vol. 7, pp. 36103–36121, 2019.

[26] C. Xiong, X. Zhang, K. Wu, and D. Yang, "A simplified fixed-complexity sphere decoder for V-BLAST systems," *IEEE Commun. Lett.*, vol. 13, no. 8, pp. 582–584, Aug. 2009.

[27] K. Lai, C. Huang, and J. Jia, "Variation of the fixed-complexity sphere decoder," *IEEE Commun. Lett.*, vol. 15, no. 9, pp. 1001–1003, Sep. 2011.

[28] L. Liu, J. Lofgren, and P. Nilsson, "Area-efficient configurable high-throughput signal detector supporting multiple MIMO modes," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 59, no. 9, pp. 2085–2096, 2012.

[29] C.P. Schnorr and M. Euchner, "Lattice basis reduction: Improved practical algorithms and solving subset sum problems," *Mathematical programming*, vol. 66, no. 1, pp. 181–199, 1994.

[30] K. E. Batcher, "Sorting networks and their applications," in *Proc. AFIPS*, Apr./May 1968, pp. 307–314.

[31] R. C. Chang, M. Wei, H. Chen, K. Lin, H. Chen, Y. Gao, and S. Lin, "Implementation of a high-throughput modified merge sort in MIMO detection systems," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 9, pp. 2730–2737, 2014.

[32] B. Y. Kong and I. Park, "Improved sorting architecture for K-best MIMO detection," *IEEE Trans. Circuits Syst., II, Exp. Briefs*, vol. 64, no. 9, pp. 1042–1046, 2017.

[33] I. A. Bello, B. Halak, M. El-Hajjar, and M. Zwolinski, "Hardware implementation of a low-power K-best MIMO detector based on a hybrid merge network," in *Proc. IEEE PATMOS*, 2018, pp. 191–197.

[34] S. Mondal, A. Eltawil, C. Shen, and K. N. Salama, "Design and implementation of a sort-free K-Best sphere decoder," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 10, pp. 1497–1501, 2010.

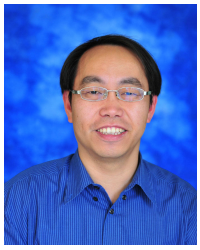
[35] M. Shabany and P. G. Gulak, "A 675 Mbps, 4 × 4 64-QAM K-best MIMO detector in 0.13 μm CMOS," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 1, pp. 135–147, 2012.

[36] M. Huang and P. Tsai, "Toward multi-gigabit wireless: Design of high-throughput MIMO detectors with hardware-efficient architecture," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 2, pp. 613–624, 2014.

[37] X. Chen, G. He and J. Ma, "VLSI implementation of a high-throughput iterative fixed-complexity sphere decoder," *IEEE Trans. Circuits Syst. II: Express Briefs*, vol. 60, no. 5, pp. 272-276, May 2013.

Yu-Xin Liu received his B.S. and M.S. degree in electrical engineering from the National Tsing Hua University, Hsinchu, Taiwan, in 2018 and 2021, respectively. He is currently a digital IC designer in MediaTek Company. His research interests include MIMO detections and VLSI designs.

SHIH-JIE JHANG received his B.S. degree in electrical engineering from the Yuan Ze University, Taoyuan, Taiwan, in 2020. He is currently working toward a Master's degree in the Department of Electrical Engineering, National Tsing Hua University, Hsinchu, Taiwan. His research interests include MIMO detections and VLSI designs.



Yeong-Luh Ueng (M'05-SM'15) received his Ph.D. degree in communication engineering from the National Taiwan University, Taipei, Taiwan, in 2001. From 2001 to 2005, he was with a private communication technology company, where he focused on the design and development of various wireless chips. Since December 2005, he has been a member of the faculty at the National Tsing Hua University, Hsinchu, Taiwan, where he is currently a Full Professor with the Department of Electrical Engineering and the Institute of Communications Engineering. In 2016, he was the recipient of the Wu Ta-You Memorial Award from the Ministry of Science and Technology (MOST). In 2018, he was the recipient of the Outstanding Electrical Engineering Professor award from the Chinese Electrical Engineering Association, as well as the Outstanding Research Award from the Ministry of Science and Technology (MOST), Taiwan. His research interests include coding theory, wireless communications, and communication ICs.