High Throughput VLSI Architecture for Soft-Output MIMO Detection Based on A Greedy Graph Algorithm [*]

Yang Sun
Depart. of Electrical and Computer Engineering
Rice University
6100 Main Street, Houston, TX 77005
ysun@rice.edu

Joseph R. Cavallaro
Depart. of Electrical and Computer Engineering
Rice University
6100 Main Street, Houston, TX 77005
cavallar@rice.edu

ABSTRACT

Maximum-likelihood (ML) decoding is a very computational-intensive task for multiple-input multiple-output (MIMO) wireless channel detection. This paper presents a new graph based algorithm to achieve near ML performance for soft MIMO detection. Instead of using the traditional tree search based structure, we represent the search space of the MIMO signals with a directed graph and a greedy algorithm is applied to compute the *a posteriori* probability (APP) for each transmitted bit. The proposed detector has two advantages: 1) it keeps a fixed throughput and has a regular and parallel datapath structure which makes it amenable to high speed VLSI implementation, and 2) it attempts to maximize the *a posteriori* probability by making the locally optimum choice at each stage with the hope of finding the global minimum Euclidean distance for every transmitted bit $x_k \in \{-1, +1\}$. Compared to the soft K-best detector, the proposed solution significantly reduces the complexity because sorting is not required, while still maintaining good bit error rate (BER) performance. The proposed greedy detection algorithm has been designed and synthesized for a 4×4 16-QAM MIMO system in a TSMC 65 nm CMOS technology. The detector achieves a maximum throughput of 600 Mbps with a 0.79 mm^2 core area.

Categories and Subject Descriptors

B.7.1 [**Types and Design Styles**]: Algorithms implemented in hardware

General Terms

Algorithm, Design, Performance

Keywords

MIMO Detection, VLSI Architecture, ASIC Design

*This work was supported in part by Nokia Corporation, and by NSF under grants CCF-0541363, CNS-0551692, and CNS-0619767.

1. INTRODUCTION

Multiple-input multiple-output (MIMO) communication systems have received tremendous attention because of their high spectral efficiency and near-capacity performance. New wireless standards, such as IEEE 802.11n, IEEE 802.16e WiMax, and 3GPP LTE, include MIMO techniques in combination with advanced outer channel codes such as low-density parity-check (LDPC) codes [3] and Turbo codes [1]. The main challenge of the soft MIMO detection is to efficiently and accurately generate the log-likelihood radios (LLRs) for the outer channel decoder. The exhaustive search with the ML criterion will consume enormous computing power and require tremendous silicon resources on the chip which makes it impossible to be employed in multiple antenna systems with higher-order modulation schemes.

To reduce the exponentially algorithmic complexity, some sub-optimal detection algorithms and their VLSI architectures have been proposed by researchers recently[4, 2, 10, 5, 7, 9]. Garret *et al.* [4] implemented a depth-first soft sphere decoding (SD) algorithm with 256 search operations at each level of the tree. Burg *et al.* [2] presented a simplified hard sphere decoding ASIC architecture. On the other hand, Wong *et al.* [10] first introduced the breadth-first K-best hard detection algorithm. Later on, Guo *et al.* [5] extended it for the soft K-best (K=5) detection by keeping a list of best candidates at each search tree level.

The depth-first SD algorithm has non-deterministic complexity and variable throughput which makes it sensitive to the channel conditions. The depth-first SD with a small candidate list size suffers significant performance degradation due to the inaccurate and especially the infinite log likelihood ratios (LLRs). On the other hand, the K-best algorithm has advantages of fixed complexity and fixed throughput, which makes it more friendly for hardware implementation. However, when K is large, the complexity of the K-best algorithm dramatically increases because a large number of paths have to be extended and sorted.

In this paper, a greedy shortest path searching algorithm and its VLSI architecture is proposed for high throughput soft MIMO detection. We transform the traditional MIMO detection problem into a shortest path finding problem. By making a locally optimum choice at each stage, this algorithm tries to find the global minimum Euclidean distance for every transmitted bit. Therefore it avoids the LLR clipping issues that both depth-first SD and K-best detectors have. Moreover, this approach is very suitable for high speed VLSI implementation because of the regular and parallel datapath structure.

2. SYSTEM MODEL

We consider a coded MIMO system with M transmit antennas and N receive antennas. The MIMO transmission can be modeled as:

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

where $\mathbf{H}$ is an $N \times M$ complex matrix, $\mathbf{s} = [s_0, ..., s_{M-1}]^T$ is an $M \times 1$ transmitted vector, $\mathbf{y}$ is an $N \times 1$ received vector, and $\mathbf{n}$ is an independent identically distributed complex zero-mean Gaussian noise vector with variance σ^2 per direction. The symbol vector $\mathbf{s}$ is obtained using the mapping function $s_m = \text{map}(\mathbf{x}^{<\mathbf{m}>}), m = 0, ..., M-1$, where $\mathbf{x}^{<\mathbf{m}>}$ is an $M_c \times 1$ vector (block) of transmitted data bits ($\mathbf{x}^{<\mathbf{m}>} = [x_0, x_1, ..., x_{M_c-1}]^T$), and M_c is the number of bits per constellation symbol. For instance, $M_c = 2$, $s \in \{1+j, 1-j, -1+j, -1-j\}$ in the case of QPSK, and $M_c = 4$, $s \in \{\pm1\pm j, \pm1\pm3j, \pm3\pm j, \pm3\pm3j\}$ in the case of 16-QAM. The concatenating $\mathbf{x}^{<\mathbf{0}>}, ..., \mathbf{x}^{<\mathbf{M-1}>}$ is written in $\mathbf{x}$. The soft-output detector is to compute the APP value of the bit x_k, for $k = 0, ..., M \cdot M_c - 1$. The APP is usually expressed as a log-likelihood ratio value (L-value):

$$L(x_k|\mathbf{y}) = \ln \frac{P[x_k = +1|\mathbf{y}]}{P[x_k = -1|\mathbf{y}]} = L_A(x_k) + L_E(x_k), \tag{2}$$

where L_A and L_E denote the *a priori* L-value and extrinsic L-value, respectively. Assuming there is no prior knowledge of the transmitted signal, using the max-log approximation [6], (2) can be simplified to

$$L(x_k|\mathbf{y}) \approx \frac{1}{2\sigma^2}\left(\min_{\mathbf{x}\in\mathbb{X}_{k,-1}} \Lambda(\mathbf{s},\mathbf{y}) - \min_{\mathbf{x}\in\mathbb{X}_{k,+1}} \Lambda(\mathbf{s},\mathbf{y})\right), \tag{3}$$

where set $\mathbb{X}_{k,+1} = \{\mathbf{x}|x_k = +1\}$ and set $\mathbb{X}_{k,-1} = \{\mathbf{x}|x_k = -1\}$. Using QR decomposition according to $\mathbf{H} = \mathbf{QR}$, where $\mathbf{Q}$ and $\mathbf{R}$ refer to an $N \times M$ unitary matrix and an $M \times M$ upper triangular matrix, respectively, $\Lambda(\mathbf{s},\mathbf{y})$ can be computed as

$$\Lambda(\mathbf{s},\mathbf{y}) = \|\mathbf{y} - \mathbf{H}\cdot\mathbf{s}\|^2 = \|\hat{\mathbf{y}} - \mathbf{R}\cdot\mathbf{s}\|^2 + C, \tag{4}$$

where $\hat{\mathbf{y}} = \mathbf{Q}^H\mathbf{y}$, and C is a constant ($C = 0$ if $M = N$), which does not affect (3).

3. GREEDY DETECTION ALGORITHM

Without loss of generality, we use a 4×4 QPSK system to explain our proposed algorithm in this section.

3.1 Graph construction

The goal of the soft MIMO detector is to generate the LLR value for each transmitted bit x_k based on (3), which requires the calculation of the minimum Euclidean distance

$$\Lambda = \left\| \begin{bmatrix} \hat{y}_0 \\ \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \end{bmatrix} - \begin{bmatrix} R_{00} & R_{01} & R_{02} & R_{03} \\ 0 & R_{11} & R_{12} & R_{13} \\ 0 & 0 & R_{22} & R_{23} \\ 0 & 0 & 0 & R_{33} \end{bmatrix} \begin{bmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \end{bmatrix} \right\|^2, \tag{5}$$

over sets $\mathbb{X}_{k,+1}$ and $\mathbb{X}_{k,-1}$. The calculation of Λ can be decomposed as: $\Lambda = w^{<0>} + w^{<1>} + w^{<2>} + w^{<3>}$, where $w^{<t>}$ is the 1-D Euclidean distance and is calculated as

$$\begin{aligned} w^{<0>} &= \|\hat{y}_3 - R_{33}s_3\|^2, \\ w^{<1>} &= \|\hat{y}_2 - (R_{22}s_2 + R_{23}s_3)\|^2, \\ w^{<2>} &= \|\hat{y}_1 - (R_{11}s_1 + R_{12}s_2 + R_{13}s_3)\|^2, \\ w^{<3>} &= \|\hat{y}_0 - (R_{00}s_0 + R_{01}s_1 + R_{02}s_2 + R_{03}s_3)\|^2. \end{aligned} \tag{6}$$

This process can be viewed using a flow graph which is shown in Figure 1. The vertices are ordered into 4 vertical slices or stages. Stages 0, 1, 2, 3 represent antennas 3, 2, 1, 0 respectively. In each stage, there are $Q = 2^{M_c}$ vertices. Each vertex represents a possible complex constellation point. Each vertex at each stage is connected to Q vertices at an earlier and Q vertices at a later stage. The vertex in stage t is represented as $v(t,i)$ $(0 \le i \le Q-1)$. The edge between $v(t-1,i)$ and $v(t,j)$ has a weight of $w_{i,j}^{<t>}$. Because of the upper triangular property of $\mathbf{R}$, from (6) we know that the weight function dose not depend on the future stages, but only depends on its current stage and all its predecessors. For instance, $w_{i,j}^{<0>}$ only depends on the vertices in the stage 0, while $w_{i,j}^{<2>}$ depends on the vertices in stages 2, 1, and 0. The edges connected to the dummy terminus node have 0 weights.

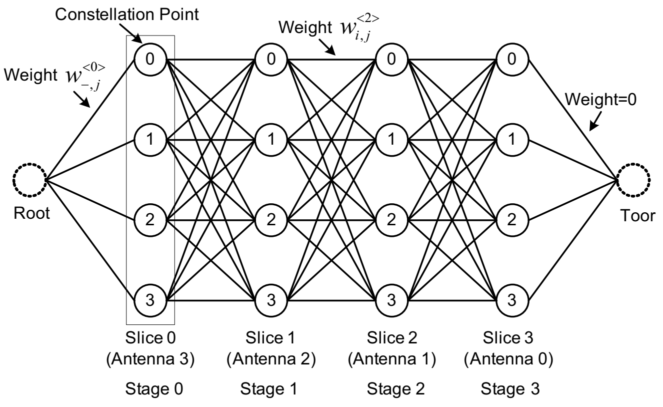


Figure 1: Flow graph for MIMO detection

3.2 Problem definition

Given a received, possibly noisy MIMO symbol, we may associate the 1-D Euclidean distance with weights on each edge in the graph so that the problem of ML detection reduces to the problem of finding the minimum-weight path from the root to the toor in the graph.

Definition 1: Hard MIMO detection problem:
Find the shortest path from the root to the toor. Then the encountered vertices are the detected MIMO signals.

Definition 2: Soft MIMO detection problem:
For each vertex i $(0 \le i \le Q-1)$ in the stage t $(0 \le t \le M-1)$, find the shortest path, which must contain this vertex, from the root to the toor. The Q conditioning shortest paths found at every stage t make a candidate list $\mathcal{L}_t$. Then the L-value of bit $x_i^{<t>}$ is calculated as:

$$L(x_i^{<t>}|\mathbf{y}) = \frac{1}{2\sigma^2}\left(\min_{\mathbf{x}\in\mathcal{L}_{t,-1}} \Lambda - \min_{\mathbf{x}\in\mathcal{L}_{t,+1}} \Lambda\right). \tag{7}$$

3.3 Greedy algorithm

The optimum solution to the hard detection problem requires full search over the entire graph whose complexity grows exponentially with the size of the graph. Solving the soft detection problem is even harder since it needs to repeatedly perform full search on the condition of every vertex being included in this shortest path. In this section, we will introduce a greedy shortest path algorithm to approximately solve the soft detection problem. Like the K-best algorithm, it takes decisions on the basis of information at hand without worrying about the effect these decisions may have in the future.

Step 1. Edge reduction
In Figure 1, each vertex i at each stage t (except for the first and last stages), is connected to Q vertices at an earlier stage and Q vertices at a later stage. Figure 2 shows the data flow at each vertex which has Q incoming subpaths $h_0, ..., h_{Q-1}$ and Q outgoing subpaths $h'_0, ..., h'_{Q-1}$.

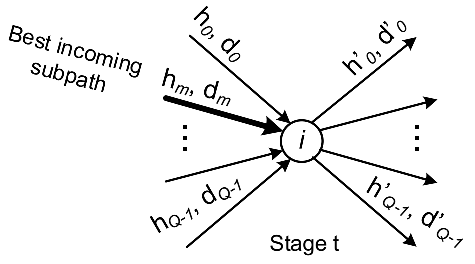


Figure 2: Data flow at vertex $v(t, i)$

To reduce the arithmetic complexity, a greedy algorithm is summarized as follows. Let the partial distance be d_k, which is the cumulative weight of the subpath h_k from the root to this vertex i. Among the Q incoming subpaths, we select the best subpath h_m with the minimum weight

$$m = \underset{m \in \{0,...,Q-1\}}{\operatorname{argmin}} d_m, \tag{8}$$

and discard the other $Q - 1$ subpaths. After choosing the best subpath h_m, the outgoing subpath to vertex $v(t+1, k)$ is updated as

$$h'_k = \{h_m, k\},\ 0 \le k \le Q - 1. \tag{9}$$

The outgoing path weight to vertex $v(t+1, k)$ is updated as

$$d'_k = d_m + w_{i,k}^{<t+1>},\ 0 \le k \le Q - 1, \tag{10}$$

where the weight function $w_{i,k}^{<t+1>}$ is calculated based on the vertices along the subpath h'_k according to (6). Moreover, among the Q outgoing subpaths we also find the shortest subpath h'_n where

$$n = \underset{n \in \{0,...,Q-1\}}{\operatorname{argmin}} d'_n. \tag{11}$$

This information will be stored in memory for later use. Figure 3 shows an example of the result graph after applying the edge reduction operation. Note that only the surviving path for each vertex is shown in Figure 3. We can see that each vertex in stage 3 is along a path to the end point (toor). These paths are presumably the shortest and can be used to form the candidate list $\mathcal{L}_3$. However not every vertex in the stages 0, 1, and 2 is along a path to the end point. To solve this issue, we need to perform a path extension operation.

Step 2. Path extension
A. Path extension for stage 2
In Figure 3, if we look at the vertices on stage 2, we will see that not every vertex is along a path to the toor. For example, vertices 0 and 3 are disconnected with the toor. Therefore, we need to extend those uncompleted paths. The **path extension algorithm** is summarized as follows. *Extend each subpath by checking all its Q outgoing edges and select the edge which leads to the shortest cumulative subpath weight. This process will be repeated in a greedy manner until it reaches the toor.* Figure 4 illustrates one level path extension operation for $v(t, i)$.

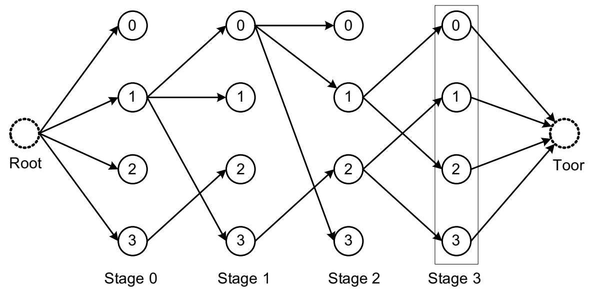


Figure 3: Edge reduction result

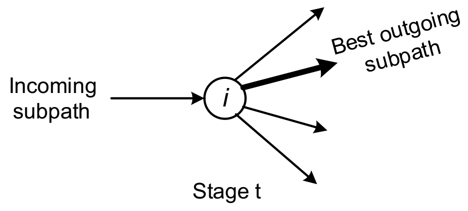


Figure 4: Path extension operation: select the best outgoing subpath

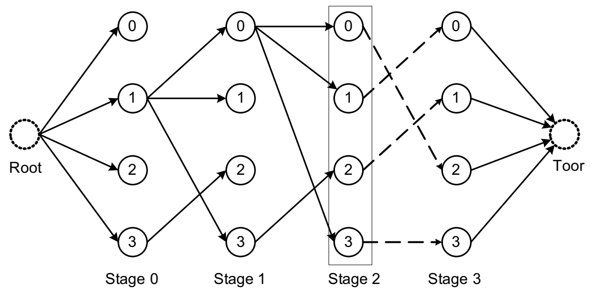
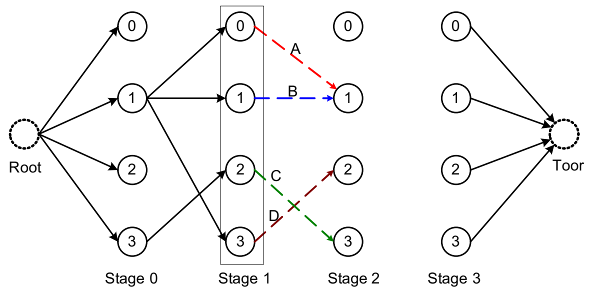


Figure 5: Path extension result for stage 2

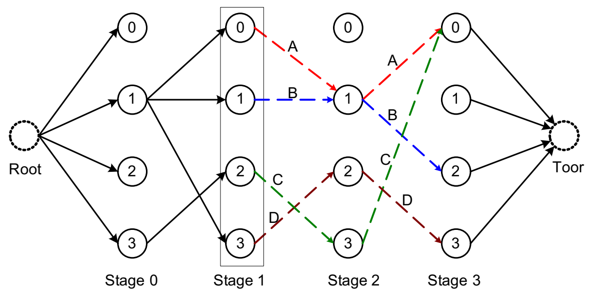
Recall that in the step of edge reduction, we have saved the shortest outgoing subpath h'_n for each vertex into memory. If we retrieve this information for each vertex in stage 2, Figure 5 shows the result graph. Here the dotted lines represent the outgoing edges retrieved from the memory. Now each vertex in stage 2 is along a path to the toor (presumably the shortest). So the candidate list $\mathcal{L}_2$ can be created in this way.

B. Path extension for stage 1
Similarly, in Figure 3, not every vertex in stage 1 is along a path to the toor. By retrieving the shortest outgoing subpath from memory for each vertex in the stage 1, we could extend these subpaths for one level as shown in Figure 6(a), where the four extended subpaths are labeled as A, B, C, and D. It is worth to mention that no re-computing, but memory read is needed for the first step of the path extension operation. However the second level path extension operation needs re-computing and comparing the subpath weights as described in the path extension algorithm. The result after applying a second level path extension is shown

in Figure 6(b). Now the subpaths {A, B, C, D} have been fully extended and can be used to form the list $\mathcal{L}_1$.



(a) Step 1: Retrieve paths from memory



(b) Step 2: Re-computing the best path for the next stage

Figure 6: Path extension for stage 1

B. Path extension for stage 0

Similarly, by applying the path extension algorithm repeatedly, Figure 7 shows the path extension result for stage 0.

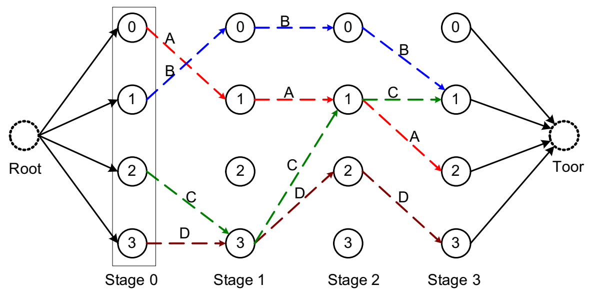


Figure 7: Path extension result for stage 0

Step 3. LLR calculation

After all the candidate lists $\mathcal{L}_t$ for $0 \le t \le M-1$ have been created, the LLR calculation defined in (7) is then very straightforward.

3.4 Algorithm complexity analysis

In the proposed greedy algorithm, calculating the partial Euclidean distance is the major contribution to the total arithmetic complexity. We only consider this part in the complexity analysis and ignore the other minor contributors such as minimization and memory operations. More generally, consider an M transmit antenna system with Q-size QAM modulation. In the edge reduction operation, the number of subpath weights that need to be calculated per stage is Q^2, so the total complexity is $\mathcal{O}(MQ^2)$. On the other hand, the complexity of the path extension operation is $\mathcal{O}(\frac{1}{2}(M-1)(M-2)Q^2)$. In a practical MIMO system, M is usually not very large. In the case of $M = 4$, the total arithmetic complexity of this greedy algorithm is approximately $\mathcal{O}(7Q^2)$.

4. SIMULATION RESULTS

Like the K-best algorithm, the proposed greedy algorithm has a deterministic complexity and a fixed throughput. To evaluate the decoding BER performance, we have compared the proposed greedy algorithm with the traditional K-best algorithm. We consider 4×4 16-QAM and 64-QAM MIMO systems (the channel matrices are assumed to have independent Rayleigh fading distribution). In the simulation, the soft-output of the detector is fed to a length 2304, rate 1/2 WiMax LDPC decoder [8], which performs up to 15 iterations. Figure 8 compares the BER performance for the proposed MIMO detector. For the 4×4 16-QAM system, our detector outperforms the K-best detector for K=16 and 32, and achieves similar performance compared with K=64. For the 4×4 64-QAM system, our detector outperforms the K-best detector with K=32, 48 and 64.

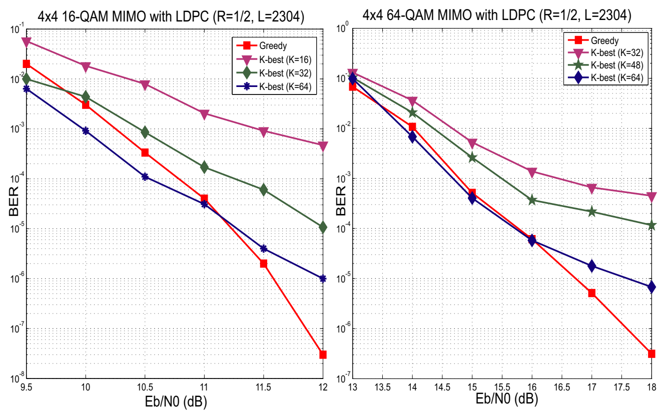
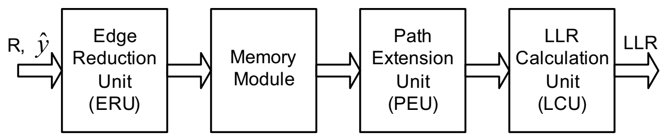


Figure 8: BER performance comparison

5. VLSI ARCHITECTURE DESIGN

In this section, we will describe the hardware architecture design for a 4×4 16-QAM MIMO system. The greedy algorithm introduced in section 3 is generic and can be easily extended for higher order modulation systems, i.e. in the case of 16-QAM, there are 16 vertices instead of 4 at each stage. Figure 9 shows the top level hardware architecture for implementing the proposed greedy detection algorithm. It contains four major units: Edge Reduction Unit (ERU), Memory Module, Path Extension Unit (PEU), and LLR Calculation Unit (LCU).



R, $\hat{y}$
Edge Reduction Unit (ERU)
Memory Module
Path Extension Unit (PEU)
LLR Calculation Unit (LCU)
LLR

Figure 9: Top level architecture

5.1 Edge reduction unit (ERU)

We define the subpath metric (SM) for vertex $v(t, i)$ as the cumulative path weight (or partial Euclidean distance) up to this vertex $v(t, i)$. In the step of edge reduction, each vertex will compare the Q incoming SMs and select the edge with the minimum SM, and prune the other $Q-1$ edges. Then the Q outgoing SMs are computed by adding the corresponding edge weight to the surviving incoming subpath weight and sent to the downstream vertices.

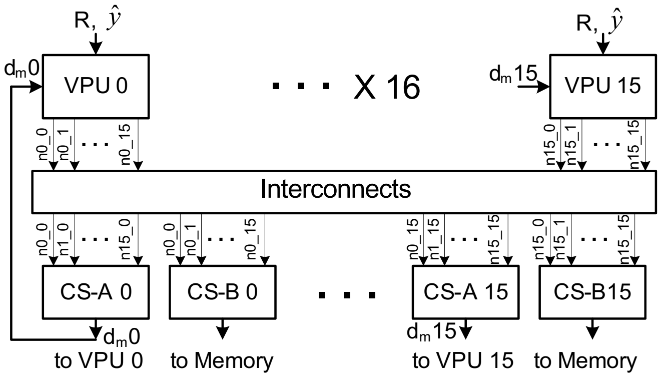


Figure 10: ERU architecture

Figure 10 illustrates the ERU architecture, where VPU stands for Vertex Processing Unit, and CS stands for Compare and Select. This is a partially-parallel architecture by having Q vertices being processed simultaneously. This is also a recursive architecture by reusing the logic for different stages. Note there are two CS units: CS-A and CS-B. CS-A unit is used to select the minimum incoming SMs and pass the survivor to the VPU in the next iteration. CS-B unit is used to select the shortest outgoing SM for each vertex and save it to memory for path extension operation. Figure 11 and Figure 12 show the VPU and CS architecture, respectively. The VPU unit is used to calculate the outgoing SMs (partial Euclidean distances). At stage t $(-1 \leq t \leq M-2)$, let $i = M-2-t$, the k-th $(0 \leq k \leq Q-1)$ outgoing SM is updated as:

$$\begin{aligned} d_k^{<t+1>} &= d_m^{<t>} + ||T + R_{i,i}S_i||^2 \\ T &= \sum_{j=i+1}^{M-1} R_{i,j}S_j - \hat{y}_i, \end{aligned} \tag{12}$$

where S_i is the complex constellation point for antenna i, the antennas are numbered 3, 2, 1, and 0, which correspond to stages 0, 1, 2, and 3 respectively. The values of $S_{i+1}, ...S_{M-1}$ are obtained based on the vertices encountered in the incoming subpath h_m. The incoming subpath metric $d_m^{<t>}$ is generated by the CS-A unit ($d_m^{<-1>}$ is initialized to 0). The SADD in Figure 11 stands for shift and add which is used for implementing $R * S$.

5.2 Path extension unit (PEU)

The function of the PEU is to extend the subpaths, which were obtained in the edge reduction step, in a greedy and recursive fashion. Both memory read and path re-computation are required for stage 0 and 1. Only memory read is required for stage 2. Figure 13 shows the PEU architecture which contains 16 VPUs and 16 CSs such that 16 subpaths can be

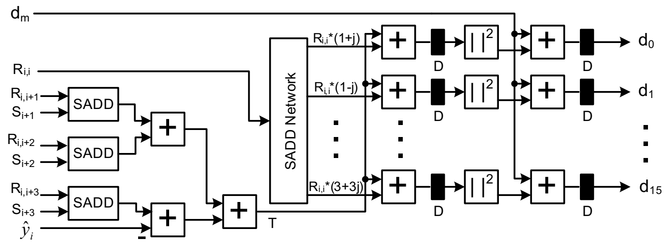


Figure 11: VPU architecture

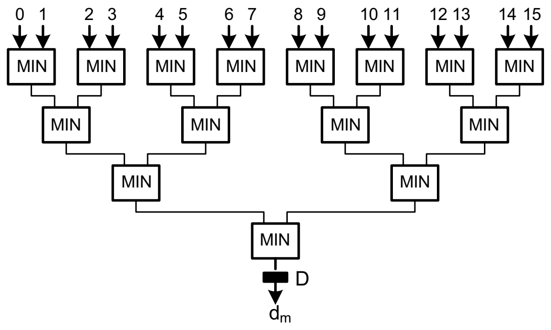


Figure 12: CS architecture

extended in parallel. The initial subpath metric d_m is read from the memory which contains the results from the edge reduction operation.

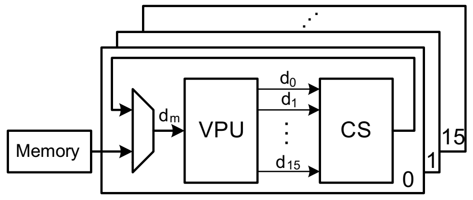


Figure 13: PEU architecture

5.3 LLR calculation unit (LCU)

After obtaining the list $\mathcal{L}_t$, for $0 \leq t \leq M-1$, the LCU implements (7) in a straightforward way. The detail implementation is omitted.

5.4 Hardware scheduling

Figure 14 shows the timing diagram for a 4×4 16-QAM MIMO system. Each stage of the edge reduction takes three cycles to finish, so it takes $3 \times 4 = 12$ cycles to perform the edge reduction operation. The path extension operation for antenna 3 can start 6 cycles later and will take 6 cycles to finish. The path extension operation for antenna 2 will take another 3 cycles. So the total latency is 15 cycles. Taking into account the two extra cycles of latency for LLR generation, the total decoding latency for one MIMO symbol is 17 cycles. In terms of throughput, because two consecutive MIMO symbols can be overlapped as shown in Figure 14, the decoding throughput for a 4×4 16-QAM MIMO system is

$$\frac{M \times M_c \times fclk}{\text{Cycle count}} = \frac{4 \times 4 \times fclk}{12} = \frac{4}{3} fclk. \tag{13}$$

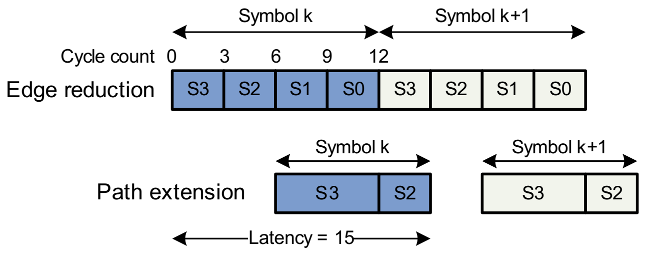


Figure 14: Timing diagram for a 4×4 16-QAM MIMO detection

6. VLSI IMPLEMENTATION RESULT

A 4×4 16-QAM soft MIMO detector has been synthesized (using Synopsys Design Compiler), placed and routed (using Cadence SoC Encounter) for a TSMC 65nm CMOS technology. Figure 15 shows the VLSI layout view of the proposed MIMO detector. The fixed-point bit precision for $\mathbf{R}$ and $\hat{\mathbf{y}}$ are 10 bits. The LLR outputs are represented in 7 bit. Based on the fixed-point simulation result, the finite word-length implementation leads to negligible performance degradation from using the floating-point representation. The maximum achievable clock frequency is 450 MHz based on the post-layout simulation. The corresponding maximum throughput is 600 Mbps.

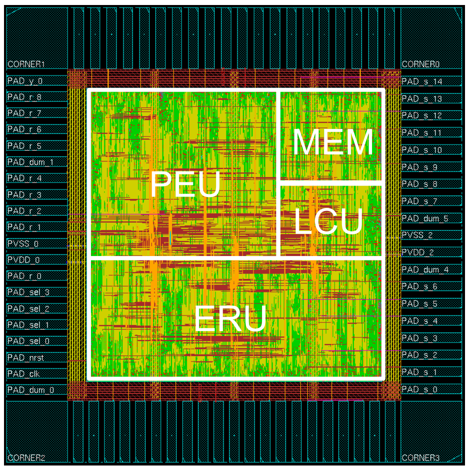


Figure 15: VLSI layout photo

Table 1 compares the detection throughput and hardware complexity of the proposed detector versus two state-of-the-art detectors from the literature: depth-first soft sphere detector with 256 search operations from [4], and soft K-best detector from [5]. In [5], a real QR decomposition is used with a small K=5. Based on the simulation results in Fig. 8, our solution has a better BER performance than [5] and can achieve a faster throughput because we avoid the sorting operation which is very expensive in the hardware implementation. On the other hand, at a cost of more hardware resources, the depth-first detector in [4] has a better BER performance than our solution. However [4] has a limited throughput because of the large number of sequential searching operations and it has variable throughput at different SNR levels. Our architecture provides a good solution in between the depth-first detector and the K-best detector.

Table 1: Architecture Comparison

	[4]	[5]	This work
Throughput	38.8 Mbps	106 Mbps	600 Mbps
Core area	10 mm^2	0.56 mm^2	0.79 mm^2
Gate count	1100 K	97 K	550 K
Max frequency	122.88 MHz	200 MHz	450 MHz
Technology	180 nm	130 nm	65 nm

7. CONCLUSION

We propose a new soft-output MIMO detector architecture based on a greedy graph algorithm. This detector can achieve a very high throughput of 600 Mbps at a hardware cost of only 550 K gates. Compared with other solutions, the proposed detector has a significant improvement in terms of detection throughput, latency, and area while still maintaining good bit error rate (BER) performance.

8. REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima. Near Shannon limit error-correcting coding and decoding: Turbo-Codes. In *IEEE Int. Conf. Commun.*, pages 1064–1070, May 1993.

[2] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bolcskei. VLSI Implementation of MIMO Detection Using the Sphere Decoding Algorithm. *IEEE J. Solid-State Circuit*, 40:1566–1577, July 2005.

[3] R. Gallager. Low-density parity-check codes. *IEEE Tran. Info. Theory*, 8:21–28, Jan. 1962.

[4] D. Garrett, L. Davis, S. ten Brink, B. Hochwald, and G. Knagge. Silicon Complexity for Maximum Likelihood MIMO Detection Using Spherical Decoding. *IEEE J. Solid-State Circuit*, 39:1544–1552, Sept. 2004.

[5] Z. Guo and P. Nilsson. Algorithm and implementation of the K-best sphere decoding for MIMO detection. *IEEE J. Selected Areas in Commun.*, 24:491–503, Mar. 2006.

[6] B. Hochwald and S. Brink. Achieving Near-Capacity on a Multiple-Antenna Channel. *IEEE Tran. Commun.*, 51:389–399, Mar. 2003.

[7] Q. Li and Z. Wang. Improved K-best sphere decoding algorithms for MIMO systems. In *IEEE Int. Symp. on Circuits and Syst.*, pages 2190–2194, Nov. 2006.

[8] Y. Sun and J. R. Cavallaro. A low-power 1-Gbps reconfigurable LDPC decoder design for multiple 4G wireless standards. In *IEEE International SOC Conference*, pages 367–370, Spet. 2008.

[9] Y. Sun and J. R. Cavallaro. A New MIMO Detector Architecture Based on a Forward-Backward Trellis Algorithm. To appear in *IEEE Asilomar Conference on Signals, Systems and Computers*, Oct. 2008.

[10] K. Wong, C. Tsui, R. Cheng, and W. Mow. A VLSI architecture of a K-best lattice decoding algorithm for MIMO channels. In *IEEE Int. Symp. on Circuits and Syst.*, volume 3, pages 273–276, May 2002.