Design and Implementation of an Improved Soft-output MIMO Detector

by

Chen Shen

A Thesis
Submitted to the Faculty
of the
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements for the
Degree of Master of Science
in
Electrical and Computer Engineering
by

January 2010

APPROVED:

Professor Xinming Huang, Major Advisor

Professor Berk Sunar

Professor Andrew G. Klein

Abstract

Multiple-input multiple-output (MIMO) technique in communication system has been widely researched. Compared with single-input single-output (SISO) communication, its properties of higher throughput, more efficient spectrum and usage make it one of the most significant technology in modern wireless communications. In MIMO system, sphere detection is the fundamental part. The purpose of traditional sphere detection is to achieve the maximum likelihood (ML) demodulation of the MIMO system. However, with the development of advanced forward error correction (FEC) techniques, such as the Convolutional code, Turbo code and LDPC code, the sphere detection algorithms that can provide soft information for the outer decoder attract more interests recently. Considering the computing complexity of generating the soft information, it is important to develop a high-speed VLSI architecture for MIMO detection.

The first part of this thesis is about MIMO sphere detection algorithms. Two sphere detection algorithms are introduced. The depth first Schnorr-Euchner (SE) algorithm which generates the ML detection solution and the width first K-BEST algorithm, which only generates the nearly-ML detection solution but more efficient in implementation are presented. Based on these algorithms, an improved nearly-ML algorithm with lower complexity and limited performance lose, compared with traditional K-BEST algorithms, is presented.

The second part is focused on the hardware design. A 4×4 16 QAM MIMO detection system which can generate both soft information and hard decision solution is designed and implemented in FPGA. With the fully pipelined and parallel structure, it can achieve a throughput of 3.7 Gbps. In this part, the improved nearly-ML algorithm is implmented as a detector to generat both the hard output and candidate list. Then, a soft information calculation block is designed to succeed the detector and produce the log-likelihood ratio (LLR) values for every bit as the soft output.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my respected advisor, Professor Xinming Huang for his incessant advice and patient guidance throughout the course of my study and research at Worcester Polytechnic Institute. His ample knowledge, rigorous working attitude, honest personality and eagerness for new technology are always my model to follow in my future study and work.

As well, I would like to thank all the people of department of Electrical and Computer Engineer, for providing such an excellent study and research environment. My special thanks give to my committee member, Professor Berk Sunar and Professor Andrew G. Klein, for their help and advice in my defense and study.

I would also like to thank all my friends at Worcester Polytechnic Institute for their friendships and support during my past study life, especially my group members, Mr. Wenxuan Guo, Mr. Kai Zhang and Mr. Yanjie Peng.

Finally, I would like to express my deepest appreciation to my family: my parents, my grandparents, my uncles and aunts. I could not even imagine I can finish my study without their help and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Research Motivation

The rapid growth in mobile computing, mobile multimedia services and other mobile applications make high-speed wireless communication be one of the fastest developing technologies in recent years. Multiple-input multiple-output (MIMO) communication technology has been researched as it satisfies the demand for both the increased capacity and improved link-reliability [7]. Due to these advantages, MIMO communication technology is adopted as a part of many current wireless communication standards, such as IEEE 802.11n, and some future technologies such as WiMAX based on IEEE 802.16e and IEEE 802.20. Meanwhile, some recent extensions of the MIMO technology, such as multi-user MIMO (MUMIMO) [21], which enlarge the communication capabilities of each individual user, have also attracted some interests from researchers.

In MIMO communication system, the most important part is MIMO detector [10], the function of which is to separate the spatially multiplexed signals received from the multi antennas [4]. This process should consider the effect of unstable fading channels, the additional environment and system noise, and the interferences among those antennas, while keep an acceptable resource usage of both hardware and time. Therefore, the research of MIMO detection technique is very attractive and useful. New detection algorithms and implementations are needed to improve the whole system's performance, including data

throughput and spectrum efficiency, while minimize the resource usage.

1.2 Introduction to MIMO wireless communication Systems

1.2.1 MIMO System Model

Consider a symbol synchronized MIMO system with M_T transmitter and M_R ($M_R \geq M_T$) receiver antennas demonstrated in Figure 1.1. The total input data, in binary, is divided into blocks and for each block it contains $M_T \times \Omega$ bits $\mathbf{x} = (\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_{M_T})^T$ where $\mathrm{x}_i = (x_{i1}, x_{i2}, \ldots, x_{i\Omega})$, as a part of the total sequence, and then each block is mapped to a M_T-dimensional transmit symbol vector $\widetilde{\mathbf{s}} = (\widetilde{s}_1, \widetilde{s}_2, \ldots \widetilde{s}_{M_T})^{\mathrm{T}}$, in which $\widetilde{s}_i$ is the transmitted symbol of the i-th antenna and it represents Ω bits of binary data. Those symbols are chosen from $\mathcal{O}$ which stands for the complex scalar constellation 2^{Ω}-QAM system and $|\mathcal{O}| = 2^{\Omega}$. According to the definition above, the baseband input-output relation for a MIMO system can be written as

$$\widetilde{\mathbf{y}} = \widetilde{\mathbf{H}}\widetilde{\mathbf{s}} + \widetilde{\mathbf{n}}, \tag{1.1}$$

$$\widetilde{\mathbf{H}} = \overbrace{\begin{bmatrix} \widetilde{h}_{11} & \widetilde{h}_{12} & \cdots & \widetilde{h}_{1M_T} \\ \widetilde{h}_{21} & \widetilde{h}_{22} & & \\ \vdots & & \ddots & \\ \widetilde{h}_{M_R1} & & & \widetilde{h}_{M_RM_T} \end{bmatrix}}^{M_T} \Big\updownarrow M_R \tag{1.2}$$

where $\mathbf{H}$ denotes the $M_R \times M_T$ channel matrix, expressed in Equation (1.2), which is assumed to be perfectly known by the receiver. In this paper it is assumed to be a Rayleigh fading channel so every element $\widetilde{h}_{ij} \sim \mathcal{CN}(0, 1)$ with independent identical distributed (i.i.d) complex zero-mean Gaussian variables with the normal variance, represents the complex transfer function from the j-th transmit antenna to the i-th receive antenna. The $\mathbf{n} = (n_1, n_2, \ldots, n_{M_R})$ is the vector of i.i.d additive white Gaussian noise (AWGN) samples and $n_i \sim \mathcal{CN}(0, \sigma^2)$.

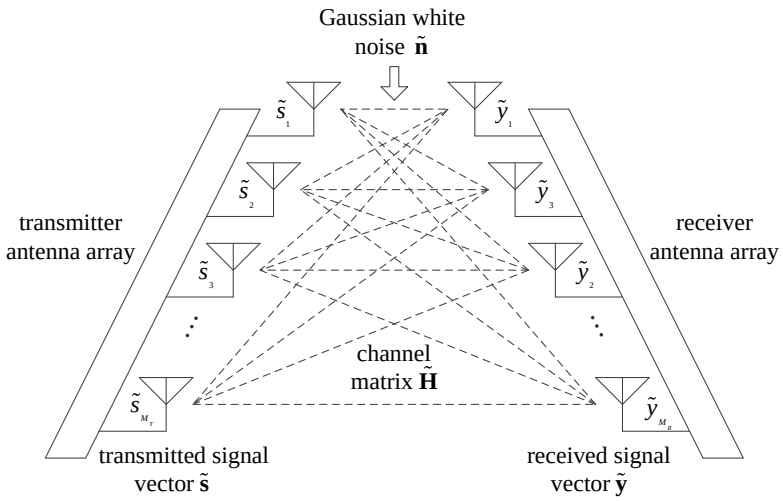


Figure 1.1: MIMO system model

Although the system is discribed in the complex domain, to simplify the presentation, it can be equivalently transformed into real domain. The complex domain Equation (1.1) is equivalent to

$$\begin{bmatrix} \Re(\widetilde{\mathbf{y}}) \\ \Im(\widetilde{\mathbf{y}}) \end{bmatrix} = \begin{bmatrix} \Re\left(\widetilde{\mathbf{H}}\right) & -\Im\left(\widetilde{\mathbf{H}}\right) \\ \Im\left(\widetilde{\mathbf{H}}\right) & \Re\left(\widetilde{\mathbf{H}}\right) \end{bmatrix} \begin{bmatrix} \Re(\widetilde{\mathbf{s}}) \\ \Im(\widetilde{\mathbf{s}}) \end{bmatrix} + \begin{bmatrix} \Re(\widetilde{\mathbf{n}}) \\ \Im(\widetilde{\mathbf{n}}) \end{bmatrix}, \tag{1.3}$$

and Equation (1.3) can be expressed as

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n}, \tag{1.4}$$

in which $\mathbf{s} = (s_1,\, s_2,\, \ldots,\, s_{2M_T})$ is chosen from the set of the real entries in the constellation Λ with $|\Lambda| = 2^{\frac{\Omega}{2}}$, e.g., $\Lambda = \{-3, -1, 1, 3\}$ in the case of 16-QAM. Notice that after the complex to real tranformation the number of entries of the input vector $\mathbf{y}$ is $2M_T$ now. For each real symbol it represents $\Omega/2$ bits of binary data. The discussions in the rest part of this thesis are all based on the term after the complex to real tranformationation except some specific examples.

Besides the MIMO transmitter and receiver atennas shown in Figure 1.1, an outer encoder/decoder can also be included in MIMO communication system. First, the transmitted

binary bits are encoded by a channel encoder, which in this paper is a convolutional encoder of rate $R = 1/2$ and interleaved. An $M_T\Omega \times 1$ dimensional binary vector of coded bits $\mathrm{x} = (x_1, x_2, \ldots, x_{M_T})^T$, in which $x_i = (x_{i1}, x_{i2}, \ldots, x_{i\Omega})$ is obtained from the whole coded sequence. The binary digit x_{ij} is assumed to have an independent value from $\{1, 0\}$. Then this sequence is mapped into a $M_T \times 1$-dimensional complex vector $\widetilde{s}$ as mentioned at the beginning of this section. Combined with the outer encoder/decoder, the performance of the MIMO system can be increased considerably. Details of the combined decoder detection will be introduced in Chapter 4.

1.2.2 MIMO Detection Methods

For a MIMO communication system the objective of the detector is to get the original information of the mapped symbols from the transmitter side. However, due to the property of the Rayleigh fading channel, getting the maximum likelihood (ML) solution is the best result that a detector can achieve. But an advantage is, by presenting training phase, inserting pilot signal and applying channel estimation, the channel matrix $\mathbf{H}$ can be assumed to be perfectly known by the receiver. Gaining the channel matrix information and the received parallel signals, algorithms created to separate the original transmitted symbols through the MIMO detecting theory can be mainly divided into the following parts.

- (1) *Zero-Forcing (ZF) Algorithm.* The ZF algorithm is a natural thought directly gained from the information of channel matrix and received symbols. It finds the pseudo inverse of channel matrix and then multiplies it by received signal symbol vector. The results are round to the constellation system without any other disposing. This algorithm just simply ignores the affection of noise and inference between each antenna, so it is an suboptimal algorithm which is efficient when the SNR is high, otherwise the performance is rather poor.

- (2) *ML Solution with Exhaustive Search.* The ML solution satisfies the function

$$\mathbf{s}_{ML} = \arg \min_{\mathbf{s} \in *^{2M_T}} \|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2 \tag{1.5}$$

  and recall in Equation (1.4) the complex system has been transformed to real domain,

so the solution $\mathbf{s}$ is with $2M_T$ dimensions instead of M_T in Equation (1.1). ML solution is the optimum solution with for uncoded system. A straightforward approach to solve Equation (1.5) is searching all the possible constellation points in the $2M_T$ dimension space. However the computational complexity for exhaustive search increases exponentially with the number of the transmitter and receiver antennas as well as the modulation size. In a 16-QAM modulation 4×4 MIMO communication system the number of possible candidate symbols is 65536. Performing exhaustive search consums a lot of resources both in time and hardware.

- (3) *Depth First Sphere Detection [3, 11].* Sphere Detection algorithm is a reduced complexity method that can lead to the ML solution for MIMO system, which avoid exponentially increased detection complexity in exhaustive search. The major idea that makes sphere detection efficient is that the number of constellation points are found inside a hypersphere area in the $2M_T$ dimension constellation space, which is dramatically smaller than the exhaustive search area. The lattice in the hypersphere can be mapped to a trellis tree and the depth first search strategy searches the tree from root to the top. When reaches the bounder it returns backward to find another possible candidate then searches forward again, until all the possible candidates are searched in the hypersphere. Although compared with the exhaustive search its computational complexity is decreased dramatically, it is still complex since the search goes both forwards and backwards, which makes the time of whole detection process uncertain. For real communication system it means that the throughput is unstable.

- (4) *Breadth First Sphere Detection [13, 8].* Although it is also called sphere detection, the solution of this method is only a nearly-ML solution. It also searches the possible candidates in the range of a hypersphere with a certain radius, which is a subset of the solution space. But unlike the depth first strategy it goes one-way, in the forward direction only. The breadth first detection searches all the possible candidates in one dimension but only keeps a certain number of them to be extended in next dimension. The same process is repeated until the final dimension is reached. The fixed of searching structure makes the implementation in a parallel and pipeline fashion

with a fixed throughput. More details will be introduced in Chapter 2.

1.3 Summary of Contribution

This thesis presents the following research results of MIMO communication systems.

- (1) *Modified MIMO Detection Algorithm.* In this thesis a modified MIMO detection algorithm is introduced based on the breadth first sphere detection. With a preprocess stage the BER performance is improved with limited additional resource usage.

- (2) *Group Sorting Combined with Parallel Sorting Network.* As the bottleneck of breadth first algorithm, the sorting between candidates often limits the decoding throughput. A group sorting algorithm is proposed in Chapter 3. Combined with the group sorting, a sorting network is also designed to make the sorting process executed in a highly parallel structure.

- (3) *LLR Soft Information Generator.* Using the foward error correction (FEC) technology, the BER performance of the MIMO system can be improved with combined detection and decoding. An LLR generator is cascaded to the MIMO detector and to generate the soft information of each received bit.

- (4) *Parallel and Pipelined Hardware Implementation.* A highly parallel and fully pipeline structure is designed for the proposed MIMO detector. Both soft information and hard information can be generated at a data throughput of 3.7 Gbps.

1.4 Thesis Organization

The rest of this thesis is organized as follows: Chapter 2 briefly introduced K-BEST MIMO detection and its modification with preprocess which improves the performance in an uncoded MIMO detection. In Chapter 3, the main bottleneck of the K-BEST MIMO detection algorithm, sorting, is replaced by an efficient parallel group sorting algorithm combined with a sorting network. Although there is a minor decrease in its BER performance, the sorting structure is simplified through this architecture and it also modified the

system delay in the sorting process. Chapter 4 presents a MIMO detection which generates a candidate list for soft-output information according to the theory of soft-output information and LLR value calculation. Compared with the uncoded ML solution, the performance of MIMO system with an outer decoder is significantly better. In Chapter 5 the hardware architecture and implementation results are presented. With a fully parallel and pipelined design this detector can work at a frequency of 230.132 MHz in Virtex6 FPGA with a throughput of 3.7 Gbps

Chapter 2

Sphere Detection Algorithms and Preprocessing

MIMO communication systems have attracted the researchers attention because of their high data throughput, increased channel capacity and improved link reliability. As the kernel of the MIMO system, the MIMO detection process affects the performance of the entire MIMO system. Although the exhaustive search yields the ML solution, which is the upper bound of the BER performance in an uncoded MIMO communication system, it is still infeasible because of the computational complexity. Consider a 16-QAM modulation 4×4 MIMO communication system, the number of possible candidate symbols is 65536 and the computational complexity increases exponentially with the growth of the number of antennas and constellation points in the modulation.

Severalof algorithms have been developed to gain the ML or nearly-ML solution for the detection process, like ZF algorithm and sphere detection. Sphere detection is considered to be an efficient way to decrease the detection complexity while keeping an acceptable system performance. The details of lattice theory can be found in [1]. Depth first search and breadth first search are the two most important search strategies which are adopted in sphere detection [5, 4, 11, 8]. This chapter provides an introduction to the proposed modified K-BEST MIMO detection algorithm. As an improvement to the traditional breadth first sphere detection, the proposed modified K-BEST algorithm makes the best of the parallel

characteristic of the breadth first search architecture, while decrease the possibility of early pruning the ML solution. Unless being declared specifically, all discussions and simulations in this thesis are all performed on the 16-QAM modulation 4×4 MIMO communication system model. MIMO systems with more sending-receiving antennas and more complex modulation process have the similar detection process.

2.1 Sphere Detection Algorithms

The MIMO detection process can be mapped to a trail search process through a hypercube space. Consider the MIMO system discribed by Equation (1.4) which has been transformed into real domain. Since the number of receiving and transmitting antennas are both M_T, after complex to real tranformation carried by Equation (1.3) the solution space equals to a hypercube with $2M_T$ dimensions. In each dimension, the possible result is picked up through Λ. So the total number of possible results combination through all the $2M_T$ dimensions is $|\Lambda|^{2M_T}$. The sphere detector is a detection process that finds the points satisfying the Equation (1.5). Compared with the exhaustive search, the space that the sphere detection goes through is a hypersphere, which is a subset of the original hypercube. The Finke-Phost and Schnorr-Euchner's research [16, 6] indicates that defining a radius through ZF is a highly efficient way to decrease searching complexity of this detection.

$$d^2 = \left\| \lceil \mathbf{H}^{-1}\mathbf{y} \rfloor - \mathbf{H}^{-1}\mathbf{y} \right\|^2 \tag{2.1}$$

In Equation (2.1) the $\lceil . \rfloor$ denotes the operation of rounding the data in each dimension to the nearest real constellation points in Λ.

2.1.1 Introduction to Sphere Detection

In Equation (1.3) each symbol s_i in dimension i of the vector $\mathbf{s}$ is referred to a layer, and is constrained to a finite set Λ. Enumeration of the ML point that satisfies the Equation (1.5) is presented in [1] as an algorithm for finding vectors with smallest norm to the original points in a hyperspace which is composed by integer lattice. Fincke-Pohst (FP) [6] and Schnorr-Euchner (SE) [16] algorithms are two computationally efficient means of

implementing the detection process and they are also the foundation of most of the existing search algorithms.

Including FP and SE, nearly all the known sphere detection algorithms are based on the QR decomposition of the channel matrix, the matrix $\mathbf{H}$ with linearly independent columns.

$$\widetilde{\mathbf{H}} = \widetilde{\mathbf{Q}} \begin{bmatrix} \widetilde{\mathbf{R}} \\ 0 \end{bmatrix} \tag{2.2}$$

where $\widetilde{\mathbf{Q}}$ is $M_R \times M_R$ and unitary matrix, $\widetilde{\mathbf{R}}$ is $M_T \times M_T$ upper triangular and invertible matrix, and 0 is an $(M_R - M_T) \times M_T$ matrix with all elements 0.

Since Equation (1.5) is invariant with orthogonal transformation, the equivalent equation can be expressed as

$$\tilde{\mathbf{s}}_{ML} = \arg \min_{\tilde{\mathbf{s}} \in \mathcal{O}^{M_T}} \left\| \widetilde{\mathbf{Q}}^{\mathrm{T}} \widetilde{\mathbf{y}} - \begin{bmatrix} \widetilde{\mathbf{R}} \\ 0 \end{bmatrix} \widetilde{\mathbf{s}} \right\|^2 \tag{2.3}$$

If $M_T < M_R$, A root distance d_{root} is separated from the Equation (2.3). and

$$\tilde{\mathbf{s}}_{ML} = \arg \min_{\tilde{\mathbf{s}} \in \mathcal{O}^{M_T}} \left(\left\| \left[\widetilde{\mathbf{Q}}^{\mathrm{T}} \widetilde{\mathbf{y}} \right]_{\overline{M_T+1}, \overline{M_T+2}, \ldots, \overline{M}_R} - \widetilde{\mathbf{R}} \widetilde{\mathbf{s}} \right\|^2 + d_{root}^2 \right) \tag{2.4}$$

in which the $[.]_{\overline{M_T+1}, \overline{M_T+2}, \ldots, \overline{M}_R}$ denotes to delete the last $M_R - M_T$ elements from the original vector so that the rest part of the vector $\left[\widetilde{\mathbf{Q}}^{\mathrm{T}} \widetilde{\mathbf{y}} \right]$ has a length of M_R and d_{root}^2 is the squared Euclidean distance of the deleted part.

Although in the application the receiver antenna array is usually larger than the transmitter antenna array, in this paper the hypothesis $M_T = M_R$ is set for simplicity. After the complex to real tranformation performed by Equation (1.3), a new channel matrix $\mathbf{H}$ is gained and QR decomposition is performed on it. Equation (2.3) is expressed as

$$\mathbf{s}_{ML} = \arg \min_{\mathbf{s} \in *^{2M_T}} \|\mathbf{Q}\mathbf{y} - \mathbf{R}\mathbf{s}\|^2 \tag{2.5}$$

$$= \arg \min_{\mathbf{s} \in *^{2M_T}} \|\overline{\mathbf{y}} - \mathbf{R}\mathbf{s}\|^2 \tag{2.6}$$

where $\mathbf{Q}$ is $2M_T \times 2M_T$ and orthogonal matrix while $\mathbf{R}$ is $2M_T \times 2M_T$ upper triangular and invertible. $\overline{\mathbf{y}} = \mathbf{Q}^{\mathrm{T}}\mathbf{y}$ denotes the $2M_T$ orthogonally transformed symbols.

Notice in Equation (2.6) R is an upper triangular matrix. This structure enables the sphere detector to decompose Equation (2.6) recursively as

$$\begin{aligned}
\|\overline{\mathbf{y}} - \mathbf{R}\mathbf{s}\|^2 &= d^2(\overline{y}_{2M_T}, r_{2M_T,2M_T} s_{2M_T}) \\
&\quad + \left\| (\overline{\mathbf{y}} - \mathrm{r}_{2M_T} s_{2M_T})_{\overline{2M_T}} - \mathbf{R}_{\overline{2M_T},\overline{2M_T}} \mathbf{s}_{\overline{2M_T}} \right\|^2 \qquad (2.7) \\
&= d^2(\overline{y}_{2M_T}, r_{2M_T,2M_T} s_{2M_T}) \\
&\quad + \left\| \overline{\mathbf{y}}(s_{2M_T}) - \mathbf{R}_{\overline{2M_T},\overline{2M_T}} \mathbf{s} \right\|^2 \qquad (2.8) \\
&= \sum_{D=2M_T}^{1} d^2(\overline{y}(s_{D+1})_D, r_{D,D} s_D) \qquad (2.9)
\end{aligned}$$

where $d^2(.)$ denotes the squared Euclidean distance; $(.)_{\overline{m}}$ and $(.)_{\overline{m,n}}$ denotes to delete the m-th element from the original vector and deleting the m-th row and n-th column from the original matrix, respectively; $r_{m,m}$ denotes the element in the m-th row and m-th column of matrix $\mathbf{R}$ and $(.)_m$ denotes the m-th column of a matrix or the m-th element of a vector:

$$d^2(\overline{y}(s_{D+1})_D, r_{D,D} s_D) = |\overline{y}(s_{D+1})_D - r_{D,D} s_D|^2 \qquad (2.10)$$

$$\overline{\mathbf{y}}(s_{D+1}) = \begin{cases} \overline{y} & D = 2M_T \\ (\overline{\mathbf{y}}(s_{D+2}) - \mathrm{r}_{D+1} s_{D+1})_{\overline{D+1}} & D = 2M_T - 1, \ldots, 1 \end{cases} \qquad (2.11)$$

The processes described by Equation (2.7) , Equation (2.8)and Equation (2.9) lead detecting ML points process to a trail tree search with multiple stages. In Figure 2.1 a trail tree is built and fully expanded. For simplicity this system only have two receiver and transmitter antennas with BPSK modulation. That means the complex to real tranformation is not needed while the parameters are set as $M_T = M_R = 2$ and $\Omega = \{-1, 1\}$. In a trail tree structure nodes in the upper layer is called parent nodes for nodes in the lower layer, and nodes in the lower layer is called child nodes which are expanded from the same parent node. For example in Figure 2.1 the node b_1 is the parent node of the nodes b_3 and b_4; the node b_3 and $b_4$are the child nodes expanded from node b_1. Nodes in the final layer are called the leaf nodes. In Figure 2.1 node b_3, b_4, b_5 and b_6 are all leaf nodes.

Although the way of computing the Euclidean distance in different sphere detections are almost the same as expressed in Equation (2.9), the strategies of searching the smallest final Euclidean distance are varied dramatically and it decides the final computational complexity and performance.

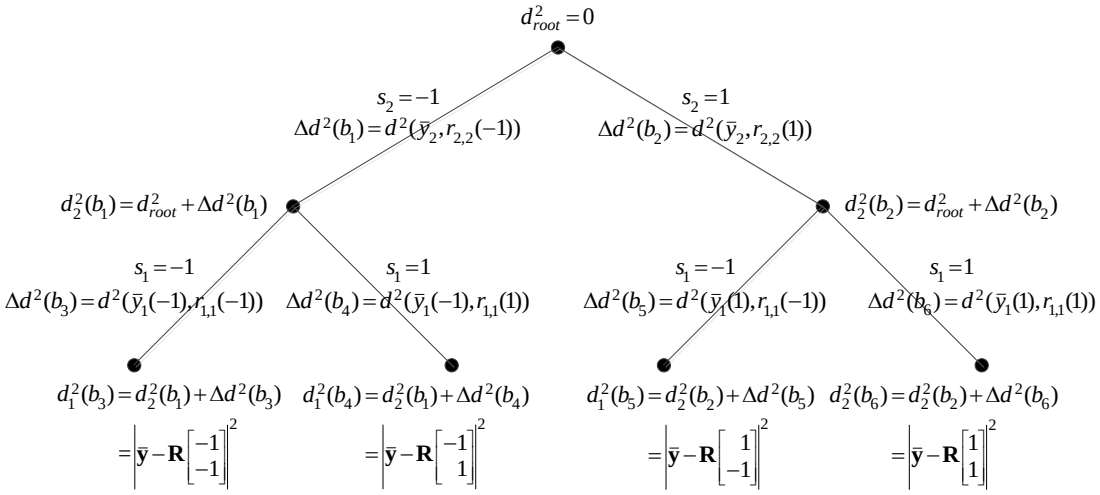


Figure 2.1: Fully expanded searching trail tree of a 2×2 MIMO system with BPSK modulation

2.1.2 Depth First Search: Precision

Consider Equation (2.7) , Equation (2.8)and Equation (2.9) which divide the hypersphere space into $2M_T$ layers. The depth first search strategy searches each layer for the smallest final Euclidean distance in both forward and backward directions through the $2M_T$ layers.

The currently smallest final Euclidean distance is the radius of the hypersphere, c, which can be set as ∞. In the $2M_T$-th layer the detector finds and stores the first point $s_{2M_T} \in \mathcal{O}$ that satisfies

$$d^2(\overline{y}_{2M_T}, r_{2M_T,2M_T} s_{2M_T}) \leq c^2 \tag{2.12}$$

where $d^2(\overline{y}_{2M_T}, r_{2M_T,2M_T} s_{2M_T}) = d^2_{2M_T}$ is the partial squared Euclidean distance in the $2M_T$-th layer. Then the first point in the $(2M_T - 1)$-th layer is stored with its partial squared Euclidean distance in the $(2M_T - 1)$-th layer, which is

$$d^2_{2M_T-1} = d^2(\overline{y}(s_{2M_T})_{2M_T-1},\, r_{2M_T-1,2M_T-1} s_{2M_T-1}) + d^2_{2M_T} \tag{2.13}$$

The same process is repeated until the 1st layer is reached. If during this process the partial squared Euclidean distance of the m-th layer is larger than c^2, the detector first check the

neighbor point of the stored node in that layer. If no node's squared partial Euclidean distance is less than c^2, then the detector goes back to the $(m+1)$-th layer, finds the next point of that layer instead of the stored one the compare and search process is executed continously.

If the 1st layer is reached and the total Euclidean distance is smaller than the the existing c, the radius will be replaced by the newly gained Euclidean distance and the detector goes to the neighbor node of the one sorted in the 1st layer, until all the nodes in the 1st layer is searched and find a smallest Euclidean distance. Then the detector goes back to the 2nd layer, picking up the next node from the initially stored one in that layer. If this partial Euclidean distance is smaller than the new radius c it will goes to the 1st layer, executing the enumeration and comparing process so that the new total Euclidean distance in the 1st layer is gained again.

Every time the search process reaches the 1st layer with a total Euclidean distance smaller than current c, the c should be replaced. If none of the total Euclidean distance in the 1st layer is smaller than c, it will goes upper to the 2nd layer and picks up the next neighbor point from the one picked up in the last search. If every possible point is examined but the total Euclidean distance in the 1st layer is still larger than radius c then the detector will goes backward to the 3rd layer and repeats the enumeration and search again. If the detector finally goes back to the $2M_T$-th layer but none nodes can satisfy Equation (2.12) then the last stored node in each layer are combined together as the result with ML solution that satisfies Equation (1.5). Figure 2.2 shows the initial stages of this searching.

In Equation (2.8) the $\overline{y}(s_{D+1})_D$ depends on the chosen nodes of upper layer, and this is the reason that every time the node in the upper layer is changed, the partial Euclidean distance of the child nodes in the lower layers has to be changed and checked with it too.

The depth first search strategy can be summarized as follows:

A modification of this algorithm is introduced by Schnorr-Euchner's research [16]. In depth first search strategy the SE algorithm can decrease the number of examined nodes and many other researchers extend this algorithm for their implementations. The SE search strategy predefine a radius according to the Babai points gained through the ZF algorithm and all the enumeration process is around the Babai points in each layer. Also, since the

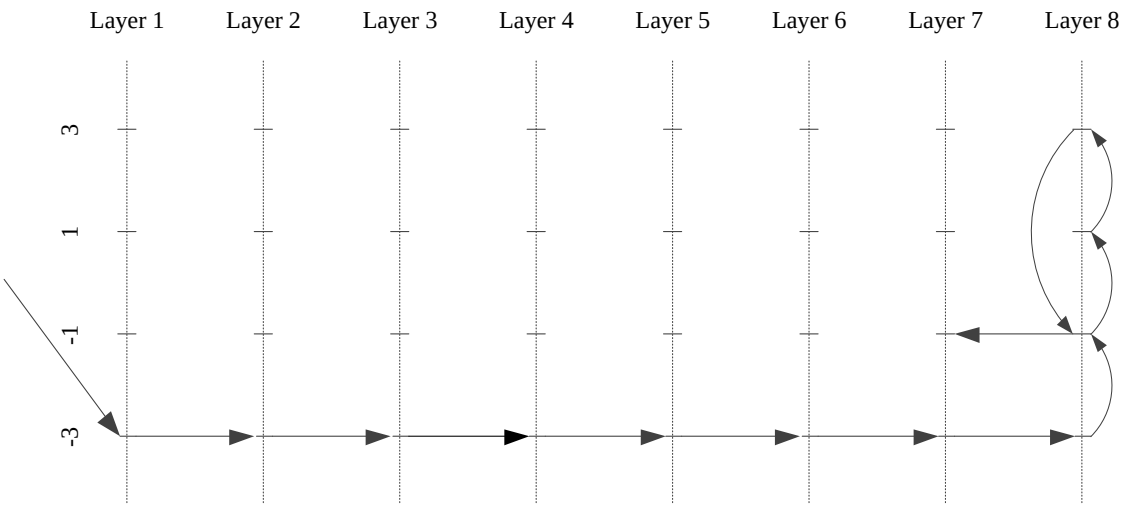


Figure 2.2: Depth firsts searching Strategy

Algorithm 2.1 Depth first search strategy

Input: $\overline{\mathbf{y}}$, $\mathbf{R}$, constellation information.

1. Set the radius of the hypersphere $c^2 = \infty$ and the initial squared partial Euclidean distance $d_{m+1}^2 = 0$;

2. Set $k = 2M_T$, $\overline{\mathbf{y}}_k = \overline{\mathbf{y}}$, $\overline{y}_k$ is set to the k-Th element of vector $\overline{\mathbf{y}}_k$;

3. Set bounds LB and UB and symbol increasing *step* according to the constellation. Set $s_k = LB - step$;

4. $s_k = s_k + step$. If $s_k \leq UB$ goto 6. Else go to 5;

5. $k = k + 1$. If $k = m + 1$ terminate the detection. Else goto 4;

6. Calculate $d_k^2 = d^2(\overline{y}_k, r_{k,k}s_k) + d_{k+1}^2$ and $\overline{\mathbf{y}}_{k-1} = (\overline{\mathbf{y}}_k - \mathrm{r}_k s_k)_{\overline{k}}$. If $d_k^2 < c^2$ goto 7. Else goto 4;

7. if $k = 1$then $c = d_{k+1}$, and goto 4. Else $k = k - 1$, and goto 3.

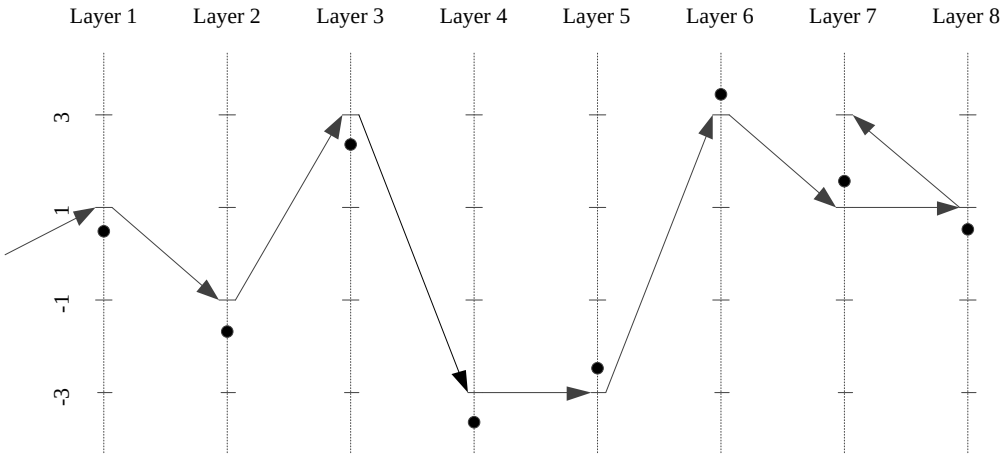


Figure 2.3: Depth first searching strategy with SE Enumeration

partial Euclidean distance is decided by the parent nodes in the upper layers, once the detection changes the point in the upper layer a new Babai point and partial Euclidean distance in the lower layer has to be enumerated. This method can also be adopted in the breadth first search for its advantage in sorting candidates in each layer. Details of this will be introduced in Section 3.2.1.

2.1.3 Breadth First Search: Speed and Throughput

As the depth first searching strategy, the breadth first search is also based on Equation (2.7), Equation (2.8) and Equation (2.9). But the breadth first searching strategy searches the best nodes in the forward direction only. In each layer the detector examines all the possible nodes, and K candidates with smallest partial Euclidean distances are kept. Then through Equation (2.7) and Equation (2.8) the possible points in the lower layer based on the parent candidates picked up are extended. According to their partial Euclidean distance, a group of K points are picked up again. Following this process the detector goes lower and lower until the 1st layer is reached. Figure 2.4 shows breadth first search in the first 3 layers of a 4×4 MIMO communication system with 16-QAM modulation. As in the first and second layer the number of candidates is less than $K = 16$, all the nodes in these

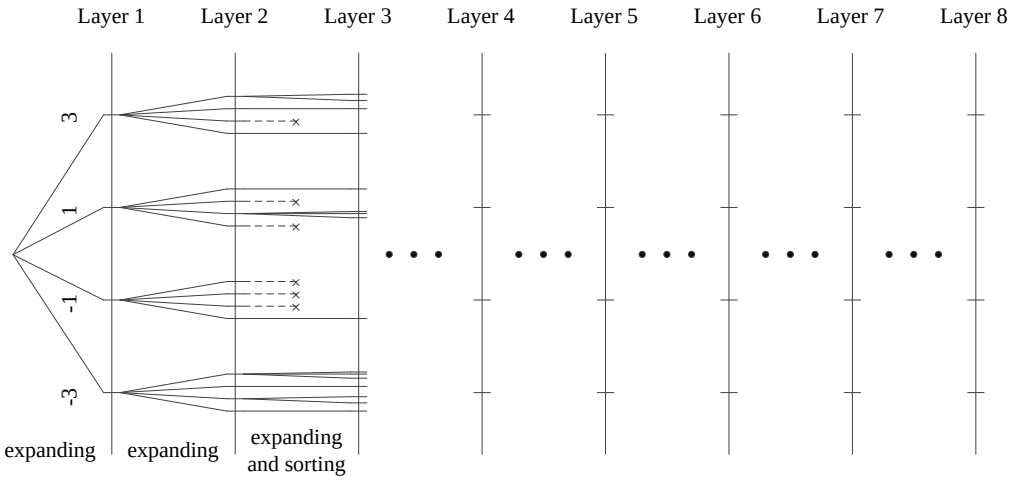


Figure 2.4: Breadth first searching strategy scheme

two layers are expanded and kept without sorting. In the third layer the number of the child nodes is $4 \times K = 64$, so after expanding and sorting only $K = 16$ child nodes are kept for next layer. This expanding and sorting stage is repeated in the rest layers. The process of breadth first strategy is outlined as below.

After picking up the nodes in 1st layer with the smallest total Euclidean distance, the detection process is ended and the final result is combined by the group of nodes with the smallest total Euclidean distance. Since in each layer the number of the selected nodes is fixed, the mainly advantage of the breadth first search based on this property is that it has a fixed data throughput. Unlike the depth first search strategy usually there does not exit a specific radius for the hypersphere. Although adding such a radius can decrease the number of points that the detector goes through, it also makes the number of examined points uncertain, which has a negative affection in the stability of data throughput. However to reach ML solution that satisfies the Equation (1.5) the breadth first search should keep the K as large as possible to keep the optimality. But this method will lead to an exhaustive search which will take a dramatically large sources both in time and hardware usage.

Although breadth first search is a sub-optimal detection algorithm that only achieve the nearly-ML solution, according to the research in [19, 13], it can also be expected to achieve

Algorithm 2.2 Breadth first searching strategy

Input: $\overline{\mathbf{y}}$, $\mathbf{R}$, constellation information.

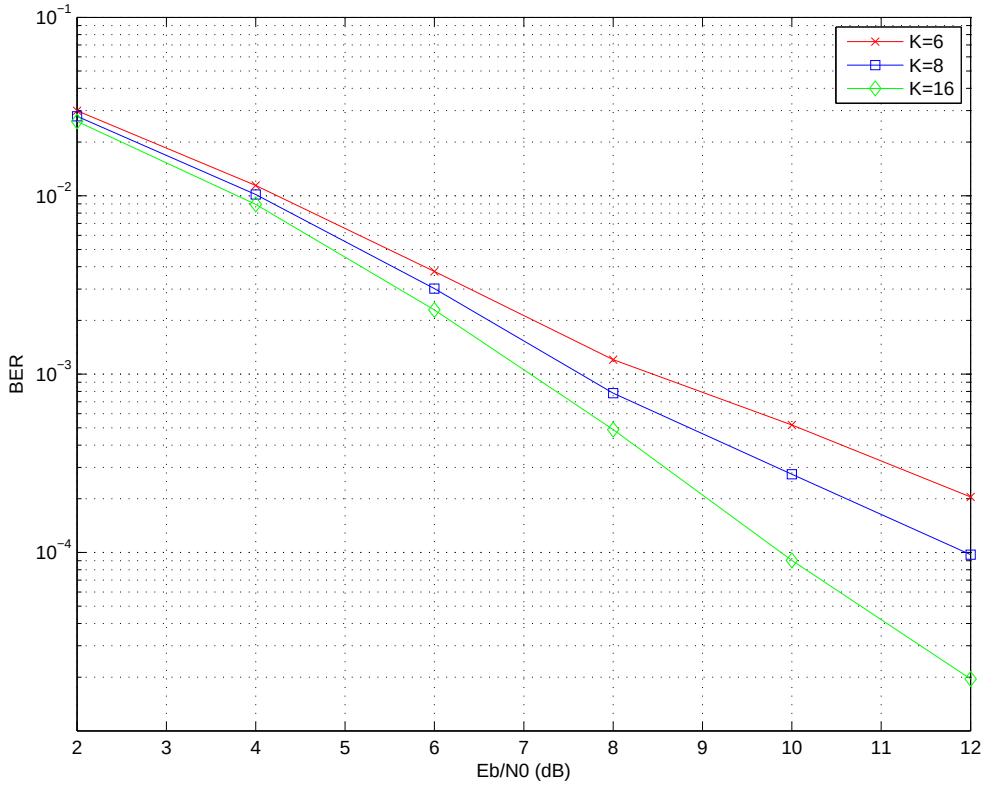
1. Set the initial root distance $d_{m+1}^2 = d_{root}^2$;

2. Set $k = 2M_T$, $\overline{\mathbf{y}}_k = \overline{\mathbf{y}}$, $\overline{y}_k$ is set to the k-th element of vector $\overline{\mathbf{y}}_k$;

3. Expand all the possible child nodes s_k. Calculate the corresponding $d_k^2 = d^2(\overline{y}_k, r_{k,k}s_k) + d_{k+1}^2$ and $\overline{\mathbf{y}}_{k-1} = (\overline{\mathbf{y}}_k - \mathrm{r}_k s_k)_{\overline{k}}$;

4. If the number of the expanded candidate $n(k) > K$, sort the candidates by their partial Euclidean distances.

5. Choose the K child nodes with smallest d_k. If $k = 1$ terminate the detection else $k = k-1$ and goto 3.



points

Figure 2.5: Simulation results: BER performance when K=6, 8, 12

a bit-error rate (BER) performance closing to the ML algorithm with a sufficient large K. This algorithm is also named as K-BEST algorithm by [8]. Compared with exhaustive search a proper value of K can reduce the complexity while keep an acceptable BER performance. Figure 2.5 shows how the value of K affects the performance of the MIMO system.

Besides the fixed data throughput, another major advantage of the breadth first detection is the easily implemented parallel architecture. Since the breadth first search strategy keeps a fixed number of nodes in each layer, the computing process of Equation (2.7) of each node can be executed at the same time. The only bottleneck, however, is the comparing and sorting part. The details will be introduced in Chapter 3.

2.2 Preprocessing with Layer Reordering

It is known that the computational complexity of depth first sphere detection is highly sensitive to the ordering of the columns of the channel matrix[18], a similar situation also happens in the breadth first sphere detection[14, 2]. Although the structure of the breadth first detection decides that reordering the column of channel matrix could not help decrease its computational complexity, it increases the BER performance of the breadth first sphere detector. Assume there are two candidate symbols $\mathbf{s}_a$ and $\mathbf{s}_b$; both are $2M_T$-dimension vectors. For the MIMO model described in Equation (1.4) the total squared Euclidean distances are $d^2(\mathbf{s}_a) = \sum_{D=2M_T}^{1} d^2(\overline{y}(s_{a\,D+1})_D, r_{D,D}s_{a\,D})$ and $d^2(\mathbf{s}_b) = \sum_{D=2M_T}^{1} d^2(\overline{y}(s_{b\,D+1})_D, r_{D,D}s_{b\,D})$. If $\mathbf{s}_a$ is the ML solution then $d^2(\mathbf{s}_a) < d^2(\mathbf{s}_b)$ should be satisfied. However, in the breadth first searching, the decision is made based on the squared partial Euclidean distance, which are ==

$$d^2_{\mathbf{s}_a}(k) = \sum_{D=k}^{1} d^2(\overline{y}(s_{a\,D+1})_D, r_{D,D}s_{a\,D}),\ k = 2M_T, \ldots, 1 \tag{2.14}$$

and

$$d_{\mathbf{s}_b}(k) = \sum_{D=k}^{1} d^2(\overline{y}(s_{b\,D+1})_D, r_{D,D}s_{b\,D}),\ k = 2M_T, \ldots, 1 \tag{2.15}$$

respectively. If in some un-leaf stage, which means $1 < k \leq 2M_T$, the squared partial Euclidean distance of $\mathbf{s}_a$ is smaller than that of $\mathbf{s}_b$ and does not belong to the K best candidates, the decision process will discard the $\mathbf{s}_a$ from then on. This explains that even

the candidates with the best partial Euclidean distance in each layer is selected it still does not necessarily mean that the ML solution will be in the candidates of final layer. This is the source of discarded BER performance of breadth first search strategy compared with the depth first strategy and that is also the reason that the breadth first search strategy is called sub-optimal detection.

Although keeping the K with a sufficient large value can improve the performance, apparently this method also makes the breadth first detection more complex, especially consider that the sphere detection algorithm is involved in a lot of sorting process. Keeping the K in a reasonable limit while increasing the BER performance as much as possible is important for the MIMO detection. Since the sphere detection algorithms, no matter depth first or breadth first search strategies, are all based on the QR decomposition performed on the channel matrix, a proper preprocess performed on the channel matrix will help to increase the system performance either in decrease its complexity or increase the BER performance. According to the analysis above to increase the BER performance of the breadth first search strategy with a small K, it is necessary to decrease the possibility of discarding the ML in the search process. One of the widely used approach is reorder the searching layer, which is permuting the channel matrix $\mathbf{H}$. Then the order of the elements in $\mathbf{s}$ is changed respectively, while keeping the total partial Euclidean distance the same.

If the reordering scheme can enlarge the differences between those partial Euclidean distances of different $\mathbf{s}$ at early levels, which means that the partial Euclidean distance increases in a decreasing way, then this scheme can be claimed that it would help to increase the performance. Notice in Equation (2.14) and Equation (2.15) if the differences of partial Euclidean distances between each layer decreases, it would make the candidates with larger partial Euclidean distance in the early stage much less possible to be the ML solution after accumulating all these differences in the final layer because the partial Euclidean distances of the candidates with smaller partial Euclidean distances at early layer increas=e slower than those of the candidates with larger partial Euclidean distances at early layer. Thus the performance of the breadth first search strategy will be improved.

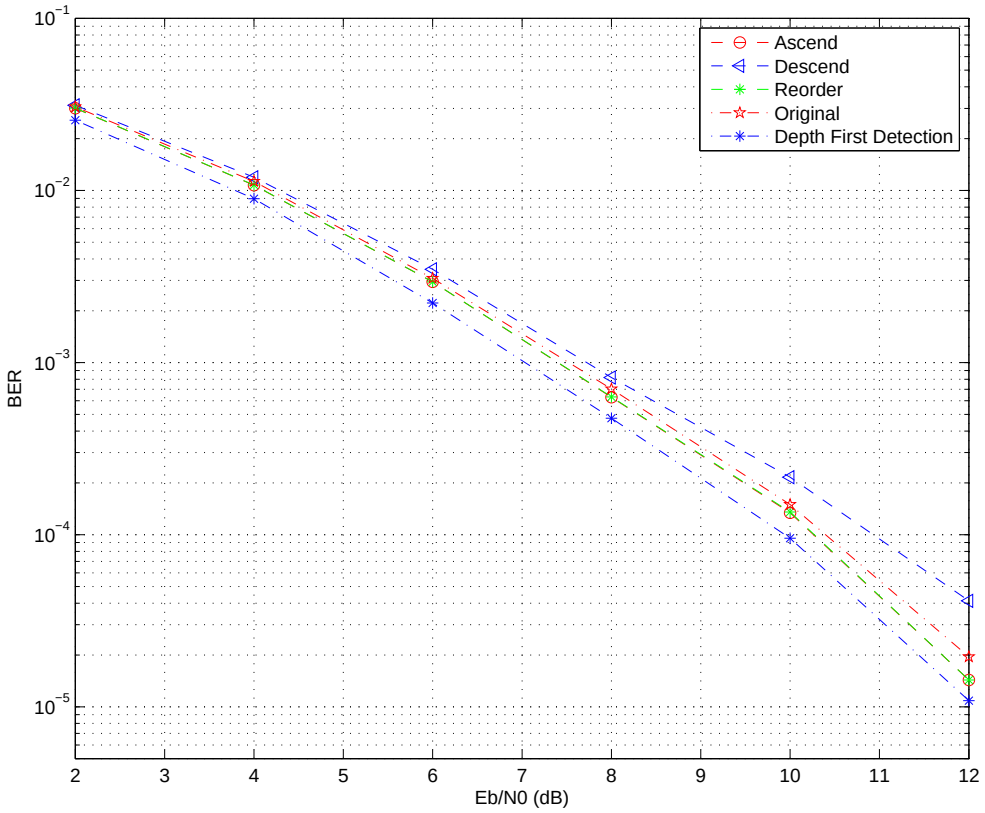
However, since the squared partial Euclidean distance of each candidate is determined by Equation (2.14) and Equation (2.15) once every new level is reached by the detection

process, both the element of the candidate s_D itself and the $r_{D,D}$ in that level will affect the partial Euclidean distance. Hence if reorder the layer the $r_{D,D}$ with larger value used in early layers, which means the matrix $\mathbf{R}$ is reordered to make the $r_{D,D}$ in a increasing way, the difference of the partial Euclidean distances between each layer would have a large possibility to be decreased.

To achieve the purpose of reordering the $r_{D,D}$ in a increased order with D increasing, many different preprocess algorithms are researched. At first they are used to decrease the computational complexity of depth first search strategy. However with the increasing interests in researching the breadth first search strategy because of its advantages in fixed data throughput, adopting the preprocess the preprocess in breadth first search also attract many researchers focuses. One of the widely used reordering algorithm is reordering the channel matrix $\mathbf{H}$ based on the norm of each column. Another possible solution is proposed by [20]. Those reordering preprocesses are all used in the depth first searching strategy at first to minimize the computational complexity. Although reordering the channel matrix $\mathbf{H}$ based on the norm of each column is quit a straight forward solution, it is not the optimal solution with the perfectly increased $r_{D,D}$ sequential. Since compared to the data throughput and signal frequency, the channel matrix changes in a quit slow frequency in most of the MIMO standards and models. There is enough time to perform the more complex reordering solution and QR decomposition in the outer co-processors. Simulation results of the preprocess adopting these two different reorder algorithms are shown.

2.3 Conclusion

In this chapter, the principle of K-BEST algorithm is introduced with details. A proper K is chosen to achieve the near-ML detection result. Figure 2.6 shows that when $K = 16$ it only has a minor decrease in the BER performance compared with the optimal depth first search strategy. As the kernel part of the detection, the channel matrix also make a strong impact on the performance. Two different reprocess methods performed on the channel matrix are adopted in the detection and their performances are evaluated. Simulation results in Figure 2.6 shows that the modified K-BEST detection algorithm with a combined



F

Figure 2.6: Simulation results: BER performance with and without preprocessing

preprocess can achieve the best performance while keeping the parallel architecture of the breadth first search strategy. In Figure 2.6 "Ascend" means to reorder the channel matrix **H** from the norm of each column in an ascending way, "Descend" denotes that the reoder process is performed by sortign the norm of each column in an descending order, while "Reorder" denotes the reoder process proposed in [20].

Chapter 3

Group Sorting Algorithm and Implementation

Although a sufficient large value of K increases the BER performance in the breadth first sphere detection, it also increases the computational complexity dramatically, due to not only the computing performed in Equation (2.7) but also the sorting process: the breadth first detection needs to find the K candidates with shortest partial Euclidean distance in each layer.

Let $|*|$ denotes the number of the possible points in the constellation system, i.e., after the complex to real tranformation the possible nodes corresponding to 16-QAM modulation becomes $* = \{-3, -1, 1, 3\}$, and so $|*| = 4$. In each layer the breadth first sphere detector should find out the K best points from the $|\Lambda| \times K$ extended points. By adopting bubble sorting in this process, the complexity in the worst case will be $O((|\Lambda| \times K)^2)$. Another disadvantage in the bubble sorting algorithm exists in its sequential process. Even the computing process in Equation (2.7) of the upper layer can be executed concurrently in a parallel structure, it still have to wait for the sorting process to be finished in the complexity of $O((|\Lambda| \times K)^2)$.

Since the sorting process becomes the major bottleneck that takes a long time in the K-BEST detection algorithm, many efforts have been made to reduce the computational complexity and execution time. In this chapter a group sorting algorithm implemented by

sorting network is introduced. Compared with other sorting algorithms like bubble sort that is performed on the whole candidates, the group sorting provides a lower complexity with a minor BER performance decrease while taking the best advantage of the parallel architecture of K-BEST MIMMO detection.

3.1 Improved Group Sorting Algorithm

In K-BEST MIMO detection, child nodes of K parent nodes are explored in every layer and K child nodes with the minimal partial Euclidean distance are chosen to be further explored in the next layer. In a 2^{Ω}-constellation system, after the complex to real tranformation the number of explored child nodes per parent node is $n = |\Lambda|$. The main idea of group sorting is that it divides child nodes in each layer into groups, and instead of being sorted in the whole layer by the traditional sorting process in the K-BEST algorithm, the child nodes are only sorted inside group. If the nodes are split into m groups then nK/m child nodes with the smallest partial Euclidean distances in each group are selected for the searching process in the next layer. The total computational complexity per layer becomes $O(m \times (nK/m)^2)$ if bubble sort is obtained in each group. Notice that m should be chosen carefully so that the number of nodes in each group, K/m, is an integer.

The architecture of group sorting in each layer is shown in Figure 3.1. The metrics in Equation (2.7) is obtained through the expanding unit (EU). After being epanded the candidates are split through the division module. Then a group of sorting modules execute the sorting process inside each group and choose best ones as their outputs.

Although group sorting is advantageous for its highly parallel architecture, its BER performance decreases compared with sorting nodes in the whole layer since sorting process inside group makes their outputs not globally optimal. Consider the example in Figure 3.2. In this example $K = 2$ and $n = 2$, while the number inside each node denotes its partial Euclidean distance. The candidate nodes are divided into 2 groups and each parent nodes generates 2 child candidates. In Figure 3.2a a skew happens in the second layer, meaning that a candidate with good child nodes is discarded during the group sorting. In this example, an erroneous node with partial Euclidean distance 7 in the second group survives

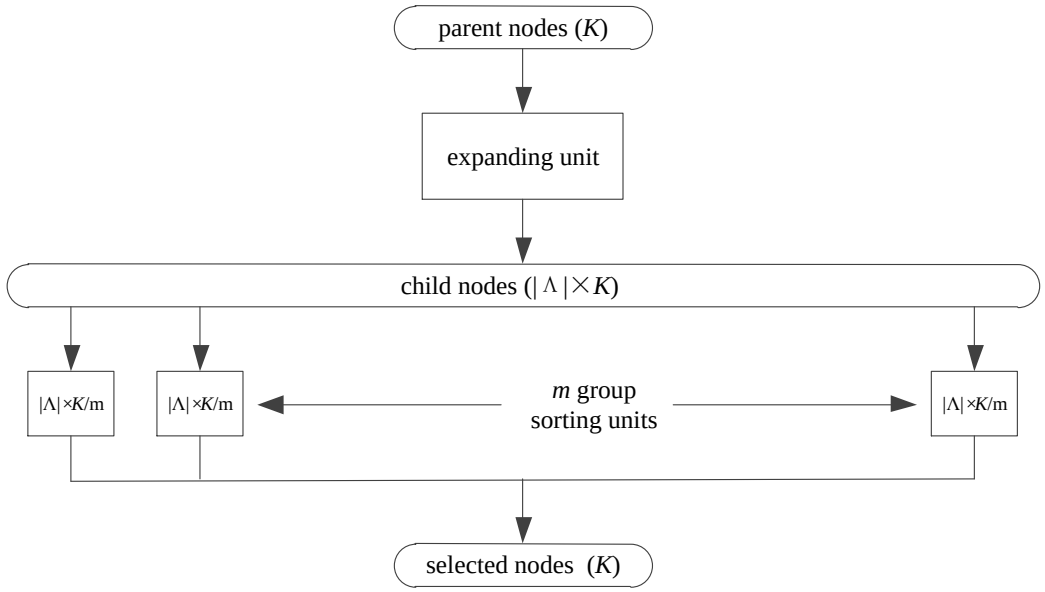
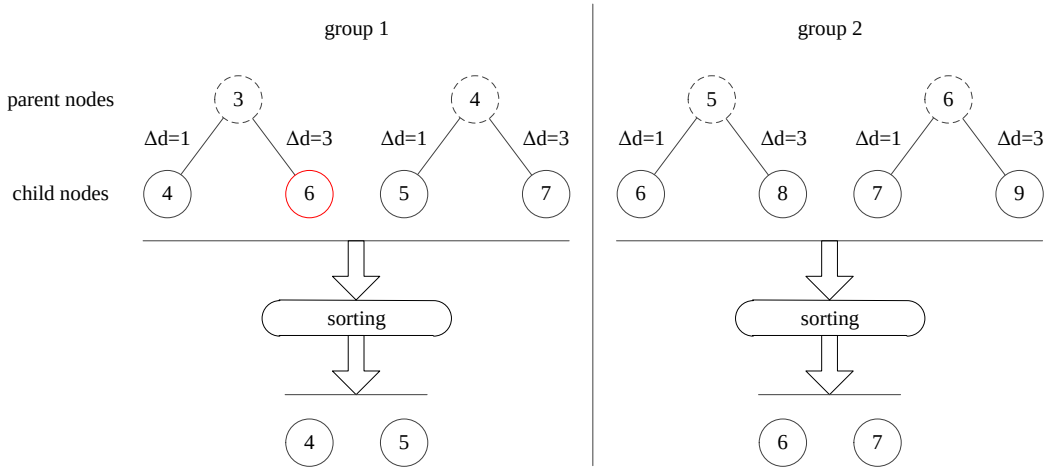


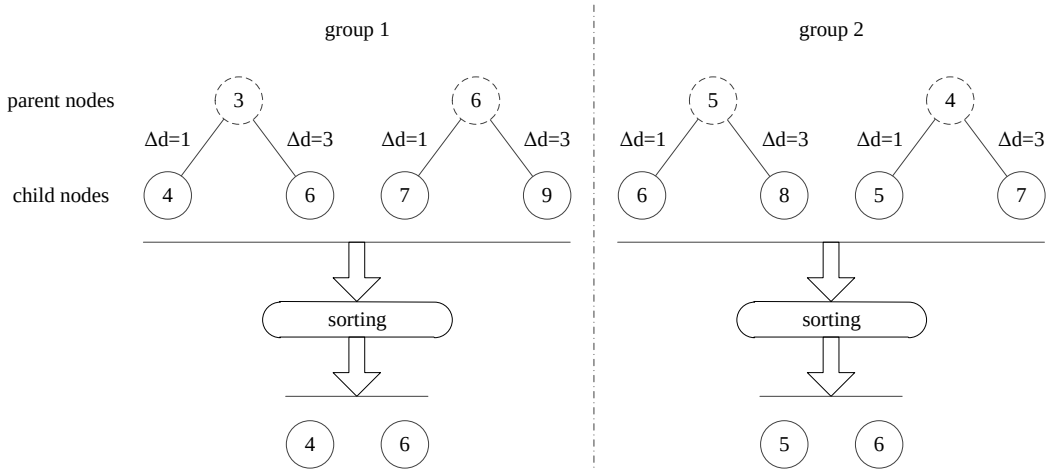
Figure 3.1: Group sorting structure

instead of the nodes with partial Euclidean distance 6 in the first group. In a complex MIMO system with a large number of antennae, this error could be accumulated during multiple stages of layered detection. However, this erroneous behavior would not occur if the distribution is even, which is shown in Figure 3.2b.

With the preprocessing part mentioned in Chapter 2, it is noticed that a parent node with small partial Euclidean distance has a large possibility to generate child nodes with small Euclidean distance [12]. If parent nodes with larger partial Euclidean distances are divided into different groups, the possibility of skew happening in group sorting could be reduced, compared with the original group sorting without distributing the parent nodes. To implement this modified group sorting the outpus of original group are interleaved to distribut those nodes with larger partial Euclidean distances into different groups, and then another group sorting is performed again. Although there are a large number of reports about the interleaving patterns, a direct approach is obtained in this part that can lead to an efficient interleaving and comparing results. This pattern is first introduced in



(a) Group sorting with skew in early level



(b) Groups sorting with interleaving

Figure 3.2: Skewed group sorting

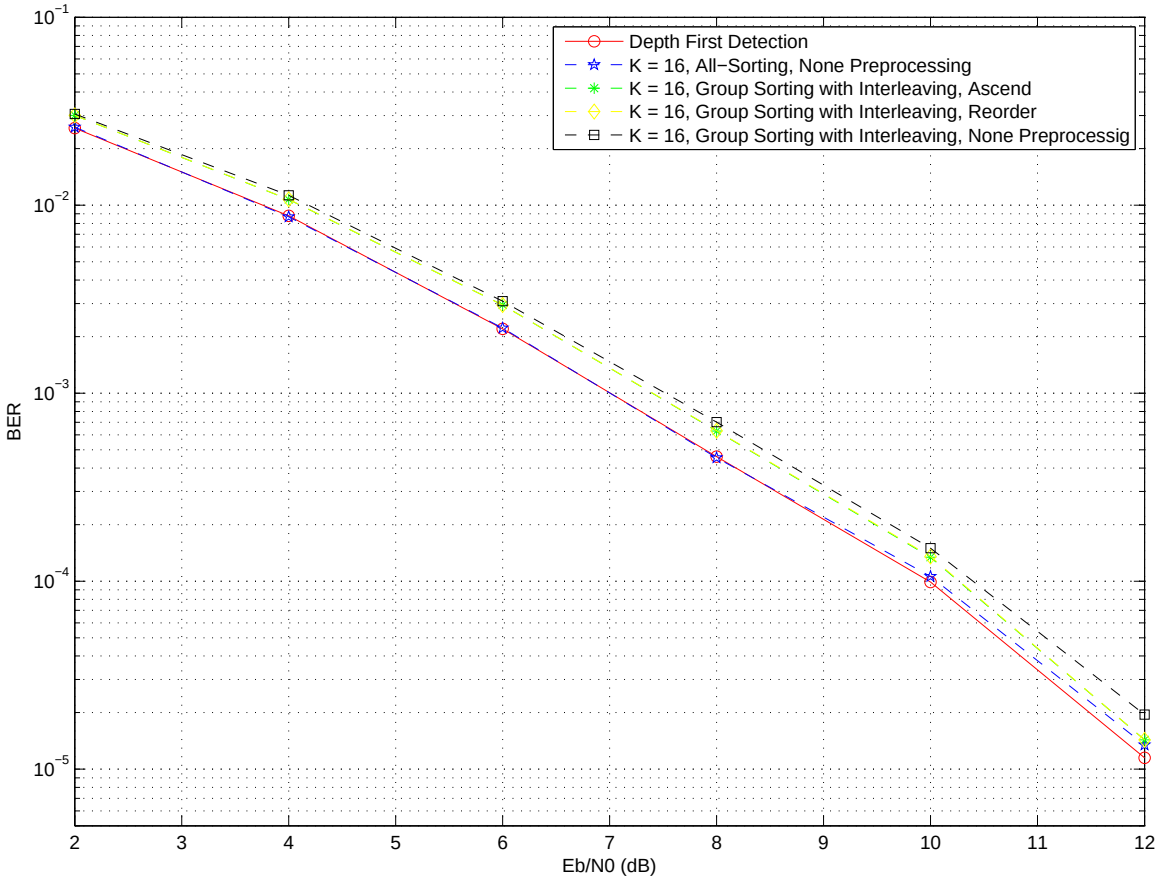


Figure 3.3: Simulation result: uncoded data

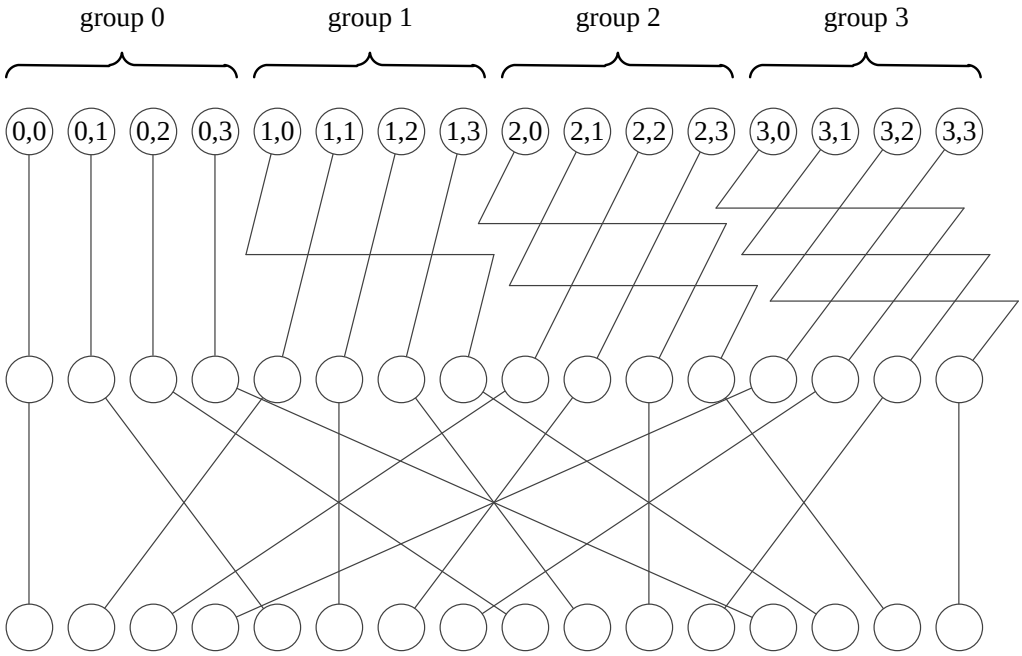


Figure 3.4: Interleaving process in sorting 16 candidates

[12]. Let G denotes the number of groups, l denotes the number of candidates in each group. For the general cases, if $l \geq G$, the sorted nodes in the w-th group are rotated by $w+((G-(wl)\%G)\%G)$, and the i-th node in the w-th group is assigned to the $((i+wl)\%G)$-th group, where % and / represent the modulo operation and integer division, respectively. Figure 3.4 is an example of interleaved sorting process performed in a list of 16 candidates divided into 4 groups. Figure 3.3 shows the results of the simulation performed in a 4×4 MIMO communication system with 16-QAM modulation. In this simulation 8 groups with $K = 16$ is obtained.

3.2 Parallel Sorting Network Architecture

As discussed at the beginning of this chapter, the computational complexity of sorting K candidates is $O(K^2)$ with bubble sorting algorithm. Although the group sorting with

interleaving dramatically decreases the computational time since the sorting process in each group can be executed in parallel, it still limits the efficiency of the whole system because the sequential property of bubble sorting. Although many different sorting algorithms are developed in computer science and mathematics, most of them only focuses on computational complexity and could not match the requirement of hardware implementation. In hardware design the property of parallel and pipelined architecture is as important as the computational complexity.

3.2.1 SE Enumeration

An important characteristic of child nodes enumerating process with the algorithm adopted by [20] is that the partial Euclidean distances of the child nodes are naturally ordered, because of the applying of Schnorr-Euchner (SE) searching strategy in the enumeration process. The principle of SE strategy is discribed in [16]. In Chapter 2 the steps of K-BEST sphere detection are introduced. For each parent node the enumeration process examines the child nodes in the constellation system and the major ideal for enumeration process in Chapter 2 is examining each nodes of the constellation from the bottom to the upper bound. Although the original enumeration is easy in implementation by software, it is not efficient enough. The SE algorithm is designed to enumerate those nodes in a different method. It is able to enumerate the candidate symbols in an ascending order according to their Euclidean distance from the Babai point. The enumerating details of step 3 in Algorithm 2.2 is shwon in Algorithm 3.1.

Figure 3.5 shows the scheme of SE enumeration performed in a 16-QAM modulation MIMO communication system. After the complex to real tranformation, candidate symbols are chosen from $* = \{-3, -1, 1, 3\}$, which are enumerated in different orders decided by the position of the Babai point in that layer. Although the pseudo code description is complex, for hardware implementation the SE enumeration can be simplified using a specific method- a look up table (LUT) . Gaining the Babai point information, the LUT would generate the child nodes in a decreasing order from their partial Euclidean distances. Then the expanding unit (EU) generates relevant metric for each nodes.

Recall Equation (2.7). It is obvious that the additive squared partial Euclidean distance

Algorithm 3.1 SE enumeration algorithm

3.a: Find the Babai point of layer k: $B_k = \overline{y}(s_{2\,D+1})_D / r_{D,D}$, set $i = 1$;

3.b: Round the Babai point to the nearest constellation point: $s_{ki} = \lceil B_k \rfloor$;

3.c: If $s_{ki} \geq B_k$ goto 3.f

3.d: If $i = |\Lambda|$terminate the enumeration. Else $i = i + 1$, $s_{ki} = s_{k(i-1)} + (i+1)step$.

3.e: If $s_{ki} < UB$ then $i = i - 1$

3.f: If $i = |\Lambda|$terminate the enumeration. Else $i = i + 1$, $s_{ki} = s_{k(i-1)} - (i-1)step$.

3.g: If $s_{ki} < LB$ then $i = i - 1$

3.h: Goto 3.g.

in each layer is decided by $d^2(\overline{y}(s_{D+1})_D, r_{D,D}s_D)$, $D = 1, 2, \ldots .2M_T$. The SE enumeration is applied to enumerate the child nodes in a decreasing order from their partial Euclidean distance. This equals to picking up the child nodes according to their distances among the possible constellation nodes and the received node, which is $\overline{s}_D = \overline{y}(s_{D+1}^{2M_T})_D / r_{D,D}$, $D = 1, 2, \ldots .2M_T$. Exploring such an enumeration process in an infinite integer set can also uses the SE enumeration but the efficiency is poor because a large size of table is required to include all the possible information. Since the candidate nodes are all in a finite set decided by the modulation constellation in MIMO, a small table is designed for the LUT implementation of SE enumeration. In a 16-QAM modulation system the constellation is $\Lambda = \{-3, -1, 1, 3\}$, and the best order will be the i-th row of the table Φ:

$$i = \phi(\overline{s}_D) = \begin{cases} 1 & \overline{s}_D \leq -2 \\ 8 & \overline{s}_D \geq 2 \\ \lceil \overline{s}_D \rceil + 3 & otherwise \end{cases} \tag{3.1}$$

where $\lceil . \rceil$ denotes rounding the value towards to the celling and

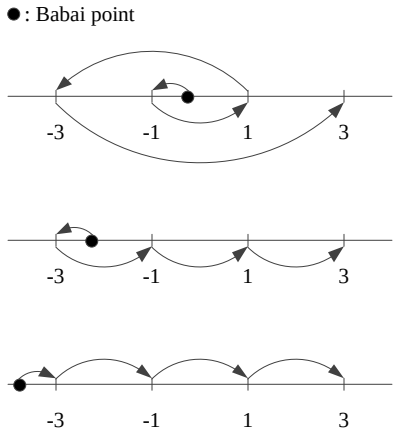


Figure 3.5: SE enumeration

$$\Phi = \begin{bmatrix} -3 & -1 & 1 & 3 \\ -1 & -3 & 1 & 3 \\ -1 & 1 & -3 & 3 \\ 1 & -1 & 3 & -3 \\ 1 & 3 & -1 & -3 \\ 3 & 1 & -1 & -3 \end{bmatrix} \tag{3.2}$$

3.2.2 Sorting Network Analysis

The traditional bubble sorting algorithm has a computational complexity of $O((4K)^2)$ when performed in the whole candidate list with a length of $4 \times K$. This sorting algorithm, although is straight forward, is low efficient when is obtained in MIMO detection, considering the length of the candidate list, the number of layers in which the sorting process is performed, and its sequential property in time domain. There are many different sorting algorithms with dramatically low computational complexity compared with bubble sorting.

The basic element of the sorting network is the comparator. Two numbers, A and B are received over the inputs of the comparator. If $A \leq B$ the outputs keep the same as

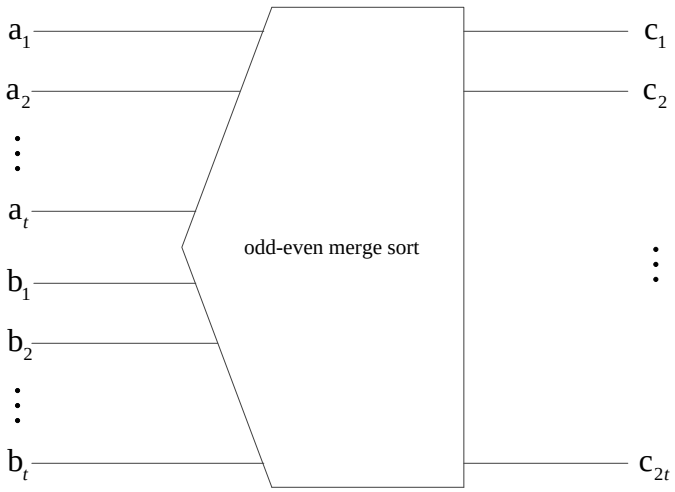
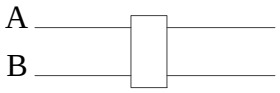


Figure 3.6: Odd-even merge sort

the input, otherwise the comparator exchange positions of A and B and get the outputs. Although sharing the same basic elements, the sorting network have different computational complexity and depth-which is also the delay in hardware design . With the partially ordered candidate list generated by the SE enumeration strategy, an odd-even merging networks is presented in this section.

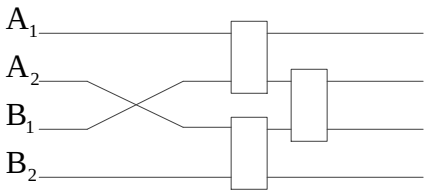
Merging means the process of aligning two ordered list of candidates into one ordered sequence. Figure 3.6 shows a merge network in which the numbers of one ascending-ordered candidates, a_1, a_2, a_3, ..., a_t are presented over t inputs simultaneously with another ascending-ordered candidates b_1, b_2, b_3, ..., b_t over another t inputs. The $2t$ outputs of the merging network present the $2t$ candidates of the merged result in ascending order, c_1, c_2, c_3, ..., c_{2t}. This is a t by t merging network. A 1 by 1 merging network is simply one comparator as the basic element. Larger networks are built by using this element iteratively and row of comparators across the outputs of smaller merging networks.

An odd-even merging network is applied in the first stage of group sorting introduced in Section 3.1 before the interleaving part, with partially ordered input candidates. A t by t merging network can be build by presenting the inputs to two smaller merging

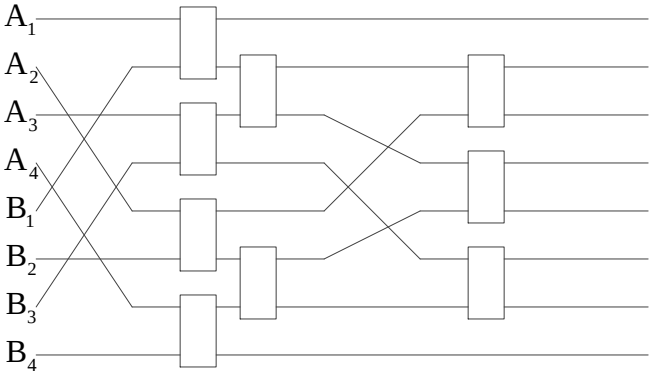


the

(a) 1×1 odd-even merge sort



(b) 2×2 odd-even merge sort



(c) 4×4 odd-even merge sort

Figure 3.7: 4×4 odd-even merge sorting network

networks. The odd-indexed numbers of the two input lists is transmitted to one small merging networks, which is the odd merge, while presenting the even indexed number to another small merging network, which is the even merge. Then the outputs of these small merges are compared by a row of comparator across them. Both the even merging network and the odd merging network have the ability of sorting $t/2$ inputs, and are also obtained by smaller odd and even merging networks, which process $t/4$ inputs with rows of comparators. Figure shows the structure of a 4 by 4 odd-even merging network. With the partially ordered inputs the 4 by 4 odd even merging network contains 25 comparators with a delay of 4 cycles.

After the interleaving process of group sorting, another sorting process is needed for obtaining the final candidates on each layer. Since the candidates are not partially ordered, a sorting network with more complexity, based on the 4×4 odd-even merging network, is applied. The architecture of this sorting network is shown in Figure 3.8. For an 8-input sorting network in each group the total cost is 28 comparators with a delay of 9 cycles. This architecture takes $28 \times 8 = 224$ comparators with 9 cycles delay. In Figure 3.8 an 8-input sorting network is added after the interleaving process. Since $K = 16$ candidates are chosen in each layer, only first 4 candidates are needed in each group. The comparators which are relevant to the last 4 outputs are not needed in each layer for the MIMO detection.

3.3 Conclusion

In this chapter the main bottleneck of sphere detection, the sorting process, is investgated. To avoid the conflict between the detection performance and a limited K value of the candidates number, the group sorting with interleaving is introduced. While keeping a limited descent in BER performance it dramatically mitigates the delay and decreases computational complexity, compared with the traditional sorting method which is performed on the whole candidates. A highly efficient SE numeration method is also introduced using the LUT. Compared with original Zig-Zag method with complex status, the LUT only need to maintain a table whose complexity is decided by the chosen modulation system. Two sorting network implementations are also applied before and after the interleaving process.

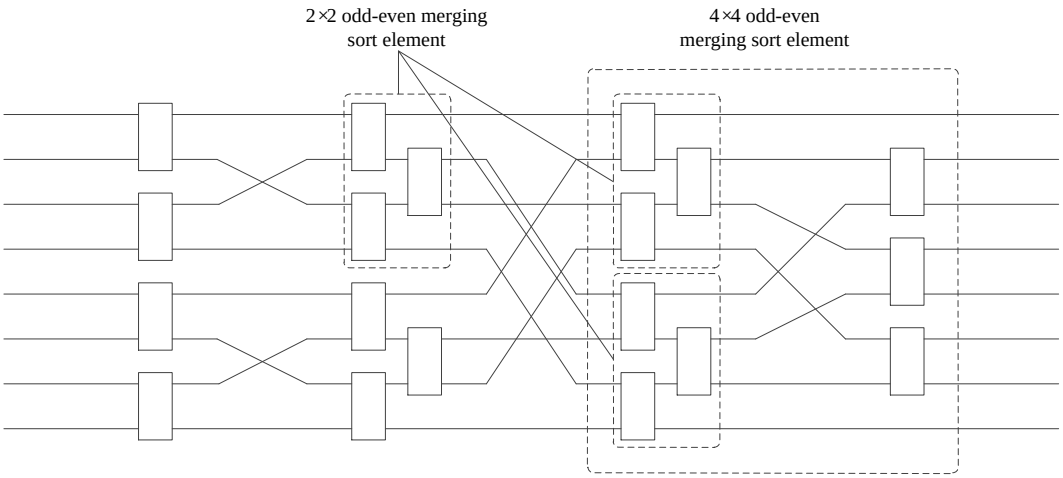


Figure 3.8: 8 input sorting network behind the interleaver

With the help of SE enumeration the odd even merging network can dramatically decrease the complexity and delay of sorting process.

Chapter 4

Soft-Output MIMO Detection

A MIMO detector can make decisions on those detected uncoded information bits either by nulling/ZF or ML detection algorithms. However, in most cases these detected bits are coded by an outer encoder/interleaver component. Consider an $M_T\Omega \times 1$ dimensional binary vector $\mathbf{x} = (\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_{M_T})^T$ in which $\mathrm{x}_i = (x_{i1}, x_{i2}, \ldots, x_{i\Omega})$, which is obtained from a large sequence. It could be a sequence of channel code with rate $R \leq 1$ that introduces redundancy and correlation among its data entries. The transmitted information contains $RM_T\Omega$ data bits. The MIMO model in Equation (1.1) is used iteratively to transmit a continuous stream of data bits which are separated into blocks. For any given block $\mathbf{x}$ symbol $\mathbf{s}$ is obtained by using a modulation function $\mathbf{s} = \mathrm{map}(\mathbf{x})$. To decode the original sequence optimally, the signal detection and channel decoding process should be operated jointly on all data blocks because of the correlation information between these blocks introduced by an outer encoder. With the soft information of all the blocks obtained by the signal detector, the channel decoder should decode the information data through an iterative method.

4.1 Introduction to Soft Information

The channel code and MIMO channel can be regarded as a serially concatenated scheme, with an outer channel encoder, bit interleaver and inner space-time constellation mapping. Its result is then transmitted through channel matrix $\mathbf{H}$. Figure 4.1 shows a standard flowchart of an iterative MIMO detector with an outer encoder/decoder system. To decode

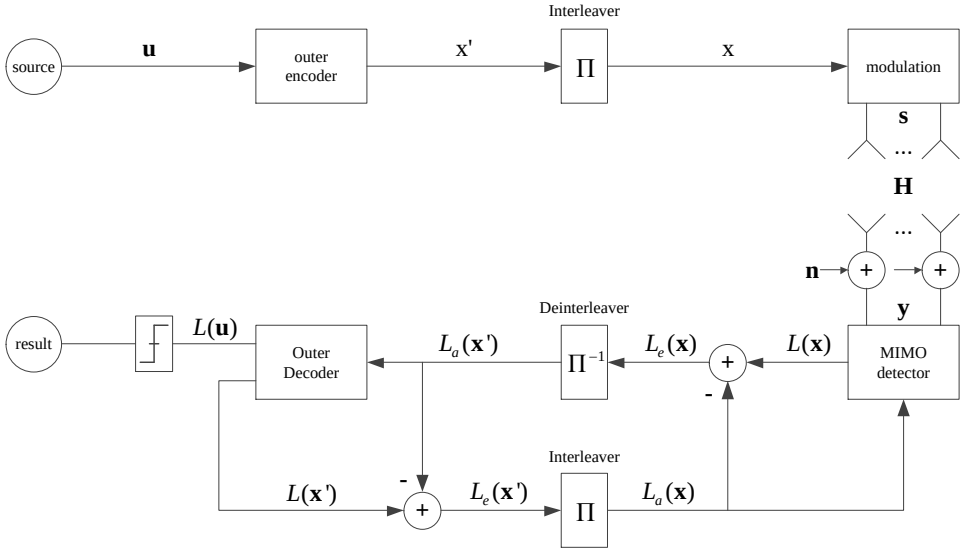


Figure 4.1: Combined detection/decoding MIMO system

the binary data $\mathbf{x}$ optimally, the joint detector/decoder system should compute the likelihood value of each bit given all the received complex data $\mathbf{y}$. An exchange/incorporate scheme is adopted to solve this problem.

In Figure 4.1 the MIMO detector incorporate soft reliability information provided by the channel decoder, and the channel decoder incorporate soft information provided by the MIMO detector. Information between the detector and decoder is then exchanged in an iterative fashion until desired performance is achieved. A soft-output MIMO detector receives the channel observations $\mathbf{y}$ as well as *a priori* information $L_a(\mathbf{x})$ by decoding the inner coded bits, and then *extrinsic* information $L_e(\mathbf{x})$ is calculated for each of the coded bits. After that through a deinterleaver *a priori* input $L_a(\mathbf{x}')$ for the outer soft-in/soft-out decoder is gained. The out decoder calculates the *extrinsic* information $L_e(\mathbf{x}')$ on the outer coded bits and then $L_e(\mathbf{x}')$ is reinterleaved and fed back as *a priori* information $L_a(\mathbf{x})$ to the inner decoder, thus completing an iteration. After some iterations the outer decoder makes decisions $\hat{\mathbf{u}}$ about the information bits by *a posteriori* information $L(\mathbf{u})$ [11].

Although the overall flow described in Figure 4.1 is generally accepted and standard, the structures and process within each of these sub blocks determine BER performance,

complexity and feasibility of the joint detection/decoding algorithm. Since the convolutional code is used in this thesis, the outer encoder and decoder are also relatively standard. To obtain soft information for outer decoder, the detection part should be modified, compared with the detector introduced in Chapter 2, so that the destination of computationally efficient, BER performance and the data throughput can be achieved.

4.1.1 APP Detection in MIMO system

In a iterative MIMO receiver, the MIMO detector needs to generate *a posteriori* probability (APP) about the inner coded bits $\mathbf{x}$. Maximizing the APP for a given bit minimizes the probability of making an error on it. The APP is usually expressed as a log-likelihood ratio value (LLR), which provides a convenient notation for describing the operation of iterative decoding algorithms. As shown in Figure 4.1 simple add/subtract operations are sufficient to separate a priori from extrinsic information during the APP detection/decoding cycle. Usually only extrinsic information is exchanged in processing cycles. A decision is made from an LLR by using its sign to tell whether the bit is a '1' or '0' and the magnitude of the LLR indicates the reliability of this decision. In this model $M_T\Omega \times 1$ coded bits are processed at one time and the LLR is defined as:

$$L(\mathrm{x}_i) = (L(x_{i1}),\, L(x_{i2}),\, \ldots L(x_{i\Omega})) \tag{4.1}$$

$$L(x_{ij}) = \ln\frac{P(x_{ij} = +1)}{P(x_{ij} = -1)} \tag{4.2}$$

and notice that the logic 0 is expressed as amplitude level -1. Conditioned on the received vector y, the LLR of bit x_{mn} is defined in the following way:

$$L(x_{mn}|\mathbf{y}) = \ln\frac{P(x_{mn} = +1|\mathbf{y})}{P(x_{mn} = -1|\mathbf{y})} \tag{4.3}$$

Since the coded bits are interleaved before constellation mapper/modulation, bits in $\mathbf{x}$ are statistically independent of one another. With standard manipulation according to

Bayes' rule, this can be modified as:

$$L(x_{mn}|\mathbf{y}) = \ln\frac{p(\mathbf{y}|x_{mn}=+1)\cdot P(x_{mn}=+1)/p(\mathbf{y})}{p(\mathbf{y}|x_{mn}=-1)\cdot P(x_{mn}=-1)/p(\mathbf{y})} \quad (4.4)$$

$$= \ln\frac{P(x_{mn}=+1)}{P(x_{mn}=-1)} + \ln\frac{\mathrm{p}(\mathbf{y}|\mathrm{x_{mn}}=+1)}{\mathrm{p}(\mathbf{y}|\mathrm{x_{mn}}=-1)} \quad (4.5)$$

By definition the first term of Equation (4.5) represents the a priori LLR of bit x_{mn}, and the second term is the *extrinsic* LLR that can be modified by taking the expectation of $p(\mathbf{y}|\mathbf{x})$ over $\mathbb{X}_{mn,\pm1} = \{\mathbf{x}|x_{mn} = \pm1\}$, which means the set of $M_T \times \Omega$ bits vector $\mathbf{x}$ has $x_{mn} = \pm1$, $m = 1, 2, \ldots M_T$ and $n = 1, 2, \ldots \Omega$:

$$L(x_{mn}|\mathbf{y}) = L_{a,mn} + \ln\frac{\sum_{\mathbb{X}_{mn,+1}} p(\mathbf{y}|\mathbf{x})\cdot P(\mathbf{x}|x_{mn})}{\sum_{\mathbb{X}_{mn,-1}} p(\mathbf{y}|\mathbf{x})\cdot P(\mathbf{x}|x_{mn})} \quad (4.6)$$

A standard simplification in the calculation of *a posteriori* LLR is the assumption that $\ln\sum a_j \approx \max\{\ln a_j\}$ and then Equation (4.6) is further simplified as:

$$L(x_{mn}|\mathbf{y}) \approx L_{a,mn} + \max_{\mathbb{X}_{mn,+1}}\{\ln p(\mathbf{y}|\mathbf{x}) + \ln P(\mathbf{x}|x_{mn})\} - \max_{\mathbb{X}_{mn,-1}}\{\ln p(\mathbf{y}|\mathbf{x}) + \ln P(\mathbf{x}|x_{mn})\} \quad (4.7)$$

4.1.2 The Iterative LLR Decoding

With the definition of a priori LLR and the independence of the bits $\mathbf{x}$,

$$P(\mathbf{x}|x_{mn}) = \prod_{(ij)\neq(mn)} P(x_{ij}) \quad (4.8)$$

$$= \prod_{(ij)\neq(mn)} \frac{\exp(x_{ij}L_a(x_{ij})/2)}{\exp(\frac{L_a(x_{ij})}{2}) + \exp(-\frac{L_a(x_{ij})}{2})} \quad (4.9)$$

and

$$\ln P(\mathbf{x}|x_{mn}) = \left(\sum_{ij} \ln P(x_{ij})\right) - \ln P(x_{mn}), \quad (4.10)$$

$$\ln P(x_{ij}) = \frac{1}{2}x_{ij}L_a(x_{ij}) - \ln(\exp(\frac{1}{2}L_a(x_{ij})) + \exp(-\frac{1}{2}L_a(x_{ij}))) \quad (4.11)$$

$$\approx \frac{1}{2}x_{ij}L_a(x_{ij}) - \frac{1}{2}\left|L_a(x_{ij})\right| \quad (4.12)$$

$$= \left|L_a(x_{ij})/2\right|\left(x_{ij}\mathrm{sign}(L_a(x_{ij})) - 1\right) \quad (4.13)$$

In [11] it is introduced that due to the matrix structure of the MIMO channel, interference between the transmitted symbols is applied at the receiver, and the probability of a

particular realization of the received vector, conditioned on the transmitted vector, is given by a multi-dimensional Gaussian distribution

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{(\pi N_0)^{M_T}} \exp(-\frac{1}{N_0} \|\mathbf{y} - \mathbf{Hs}\|^2) \quad (4.14)$$

$$\ln p(\mathbf{y}|\mathbf{x}) = -\frac{M_T}{2} \ln(\pi N_0) - \frac{1}{N_0} \|\mathbf{y} - \mathbf{Hs}\|^2 \quad (4.15)$$

Applying Equation (4.15) and Equation (4.10) to the extrinsic part of Equation (4.6), the extrinsic LLR can be expressed as follows:

$$L_e(x_{mn}|\mathbf{y}) = \max_{\mathbb{X}_{mn,+1}} \{\Lambda(\mathbf{x}, \mathbf{y}, L(\mathbf{x}))\} - \max_{\mathbb{X}_{mn,-1}} \{\Lambda(\mathbf{x}, \mathbf{y}, L(\mathbf{x}))\} \quad (4.16)$$

$$\Lambda(\mathbf{x}, \mathbf{y}, L(\mathbf{x})) = -\frac{1}{N_0} \|\mathbf{y} - \mathbf{Hs}\|^2 + \sum_{ij} \ln P(x_{ij}) \quad (4.17)$$

where $\ln P(x_{ij})$ can be calculated from the a priori LLR $L_a(x_{ij})$.

4.2 Soft Information Generation Algorithms

The first term of Equation (4.17) can be implemented using the detection algorithms introduced in Chapter 2 by searching all the possible points in the multi-dimensional lattice. However to avoid exhaustive search, [9] proposed an algorithm to search in a limited set of candidates, which is a subset of the whole lattice. A candidate list is generated once for each block by the MIMO detector but the a priori LLR L_a and APP value $L(x_{mn}|\mathbf{y})$ has to be updated in every iteration. In an uncoded MIMO detection system the detector in Figure 4.1 actually is a candidate list generation block combined with a soft information calculating block.

4.2.1 Candidate List with Distance Information Generating

In [11] the depth first searching strategy is applied to limit the searching list for Equation (4.16). Inspired by this the breadth first searchig strategy is also able to be adopted in obtaining a similar candidate list. Recall the modified K-BEST algorithm introduced in Chapter 2. As a modification of original breadth first sphere detection, for each layer it keeps K candidates which are expanded for the next layer. In the last layer, after the sorting

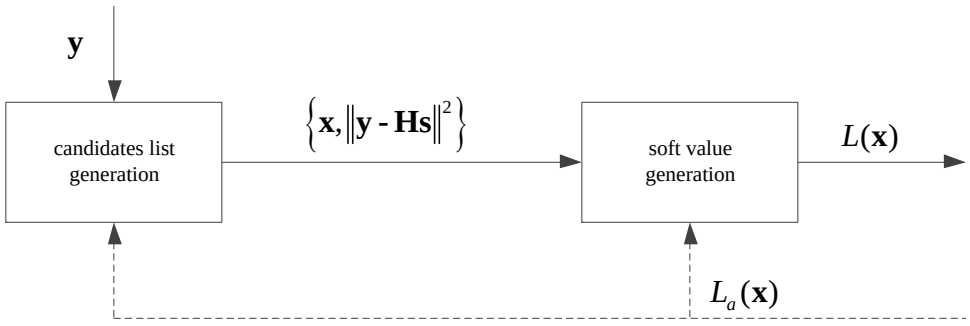


Figure 4.2: MIMO iterative detection/decoding model

network K final candidates are kept, which means that the proposed detection algorithm support both soft-output as well as hard-outputs. The K best candidates retained at the last layer can be used as the candidates list, in which the candidate with the best total Euclidean distance is the hard-output. Since the outer decoder makes decision based on the LLR by using its sign to decide whether the bit is a '1' or '0' while the magnitude of the LLR indicates the reliability of this decision, ML point is not a necessary part for LLR calculation but only helps to increase BER performance. Consequently the modified K-BEST algorithm only need a "small" expanded list to gain the LLR value for the outer decoder. In other words a detector with the capability of generating a candidates list is simply a soft MIMO detector.

Although [11] introduced an important soft information detection algorithm based on the depth first searching strategy, compared with the breadth first searching strategy, the iterative two-direction search process makes its data throughput unstable and so there are many limitations on its application, like large IO buffers that maybe an extra overhead in the real practical system. The proposed modified K-BEST detection, however, has advantages in hardware implementations because of its one-direction searching strategy and parallel property. Moreover, due to the layered single direction search process, the modified K-BEST detection is able to be implemented in a pipelined architecture. Each pipeline stage corresponds to one layer in the algorithm. The fixed high detection throughput, thus, is

possible in this detection.

Although in each detection cycle the detector with the modified K-BEST algorithm is able to generate a candidate list, the soft information calculating part, however, as implied in Equation (4.16), requires a iterative decoder process. The a priori LLR L_a should be updated iterativly in each iteration. Although the performance could be improved with the growing of the number of iterations, the computational complexity is also significantly increased compared with the case in which the second term of Equation (4.17) is ignored. In this thesis only the first part of Equation (4.17) is taken into account. So the extrinsic LLR is modified to:

$$\begin{aligned} L_e(x_{mn}|\mathbf{y}) &= \max_{\mathbb{X}_{mn,+1}} \{\Lambda(\mathbf{x}, \mathbf{y}, L(\mathbf{x}))\} - \max_{\mathbb{X}_{mn,-1}} \{\Lambda(\mathbf{x}, \mathbf{y}, L(\mathbf{x}))\} \\ &\approx \max_{\mathbb{X}_{mn,+1}} \{-\frac{1}{N_0} \|\mathbf{y} - \mathbf{Hs}\|^2\} - \max_{\mathbb{X}_{mn,-1}} \{-\frac{1}{N_0} \|\mathbf{y} - \mathbf{Hs}\|^2\} \\ &= \min_{\mathbb{X}_{mn,+1}} \{\frac{1}{N_0} \|\mathbf{y} - \mathbf{Hs}\|^2\} - \min_{\mathbb{X}_{mn,-1}} \{\frac{1}{N_0} \|\mathbf{y} - \mathbf{Hs}\|^2\} \end{aligned} \tag{4.18}$$

4.2.2 Computing LLR Value

In Equation (4.18), when the modified K-BEST detection is adopted, the possible candidate set, $\mathbb{X}$, is replaced by the detected candidate list, $\mathbb{L}$. Instead of searching all the possible nodes in the solution domain, the search is limited inside the small candidate listed in $\mathbb{L}$. The LLR value calculating process, so, is to perform Equation (4.19) for each bit.

$$L_e(x_{mn}|\mathbf{y}) = \min_{\mathbb{L}_{mn,+1}} \{\frac{1}{N_0} \|\mathbf{y} - \mathbf{Hs}\|^2\} - \min_{\mathbb{L}_{mn,-1}} \{\frac{1}{N_0} \|\mathbf{y} - \mathbf{Hs}\|^2\} \tag{4.19}$$

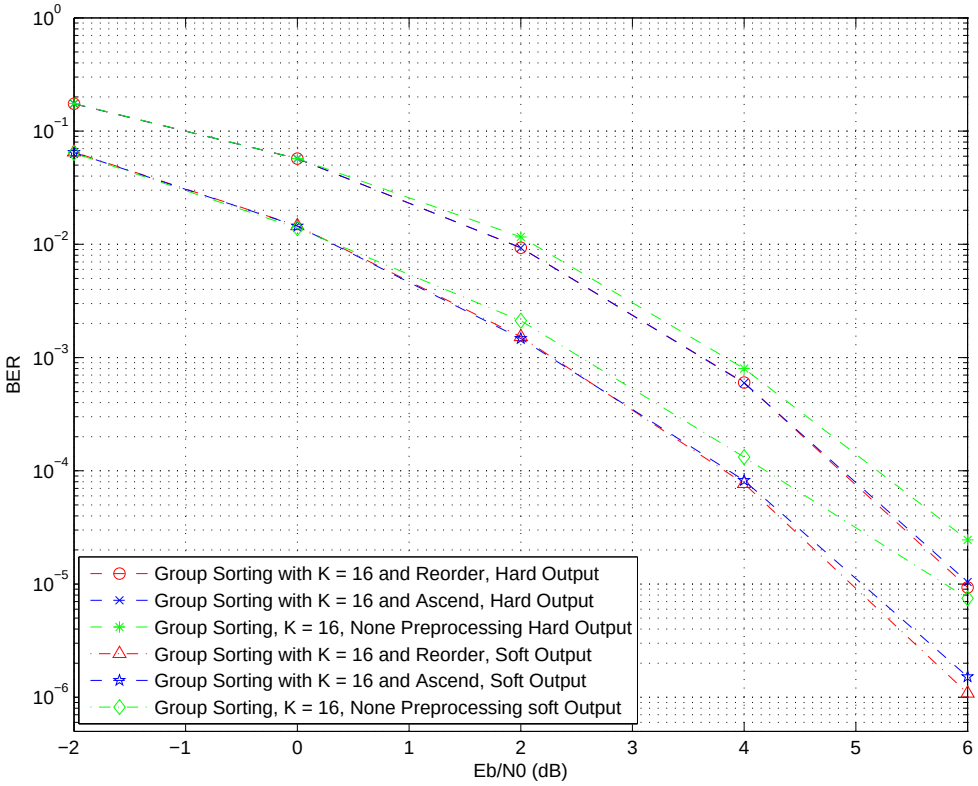
After the detection process in the last layer, all the candidates are ordered from their total Euclidean distance. Those candidates, which are represented by symbols in the constellation, are mapped into binary bits. Notice that the binary 0 should be transformed into -1, i.e. every $M_T\Omega$ bits correspond to one distance and there are K groups of these bits corresponding with K distance in an ascending order. So for Equation (4.19) $\mathbb{L} = \{L_1, L_2, \dots L_K\}$ and $L_i = (l^i_{11}, l^i_{12}, \dots l^i_{1\Omega}, \dots l^i_{M_T1}, l^i_{M_T2}, \dots l^i_{M_T\Omega})$; $m = 1, 2, \dots M_T$ and $n = 1, 2, \dots \Omega$.

For the first group of candidates no matter whether each bit of it is -1 or $+1$, this group of bits satisfies either the first term or the second term of Equation (4.19) since all bits in this group are corresponding the minimal Euclidean distance. The problem now, is to find the minimal Euclidean distance in other groups with a bit having opposite value of the bit at the corresponding position in the first group so that the other unknown term of the Equation (4.19) could be matched.

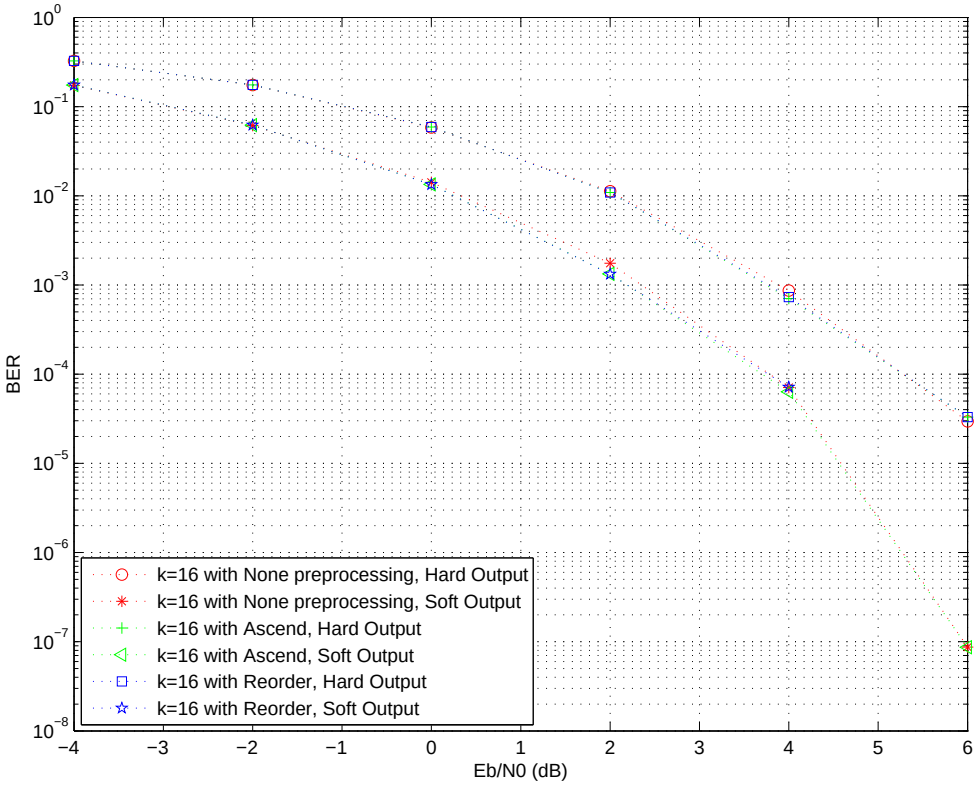
Suppose the LLR of the k-th bit needs to be found with $k = 1, 2, \ldots M_T\Omega$. In the first group L_1 the k-th bit a, $a \in \{1, -1\}$, is found as well as the minimal Euclidean distance d_1. From the 2nd to the K-th group each candidate's k-th bit l_k^i, $i = 1, 2, \ldots K$ is checked. If $l_k^i = -a$ then the target one with smallest Euclidean distance d_i as well as an opposite k-th bit value is obtained. As each bit is independent from others in every candidate the checking process for the whole $M_T\Omega$ bits can be executed at the same time.

4.3 Conclusion

In this chapter, the soft information decoding process is introduced and obtained with the modified K-BEST detection. Combined with outer decoder, like Turbo decoder or Viterbi decoder the performance of the MIMO communication system can be increased dramatically. To obtain soft information the definition of log-likelihood value (LLR) is introduced as well as the algorithm of LLR calculating process. With modification over K-BEST detection the candidate list generation and LLR computing parts are added with the detection module. Figure 4.3 shows the simulation results of a coded MIMO system which generates both the soft and hard outputs. The group sorting algorithm decreasse BER performance of MIMO detection but the soft-output with proper preprocessing helps to decrease the difference between these two sorting strategies. As mentioned at the beginning this system has a 4×4 antenna array and 16-QAM modulation is adopted with a rate $1/2$ convolutional code.



(a) Group sorting



Chapter 5

Modified MIMO Detector Hardware Architecture

This chapter is focused on the hardware implementation and architecture design. Details of implementation for each part based on the principles of the modified K-BEST MIMO detection and soft information generation introduced before is represented. To make the best use of the parallel property of proposed algorithm, different hardware implementation techniques, including the finite state machine (FSM), look-up-table (LUT), etc.

5.1 Architecture Design

The whole modified K-BEST MIMO detection algorithm is built based on the QR decomposition which transfers the complex channel matrix into upper triangle matrix. This makes the received signals divisible into layers so that a trail tree searching model can be applied in the detection as shown in Figure 2.1. Although many researchers proposed their results in implementing the QR decomposition algorithm into hardware structure [15], it is not necessarily required, since in the application the channel model is considered to keep the same value for a long time, compared to the transmitted data that is changing at a high rate, around Gbps. In this thesis the QR decomposition is supposed to be carried out by a co-processor, which can use the mature algorithms implemented in C or other high

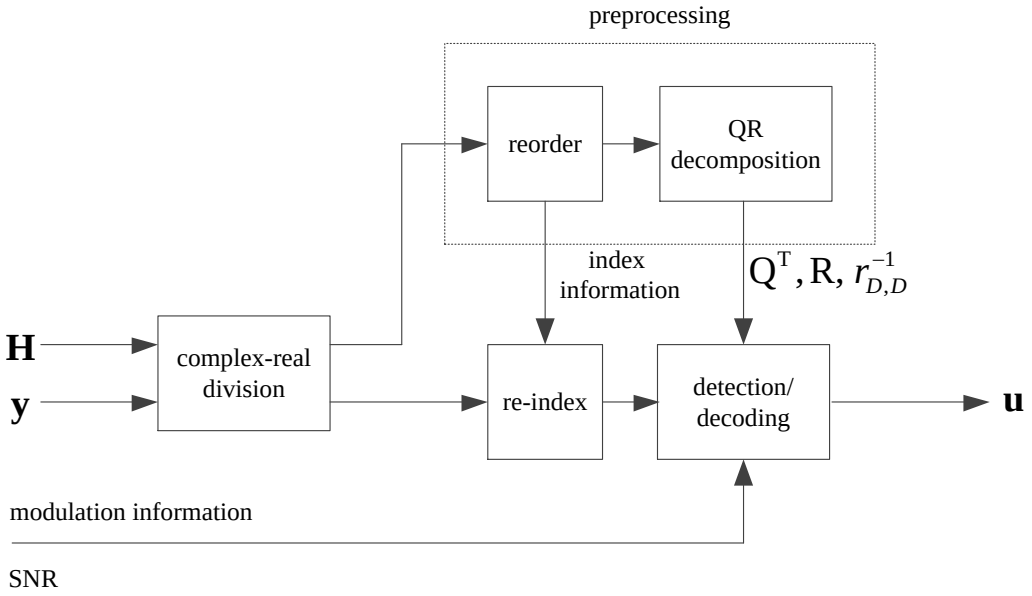


Figure 5.1: The block diagram of MIMO detection system

level languages and then is complied into machine code and executed. Although compared with the FPGA or ASIC the speed of the general processor is slow, there is still enough time for it to solve the decomposition results and transmit them to the detector. The main structure of the proposed modified K-BEST detection is shown in Figure 5.1.

In Figure 5.1 the complex to real tranformation is performed first on the channel matrix and the received signals, according to Equation (1.3). Then the channel matrix is reordered and the QR decomposition is obtained through the QR decomposition block. both of these two processes are performed in the preprocessing part. This preprocessing is also performed on an outer co-processor. The re-indexed block reorder the received signals **y** according to the index information supplied by the reorder block. The detection/decoding block is the kernel part of the implementation proposed by this thesis.

5.2 Layer Detection Part

For a 4×4 MIMO communication system with 16-QAM modulation, after the complex to real tranformation, the detection process, as introduced in Equation (2.7), (2.8) and (2.9), can be divided into 8 layers. At the beginning layer, Layer 8, as introduced in Figure 2.1, 4 different child nodes are expanded in an order from their partial Euclidean distances to the Babai point of that layer by SE enumeration. The order information is gained through the LUT structure.

The EU block, as shown in Figure 5.2, is designed to calculate the value of $\overline{y}(s_{D+1})$ and the partial Euclidean distance, which are introduced in Equation (2.11), (2.12), and (2.13), respectively. The partial Euclidean distance and $\overline{y}(s_{D+1})$, according to the analysis in Chapter 3, is a function of the selected points in all the layers before the current one. So the examined candidate nodes information should be kept in each layer and transmitted to next layer for the further use in the EU block.

In a MIMO detection system which obtains breadth first search strategy, in Layer D, with $K \leq |\Lambda|^D$, the number of candidate nodes is larger than the required candidate list length. Then a sorting unit is required to choose the K candidates with the best partial Euclidean distance from all the $|\Lambda| \times K$ possible child nodes. And as analyzed before, the sorting unit should be able to transmit the candidate information as well as the partial Euclidean distance value of each selected candidate to the next level.

The designed structure of the layer detection is shown in Figure 5.2. The LUT, as introduced in Section 3.2.1, is designed for generating the child nodes with SE enumeration algorithm and so that the their partial Euclidean distance is aligned in a descending order. Equation (2.11) is implemented by multiple steps of substraction combined with an EU block, whose inputs are the value of s_{D+1} and $\overline{y}(s_{D+2})$ from the LUT and buffer of last layer, respectively. As the EU module takes multiple cycles to finish the expanding process, a serial of buffers are obtained between each layer so that the detection process can work with a fully pipelined structure.

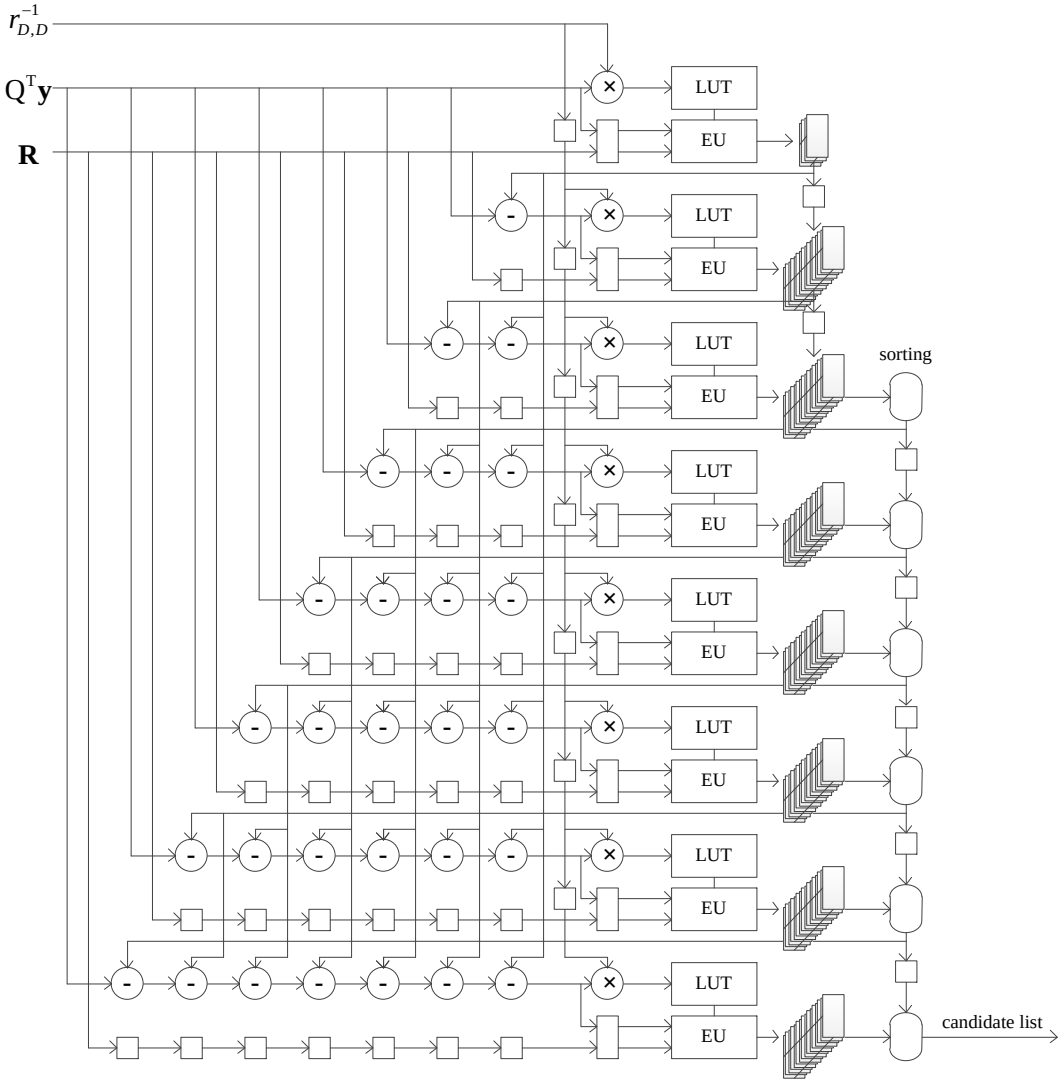


Figure 5.2: MIMO layer detector structure

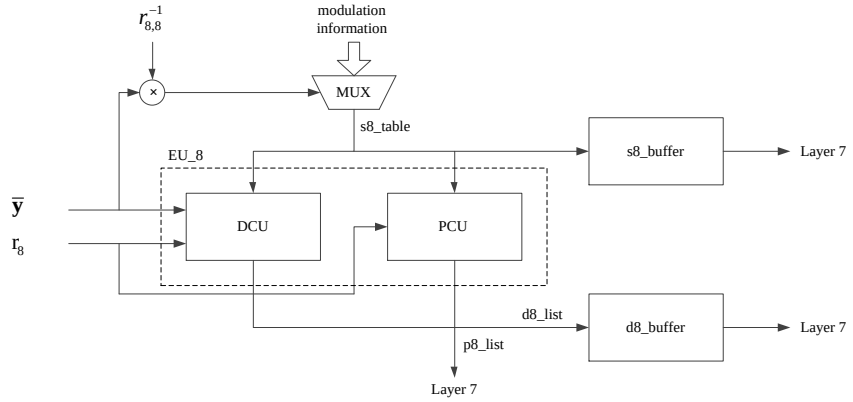


Figure 5.3: EU structure in Layer 8

5.2.1 Matrix Elements Control

The modified K-BEST MIMO detection algorithm, as implied in Chapter 2, is decided by two iteratively changed elements, the partial Euclidean distance and its parameter $\overline{y}(s_{D+1})$. Given the input information of the decomposed channel matrix and the received signals, the EU module is designed to generate the renewed partial Euclidean distance as well as the parameter $\overline{y}(s_{D+1})$ for the sorting and expanding process of the next layer. The structure of the EU module in Layer 8 is shown in Figure 5.3. According to Equation 2.13 $d_8^2 = d^2(\overline{y}_8, r_{8,8}s_8)$. Since the candidates have 4 different possible results, the partial Euclidean distances also have 4 different values. The outputs of LUT are a group of values that are used for computing the partial Euclidean distance and are transmitted to the next stage for generating the new $\overline{y}$, so in the next layer 4 EU modules is required for the parallel computing structure, as shown in Figure 5.4. As introduced by [17] there is a significant simplification for computing the squared partial Euclidean distance, which use the absolute value of the partial Euclidean distance instead of the squared value. This method decreases the complexity a lot for saving the resources consumed by multiplier.

In Figure 5.3 the EU block is composed by 2 parts, the distance computing unit (DCU) and the parameter computing unit (PCU). The inputs of the DCU block are $\overline{\mathbf{y}} = \mathbf{Q}^{\mathrm{T}}\mathbf{y}$, R_8, the 8th column of the upper triangle matrix, as well as the ordered candidate nodes

candidate s_D	implementation object	simplified implementation
-3	$r_{DD} \times s_D$	$-(r_{DD} + r_{DD} \ll 1)$
-1	$r_{DD} \times s_D$	$-r_{DD}$
1	$r_{DD} \times s_D$	r_{DD}
3	$r_{DD} \times s_D$	$r_{DD} + r_{DD} \ll 1$

Table 5.1: Multiplier implementation for 16-QAM modulation

generated through the LUT block. The function of DCU block is to calculate the increased Euclidean distance. which is described in Equation (2.9) for each layer. Since for the 8th layer with $M_T = M_R$, there is no root distance, the increased Euclidean distance is also the partial Euclidean distance. Equation (2.10) describes the details of the calculating process, performed by multiplication and subtraction. Since for a specific modulation system the candidate is chosen from a fixed set, i.e. in 16-QAM modulation the possible candidates after complex to real tranformation it is $\Lambda = \{-3, -1, 1, 3\}$, multiplication can be replaced by a carefully designed shift-accumulate process. The multiplication in the proposed 16-QAM system is implemented as follows:

The PCU module, which implements the operation of Equation (2.11), is designed to compute the new $\overline{\mathbf{y}}(s_8)$ that is used in the next Layer. The same way of simplified implementation is also obtained in the PCU module as in DCU. To make the PCU and DCU work in a pipelined structure, the outputs required in detection of the next layer should be buffered.

For Layer 7, as it accepts the expanded information from Layer 8, to make the structure fully parallel, for each possible expanded trail a DCU and PCU module is designed. It makes the complexity of the detection in Layer 7 become 4 times of that in Layer 8, as implied in Figure 5.4. Another additional part is the accumulation of the partial Euclidean distance.

After Layer 7, with the number of expanded nodes reaching the designed width K, a sorting module is added to sort the candidates according to their partial Euclidean distances. Every candidate involved in the sorting actually contains 3 parts: the expanded node, the partial Euclidean distance and the parameters generated with it which would be used for the further expanding in next layer. The sorting process should change all these three parts

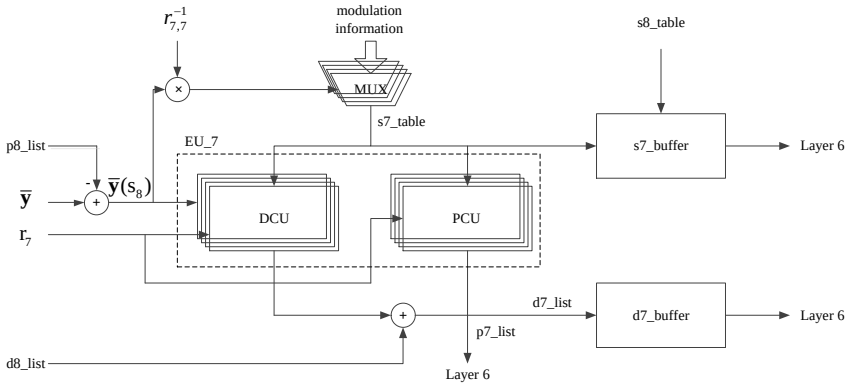


Figure 5.4: EU structure in Layer 7

at the same time to the corresponding positions in each of their new sequence.

5.2.2 Candidates Sorting and Selection

As introduced in Chapter 3, sorting process is the main bottleneck of the modified K-BEST MIMO detection. Figure 5.2 shows that in those layers where the number of candidates is larger than the designed width K, a sorting network is attached to choose the candidates that could match partial Euclidean distance requirement and be further expanded for next layer. Although the original ideal of sorting process is sorting all the expanded candidates globally, in Chapter 3 a group sorting algorithm is obtained to decrease the time consuming. Although this sorting algorithm decreases the BER performance as well compared with the system which obtain a full sorting algorithm, it dramatically increases the data throughput.

With $K = 16$ after Layer 8 and Layer 7 the sorting process is implemented. The group sorting algorithm divides the child nodes into different groups. As introduced in Section 3.2.1, the SE enumeration process, which is implemented by the LUT in Figure 5.3, 5.4 and 5.5, generates the child nodes for each of the parent nodes with a descending order from their partial Euclidean distance. In this design the length of the candidate list is $K = 16$. In layers where the detection has K parent nodes, $|\Lambda| \times K = 64$ child nodes are expanded, and using the group sorting theory they are divided into 8 groups. Within each group the

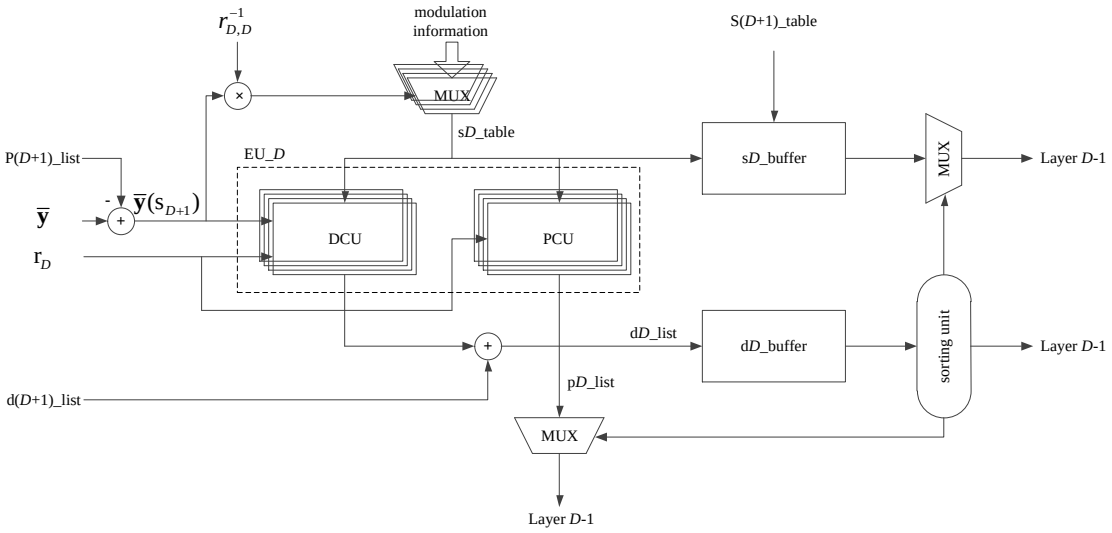


Figure 5.5: EU structure after Layer 7

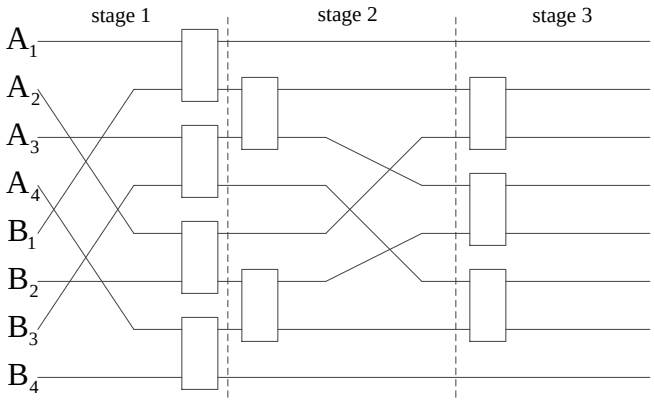


Figure 5.6: Pipelined merge sorting before interleaving

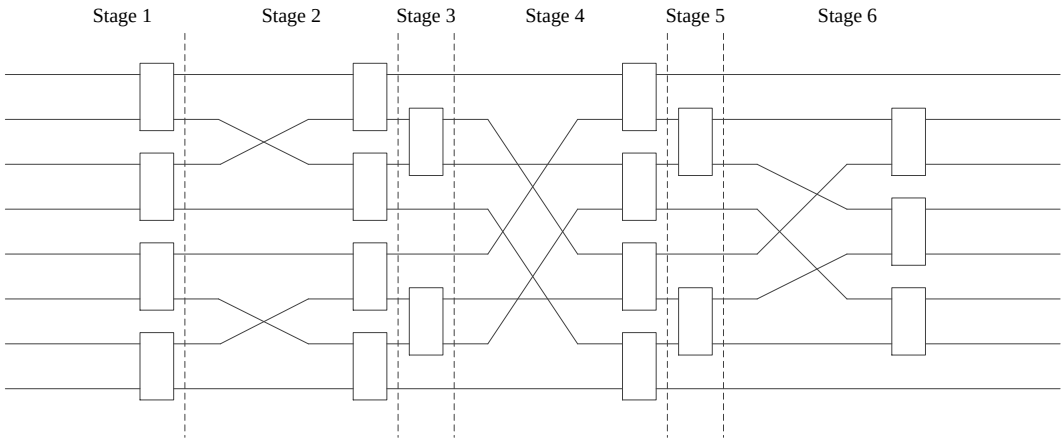


Figure 5.7: Pipelined sorting structure before interleaving

candidatesA_1, A_2, A_3, A_4 and B_1, B_2, B_3, B_4 are child nodes which are generated using SE enumeration by LUT. The sorting unit in the detection is composed by 3 parts: the odd-even merge sorting structure before interleaving, the interleaver and the sorting network after interleaving, as shown in Figure 5.6 and 5.7. The odd-even merge sorting, with a pipelined architecture, finishes the sorting process from two groups of sorted candidates as inputs and generates one group of sorted candidates as outputs in 3 clock cycles. The interleaving part, in hardware design, is composed by wires connecting the two sorting networks so there is no hardware resource consuming in this part. The sorting network after interleaving is modified from the odd-even sorting structure, since after interleaving the candidates in each group are all mixed up without any orders. Three stages are added to the odd-even merge sorting structure and the total sorting part consumes $3 + 9$ clock cycles in one layer.

An additional sorting network is required in the last layer after the normal sorting. After the normal sorting process which is the same as the one executed in the layers before it, a candidate list is generated, which is the $\mathbb{L}$ in Equation (4.19). For generating the hard-output, candidate in $\mathbb{L}$ with the smallest total Euclidean distance, and also for generating the soft-output (LLR) which is implemented according to Equation (4.19), a fully sorted candidate list is required. Since after the group sorting 2 ordered candidates are gained

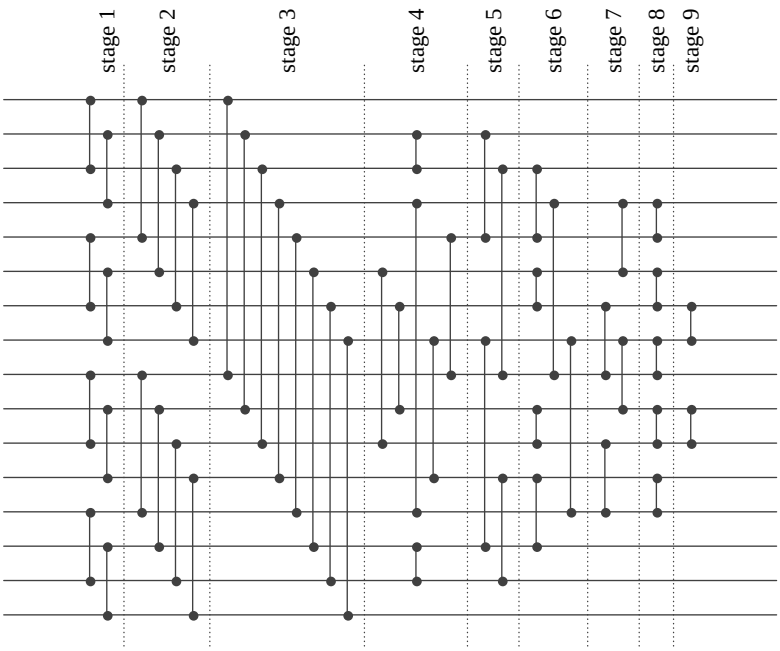


Figure 5.8: Pipelined sorting network for candidate list generation

inside each group, the additional sorting network is more complex than the 4×4 odd-even merge sorting network. Many computer science researchers have proposed relevant results and a pipelined sorting network is implemented in this thesis. The final sorting process requires 9 clock cycles to finish the whole sorting algorithm, as shown in Figure 5.8

After the additional sorting network in the final stage all the candidates in the list are in a descending order from their total Euclidean distance. The first one with the smallest distance is used as the hard input for outer decoder, or the final output in an uncoded system. However the LLR value is required for a soft-in soft-out decoder after the detection. The details of implementing LLR calculating algorithm as shown in Section 4.2, is introduced in the next section.

5.3 LLR Calculation Block

The research proposed in [8] shows that the *extrinsic* probability, which can be expressed in LLR value, as shown in Equation (4.19), is the most important soft input information for outer decoder. After the MIMO detection process, a candidate list is generated, as shown in Figure 5.1. In this section the LLR calculation proposed in Section 4.2 is implemented.

In Section 4.2, the candidate list generated by the MIMO detector contains 2 parts of information: 16 candidates in binary, each of which has 16 bits, and 16 candidates' Euclidean distances. After passing the sorting network proposed in Figure 5.8, all those candidates are sorted. According to the LLR value of the extrinsic probability expressed in Equation (4.19), the implementation is complex because the 2 candidates, each of which has a value of 0 and 1, respectively, in the specific bit with the smallest Euclidean distance, need to be obtained and totally 16 LLR values are required. This process can be obtained by performing exhaustive search in the candidate list for every bit. However since the candidates are all sorted the first candidate contains half of the information of those 16 LLR values expressed in Equation (4.19), either the '1' part-$\mathbb{L}_{mn,+1}$-or the '-1' part ('0' part)-$\mathbb{L}_{mn,+1}$.

The LLR computing architecture is designed to find the rest half information of those LLR values. Since the value of each bit of the sorted candidates can not be predicted, to make the computing process parallel, 15 status finite state machine is applied in this process:

The **XOR** and computing process is expressed in Figure 5.9. A pipelined structure is applied to the LLR value computing module and the structure is shown in Figure 5.10. For each clock cycle the detector generates the candidate list and transmits it to one block of the structure. After 15 clock cycle the first LLR block finishes the computing process and a LLR value for the first 16 bits is gained, while a new group of candidate list is transmitted into that block and new cycle begins.

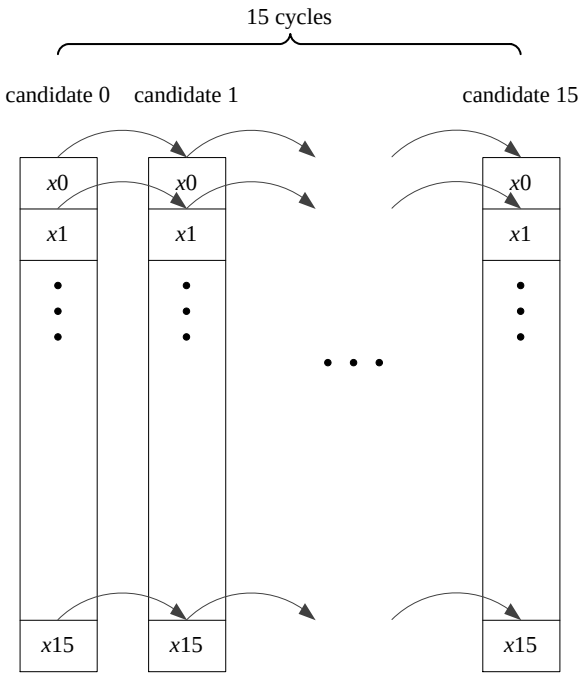


Figure 5.9: Block diagram for LLR calculation

Algorithm 5.1 LLR value computing process

Input: 16 candidates $cand[0] \sim cand[15]$ and 16 Euclidean distance $d[0] \sim d[15]$, SNR.

STATE 0. **XOR** each bit of $cand[1]$ and the respective bit of $cand[0]$. If the result is 1 and the LLR of that bit is not set then set the LLR value according to the Equation 4.19, goto next state. Else goto next set directly.

... ...

STATE n. **XOR** each bit of $cand[n+1]$ and the respective bit of $cand[0]$. If the result is 1 and the LLR of that bit is not set then set the LLR value according to the Equation 4.19, goto next state. Else goto next set directly.

... ...

STATE 14. **XOR** each bit of $cand[15]$ and the respective bit of $cand[0]$. If the result is 1 and the LLR of that bit is not set then set the LLR value according to the Equation 4.19, goto next state. Else set a fixed value to the LLR of that bit according to the respective bit of $cand[0]$.

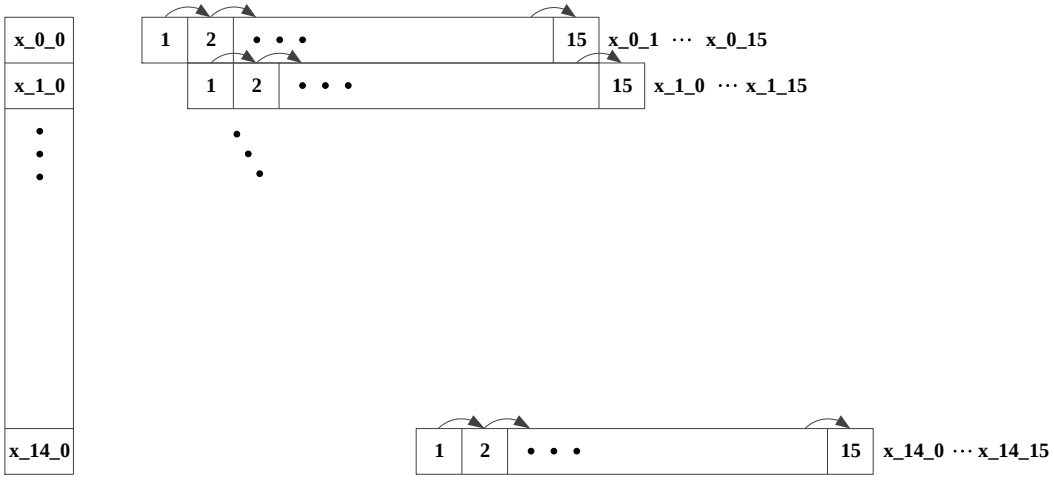


Figure 5.10: Pipelined LLR structure

5.4 Resource and Delay Analysis

The 4×4 MIMO detection process with 16-QAM modulation is divided into 8 layers. The channel matrix $\mathbf{H}$ changes much slower, compared with the speed of the transmitted data. That means the QR decomposition, as well as the related parameters like r_{DD}^{-1} and the SNR for the LLR computing part, could be processed by an outer processor. With a specific modulation all the parameters involved in the multiplication and division are fixed and so those operation could be designed and implemented to be finished in one clock cycle by shifting and accumulation.

With $K = 16$ after Layer 8 and Layer 7 the sorting process is required. For Layer 8 and Layer 7, parallel structures are designed for EU and LUT, as introduced in Section 5.2. The LUT, as well as its input which is involved with a multiplication, takes one clock cycle, and the EU, which contains the DCU and PCU takes two clock cycles to get the final result. So 3 clock cycles are needed for each layer of Layer 8 and Layer 7. After that a sorting process is added. As introduced in Section 5.2.2 $3 + 6 = 9$ clock cycles are needed for the interleaving-sorting process, which leads $9 + 3 = 12$ clock cycles for every layer from Layer 6 to Layer 2. In Layer 1 since the hard-output as well as the input for soft information computing part is implemented, an additional sorting process is designed to make its output fully sorted from the Euclidean distances. As shown in Figure 5.8, 9 clock cycles are needed for this additional sorting. For the LLR computing unit 15 clock cycles are required to compute LLR values. In conclusion, the proposed structure uses $3 \times 2 + (3 + 3 + 6) \times 5 + (3 + 3 + 6 + 9) + 15 = 102$ clock cycles to obtain the soft information of 16 bits data. Although the delay is long, with the pipelined structure introduced in this chapter the proposed MIMO detection system is able to generate the hard-output and soft-output continuously.

The whole system is implemented into Xilinx 6 FPGA. Table 5.2 shows the hardware resource usage of the implementation result. The directly implementation that use distributed registers and large amount of buffers makes the hardware consuming large. A modification that use the embedded memory block should be able to minimize the hardware resources.

The detection and soft information generation system implemented into Xilinx 6 is able

Number of Slice Registers	178245
Number of Slice LUTs	225566
Number of fully used LUT-FF pairs	136120
Number of bonded IOs	747
Number of BUFG/BUFGCTRLs	2

Table 5.2: Implementation result on Xilinx Virtex6 FPGA

to run at a frequency of 230.132 MHz. Since after the initial delay introduced before the detection results are generated continuously, the data throughput of the designed system can achieve at 3.7 Gbps. It is the fastest MIMO detection so far as the thesis is composed due to its highly parallel and pipelined architecture.

5.5 Conclusion

In this chapter the detailed hardware structure of the proposed MIMO detection algorithm that can generate both the hard-output and soft-output for outer decoder is introduced. The parallel architecture, like the parallel sorting network and expanding/computing process with a pipelined design consumes a large amount of hardware resources. However a high data throughput, around 3.7 Gbps, is also achieved. This makes the proposed MIMO detection system the fastest one so far as the thesis is composed.

Chapter 6

Conclusions and Future work

This thesis introduced a MIMO detection method which is able to generate both the hard and soft-output at a data throughput of 3.7 Gbps, which is the fastest MIMO detection implementation so far as the thesis is composed. This method is modified from a parallel MIMO detection method which obtains the breadth first search as its principle searching strategy. Based on this strategy, a parallel group sorting structure is introduced with interleaving which decreases the possibility of a skew process that might happen inside each group and also minimizes the decrease of the BER performance of this system.

In this design a preprocessing stage is performed every time the channel and the noise level changes. This preprocess is designed to be performed by an outer processor, which has a lower speed compared with the specific designed parallel detection part. However since the data throughput is relatively higher than the frequency of the channel and noise level changing, the outer processor should still be able to cooperate with the detection process properly.

6.1 Future Work

So far only convolutional code is tested with the proposed detection system. The soft-output is generated in a detection without any iterative process. However, with a *priori* probability information, iteratively detection can also be achieved. In the future some other channel code with different outer decoder, like Turbo code decoder and LDPC decoder

can be integrated into the detection system. The soft information can be generated by an iterative detection process combined with the decoder to achieve a better BER performance.

Another future modification is about the hardware resources. With the pipelined structure the distributed buffers make the detection system complex and occupied a large area. Since the frequency of the detection is around 200Mhz an integrated memory block should be able to handle all the datas stored in the buffers. Much more future work should be finished in the hardware implementation stage to optimize the chip area and energy consuming of the detection system.

Bibliography

[1] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, *Closest point search in lattices*, **48** (2002), no. 8, 2201–2214.

[2] L.G. Barbero and J.S. Thompson, *Performance analysis of a fixed-complexity sphere decoder in high-dimensional MIMO systems*, Proc. IEEE ICASSP06, 557–560.

[3] S. Bittner, E. Zimmermann, W. Rave, and G. Fettweis, *List sequential mimo detection: Noise bias term and partial path augmentation*, Proc. IEEE International Conference on Communications, vol. 3, June 2006, pp. 1300–1305.

[4] M. O. Damen, H. El Gamal, and G. Caire, *On maximum-likelihood detection and the search for the closest lattice point*, **49** (2003), no. 10, 2389–2402.

[5] O. Damen, A. Chkeif, and J.-C. Belfiore, *Lattice code decoder for space-time codes*, **4** (2000), no. 5, 161–163.

[6] U. Fincke and M. Pohst, *Improved methods for calculating vectors of short length in a lattice, including a complexity analysis*, Mathematics of computation **44** (1985), no. 170, 463–471.

[7] A. Goldsmith, S. A. Jafar, N. Jindal, and S. Vishwanath, *Capacity limits of mimo channels*, **21** (2003), no. 5, 684–702.

[8] Zhan Guo and P. Nilsson, *Algorithm and implementation of the k-best sphere decoding for mimo detection*, **24** (2006), no. 3, 491–503.

[9] J. Hagenauer and C. Kuhn, *The List-Sequential (LISS) algorithm and its application*, IEEE Transactions on Communications **55** (2007), no. 5, 918.

[10] B. Hassibi and H. Vikalo, *On the sphere-decoding algorithm i. expected complexity*, **53** (2005), no. 8, 2806–2818.

[11] B. M. Hochwald and S. ten Brink, *Achieving near-capacity on a multiple-antenna channel*, **51** (2003), no. 3, 389–399.

[12] B. Kim and IC Park, *K-best MIMO detection based on interleaving of distributed sorting*, Electronics Letters **44** (2008), no. 1, 42–42.

[13] D. Le Ruyet, T. Bertozzi, and B. Ozbek, *Breadth first algorithms for app detectors over mimo channels*, Proc. IEEE International Conference on Communications, vol. 2, June 20–24, 2004, pp. 926–930.

[14] Q. Li and Z. Wang, *Reduced Complexity K-Best Sphere Decoder Design for MIMO Systems*, Circuits, Systems, and Signal Processing **27** (2008), no. 4, 491–505.

[15] P. Luethi, A. Burg, S. Haene, D. Perels, N. Felber, and W. Fichtner, *VLSI implementation of a high-speed iterative sorted MMSE QR decomposition*, (2007), 1421–1424.

[16] CP Schnorr and M. Euchner, *Lattice basis reduction: Improved practical algorithms and solving subset sum problems*, Mathematical programming **66** (1994), no. 1, 181–199.

[17] C. Studer, M. Wenk, A. Burg, and H. Bolcskei, *Soft-output sphere decoding: Performance and implementation aspects*, Proc. Fortieth Asilomar Conference on Signals, Systems and Computers ACSSC '06, October 2006, pp. 2071–2076.

[18] K. Su and I.J. Wassell, *Efficient MIMO detection by successive projection*, (2005).

[19] M. Wenk, M. Zellweger, A. Burg, N. Felber, and W. Fichtner, *K-best MIMO detection VLSI architectures achieving up to 424 Mbps*, (2006), 1151–1154.

[20] A. Wiesel, X. Mestre, A. Pages, and J. R. Fonollosa, *Efficient implementation of sphere demodulation*, Proc. 4th IEEE Workshop on Signal Processing Advances in Wireless Communications SPAWC 2003, June 15–18, 2003, pp. 36–40.

[21] Xiaojun Yuan, Qinghua Guo, and Li Ping, *Low-complexity iterative detection in multi-user mimo isi channels*, **15** (2008), 25–28.