VLSI Implementation of Digital Signal Processing Algorithms for MIMO/SISO Systems

by

Mahdi Shabany

A thesis submitted in conformity with the requirements for the degree of Doctor of Philosophy
Graduate Department of Electrical and Computer Engineering
University of Toronto

© Copyright by Mahdi Shabany 2009
VLSI Implementation of Digital Signal Processing Algorithms for MIMO/SISO Systems

Mahdi Shabany

Doctor of Philosophy, 2009
Graduate Department of Electrical and Computer Engineering
University of Toronto

Abstract

The efficient high-throughput VLSI implementation of near-optimal multiple-input multiple-output (MIMO) detectors for $4 \times 4$ MIMO systems in high-order quadrature amplitude modulation (QAM) schemes has been a major challenge in the literature. To address this challenge, this thesis introduces a novel scalable pipelined VLSI architecture for a $4 \times 4$ 64-QAM MIMO receiver based on K-Best lattice decoders. The key contribution is a means of expanding/visiting the intermediate nodes of the search tree on-demand, rather than exhaustively along with three types of distributed sorters operating in a pipelined structure. The combined expansion and sorting cores are able to find the $K$ best candidates in $K$ clock cycles. The proposed architecture has a fixed critical path independent of the constellation order, on-demand expansion scheme, efficient distributed sorters, and is scalable to a higher number of antennas/constellation orders. Fabricated in 0.13µm CMOS, it operates at a significantly higher throughput ($5.8 \times$ better) than currently reported schemes and occupies 0.95 mm$^2$ core area. Operating at 282 MHz clock frequency, it dissipates 135 mW at 1.3 V supply with no performance loss. It achieves an SNR-independent decoding throughput of 675 Mbps satisfying the requirements of IEEE 802.16m and Long Term Evolution (LTE) systems. The measurements confirm that this design consumes $3.0 \times$ less energy/bit compared to the previous best design.
Acknowledgments

This dissertation bears my name as the sole author, yet as any endeavor that spans the course of several years, it would have been impossible for me to complete without the help and encouragement of numerous people. First and foremost, I would like to express my most sincere gratitude towards my supervisor Professor P. G. Gulak, for being a role model through his relentless work ethic, skillful administration, insightful teaching methods, intelligent approach to research and boundless enthusiasm.

I thank the members of my Ph.D. defense committee, Prof. Paul Chow, Prof. T. J. Lim, Prof. J. Poon, and the external examiner Prof. X. Wang for their time and insightful suggestions.

I would also like to gratefully acknowledge the financial support provided by University of Toronto, Natural Sciences and Engineering Research Council of Canada (NSERC), Canadian Microelectronics Corporation (CMC), and Ontario Graduate Scholarship (OGS).

I thank Jaro Pristupa for solving CAD-related problems with speed and skill.

I feel blessed for getting to know so many good friends during my studies at the University of Toronto. I have learned a lot from them and I am grateful to all of them. Special thanks to Hamed Samadi and his wife for being intimate, supportive and wonderful friends. Many thanks to the gangs I spent most of my memorable times with, Meysam Roodi, Zahra Yazdizadeh, Hossein Sheikh Attar, Marzieh Abdollahi, Hamed Samadi, Narges Safari, Hesam Chniforooshan, Zeinab Hejazi, Saeed Moradi, and Sepideh Zarin. I also thank friends from BA5000, BA5158, Glenn’s group and those from outside the department. In particular, I would like to thank Mohamed Youssef Abdollah, Mehdi Ahmadi, Hossein Alizadeh, Kevin Banovic, Ahmad Darabiha, Roya Doostnejad, Amir Ghasemi, Afshin Haftbaradaran, Mohammad Hajirostam, David Halupka, Mohammad Ali Honarvar, Meisam Honarvar, Mahdi Lotfinezhad, Amir Mohammad Mazouchi, Ali Najj, Nasim Nikkhoo, Alireza Nilchi, Amir Parayandeh, Dimpesh Patel, Amir Hossein Ramezanianpour, Peyman Razzaghi, Siamak Sarvari,
Mehrdad Shamsi, Karen Su, in the alphabetic order.

I am grateful to my parents, for their love and continuous support. Without their sacrifices my dreams would have remained dreams.

No words are sufficient to express my gratitude and love for my wife Atieh, who has provided infinite support during the course of my Ph.D. and every aspect of my career, for which she has made many sacrifices. Her pride, love, encouragement, and devotion have sustained me through the ups and downs of academic and family life. She is the best wife and friend I could have dreamed of, and she enriches my life in every way.

I also would like to express my highest level of excitement to my expected baby boy who has significantly pumped a source of love and passion to my life although he has not yet come at the time of my defense. Naming him can be listed as a future work in this dissertation!

Last but definitely not least, I thank the person to whom I owe all of my achievements. His highness is an extraordinary person whom I have been impatiently waiting for since I found myself in this small world. May God bless him and expedite his appearance.
# Contents

List of Figures ix

List of Tables xiv

1 Introduction to MIMO Systems & Contributions 1

1.1 MIMO Technology ............................................. 1
1.2 Challenges ..................................................... 2
1.3 Contributions .................................................. 5
1.4 Published Papers .............................................. 6
1.5 Thesis Outline ................................................ 7

2 Fundamentals of MIMO Detection 8

2.1 System Model .................................................. 8
2.2 Processing Rates ............................................. 12
2.3 Simulation Framework ....................................... 12
2.4 Preprocessing Block ........................................ 13
  2.4.1 LMMSE-based Preprocessing ............................ 13
2.5 MIMO Detection Schemes ................................. 14
  2.5.1 ML Detection ........................................... 15
  2.5.2 Linear Detectors ....................................... 16
  2.5.3 Non-linear Detectors ................................ 19
2.6 Antenna Correlation ....................................... 26

3 The K-Best MIMO Detection Algorithm 27

3.1 Introduction .................................................. 27
3.2 K-Best Algorithm ........................................... 27
  3.2.1 Theory .................................................. 27
## 3.2.2 Challenges

3.3 Proposed On-demand Expansion and Distributed Sorting for the K-Best Algorithm

3.3.1 Real Domain

3.3.2 First/Next Child Calculation

3.3.3 Complex Mode

3.4 Complexity Analysis

3.5 Simulations

3.6 Summary

## 4 VLSI Implementation of a Scalable K-Best Detector

4.1 General VLSI Architecture

4.2 Detailed VLSI Architecture

4.2.1 Inputs

4.2.2 Level I

4.2.3 Level II

4.2.4 Sorter Block

4.2.5 PE I Block

4.2.6 NC-Block

4.2.7 PE II Block

4.2.8 FC-Block

4.2.9 Latency and Bit-true Simulation

4.3 Complexity Analysis

4.4 Simulations

4.5 Extension to 256-QAM Scheme

4.6 Design Comparison

4.7 Test Results

4.8 Summary

## 5 Joint Lattice-Reduction and K-Best Algorithm

5.1 Introduction

5.2 System Model

5.2.1 Lattice-Reduction

5.3 Problem Definition (LR-Aided K-Best)
## Contents

5.4 Proposed Scheme ........................................ 103
  5.4.1 Sorting Scheme ..................................... 103
  5.4.2 On-demand Expansion Scheme ....................... 106
5.5 Complexity Analysis .................................. 107
5.6 Simulations ........................................... 108
  5.6.1 The Effect of Antenna Correlation ................. 109
5.7 Summary ............................................... 110

6 Compensation of the Nonlinearity of Power Amplifiers Using Sequential Monte Carlo 112
  6.1 Introduction ......................................... 112
  6.2 System Model ........................................ 115
    6.2.1 HPA Model .................................... 115
    6.2.2 Predistorter .................................. 117
  6.3 The SMC Receiver ................................... 119
    6.3.1 SMC Methodology ............................... 119
    6.3.2 Application of SMC to SSPA ................. 120
    6.3.3 Known Parameters ............................. 121
    6.3.4 Unknown Parameters (Adaptive scheme without memory) .. 122
  6.4 SMC Algorithm ...................................... 123
    6.4.1 Unknown Parameters (Adaptive Scheme with Memory) ..... 125
  6.5 Complexity Analysis ................................ 127
    6.5.1 Adaptive Scheme without Memory ............. 127
    6.5.2 Adaptive Scheme with Memory ................ 127
  6.6 Performance Analysis and Simulation Results ......... 128
    6.6.1 Known Parameters .............................. 128
    6.6.2 Unknown Parameters ............................ 137
  6.7 Limitations of a Multi-carrier System ............... 141
  6.8 Summary ............................................ 145

7 Conclusions and Future Directions 147
  7.1 Conclusions ......................................... 147
  7.2 Future Work ......................................... 148
    7.2.1 MIMO Detection ............................... 148
7.2.2 Lattice Reduction ................................. 150
7.2.3 SSPA Compensation ............................... 150

A Detailed Measurement Results .......................... 151
A.1 Test Results @ 80°C ................................ 151

B Efficient Architectures for SMC Resampling .................. 158
B.1 Introduction ................................... 158
B.2 Centralized Implementation ........................... 159
B.3 Distributed Implementation ............................ 160
B.4 Distributed Resampling Scheme ....................... 161
  B.4.1 Offset Passing ................................... 161
  B.4.2 Access List Derivation ........................... 163
  B.4.3 Scheduling ........................................ 165
B.5 Performance Analysis And Simulation Results ............. 167
B.6 Summary ............................................... 169

References .................................................. 170
## List of Figures

1.1 Processing requirements of MIMO algorithms in different standards along with the capabilities of different hardware architectures [1].  
2.1 The MIMO system under consideration. The indicated data rates are that achieved in a realization of the MIMO detector presented in this thesis where $N_T = 4$ and $N_R = 4$.  
2.2 Taxonomy of MIMO detection algorithms. The focus of this thesis is highlighted.  
2.3 The comparison of various sub-optimal detectors with the ML detector in a $4 \times 4$ system with 16-QAM modulation.  
2.4 The concept of SD with the sphere constraint $r$.  
3.1 Real and Complex interpretation of the MIMO detection problem for a $2 \times 2$, 4-QAM MIMO system.  
3.2 The K-Best algorithm for $\sqrt{M} = 4$ and $N_T = N_R = 2$.  
3.3 The order of the SE row-enumeration for four consecutive enumerations in 16-QAM.  
3.4 The proposed distributed K-Best algorithm for $\sqrt{M} = 4$ and $K = 3$ and example PED values.  
3.5 The three-level tree used for enumeration of the complex constellation $\mathcal{O}$.  
3.6 The first four best children using complex SE enumeration in a 16-QAM Constellation scheme: (a) $\mathcal{L} = \{-1+j\}$, (b) $\mathcal{L} = \{-1-j, +1+j\}$, (c) $\mathcal{L} = \{-1-j, 1-j, -3+j\}$ and (d) $\mathcal{L} = \{-1+3j, 1-j, -3+j\}$.  
3.7 Six possible cases for proof of the functionality of the complex SE enumeration.
3.8 The variation of the value of $|L|$ for 16-QAM for a specific received symbol: (a) $|L| = 3$, (b) $|L| = 4$, (c) $|L| = 4$, (d) $|L| = 4$, (e) $|L| = 4$, (f) $|L| = 1$. ...................................................... 49

3.9 The BER performance of the K-Best real-domain scheme vs. the ML detector for different values of $K$ for a $4 \times 4$, 64-QAM MIMO detector. 51

3.10 K-Best vs. ML BER for different values of $K$ in both real and complex domain for $4 \times 4$ 16-QAM MIMO detection. ........................... 52

3.11 K-Best vs. ML BER for different values of $K$ in both real and complex domain for $4 \times 4$ 64-QAM MIMO detection. ......................... 52

4.1 One of $2N_T$ pipeline stages of the K-best VLSI architecture proposed in [2]. ......................................................... 56

4.2 KBU unit [2] that performs the merging for $K = 5$. ................. 57

4.3 The proposed pipelined VLSI architecture of the K-Best algorithm for the detection of a $4 \times 4$, 64-QAM system with $K = 10$. .......... 59

4.4 The scheduling for reading $r_{ij}$ and $z_j$ values. .................. 62

4.5 Alternative architecture for multiplication (MU). ..................... 63

4.6 The architecture of the Mapper, where $s_i^{[0]} = 2 \left( \frac{s_i^{[0]} + 1}{2} + 0.5 \right) - 1$. .......................... 64

4.7 The architecture for the Limiter block. ............................. 64

4.8 The architecture for Level I with the critical path highlighted in a gray box. ......................................................... 65

4.9 The performance of a $4 \times 4$ 64-QAM MIMO system with $K = 10$ for $\ell^1$-norm and $\ell^2$-norm case. ............................. 66

4.10 The architecture for Level II with the critical path highlighted. . 67

4.11 The architecture for the Sorter block with the critical path highlighted. 68

4.12 The architecture for the PE I block with the critical path highlighted. 69

4.13 The architecture for the NC-Block with the critical path highlighted. 71

4.14 The architecture for the NC-Block with improved critical path. .... 72

4.15 The architecture for the PE II block with the critical path highlighted. 73

4.16 The pairwise data transfer from PE II to PE I, (a) two entries at a time, (b) one entry at a time. ........................................ 74

4.17 The timing scheduling between a typical pair of PE II and PE I. .... 75

4.18 The architecture for the FC-Block inside the PE II block with the critical path highlighted. ................................. 76
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.19</td>
<td>K-Best floating/fixed-point vs ML for $4 \times 4$, 16-QAM with $K = 5$.</td>
<td>80</td>
</tr>
<tr>
<td>4.20</td>
<td>K-Best floating/fixed-point vs ML for $4 \times 4$, 64-QAM with $K = 10$.</td>
<td>80</td>
</tr>
<tr>
<td>4.21</td>
<td>K-Best vs ML for $4 \times 4$, 256-QAM with $K = 15$.</td>
<td>81</td>
</tr>
<tr>
<td>4.22</td>
<td>Micrograph of the implemented ASIC.</td>
<td>86</td>
</tr>
<tr>
<td>4.23</td>
<td>Throughput vs. gate count compared to previously published works.</td>
<td>87</td>
</tr>
<tr>
<td>4.24</td>
<td>Test setup (Agilent(Verigy) 93K tester, Temptronic TP04300 thermal forcing unit head, and the chip).</td>
<td>87</td>
</tr>
<tr>
<td>4.25</td>
<td>Maximum operating frequency vs. supply voltage ($V_{dd}$) at 25°C.</td>
<td>88</td>
</tr>
<tr>
<td>4.26</td>
<td>Power dissipation vs. supply voltage ($V_{dd}$) at 25°C.</td>
<td>89</td>
</tr>
<tr>
<td>4.27</td>
<td>Measurement plots for maximum frequency and power dissipation vs. supply voltage ($V_{dd}$) at 25°C.</td>
<td>90</td>
</tr>
<tr>
<td>4.28</td>
<td>Measurement plots for maximum frequency and power dissipation vs. supply voltage ($V_{dd}$) at 0°C.</td>
<td>91</td>
</tr>
<tr>
<td>4.29</td>
<td>Measured throughput/area vs. energy/bit, with area measured in kilogates (KG) @ 282 MHz, 1.3 V and 25°C. Results of the designs in [3] and [4] have been scaled to a 0.13μm equivalent CMOS process.</td>
<td>92</td>
</tr>
<tr>
<td>4.30</td>
<td>Measured throughput vs energy/bit @ 282 MHz, 1.3 V and 25°C. Results of the designs in [3] and [4] have been scaled to a 0.13μm equivalent CMOS process.</td>
<td>93</td>
</tr>
<tr>
<td>4.31</td>
<td>Measured BER at a clock rate of 282 MHz at a measured sustained throughput of 675Mb/s dissipating 135mW @ 1.3V supply and 25°C.</td>
<td>94</td>
</tr>
<tr>
<td>5.1</td>
<td>Typical detection framework.</td>
<td>97</td>
</tr>
<tr>
<td>5.2</td>
<td>The introduction of LR to the detection framework.</td>
<td>100</td>
</tr>
<tr>
<td>5.3</td>
<td>The possible integer values of (a) $s$ based on $H$, (b) $X$ based on the new bases of $H$.</td>
<td>102</td>
</tr>
<tr>
<td>5.4</td>
<td>LR-aided K-Best vs. ML for $4 \times 4$ for 16-QAM.</td>
<td>108</td>
</tr>
<tr>
<td>5.5</td>
<td>LR-aided K-Best vs. ML for $4 \times 4$ for 64-QAM.</td>
<td>109</td>
</tr>
<tr>
<td>5.6</td>
<td>LR-aided K-Best vs ML for $4 \times 4$ for 256-QAM ($K = 15$).</td>
<td>110</td>
</tr>
<tr>
<td>5.7</td>
<td>LR-aided K-Best, K-Best and ML for $4 \times 4$ 64-QAM, with correlation $(\rho = 0.1)$.</td>
<td>111</td>
</tr>
<tr>
<td>5.8</td>
<td>LR-aided K-Best, K-Best and ML for $4 \times 4$ 64-QAM, with correlation $(\rho = 0.4)$.</td>
<td>111</td>
</tr>
<tr>
<td>6.1</td>
<td>System model for the SMC receiver.</td>
<td>115</td>
</tr>
</tbody>
</table>
6.2 Characteristic function of the SSPA, the predistorter, and SSPA+predistorter, where $\alpha = 0.1$, $A_0 = 1$, $A_a = 2.65$, $p = 2$, and $\nu = 1$. .................. 118
6.3 The system under simulation for the predistorter. .................... 119
6.4 The adaptive SMC scheme with memory. ............................. 125
6.5 Performance of SMC compared to the predistorter with different input backoff values for a 4-QAM scheme: (a) IBO = 6 dB, (b) IBO = 9 dB, (c) IBO = 12 dB and (d) IBO = 15 dB. ...................... 129
6.6 The received points with different values of IBO for 16-QAM at SNR = 16: (a) IBO = 4 dB and (b) IBO = 10 dB. ...................... 130
6.7 Performance of SMC compared to the predistorter with different input backoff values for a 16-QAM scheme: (a) IBO = 7 dB, (b) IBO = 9 dB, (c) IBO = 12 dB, and (d) IBO = 15 dB. ...................... 131
6.8 Performance of SMC compared to the predistorter with different input backoff values for a 64-QAM scheme: (a) IBO = 9 dB, (b) IBO = 10 dB, (c) IBO = 12 dB, (d) IBO = 15 dB. ...................... 132
6.9 Performance of SMC compared to the predistorter with different input backoff values for a 256-QAM scheme: (a) IBO = 10 dB, (b) IBO = 12 dB. ................................. 133
6.10 Predistorted points before amplification at IBO=9 dB for: (a) 16-QAM, (b) 256-QAM. ......................... 134
6.11 The percentage of the points in the saturation region vs. IBO value for the predistorter (black bars) and SMC (white bars), (a) 4-QAM, (b) 16-QAM, (c) 64-QAM, (d) 256-QAM. ......................... 135
6.12 Total degradation of different modulation schemes vs. OBO for both SMC and the predistorter for SER = $10^{-2}$: (a) 16-QAM, (b) 64-QAM, (c) 256-QAM, (d) All. ................................. 136
6.13 Adaptive SMC receiver for 16-QAM for IBO = 7 dB. .................. 139
6.14 Adaptive SMC receiver for 64-QAM for IBO = 10 dB. .................. 140
6.15 Sequential adaptive vs adaptive receiver for 64-QAM for IBO = 10 dB. ................................. 141
6.16 The spectral mask of IEEE802.11g. ................................. 142
6.17 The spectral shape for a multi-carrier system with 16-QAM modulation scheme for OBO values of 0 dB, 1.3 dB, 1.9 dB, and 3 dB. .............. 143
6.18 The spectral shape for a multi-carrier system with 64-QAM modulation scheme for OBO values of 1 dB, 2.7 dB, 3.2 dB, and 4.2 dB. .............. 144
6.19 The preferred operating region of the SMC and predistorter as a function of OBO considering the mask constraint for: (a) 16-QAM, (b) 256-QAM. ................................................................. 145

A.1 Measurement plots for maximum frequency and power dissipation vs. supply voltage ($V_{dd}$) at 80°C. ................................................................. 152

B.1 Resampling routing scheme. ................................................................. 159
B.2 Offset passing scheme. ................................................................. 162
B.3 Pre-section core for access list derivation. ................................................................. 162
B.4 The detailed function of the $i$-th processing element used in pre/post-section core. ................................................................. 163
B.5 Post-section core for access list derivation. ................................................................. 164
B.6 An example of the pre-section core for access list derivation. ................................................................. 164
B.7 Timing flow comparison of the whole SMC process between sequential resampling and our proposed distributed resampling. ................................................................. 166
B.8 Performance comparison of various resampling schemes. ................................................................. 168
B.9 The comparison between the execution time vs. the number of PEs for both RNA and our proposed scheme. ................................................................. 169
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>The K-Best Algorithm.</td>
<td>30</td>
</tr>
<tr>
<td>3.2</td>
<td>Distributed K-Best Algorithm.</td>
<td>34</td>
</tr>
<tr>
<td>3.3</td>
<td>First/Next Child Selection Procedure for Node $j$.</td>
<td>36</td>
</tr>
<tr>
<td>3.4</td>
<td>The Proposed Implementation for the K-Best Algorithm.</td>
<td>38</td>
</tr>
<tr>
<td>3.5</td>
<td>Comparison of Different K-Best Implementations.</td>
<td>47</td>
</tr>
<tr>
<td>4.1</td>
<td>Fixed-point Word-Length (bits) of Parameters.</td>
<td>78</td>
</tr>
<tr>
<td>4.2</td>
<td>Comparison of Different K-Best Implementations.</td>
<td>79</td>
</tr>
<tr>
<td>4.3</td>
<td>Hardware Increase from 64-QAM to 256-QAM</td>
<td>82</td>
</tr>
<tr>
<td>4.4</td>
<td>Comparison of the Current ASIC Implementations of $4 \times 4$ MIMO Detectors.</td>
<td>84</td>
</tr>
<tr>
<td>4.5</td>
<td>Characteristics Summary of Detector and Measured Results.</td>
<td>95</td>
</tr>
<tr>
<td>5.1</td>
<td>The Proposed Scheme for LR-aided K-Best Algorithm.</td>
<td>105</td>
</tr>
<tr>
<td>5.2</td>
<td>First/Next Child Selection Procedure.</td>
<td>106</td>
</tr>
<tr>
<td>5.3</td>
<td>Complexity of the LR-aided K-Best Scheme for a $4 \times 4$ MIMO System.</td>
<td>107</td>
</tr>
<tr>
<td>A.1</td>
<td>Measurement Results for Chip #1 @ 0°C.</td>
<td>153</td>
</tr>
<tr>
<td>A.2</td>
<td>Measurement Results for Chip #1 @ 25°C.</td>
<td>153</td>
</tr>
<tr>
<td>A.3</td>
<td>Measurement Results for Chip #1 @ 80°C.</td>
<td>153</td>
</tr>
<tr>
<td>A.4</td>
<td>Measurement Results for Chip #2 @ 0°C.</td>
<td>154</td>
</tr>
<tr>
<td>A.5</td>
<td>Measurement Results for Chip #2 @ 25°C.</td>
<td>154</td>
</tr>
<tr>
<td>A.6</td>
<td>Measurement Results for Chip #2 @ 80°C.</td>
<td>154</td>
</tr>
<tr>
<td>A.7</td>
<td>Measurement Results for Chip #3 @ 0°C.</td>
<td>155</td>
</tr>
<tr>
<td>A.8</td>
<td>Measurement Results for Chip #3 @ 25°C.</td>
<td>155</td>
</tr>
<tr>
<td>A.9</td>
<td>Measurement Results for Chip #3 @ 80°C.</td>
<td>155</td>
</tr>
<tr>
<td>A.10</td>
<td>Measurement Results for Chip #4 @ 0°C.</td>
<td>156</td>
</tr>
</tbody>
</table>
A.11 Measurement Results for Chip #4 @ 25°C. . . . . . . . . . . . . . . . 156
A.12 Measurement Results for Chip #4 @ 80°C. . . . . . . . . . . . . . . . 156
A.13 Measurement Results for Chip #5 @ 0°C. . . . . . . . . . . . . . . . 157
A.14 Measurement Results for Chip #5 @ 25°C. . . . . . . . . . . . . . . . 157
A.15 Measurement Results for Chip #5 @ 80°C. . . . . . . . . . . . . . . . 157

B.1 Comparison of Resampling Schemes with J Samples and K PEs. . . 167
B.2 Memory Usage Breakdown for Parallel Implementation of Resampling. 167
List of Symbols

**MIMO Detection Framework:**

- $y$: Real received symbol vector
- $s$: Real transmitted symbol vector
- $\tilde{s}$: Complex transmitted symbol vector
- $H$: Real MIMO channel matrix
- $v$: Real noise vector
- $Q$: Unitary matrix
- $R$: Upper triangular matrix with real entries
- $z$: Post processed real received symbol vector
- $\tilde{y}$: Complex received symbol vector
- $\tilde{s}$: Complex transmitted symbol vector
- $\tilde{H}$: Complex MIMO channel matrix
- $\tilde{v}$: Complex noise vector
- $N_R$: Number of received antenna
- $N_T$: Number of transmit antenna
- $x(n)$: Transmitted bit vector at time $n$
- $\hat{x}$: Estimated version of the transmitted vector
- $O$: Complex constellation
- $M$: Constellation size/ordinality
- $M_c$: Number of bits per constellation point
- $R$: Number of bits per channel use
- $\sigma^2$: Noise variance
- $N_c$: Complex Gaussian distribution
- $\Re\{\cdot\}$: Real part of a complex number
- $\Im\{\cdot\}$: Imaginary part of a complex number
- $\Omega$: Set of possible real entries in $O$
- $K$: Number of K-Best candidates in each level of the tree
$T_l(s^{(l)})$ Accumulated partial Euclidean distance in level $l$

$e_l(s^{(l)})$ Distance increment between two successive nodes in level $l$

$\mathcal{K}_l$ List of K-Best children in level $l$

$\mathcal{C}_l$ The set of all the current best child of all parents

$\mathcal{D}_l$ PED values of the elements of $\mathcal{C}_l$

$\gamma$ Total transmit power at the transmitter

$\eta$ Total transmit power of each antenna

$\mathbf{P}$ Augmented channel matrix

$\mathbf{G}$ General linear estimator matrix

$\mathbf{G}_{ZF}$ ZF estimator matrix

$\mathbf{G}_{MMSE}$ MMSE estimator matrix

$Q(\cdot)$ Slicing operation

$E\{\cdot\}$ Expectation operation

$h_l$ $l$-th column of channel matrix $\mathbf{H}$

$\hat{s}_i$ $i$-th estimated symbol at the receiver

$r$ Sphere constraint in SD

$\lambda$ Signal wavelength

$\Phi_T$ Correlation matrix at the transmitter

$\rho_T$ Correlation coefficient at the transmitter

$\Phi_R$ Correlation matrix at the receiver

$\rho_R$ Correlation coefficient at the receiver

$r_{ij}$ An entry of matrix $\mathbf{R}$

$\bar{r}_{ij}$ The scaled version of $r_{ij}$ by $r_l$

$s^{[k]}_l$ $k$-th best child of a parent in level $l$

$\mathcal{L}$ All visited points, which have not been announced as the next best sibling
**SMC Framework:**

\[ x(t) \] Transmitted signal  
\[ s(t) \] Modulated signal  
\[ y(t) \] Amplified signal  
\[ r(t) \] Received signal  
\[ \hat{s}(t) \] Estimated symbol at the receiver  
\[ \rho(t) \] Signal amplitude  
\[ \varphi(t) \] Signal phase  
\[ G(\cdot) \] SSPA characteristic function  
\[ G[\rho(t)] \] AM/AM conversion characteristic function  
\[ \Phi[\rho(t)] \] AM/PM conversion characteristic function  
\[ \nu \] SSPA small-signal gain  
\[ A_o \] SSPA output saturation voltage  
\[ A_s \] SSPA input saturation voltage  
\[ p \] Control parameter for SSPA smoothness  
\[ \bar{P}_O \] Mean power of the transmitted signal  
\[ P_{O,\text{sat}} \] Maximum output power  
\[ P_{1,\text{sat}} \] Input power corresponding to the maximum output power  
\[ P_1 \] Mean power of the signal at the input of the SSPA  
\[ \chi_t \] Discrete random measure  
\[ \delta(\cdot) \] Dirac delta function  
\[ x_{\text{pt}}^{(i)} \] Sample set  
\[ \omega_i^{(i)} \] Weight set  
\[ \mathcal{E} \] Set of SSPA main parameters  
\[ W \] Sum of all the weights  
\[ J \] Number of samples  
\[ \mathcal{N}(\cdot) \] Gaussian distribution  
\[ M \] Constellation size  
\[ 1/T \] Sampling rate in SSPA  
\[ N_j \] Weights after resampling  
\[ f_c \] Carrier frequency
List of Acronyms

A/D  Analog-to-Digital
ASIC  Application-Specific Integrated Circuit
AWGN  Additive White Gaussian Noise
BER  Bit-Error-Rate
BLAST  Bell Labs Layered Space-Time
bpcu  Bits per Channel Use
CDMA  Code Division Multiple Access
CMOS  Complementary Metal Oxide Semiconductor
D/A  Digital-to-Analog
DSP  Digital Signal Processor
FC  First Child
FFT  Fast Fourier Transform
HPA  High Power Amplifier
HSDPA  High-Speed Downlink Packet Access
IBO  Input Backoff
KBU  K-Best Unit
LLL  Lenstra, Lenstra, Lovasz
LLR  Log-Likelihood Ratio
**LMMSE** Least Minimum Mean Squared Error

**LR** Lattice Reduction

**LTE** Long Term Evolution

**Mbps** Mega bits per second

**MCU** Metric Computation Unit

**MIMO** Multiple-Input Multiple-Output

**ML** Maximum-Likelihood

**MUX** Multiplexer

**NC** Next Child

**OBO** Output Backoff

**OFDM** Orthogonal Frequency-Division Multiplexing

**PAPR** Peak-to-Average-Power-Ratio

**PE** Processing Element

**PED** Partial Euclidean Distance

**PSK** Phase Shift Keying

**QAM** Quadrature Amplitude Modulation

**QoS** Quality-of-Service

**RNA** Resampling Non-proportional Allocation

**RPA** Resampling Proportional Allocation

**RVD** Real-Valued Decomposition

**S/P** Serial to Parallel Conversion (Demux)

**SA** Seysen’s Algorithm
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SD</td>
<td>Sphere Decoding</td>
</tr>
<tr>
<td>SE</td>
<td>Schnorr-Euchner</td>
</tr>
<tr>
<td>SER</td>
<td>Symbol Error Rate</td>
</tr>
<tr>
<td>SIC</td>
<td>Sequential Interference Cancelation</td>
</tr>
<tr>
<td>SINR</td>
<td>Signal-to-Interference-and-Noise Ratio</td>
</tr>
<tr>
<td>SISO</td>
<td>Single-Input Single-Output</td>
</tr>
<tr>
<td>SM</td>
<td>Spatial Multiplexing</td>
</tr>
<tr>
<td>SMC</td>
<td>Sequential Monte Carlo</td>
</tr>
<tr>
<td>SNR</td>
<td>Signal-to-Noise-Ratio</td>
</tr>
<tr>
<td>SSPA</td>
<td>Solid-State Power Amplifier</td>
</tr>
<tr>
<td>TD</td>
<td>Total Degradation</td>
</tr>
<tr>
<td>TWTA</td>
<td>Traveling Wave Tube Amplifier</td>
</tr>
<tr>
<td>VLSI</td>
<td>Very Large Scale Integration</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local Area Network</td>
</tr>
<tr>
<td>WMAN</td>
<td>Wireless Metropolitan Area Network</td>
</tr>
<tr>
<td>WiMAX</td>
<td>Worldwide Interoperability for Microwave Access</td>
</tr>
<tr>
<td>ZF</td>
<td>Zero-Forcing</td>
</tr>
</tbody>
</table>
1 Introduction to MIMO Systems & Contributions

1.1 MIMO Technology

Due to the high spectral efficiency, Multiple-Input-Multiple-Output (MIMO) systems [5] have attracted significant attention as the technology of choice in many standards. For instance, in the IEEE 802.11n Wireless Local Area Network (WLAN) standard, MIMO is the key technology to achieve the target throughput of over 480 Mbps. MIMO is also adopted for high data-rate modes for IEEE 802.16e Wireless Metropolitan Area Network (WMAN) system, also known as Worldwide Interoperability for Microwave Access (WiMAX) [6], as well as the next generation WiMAX systems (IEEE 802.16m standard), and post-3G cellular systems such as the 3rd Generation Partnership Project (3GPP) release 6, which introduces antenna array technologies into the second phase of the High-Speed Downlink Packet Access (HSDPA) specification. The future 3GPP roadmap after HSDPA is being developed in the Long Term Evolution (LTE) project, which aims at up to 100 Mbps data rate for downlink and 50 Mbps for uplink.

In fact MIMO systems employ multiple antennas at both the transmitter and at the receiver to meet the requirements of these standards. From an information theoretic perspective, increasing the number of antennas provides a vehicle to achieve higher spectral efficiency compared to Single-Input Single-Output (SISO) systems. Actual transmission schemes exploit this higher capacity by leveraging three types of gains [7]:

- **Array gain** refers to picking up a larger share of the transmitted power at the receiver, which allows one to extend the range of a communication system and to suppress interference.

- **Diversity gain** describes the behavior of an algorithm in the limit of high signal-to-noise (SNR), and the diversity order corresponds directly to the slope
of the bit-error-rate (BER) curve. The uncoded spatial multiplexing system (without transmit channel knowledge) can achieve a maximum diversity order of $N_N$ with an optimum receiver, where $N_N$ is the number of receive antennas. In fact diversity gain counters the effect of variations in the channel, known as fading, which increases link-reliability and hence Quality-of-Service (QoS).

- **Multiplexing gain** allows for a linear increase in spectral efficiency and peak data rates by transmitting multiple data streams concurrently in the same frequency band using $N_T$ transmit antennas. The number of parallel streams is thereby limited by the number of transmit or receive antennas, whichever is smaller.

A tradeoff exists between these three gains, as maximizing each of them requires different transmission schemes. Space-time coding [8], for example, mainly exploits the diversity. Beamforming [9] uses multiple antennas to suppress interference and to maximize the array gain. Opportunistic beamforming [10] is also used to achieve the diversity gain. Finally, the full-rate Spatial Multiplexing (SM) scheme uses all available antennas to achieve the highest possible peak data rates and the maximum possible spectral efficiency through the multiplexing gain. The prospect of these tremendous gains has recently led to considerable efforts to incorporate MIMO technology into various important wireless standards.

### 1.2 Challenges

The significant performance improvements associated with MIMO systems come at the expense of significantly more complex signal processing at the transmitter and receiver. In particular, with spatial multiplexing, the linear increase in spectral efficiency, which is proportional to the minimum of the number of antennas at the transmitter and the receiver, comes with a more than linear increase in the decoder complexity. In other words, exploiting the full potential of multi-antenna technology to meet the requirements of the current and future standards requires algorithms that have even higher complexity, which might exceed the limits of what is economically feasible with today’s digital signal processors (DSPs) or other software programmable processing architectures as shown in Fig. 1.1. However, the key to the successful commercialization of MIMO technology is the availability of highly integrated and
affordable terminals. Therefore, one of the major challenges in MIMO systems is to design low-complexity receiver algorithms and to develop efficient dedicated Very Large Scale Integration (VLSI) architectures for their implementation.

One of the most challenging parts of a MIMO receiver in terms of the complexity is the MIMO detector for the SM scheme. In the SM mode, the task of a MIMO detector is to separate the spatially multiplexed data streams at the receiver. In the literature, complexity analysis of MIMO receiver algorithms has mostly been based on the considerations of their complexity order, which is only applicable to qualitative comparisons between algorithms in the limit of a large number of antennas [1]. As in most practical scenarios, the number of antennas is small (typically 2-4), the corresponding results are of little practical interest.

A more detailed complexity analysis and algorithm optimizations for complexity reduction are often performed with DSP implementations in mind. However, DSP implementations and implementations on other programmable processing architectures usually cannot meet the requirements of currently emerging and future wideband MIMO systems. Consequently, dedicated VLSI architectures are still needed for the implementation of the most computationally complex algorithms. In fact, actual VLSI implementations of MIMO algorithms have only emerged recently. The
few algorithms and designs that have been published provide initial reference points defining the silicon complexity of MIMO detectors and illustrate suitable hardware architectures. Nevertheless, high-throughput wide-band MIMO systems require further improvements and optimizations to ensure that system performance is ultimately only limited by the wireless channel capacity and not by the available receiver technology.

One field of focus of this dissertation is thus to design such a dedicated VLSI architecture for MIMO systems employing the spatial multiplexing scheme. The main objective is to propose an efficient framework for the VLSI implementation of MIMO detectors with a reasonable complexity while achieving the envisioned throughput in the future standards. Thus the target of the first part of this thesis is to develop a framework that is suitable for implementation of MIMO detectors with large constellation size (64-QAM or 256-QAM) and large number of antennas (say larger than 4). This is due to the fact that an efficient architecture, scalable to high constellation sizes and/or large number of transmit antennas, is still a significant challenge and has not been properly addressed in the literature.

Another challenge for MIMO systems and any other communication system is the nonlinearity of the power amplifiers, which either forces having a back-off resulting in low-efficiency amplifiers or leads to interference in adjacent carriers especially in multi-carrier modulation schemes. The second field of focus of this dissertation is to address this issue to develop a novel framework for compensating the amplifier nonlinearities. This study is of extreme importance since in the case of wireless systems, where power is a costly and often a limited resource, the power amplifiers are the most power consuming component in the overall transceiver power budget. The main scope of the discussion relates to single-input single-output (SISO) systems with one antenna at the transmitter and receiver, but the extension of the proposed scheme to MIMO systems is straightforward.
1.3 Contributions

1. The development of a novel K-Best scheme for near-optimal MIMO detection with the following features:
   - Complexity independent of the constellation.
   - Scales sub-linearly with the constellation size.
   - Fixed-length critical path independent of the constellation size.
   - Finds $K$ best candidates in $K$ clock cycles.
   - Expands a very small fraction of all the possible children compared to the exhaustive K-Best approach.
   - Can be applied to infinite lattices.
   - Can be jointly applied with the lattice reduction.
   - Provides the exact K-Best solution without any approximation.
   - Can be extended to the complex mode.

2. The extension of the proposed K-Best detector to the complex domain.

3. Proposing a framework for the joint application of lattice reduction and the K-Best algorithm to improve the diversity gain of the K-Best algorithm in high SNR regimes.

4. Design, fabrication and successful test of an Application Specific Integrated Circuit (ASIC) implementation of the proposed K-Best scheme in 0.13µm CMOS technology, achieving 675 Mbps for a $4 \times 4$ 64-QAM MIMO system. The tested design achieves a $5.8 \times$ greater throughput and $3 \times$ lower energy-per-bit than that found in the literature for comparable systems.

5. Proposing a novel method for compensation of the nonlinearity of the solid-state power amplifiers for low-IBO and/or high-order constellation schemes based on the Sequential Monte Carlo (SMC) methodology.

6. Develop an efficient architecture for the implementation of the resampling core, an essential processing core found in the SMC algorithm.
1.4 Published Papers

The following papers have been published based on the content of this thesis:


### 1.5 Thesis Outline

The outline of the thesis is as follows. Chapter 2 provides background on the various MIMO detectors with their performance and complexity characteristics. Chapter 3 describes the proposed on-demand K-Best algorithm implementation from the algorithmic point-of-view for both the real and complex domain. Chapter 4 addresses the VLSI implementation aspects of the proposed scheme and reports the ASIC implementation and the test results for the fabricated design. Chapter 5 investigates the integration of the K-Best algorithm with lattice reduction schemes and proposes a joint algorithm achieving close-to-optimal performance results. Chapter 6 discusses the sequential Monte Carlo (SMC) algorithm and its application to the compensation of the nonlinearity of the power amplifiers in the MIMO framework. Finally Chapter 8 concludes the thesis and provides potential venues for future work.
2 Fundamentals of MIMO Detection

The first part of this chapter provides a description of the MIMO system under consideration and introduces the concept of MIMO detection as well as the notation and terminology that will be used throughout this thesis. The detailed description of the state-of-the-art algorithms for MIMO detection in the literature will be addressed in the subsequent parts of the chapter.

2.1 System Model

It is well-known that using the proper modulation technique, such as Orthogonal Frequency-Division Multiplexing (OFDM), or with proper equalization, most wide-band MIMO communication systems can be reduced to a set of narrow-band MIMO systems. Therefore, a narrow-band system model can be considered as a simple canonical form based on which it is straightforward to derive corresponding receivers for wide-band MIMO communication systems. Hence, a narrow-band system model shall serve as the basis for subsequent discussions to ensure that the results are applicable to a wide range of communication scenarios and to provide a common basis for the comparison of different algorithms.

Consider a MIMO system shown in Fig. 2.1, where the number of transmit antennas is denoted by $N_T$ and the number of receive antennas is denoted by $N_R$. In this thesis, it is always assumed that $N_R \geq N_T$. At time $n$, the bit sequence $x(n) = [x_1(n), \ldots, x_{M_cN_T}(n)]^T$ is sent to $N_T$ parallel streams using a serial-to-parallel (S/P) block, which are mapped into a complex vector $\tilde{s}(n) = [\tilde{s}_1(n), \ldots, \tilde{s}_{N_T}(n)]^T$ by $N_T$ linear modulators at the transmitter front end. Each element $\tilde{s}_i(n)$ is taken

---

1In this thesis, complex variables are distinguished from real variables by a “∼” sign. Moreover, matrices and vectors are distinguished from scalars by using a bold font. For instance, the complex channel matrix is referred to by $\tilde{H}$ whereas the real channel matrix is denoted by $H$. 

Figure 2.1: The MIMO system under consideration. The indicated data rates are that achieved in a realization of the MIMO detector presented in this thesis where $N_T = 4$ and $N_R = 4$.

from a complex constellation $\mathcal{O}$ (such as rectangular Quadrature Amplitude Modulation (QAM)) composed of $M = |\mathcal{O}| = 2^{M_c}$ distinct points meaning that every $M_c$ consecutive bits is mapped to a complex constellation point. In fact, this implies that $\tilde{s} \in \mathcal{O}^{N_T}$, where the index $n$ is removed hereafter for brevity. The transmission rate of the corresponding MIMO system, with $N_T$ transmit antennas in spatial multiplexing (SM) mode is then given by $R = N_T \log_2 M = N_T M_c$ bits per channel use (bpcu). For a fair comparison, which is independent of the number of transmit antennas and of the modulation scheme, the signal vector $\tilde{s}$ is normalized before transmission in such a way that the average transmitted power is one (i.e., $E\{\|\tilde{s}\|^2\}=1$).

The complex baseband equivalent model of the MIMO wireless channel that yields the $N_R$-dimensional received vector $\tilde{y} = [\tilde{y}_1, \ldots, \tilde{y}_{N_R}]^T$ is given by the following input-output relation

$$\tilde{y} = \tilde{H}s + \tilde{v}, \tag{2.1}$$

where $\tilde{H} = \{H_{ij}\}_{i=1}^{N_R} \_{j=1}^{N_T}$ denotes a $N_R \times N_T$ dimensional channel matrix representing the complex-valued channel gains between each transmit and each receive antenna and $\tilde{v} = [\tilde{v}_1, \ldots, \tilde{v}_{N_R}]^T$ represents the $N_R$ dimensional independent identically distributed (i.i.d) circularly symmetric complex zero-mean Additive White Gaussian Noise (AWGN) thermal noise vector with variance $\sigma^2$ per complex dimension, i.e.,
\( \tilde{v}_i \sim \mathcal{N}_c(0, \sigma^2). \) For simulation purposes, in this thesis, an i.i.d. Rayleigh fading channel model with no spatial correlation is assumed. Hence, the entries of \( \tilde{H} \) are chosen independently as zero-mean complex Gaussian random variables with variance one per complex dimension. The signal-to-noise-ratio (SNR) is defined as the ratio between the total transmitted power, which is normalized to one, and the variance of the thermal noise, i.e., \( \text{SNR} = \frac{1}{\sigma^2}. \)

The task of the MIMO detector at the receiver\(^2\) is to obtain the best possible estimate of the transmitted signal vector \( \tilde{s} \) in the Euclidean sense based on the received vector \( \tilde{y} \), i.e.,

\[
\hat{s} = \arg \min_{\tilde{s} \in \mathbb{C}^{N_T}} \| \tilde{y} - \tilde{H}\tilde{s} \|^2.
\] (2.2)

After being detected by the MIMO detector, the symbols are transformed back into their corresponding bit representations using the demapper block. Digital-to-Analog (D/A) and Analog-to-Digital (A/D) converters are used at the transmitter and receiver, respectively to convert the signals from digital to analog and vice versa. Note that some other blocks such as the channel estimator block, preprocessing block, as well as the lattice reduction block are also shown in Fig. 2.1 at the receiver. The channel estimator provides the estimate of the current channel status based on the pre-known transmitted pilot symbols. However, in this thesis we assume that the channel is perfectly known to the receiver. The task of the channel preprocessing block and the lattice reduction block will be discussed in Section 2.4 and Chapter 5, respectively.

In addition to the above complex model, the equivalent real model can also be derived using a real-valued decomposition (RVD) scheme [3]. However, in this thesis, in order to simplify the hardware implementation, a slightly different approach is used for the RVD scheme, which is more suitable for concurrent computations and the VLSI implementation. The real model of (2.1) can be written as

\[
y = Hs + v,
\] (2.3)

where \( y = [y_1, y_2, \cdots, y_{2N_R-1}, y_{2N_R}]^T \), \( s = [s_1, s_2, \cdots, s_{2N_T-1}, s_{2N_T}]^T \) and \( H \) are the equivalent real-valued vectors with the following mappings:

\(^2\)It is assumed that the receiver is provided with an accurate estimate of the channel \( \tilde{H} \), which can be obtained during a separate training phase with the aid of pilot symbols.
\[ y_{2k-1} = \mathcal{R}\{\tilde{y}_k\}, \quad y_{2k} = \mathcal{I}\{\tilde{y}_k\} \]
\[ s_{2k-1} = \mathcal{R}\{\tilde{s}_k\}, \quad s_{2k} = \mathcal{I}\{\tilde{s}_k\} \]
\[ v_{2k-1} = \mathcal{R}\{\tilde{v}_k\}, \quad v_{2k} = \mathcal{I}\{\tilde{v}_k\}, \]
\[(2.4)\]

and \( \mathbf{H} \) is derived from \( \tilde{\mathbf{H}} \) based on the following mapping

\[
\mathbf{H} = \begin{bmatrix}
\mathcal{R}(\tilde{H}_{11}) & -\mathcal{I}(\tilde{H}_{11}) & \cdots & \mathcal{R}(\tilde{H}_{1N_T}) & -\mathcal{I}(\tilde{H}_{1N_T}) \\
\mathcal{I}(\tilde{H}_{11}) & \mathcal{R}(\tilde{H}_{11}) & \cdots & \mathcal{I}(\tilde{H}_{1N_T}) & \mathcal{R}(\tilde{H}_{1N_T}) \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
\mathcal{R}(\tilde{H}_{N_R1}) & -\mathcal{I}(\tilde{H}_{N_R1}) & \cdots & \mathcal{R}(\tilde{H}_{N_RN_T}) & -\mathcal{I}(\tilde{H}_{N_RN_T}) \\
\mathcal{I}(\tilde{H}_{N_R1}) & \mathcal{R}(\tilde{H}_{N_R1}) & \cdots & \mathcal{I}(\tilde{H}_{N_RN_T}) & \mathcal{R}(\tilde{H}_{N_RN_T}) 
\end{bmatrix}_{2N_R \times 2N_T},
\]
\[(2.5)\]

where \( \mathcal{R}(\cdot) \) and \( \mathcal{I}(\cdot) \) denote the real and imaginary parts of a complex variable, respectively. Note that

\[
s_i \in \Omega = \left\{ \frac{-\sqrt{M}+1}{E_s}, \ldots, -1, \frac{1}{E_s}, \ldots, \frac{+\sqrt{M}-1}{E_s} \right\},
\]
\[(2.5)\]

where \( \Omega \) is the set of possible real entries in the constellation for in-phase and quadrature parts with \(|\Omega| = \sqrt{M}\), and \( E_s = 2(M - 1)/3 \) is the average symbol energy for an \( M\)-QAM constellation. The set \( \{\mathbf{H}s\} \) can be considered as the lattice \( \Lambda(\mathbf{H}) \) generated by \( \mathbf{H} \). The columns of \( \mathbf{H} \) are called basis vectors for \( \Lambda(\mathbf{H}) \), while the transmitted vector \( \mathbf{s} \) represents a lattice point. Another way to describe (2.2) is to say the objective of the MIMO Maximum-Likelihood (ML) detection method is to find the closest transmitted vector \( \hat{\mathbf{s}} \) based on the observation \( \mathbf{y} \), i.e.,

\[
\hat{\mathbf{s}} = \arg \min_{\mathbf{s} \in \Omega^{2N_T}} \| \mathbf{y} - \mathbf{H}s \|^2.
\]
\[(2.6)\]

The above definitions, imply that \(|\Omega|^{2N_T} = |\Omega|^{N_T}\) meaning that a complex \( N_R \times N_T \)
MIMO system can be modeled as a real $2N_R \times 2N_T$ MIMO system.

2.2 Processing Rates

From the system-level viewpoint, there are two categories of processing in the MIMO detection core.

- **Channel-rate processing** is often also referred to as preprocessing. The term comprises all operations that need to be carried out only when the channel estimate changes.

- **Symbol-rate processing** comprises all those operations that need to be carried out for each received symbol in order to estimate the transmitted vector symbol. We shall refer to this part of the receiver as the detector.

In practice, the channel can often be assumed to be constant over a large number of received symbols, so that the channel-rate processing is less critical. This assumption may, however, no longer hold in high-mobility scenarios, under stringent latency constraints, or in wide-band MIMO systems with frequency selective fading. Still it is justified, to consider the channel-rate processing complexity separate from the symbol-rate processing, as the frequency of the operation and the performance requirements are dictated by a completely different set of system parameters\(^3\).

2.3 Simulation Framework

The bit-error-rate (BER) results in this thesis have been obtained from computer simulations and/or tested chip measurements based on the i.i.d. channel model assumption. This model is valid in rich-scattering environments with sufficient spacing between the antennas (on the order of one wavelength) unless explicitly mentioned otherwise. It is further noted that all presented simulation results assume perfect channel knowledge at the receiver so that the channel estimation and detection can be separated. In terms of the modulation selection, the simulation results for all

\(^3\)In this thesis, the channel estimate is assumed to be valid over four consecutive received symbol vectors.
modulation schemes ranging from 4-QAM to 256-QAM\textsuperscript{4} are presented. However, for implementation purposes, 64-QAM was chosen for two reasons. First, most of the hardware implementations reported in the literature to-date focus on the 16-QAM scheme due to the higher complexity of the designs in 64-QAM constellation, which motivates us to fill this gap. Secondly, 64-QAM is chosen to be one of the mandatory supported constellations in several standards including IEEE 802.16e (WiMAX $2 \times 2$), IEEE 802.16m (WiMAX $4 \times 4$), IEEE 802.11n WLAN ($2 \times 2$ MIMO) and 3GPP LTE, which practically justifies its implementation. Both floating-point and fixed-point simulation results are presented and discussed throughout the dissertation.

2.4 Preprocessing Block

In order to reduce the computational complexity or to improve the BER performance of the detector, the channel matrix $\tilde{H}$ is commonly preprocessed in various practical MIMO detectors [11]. The basic idea of the preprocessing is to carry out the detection starting from the “strongest” signal down to the “weakest” signal, so that the error-propagation effect due to a wrongly-detected symbol is minimized\textsuperscript{5}. The preprocessing can be partitioned into two categories, i.e., based on the Zero-Forcing (ZF) criterion or Linear Minimum Mean Squared Error (LMMSE) criterion, according to the ordering by the postdetection SNR and the consideration of the channel noise level. Since the LMMSE criterion is known to have a better performance than the ZF criterion [3], we will limit most of our discussion to the LMMSE-based preprocessing, described in the following.

2.4.1 LMMSE-based Preprocessing

Consider the augmented channel matrix $[I\sqrt{\eta}\tilde{H}^T]^T$, with $\eta = \frac{\gamma}{N_T}$, where $\gamma$ represents the total transmit power at the transmitter. Let’s denote $P = (I + \eta\tilde{H}^H\tilde{H})^{-1}$. The algorithm proceeds with finding the minimum diagonal entry of $P$ and reordering the

\textsuperscript{4}The 256-QAM modulation scheme appears to be feasible for implementation as the required local oscillator’s phase noise specifications seem to be achievable for this constellation in the near future.

\textsuperscript{5}Here the terms “strong” and “weak” are a measure of the post-detection SNR based on the ZF and/or LMMSE criterion.
channel matrix followed by deflating the channel matrix by deleting the corresponding column. Then, a new matrix $P$ is computed with the deflated channel matrix and the process is repeated to find the next symbol to be detected. The complexity of the (optimal ordering) algorithm described above is $O(N_T^4)$. The repeated calculation of the pseudo-inverse of the augmented channel matrix, $P$, accounts for most of the computation load. This repeated computation can be avoided by using the square-root algorithm proposed in [12] with a complexity of $O(N_T^3)$. Further reduction in complexity is possible using the steps outlined in [13]. Alternatively, MMSE decoding based on the sorted QR-decomposition has been proposed in [14] and an MMSE-based lattice reduction scheme has been proposed in [15].

It is worth noting that in slowly-varying channels, these computations are performed only once at the beginning of each block, and hence form only a small fraction of the overall computations, which are dominated by the detection process. Therefore, in what follows in this thesis, we focus only on reducing the computational complexity of the MIMO detection scheme and we assume that the preprocessing block has been implemented in the preceding stages. Moreover, all of the simulation results presented in this thesis are based on the preprocessing block proposed in [12].

### 2.5 MIMO Detection Schemes

For spatial multiplexing schemes, we assume that the channel matrix $\tilde{H}$ is perfectly known at the receiver. Therefore, the task of a MIMO detector is to provide the decision (either hard or soft as described below) on transmitted symbol $\tilde{s}$ given the received signal $\tilde{y}$. Such a MIMO detection problem also shows up in other setups, including the multi-user detection [16], filter banks [17], modulated coding [18], and multi-carrier CDMA schemes [19]. Thus the solution to the MIMO detection problem can also offer benefits to designing these systems.

There are two classes of MIMO detectors: *hard-decision* detectors and *soft-decision* detectors. The first one is useful for detecting uncoded transmissions, where the decision of MIMO detectors will be used as the final decision. A soft-decision detector, however, is normally used in coded MIMO systems, where an iterative detection and decoding scheme needs soft information being exchanged between detection and decoding modules following the “turbo principle”, see e.g., [20]. In this thesis, we focus
on the hard detection problem as most of the underlying challenges in the VLSI implementation is the same for both detectors. Moreover, the extension of the hard-decision scheme to the soft version is shown to be straightforward in [3].

As shown in Fig. 2.2, the current MIMO detection schemes can be listed within the context of the following main categories:

- Exhaustive search Maximum-Likelihood (ML) detection.
- Sub-optimal linear receivers (ZF, MMSE).
- Sub-optimal non-linear receivers (V-BLAST, SIC, ...).
- Near-optimal non-linear receivers (Sphere Decoder (SD), K-Best).

The focus of this thesis, i.e., the K-Best detector, is highlighted with a gray box in the Fig. 2.2.

### 2.5.1 ML Detection

Denoting the alphabet size of the scalar complex constellation transmitted from each antenna by $M$, the ML detector needs to search over a total of $M^{N_{T}}$ vectors rendering the complexity exponential in the number of transmit antennas. It has been
shown that the implementation of the exhaustive-search ML is feasible in low-rate schemes, where the number of bits per channel use (bpcu) is less than eight [21]. However, the complexity of ML detection becomes quickly unfeasible to implement as the transmission rate per channel use or the number of antennas increases\(^6\) [22].

### 2.5.2 Linear Detectors

Linear MIMO detection methods formulate the detection problem in a MIMO system as a linear estimation problem, which can be solved according to a least-square (i.e., ZF or MMSE) criterion. To this end, corresponding receivers try to reverse the effect of the channel by multiplying the received signal vector \(\mathbf{y}\) with an estimator matrix \(\mathbf{G}\) to obtain

\[
\hat{x} = \mathbf{Gy},
\]

which is an unconstrained estimate of the transmitted signal vector \(\mathbf{s}\). This estimate completely ignores the fact that the entries of \(\mathbf{s}\) are known to be constrained to the limited set of constellation points \(\mathcal{O}\). Hence, the actual detection process (i.e., the mapping to a valid constellation point) requires an additional step in which slicing is performed independently on each of the entries \(\hat{x}_i\) of \(\hat{x}\) to obtain the nearest constellation points according to

\[
\hat{s}_i = Q(\hat{x}_i),
\]

where \(Q(\cdot)\) denotes the slicing operator for a given modulation scheme. The main drawback of linear detection schemes is that they can only achieve a diversity order of \(N_R - N_T + 1\) [23], which translates to a poor BER performance result. The impact of that lack of diversity becomes especially apparent in a symmetric system configuration with \(N_T = N_R\) where the corresponding poor BER performance at high SNR is clearly visible. In brief, sub-optimal linear detectors include linear ZF and linear MMSE detectors [24], [25], described in the sequel.

**A. Zero-Forcing Detector:**

\(^6\)For instance, in the case of a 4 \(\times\) 4, 64-QAM MIMO system, the number of bpcu is \(4 \times 6 = 24\), which is not a suitable framework for the ML detector.
In a ZF detector, the estimator matrix/filter can be written as

$$G_{ZF} = (H^H H)^{-1} H^H,$$  \hspace{1cm} (2.9)

which is the Moore-Penrose pseudo-inverse of the channel matrix [26], [27]. Each element of the filter output vector

$$\hat{x}_{ZF} = G_{ZF} y = s + (H^H H)^{-1} H^H v$$  \hspace{1cm} (2.10)

is mapped onto the symbol alphabet by a minimum distance quantization. The estimation error corresponding to the main diagonal elements of the error covariance matrix is

$$E\{(\hat{x}_{ZF} - s)(\hat{x}_{ZF} - s)^H\} = \sigma^2 (H^H H)^{-1},$$  \hspace{1cm} (2.11)

which equals the covariance matrix of the noise after the receive filter. Obviously, the small eigenvalues of $H^H H$ (when $H$ is close to singular) will lead to a large error due to the noise amplification. The performance of a ZF detector is thus far from optimum especially for ill-conditioned channels. In fact, in the ZF scheme, the interference signals can be completely suppressed if the number of receive antennas is equal to or greater than the number of transmit antennas. Thus, ZF is widely used in the high-SNR region where interference is a dominant factor.

**B. MMSE Detector:**

The problem of noise enhancement of zero-forcing can be addressed by including the noise term in the design of the filter matrix $G$. This is done by the MMSE detection scheme, which minimizes the mean squared-error between the actual transmitted symbols and the output of the linear detector [16]. The MMSE estimator filter can be written as

$$G_{MMSE} = (H^H H + \sigma^2 I_{N_T})^{-1} H^H,$$  \hspace{1cm} (2.12)

which represents a tradeoff between the noise amplification and interference
Figure 2.3: The comparison of various sub-optimal detectors with the ML detector in a $4 \times 4$ system with 16-QAM modulation.

suppression. The output of the resulting MMSE detector is given by

$$\hat{x}_{\text{MMSE}} = G_{\text{MMSE}} y = (H^H H + \sigma^2 I_{N_T})^{-1} H^H y,$$  \hspace{1cm} (2.13)

and the error covariance matrix is found to be

$$E\{(\hat{x}_{\text{MMSE}} - s)(\hat{x}_{\text{MMSE}} - s)^H\} = \sigma^2 (H^H H + \sigma^2 I_{N_T})^{-1}.$$ \hspace{1cm} (2.14)

The MMSE detector offers a better performance over the ZF detector, however, it is still far from optimum. Iterative MMSE receivers ([28], [29], [30]) have been considered for their simplicity and improved performance but their performance results are not close to the ML.

Although linear receivers can greatly reduce the computational complexity, they suffer from a significant performance loss (see Fig. 2.3 for a $4 \times 4$ system with 16-QAM modulation). Non-linear detectors can be used to improve the performance.
2.5.3 Non-linear Detectors

Sub-optimal Non-linear Receivers

Two examples of sub-optimal non-linear receivers are as follows:

- Successive Interference Cancelation (SIC) with iterative least squares [31].
- BLAST nulling/cancelling [32].

A. SIC Detector:

SIC is based on the previously described linear estimation algorithms. However, a nonlinear interference cancelation stage partially exploits the knowledge that the entries of the transmitted vector have been chosen from a finite set of constellation points $\mathcal{O}$. To this end, the symbols of the parallel data streams are no longer all detected at once. Instead, they are considered one after another and their contribution (after slicing and remodulation) is subtracted (removed) from the received vector before proceeding to detect the next stream. This process is performed iteratively. Compared to the linear detection schemes, SIC achieves an increase in diversity order with each iteration. While the first detected stream still sees a diversity order of $N_R - N_T + 1$, the second has already a diversity order of $N_R - N_T + 2$ and so forth. Unfortunately, the overall average BER performance is dominated by the stream that is detected first and error propagation also has a considerable impact on the performance of the subsequent streams. Hence, the detection order is important to improve the BER performance [31]. The Bell Labs Layered Space-Time (BLAST) scheme, described in the following, is one famous example of the SIC approach with a detection order.

B. BLAST Detector:

For a better performance than simple linear detectors, a successive interference cancelation technique can be used. Bell Labs Layered Space-Time (BLAST) is one famous example based on both successive cancelation and zero nulling principles [32], [33]. In the BLAST detector, the symbols are not detected in parallel as in ZF or MMSE detectors. Instead, they are detected consecutively one after another. Consider the complex domain and assume the symbols are
detected in the order of $k_1, k_2, \ldots, k_{N_T}$, which is a permutation of the integers $1, 2, \ldots, N_T$. To detect the $k_i$-th symbol $(\tilde{s}_{k_i})$, the interference from all the symbols other than the $k_i$-th symbol should be perfectly suppressed. This can be accomplished by linearly weighting the received signal vector with a zero-forcing nulling vector. In other words, in order to detect symbol $\tilde{s}_{k_i}$, the nulling vector $\tilde{w}_{k_i}$ has to be orthogonal to $\tilde{h}_l$, the $l$-th column of $\tilde{H}$, for $l > k_i$ as

$$\tilde{w}_{k_i} \cdot \tilde{h}_l = \begin{cases} 1 & l = k_i \\ 0 & l > k_i \end{cases}. \quad (2.15)$$

Using the above nulling process, the BLAST detector generally proceeds as follows:

1. Set $\tilde{y}_{k_1} = \tilde{y}$.
   For $k_1, k_2, \ldots, k_{N_T}$ perform the following steps:

2. Find $z_{k_i}$ based on:
   $$\tilde{z}_{k_i} = \tilde{w}_{k_i}^T \tilde{y}_{k_i}. \quad (2.16)$$

3. Obtain $\hat{s}_{k_i}$ by quantization:
   $$\hat{s}_{k_i} = Q(\tilde{z}_{k_i}). \quad (2.17)$$

4. Assume that $\tilde{s}_{k_i}$ is the right estimate of $\tilde{s}_{k_i}$, cancel the contribution of $\tilde{s}_{k_i}$ from signal vector $\tilde{y}_{k_i}$, resulting in the updated received signal vector $\tilde{y}_{k_i+1}$:
   $$\tilde{y}_{k_i+1} = \tilde{y}_{k_i} - \tilde{s}_{k_i} \tilde{h}_{k_i}. \quad (2.18)$$

5. If $i \neq N_T$, set $i \leftarrow i + 1$ and go to step (b).

The derivation of the linear nulling filter vector $\tilde{w}_{k_i}$ can be based on ZF to maximize the SNR after the interference cancelation in each search step, or based on MMSE to maximize the signal-to-interference-and-noise-ratio (SINR). For example, if ZF is used, the nulling filter vector $\tilde{w}_{k_i}^T$ is the $k_i$-th row of the Moore-Penrose pseudo-inverse matrix in (2.9). Once each symbol $\tilde{s}_{k_i}$ is detected, the channel matrix $\tilde{H}$ will also be updated by zeroing the corresponding column.
\[ \tilde{h}_k. \] In this way, after the first \( i \) symbols are detected, the updated channel matrix corresponds to an equivalent system with \( N_T - i \) transmit antennas and \( N_R \) receive antennas. Note that the linear nulling filter vector is derived from the updated channel matrix in each interference cancelation step.

The detection order of the symbols significantly affects the error probability of the BLAST detector [32]. In order to achieve the best performance, it is optimal to start the detection process from the symbol with the smallest estimation error, or equivalently the largest SNR after linear nulling of the interferences [32]. For instance, in the ZF-based BLAST, the first symbol to start with, \( \tilde{s}_{k_1} \), can be identified to be the one associated with the nulling filter vector that has the lowest Euclidean norm, because this vector causes the smallest noise enhancement. Once \( \tilde{s}_{k_1} \) is detected and the channel matrix is updated by zeroing \( \tilde{h}_{k_1} \), the second symbol to be detected can be identified according to the nulling filter vector norms derived based on the updated channel matrix (see [32], [33] for more detail).

C. BLAST with QR-decomposition:

In [14], [34], and [35], it has been shown that the BLAST detector, mentioned above, can be implemented using the QR-decomposition of the channel matrix. Considering the complex-domain implementation, channel matrix \( \tilde{H} \) can be written as

\[
\tilde{H} = \tilde{Q} \tilde{R}, \tag{2.19}
\]

where \( \tilde{Q} \) is a unitary matrix of size \( N_R \times N_T \) and \( \tilde{R} = \{\tilde{r}_{i,j}\} \) is an upper triangular \( N_T \times N_T \) matrix. Performing the nulling operation by \( \tilde{Q}^H \) results in

\[
\tilde{z} = \tilde{Q}^H \tilde{y} = \tilde{R} \tilde{s} + \tilde{w}, \tag{2.20}
\]

where \( \tilde{w} = \tilde{Q}^H \tilde{v} \). Since the nulling matrix is unitary, the noise, \( \tilde{w} \), remains spatially white. Due to the upper triangular structure of \( \tilde{R} \), the \( k \)-th element of \( \tilde{z} \) is

\[
\tilde{z}_k = \tilde{r}_{k,k} \tilde{s}_k + \sum_{i=k+1}^{N_T} \tilde{r}_{k,i} \tilde{s}_i + \tilde{w}_k, \tag{2.21}
\]
where $\tilde{r}_{i,j}$ represents an element of $\tilde{R}$, and the diagonal elements $(\tilde{r}_{k,k})$ are all real numbers. Thus $\tilde{z}_{N_T}$ is free of interference and can be used to estimate $\tilde{s}_{N_T}$ after scaling with $1/\tilde{r}_{N_T,N_T}$. Using this detected symbol, we proceed with $\tilde{z}_{N_T-1}, \cdots, \tilde{z}_1$ one after the other, where the interference can be perfectly removed in each detection step assuming previous decisions are correct. Again, the detection order is crucial due to the error propagation.

The BLAST detectors have total complexity in the order of $O(N_T^2)$ to $O(N_T^3)$. Note that this complexity can greatly increase if the channel coherence time is too small. This translates to more frequent channel preprocessing to find the detection ordering as well as QR-decomposition, which is as a result of fast channel variations. This eventually means that the complexity of the BLAST with QR-decomposition can have higher complexity than the non-linear receivers in fast varying channels. Although BLAST detectors outperform the linear receivers, they still reveal a considerable performance gap from the ML detector (Fig. 2.3).

**Near-optimal Non-linear Lattice Decoders**

Lattice decoders are another family of receivers, which have near-optimal detection performance. They can actually trace their roots back to the theory and algorithms developed for solving the shortest/closest lattice vector problem for integer programming applications. The noiseless received signal vector can be interpreted as a point of the lattice spanned by $H$, where the columns of the channel matrix are the bases of the lattice. Considering the effect of a Gaussian noise, we obtain the optimum estimate for the transmitted symbol vector if we can find the closest point in the lattice constellation with the minimum Euclidian distance$\footnote{This is because this measure is optimal for Gaussian noise.}$ to the received signal vector $y$. If the wireless signal is transmitted in a rich scattering environment, channel entries tend to be independent random variables and the lattice bases become less correlated, meaning that each transmitted symbol has a more unique spatial signature. Intuitively, it is easier to perform detection (i.e., differentiate the lattice points from each other) if the lattice basis are close to orthogonal.

If the lattice bases are orthogonal, the closest point search becomes extremely easy. However, since the lattice basis are built with the wireless channel matrix and is in
general completely arbitrary, the complexity of the closest point problem has been shown to be NP-hard. All known algorithms for solving this problem optimally have exponential complexity with the degree of freedom in the lattice. Basically, the lattice search problem can be reformulated into a tree-search problem.

*Tree pruning* is the key to the complexity reduction in tree-search algorithms. The fundamental idea is to reduce the number of leaves that must be considered in the search for the solution of the ML detection problem by pruning the entire subtrees that are unlikely to lead to the desired solution. The decision, whether a node should be pruned together with all its children is normally made based on a performance metric (here, based on its Euclidean distance to the received signal). Depending on how they carry out the non-exhaustive search through the tree pruning, near-optimal non-linear lattice detectors generally fall into two main categories [36].

- Depth-first methods, such as Sphere Decoding (SD) [37], [38], [39].
- Breadth-first methods such as the K-Best algorithm a.k.a. M-Algorithm [40].

**A. Depth-First Tree Traversal:**

Depth-first tree-search is a recursive scheme, which starts from the root and traverses the tree in both forward and backward directions. As opposed to a breadth-first search, the algorithm first explores all admissible children of a parent node before visiting the admissible siblings of that parent node. In other words, the algorithm first tries to identify an admissible child of the current node that has not been visited yet. If such a child exists, it is chosen as the new parent node. If no child is admissible, or if all children of a node have already been visited, the decoder returns to the parent of the current node and considers the remaining admissible children thereof.

Sphere decoding (SD) [38] is the most attractive depth-first approach that fits well into the framework of tree-search algorithms. The fundamental idea is to reduce the number of candidate vector symbols that need to be considered in the search for the ML solution. To this end, the search is constrained to only those candidate vector symbols \( \mathbf{s} \) for which \( \mathbf{Hs} \) lies inside a hyper-sphere with radius \( r \) around the received point \( \mathbf{y} \). The corresponding inequality is given by

\[
\| \mathbf{y} - \mathbf{Hs} \|^2 < r^2. \tag{2.22}
\]
The radius $r$ is referred to as the *sphere constraint*. However, so far, the challenge has merely shifted from solving (2.6) to identifying the candidate vector symbols that meet the sphere constraint (2.22). Complexity reduction through tree pruning is enabled by realizing that the sphere constraint can be applied to identify admissible nodes on all levels because it is known that if any node within the search tree violates the constraint, all of its children and eventually also the corresponding leaves will also violate the sphere constraint. This concept is shown in Fig. 2.4, where it is assumed the branches, which do not violate the constraint are depicted inside the hyper-sphere.

In principle, SD can be performed by traversing the tree breadth-first or depth-first. However, with respect to its implementation, a strict breadth-first search has two major disadvantages: The first problem is associated with the need to choose an appropriate initial radius. If it is chosen too small, no candidate vector symbol may meet the constraint and the algorithm must be restarted with a larger radius. If the radius is chosen too large, a considerable number of candidate vector symbols could meet the constraint and the complexity will be high. The second problem is a consequence of the inability to determine a radius that guarantees that the number of nodes meeting the constraint is low. Thus, it may happen that all nodes on the level just before the leaves meet the constraint. To cover this worst-case scenario, an implementation that does not compromise BER performance must provide considerable amounts of memory to be able to store all nodes on that level, before it can proceed to the next level. Taking the depth-first SD mode, results in reduced memory requirements.
and the fact that the depth-first algorithm quickly identifies candidate vector symbols that meet an initial radius constraint. In fact, this early identification of possible solutions alleviates the problem of initial radius choice and leads to a significant complexity reduction [11]. Its main disadvantage is that it results in a throughput dependent on the SNR value, as SNR determines the sphere constraint [41].

The performance of SD is ML under the assumption of unlimited execution time [42] at a lower average computational complexity than the ML method. However, in [43] it has been shown that contrary to the popular belief that the expected complexity of the sphere decoder is polynomial in terms of the number of transmit antennas, for a given SNR and constellation size, its average complexity is exponential in the number of transmit antennas. Moreover, the actual runtime of the algorithm is dependent not only on the channel realization, but also on the operating SNR. Thus leading to a variable throughput rate, which results in an extra overhead in the hardware due to the extra required I/O buffers and lower hardware utilization.

B. Breadth-First Tree Traversal:

Breadth-first tree traversal is a nonrecursive scheme, which starts from the root and traverses the tree in forward direction only. On each level, the algorithm visits all admissible nodes and considers their associated children to construct a new set of admissible nodes on the next level before it proceeds. In each level, a subset of all visited nodes are chosen as the surviving admissible nodes based on a criterion (e.g., their Partial Euclidean Distance (PED) from the received symbol). For the final level, the examined children, corresponding to the admissible leaves, consists of a set among which the decoder finally searches for the solution of (2.6).

Among the breadth-first search methods, the most well-known approach is the K-Best algorithm [44]. The K-Best detector guarantees a fixed SNR-independent throughput with a performance close to ML. Being fixed-throughput in nature along with the fact that the breadth-first approaches are feed-forward detection schemes with no feedback, makes them especially attractive for VLSI implementation. The MIMO detector proposed in this thesis is based on the K-Best algorithm, which will be addressed in Chapter 3 and Chapter 4.
2.6 Antenna Correlation

The diversity and array gain intrinsic to MIMO systems, are based on the assumption that the transmit antennas are independent and uncorrelated at the transmitter and receiver. The violation of this condition may result in some degradation in the BER performance. The correlation between the antennas is caused because of the physical configuration of the antennas. For instance, in WiMAX systems, there are four different defined antenna correlations, which specifies various levels of correlation in the antennas, i.e., no, low, medium, and high correlation. In fact, if the antennas are spaced less than $\lambda/2$, where $\lambda$ is the signal wavelength, the antennas experience non-zero correlation. The actual amount of correlation also depends on their physical configuration with respect to one another.

In order to simulate the transmission over correlated MIMO channels, normally the popular Kronecker model [45] is used as follows:

$$\tilde{H} = \Phi_T^{1/2} \tilde{B} \Phi_T^{1/2},$$

(2.23)

with $\tilde{B}$ consisting of uncorrelated complex Gaussian coefficients $\tilde{g}_{i,j}$ of unit variance. According to the correlation model presented in [46], the spatial correlation matrix at the transmitter $\Phi_T = E\{\tilde{H}^H \tilde{H}\}$ and at the receiver $\Phi_R = E\{\tilde{H} \tilde{H}^H\}$ can be modeled as a function of the correlation coefficient $0 \leq \rho_T, \rho_R \leq 1$. Using their definition, the $N_T \times N_T$ correlation matrix at the transmitter is given by the Toeplitz matrix:

$$\Phi_T = \begin{pmatrix}
1 & \rho_T & \rho_T^2 & \cdots & \rho_T^{(N_T-1)^2} \\
\rho_T & 1 & \rho_T & \cdots & : \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
\rho_T^{(N_T-1)^2} & \cdots & \rho_T^4 & \rho_T & 1
\end{pmatrix},$$

(2.24)

and a corresponding definition holds for the $N_R \times N_R$ matrix $\Phi_R$ with coefficient $\rho_R$. The correlation model can be further simplified by assuming $\rho_R = \rho_T = \rho$, yielding a single parameter model [46]. The given model can range from the uncorrelated case ($\rho = 0$) to the fully correlated scenario ($\rho = 1.0$). In this thesis, the simulation results for both the uncorrelated and correlated antenna will be presented and discussed in Chapter 3 and Chapter 5.
3 The K-Best MIMO Detection Algorithm

3.1 Introduction

The problems in (2.2) and (2.6) can be thought of as the detection problems on a tree with complex and real nodes, respectively. These two trees for a $2 \times 2$ 4-QAM MIMO system have been shown in Fig. 3.1. As shown, the real tree is twice as deep, which translates to a larger latency in the hardware implementation. On the complex tree, however, the number of possible children to be expanded per parent is twice and the sorting per level is more complicated. Moreover, all the operations including the Euclidean distance calculation in all levels are in the complex domain. Depending on the objectives and the specifications of the targeted MIMO detector core, both the real implementation [3], [47] and the complex implementation [41] have been addressed in the literature. Due to the benefits of the real-domain implementation, which will be address in Section 3.4, almost all the K-Best schemes to date are in the real domain. In this thesis, we propose a novel framework to implement the K-Best algorithm both in the real domain as well as the complex domain.

3.2 K-Best Algorithm

3.2.1 Theory

Consider a $N_R \times N_T$ M-QAM MIMO system. The detection problem of such a system can be formulated as a tree-search problem with $N_T$ levels in the complex domain and $2N_T$ levels in the real domain through the RVD scheme. Therefore, given an implementation in the real-domain, the problem in (2.6) can be considered as a tree-search problem with $2N_T$ levels. The K-Best algorithm explores this tree from the root to the leaves by expanding each level and selecting the $K$ best candidates in each
The K-Best MIMO Detection Algorithm

level, which are called the surviving nodes of that level based on a criterion [48]. To make this clearer, let’s consider $K$ surviving nodes in level $i$. Each of these nodes has $\sqrt{M}$ possible children in level $i + 1$, from the symmetry in the M-QAM constellation. The K-Best algorithm visits all these children and calculates their Partial Euclidean Distances (PEDs) resulting in $K\sqrt{M}$ children at level $i + 1$. Once the PED values are calculated, the K-Best algorithm sorts all these $K\sqrt{M}$ children and selects the $K$ best children as the surviving nodes in level $i + 1$ (see Fig. 3.2, which is a simple example for $M = 16$ and $N_T = N_R = 2$). The K-Best algorithm is a feed-forward detection method proceeding in the forward direction only. This method offers a trade-off between optimality and complexity with respect to the value of $K$ [44], [49]. Thus an appropriate value of $K$ should be determined using extensive simulations for each system framework and/or application.

Consider the problem in (2.6), and let’s denote the QR-decomposition of the channel matrix as $H = QR$, where $Q$ is a unitary matrix of size $2N_R \times 2N_T$ and $R$ is an upper triangular $2N_T \times 2N_T$ matrix. Performing the nulling operation by $Q^H$ results in

$$z = Q^H y = Rs + w,$$

where $w = Q^H v$. Since the nulling matrix is unitary, the noise, $w$, remains spatially white and the norm vector in (2.6), which represents the ML detection rule, can be rewritten as [11]

$$\hat{s} = \arg \min_{s \in \Omega^{2N_T}} \| z - Rs \|^2.$$
Exploiting the upper triangular nature of $\mathbf{R}$, this norm vector can be further expanded as

$$\tilde{s} = \arg \min_{\mathbf{s} \in \Omega^{2N_T}} \sum_{l=1}^{2N_T} |z_l - \sum_{j=l}^{2N_T} r_{lj}s_j|^2,$$

which can be thought of as a tree-search problem with $2N_T$ levels. Due to the upper-triangular structure of matrix $\mathbf{R}$, the K-Best algorithm starts from the last row of the matrix (detection tree), i.e., row(level) $l = 2N_T$. Therefore, equation (3.3) can be evaluated in a recursive manner as follows.

$$T_l(s^{(l)}) = T_{l+1}(s^{(l+1)}) + |e_l(s^{(l)})|^2$$

(3.4)

$$e_l(s^{(l)}) = z_l - \sum_{j=l}^{2N_T} r_{lj}s_j = L_l(s^{(l)}) - r_ll,$$

(3.5)

where $s^{(l)} = [s_l \ s_{l+1} \cdots s_{2N_T}]^T$, $T_l(s^{(l)})$ is the accumulated partial Euclidean distance (PED) with $T_{2N_T+1}(s^{(2N_T+1)}) = 0$, $|e_l(s^{(l)})|^2$ denotes the distance increment between
Table 3.1: The K-Best Algorithm.

1) Initialization: Set one path at level $2N_T + 1$ with PED = 0.

2) Expansion: Expand each of the $K$ surviving paths from the previous level to $\sqrt{M}$ new possible children in $\Omega$ and calculate the updated PED of all $K\sqrt{M}$ resulting paths.

3) Sorting: Sort all $K\sqrt{M}$ existing paths according to their accumulated PEDs and select the best $K$ paths.

4) If not at the last level of the tree, then go to step 2, otherwise announce the path with the lowest PED as the detected path.

two successive nodes/levels in the tree, and

$$L_i\left(s^{(l)}\right) = z_l - \sum_{j=l+1}^{2N_T} r_{lj}s_j = r_U(\bar{z}_l - \sum_{j=l+1}^{2N_T} \bar{r}_{lj}s_j) = r_U\bar{L}_i\left(s^{(l)}\right), \quad (3.6)$$

where $\bar{z}_l$, and $\bar{r}_{lj}$ denote the scaled $z_l$ and $r_{lj}$ by $r_U$, respectively, (i.e., $z_l = \bar{z}_l r_U$, and $r_{lj} = \bar{r}_{lj} r_U$). Based on the above formulation\(^1\), the K-Best algorithm can be described as in Table 3.1.

The path with the lowest PED at the last level of the tree is the hard-decision output of the detector, whereas, for a soft-decision output, all of the existing paths at the last level are considered to calculate the Log-Likelihood Ratios (LLRs).

### 3.2.2 Challenges

Regardless of dealing with the hard-decision or soft-decision output, there are two main computations in the above algorithm that play critical roles in the total computational complexity of the algorithm, namely, (i) the expansion of the surviving paths and (ii) the sorting. Therefore, the important part of any VLSI realization of

\(^1\)A typical detection core for MIMO systems consists of a preprocessing core whose output is the $R$ matrix. In this thesis, we assume that the sorted QR-decomposition algorithm according to [34] is applied to the channel matrix, which then serves as a primary input to our ASIC.
the K-Best algorithm is to propose an efficient architecture to implement these two computation cores, described in the following.

1) **Expansion**: The K-Best algorithm enumerates all the possible children of a parent node in each level. Since there are $K$ parent nodes at each level and $\sqrt{M}$ children per parent, thus $K\sqrt{M}$ children needs to be computed in each level, which incurs a large computational complexity\(^2\). For instance, in the case of 64-QAM with $K = 10$, a total of 80 children need to be expanded and their Euclidean distances from the received vector should be calculated. The Phase Shift Keying (PSK) enumeration scheme [50], which is based on the search over multiple base-centric circles, or its simplified version for $M$-QAM systems, [41], have been proposed to simplify the enumeration process towards a more efficient implementation. Moreover, in [51], a different flavor of the base-centric search methodology is used, in which the joint SD algorithm and successive interference cancelation are employed. A relaxed K-Best enumeration scheme is also proposed in [47] based on the PSK enumeration idea with local sorters. Although these methods are simpler to implement, they do not linearly scale with the constellation size (such as [50]) and/or have performance loss compared to the strict K-Best implementation\(^3\) (such as [47] and [51]). In this thesis, we propose an efficient expansion method called the on-demand expansion scheme, which avoids the enumeration of all the possible children of a parent while providing all the information required for the exact K-Best implementation without any performance degradation. To the best of our knowledge, the proposed scheme in this thesis is the only expansion scheme to-date with a computational complexity independent of the constellation size.

2) **Sorting**: Based on the algorithm in Table 3.1, in each level of the tree there are $K\sqrt{M}$ children to be sorted. In general, using the parallel sorting algorithms\(^4\), sorting a list of $K\sqrt{M}$ numbers has a complexity of $O(K^2M)$. Among all the sorting algorithms addressed in [52], bubble sorting is the most effective one, which distributes the sorting over multiple cycles [44]. Using bubble sorting, it takes $K\sqrt{M}$ cycles to obtain the sorted list of the children in each level, which is time-intensive for large

\(^2\)In some implementations, a metric such as a radius constraint is used to limit the number of expanded children [3]. In this thesis, we consider a general K-Best implementation without a metric restriction.

\(^3\)Strict K-Best refers to the K-Best algorithm implementation without any approximation and/or modification, which may result in performance loss.

\(^4\)A sequential sorting algorithm has a computational complexity of $K\sqrt{M}\log(K\sqrt{M})$.
values of $K$ and $M$. For instance, for $K = 10$, and $M = 64$, it takes $K\sqrt{M} = 80$
cycles to perform the sorting. In [2] a distributed sorting method is proposed based on
the Schnorr-Euchner (SE) ordered search technique [42], [53]. However, the proposed
approach requires all the children of a parent node to be calculated by a Metric
Computation Unit (MCU) and is applicable only for $K \leq \sqrt{M}$, and thus cannot be
applied to $K = 10$, and $M = 64$. Furthermore, it cannot be extended to the complex
equivalent model as the distributed sorting scheme uses the natural ordering of the
integer numbers on the real axis. Moreover, for higher values of $K$, the proposed
single-cycle merge core in [2] becomes increasingly complex resulting in a long critical
path. Therefore, [2] is not a suitable platform to achieve high throughput for higher
order modulations like 64-QAM and 256-QAM where the value of $K$ is large (e.g., $K =
10$ for 64-QAM and $K = 15$ for 256-QAM). The detail of the proposed architecture
in [2] will be addressed in Chapter 4 and will be compared with the proposed scheme
in this thesis. An improved K-Best approach was also proposed in [54], where the
number of children to be sorted is reduced and the sorting is also distributed such that
for $K < \sqrt{M}$ the number of children to be sorted is reduced from $K\sqrt{M}$ to $K^2$ (still
complex to implement). However, there is no simplification when $K > \sqrt{M}$. Finally
in [47], a relaxed approach to implement the K-Best algorithm using a distributed
sorting scheme is proposed, where the entire range of children is divided into a certain
number of groups and the sorting is only performed at the group level. This approach
is simpler to implement but associated with it is a performance loss compared to the
strict K-Best approach. Moreover, the implemented Application-Specific Integrated
Circuit (ASIC) occupies large silicon area while having moderate throughput. A
modified version of the above distributed sorting idea based on a node interleaving
scheme is also presented in [55].

In this thesis, we propose a distributed sorter, working in a pipelined fashion with
the proposed on-demand expansion scheme, which finds the $K$ best candidates in $K$
clock cycles. It works for any value of $K$ and $M$ and its complexity is independent
of the constellation size while scaling linearly with the value of $K$. It also does not
compromise the performance and provides the exact K-Best results. Moreover, the
proposed idea can be easily extended to the complex domain [56].

Both the on-demand expansion and the distributed sorting scheme in this thesis
are used to design and fabricate an ASIC implementation for a $4 \times 4$ 64-QAM MIMO
hard detector achieving the highest throughput reported to date in the literature
while resulting in the strict K-Best performance with no BER loss.

Note that $4 \times 4$ systems have found applications in next-generation standards such as IEEE 80.16m, which requires the implementation of a $4 \times 4$, 64-QAM MIMO system. On the implementation side, even if the handsets might have at most two antennas located on their body, with the so-called collaborative MIMO architecture proposed in WiMAX systems [57], [58] the whole system can be considered as a higher-order MIMO system. For instance, if two users transmit at the same time and in the same frequency band with two antennas each, and assuming four antennas at the base station, the overall system can be thought of as a $4 \times 4$ MIMO system.$^5$

3.3 Proposed On-demand Expansion and Distributed Sorting for the K-Best Algorithm

3.3.1 Real Domain

Consider a $2N_R \times 2N_T$ real-model MIMO system with channel matrix $H$ (see (2.3)). As mentioned, the detection problem can be thought of as a tree-search problem in a tree with $2N_T$ levels, $K$ nodes per level and $\sqrt{M}$ children per node. Because of the upper triangular structure of matrix $R$, the algorithm starts from the last row of the matrix ($2N_T$-th row, which is the $2N_T$-th level of the detection tree) and goes all the way up to the first row of the matrix, which is the first level of the detection tree. An example of this interpretation is shown in Fig. 3.2 for $M = 16$ and $N_T = N_R = 2$, which implies $2N_T = 4$ levels in the tree. In each level, there are $K$ surviving nodes, called the parent nodes, and there are $\sqrt{M} = 4$ children per parent node. After the expansion of all parent nodes, the resulting $K\sqrt{M} = 4K$ nodes are sorted among which the best $K$ nodes with the lowest PED are selected and are claimed as the parent nodes of the next level. This process is repeated for all levels. Note that in this scheme, all the possible children of a level are expanded exhaustively. This exhaustive expansion grows significantly with the constellation.

$^5$In the IEEE802.16e standard, each user in the uplink transmits using one of its antennas. However, using the collaborative MIMO scheme at the uplink, the system can be considered to be a $2 \times 2$ MIMO system as there are two antennas at the base station.
The K-Best MIMO Detection Algorithm

Table 3.2: Distributed K-Best Algorithm.

1) Find the K-Best children of level $2N_T (K_{2N_T})$.
   For $l = 2N_T - 1 : -1 : 1$
2) Find the first child of all candidates in $K_{l+1}$. Call this set $C_l$.
3) For $k = 1 : K$
   3.1) Select the child in $C_l$ with the lowest PED.
   3.2) Announce this child as the next K-Best candidate in level $l$ (i.e., add it to $K_l$).
   3.3) Replace it with its next best sibling.
End
End

Therefore, more clever ways are needed to calculate the $K$ best candidates of each level without performing an exhaustive search.

Let’s consider level $l$ of the tree and assume that the set of K-Best candidates in level $l + 1$ (denoted by $K_{l+1}$) is known. Each node in level $l + 1$ has $\sqrt{M}$ possible children, so there are $K\sqrt{M}$ possible children in level $l$. One of the main elements of our proposed scheme is to find the children of each node on-demand and in the order of increasing PED rather than calculating the PED of all the children exhaustively. In other words, the key idea of the proposed distributed K-Best scheme is to find the First Child$^6$ (FC) of each parent node in $K_{l+1}$. Among these first children the one with the lowest PED is definitely one of the K-Best candidates in $K_l$. That child is selected and is replaced by its next best sibling$^7$. This process is repeated $K$ times to find the K-Best candidates in level $l$ ($K_l$). The same procedure is performed for each level of the tree, resulting in the algorithm in Table 3.2 for the detection problem throughout the whole tree.

$^6$The first child refers to the child with the lowest local PED among all children of a parent. Finding the first child does not require any sorting and can be realized using a simple rounding operation, explained in Section 3.3.2.

$^7$The next sibling refers to the child with the next lowest local PED.
3.3.2 First/Next Child Calculation

In the distributed scheme in Table 3.2, the first and next child are required to be determined. Based on the system model in (3.4), the first child \(s_l^{[1]}\) of a node in \(\mathcal{K}_{l+1}\) is the one minimizing \(e_l(s^{(l)})\), i.e.,

\[
s_l^{[1]} = \arg \min_{s_l \in \Omega} |e_l(s^{(l)})|^2 = \arg \min_{s_l \in \Omega} |L_l(s^{(l)}) - r_ll|^2.
\] (3.7)

This is because \(T_{l+1}(s^{(l+1)})\) is in common between all children of a parent.

Therefore, \(s_l^{[1]}\) can be found by rounding \(s_l^{[0]} = L_l(s^{(l)})/r_ll\) to the nearest odd number in \(\Omega\) (represented by \(\lceil \cdot \rceil\) in this dissertation). In order to find the next children (NC), the Schnorr-Euchner technique, [53], is employed, which implies a zig-zag movement around \(s_l^{[0]}\) to select the consecutive elements in \(\Omega\). Fig. 3.3 shows such an enumeration for \(\sqrt{M} = 4\). As shown, the SE enumeration finds the closest points in a real domain one-by-one by changing the search direction. In this example -3 is the closest point to \(s_l^{[0]}\) (denoted by arrow 1), the next point is -1 (denoted by arrow 2) and so on. Note that the direction is negative and then positive, and normally alternates between the two. However, if the SE enumeration reaches the upper/lower bound of \(\Omega\) (+3/-3 in this example), the direction of the search remains fixed, as is the case for 3-rd and 4-th children in Fig. 3.3. Based on this strategy, the procedure of selecting the first/next child of node \(j\) in level \(l\) is described in Table 3.3, where \(n_j^l\) denotes the number of moves, and \(\text{SignBit}\) represents the direction. In fact, \(\text{SignBit}\)

---

8 Rounding \(s_l^{[0]}\) to the nearest odd number in \(\Omega\) is equivalent to calculating \(2\lceil s_l^{[0]}/2 + 0.5 \rceil - 1\), where \(\lceil \cdot \rceil\) represents the truncation operation. Therefore, no sorting operation is required.
Table 3.3: First/Next Child Selection Procedure for Node $j$.

A) First Child

A.1) $s_{[0]}^j = \ell_l$
A.2) $s_{[1]}^j \leftarrow \lceil s_{[0]}^j \rceil$, $n_j^l \leftarrow 2$.

B) Next ($k$-th) Child

B.1) \[
\text{SignBit} = \begin{cases} 
\text{Sign}(s_{[k-1]}^l - s_{[k-2]}^l) & \text{if } s_{[k-1]}^l \neq \pm(\sqrt{M} - 1) \\
-1 & \text{if } s_{[k-1]}^l = (\sqrt{M} - 1) \\
+1 & \text{if } s_{[k-1]}^l = -(\sqrt{M} - 1)
\end{cases}
\]

B.2) $s_{[k]}^j \leftarrow s_{[k-1]}^l + n_j^l \times \text{SignBit}$.

B.3) $n_j^l = \begin{cases} 
2 & \text{if } s_{[k-1]}^l = \pm(\sqrt{M} - 1) \\
2 + 2 & \text{otherwise}
\end{cases}$

alternates between positive and negative unless it reaches $\pm(\sqrt{M} - 1)$. The number of moves also increases by 2 every time and is reset to 2 if boundaries of $\Omega$ are reached. The hardware implementation of this pseudo-code will be discussed in Sections 4.2.6, and 4.2.8.

The proposed scheme in Table 3.2 is pictorially depicted in Fig. 3.4 for level $l$ where $\sqrt{M} = 4$ and $K = 3$. It shows the way that $\mathcal{K}_l$ is derived from $\mathcal{K}_{l+1}$. The input to the algorithm is the $K$ best selected nodes of level $l + 1$, which are the current parents with corresponding PEDs of 0.1, 0.4, and 0.6. Each parent can be further expanded to four offsprings resulting in 12 children whose PEDs are shown in Fig. 3.4. Using the technique mentioned in Table 3.3, each parent can find its own first child without visiting all its children. Let $\mathcal{C}_l$ represent the set consisting of all the current best child of all parents, and $\mathcal{D}_l$ represent their corresponding PEDs (in Fig. 3.4, $\mathcal{C}_l = \{c_{12}^l = -1, c_{21}^l = -3, c_{34}^l = 3\}$ with $\mathcal{D}_l = \{d_{12}^l = 0.9, d_{21}^l = 0.5, d_{34}^l = 0.2\}$, where $c_{ij}^l$ represents the $j$-th child of the $i$-th parent in level $l$). It is easy to see that the child with the lowest PED in $\mathcal{C}_l$ is definitely one of the K-Best candidates in $\mathcal{K}_l$, e.g., $c_{34}^l$ in Fig. 3.4 (1st cycle). This is because the child with the lowest PED in $\mathcal{C}_l$

---

9Visiting a child means calculating its PED to the received symbol vector.
has globally the lowest PED. This child can be mathematically represented as $c_{\tilde{k}\tilde{m}}$, where

$$\tilde{k}\tilde{m} = \arg \min_{k,m}(d_{km}^{l} \in D_{l}).$$

(3.8)

Thus this child should be added to $\mathcal{K}_{l}$, i.e., $\mathcal{K}_{l} \leftarrow \mathcal{K}_{l} + \{c_{\tilde{k}\tilde{m}}\}$. To find the next best child in the K-Best list, $c_{\tilde{k}\tilde{m}}^{l}$ and its corresponding PED ($d_{\tilde{k}\tilde{m}}^{l}$) are removed from $\mathcal{C}_{l}$, and $\mathcal{D}_{l}$, respectively (2nd cycle in Fig. 3.4). Following this removal, the next best sibling of this child is added to $\mathcal{C}_{l}$ ($c_{31} = -3$ with $d_{31} = 1.2$ in Fig. 3.4). Taking the same approach, the child in $\mathcal{C}_{l}$ with the lowest PED is definitely among the final K-Best list ($c_{21} = -3$ with $d_{21} = 0.5$). This child should be added to $\mathcal{K}_{l}$, be removed from $\mathcal{C}_{l}$ and be replaced by its next best sibling (3rd cycle). This procedure repeats $K = 3$ times to find all the K-Best candidates (see Fig. 3.4). The final children in the K-Best list are $c_{341}$, $c_{21}$, and $c_{23}$ with PEDs 0.2, 0.5, and 0.7, respectively. Note that using the proposed scheme, only 5 children of 12 possible children are visited in Fig. 3.4. This amount of saving becomes increasingly significant for large $K$ and/or $M$ values.
Table 3.4: The Proposed Implementation for the K-Best Algorithm.

I. INITIALIZATION
1) Find the K-Best children of level $2N_T$ of tree $C_{2N_T}$.
2) Set $K_{2N_T} = C_{2N_T}$.

II. EXPANSION & SORT
For $l = 2N_T - 1 : -1 : 1$
1) $K_l = \emptyset$.
2) Find $C_l$, the set of the first child of each node in $K_{l+1}$.
3) Calculate $D_l$, the weights of the elements in $C_l$.
   For $k = 1 : K$
   4.1) $\vec{k}\vec{m} = \arg\min_{k,m} \{d_{k,m}^l \in D_l\}$
   4.2) $K_l \leftarrow K_l + \{c_{\vec{k}\vec{m}}^l\}$
   4.3) $C_l \leftarrow C_l - \{c_{\vec{k}\vec{m}}^l\}$, and $D_l \leftarrow D_l - \{d_{\vec{k}\vec{m}}^l\}$.
   4.4) Find the next best child of $\vec{k}$-th parent, $c_{\vec{k}\vec{m}}^l$.
   4.5) $C_l \leftarrow C_l + \{c_{\vec{k}\vec{m}}^l\}$, and $D_l \leftarrow D_l + \{d_{\vec{k}\vec{m}}^l\}$.
End
End

III. DETECTION
1) $\vec{k}\vec{m} = \arg\min_{k,m} \{d_{k,m}^1 \in D_1\}$
2) Announce $c_{\vec{k}\vec{m}}^1$ with all of its parents up to the $2N_T$-th level of tree as the hard decision output $\hat{s}$. 
In order to describe the proposed idea using mathematical representations, the detailed pseudo-code of the proposed scheme is shown in Table 3.4. In the initialization step, the $K$ best nodes of the $2N_T$-th level are selected creating $\mathcal{K}_{2N_T}$ (Step I.1 and I.2). For each of the elements in $\mathcal{K}_{2N_T}$, the first child and its corresponding weight are found (Step II.2 and II.3). The child with the lowest weight is selected as one of the $K$ best nodes (Step II.4.1 and II.4.2) and is replaced by its next best sibling (Step II.4.3-II.4.5). This process is repeated $K$ times to find all the K-Best nodes of level $l$ and is performed for all the levels down to level one. Among the K-Best nodes of level one, the node that has the lowest PED $(\tilde{k}\tilde{m} = \arg \min_{k,m} \{d^1_{k,m} \in D_1\})$ with all of its ancestors up to the $2N_T$-th level are announced to be the output of the hard-decision detection problem (Step III.1 and III.2).

This scheme takes exactly $K$ clock cycles to find the $K$ best nodes in each level independent of the constellation scheme. In other words, the complexity of this scheme linearly increases with the value of $K$. Since the value of $K$ scales sub-linearly with the constellation scaling (as will be shown in Section 3.5 and 4.5), the complexity of the proposed scheme scales sub-linearly with respect to the constellation size. It is also efficient with regard to the number of visited nodes in each level using the proposed on-demand scheme. Therefore, it is a promising method for the implementation of the K-Best algorithm in high-order constellation schemes such as 64-QAM and 256-QAM.

The proposed scheme has the following promising features:

1. Complexity independent of the constellation.
2. Considering the $K$ value, it scales sub-linearly with the constellation size.
4. Expands a very small fraction of all the possible children in the exhaustive K-Best approach, $2K-1$ children out of $KM$ possible children, while not sacrificing the performance.
5. Provides the exact K-Best performance without any approximation.
6. Has a fixed-length critical path independent of the constellation size.
7. Can be applied to infinite lattices\textsuperscript{10}.

\textsuperscript{10}Because children are expanded one-by-one and based on an on-demand scheme. Thus the number of children per parent is not required and can be infinite.
8. Can be jointly applied with the lattice reduction (Chapter 5, [59]).

9. Can be easily extended to the complex domain (Section 3.3.3, and [56]).

### 3.3.3 Complex Mode

The extension of the proposed distributed K-Best algorithm, proposed in Section 3.3.1, to the complex domain is required. This is due to the fact that depending on the specification of the target ASIC, the complex-domain implementation might be of more interest than the real-domain implementation. As opposed to some of the K-Best schemes in the literature that cannot be extended to the complex domain, such as [2], the proposed distributed K-Best algorithm in this thesis can be easily shaped for complex constellations without using RVD by extending the SE enumerating technique to the complex domain.

The main challenge in developing the complex domain enumerator lies in devising a means of iteratively enumerating the elements of the complex constellation in order of increasing squared distance from the (complex) unconstrained value, i.e., the PED value. Unlike the real numbers, the complex number field lacks a natural notion of ordering. For instance, complex-valued inequalities are not well-defined. However, because $\mathcal{O}$ is a discrete alphabet, we can obtain an efficient iterative enumeration.

**First Child:** Based on the complex version of the system model in (3.4), the first child of a node in $\mathcal{K}_{l+1}(\tilde{s}_i^{[1]})$ is the one that minimizes $|e_l(\tilde{s}_i^{(l)})|$. In other words\textsuperscript{11},

$$
\tilde{s}_i^{[1]} = \arg \min_{\tilde{s}_i \in \Omega} \left| e_l(\tilde{s}_i^{(l)}) \right|^2 = \arg \min_{\tilde{s}_i \in \Omega} \left| L_i(\tilde{s}_i^{(l)}) - r_{ll}\tilde{s}_i \right|^2 \\
= \arg \min_{\Re(\tilde{s}_i) \in \Omega} \left| \frac{\Re[L_i(\tilde{s}_i^{(l)})]}{u_l^R} - \Re(\tilde{s}_i) \right|^2 + \arg \min_{\Im(\tilde{s}_i) \in \Omega} \left| \frac{\Im[L_i(\tilde{s}_i^{(l)})]}{u_l^I} - \Im(\tilde{s}_i) \right|^2 \\
= a_{R[1]}^{l} + ja_{I[1]}^{l}, \hspace{1cm} (3.9)
$$

where $\Omega = \{-\sqrt{M} + 1, \cdots, -1, +1, \cdots, +\sqrt{M} - 1\}$ represents all the possible values of the real/imaginary part of the constellation points, $a_{R[1]}^{l} = \Re[\tilde{s}_i^{[1]})]$, $a_{I[1]}^{l} = \Im[\tilde{s}_i^{[1]})]$,\textsuperscript{11}Because $T_{l+1}(\tilde{s}_i^{(l+1)})$ is in common for all the children of a parent node.
and the index $l$ was removed as we focus on one parent node in level $l+1$ and try to enumerate its children in level $l$. Note that (3.9)-(3.11) are derived based on the fact that $r_{ll}$ is a real number, which is a result of the QR-decomposition.

Let’s define $\tilde{s}_l^{[0]} = \frac{L_i(\tilde{s}_l^{[0]})}{r_{ll}}$ as the unconstrained received value. Considering the square symmetric M-QAM constellation schemes, there are $|\Omega| = \sqrt{M}$ possible integers on both real and imaginary axis. Thus the optimization in (3.9)-(3.11) are computationally inexpensive to implement as $u_l^R = R[\tilde{s}_l^{[0]}]$ and $u_l^I = I[\tilde{s}_l^{[0]}]$ can be easily rounded to the nearest integers in $\Omega$ to find the optimized value for $R(\tilde{s}_l^{[1]})$ and $I(\tilde{s}_l^{[1]})$, respectively. This optimized value is denoted by $a_{[1]}^R + ja_{[1]}^I$ in (3.11), which is the first child of the current parent. Therefore, the first child can be easily implemented through a two-dimensional slicer.

**Next Child:** To describe how the next child in the complex domain is calculated let us denote all the points in the complex constellation $\Omega$ by a three-level tree shown in Fig. 3.5, where $\tilde{s}_l^{[0]}$ is at the root (Level 1).

Once the first child (i.e., $\tilde{s}_l^{[1]} = a_{[1]}^R + ja_{[1]}^I$) was determined, it is selected as the first node in Level 2 (the leftmost node in Level 2 in Fig 3.5) and the $\sqrt{M}$ Level-2 siblings are chosen as those that share the same complex value $a_{[1]}^I$ as the first child$^{12}$. Therefore, their squared distances from $\tilde{s}_l^{[0]}$ vary directly with those of their real components from the real part of $\tilde{s}_l^{[0]}$. Since they are all in one line with different real parts, the real SE enumeration technique discussed in Section 3.3.1 can be applied to

---

$^{12}$This is because of the fact that there are $\sqrt{M}$ possible real values and $\sqrt{M}$ possible imaginary values in the constellation.
enumerate them in the order of non-decreasing squared distance from $\tilde{\mathbf{s}}_l^{[0]}$ (SE row-enumeration). This means that the nodes in the second level of the tree are positioned in the nondecreasing PED order.

For the third level of the tree in Fig. 3.5, $\sqrt{M} - 1$ siblings are assigned to each Level-2 parent node such that they share the same real value inherited from their common parent node whereas their imaginary parts take the values $a_{[2]}^l, \ldots, a_{[\sqrt{M}]}^l$. For instance, the elements of the leftmost subtree in Fig. 3.5 are $\left\{a_{[1]}^l + ja_{[2]}^l, \ldots, a_{[\sqrt{M}]}^l + ja_{[\sqrt{M}]}^l\right\}$ and those of the rightmost subtree are $\left\{a_{[\sqrt{M}]}^l + ja_{[2]}^l, \ldots, a_{[\sqrt{M}]}^l + ja_{[\sqrt{M}]}^l\right\}$. Similarly to the Level-2 nodes, the real SE enumeration can be applied to their imaginary components to enumerate them in the order of nondecreasing squared distance from $\tilde{\mathbf{s}}_l^{[0]}$ (SE column-enumeration). This implies that using the SE enumeration, all the children of a node in Level 3 are positioned in the tree from left to right in the order of nondecreasing PED.

Based on the above three-level tree structure, the next child is calculated as follows. Recall that the first child corresponds to the $\tilde{\mathbf{s}}_l \in \mathcal{O}$ that minimizes $|e_l(\tilde{\mathbf{s}}^{(l)})|$. By definition, its next best sibling $(\tilde{\mathbf{s}}_l^{[2]}$) is the one that has the next smallest incremental distance $|e_l(\tilde{\mathbf{s}}^{(l)})|$, i.e., it is the one in $\tilde{\mathbf{s}}_l \in \{\mathcal{O} - \tilde{\mathbf{s}}_l^{[1]}\}$ that minimizes $|e_l(\tilde{\mathbf{s}}^{(l)})|$. Let $\mathcal{L}$ denote the set of all visited points in the constellation, which have not yet been announced as the next best sibling. As a new point is enumerated, it is added to $\mathcal{L}$ so initially $\mathcal{L} = \{\tilde{\mathbf{s}}_l^{[1]}\}$. At each step, the point in $\mathcal{L}$ with the lowest PED is announced to be the next best sibling. That point is removed from $\mathcal{L}$ and replaced by one or two new points that are generated by the real SE row-enumeration and/or SE column-enumeration performed on it. In fact, the row and column enumerations enable the coverage of the possible sets of values for $\Re[\tilde{\mathbf{s}}_l]$ and $\Im[\tilde{\mathbf{s}}_l]$, respectively. These new point(s) are said to be visited and are then added into $\mathcal{L}$.

Let’s describe the complex SE enumeration with an example shown in Fig. 3.6, where the bold crosses represent the visited points while the bold circled crosses represent the points announced as the next best child. Starting from $\tilde{\mathbf{s}}_l^{[1]}$ (Fig. 3.6(a), which is the left-most node in Level 2, its first child in Level 3 and its next sibling in Level 2 are visited and are added to $\mathcal{L}$. Among these two new nodes in $\mathcal{L}$, the one with the lowest PED is chosen ($+1 + j$ in Fig. 3.6(b)). If the chosen node is a
Figure 3.6: The first four best children using complex SE enumeration in a 16-QAM Constellation scheme: (a) $\mathcal{L} = \{-1 + j\}$, (b) $\mathcal{L} = \{-1 - j, +1 + j\}$, (c) $\mathcal{L} = \{-1 - j, 1 - j, -3 + j\}$ and (d) $\mathcal{L} = \{-1 + 3j, 1 - j, -3 + j\}$.

Level-2 node, its next sibling in Level 2 and its next child are both enumerated and are added to $\mathcal{L}$, which is equivalent to running both the row and column enumeration simultaneously, see Fig. 3.6(c). However, if the chosen node in $\mathcal{L}$ is a Level-3 node, since it has no children, only its next sibling is counted and is added to $\mathcal{L}$, as it is the case in Fig. 3.6(d). In other words, after finding $\tilde{s}_i^{[2]}$ in 3.6(b), since it is a Level-2 node, both row and column enumerations are performed resulting in the addition of $+1 - j$ and $-3 + j$ to $\mathcal{L}$. On the other hand, since in Fig. 3.6(c) the node $\tilde{s}_i^{[3]}$ is a Level-3 node, only the column enumeration is performed resulting in the addition of $-1 + 3j$ to $\mathcal{L}$. This process is performed until all the points in the constellation are covered. Repeated application of this procedure ensures on-demand enumeration of the complex constellation points in the order of increasing local PED. Note that
since the expansion scheme proposed here is on-demand, not all of the nodes in the constellation are necessarily visited.

The sequence in Fig. 3.6 shows the process of finding the first 4-best children of a particular parent node using the proposed scheme with the elements of $\mathcal{L}$ listed in each stage in the caption. At each stage, a dashed gray circle indicates the distance of the most recent announced K-Best node to $\tilde{s}_i^{[0]}$. One argument should be proven to guarantee the correct functionality of this complex enumeration scheme, i.e., it should be ensured that when a node in $\mathcal{L}$ is announced as the next K-Best node, any other node in $\mathcal{O}$ with a lower PED has already been visited and announced as the K-Best candidate. In other words, all the unannounced nodes in $\mathcal{O}$ should have a larger PED than the most recent announced K-Best node.

**Proposition 3.1** Using the proposed SE complex enumeration scheme, nodes are visited in the order of the increasing PED value.

**Proof 3.1** The following two observations are used for this proof.

**Notes:**

(i) Any Level-3 node has a larger PED than that of its corresponding Level-2 parent node.

(ii) If a node is announced as the next K-Best node, according to (3.8), it has the lowest PED among the nodes in $\mathcal{L}$.

Let $\mathcal{U} = \mathcal{O} - \mathcal{K} - \mathcal{L}$, where $\mathcal{O}$ is the set of all nodes in the constellation, $\mathcal{K}$ is the set of nodes, which have been announced as the K-Best nodes, and $\mathcal{L}$ is the set of nodes visited but has not announced as the K-Best candidate yet. It should be proved that all the unannounced nodes in $\mathcal{O}$ have a larger PED than the nodes in $\mathcal{K}$. In a mathematical form:

$$\forall \tilde{s}_i^* \in \{\mathcal{O} - \mathcal{K}\}, \forall \tilde{s}_i^{[q]} \in \mathcal{K}: PED(\tilde{s}_i^*) > PED(\tilde{s}_i^{[q]}),$$

(3.12)

where $PED(\tilde{s}_i^{[q]})$ denotes the PED value of the node and its all ancestors to the $2N_T$-th level of the detection tree. Since K-Best nodes are announced in the nondecreasing order of PED, it is only needed to prove that:

$$\forall \tilde{s}_i^* \in \{\mathcal{O} - \mathcal{K}\}: PED(\tilde{s}_i^*) > PED(\tilde{s}_i^{[k]}),$$

(3.13)

where $\tilde{s}_i^{[k]}$ represents the most recent announced K-Best node in $\mathcal{K}$.

To prove, let’s consider the contrary, i.e., assume there is a node, represented by $\tilde{s}_i^*$, which is not in $\mathcal{K}$ and has a lower PED than $PED(\tilde{s}_i^{[k]})$. There are six possible cases to look at shown in Fig. 3.7 and described in the following:
Figure 3.7: Six possible cases for proof of the functionality of the complex SE enumeration.

(1) This case implies that a Level-2 node \( \tilde{s}_i^* \) has a lower PED than a Level-2 \( \tilde{s}_i^{[k]} \), which is contrary to the concept of the ordered SE-row enumeration.

(2) This means that a Level-2 node \( \tilde{s}_i^* \in L \) has a lower PED than the Level-3 \( \tilde{s}_i^{[k]} \), which is contrary to Note (ii) above.

(3) This case means that a Level-2 node \( \tilde{s}_i^* \in U \) has a lower PED than the Level-3 \( \tilde{s}_i^{[k]} \). This implies that there is one unannounced Level-2 node in \( L \) whose PED is larger than PED(\( \tilde{s}_i^{[k]} \)) resulting in the conclusion that PED(\( \tilde{s}_i^* \)) < PED(\( \tilde{s}_i^{[k]} \)), which is contrary to the ordered SE-row enumeration.

(4) This case implies that a Level-3 node \( \tilde{s}_i^* \in L \) has a lower PED than the node \( \tilde{s}_i^{[k]} \), which is contrary to the Note (ii) above.

(5) This means that a Level-3 node \( \tilde{s}_i^* \in U \) whose Level-2 parent node is in \( L \) has a lower PED than the node \( \tilde{s}_i^{[k]} \), which is contrary to the Note (i) and Note (ii) above.

(6) This case implies that a Level-3 node \( \tilde{s}_i^* \in U \) whose Level-2 parent node is in \( U \) has a lower PED than the node \( \tilde{s}_i^{[k]} \). This means that regardless of the Level of \( \tilde{s}_i^{[k]} \), there is one unannounced Level-2 node in \( L \) (not including \( \tilde{s}_i^{[k]} \)), which has a lower PED than the node \( \tilde{s}_i^{[k]} \). This is contrary to the ordered SE-row enumeration and Note (i) above. ■
Considering the nature of the SE enumeration on both dimensions along with the running mechanism on them makes the proof intuitively. In other words, the nodes in the 3-Level tree shown in Fig. 3.5 are sorted while being enumerated in the tree on both Level 2 and Level 3 from left to right. Therefore, the selection of the nodes starts from left to right on both levels. In fact, moving from left to right on the 3-level tree corresponds to visiting the nodes with nondecreasing PED value. Therefore, if there is one unannounced Level-3 node among children of a Level-2 parent node, it is guaranteed that all of its Level-3 siblings on its right side have a higher PED. Also if there is one unannounced Level-2 node, all the other Level-2 nodes on its right side and their children have a higher PED than this node.

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

### 3.4 Complexity Analysis

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

Table 3.5 shows the complexity comparison between different schemes. For comparison metrics, the number of visited children (called \textit{expand} in Table 3.5), the required number of clock cycles to do the sorting (called \textit{sort}), and the total latency are considered. The latency shows the number of clock cycles it takes to detect a symbol.
Table 3.5: Comparison of Different K-Best Implementations.

<table>
<thead>
<tr>
<th>Operation</th>
<th>[3] (real)</th>
<th>[2] (real)</th>
<th>[54] (real)</th>
<th>This work (real)</th>
<th>This work (complex)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>4 × 4 16-QAM</strong> (5,6)</td>
<td>expand⁴ 160</td>
<td>160</td>
<td>160</td>
<td>72</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>sort⁵ 160</td>
<td>40</td>
<td>280</td>
<td>37</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>latency⁶ 240</td>
<td>49</td>
<td>440</td>
<td>40</td>
<td>35</td>
</tr>
<tr>
<td><strong>4 × 4 64-QAM</strong> (10,11)</td>
<td>expand 640</td>
<td>640</td>
<td>640</td>
<td>152</td>
<td>84</td>
</tr>
<tr>
<td></td>
<td>sort 640</td>
<td>80</td>
<td>900</td>
<td>74</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>latency 720</td>
<td>99</td>
<td>1540</td>
<td>80</td>
<td>65</td>
</tr>
<tr>
<td><strong>4 × 4 256-QAM</strong> (15,17)</td>
<td>expand 1920</td>
<td>1920</td>
<td>1920</td>
<td>232</td>
<td>140</td>
</tr>
<tr>
<td></td>
<td>sort 1920</td>
<td>120</td>
<td>2700</td>
<td>113</td>
<td>72</td>
</tr>
<tr>
<td></td>
<td>latency 2000</td>
<td>149</td>
<td>4620</td>
<td>120</td>
<td>107</td>
</tr>
</tbody>
</table>

¹ Number of visited nodes.
² Number of clock cycles required for sorting.
³ Initial latency (in clock cycles) before the first estimation comes out.

vector, \( s \). The sorting for [3], and [2] are calculated with the assumption that it can be implemented with a linear complexity using bubble sort [44]. The value listed in the expand row refers to the number of visited children or equivalently the number of Euclidean distances required to be calculated. This directly translates to more area and power of the silicon implementation. However, if all the distance calculations are done in parallel there is no significant throughput loss. The BER performance result of our proposed approach is similar to [3], [2], and [54]. Thus the major difference between all these schemes, including this work, is the way the K-Best method is implemented, which translates to different throughputs and/or hardware complexity.

As can be seen, using our proposed scheme, both in real and complex modes, significant complexity savings are achieved. Table 3.5 shows the total number of distance calculations, sorting period as well as the total latency of the search throughout the tree in a 4 × 4 MIMO configuration for different constellation schemes. The numbers in parentheses in the first column show the value of \( K \) for real and complex modes in each case, respectively. For instance, for 16-QAM, (5,6) means that \( K = 5 \) and \( K = 6 \) for real-domain and complex-domain implementations, respectively. Note that both our real and complex domain versions outperform [2] in terms of the latency, total distance calculation as well as the sorting time. Although, the complexity of the complex-domain implementation might seem to be lower than that of the real-domain implementation, since the complex-domain implementation requires complex
operations and has lower granularity compared to the real-domain implementation, it is not suitable for hardware implementations. In fact, in order to implement the on-demand version of the complex domain, as was shown in Section 3.3.3, to find the next child in each level, the PED value of $\mathcal{L}$ nodes should be calculated and sorted. This is an extra computation that is avoided by using the real-domain implementation. Moreover, since it adds additional cycles for this extra sorting, it might affect the overall throughput. The only benefit of using the complex-domain implementation is a smaller latency at the beginning of the detection process, which is not important as it does not affect the achieved throughput.

Another promising aspect of our approach is that the number of visited nodes is NOT a function of the constellation order. The fact that makes this feature possible is the ordered expansion (using the on-demand strategy) along with the pipelined sorting scheme described in Chapter 4. It is worth noting that the complex version requires extra circuitry to implement the above proposed expansion scheme in the complex domain. However, since it is implemented in an on-demand basis, and the fact that $\mathcal{L}$ does not populate proportionally, this extra circuitry is not considerable. In fact, based on the following proposition, it can be shown that $|\mathcal{L}| \leq \sqrt{M}$.

**Proposition 3.2** Using the above complex SE enumeration scheme, the value of the $\mathcal{L}$ is always less than $\sqrt{M}$, where $M$ represents the size of the constellation.

**Proof 3.2** : Based on the above proposed scheme, there are two levels of nodes in the tree (Level-2 nodes and Level-3 nodes). If any of the Level-2 nodes is selected, it is excluded from $\mathcal{L}$ and at most two more constellation points are added to $\mathcal{L}$ (its next sibling in Level-2 and its next child in Level-3). Therefore, the selection of any Level-2 node would increase the value of $|\mathcal{L}|$ at most by one. However, if the selected node is in Level-3, it is excluded from $\mathcal{L}$ and at most its sibling if any is added to $\mathcal{L}$. Therefore, if a Level-3 node is selected, the value of $|\mathcal{L}|$ does not change or may even decrease. Having said that, since there are $\sqrt{M}$ Level-2 nodes in any $M$-QAM constellation, the value of $|\mathcal{L}|$ can be increased by $\sqrt{M}$ units resulting in $|\mathcal{L}| \leq \sqrt{M}$. 

This fact has been shown in Fig. 3.8, where it shows the complex enumeration in a 16-QAM constellation for a specific received symbol, where the received symbol is depicted by “x” in the figure. Note that dots (“.”) represent the constellation points, circled dots (“○”) represent the visited candidates not selected yet (or the elements of $\mathcal{L}$), and finally the black circles (“•”) denote the announced constellation so far (the elements of $\mathcal{K}$).
Figure 3.8: The variation of the value of $|\mathcal{L}|$ for 16-QAM for a specific received symbol:
(a) $|\mathcal{L}| = 3$, (b) $|\mathcal{L}| = 4$, (c) $|\mathcal{L}| = 4$, (d) $|\mathcal{L}| = 4$, (e) $|\mathcal{L}| = 4$, (f) $|\mathcal{L}| = 1$. 
The arrows show the flow of the enumeration in the 16-QAM constellation, which depends on the location of the received symbol and the number on each arrow represents the time step at which the target node is visited. For instance in Fig. 3.8(a), \(+1 - j\) and \(-1 - 3j\) are visited in the second step of the enumeration, while \(+1 + j\) is visited in the fifth step of the enumeration (Fig. 3.8(c)). The value of \(|\mathcal{L}|\) is the number of visited points not selected yet, i.e., circled dots (“⊙”). By looking at Fig. 3.8 the largest value of \(|\mathcal{L}|\) is 4. The value of \(|\mathcal{L}|\) for each case is mentioned in the caption of the figure.

Fig. 3.8(a) shows the enumeration status after the third step, which results in 3 elements in \(\mathcal{L}\). Note that in the fourth step (Fig. 3.8(b)) since the selected node is \(+3 - 3j\), which is a Level-2 node, it will enumerate two more nodes increasing the value of \(|\mathcal{L}|\) by one. However, in case the chosen node is a Level-3 node, such as \(+1 - j\) in Fig. 3.8(c), it will enumerate one more node in the constellation and does not change the value of \(|\mathcal{L}|\). This process is repeated all the way to the end until the enumeration reaches the boundaries of the constellation as shown in Fig. 3.8(f), where it shows the final step of the enumeration results in no more nodes and the value of \(|\mathcal{L}|\) is decreased until no more node are available to select.

### 3.5 Simulations

In theory, the K-Best algorithm might miss the hard-ML point and might have performance loss as a result. However, for a proper choice of \(K\), the BER performance of the K-Best method approaches the optimal case for a reasonable range of SNR values. In the following, the simulation results for a single-carrier \(4 \times 4\) MIMO system is presented for both 16-QAM and 64-QAM schemes. Throughout this thesis the simulation results for the BER curve is performed for 100,000 packets, where each packet consists of \(4 \times 6 \times 4 = 96\) bits (9.6Mbits in total) for a \(4 \times 4\) MIMO system. Test vectors are created using: (i) pseudo-random data, (ii) complex-valued random Gaussian channel matrix \(\tilde{\mathbf{H}}\) with statistically independent elements updated per four channel use, and (iii) additive white Gaussian (circularly symmetric) complex random noise. Fig. 3.9 shows the BER performance curves for a \(4 \times 4\) 64-QAM MIMO system using the K-Best scheme vs. the ML detector. It reveals the behavior of the K-Best algorithm for different values of \(K\). It is seen that by increasing the value of \(K\), the performance result becomes close to ML detection. However, a higher \(K\) value
results in more hardware complexity as will be discussed in Chapter 4. For instance, in the case of real K-Best implementation of a $4 \times 4$ 64-QAM MIMO system (Fig. 3.9) \( K = 15 \) results in a ML-like result while \( K = 8 \) comes with less diversity in high-SNR regimes. The performance of the K-Best scheme with \( K = 10 \) is close to the ML while having a moderate complexity. Thus \( K = 10 \) is chosen as the framework for the hardware implementation in this dissertation.

Fig. 3.10 shows the BER performance of a 16-QAM scheme for different values of \( K \) for real and complex equivalent models as well as the optimal ML result. The SNR value is defined as the signal-to-noise ratio per transmitted symbol. As shown, for the normal SNR range of interest (0-23 dB) the performance loss is small. Based on this simulation, the real-domain implementation outperforms the complex domain for the same value of \( K \) (\( K = 5 \) in this Fig. 3.10). However, by increasing the value of \( K \) in the complex domain to 6, the performance result of the complex and real models are almost the same. Increasing the \( K \) value from 5 to 6 increases the complexity by 20% for the calculation of the extra children. However, since the number of levels in the complex domain is 4 vs 8 in the real case, the total complexity saving in expansion is about 40% in this case.

Fig. 3.11 shows the BER performance of the 64-QAM scheme for both the real and complex implementations. For the normal SNR range of interest (0-30 dB) the
Figure 3.10: K-Best vs. ML BER for different values of $K$ in both real and complex domain for $4 \times 4$ 16-QAM MIMO detection.

Figure 3.11: K-Best vs. ML BER for different values of $K$ in both real and complex domain for $4 \times 4$ 64-QAM MIMO detection.
performance loss is small. Moreover, the performance of the real model with $K = 10$ is the same as that of the complex model for $K = 11$. This results in a saving of 45% in the sorting core compared to the best real model implementation in the literature (see Table 3.5). The performance curves for $K = 15$, and $K = 7$ are also shown. The loss associated with $K = 7$ is considerable whereas in the case of $K = 15$, there is only 0.4 dB improvement at 30dB.

3.6 Summary

A novel on-demand scheme for the K-Best algorithm suitable for high-order constellations was proposed. It has the lowest latency reported to-date as well as the smallest number of visited nodes. It is applicable to the unbounded infinite lattice decoding problem and has the fixed-critical path independent of the constellation scheme or the lattice size. The key contribution was the introduction of a novel joint on-demand expansion and distributed sorting scheme operating in a pipelined fashion. The simulation results for both floating and fixed-point simulations show a near-ML performance result even in the presence of quantization/truncation noise.
4 VLSI Implementation of a Scalable K-Best Detector

The K-Best algorithm is a well-known approach and an alternative to sphere decoding for MIMO detection offering the important advantage of SNR-independent throughput. Spatial Multiplexing (SM) MIMO detectors based on the K-Best algorithm for 16-QAM systems with 4-transmit and 4-receive antennas ($4 \times 4$) have been realized for low-order constellations [3]. However, a scalable high-throughput silicon implementation for a $4 \times 4$ 64-QAM system, envisioned in LTE and WiMAX systems, remains a significant technical challenge especially for high multimedia data rates that are envisaged to approach 1Gb/s in future systems. State-of-the-art receivers in the literature consume large silicon area, [47], achieve only medium-rate (115Mb/s) data throughput rates, [47], [51], that falls short of the most aggressive next-generation 4G applications, have a critical path that scales quadratically with the constellation size, exhibit variable data throughput that is a function of signal-to-noise-ratio (SNR) and, all implementations to-date reveal some performance loss, [51], relative to the perfect K-Best.

There has been some effort in the literature on the VLSI implementation of the K-Best algorithm [2], [3], [4], [44], [47], [60]. However, the child expansion and sorting schemes utilized are not efficient/scalable for higher-order constellation schemes such as 64-QAM and 256-QAM. In most of the architectures, the critical path increases with the scaling of the modulation order, which ultimately limits the maximum throughput [2]. Moreover, in spite of having various architectures for the implementation of $4 \times 4$ 16-QAM systems, an efficient high-throughput ASIC implementation for 64-QAM systems is still a significant challenge and has not been addressed properly in the literature.
The proposed approaches in the literature have the following deficiencies:

• **Performance Loss**: Most of the low-complexity approaches have performance loss compared to the strict K-best algorithm.

• **Low Throughput**: A high-throughput implementation for a $4 \times 4$, 64-QAM system has yet to be developed.

• **Large Silicon Area**: Realizations with no performance loss are both area-hungry and low-throughput.

• **Nonlinear Complexity**: The complexity of the current schemes are not linear with respect to the constellation size.

• **Constellation-dependent Critical Path**: Most of the designs to-date have critical path, which scales up with the constellation size.

• **SNR Dependent**: Most of the design have a variable throughput with respect to the operating SNR.

• **Complicated Integration**: The current K-Best schemes cannot be applied jointly with other performance enhancing blocks such as lattice reduction.

To the best of our knowledge, all the designs in the literature have a few of the above deficiencies. As an example, to illustrate the bottleneck in the current proposed schemes, the architecture proposed in [2], which has the highest-throughput reported for a $4 \times 4$ 16-QAM system is briefly reviewed. The K-Best detector in [2] is pipelined such that one layer of the tree is always processed in one pipeline stage (Fig. 4.1). Each stage consists of an MCU, a K-Best Unit (KBU) that determines the $K$ smallest PEDs, and a register bank $L_K$ where the $K$ smallest nodes of the previous layer are stored. Together, they form a computation unit. Resource sharing is applied such that the $K$ nodes at the input of the stage are processed one after the other. In each cycle, the MCU delivers the PEDs of all children of a parent node in $L_K$. These PEDs need to be sorted into a list $\hat{L}_K$ where the $K$ smallest PEDs found so far are stored. After $K$ iterations, all children of the nodes in $L_K$ have been computed by the MCU. The KBU has determined the $K$ smallest PEDs and delivers them to the next pipeline stage. In total, $2N_T$ identical copies of the computation unit form the $2N_T$ pipeline stages of the detector.
The main challenge in the design of a throughput-optimized K-Best decoder, which considers more than one child in parallel, is in the implementation of the KBU. In each cycle, the corresponding circuit needs to update the contents of its preferred children cache by selecting the $K$ nodes with the smallest PEDs from the $K$ entries of that cache and from the additional list of $\sqrt{m}$ PEDs that it receives from the MCU. Unfortunately, sorting the corresponding compound list with $\sqrt{m} + K$ entries in order to identify the $K$ preferred children in a single cycle is costly in terms of hardware. In order to reduce the complexity, the SE enumeration is used to sort the outputs of the MCU in ascending order of their PEDs. Assuming that the list in the preferred children cache is also ordered, the task of sorting a long, fully unordered list of PEDs now reduces to merging two already sorted lists into a new list, in which we are only interested in the $K$ smallest entries. The block diagram of a corresponding circuit for $K = 5$ and 16-QAM modulation is shown in Fig. 4.2 together with an example of the merging operation. The bold line in the figure indicates the longest path of the circuit whose length is roughly a linear function of $K$ and thus matches the expected complexity scaling behavior of the merging algorithm, which is given by $O(K)$. However, note that for larger $K$, throughput is rapidly decreasing because more cycles in the same pipeline stage are necessary and the merging core and its critical path becomes increasingly complex, which significantly reduces the maximum clock frequency. Moreover, this scheme only works for cases where $K \leq \sqrt{M}$.

In this dissertation, we propose an efficient VLSI architecture for K-Best infinite
Figure 4.2: KBU unit [2] that performs the merging for $K = 5$. 
lattice decoders, which alleviates all of the above problems and operates at a significantly higher throughput than currently reported schemes. The promising feature of the proposed ASIC in this dissertation is its scalability to higher-order constellation sizes such as 256-QAM, which is inevitable for next-generation WiMAX systems. In fact, the fabricated design in this dissertation is the only design to-date that has none of the above deficiencies.

In this chapter, we solely focus on the implementation of the K-Best algorithm. Thus it is assumed that the sorted QR-decomposition algorithm according to [34] is applied to the channel matrix before our implemented ASIC to generate the $R$ matrix using the scaled and decoupled architectures [61], which also applies the generated $Q$ matrix to the received vector to generate $z$. Without loss of generality, it is also assumed that $l$-th row of matrix $R$ and $z_l$ are normalized by $r_{ll}$. Therefore, $L_l(s^{(l)})$ in (3.6) can be written as

$$L_l(s^{(l)}) = r_{ll}(\bar{z}_l - \sum_{j=l+1}^{2N_T} \bar{r}_{lj}s_j), \quad (4.1)$$

where $\bar{z}_l$, and $\bar{r}_{lj}$ denote the scaled $z_l$ and $r_{lj}$ by $r_{ll}$, respectively, ($z_l = \bar{z}_lr_{ll}$, and $r_{lj} = \bar{r}_{lj}r_{ll}$).

### 4.1 General VLSI Architecture

One challenge that needs to be addressed toward a high-throughput architecture is the implementation of $K$ minimizations in each level (Step 4.1 in Table 3.4), which is still computationally complex even with state-of-the-art bubble sorting [44]. For instance, in Fig. 3.4, in order to determine each child in the K-Best list, the lowest value among three PEDs should be calculated. In order to resolve this problem, we propose a pipelined structure, which performs the child expansion (Step 4.4-4.5 in Table 3.4) and minimization (Step 4.1 in Table 3.4) jointly in a pipelined fashion and implements the sorting in a distributed way without sacrificing the throughput.

The proposed architecture with all intermediate parameters for a $4 \times 4$, 64-QAM MIMO system with $K = 10$ and $\Omega = \{-7, -5, -3, -1, +1, +3, +5, +7\}$ is shown in Fig. 4.3. There are $2N_T = 8$ levels in the tree. The 8-th level of the tree, corresponding to the last row of equation (3.1), opens up all the possible values in $\Omega$, and calculates their corresponding PEDs. The output of this stage is $\mathcal{K}_8$ resulting...
Figure 4.3: The proposed pipelined VLSI architecture of the K-Best algorithm for the detection of a $4 \times 4$, 64-QAM system with $K = 10$. 
in $|\Omega| = 8$ PED values, which is done by **Level I** in Fig. 4.3. For each of the nodes in $\mathcal{K}_8$, the first child is found (**Step 2**) and its PED is updated using the FC-Block in **Level II** (more detail in Section 4.2.8). The inputs to the FC-Block are $\bar{z}_7$, and $r_{77}$. Then the FC with the lowest PED should be determined (**Step 4.1**), which requires all the FCs to be sorted. This is done using the **Sorter** block in Fig. 4.3, which sorts all eight resulting PEDs in four cycles. This sorter is one of the contributions of this dissertation, which can be applied to the general sorting problem. Using this sorter, the number of clock cycles required for sorting is half as much as that of the classic bubble sorting. The key idea that makes this sorter faster, is the implementation of two tasks (max/min and the data exchange) in one clock cycle through the introduction of intermediate registers. The detailed architecture of this block will be discussed in Section 4.2.4.

The output of the **Sorter** block is the sorted FCs of level 7, i.e. $\mathcal{C}_7$, (FC-L7 in Fig. 4.3), which are all loaded simultaneously to the next stage (i.e., **PE I**) to form $\mathcal{K}_7$. Generally speaking, in each level, one **PE II** block is used to generate and sort the list of all FCs of the current level and one **PE I** block is used to generate the K-Best list of the current level. This fact is denoted in Fig. 4.3 by FC-L$i$ and NC-L$i$ labels under each level’s block.

The task of the **PE I** block is to take the FCs of each level as an input and generates the K-Best list of that level one-by-one. The node in $\mathcal{C}_7$ with the lowest PED is definitely one of the K-Best candidates in level 7 (**Step 4.1-4.2**). This value is passed to the **PE II** block in FC-L6. Upon the removal of this FC, its next sibling needs to be calculated, which is done by the core called **NC-Block** in the feedback loop of the **PE I** block (Section 4.2.5 and 4.2.6), and substitutes the FC in $\mathcal{C}_7$ (**Step 4.3-4.5**). The PED of this sibling needs to be compared with the other FCs, already present in the NC-L7 stage. The next K-Best candidate has the lowest PED among this new set. This process is repeated 10 times (taking 10 cycles) till all the K-Best values of the second level of the tree are generated and passed to the **PE II** block in FC-L6.

The **PE II** receives the K-Best candidates of level 7 one after the other and generates the FC of each received K-Best candidate one-by-one and sorts them as they arrive. It finally transfers them to its following **PE I** block. Note that once all the FCs of the current level (level 6) have been calculated and sorted, they are passed to the **PE I** block of the next stage in a set of pairs. The reason behind this scheduling
as well as the detailed description of this block will be discussed in Section 4.2.7. Therefore, the output of the PE II block in FC-L6 are the FCs of level 6, which are sorted and passed to the next stage. This process repeats for all the levels down to the first level. Since at the first level only the FC with the lowest PED is of concern, only one PE II block is used for the first level (FC-L1), whose output is the solution to the hard detection symbol \( \hat{s} \).

### 4.2 Detailed VLSI Architecture

#### 4.2.1 Inputs

The inputs to the architecture are the entries of the \( R \) matrix as well as the \( z \) vector in (3.1). Using the RVD scheme, proposed in (2.5), the resulting \( R \) matrix after the QR-decomposition has symmetry features that can be exploited to advantage, which are explained by the following example. Consider a 4\( \times \)4, 64-QAM MIMO system (8\( \times \)8 real domain). The \( R \) matrix is as follows:

\[
R = \begin{bmatrix}
    r_{22} & 0 & r_{24} & -r_{23} & r_{26} & -r_{25} & r_{28} & -r_{27} \\
    0 & r_{22} & r_{23} & r_{24} & r_{25} & r_{26} & r_{27} & r_{28} \\
    0 & 0 & r_{44} & 0 & r_{46} & -r_{45} & r_{48} & -r_{47} \\
    0 & 0 & 0 & r_{44} & r_{45} & r_{46} & r_{47} & r_{48} \\
    0 & 0 & 0 & 0 & r_{66} & 0 & r_{68} & -r_{67} \\
    0 & 0 & 0 & 0 & 0 & r_{66} & r_{67} & r_{68} \\
    0 & 0 & 0 & 0 & 0 & 0 & r_{88} & 0 \\
    0 & 0 & 0 & 0 & 0 & 0 & 0 & -r_{88}
\end{bmatrix}.
\] (4.2)

This implies that two consecutive rows of the \( R \) matrix share the same entries with a possible sign flip. Therefore, in the VLSI architecture, the input values of two consecutive levels share the \( r_{ij} \) values meaning that the above RVD both reduces the number of input pads and the required memory to buffer the \( r_{ij} \) values. The other implication of this structure is that the first children of the odd rows do not depend on the K-Best list of the preceding even row. For instance, the first child of the fifth row is independent of \( s_6 \) and that is due to the fact that \( r_{56} = 0 \).

Note that in the proposed architecture, it is assumed that the channel is quasi-static and is updated every four channel use. This implies that the QR-decomposition and the input \( R \) entries need to be updated with a proper frequency. In total, there
are 16 distinct entries in the $R$ matrix as well as 32 entries corresponding to the input $z$ vector\textsuperscript{1}. Assuming that each entry requires 16 bits on average, the number of input pads for the chip will be 768 pads, which is not feasible for a cost-efficient implementation\textsuperscript{2}. In order to avoid this number of pads in our ASIC, input values are received one-by-one at the input, are buffered and consumed later at the proper time.

In other words, the input $r_{ij}$ values are received at the input pads in the following order:

$$\{r_{88}, r_{66}, r_{67}, r_{68}, r_{44}, r_{45}, r_{46}, \ldots, r_{28}\}.$$ \hfill (4.3)

Moreover, we use a multiplexed scheme where each 16-bit $r_{ij}$ is received in two consecutive clock cycles, i.e., 8 bits per clock cycle. The received entries are buffered at the input and are used at the proper time in the architecture. Using this scheme the number of pads is significantly reduced. The drawback is a longer initial latency\textsuperscript{3}. Fig. 4.4 shows such a scheduling at the input for two consecutive iterations. In the first 32 cycles the values of $r_{ij}$ and $z_j$ are buffered at the input. At the end of the 40 cycles the buffered samples are used for the detection core. As shown in Fig. 4.4, each $r_{ij}$ value is read in two cycles (different cycle colors in the figure) while each $z_j$ requires only one clock cycle. This is due to the fact that there are 16 $r_{ij}$ per channel matrix and 32 $z_j$ values for 4 consecutive channel realization. All numbers are implemented using 2’s complement representation.

There are a few sub-blocks used throughout the architecture, which are explained first, after which the major functional blocks are discussed.

\textsuperscript{1}There are four received $z$ vectors per channel matrix and each $z$ vector has eight entries corresponding to eight levels of the tree, which results in a total of 32 entries.

\textsuperscript{2}Note that this argument is valid for the case where the MIMO detector is designed as a stand-alone chip. However, in case of an Intellectual Property (IP) core, the number of pins in the interface is not an issue.

\textsuperscript{3}Note that all these multiplexing schemes are utilized to decrease the number of pads. This is due to the fact that early synthesis results show that our design is pad limited.
Multiplication (MU)

There are two types of multiplications involved in the overall architecture. The multiplication of $z_i \times r_{ll}$ and $s_k \times r_{lj}$ (see (4.1)). The first one is realized using a 13-bit×13-bit multiplier. However, the second multiplication can be implemented using an alternative architecture, which takes less area. This architecture, called MU, is shown in Fig. 4.5 using a few multiplexers (MUX) and adders (SUM), where the numbers on the right represent the bit location in $s_j$ (i.e., $s_j$ can be represented with 4 bits), ”$\ll n$” represents $n$ shifts to the left and the tiny bubble “$\oplus$” denotes the negation operation$^4$. This implementation of the multiplication operation is possible due to the fact that the values of $s_k$ are drawn from a finite pre-determined odd-integer set $\Omega = \{(-\sqrt{M} + 1), \ldots, -1, +1, \ldots, (+\sqrt{M} - 1)\}$, where $M$ is the constellation size. The Least Significant Bit (LSB) of $s_j$ is not used as all the numbers in $\Omega$ are odd numbers. The structure of the MU block is such that the odd multiples of $r_{ij}$ (i.e., $r_{ij}, 3r_{ij}, 5r_{ij}, 7r_{ij}$) are generated using the shift-and-add operations then using the MUXs, the right multiple is sent to the output, the choice of which is determined by the value of $s_j$. This way of implementation of the multiplication operation is much faster than a normal multiplier implementation. As will be discussed in Section 4.2.6, the motivation for designing the MU block is due to the fact that this multiplication lies in the critical path of the architecture, which is on a feedback path. Since the

$^4$All number representations are in 2’s complement.
Figure 4.6: The architecture of the Mapper, where \( \tilde{s}_l^{[0]} = 2\left[ \frac{s_l^{[0]}}{2} + 0.5 \right] - 1. \)

Figure 4.7: The architecture for the Limiter block.

The fine-grained pipelining technique cannot be used in the feedback path to improve the overall throughput, an efficient implementation of the multiplier using this scheme is critical to enhance the maximum operating frequency.

**Mapper**

Once \( s_l^{[0]} \) (see Table 3.3) was calculated based on \( \tilde{L}_l(s^{[l]}) = L_l(s^{[l]})/r_{ll} \), the first child needs to be calculated by mapping the value \( s_l^{[0]} \) to the nearest point in \( \Omega \) (slicing). This is done in two consecutive stages. First \( s_l^{[0]} \) is mapped to its nearest odd integer number \( (\tilde{s}_l^{[0]}) \) using the Mapper block and then if \( \tilde{s}_l^{[0]} \) is outside the allowed boundary of \( \Omega \), it will be bounded by the Limiter block to generate \( s_l^{[1]} \). The process of mapping to the nearest odd integer number is implemented by the Mapper block shown in Fig. 4.6. The reason behind this implementation is the fact that the nearest odd number to \( s_l^{[0]} \) is \( 2\left[ \frac{s_l^{[0]}}{2} + 0.5 + 0.5 \right] - 1 = 2\left[ \frac{s_l^{[0]}}{2} + 1 + 0.5 \right] - 1 \), where \( [\cdot] \) represents the truncation operation.
The process of limiting the value in the predefined range is done through the Limiter block. In other words, if \( s_i^{[0]} \) is outside the boundaries of \( \Omega \), the Limiter block guarantees that the upper/lower bounds (e.g., +7/-7 in 64-QAM) are chosen as the selected points in \( \Omega \). The Limiter block is shown in Fig. 4.7 with examples on the action taken on \( s_i^{[0]} \) to determine \( s_i^{[1]} \) for three different cases.

### 4.2.2 Level I

The input to Level I is \( r_{88} \) and \( \bar{z}_8 \) and its output is the PEDs of all the elements in \( \Omega \), which are the nodes in the 8-th level of the tree. The detail of the architecture is shown in Fig. 4.8. It employs a 13-bit×13-bit multiplier, eleven adders and the absolute value block. Note that the absolute value block, representing the \( \ell^1 \)-norm, can be replaced by a squaring operation block, \( \ell^2 \)-norm, which can be easily implemented using a carry-save-adder technique [3]. However, simulation results show that the difference in the BER performance is negligible [2]. Fig. 4.9 confirms this and shows the performance result for a 4×4, 64-QAM MIMO system with \( K = 10 \) for \( \ell^1 \)-norm and \( \ell^2 \)-norm cases. It shows that the performance result for both cases is almost
4 VLSI Implementation of a Scalable K-Best Detector

Figure 4.9: The performance of a $4 \times 4$ 64-QAM MIMO system with $K = 10$ for $\ell^1$-norm and $\ell^2$-norm case.

the same but due to the low-complexity nature of the $\ell^1$-norm, it is the preferred approach for the implementation.

Since the Level I is on the feed-forward path of the architecture, a fine-grained pipelining technique can be employed inside the block in order to increase the system throughput. A two-stage pipelining is employed in Level I, which is shown by two and five positive-edge-triggered registers/flip-flops added in stage one and two, respectively (Fig. 4.8). In fact, by using the registers the block is broken down to two consecutive stages, which avoids a long critical path and implies that the critical path of the architecture contains only one multiplication. It is assumed that all the registers used in this thesis are triggered by the positive-edge of the clock.

4.2.3 Level II

The input to Level II is the PED values of the 8-th level and its output is the PED values of the first children in the 7-th level of the tree. In fact, in the Level II block, the first children of the eight nodes in the 8-th level are determined. Note that due to the structure of the $R$ matrix in (4.2), the first children in the 7-th level of the tree are all the same and independent of their parents in level 8 (because $r_{78} = 0$). This
child is determined and is used to calculate the updated PED values of the nodes in the 7-th level. Since \( r_{77} = -r_{88} \), no extra input is required for the calculations in Level II. The equation of the 7-th level can be written as

\[
r_{77}\bar{z}_7 = r_{77}s_7 + \nu_7 \Rightarrow \bar{z}_7 = s_7 + \frac{\nu_7}{r_{77}}.
\] (4.4)

This implies that in order to find the first child in the 7-th level, \( \bar{z}_7 \) is applied to the input of the Mapper/Limiter block whose output is the first child. The architecture of the Level II block is shown in Fig. 4.10. Once the first child was determined it is multiplied by \( r_{77} \) using the MU block. The input normalized \( \bar{z}_7 \) value is also multiplied by \( r_{77} \) after which the Euclidean distance between the first child and the received vector (i.e., \(|r_{77}\bar{z}_7 - r_{77}s_7|\)) is calculated and the result is added to the PED values of the 8-th level PEDs to derive the eight updated PEDs of the 7-th level. The fine-grained pipelining technique has also been employed in this block to break it into four stages in order to limit the length of the critical path. Note that in order to guarantee the correct functionality of the architecture after the pipelining, three stages of registers are required to be inserted on all input PEDs coming from the Level I block (i.e., \( T_1(-1), \ldots , T_1(+7) \)).
4.2.4 Sorter Block

The input to the Sorter block is the set of eight PED values of the 7-th level FCs and the main task of this block is to generate the sorted list of these PED values. The architecture of the Sorter is shown in Fig. 4.11. The eight inputs are denoted by $D_0-D_7$. The Ctrl signal is used to load the data in one clock cycle. Using this architecture, it takes four clock cycles to sort all the eight PED values, which are stored in the positive-edge-triggered registers after the comparators. This architecture can be used as a general sorter, which sorts $K$ numbers in $K/2$ clock cycles. This efficient architecture performs the sorting operation twice as fast as a bubble sorter [44] because two consecutive minimizations/maximizations are implemented in one clock cycle between two registers. One such set of consecutive minimizations is highlighted in Fig. 4.11, which is also the critical path of the Sorter block. Since the global critical path of the MIMO detector is larger than that of the Sorter block, the two consecutive minimizations do not limit the total throughput. Note that the factor $N$ on the registers, shown in Fig. 4.11, represents a register bank (RB) of length $N$ bits, used to store the child list (path history) as well as the updated PED values.

4.2.5 PE I Block

PE I is a general block used for all the levels from level 7 to level 2. It receives the sorted list of the first children of each level and generates the $K$ best candidates of

---

5. The comparators are realized using the blocks provided in the ARM standard cell library.

6. The path list grows from one level to another so does the value of $N$. This means that the value of $N$ is different for each PE I/II stage.
that level. For instance the output of the PE I in level 7, called NC-L7 in Fig. 4.3, is the $K = 10$ consecutive best candidates with the lowest PED values in level 7, generated one-by-one in series at the output. In other words, PE I implements Steps 4.1-4.5 of the algorithm in Table 3.4. The architecture of PE I is shown in Fig. 4.12. It consists of a sorter, and a block called NC-Block on the feedback path. In fact, PE I receives the sorted list of the PEDs from the preceding stage. It finds the best one with the lowest PED and announces it as the next K-Best candidate at the output, and then calculates the next best sibling of the announced child through the NC-Block and feeds it back to the sorter to locate the correct location of the new sibling in the already sorted list in PE I. Finally it performs the comparison and announces the next K-Best candidate in the next clock cycle. The following points clarify the details of this architecture:

- The main task of the sorter in this block is to receive a sorted list and finds the right position of a new entry in the sorted list, while announcing the entry with the lowest PED every clock cycle. In other words, as a new entry comes into the sorter, it walks through the sorter and finds its correct ordered position. This position is updated every clock cycle by the introduction of the new sibling from the feedback path.

- Before the sorted PED values of the preceding stage are loaded into the PE I block, there is a reset signal, Rst in Fig. 4.12, that initializes all the register banks (except the one attached to the output) to the maximum possible number.
This is necessary to avoid any interference from the previous values stored in them and makes them ready to process the new list. The Rst signal also initializes the control signal, Ctrl in Fig. 4.12, to zero whose value increases every clock cycle. The Ctrl signal is used to load the sorted list from the preceding stage to the PE I block. Note that the data in the sorted list is loaded one pair at the time. For instance, when Ctrl=0, D₀ and D₁ are loaded and when Ctrl=4, D₈ and D₉ are loaded. The reason is to guarantee the proper functionality of the architecture when PE I and PE II are co-operating together. Additional details will be provided in Section 4.2.7. A snapshot of the signal transitions and the relation between the Clk, Rst, and Ctrl signals are also shown by an example in Fig. 4.12.

• The critical path of the PE I block is highlighted in Fig. 4.12. It contains a MUX, a comparator and the NC-Block. The main task of the NC-Block is to determine the next best sibling of an already announced best child. It also finds the PED value of this sibling and sends the information to the sorter part of the PE I block. Since the NC-Block is on the feedback portion of the architecture, pipelining cannot help to increase the throughput of the architecture. In fact, this path is the critical path of the whole MIMO architecture presented in Fig. 4.3. Therefore, an efficient architecture needs to be proposed for the NC-Block to ensure an overall high-throughput architecture.

### 4.2.6 NC-Block

The detail of the NC-Block architecture is shown in Fig. 4.13. As was addressed in Table 3.3, the main task of the NC-Block is to determine the next sibling of the currently announced best child using the SE enumeration technique. Thus, the NC-Block in the i-th level needs to calculate the number of jumps nᵢ (in Table 3.3), the direction of the next move, SignBit, and finally calculates the PED value of the new sibling\(^7\). These three tasks are implemented by the architecture shown in Fig. 4.13, where SignBit determines the direction of the SE enumeration for the next child and Uout(Lout) determines if the SE enumeration has reached the upper (lower) boundary of the \(\Omega\) set. In fact, according to (3.4)-(3.5), the PED value of the

\(^7\)The signal SB in Fig. 4.13 represents the sign bit of the result of the adder.
new sibling can be determined as follows:

\[ T_l = T_{i+1} + |L_l - r_l s_l| = T_{i+1} + r_l |\bar{L}_l - s_l|, \tag{4.5} \]

where \( L_l \) refers to the quantity defined in (3.6) while \((s^{(l)})\) was omitted for brevity of discussion and \( \bar{L}_l = \bar{z}_l - \sum_{j=l+1}^{2N_{l-1}} \bar{r}_j s_j \). As mentioned, any effort to simplify this block and/or reduce its critical path, has a direct and significant effect on the total achievable data rate. In order to optimize its critical path, the following two efficient techniques were utilized in our VLSI architecture:

1. Avoid multiplication: Since the value of \( \bar{L}_l \) in (3.6) depends only on the selected symbols up to level \( l \) and is independent of the current sibling \((s_l)\), the values of \( \bar{L}_l \) and \( \bar{L}_l r_l \) can be calculated using the FC-Block in the preceding block\(^8\) (Section 4.2.8) and forwarded to the NC-Block as an input (see Fig. 4.13). This is a preferred approach as the required multiplication to calculate \( \bar{L}_l r_l \) will be rescheduled and removed from the critical path and is shifted to a block that is pipelineable. Moreover, the second multiplication, i.e., \( r_l s_l \), is realized using the MU block.

\(^8\)For PE I of NC-L7, the preceding block is Level II. For all other PE I blocks, this block is the PE II block in the preceding stage. For instance for PE I in NC-L3, this is done in PE II in FC-L3.
2. **Broken critical path:** As can be seen from the NC-Block architecture (Fig. 4.13), the critical path has three adders (one 4-bit and two 16-bit adders), as well as the MU block. The critical path associated with this architecture is 4.8ns in 0.13 μm CMOS technology using the ARM standard cell library. The first part of the critical path (specified by 1-st section in Fig. 4.13) calculates the next sibling, which can be transferred to the FC-Block in the preceding block. This means that the FC-Block would calculate both the first and second best child of each parent and sends them to the NC-Block. The NC-Block calculates the PED value of the second best child while determining the third best child and so on. This implies that the NC-Block block always calculates one child ahead. Using this approach in our ASIC implementation yields a critical path of length 3.65ns, thus higher overall throughput. The use of this scheduling technique effectively breaks the critical path of the NC-Block down into two smaller parts (1-st and 2-nd section in Fig. 4.13). This is shown in Fig. 4.14 with the improved critical path. The first section of the NC-Block on the right hand side of Fig. 4.14 is denoted by NCSub, which is the block that will be added to the
4 VLSI Implementation of a Scalable K-Best Detector

Figure 4.15: The architecture for the PE II block with the critical path highlighted.

preceeding FC-Block in order to calculate the second best child.

In brief, by forwarding all the processing required to start the second section in the NC-Block to the preceding FC-Block, the critical path is significantly improved.

4.2.7 PE II Block

The output of PE I is the serial list of K-Best candidates of the the current level, generated one-by-one at the output. As each of the K-Best candidates is generated, it is sent to the PE II block to calculate the first children of the next level and sort them as they arrive. The architecture of the PE II block is shown in Fig. 4.15, where $D_{in}$ is the input port and $D_0$-$D_9$ are the output ports. At the beginning, the first child of the K-Best candidate of the previous stage and its updated PED value are calculated by the FC-Block, and then using the sequential sorter, the calculated PED values are sorted as they arrive. Note that this process is performed on a cycle basis since the PE II block is connected to the output of the PE I block in a pipelined fashion.

In the proposed architecture for PE II, the sorted PEDs are stored in the register banks, depicted by $N$-bit registers in Fig. 4.15 and denoted by $RB_0$ – $RB_8$. At every clock cycle, two register banks are updated at the same time. This is because of the fact that the registers on the upper part of the sorter are located in every other stage. This is required to guarantee the correct functionality of the PE I. The functionality of the sorter is such that the larger values are shifted to the right while the smaller values are shifted to the left. Once the last element (10-th element in 64-QAM) enters the sorter, it updates the first two register banks, thus the first two are guaranteed to have the two smallest PED values. Therefore, at the next clock cycle, they can be transferred to the following PE I block. After the second clock cycle, the next two register banks are updated and they are also ready to be transmitted. Therefore, the PED values are transferred to the next level on a pair-by-pair basis. This fact is
shown in Fig. 4.3 with grey lines between the PE II block and the PE I block and the numbers on them represent the number of clock cycles after the arrival of the last K-Best candidate to the PE II in which they are transferred. This transfer is performed only once every $K$ clock cycles.

The reason that the data is transferred one pair at a time can be understood in Fig. 4.16. This example is for $K = 6$, and six register banks in PE I, which are represented by $RB_0$-$RB_5$. In fact, Fig. 4.16 shows the internal register updates between different register banks in PE I. The first schedule (Fig. 4.16.a) shows the transitions when the data is transferred in a pair, while the second schedule (Fig. 4.16.b) shows the case where the data is pushed into PE I one at a time. The expected correct sorted list for this example at the output of the PE II block is \{1, 5, 10, 12, 15, 17\}. For example, in the first schedule, \{1, 5\} are transferred to PE I in one clock cycle and in the next cycle, \{10, 15\} are transferred. However, for the second schedule, each element in the FC list is fed one-by-one to the PE I block. Note that in the PE I block, each FC that is chosen as the best candidate is fed back to the NC-Block to calculate its next sibling. This is represented in Fig. 4.16 by the arrows and the associated PED value of the next sibling next to it. For instance, the PED value of the next sibling of the candidate with PED=1 is 21. The first register bank $RB_0$ is connected to the output, which represents the list of the K-Best candidates of the current level. This output is highlighted by the grey line in the schedules (the first row of each schedule).
Figure 4.17: The timing scheduling between a typical pair of PE II and PE I.

Note that the one-by-one strategy, i.e., Fig. 4.16(b), results in the wrong result at the output (i.e., \{1, 5, 21, 10, 12, 15\}), while feeding the data one pair at a time (Fig. 4.16(b)) results in the proper functionality (i.e., \{1, 5, 10, 12, 15, 17\}).

Note also that once the last element comes in and the first two register banks are sent to the next stage, the internal min/max functions should be initialized to the highest positive number to avoid the comparison between the first element of the next iteration and the last element of the current iteration. This is implemented through the introduction of a MUX before the min/max functions and is controlled by a control signal (signal \textit{C} in Fig. 4.15). This control signal is incremented every clock cycle and is initialized to zero at the end of each iteration. For instance, when \textit{C}=1, the input to the first two min/max functions are initialized to the largest 16-bit number (i.e., \texttt{16'b1}), thus once the first K-Best candidate of the next iteration comes in, it would not be compared with the previous stored values in the register banks from the previous iteration. This makes the core utilization 100\% as PE I and PE II are fully pipelined with zero latency with respect to one another.

The scheduling between PE II and PE I blocks along with their control signals such as signal \textit{C}, Ctrl(PE I), and Ctrl(PE II) is shown in Fig. 4.17. Two \textit{Rst} signals are two input signals whereas the other signals are internal signals that perform the scheduling of various data exchange happening in the two blocks. For instance, one clock cycle after the Ctrl signal of PE II becomes high, the new input signals are inserted into PE II and the internal signal \textit{C} is initialized, which guarantees the proper implementation of comparison and avoids the interference between two consecutive iterations. It takes 5 clock cycles to insert all the entries into PE II and takes 5 more clock cycles to generate the first two FC candidates in the output of PE II. This is
4.2.8 FC-Block

The main task of the FC-Block is to calculate the $\bar{L}_l$ value in (3.6), the first child of the current parent based on the calculated $\bar{L}_l$, and calculates its PED. It also determines the second best child and its corresponding PED value as well as the $\bar{L}_{rll}$ value for its following NC-Block as mentioned in Section 4.2.6. The proposed architecture for the FC-Block is shown in Fig. 4.18. In order to increase the total throughput in the
architecture, pipelining has been used by the introduction of the registers on all the forward paths. It increases the throughput at the cost of additional latency in the circuit.

The proposed architecture for the FC-Block consists of five pipeline levels. Each FC-Block is used inside a PE II block. In the first pipeline level (shown in Fig. 4.18), there are six MU blocks. However, depending on the PE II block in which it is used, only part of these MU blocks are used. For instance, for the PE II blocks of stage FC-L6 and FC-L5, only the first two MUs are implemented. For PE II blocks of stage FC-L4 and FC-L3, the first four MUs and finally for PE II blocks of stage FC-L2 and FC-L1, all of the six MUs are implemented.

The first two levels of the architecture calculates the $\sum_{j=l+1}^{2N_T} \bar{r}_{lj}s_j$ in (3.6) and in the third level, the value of $\bar{L}_l$ is calculated. Using the $\bar{L}_l$, based on the steps I.1 and I.2 of the algorithm in Table 3.3, in level 4, the first child is calculated using the Mapper and Limiter blocks in the FC-Block. The number of moves as well as the direction of the move for the SE enumeration to determine the next best child required by the NC-Block are also determined in level 4 through the SignBit, Uout, and Lout signals. Finally the blocks in level 5 calculate the PED value of the announced first child. It also calculates the second best child through the introduction of the NCSub block, which was described in Section 4.2.6. The updated values of SignBit, Uout, and Lout as well as the second best child and its scaled value are sent to the output.

All of the above blocks are interconnected together in a pipelined fashion and every clock cycle data exchange occurs between the adjacent blocks. This means all the data are calculated and transferred sequentially operand-by-operand between the blocks. A proper scheduling scheme at the input of the chip guarantees the delivery of the correct $\bar{r}_{lj}$ and $\bar{z}_l$ values to the blocks.

### 4.2.9 Latency and Bit-true Simulation

The fine-grained pipelining used inside the blocks improves the throughput at the cost of the larger latency. Starting from the first block, the latency of Level I is 2 cycles, Level II has a 3-cycle latency, the Sorter block’s latency is 4 cycles, the PE I block’s latency is $K = 10$ cycles, and finally PE II’s latency is 10 cycles plus an additional 6 cycles for the pipelined FC-Block. Therefore, according to the architecture in Fig. 4.3, the total latency of the architecture is $2 + 3 + 4 + 6 \times 10 + 6 \times 16 = 165$ cycles.
Table 4.1: Fixed-point Word-Length (bits) of Parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Word Length†</th>
</tr>
</thead>
<tbody>
<tr>
<td>(r_{ii})</td>
<td>[13:12]</td>
</tr>
<tr>
<td>(r_{ij})</td>
<td>[16:11]</td>
</tr>
<tr>
<td>(z_i)</td>
<td>[13:4]</td>
</tr>
<tr>
<td>(s_i)</td>
<td>[3:0]</td>
</tr>
<tr>
<td>(T_i)</td>
<td>[16:13]</td>
</tr>
<tr>
<td>(L_i)</td>
<td>[16:7]</td>
</tr>
</tbody>
</table>

† \([n : m]\) an \(n\)-bit number with \(m\) bits for the fractional part.

Table 4.1 shows the number of bits associated with different variables in the algorithm for the bit-true simulation\(^9\) as well as the hardware implementation for the case of a 4 × 4, 64-QAM constellation in the form of \([n : m]\), where \(n\) and \(m\) denote the total number of bits for and the number of bits in the fractional part, respectively. The fixed-point simulations are performed using the 2’s complement number representation. Note that the word lengths in Table 4.1 have been derived based on extensive bit-true simulation results to minimize the BER loss relative to the floating-point result (i.e., less than 0.5 dB at BER = 10\(^{-3}\)). This means that based on the long floating-point simulations, the dynamic range of all the variables were determined, based on which the required number of bits for integer and fractional parts were calculated.

In the proposed architecture, the critical path (in NC-Block) is fixed and independent of the constellation size while providing the exact K-Best performance result. In fact, the complexity of our approach only depends on the value of \(K\), which scales sub-linearly with the constellation order. This is one of the most promising features of our scheme and distinguishes it from the other alternatives in the literature. It is practically attractive for next-generation WiMAX/LTE systems where the implementation of 64-QAM and 256-QAM schemes are required.

### 4.3 Complexity Analysis

Table 4.2 shows the complexity comparison between different schemes. For the sake of comparison, the number of visited children (expand), the required number of clock cycles to do the sorting (sort), and the total latency are considered. The values listed in the expand row refers to the number of PED values required to be calculated, which directly translates to more area and power. A key feature in our architecture is that

---

\(^9\)Bit-true simulation refers to the simulation results with finite word-length effect, which is also called the fixed-point simulation.
Table 4.2: Comparison of Different K-Best Implementations.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Metric</th>
<th>3</th>
<th>2</th>
<th>54</th>
<th>This work</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 × 4</td>
<td>expand</td>
<td>160</td>
<td>160</td>
<td>160</td>
<td>72</td>
</tr>
<tr>
<td>16-QAM (K=5)</td>
<td>sort(cycles)</td>
<td>160</td>
<td>40</td>
<td>280</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>latency(cycles)</td>
<td>240</td>
<td>49</td>
<td>440</td>
<td>40(85)†</td>
</tr>
<tr>
<td>4 × 4</td>
<td>expand</td>
<td>640</td>
<td>640</td>
<td>640</td>
<td>152</td>
</tr>
<tr>
<td>64-QAM (K=10)</td>
<td>sort(cycles)</td>
<td>640</td>
<td>80</td>
<td>900</td>
<td>74</td>
</tr>
<tr>
<td></td>
<td>latency(cycles)</td>
<td>720</td>
<td>99</td>
<td>1540</td>
<td>80(165)</td>
</tr>
<tr>
<td>4 × 4</td>
<td>expand</td>
<td>1920</td>
<td>1920</td>
<td>1920</td>
<td>232</td>
</tr>
<tr>
<td>256-QAM (K=15)</td>
<td>sort(cycles)</td>
<td>1920</td>
<td>120</td>
<td>2700</td>
<td>113</td>
</tr>
<tr>
<td></td>
<td>latency(cycles)</td>
<td>2000</td>
<td>149</td>
<td>4620</td>
<td>120(245)</td>
</tr>
<tr>
<td>M ↑</td>
<td>critical path</td>
<td>↑</td>
<td>↑</td>
<td>↑</td>
<td>constant</td>
</tr>
</tbody>
</table>

† The number in parenthesis represents the latency after pipelining.

only 2K − 1 children need to be calculated in each level whereas in others, e.g. [2], the PED of all the children of all parent nodes should be calculated. The last row of the table shows whether the length of the critical path of the architecture grows with the constellation order. The number in parenthesis refers to the latency of the pipelined architecture.

4.4 Simulations

Since the K-Best algorithm is not an ML-optimal detector, it might miss the hard-ML point, resulting in performance loss. However, for a proper choice of K, its bit-error-rate (BER) approaches the optimal case for a reasonable range of SNR values. In the following, the simulation results for a single-carrier 4 × 4 MIMO system for different constellation sizes, i.e., 16-QAM, 64-QAM, and 256-QAM, for both the floating-point and hardware results (fixed-point) are presented in Fig. 4.19, Fig. 4.20, and Fig. 4.21, respectively. In all the simulations, the K-Best approach has been tested for 100,000 packets, where each packet consists of 96 bits\textsuperscript{10} (9.6Mbits in total). Test vectors are created using: (i) pseudo-random data, (ii) complex-valued random Gaussian channel matrix $\tilde{H}$ with statistically independent elements updated per four channel use, and (iii) additive white Gaussian (circularly symmetric) complex random noise. The value

\textsuperscript{10}This is because $96 = 4(N_T) \times 6$ (Number of bits per constellation symbol) \times 4 (Channel updates every four channel use).
Figure 4.19: K-Best floating/fixed-point vs ML for $4 \times 4$, 16-QAM with $K = 5$.

Figure 4.20: K-Best floating/fixed-point vs ML for $4 \times 4$, 64-QAM with $K = 10$. 
of $K$ is chosen to be 5, 10 and 15 for 16-QAM, 64-QAM, and 256-QAM, respectively, which are typically used in all the K-Best designs in the literature. The following simulation results confirm that these $K$ values guarantee that the performance loss at BER=$10^{-3}$ with respect to the ML is less than 1 dB. Based on the given simulations in the normal range of interest (i.e., 0 to 20 dB for 16-QAM, 0 to 30 dB for 64-QAM and 0 to 36 dB for 256-QAM) the performance loss is negligible. The fixed-point results for all cases are within a reasonable distance from the floating-point results (i.e., less than 0.5 dB at BER = $10^{-3}$). The number of bits for various parameters in the fixed-point simulation is based on Table 4.1. Note that the value of $K$ scales sub-linearly with the constellation size, i.e., $K = 5$ for 16-QAM, $K = 10$ for 64-QAM, and $K = 15$ for 256-QAM. Since the complexity of the proposed scheme in this dissertation is proportional to the value of $K$, we can conservatively conclude that the proposed ASIC scales sub-linearly with the constellation size.

4.5 Extension to 256-QAM Scheme

It is worth looking at the possible extension of the above work to a 256-QAM constellation scheme for a $4 \times 4$ MIMO system. Based on the simulation results in Section
Table 4.3: Hardware Increase from 64-QAM to 256-QAM

<table>
<thead>
<tr>
<th>Block</th>
<th>Increase</th>
</tr>
</thead>
<tbody>
<tr>
<td>MU</td>
<td>100 %</td>
</tr>
<tr>
<td>Mapper</td>
<td>0 %</td>
</tr>
<tr>
<td>Limiter</td>
<td>0 %</td>
</tr>
<tr>
<td>Level I</td>
<td>&lt;100 %</td>
</tr>
<tr>
<td>Level II</td>
<td>&lt;100 %</td>
</tr>
<tr>
<td>Sorter</td>
<td>100 %</td>
</tr>
<tr>
<td>PE I</td>
<td>50 %</td>
</tr>
<tr>
<td>PE II</td>
<td>50 %</td>
</tr>
<tr>
<td>NC-Block</td>
<td>0 %</td>
</tr>
<tr>
<td>FC-Block</td>
<td>&lt;50 %</td>
</tr>
</tbody>
</table>

4.4, the appropriate value for $K$ in 256-QAM framework is $K = 15$. Taking that into account, we investigate how various blocks in the proposed architecture for the K-Best algorithm in this chapter scale accordingly. Table 4.3 shows the hardware increase with respect to the 64-QAM implementation. The MU block takes 100% more hardware as the number of input odd-integers is doubled from 64-QAM to 256-QAM. The Mapper and Limiter blocks do not change with the constellation size. Level I, Level II, require less than 100% more hardware as they need to calculate 16 PED values (instead of 8) but they still require one multiplication operation. Sorter block requires twice as much circuitry and that is because 16 candidates should be sorted. PE I and PE II blocks require 50% more hardware because of a larger value of $K = 15$. The hardware increase in the NC-Block block is negligible as only its MU block scales up in the case of 256-QAM. Finally the FC-Block block requires less than 50% more hardware because the number of input calculations increases by 50% while the internal logic of the block remains the same except any instantiation of the MU block. Therefore, the overall increase in the hardware requirements is less than 50% considering the fact that the PE I, and PE II blocks form the main part of the architecture.
4.6 Design Comparison

The proposed VLSI architecture was modeled in Verilog HDL, synthesized using Synopsys Design Compiler and placed and routed using Cadence SoC Encounter/Silicon Ensemble. The RTL and gate level netlists were verified with the golden model generated from the fixed-point MATLAB model. The final ASIC was fabricated in 0.13 µm IBM CMOS technology using ARM standard library cells. Table 4.4 give an overview and comparison of all the reported ASIC implementations in the literature for a 4 × 4 16-QAM and 64-QAM MIMO system, respectively, including this work. The applied algorithm and the value of the $K$ parameter in case of the K-Best algorithm, the fabricated technology, as well as some detailed specifications of each ASIC are listed. If information on other designs is not provided by the authors, this is indicated by the entry N/A in Table 4.4. From this table, the following points can be inferred:

1. Most of the ASIC implementations are for a 16-QAM constellation scheme. This is mostly because of the fact that both the expansion and sorting scheme for this constellation are simpler with a much lower hardware complexity.

2. Scalability refers to how the expansion scheme, sorting scheme and/or the critical path of the proposed ASIC scales with the constellation size. Most of the approaches have a complexity order of $O(K|Ω|)$ meaning that the value of $K$, the number of expanded children and/or the critical path increase with the constellation size $Ω$, which significantly reduces the maximum clock frequency and hence the throughput. This thesis, however, introduces an approach that scales sub-linearly. This is due to the fact that the number of expanded and sorted children in this thesis increase only with the value of $K$ and are independent of the constellation size whereas in the other K-Best implementations they increase according to the constellation size. For instance, assuming that $K = 5$ and $K = 10$ are used for 16-QAM and 64-QAM, respectively, using a normal K-Best implementation, the number of expanded children in each level for 16-QAM is 20 whereas for 64-QAM this number is 80, thus 4 times as much. However, using the method introduced in this dissertation, for 16-QAM, and 64-QAM this number is $K \times 2 - 1 = 9$, and 19, respectively, almost twice as much, and hence a sub-linear increase. This provides a significant saving towards an area efficient implementation for 64-QAM and higher-order constellations.
Table 4.4: Comparison of the Current ASIC Implementations of 4 × 4 MIMO Detectors.

<table>
<thead>
<tr>
<th>Reference</th>
<th>[3]</th>
<th>[3]</th>
<th>[2]</th>
<th>[41]</th>
<th>[44]</th>
<th>[2]†</th>
<th>[47]</th>
<th>[4]</th>
<th>[51]</th>
<th>This work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modulation</td>
<td>16-QAM</td>
<td>16-QAM</td>
<td>16-QAM</td>
<td>16-QAM</td>
<td>16-QAM</td>
<td>64-QAM</td>
<td>64-QAM</td>
<td>64-QAM</td>
<td>64-QAM</td>
<td>64-QAM</td>
</tr>
<tr>
<td>K-value</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>N/A</td>
<td>10</td>
<td>10</td>
<td>64</td>
<td>64</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>Process</td>
<td>0.35 μm</td>
<td>0.13 μm</td>
<td>0.25 μm</td>
<td>0.25 μm</td>
<td>0.35 μm</td>
<td>0.25 μm</td>
<td>0.13 μm</td>
<td>65 nm</td>
<td>90 nm</td>
<td>0.13 μm</td>
</tr>
<tr>
<td>Core Area†</td>
<td>91 KG</td>
<td>97 KG</td>
<td>93 KG</td>
<td>50 KG</td>
<td>52 KG</td>
<td>135 KG</td>
<td>5270 KG</td>
<td>174 KG</td>
<td>294 KG</td>
<td>114 KG</td>
</tr>
<tr>
<td>Max Freq.</td>
<td>100 MHz</td>
<td>200 MHz</td>
<td>132 MHz</td>
<td>71 MHz</td>
<td>100 MHz</td>
<td>52 MHz</td>
<td>270 MHz</td>
<td>200 MHz</td>
<td>166 MHz</td>
<td>282 MHz</td>
</tr>
<tr>
<td>Throughput</td>
<td>54 Mbps</td>
<td>107 Mbps</td>
<td>424 Mbps</td>
<td>170 Mbps</td>
<td>10 Mbps</td>
<td>83 Mbps</td>
<td>100 Mbps</td>
<td>115 Mbps</td>
<td>95 Mbps</td>
<td>675 Mbps</td>
</tr>
<tr>
<td>Latency</td>
<td>2.4 μs</td>
<td>1.2 μs</td>
<td>0.4 μs</td>
<td>N/A</td>
<td>N/A</td>
<td>1.7 μs</td>
<td>N/A</td>
<td>N/A</td>
<td>0.25 μs</td>
<td>0.6 μs</td>
</tr>
<tr>
<td>Power</td>
<td>626 mW</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>847 mW</td>
<td>11 mW</td>
<td>N/A</td>
<td>135 mW</td>
</tr>
<tr>
<td>Energy/bit*</td>
<td>594pJ/b</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>8470pJ/b</td>
<td>765pJ/b</td>
<td>N/A</td>
<td>200pJ/b</td>
</tr>
<tr>
<td>Scalability</td>
<td>$K</td>
<td>\Omega</td>
<td>$</td>
<td>$K</td>
<td>\Omega</td>
<td>^{\dagger}</td>
<td>$K</td>
<td>\Omega</td>
<td>^{\dagger}</td>
<td>$K</td>
</tr>
<tr>
<td>SNR Dep.</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Domain</td>
<td>Real</td>
<td>Real</td>
<td>Real</td>
<td>Complex</td>
<td>Real</td>
<td>Real</td>
<td>Complex</td>
<td>Complex</td>
<td>Complex</td>
<td>Real</td>
</tr>
<tr>
<td>Tested Chip</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

† Although a 64-QAM system has not been implemented in [2], the complexity results for $K = 10$ has been reported, which is equivalent to that used in our 64-QAM design. We use this as a set of optimistic values for comparison purposes.

‡ in kilo-gates (KG).

* Results of the designs in [3] and [4] have been scaled to a 0.13μm equivalent CMOS process.
3. Comparing this thesis to [47], which are both fabricated in 0.13 \( \mu m \) CMOS, reveals a significant reduction in the area achieved using our proposed scheme\textsuperscript{11}. This is because of the fact that only a tiny fraction of all the children are expanded in our architecture. Moreover, \( K = 64 \) in [47] is in the complex domain as opposed to \( K = 10 \) in our case in the real domain with a finer granularity.

4. Since for 64-QAM, both the value of \( K \) increases and the critical path of the sorting scales up\textsuperscript{12}, [2], the fastest 64-QAM implementation reported to date, [4], operates at 115 Mbps. Since the problem of scaling was alleviated in this dissertation, the ASIC implementation achieved a significantly higher throughput of 675 Mbps with a maximum operating clock frequency of 282 MHz. Therefore, the achieved throughput in this thesis is almost 5.8\texttimes higher than the best implementation to-date. This data rate guarantees the peak data rates required for next-generation WiMAX (IEEE 802.16m standard)\textsuperscript{13} and LTE systems\textsuperscript{14}.

5. ML-like refers to a performance result close to the optimal performance. Most schemes such as [47], [51], and [4] have performance loss compared to the strict K-Best algorithm method due to the approximated/relaxed schemes whereas this thesis has higher throughput while providing the exact K-Best results.

6. The proposed architecture in this work has a significant advantage of having SNR-independent throughput, which is not the case for some of the designs in the literature (see Table 4.4. For instance, in [4] by changing the SNR value from 17.7 dB to 16.2 dB, the throughput decreases from 115 Mbps to 42 Mbps.

\textsuperscript{11}The average gate density of the ARM 0.13 \( \mu m \) standard cell library is 120 KG/mm\textsuperscript{2}.

\textsuperscript{12}Although a 64-QAM system has not been implemented in [2], the complexity results for \( K = 10 \) has been reported, which is equivalent to that used in our 64-QAM design. We use this as a set of optimistic values for comparison purposes.

\textsuperscript{13}Proposed peak data rates for IEEE 802.16m are as follows [58]:
- Very low rate Data: 16 Kbps
- Low rate Data & Low Multimedia: 144 Kbps
- Medium multimedia: 2Mbps
- High multimedia: 30 Mbps
- Super high multimedia: 30 Mbps \( \sim \) 100Mbps/1 Gbps

1 Gbps throughput is easily achieved in 65 nm CMOS, as FO4 gate delays scale from 20 ps in 0.13 \( \mu m \) CMOS technology of our prototype to 10 ps typical of 65 nm CMOS technology.

\textsuperscript{14}The peak data rates in LTE are up to 100 Mbps for downlink and 50 Mbps for uplink.
The die micrograph is shown in Fig. 4.22. Fig. 4.23 compares this thesis to previously published works. It compares the total achieved throughput as a function of the number of kilo-gates (KG) used in each design. The design for 16-QAM (▲) and 64-QAM (■) have been distinguished with different icons. For SNR-dependent schemes, their highest achieved throughput is reported. It can be seen that our implementation has the highest throughput ever reported for the $4 \times 4$, 64-QAM designs. As was expected, in general, the designs in 16-QAM take lower area as the value of $K$, the number of possible children per parents, and the sorting cores are almost fours times smaller compared to that of 64-QAM.

### 4.7 Test Results

The K-Best MIMO detector was fabricated in a 0.13 $\mu$m IBM CMOS-8LM process and was tested using an Agilent(Verigy) 93000 high-speed digital tester and a Temptronic TP04300 thermal forcing unit. The test setup consisting of the 93K SoC tester, Temptronic TP04300 thermal forcing unit, load board and the DUT is shown in Fig. 4.24. The core supply voltage is 1.2 V whereas the I/O voltage is 2.5 V. With a 1.2-V supply, the measure maximum clock frequency is 250 MHz. The functionality of the detector was verified by passing input vectors at different SNR values to the chip through the tester and comparing the detector outputs with the expected values from the bit-true simulations both from MATLAB and NC-Verilog simulations. The
Figure 4.23: Throughput vs. gate count compared to previously published works.

Figure 4.24: Test setup (Agilent/Verigy 93K tester, Temptronic TP04300 thermal forcing unit head, and the chip).
BER performance of the detector was measured as follows:

1. Complex-valued random Gaussian channel matrix, updated per four channel use, was generated and used to transmit the transmit symbols.

2. For a given SNR value, additive white Gaussian noise with the desired variance was generated and added to the received symbols.

3. Using MATLAB simulations, the detected bits for that specific SNR value were determined.

4. A test vector, including the input received symbols as well as all the required control and reset signals, was generated and passed to the tester.

5. Using the generated test vector, a VCD file is generated using NC-Verilog.

6. Using the VCD file, using the SmartTest tool [62], timing file (".tim"), configuration file (".pin"), and vector file (".binl") required for testing are generated.

7. Core supply voltage along with the I/O supply voltage are set appropriately.
8. At-speed test was run and the outputs are compared against the desired bit stream generated by the MATLAB simulation.

Fig. 4.25 and 4.26 show the effect of scaling the detector’s supply voltage on its maximum clock frequency and the corresponding power consumption, respectively. The dotted lines are the predicted values based on the MOS square-law equation with $V_t = 300 \text{ mV}$. It can be seen that the measured results closely follow the extrapolated results both for the maximum operating frequency and for the power consumption. These plots suggest that for lower-throughput applications, energy efficiency, defined as the amount of energy required for correct detection of a bit, can be improved using the same detector architecture/chip by operating at a lower supply voltage [63].

Fig. 4.27 shows a Shmoo plot depicting the maximum operating frequency and the total power dissipation of the design versus the supply voltage. A total of five chips were tested, where the average and the max/min values of the achieved frequency have been shown in Fig. 4.27. The detailed measurement results are presented in Appendix A in Table A.1 to Table A.15. Operating at a clock rate of 282MHz results in a measured sustained throughput of 675Mb/s dissipating 135mW @ 1.3V supply and 25°C. (This translates into a sustained throughput of 170Mb/s per transmit antenna in a $4 \times 4$ MIMO receiver.). The temperature was forced to be at 25°C using
Figure 4.27: Measurement plots for maximum frequency and power dissipation vs. supply voltage ($V_{dd}$) at 25°C.
Using a Temptronic TP04300 thermal forcing unit, test results at 0°C yield the plot shown in Fig. 4.28 for maximum operating frequency and power dissipation at 0°C. The results yields a clock rate of 291MHz while dissipating 139mW @ 1.3V producing a sustained data rate of 700Mbps. Test results at 80°C yield the plot shown in Fig. A.1 in the Appendix A for maximum operating frequency and power dissipation at 80°C. The results yields a clock rate of 250MHz while dissipating 104mW @ 1.2V producing a sustained data rate of 600Mbps.

Fig. 4.29 shows a comparison between the reported ASIC implementations of 4×4 64-QAM as well as the 16-QAM MIMO Detectors. Previous publications with measured or estimated power dissipations have been shown in the figure. The values of the [3] and [4] have been scaled to a 0.13µm equivalent CMOS process. The comparison graph confirms that measurements from this design achieves 8.3× better throughput per area compared to the best reported design and at the same time consumes 3.0× less energy per bit compared to the previous best design.
Figure 4.29: Measured throughput/area vs. energy/bit, with area measured in kilo-gates (KG) @ 282 MHz, 1.3 V and 25°C. Results of the designs in [3] and [4] have been scaled to a 0.13µm equivalent CMOS process.
It has been shown in [3] that by using efficient hardware implementation techniques at the last stage for child enumeration, throughput is not compromised when a soft detector is realized instead of a hard detector. (i.e., the soft detection core is not in the critical path of the design and hence it is entirely fair to compare soft and hard detectors with respect to overall throughput. i.e., the present thesis could be redesigned as a soft detector without compromising the measured throughput presented here.) However, it is acknowledged that an incrementally higher silicon area might be needed to implement the soft core. Therefore, it is worthwhile to compare this thesis to all previously published results looking at achieved throughput alone, which is shown in Fig. 4.30. Measured results of the present work demonstrate a $5.8\times$ improvement in throughput over [4], which is a soft detector. The energy efficiency improvement of the present work is $3.0\times$ better than [3] (200pJ/b vs. 594pJ/b), which is a hard detector.

The measured BER results are shown in Fig. 4.31. Test vectors used to test the chip and generate the BER curve below represent a total of 100,000 packets, where each packet consists of 96 bits (for total of 9.6Mbits of transmitted payload). Test
Figure 4.31: Measured BER at a clock rate of 282 MHz at a measured sustained throughput of 675Mb/s dissipating 135mW @ 1.3V supply and 25°C.

results agree with the expected golden vector set confirming correct operation of the test chip.

Table 4.7 summarizes the design and performance characteristics. Note that 5 working chips were tested (all that were available, whose detailed measurement results are presented in Appendix A), which resulted in the 50% yield throughput of 655 Mbps and 100% yield of 655 Mbps @ 1.3 V.

4.8 Summary

A novel high-throughput silicon implementation of the K-Best algorithm suitable for high-order constellation schemes has the smallest number of visited nodes, as well as the highest achieved throughput reported to-date. It is applicable to the unbounded infinite lattice decoding problem and has a fixed-critical path independent of the constellation scheme or the lattice size. The key contribution is the introduction of a novel joint on-demand expansion and distributed sorting scheme operating in a pipelined fashion. In 0.13 μm CMOS, it achieves 607 Mbps at 270 MHz satisfying the throughput requirements of next-generation WiMAX systems.
Table 4.5: Characteristics Summary of Detector and Measured Results.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modulation</td>
<td>64-QAM</td>
</tr>
<tr>
<td>MIMO Configuration</td>
<td>$4 \times 4$</td>
</tr>
<tr>
<td>Method</td>
<td>K-Best</td>
</tr>
<tr>
<td>K-Value</td>
<td>10</td>
</tr>
<tr>
<td>CMOS Process</td>
<td>0.13 $\mu$m 1P8M</td>
</tr>
<tr>
<td>Core Area</td>
<td>0.95 mm$^2$</td>
</tr>
<tr>
<td>Total Area</td>
<td>3.24 mm$^2$</td>
</tr>
<tr>
<td>Gate Count</td>
<td>114 KG</td>
</tr>
<tr>
<td>Package</td>
<td>PGA68</td>
</tr>
<tr>
<td>Core Supply</td>
<td>1.3 V</td>
</tr>
<tr>
<td>I/O Supply</td>
<td>2.5 V</td>
</tr>
<tr>
<td>Maximum Clock Frequency</td>
<td>282 MHz</td>
</tr>
<tr>
<td>50% Yield Throughput</td>
<td>675 Mbps</td>
</tr>
<tr>
<td>100% Yield Throughput</td>
<td>655 Mbps</td>
</tr>
<tr>
<td>Temperature (@ stated Throughput)</td>
<td>25°C</td>
</tr>
<tr>
<td>Power (@ max. clock freq.)</td>
<td>135 mW</td>
</tr>
<tr>
<td>Normalized Power (@ max. clock freq.)</td>
<td>0.2mW/Mbps</td>
</tr>
<tr>
<td>Energy Efficiency</td>
<td>200 pJ/bit</td>
</tr>
<tr>
<td>Leakage Power</td>
<td>$&lt; 1$ mW</td>
</tr>
<tr>
<td>Latency</td>
<td>165 Clock Cycles</td>
</tr>
<tr>
<td>Performance</td>
<td>Strict K-Best</td>
</tr>
<tr>
<td>Output Detection (soft/hard)</td>
<td>Hard</td>
</tr>
<tr>
<td>Detection Domain</td>
<td>Real</td>
</tr>
<tr>
<td>Fabricated and Tested</td>
<td>Yes</td>
</tr>
</tbody>
</table>
5 Joint Lattice-Reduction and K-Best Algorithm

5.1 Introduction

Theoretically, in the spatial multiplexing (SM) mode of MIMO systems, the maximum achievable diversity order is $N_R$, where $N_R$ is the number of received antennas [7]. However, there are two impairments at the system level that potentially result in less achievable diversity in SM MIMO systems (i) non-orthogonal channel columns, which is as a result of the correlation between the transmit or receive antennas, and (ii) sup-optimal detection schemes that result in obscuring the diversity available in the channel. Channel correlation violates the assumption of having fully independent channel coefficients in various paths in a MIMO system, which essentially results in high channel condition number\(^1\). This in turn translates to lower diversity in the BER performance curve. The same phenomenon happens when the receiver algorithm is not capable enough to exploit the full diversity. For instance, as was mentioned in Section 2.5.2, linear detectors can achieve a diversity order of $N_R - N_T + 1$.

Lattice reduction (LR) is a way to tackle these two problems. In fact, lattice-reduction has been proposed in conjunction with linear low-complexity detection schemes to improve the achieved diversity of suboptimal methods via transforming the system model into an equivalent one with a better-conditioned channel matrix [64]. It has been shown that the LR-aided detectors with simple linear receivers give performance results closer to the maximum-likelihood (ML) receiver with full diversity [65] and yet with the same complexity as linear receivers [64]. In fact, the LR technique normalizes the channel matrix to a more orthogonal one, and lowers the likelihood of noise perturbations in the detection process. The LR technique has been applied to both the linear receivers such as ZF-based lattice reduction receiver [64], [66] and MMSE-based LR receivers [67], and non-linear receivers such as V-BLAST for

\(^1\)Channel condition number is defined as the ratio of the largest eigenvalue to the smallest eigenvalue of the channel matrix.
MIMO-OFDM systems [66].

A LR-aided implementation of the K-Best algorithm has also been proposed in [68] with a novel scheme for candidate generation/pruning. However, it has three drawbacks, first, the candidate generation is not efficient, which leads to large computational complexity. Secondly, the sorting core becomes the bottleneck in the scheme due to the large number of candidates to sort. Thirdly, the proposed scheme is not applicable to infinite lattices as it requires predetermined displacements to estimate the best candidates in each level of the detection tree. In this chapter, we propose a novel lattice-reduction-aided implementation of the K-Best algorithm for the MIMO detection problem at the receiver with near-ML performance results, especially for high-SNR regimes, that alleviates all of the above problems using an efficient distributed architecture.

5.2 System Model

Consider the system model described in Chapter 2 where a MIMO system with $N_T$ transmit and $N_R$ receive antennas are under consideration (Fig. 5.1). We rewrite (2.3) here, which is the equivalent real baseband model of the Rayleigh fading channel between the transmitter and the receiver.

$$y = Hs + v,$$

Using the same terminology as in [42], for a given matrix $H$, the set $\{Hs\}$ represents a subset of the lattice $\Lambda(H)$ generated by $H$. The columns of $H$ are called basis vectors for $\Lambda(H)$, while the transmitted vector $s$ can be considered as the coordinates of a lattice point. Thus, the received signal vector $y$ is an observation of lattice point $Hs$. 

Figure 5.1: Typical detection framework.
corrupted by complex AWGN $v$. The objective of the MIMO detection method is to find the closest lattice point $\hat{s}$ for a given lattice $\Lambda(H)$, i.e., $\hat{s} = \arg \min_{s \in \Omega^{2N_T}} \| y - Hs \|^2$, where $\Omega = \{-\sqrt{M} + 1, \cdots, -1, +1, \cdots, +\sqrt{M} - 1\}$ is the set of real entries in the constellation. Note that based on the above definitions $|\Omega|^{2N_T} = |\mathcal{O}|^{N_T}$. This implies that a complex $N_R \times N_T$ MIMO system can be modeled as a real $2N_R \times 2N_T$ MIMO system. In the presence of AWGN, it can be shown that optimal maximum-likelihood (ML) MIMO detection is equivalent to searching for the closest lattice point $Hs$ to the received point $y$ in lattice $\Lambda(H)$, [42], but its complexity grows exponentially with the number of transmit antennas.

### 5.2.1 Lattice-Reduction

The undisturbed received signal $Hs$ is a point in $\Lambda(H)$, which is generated by the linear combinations of a set of linearly independent vectors $h_1, \cdots, h_{N_T}$, (the columns of $H$) with integer coefficients. The set of vectors $h_1, \cdots, h_{N_T}$ is called a basis of $\Lambda(H)$, and the $N_R \times N_T$ matrix $H = h_1, \cdots, h_{N_T}$ is called the generator matrix of $\Lambda(H)$.

The basis of the lattice is not unique. Indeed, we can obtain a new generator matrix by multiplying the old generator matrix by any $N_T \times N_T$ unimodular matrix, where a unimodular matrix is defined as an integer matrix whose inverse has also integer entries. In many applications, a basis consisting of relatively short and nearly orthogonal vectors is desirable. The procedure of finding such a basis for a lattice is called lattice basis reduction. In other words, it can be shown that matrix

$$\bar{H} = HT, \quad (5.2)$$

will generate the same lattice as $H$, i.e., $\Lambda(\bar{H}) = \Lambda(H)$, if and only if the $2N_T \times 2N_T$ matrix $T$ is unimodular, i.e., $T$ contains only integer entries with $\det(T) = \pm 1$ [67]. In fact, each column of $H$ is expressible as an integer combination of the columns of $\bar{H}$, and vice-versa. This means the entries of both $T$ and $T^{-1}$ are integers, which implies that $\det(T) = 1/\det(T)$, thus $\det(T) = \pm 1$ meaning $T$ is unimodular.

The aim of LR is to choose $T$ so as to transform a given basis $H$ into a new basis $\bar{H}$ with more orthogonal bases $\bar{H}$ [64]. The challenge of LR-aided schemes lies in the efficient implementation of LR. There are three popular LR algorithms, which are:
1. **LLL (Lenstra-Lenstra-Lovasz) Algorithm** [69]: is an effective though suboptimal LR algorithm, which iterates between a weak Gram-Schmidt decomposition and column-swapping. In fact, it consists of two steps: (i) **size reduce**, where it reduces the lengths of the basis vectors by subtracting integer components in the direction of other basis vectors. This aims to make basis vectors shorter and more orthogonal. (ii) **Basis vector swapping**, where it swaps the basis vectors if required. Both the real LLL [69], [70], [67] and complex LLL [71] have been studied extensively in the literature. In [72] it has been proven that the LLL algorithm achieves full ML diversity.

2. **Seysen’s Algorithm (SA)** [73], [74], [75]: Unlike the LLL algorithm which aims to increase the orthogonality of the channel matrix only, SA jointly optimizes the orthogonality of the channel matrix and its inverse [76], [75]. In [73], greedy and lazy methods of SA are described. Furthermore, it has been shown via numerical results that the greedy implementation of SA requires fewer basis update iterations than does the LLL method, which implies lower complexity. However, SA’s per iteration complexity is larger than that of LLL and it has not yet been proved to achieve the full diversity.

3. **Brun’s Algorithm** [77], [78], [79], [80]: Brun’s algorithm is also an iterative algorithm, which assumes that only one of the eigenvalues of the channel is too small and its value is far from the others. Because of this restriction, its applicability is limited and reveals performance loss compared to the LLL algorithm.

Based on the above discussion, we employ the LLL algorithm as the underlying engine for the LR algorithm implementation. Regardless of the employed LR algorithm, the final LR-reduced matrix consists of roughly orthogonal columns, in roughly increasing order of their Euclidean norms. Consequently, an LR-reduced matrix exhibits a rather constant and small condition number, regardless of the condition of the original matrix. This makes LR-aided detectors robust to spatial channel correlations\(^2\) [67]. Note that the proposed joint lattice reduction and the K-Best algorithm in this chapter is independent of the LR algorithm used. So it can be extended and applied to any setting if required.

\(^2\)This is true if \(\rho \neq 1\).
Using the transformation in (5.2), equation (5.1) can be rewritten as

\[ y = Hs + v = \left( HT \right) \begin{pmatrix} T^{-1} \end{pmatrix} s + v = \bar{H}X + v = \bar{Q}\bar{R}X + v, \quad (5.3) \]

where \( \bar{Q}\bar{R} \) is the QR-decomposition of \( \bar{H} \) and \( X = T^{-1}s \). Note that \( Hs \), and \( \bar{H}X \) describe the same point in the lattice but the LR-reduced matrix \( \bar{H} \) is usually much better conditioned than the original channel matrix \( H \). Therefore, the detector finds \( X = T^{-1}s \) in the lattice-reduced constellation and recovers the original symbol via \( s = TX \). Such detection has better performance due to a better-conditioned channel \( \bar{H} \). The system framework before and after the introduction of the LR block are shown in Fig. 5.1 and Fig. 5.2, respectively. In fact, the first part in Fig. 5.2 is just a different interpretation of the effect of the channel on the transmitted signal and does not change the system model at the transmitter. In other words, the only modification is on the receiver side.

In order to use the LR technique, the original points in the constellation are required to consist of consecutive integers in \( \mathbb{Z} \) (the set of integers). However, the real and imaginary parts of QAM constellations does not consist of consecutive integers, thus, it is necessary to shift and scale the original constellation. Normally a displacement vector \( d_{2N_T \times 1} = [1, \cdots, 1]^T \) is used to shift and scale the received signal \( y \) as follows.

\[ \bar{y} = (y + Hd)/2 = \bar{Q}\bar{R}(X + d)/2 + v/2 = \bar{Q}\bar{R}\bar{X} + v/2, \quad (5.4) \]

where \( \bar{X} = (X + d)/2 \). Therefore, the shifted and scaled signal, \( \bar{y} \), is hereafter used for detection purposes and the detected vector, i.e., \( \bar{X} \), is transformed to the original format via \( X = 2\bar{X} - d \).
5.3 Problem Definition (LR-Aided K-Best)

The application of the LR technique to linear receivers as well as the V-BLAST/SIC receiver is straightforward as in all these techniques at each level, only the best candidate of the current level is selected. Thus once the LR technique is applied, the LR-reduced version of (3.1), i.e., \( \bar{z} = \bar{Q}^H \bar{y} \), is derived and the rest is the normal lattice reduction problem using linear receivers. However, the application of the LR technique to the K-Best detector is not straightforward. In other words, the LR core cannot be implemented separately from the K-Best core. This is because at each level of the tree, the \( K \) best nodes of the previous level need to be expanded to the new offsprings. In the normal K-Best implementation for \( M \)-QAM systems, there are \( \sqrt{M} \) children per parent to be considered. Thus the number of children per parent and their values are known a priori. However, once the LR technique is applied, since the bases of the LR-reduced channel \( \bar{H} \) are different, the number of children, and their values are not known in advance. Therefore, the joint implementation of the LR technique and the K-Best algorithm is technically challenging.

This fact is shown in Fig. 5.3, where part of an infinite lattice with the boundary control for a \( 2 \times 2 \) MIMO system with the original bases \( h_1 \), and \( h_2 \) is depicted\(^3\). The black crosses show the desired points of the lattice, whereas the gray crosses are part of an infinite lattice. In a \( 2 \times 2 \) scheme, there are two levels in the tree. Fig. 5.3(a) shows the original bases, which are not orthogonal. In this case the number of possible children and their values are fixed and predetermined, i.e., there are 4 children per level with the values 0, 1, 2 and 3. After applying the LR technique, the bases are more orthogonal resulting in a better performance. However, the number of children and their values are not predetermined. As shown in Fig. 5.3(b), there are 10 children at the first level with values 0-9 with 1 or 2 children per parent in the second level with different values. These values correspond to \( \bar{X} \). Note that the number of final children is the same in both cases and they both refer to the same lattice. As \( H \) varies over time so does \( \bar{H} \), thus the value/number of children per parent is not fixed over time.

The fact that the number/values of the nodes are not predetermined in advance, makes it hard to implement the joint LR/K-Best algorithm. This problem becomes even more complicated as the number of antennas/levels of the tree increases. One

\(^3h_1 \) and \( h_2 \) represent the first and second columns of \( H \), respectively.
way to tackle this problem is to calculate all the possible cases to determine the number and value of the nodes in advance using $T^{-1}\bar{Z}2^{N_T}$, where $\bar{Z}$ represents all the possible values of the children before applying LR, e.g., $\bar{Z} = \{0, 1, 2, 3\}$ in Fig. 5.3(a). Obviously this solution is not feasible and has the same complexity as ML-detection.

The other way is to predict a set of predetermined displacements to generate/estimate the value of the nodes as was suggested in [68], which can be described for level $l$ as follows.

1. Take one of the $K$ best survivors (parent nodes) of the previous level, i.e., level $l + 1$.

2. Calculate an estimate of the $z_l$ based on the value of $\hat{z}_l = L_l(s^{(l)})/r_{ll}$.

3. Grow an $N$-point neighborhood around $\hat{z}_l$ as $[Q(\hat{z}_l + \delta_1, \cdots + Q(\hat{z}_l), \delta_N)]$, where $N > K$, $Q(\cdot)$ rounds real and imaginary parts separately, and $[\delta_1, \cdots, \delta_N]$ is a predetermined set of displacements generating a neighborhood, with each element taking on integer real and imaginary parts.

4. Calculated the PED values of all the $N$ offsprings of the given survivor.

5. Steps 1-4 are repeated for all the $K$ parent nodes in level $l$, resulting in $KN$ symbol vectors, out of which we choose $K$ that have the smallest PED.

Figure 5.3: The possible integer values of (a) $s$ based on $H$, (b) $X$ based on the new bases of $H$. 
The drawback of this solution is as follows. First, in each level of the tree, $KN$ children need to be generated, where $N > K$ denotes the number of displacements. Considering that $KN$ becomes very large for high-dimensional problems, the expansion core becomes time-consuming. Secondly, the sorting core is required to sort $KN$ nodes at each level and select the best $K$ candidates, which is itself computationally intensive. Third, the displacements cannot be easily determined and should be modified as $H$ varies and finally it has some performance loss as it differs from the strict K-Best algorithm due to the approximation made as only a selected set of nodes are visited.

\section*{5.4 Proposed Scheme}

As mentioned any scheme for the K-Best implementation that requires all the children of all parents to be calculated and then be sorted to find the next $K$ best candidates, would fail to be integrated with the LR technique with a reasonably low complexity. Therefore, the current proposed methods for the K-Best implementation in the literature, [3], [2], are not suitable for this purpose. The target of this chapter is to propose a scheme which (i) performs the sorting in a distributed way, (ii) results in a true K-Best performance result, (iii) does not require finding all the children, and (iv) does not require to know the displacements in advance as is the case in [68].

\subsection*{5.4.1 Sorting Scheme}

Consider a $2N_R \times 2N_T$ real-domain MIMO system with channel matrix $H$ and its LR-reduced version $\tilde{H}$. The system can be thought of as a detection problem in a tree with $2N_T$ levels and $\sqrt{M}$ children per parent. The main body of the K-Best detector in this case is the same as the one explained in detail in Chapter 3. It is briefly described here again for ease of reference and understanding of the joint scheme with the lattice reduction. Let’s consider level $l$ of the tree and assume that the set of K-Best nodes in level $l + 1$ are already determined and denoted by $\mathcal{K}_{l+1}$. Each node in level $l + 1$ has $\sqrt{M}$ possible children, so there are $K \sqrt{M}$ possible children in level $l$. Hereafter, we call the nodes in level $l + 1$ the parent nodes and their offsprings in level $l$ as their children. Let $c_{km}^l$ denote the $m$-th child of the $k$-th parent in level $l$, and $d_{km}^l$ represent its corresponding accumulated PED. Note that the parent of $c_{km}^l$
is the $k$-th node in level $l + 1$. Therefore, the input to the K-Best algorithm at level $l$ is $K_{l+1}$, and its output is $K_l$.

Let’s assume that each parent node knows its first child without visiting all of its children and let $C_l$ represent the set consisting of all the current best children of all parents, and $D_l$ represent their corresponding PEDs. It is easy to see that the child with the lowest PED in $C_l$ is definitely one of the K-Best candidates in $K_l$. This child can be mathematically represented as $c_{km}^l$ where $\bar{k}\bar{m} = \arg\min_{k,m} (d_{km}^l \in D_l)$.

Thus this child should be added to $K_l$, i.e., $K_l \leftarrow K_l + \{c_{km}^l\}$. To find the next best child in the K-Best list, $c_{km}^l$ and its corresponding PED ($d_{km}^l$) are removed from $C_l$, and $D_l$, respectively. Following this removal, the next best sibling of this child is added to $C_l$. Taking the same approach, the child in $C_l$ with the lowest PED is definitely among the final K-Best list. That child should be added to $K_l$, be removed from $C_l$ and be replaced by its next best sibling. If we repeat this $K$ times, all the $K$ best children in level $l$ are determined.

The pseudo-code of the proposed scheme including the LR technique and the distributed expansion and sorting scheme is shown in Table 5.1. After the shift/scale (Step I.1), the LR technique is applied to $H$ to derive $\tilde{H}$ (Step II.1). QR-decomposition is performed and the nulling matrix $\tilde{z}$ is obtained (Step II.1, and Step II.2). Starting from the last row of $\tilde{z}$, in the initialization step, the $K$ best nodes of the 8-th level are selected creating $K_{2N_T}$ (Step III.1, and III.2). For each of the elements in $K_{2N_T}$, the first child and its corresponding weight are found (Step IV.2, and IV.3). The child with the lowest weight is selected as one of the $K$ best nodes (Step IV.4.1 and IV.4.2) and is replaced by its next best sibling (Step IV.4.3-IV.4.5). This process is repeated $K$ times to find all the $K$ best nodes of level $l$. Among the $K$ best nodes of level 1, the one that has the lowest PED ($\bar{k}\bar{m} = \arg\min_{k,m} \{d_{km}^1 \in D_1\}$) with all of its ancestors all the way to the level $2N_T$ are announced to be the output of the hard detection problem (Step V.1 and V.2). The detected signal is transformed back to the before-LR format (Step V.3). After applying the boundary control, it is finally scaled/shifted back to the QAM constellation format (Step V.5).
Table 5.1: The Proposed Scheme for LR-aided K-Best Algorithm.

I. Shift & Scale
1) \( \bar{y} \leftarrow (y + Hd)/2. \)

II. Lattice-Reduction & QR
1) \( H \leftarrow LR(H). \)
2) \( \bar{H} = QR. \)
3) \( \bar{z} = Q^T \bar{y} = \bar{R}\bar{X} + \bar{v}/2. \)

III. Initialization
1) Start from the last row of \( \bar{z} \), i.e., row \( 2N_T \).
   Find the K-Best children of level \( 2N_T \), \( C_{2N_T} \).
2) Set \( K_{2N_T} = C_{2N_T} \).

IV. Expansion & Sort
For \( l = 2N_T - 1 : -1 : 1 \)
1) \( K_l = \emptyset. \)
2) Find \( C_l \), the set of the first child of each node in \( K_l \).
3) Calculate \( D_l \), the weights of the elements in \( C_l \).
   For \( k = 1 : K \)
   4.1) \( \bar{k}\bar{m} = \arg\min_{k,m} \{d_{k,m}^l \in D_l\} \)
   4.2) \( K_l \leftarrow K_l + \{c_{k\bar{m}}^l\} \)
   4.3) \( C_l \leftarrow C_l - \{c_{k\bar{m}}^l\} \), and \( D_l \leftarrow D_l - \{d_{k\bar{m}}^l\} \).
   4.4) Find the next best child of \( k \)-th parent, \( c_{k\bar{m}}^l \).
   4.5) \( C_l \leftarrow C_l + \{c_{k\bar{m}}^l\} \), and \( D_l \leftarrow D_l + \{d_{k\bar{m}}^l\} \).
End

V. Detection
1) \( \bar{k}\bar{m} = \arg\min_{k,m} \{d_{k,m}^1 \in D_1\} \)
2) Announce \( c_{k\bar{m}}^1 \) with all of its parents up to the last level as the hard output \( \bar{X} \).
3) \( \bar{s} = TX. \)
4) \( \bar{s} \leftarrow BoundaryControl (\bar{s}). \)
5) \( s = \bar{s} * 2 - d. \)
Table 5.2: First/Next Child Selection Procedure.

A) First Child
A.1) \( s_{0}^{[0]} = L_{i}(s_{i-1}^{[0]})/r_{il} \).
A.2) \( s_{i}^{[1]} \leftarrow \lfloor s_{i}^{[0]} \rfloor, n_{i}^{2} \leftarrow 2. \)

B) Next (k-th) Child
B.1) SignBit \( = \text{Sign}(s_{i}^{[k-1]} - s_{i}^{[k-2]}). \)
B.2) \( s_{i}^{[k]} \leftarrow s_{i}^{[k-1]} + n_{i}^{2} \times \text{SignBit}. \)
B.3) \( n_{i}^{2} = n_{i}^{2} + 2. \)

5.4.2 On-demand Expansion Scheme

In fact, the on-demand expansion scheme used in the joint LR and K-Best problem is the same as the scheme used in the real-domain implementation of the distributed K-Best algorithm proposed in Chapter 3.

Note that there is an important difference between the first-child next-child calculation in the on-demand implementation of the K-Best algorithm, Table 3.4, with that of in the joint implementation of the LR and K-best algorithm, Table 5.1. In fact, in the on-demand version of the K-Best algorithm, the next child is always determined under the boundary control assumption. For instance in 64-QAM, since no child grater than 7 exists, the expansion limit should be limited. This has been done through the algorithm in Table 3.3 by controlling the values of SignBit and \( n_{i}^{2}. \) Since the children have been limited in the expansion stage, there is no need to apply the boundary control at the detection stage. However, in the case of joint LR and K-Best implementation, there is no boundary control while expanding the children (see Table 5.2). This is because of the fact that the children have no limitation after the application of the LR to the channel matrix as explained in Section 5.3. However, at the detection stage, there might be some children detected outside the constellation allowed boundary. Therefore, the boundary control is needed at the detection stage in the joint implementation as shown in Table 5.1.
Table 5.3: Complexity of the LR-aided K-Best Scheme for a 4 × 4 MIMO System.

<table>
<thead>
<tr>
<th>Case</th>
<th>expansion (# of nodes)</th>
<th>sort (cycles)</th>
<th>Case</th>
<th>expansion (# of nodes)</th>
<th>sort (cycles)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This work [68] saving</td>
<td></td>
<td>This work [68] saving</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16-QAM</td>
<td>67</td>
<td>120</td>
<td>44%</td>
<td>35</td>
<td>110</td>
</tr>
<tr>
<td>64-QAM</td>
<td>141</td>
<td>384</td>
<td>63%</td>
<td>70</td>
<td>372</td>
</tr>
<tr>
<td>256-QAM</td>
<td>218</td>
<td>790</td>
<td>72%</td>
<td>120</td>
<td>773</td>
</tr>
</tbody>
</table>

In brief, the proposed sorting scheme takes exactly $K$ clock cycles. Moreover, in the expansion scheme (Section 5.4.2) the best children of each parent are generated one-by-one and on-demand without visiting all the other points. The complexity of both of these cores is independent of the constellation order ($M$). Thus the above implementation of the K-Best algorithm is a promising method for high-order constellation schemes such as 64-QAM and 256-QAM. In fact, due to the on-demand nature of our proposed expansion scheme there is no need to calculate the displacements in advance, which essentially results in a significant saving in computation.

5.5 Complexity Analysis

Focussing on the detection part (Steps III and IV in Table 5.1) excluding the QR-decomposition, channel preprocessing, and the LR technique as they are the common blocks between all the schemes, we compare the complexity of our proposed scheme with the latest proposed scheme for joint LR and K-Best in [68]. The result of the complexity analysis is shown in Table 5.3 for a 4 × 4 MIMO system with different constellation schemes. The assumed values of $K$ in Table 5.3 are 5, 10 and 15 for 16-QAM, 64-QAM and 256-QAM, respectively. Using our proposed distributed scheme for joint LR and K-Best implementation, a significant complexity reduction is achieved. For instance, in 16-QAM, the number of expanded children in the search is 67 versus 120 children in [68]. This saving is as a result of the proposed on-demand expansion scheme. In fact, in our scheme there are $2K-1=9$ children in each level (except the 8-th level as the 8-th level has just four children). Thus, the total number of children is $9 \times 7 + 4 = 67$ (8 levels)$^4$. In terms of the number of sorting cycles, in

$^4$Note in [68] the set of predetermined displacements are calculated in a preprocessing stage, however: (i) it will require additional hardware and (ii) the hardware sorting burden of $KN$ nodes is significant.
each level (except the 8-th level) $K = 5$ cycles is required to do the sorting. Thus the total number of cycles for sorting is $5 \times 7 = 35$. However, in [68] there are $K \times N$ children to be calculated in each level. This complexity saving becomes even more noticeable in higher order modulation schemes resulting in a saving of 68%, 81%, and 84%, for 16-QAM, 64-QAM and 256-QAM, respectively.

### 5.6 Simulations

In the following, the simulation results for a single-carrier $4 \times 4$ MIMO system is presented for both 16-QAM and 64-QAM schemes. Fig. 5.4 shows the BER performance of a 16-QAM scheme for ML, K-Best, and joint K-Best/LR for different values of $K$.

The SNR value is defined as the signal-to-noise ratio per transmitted symbol. It shows that for a fixed value of $K = 5$, the LR technique can greatly improve the performance in high-SNR regimes. The improvement achieved by the LR technique might motivate us to decrease the value of $K$ to have the same performance with a lower complexity. The result for $K = 3$, and $K = 2$ with K-Best/LR is also shown Fig. 5.4. As can be seen, the K-Best/LR with $K = 2$ has quite the same performance as K-Best with $K = 5$.

Fig. 5.5 shows the performance of the K-Best/LR technique for $4 \times 4$ 64-QAM
system. It also confirms that the joint K-Best/LR outperforms the K-Best algorithm for a fixed value of $K$. Note that the value of $K$ scales sub-linearly from 5 in 16-QAM to 10 in 64-QAM. In fact, it can be shown that the performance improvement introduced by the LR techniques is more pronounced for higher constellation orders, which is shown in Fig. 5.6, where the performance of the LR-aided K-Best scheme is closer to the ML result than that of for 64-QAM and 16-QAM schemes.

One important conclusion from the above simulation results is that the LR block can be implemented as a side-block to the K-Best detector. When operating in low-SNR regimes, the LR core can be inactive to save power as it does not contribute much in improving the BER performance. However, when operating in the high-SNR regimes, activating the LR block contributes significantly to improve the detection diversity and BER performance.

### 5.6.1 The Effect of Antenna Correlation

Fig. 5.7 and Fig. 5.8 show the performance of the K-Best/LR technique for $4 \times 4$ 64-QAM system for different correlation parameter $\rho$ introduced in Section 2.6. Fig. 5.7 shows the performance result for $\rho = 0.1$ whereas Fig. 5.8 is the BER curve for $\rho = 0.4$. It is implied from these figures that all the detection schemes result in performance degradation. However, the ML detector and the joint K-Best and LR
detector reveal more robustness to the antenna correlation than K-Best detector. For instance, at BER=$10^{-4}$, the BER loss of the joint K-Best and LR with respect to the ML detector is almost the same for both correlation scenarios whereas the BER loss for the K-Best detector increases from 5.2 dB in $\rho = 0.1$ to 8 dB in $\rho = 0.4$.

### 5.7 Summary

An efficient LR-aided implementation of the K-Best algorithm has been proposed. The proposed scheme does not require any a priori knowledge of the candidate displacement, is scalable in terms of the number of transmit antennas and its complexity grows sub-linearly with the constellation order. The main advantage of the joint implementation is a higher diversity and a better performance result at the receiver.
Figure 5.7: LR-aided K-Best, K-Best and ML for $4 \times 4$ 64-QAM, with correlation ($\rho = 0.1$).

Figure 5.8: LR-aided K-Best, K-Best and ML for $4 \times 4$ 64-QAM, with correlation ($\rho = 0.4$).
6 Compensation of the Nonlinearity of Power Amplifiers Using Sequential Monte Carlo

6.1 Introduction

In the context of detection schemes, there are PHY-level system issues that need to be addressed in order to achieve an efficient high-performance realization. These issues are:

- High-order constellation schemes are critical to achieve high bandwidth efficiency and for achieving the bit rates required for high data-rate multimedia systems. Therefore, developing efficient detectors for MIMO systems that support high-order constellations is of significant importance.

- Developing efficient signal processing schemes to increase the efficiency of the power amplifiers or to compensate for their nonlinearity and especially in the context of supporting high-order constellations. This is of great importance as the power amplifier is the most power consuming element in the overall transceiver. MIMO systems with their multiple transmit antennas and associated power amplifiers accentuate this fact with respect to the overall system power budget. This is also a critical issue for OFDM systems for their well-known high peak-to-average-power-ratio (PAPR) problem.

The first part of this thesis addressed the first challenge through the introduction of a novel detection scheme for MIMO systems. This chapter addresses the second issue and is aimed to develop a novel framework for compensating the amplifier nonlinearities especially for high-order constellation schemes. This in fact unifies the results of this chapter with the previous chapters in the context of high-order constellation schemes. Both issues are addressed in the baseband receiver. The main scope of the discussion in this chapter relates to SISO systems but the extension of the proposed scheme to MIMO systems is straightforward. This is because of the fact that after
application of the ZF, MMSE filter the MIMO system can be transformed to multiple SISO systems. In the case of non-linear receivers, after the application of the QR-decomposition starting from the last level, each symbol is detected independently while the interference from the other antennas is removed, thus can be treated as a SISO system. Therefore, in the following we only focus on the SISO systems.

In fact, high power amplifier (HPA) linearization methods have been of great importance in recent years due to the demand for the high power efficiency, good linearity, and low cost. One way to guarantee the required HPA linearity is by backing off the amplifier, i.e., the operating point of the amplifier is backed off from the saturation region where it exhibits nonlinear characteristics such as amplitude and phase distortion [81]. The main drawback of this approach is that the efficiency of the amplifier is drastically reduced since the amplifier is more efficient at or near the saturation level. This fact is especially important in the case of wireless systems where power is a costly and often a limited resource, and because the HPA is the most power consuming component in the overall transceiver power budget. Therefore, a main concern in HPAs is to operate closer to the saturation region while mitigating the associated amplitude and phase distortion.

There are a few approaches dealing with the equalization of general nonlinear power amplifiers [82], [83]. These methods assume that the transfer function of the nonlinear system can be modeled as a Volterra filter [84]. However, these approaches either: (i) do not converge, meaning that the Volterra filter is unstable and that the system is not invertible with some modelization, or (ii) needs many iterations to converge [85].

Linearization techniques have also been extensively studied in the literature for single-carrier transmission schemes employing multilevel amplitude and phase modulation. The most popular technique is to insert a predistorter (PD) between the transmission shaping filter and the HPA such that the input-output characteristic of the predistorter compensates for the intrinsic nonlinearity of the HPA using polynomials [86], [83], look-up-tables where a complete input-output mapping of the PD is stored in memory [87], [88], [89], or rational functions [90] to model the amplifier. These methods can be implemented both in analog and digital domains [88], [91].

Although applying predistortion to the signal before amplification is an effective approach to compensate for nonlinearities, it provides no advantage in low input backoff (IBO) regimes [92]. Additionally, as is also shown in this chapter, even for high-IBO regimes, the predistorter has a poor performance for high-order QAM constellation
6 Compensation of the Nonlinearity of Power Amplifiers Using Sequential Monte Carlo

schemes.

On the other hand, particle filtering, known also as the Sequential Monte Carlo (SMC) method, has been shown to offer promising performance advantages to different applications in digital communication receivers, especially when the system is non-linear or non-Gaussian [93]. In this work, an SMC framework is proposed as an effective vehicle to mitigate the nonlinearity intrinsic to Solid-State Power Amplifiers (SSPAs). The key idea is to draw a few samples from the desired posterior distribution and perform the detection based on the drawn samples. It is shown that SMC outperforms the predistorter especially for high-order modulation schemes or in low IBO regimes. The proposed method has two main advantages compared to the other alternatives. First, it results in much higher power efficiency for the power amplifier, and secondly, it fits into any wireless standard without modification of the transmitter. This is because the SMC method implements all of the signal processing at the receiver, as opposed to the predistorter, where the transmitter needs to be modified.

From the algorithmic point of view, the repetitive nature of SMC methods is very attractive for parallel and pipelined VLSI implementation. The SMC receiver consists of the SMC core, weight calculator, sampler, and resampler. Efficient implementation of SMC requires efficient architectures of all the sub-blocks of the SMC. Since resampling is the sequential step of the algorithm, which ultimately limits high speed implementations, special care should be taken in this regard. In fact, the proposed architecture for the resampling can directly affect the performance and the speed of the whole SMC receiver.

The general problems with the parallel implementation of resampling using a collection of processing elements (PEs) are related to the lack of concurrency among iterations, and non-deterministic communications among PEs. These problems motivate us to develop a parallel architecture of the resampling step of the algorithm. Some methods have been proposed in the literature trying to tackle this issue [94]. However, these solutions have some drawbacks including having non-deterministic routing scheme, being memory intensive, being non-scalable for high levels of parallelism, or lack of achievable performance, which ultimately limits their applicability.

Therefore, in Appendix B, a novel resampling scheme is proposed, which has close-to-minimum execution time, while resulting in the same achievable performance as sequential implementations. We show that our scheme requires less memory compared to the alternatives in the literature and can be pipelined with the sampling step. The
nice feature of the proposed scheme is its scalability in the level of parallelism.

6.2 System Model

Fig. 6.1 shows the block diagram of a typical transmit-receive system using an M-QAM modulation scheme in a nonfading AWGN channel. The system is meant to be simple in order to highlight the benefits of the SMC receiver in the presence of nonlinearities. The transmitted signal, \( x(t) \), is first mapped to the proper constellation points, based on the M-QAM modulation scheme. The modulated signal, \( s(t) \), is then amplified by the SSPA. The amplified signal, \( y(t) \), is transmitted through the AWGN channel. Thus, the received signal, \( r(t) \), is a noisy version of the transmitted signal. The SMC receiver performs the detection, which results in \( \hat{s}(t) \), and finally uses the demapper to yield the detected data \( \hat{x}(t) \).

6.2.1 HPA Model

There are two types of HPAs used in wireless communication systems, namely the traveling wave tube amplifier (TWTA) [95] and the SSPA [96]. The TWTA is normally used in wireless satellite communications whereas the SSPA, because of its small size, is used in many other applications such as mobile communication systems. In general, the output of HPAs is prone to amplitude distortion, i.e., amplitude-modulation/amplitude-modulation (AM/AM), and/or phase distortion, i.e., amplitude-modulation/phase-modulation (AM/PM). These distortions are both functions of the amplitude of the input signal. Since in this work, the detection problem in mobile communication systems is considered, hereafter, we only discuss results for SSPAs. In [97], Monte Carlo methods are proposed for blind equalization.
of a satellite communication channel in the presence of a TWTA amplifier. Besides
a different application and power amplifier characteristic, [97] does not address the
problem of constellation scaling and low-IBO trade-offs, which are the main focus
of this Chapter. Furthermore, the performance results of SMC versus the predistorter
and the analysis of their effective applicable regions have not been covered. Moreover,
the performance of the SMC receiver in multi-carrier systems is not addressed, which
is mainly because of the considered application in satellite communications.

In this work, the system encompasses an HPA, which may exhibit amplitude clip-
ing and phase distortion. A widely accepted HPA model is a nonlinear memoryless
transformation between the complex envelopes of the input and output signals [95].
If the complex envelope of the input signal to the SSPA, $s(t)$, is written as
\begin{equation}
    s(t) = \rho(t)\exp[j\varphi(t)],
\end{equation}
where $\rho(t)$, and $\varphi(t)$ denote the signal amplitude and phase, respectively, and the
SSPA characteristic function is denoted by $G(\cdot)$, then the complex envelope of the
output signal can be represented by
\begin{equation}
    y(t) = G[s(t)] = G[\rho(t)]\exp\left\{j\left[\varphi(t) + \Phi[\rho(t)]\right]\right\},
\end{equation}
where $G[\rho(t)]$, and $\Phi[\rho(t)]$ represent the AM/AM and AM/PM conversion character-
istics of the nonlinear amplifier, respectively, which can be expressed as [98], [99],
\begin{align}
    G[\rho(t)] &= \nu \rho(t) \left\{\frac{\rho(t)^{2p}}{1 + \left(\frac{\rho(t)}{A_o}\right)^{2p}}\right\}^{1/2p}, \\
    \Phi[\rho(t)] &= \alpha_\phi \left(\frac{\nu \rho(t)}{A_o}\right)^4,
\end{align}
where, $\nu$ is the small-signal gain, $A_o$ is the amplifier output saturation voltage, $A_s$ is
the amplifier input saturation voltage, $p$ is a parameter that controls the smoothness
of the transition from the linear region to the saturation region, and $\alpha_\phi$ is a scaling
parameter, which is typically set to zero because the amplitude distortion is considered
to be dominant for a SSPA [98]. However, in order to consider the general case of
the SMC receiver, we set $\alpha_\phi = 0.1$ in our simulations. Hereafter, we use $A_o = 1$,
$A_s = 2.65$, $p = 2$, and $\nu = 1$. Note that, the larger the value of $p$, the closer the
6 Compensation of the Nonlinearity of Power Amplifiers Using Sequential Monte Carlo

AM/AM curve is to the ideal soft limiter, i.e.,

\[ G[\rho(t)] = \begin{cases} 
  \nu \rho(t), & 0 \leq \rho(t) \leq A_s \\
  A_o, & \rho(t) \geq A_s 
\end{cases} \] \hspace{1cm} (6.5)

The operating point of the power amplifier is usually identified by the backoff. Two common backoff parameters are output backoff (OBO), and input backoff (IBO), which are defined as follows:

\[
\text{OBO} = 10 \log_{10} \frac{P_{O,\text{sat}}}{\bar{P}_O} \hspace{1cm} (6.6)
\]

\[
\text{IBO} = 10 \log_{10} \frac{P_{I,\text{sat}}}{\bar{P}_I} \hspace{1cm} (6.7)
\]

where \( \bar{P}_O \), is the mean power of the transmitted signal, \( P_{O,\text{sat}} \), is the maximum output power, \( P_{I,\text{sat}} \), is the input power corresponding to the maximum output power, and finally \( \bar{P}_I \), is the mean power of the signal at the input of the HPA. The effects of the nonlinearities can be reduced by working with high backoff values, which corresponds to moving the operating point of the amplifier to the linear region. Unfortunately, this leads to a loss in power efficiency of the HPA, which is defined as the ratio of the average output power to the average input power. Note that the input power includes the HPA DC power.

6.2.2 Predistorter

Predistortion is provided by a signal processing block placed before the SSPA and is defined by the generic complex function, \( F(\cdot) \), which has to be designed such that the following relationship applies

\[ G\left\{ F[s(t)] \right\} = s(t). \hspace{1cm} (6.8) \]

This means that the input-output characteristic of the predistorter is obtained by inversion, i.e.,

\[ F[s(t)] = G^{-1}[\rho(t)] \exp\left\{ j \left[ \varphi(t) - \Phi[\rho(t)] \right] \right\}, \hspace{1cm} (6.9) \]
where in the case of the SSPA,

\[
G^{-1}[\rho(t)] = \begin{cases} 
\rho(t) & 0 \leq \rho(t) < A_o, \\
\sqrt{\frac{2p}{1 - \left[\frac{\rho(t)}{A_o}\right]^{2p}}} & \rho(t) \geq A_o,
\end{cases}
\]  
(6.10)

This is because there is a discontinuity at \( \rho(t) = A_o \). Thus the amplitude is limited. Fig. 6.2 shows the transfer function of the SSPA, the predistorter, and the SSPA+predistorter. The values of \( A_o \), and \( A_s \) are also shown in the figure.

The block diagram of a typical transmit/receive system using the predistorter is shown in Fig. 6.3. After mapping to the constellation points, the transmitted symbols are predistorted first by the function \( F(\cdot) \) in (6.10), and then amplified by the SSPA at the transmitter front end. On the receiver side, a demapper is used to detect the transmitted bit stream. The poor performance of the predistorter in both low-IBO regimes as well as in high-order constellation schemes, is the primary motivation in
this chapter to develop the SMC receiver.

### 6.3 The SMC Receiver

The goal of this section is to apply SMC/particle filtering to the blind detection problem at the receiver in the presence of a nonlinear solid-state power amplifier at the transmitter. In other words, the particle filter at the receiver compensates for the nonlinearities resulting from the SSPA. This is a promising approach as it does not require signal processing at the transmitter and unlike the predistorter, doesn’t need to find the inverse function, which might not be straightforward to realize. Generally the advantages of the SMC compared to the predistorter are as follows:

- The SMC method is designed to perform reasonably well in the low-IBO regimes where the predistorter fails to compensate.
- The SMC method has a soft nature as opposed to the hard-detection nature of the predistorter, and thus does not exhibit an error floor.
- The performance of the SMC method is more robust to high-order QAM constellation schemes.

#### 6.3.1 SMC Methodology

In SMC methods, the distributions are approximated by discrete random measures defined by particles and their associated weights. Let $\chi_t = \{x^{(i)}_t, \omega^{(i)}_t\}_{i=1}^N$, as the discrete random measure, denotes the sample-weight set representing the distribution of interest, $p(x_0:t|y_{1:t})$, where $\{y_{1:t}\}$ represents the set of observations. Therefore,
6 Compensation of the Nonlinearity of Power Amplifiers Using Sequential Monte Carlo

\[ p(x_{0:t}|y_{1:t}) \approx \sum_{i=1}^{N} \omega^{(i)} \delta(x_{0:t} - x_{0:t}^{(i)}), \quad (6.11) \]

where \( \delta(\cdot) \) is the Dirac delta function. Using this approximation, computations of expectations, which involve complicated integrations, are simplified to summations. For example:

\[
E_{p(x_{0:t}|y_{1:t})}(g(x_{0:t})) = \int g(x_{0:t})p(x_{0:t}|y_{1:t})dx_{0:t}, \quad (6.12)
\]

is approximated by:

\[
E_{p(x_{0:t}|y_{1:t})}(g(x_{0:t})) \approx \sum_{i=1}^{N} \omega^{(i)} g(x_{0:t}^{(i)}). \quad (6.13)
\]

There are some fundamental issues regarding the above approximation. The first point is the sampling method, which determines the way we draw samples from the distribution of interest. In some applications, drawing from the distribution of interest is intractable, while drawing the samples from another distribution, called proposal distribution, is feasible \[100\]. In these cases, the proper selection of proposal distribution is central. The second point is regarding a major problem with particle filtering, which is the fact that the discrete random measure degenerates quickly. To reduce this effect, resampling and variance reduction techniques are employed, which are other important issues in particle filtering. Finally, diversity adding methods are used to reduce the negative effects of resampling and variance reduction. Resampling is the sequential part of the SMC method, which will be covered in detail in Appendix B.

6.3.2 Application of SMC to SSPA

As mentioned, the idea behind the SMC is to represent the a posteriori distribution function by a set of samples and their associated weights (see [93] for details). The received signal, \( r(t) \), is assumed to be sampled at symbol rate \( 1/T \). The problem is formulated by estimating the transmitted signal \( x(t) \) based on the received signal knowing the SSPA generic model, and the received noise characteristic. The Bayesian
approach to formulate this problem utilizes the posterior distribution function as

$$P[x(t)|r(t), \sigma^2, \nu, A_o, p]$$

(6.14)

where $\sigma^2$ is the variance of the Gaussian noise, $n(t)$, at the receiver. Using the Bayesian concept, the posteriori probability density function is proportional to

$$P[r(t)|x(t), \sigma^2, \nu, A_o, p]$$

(6.15)

where $E = \{\nu, A_o, p\}$, and (6.15) follows because the transmitted signals are independent and identically distributed (i.i.d), and are statistically independent from the SSPA parameters. There are two main scenarios within which the above problem can be addressed. First, we assume all the parameters of the amplifier, $E$, are known whereas in the second scenario all these parameters or a subset of them, are assumed to be unknown to the receiver. In this chapter, we address both of these cases and compare their performance results.

### 6.3.3 Known Parameters

Assuming that the a priori distribution of $x(t)$ to be uniform, i.e., the constellation points are equiprobable, based on the amplified signal, $y(t)$, the likelihood probability in (6.15) can be written as

$$\propto \int P[r(t), y(t)|x(t), \sigma^2, E] \, dy(t)$$

$$= \int P[r(t)|y(t), x(t), \sigma^2, E] P[y(t)|x(t), \sigma^2, E] \, dy(t)$$

$$= \int P[r(t)|y(t), s(t), \sigma^2, E] P[y(t)|s(t), \sigma^2, E] \, dy(t)$$

$$\propto \int \exp\left\{\frac{[r(t) - y(t)]^2}{2\sigma^2}\right\} \delta\left\{y(t) - G[s(t)]\right\} \, dy(t)$$

(6.16)

where $s(t)$ was interchangeably used instead of $x(t)$ as it is the mapped version of the transmitted signal according to the constellation scheme.

Equation (6.16) indicates that samples from the constellation scheme, $\{\hat{s}(j)(t)\}_{j=1}^M$,
can be used to maximize the likelihood function in (6.14). In other words, at the transmitter, we simply send the signal without any predistortion and at the receiver, we sample from the points in the M-QAM constellation and find the one that maximizes the likelihood problem in (6.15). This task is done by a block called the SMC core, that forms the heart of the SMC receiver, and more generally is the fundamental block needed to implement adaptive schemes where no information about the parameters of the SSPA is available. Using the SMC core, we develop a SMC technique in order to estimate unknown parameters of the amplifier. It is also shown that using our proposed adaptive scheme, the SMC can efficiently compensate for variations in the SSPA parameters. These variations in the SSPA can arise because of manufacturing tolerances, environmental effects such as temperature or aging, and sometimes base-station-to-base-station variation, which occur when the receiver moves from one transmitter to the other.

### 6.3.4 Unknown Parameters (Adaptive scheme without memory)

The parameters of the amplifier are sometimes unknown, or time-varying as the SSPA is time-varying due to the above-mentioned variations. In this case, considering the SSPA parameters ($\mathcal{E}$), the posterior distribution associated with the likelihood in (6.14) can be written as

$$P(x, \mathcal{E} | r, \sigma^2) = P(s, \mathcal{E} | r, \sigma^2),$$

(6.17)

where time index $t$ is omitted for brevity. In order to solve this problem based on SMC, a set of samples from the above posterior distribution should be made available, which is complicated. To simplify matters, we rely on the Rao-Blackwellisation [93], which is a well-known technique in mathematical statistics that allows us to sample from a lower-dimensional distribution. Using the Rao-Blackwellisation technique and (6.17), the probability of a specific symbol, $s_i$, being transmitted is

$$P(s = s_i | r, \sigma^2) = \int P(s_i, \mathcal{E} | r, \sigma^2) d(\mathcal{E}).$$

(6.18)
6 Compensation of the Nonlinearity of Power Amplifiers Using Sequential Monte Carlo

which is proportional to

\[ \int P(r|s_i, \mathcal{E}, \sigma^2)P(s_i, \mathcal{E}|\sigma^2)d(\mathcal{E}) \]  

(6.19)

\[ = \int \int P(r, y|s_i, \mathcal{E}, \sigma^2)P(s_i, \mathcal{E})d(\mathcal{E})dy \]  

(6.20)

\[ = \int \int P(r|y, s_i, \mathcal{E}, \sigma^2)P(y|s_i, \mathcal{E})P(s_i, \mathcal{E})d(\mathcal{E})dy \]  

(6.21)

\[ \propto \int \exp\left\{ \frac{(r-y)^2}{2\sigma^2} \right\}P(y|s_i, \mathcal{E}, \sigma^2)P(s_i, \mathcal{E})d(\mathcal{E}) \]  

(6.22)

where (6.19) is derived from (6.18) based on the Bayes’ theorem. Inserting the output signal of the SSPA into equation (6.19) results in (6.20), where it is assumed that \( s_i \) and the SSPA parameters, \( \mathcal{E} \), are statistically independent of the noise variance. Equation (6.21) is derived based on the system model in Fig. 6.1. Finally (6.22) is written using the fact that \( P(y|s_i, \mathcal{E}, \sigma^2) = \delta[y-G(s_i, \mathcal{E})] \), and a uniform distribution for all the possible constellation points is assumed, i.e., \( P(s = s_i) = 1/M \). Moreover, it is assumed that the transmit signal \( s_i \) is statistically independent from \( \mathcal{E} \).

6.4 SMC Algorithm

SMC methods are commonly used for the approximation of intractable integrals and rely on the ability to draw random samples from a probability distribution. Considering \( \mathcal{E} \) as the state variable, the key idea in SMC is to represent the target posterior density function of the unobserved states, \( P(\mathcal{E}) \), by a set of \( J \) random samples, \( \{\mathcal{E}^{(j)}\}_{j=1}^J \), and their associated weights, \( \{\omega^{(j)}\}_{j=1}^J \) [93]. In other words, if \( P(\mathcal{E}) \) can be written as:

\[ P(\mathcal{E}) = \frac{1}{W} \sum_{j=1}^{J} \omega^{(j)} \delta(\mathcal{E} - \mathcal{E}^{(j)}), \]  

(6.23)

where

\[ W = \sum_{j=1}^{J} \omega^{(j)}. \]  

(6.24)
Therefore, using (6.23), the likelihood problem in (6.22) can be approximated as

\[
P(s = s_i) = \frac{1}{W} \sum_{j=1}^{J} \omega^{(j)} \exp \left\{ \frac{[r - G(s_i, \mathcal{E}^{(j)})]^2}{2\sigma^2} \right\},
\]  

(6.25)

where

\[
G(s_i, \mathcal{E}^{(j)}) = \frac{\nu^{(j)}|s_i|}{\left\{ 1 + \left[ \frac{\nu^{(j)}|s_i|}{A_o^{(j)}} \right]^{2p^{(j)}} \right\}^{1/2p^{(j)}}}.
\]  

(6.26)

The only point left to address is the prior distributions from which samples are drawn. Since the parameters of the SSPA, \(\nu, A_o, p\), are assumed to be statistically independent, the prior distribution can be modeled as

\[
P(\mathcal{E}) = P(A_o, \nu, p) = P(A_o)P(\nu)P(p)
\]

(6.27)

\[= \mathcal{N}(\bar{A}_o, \sigma_{A_o})\mathcal{N}(\bar{\nu}, \sigma_{\nu})\mathcal{N}(\bar{p}, \sigma_p),\]

where \(\mathcal{N}(\cdot, \cdot)\) represents the Gaussian distribution, and \(\bar{\nu}, \bar{A}_o, \bar{p}\), are the nominal/prior values for \(\nu, A_o, p\), respectively, and \(\sigma_{A_o}, \sigma_{\nu}, \sigma_p\) are their respective variances. Note that one or all of the parameters of the SSPA can be unknown. Therefore, in the case that just one of them is unknown, the above prior distribution is reduced to one prior. Based on the central limit theorem, and considering several factors in the underlying mechanism that leads to the parameter variations, the selection of the Gaussian distribution as the a priori distribution is justified. However, any distribution can be accommodated in this mathematical formulation without changing the formulation. Once all samples from \(P(\mathcal{E})\) are available, the expansion in (6.25) can be easily computed, which solves the likelihood problem in (6.18). This scheme is called adaptive as the amplifier parameters are estimated frequently enough so that their variations are tracked by this scheme.

Since the input sampling rate is \(1/T\), and there are \(M\) possible constellation points for \(s_i\), considering \(J\) required samples for the SMC estimation per constellation point, the sampling rate in this case is \(MJ/T\). Note that since the amplifier parameters change much slower than the symbol rate, once they are estimated, the system can switch to the “known parameters” mode (Section 6.3.3). Therefore, with a predefined rate, it switches back to the adaptive case (Section 6.3.4) if required. For instance,
when a user moves from one base-station to another, the receiver may switch to the adaptive case to estimate the parameters of the new transmitter’s SSPA. Thus considering this switching mode the overall sampling rate is significantly smaller (i.e., the sampling rate is $M\tau/T$, where $\tau \ll J$).

### 6.4.1 Unknown Parameters (Adaptive Scheme with Memory)

Using the above adaptive scheme, one can think of using an additional memory that keeps track of important samples from one interval to another. This would save computations in sampling from the distribution in (6.27). Fig. 6.4 shows the proposed adaptive scheme for one symbol $s_i$ for two consecutive time instants $t$, and $t + 1$. In Level I, $J$ samples are drawn from $P(E_t)$, where $t$ represents the time index. The weights of these samples are calculated in Level II and III based on the posterior distribution $P_t(r, y|s_i, E, \sigma^2)$. Once the weights are calculated, the samples with large
weights are duplicated while the ones with small weights are ignored. This process is called \textit{resampling} (Level III). In other words, after resampling, $K$ samples are generated based on the original $J$ samples. These $K$ samples are the best samples to estimate the SSPA parameters at time $t$, which are forwarded to the next step (white small circles in Level IV). To compensate for the ignored samples, a few random samples are also drawn from $P(E_{t+1})$ in Level IV, which are shown as gray small circles in Fig. 6.4. Based on the final samples (both white and gray circles) at time $t+1$, the whole process is repeated for the next time instant (Level V). Note that this process is implemented for each individual symbol. In other words, for each candidate symbol $s_i$, the additional memory saves the samples that contribute more to the final estimation. The metric that represents this contribution, $\omega^{(j)}(s_i)$, is defined based on the Euclidean distance between the received signal and the estimated version as follows:

$$\omega^{(j)}(s_i) = \omega^{(j)}P(r|y, s_i, E, \sigma^2)$$

$$= \omega^{(j)}\exp\left\{ \frac{(r - G(s_i, E^{(j)}))^2}{2\sigma^2} \right\}$$

(6.28)

(6.29)

Note that $\omega^{(j)}(s_i)$ is defined per symbol whose value represents the importance of the corresponding sample. Using resampling, which removes the samples with small contribution and replicates the samples with large contribution, the adaptive scheme can effectively propagate the reliable values for the parameter set $E$ over the time. In other words, resampling converts the properly weighted sample set $\{E^{(j)}_t, \omega^{(j)}_t\}_{j=1}^J$ to the updated set $\{\hat{E}^{(j)}_t, N_j\}_{j=1}^J$, where $N_j$s are integer numbers so that $\sum_j N_j = K$, $N_j = 0$ for some samples, and $K < J$. There are several approaches to implement the resampling. An efficient yet simple way to implement the resampling is proposed in [101].

Therefore, after resampling for each symbol $s_i$, there are $K$ samples of the SSPA parameters that are used for the estimation at the next time slot $t + 1$. The rest of the samples ($J - K$ samples) are drawn randomly based on the a priori distribution in (6.27). This scheme is sufficiently accurate because the reliable samples are propagated while the other values are still considered as possible candidates through the introduction of $J - K$ new samples each time the algorithm is run. Thus the instantaneous variations in the SSPA parameters are compensated by the random
samples while the sequential deterministic samples avoid the portion of the sampling burden from (6.27) at every single time instant.

In terms of implementation, resampling can be implemented using parallel architectures [101]. This would facilitate the integration of the sampling and resampling as they both can be implemented in parallel. Although the implementation of resampling might add an extra computation core to the system, it would save computations by limiting the number of required samples in each time instant. The amount of computation required is a function of $K$, $J$, and the distribution of the metric values. Note that due to fewer drawn samples in the adaptive case with memory, the continuous implementation in this case requires a smaller sampling rate, i.e., $M(J-K)/T$. Again employing the occasional switching to adaptive mode, results in an even smaller sampling rate. An efficient architecture for implementation of the resampling is proposed in Appendix B.

6.5 Complexity Analysis

6.5.1 Adaptive Scheme without Memory

The adaptive scheme without memory requires $J$ clock cycles to draw all the samples, hence the sampling rate of $J$. The weights of the samples are calculated in $J$ cycles in a pipelined fashion. Once the samples and weights are ready, the estimation can be done using (6.25). Note that in (6.25), the function $G$ can be implemented using a look-up table (LUT). The corresponding value of the function is determined by the drawn samples. If this method is implemented using a pipelined scheme, the sampling, weight calculation, and addressing the $G$ function LUT can all be implemented in a pipelined fashion, meaning that once the first sample is drawn, its weight can be calculated, and at the same time the second sample is drawn, and so on. Using a pipelined scheme, a total of $J + 1$ cycles per symbol is required to estimate the parameters. If multiple samples are sent consecutively, which is normally the case, this corresponds to a latency of $J + 1$ cycles in the architecture.

6.5.2 Adaptive Scheme with Memory

For the adaptive scheme with memory (Section 6.4.1), since after the resampling, $K$ samples are already determined from the previous time instant, only $J - K$ samples
should be drawn. Thus $J - K$ cycles for sampling, and $J - K$ cycles for the weight calculation are required, hence the sampling rate of $J - K$. In this scheme, however, resampling needs to be implemented as well, which takes $J$ cycles to be implemented [101], [102]. Similar to the previous case, using the pipelined implementation the number of cycles for the sampling and weight calculation would be $J - K + 1$. Thus the total number of cycles is $2J - K + 1$ including the resampling. Considering the fact that $J > K$, the complexity of this case is higher while having a lower sampling rate and better performance results (see Section 6.6).

6.6 Performance Analysis and Simulation Results

In this section, simulation results comparing the performance of the predistorter and SMC is presented for M-QAM constellations, where $M = 4, 16, 64,$ and 256. The SMC receiver shows a significant improvement compared to the predistorter while the predistorter is better only in a very limited number of cases. The system under simulation for the SMC is the same as the one in Fig. 6.1, while the system structure for the simulation of the predistorter is based on Fig. 6.3 where for both an AWGN channel with a known variance for the received noise is assumed. The effect of other blocks of the system such as the matched filter, are not considered in this work. The simulation results for all of the above mentioned constellation schemes are presented, as the comparison between them elucidates some important points. The results are considered under two scenarios, known and unknown SSPA parameters.

6.6.1 Known Parameters

Fig. 6.5 shows the performance of the predistorter and particle filtering (SMC) receiver for a 4-QAM scheme with different IBO values. The performance curve for the ideal linear amplifier as well as the case where there is no compensator (HPA only) are also depicted. For these simulations we assumed $\nu = 1, A_o = 1, p = 2, A_s = 2.65$ and equal weights. The IBO value is defined as in (6.7) whose value determines how far the average power of the transmitted symbols are from the saturation region. In other words, low values of IBO implies that most of the points are in the saturation region while high values of IBO correspond to the case where most of the points are in the linear region of the SSPA characteristic curve. Note that in (6.7), $P_{I,sat} \propto A_{in,sat}^2$, where $A_{in,sat}$ is the input voltage associated with the maximum output voltage, and
Figure 6.5: Performance of SMC compared to the predistorter with different input backoff values for a 4-QAM scheme: (a) IBO = 6 dB, (b) IBO = 9 dB, (c) IBO = 12 dB and (d) IBO = 15 dB.

\( \bar{P}_1 \) is the average power of the input symbols, which is a function of the constellation scheme. For our chosen parameters, \( A_{\text{in, sat}} = 2.63 \). Fig. 6.6 also shows the amplified symbols with different values of IBO when SNR = 16 dB. For high-IBO values, the distortion is negligible while for low-IBO values nonlinear distortion is apparent.

There are a few points worth noting in Fig. 6.5. Firstly, for a 4-QAM case, the performance of the predistorter is better or equal to the SMC performance. The reason is that in this case there is only one point in each quadrant of the constellation map. Therefore, the amplitude of the amplified symbol using a predistorter is always larger or equal to the amplitude of the symbol in the case of SMC (see Fig. 6.2 and compare the “SSPA+predistorter” line with the “SSPA” line). Thus the predistorter results in a higher effective received SNR. Since there is almost no phase distortion in
Figure 6.6: The received points with different values of IBO for 16-QAM at SNR = 16: (a) IBO = 4 dB and (b) IBO = 10 dB.

the amplified signal, the detection performance for a 4-QAM scheme loosely depends on the input IBO. Thus the predistorter outperforms SMC in this case. Secondly, as the IBO increases, e.g., IBO = 12 dB, 15 dB, the performance of the predistorter and SMC gets closer. This is because all the points lie in the linear region of the amplifier.

This concept can also be extended as follows. When constant-envelope constellations such as PSK schemes are employed, assuming there is no phase distortion, the predistorter outperforms the SMC in low-IBO regimes. This is because of the fact that in these constellation schemes, information is in the phase and not the amplitude of the transmitted symbol. So assuming no phase distortion, the hard cut-off in the amplitude, as being done in the predistorter, does no harm to the detection process. However, in the presence of the phase distortion the SMC outperforms the predistorter because the phase distortion is compensated through the sampling from (6.4). In other words, the important feature of the SMC approach is the ability to distinguish between the transmitted symbols in a M-QAM system where the symbol phases and amplitudes are different as will be shown in the sequel for 16-QAM, 64-QAM, and 256-QAM systems. These constellation schemes are typical in current and future standards such as IEEE802.16e/m.

For instance, Fig. 6.7 shows the performance curves for a 16-QAM scheme. In this case the SMC method outperforms the predistorter under certain conditions, especially for low-IBO values. For IBO = 9 dB the predistorter performance is poor.
Figure 6.7: Performance of SMC compared to the predistorter with different input backoff values for a 16-QAM scheme: (a) IBO = 7 dB, (b) IBO = 9 dB, (c) IBO = 12 dB, and (d) IBO = 15 dB.

as most of the constellation points are in the saturation region and the predistorter cannot distinguish between them. However, the SMC receiver because of its soft nature, still has relatively a good performance. Soft means that there is no hard limiter in the SMC method. This is because SMC draws samples from the SSPA transfer function independent of the IBO value, as opposed to the predistorter, which has a hard clipping function for \( A_{\text{in}} > A_0 \) (see Fig. 6.2).

Fig. 6.7(a) shows that for low-SNR values the predistorter and SMC has the same performance, however, by increasing the SNR, SMC outperforms the predistorter. In other words, the higher the SNR, the more concisely the SMC receiver can draw samples, i.e., higher resolution. However, it doesn’t benefit the predistorter, as increasing the SNR does not change the number of points in the saturation region. Another
Figure 6.8: Performance of SMC compared to the predistorter with different input backoff values for a 64-QAM scheme: (a) IBO = 9 dB, (b) IBO = 10 dB, (c) IBO = 12 dB, (d) IBO = 15 dB.

interesting fact is illustrated in Fig. 6.7(b) where it shows that for higher IBO values, the predistorter starts to outperform SMC but never by more than 0.8 dB, since in this case all the points are in the linear region. Fig. 6.7(c) and 6.7(d), on the other hand, show that by increasing the IBO values, the performance of all approaches become closer and finally the same for really high IBO values.

Fig. 6.8 shows the performance results for 64-QAM and reveals other details. Comparison between Fig. 6.7(b) and Fig. 6.8(a) clarifies the fact that even if IBO is forced to be constant, by scaling the constellation up, the predistorter performs poorer than SMC. Note that, as opposed to the predistorter, the curve for SMC does not have an error floor. This implies that SMC has a robust performance with respect to the constellation scheme. This is an interesting and promising feature of
Figure 6.9: Performance of SMC compared to the predistorter with different input backoff values for a 256-QAM scheme: (a) IBO = 10 dB, (b) IBO = 12 dB.

the proposed SMC receiver in the context of adaptive modulation, which dominates most modern standards. For IBO at 12 dB and 15 dB, the difference between the SMC receiver and the predistorter is at most 0.7 dB and 0.1 dB, as shown in Fig. 6.8(c) and 6.8(d), respectively.

An important point, revealed from 6.8(b), is that the higher the SNR, the more precisely the SMC can draw samples from the target distribution in (6.17). In other words, the weights are a true representative of the importance of the samples. This is because of the fact that at the near saturation region of the SSPA cure, if the noise variance is smaller, the SMC due to its soft nature can distinguish between the symbols more easily. This is the main reason behind the cross-over between in the predistorter and SMC curves in Fig. 6.8(b). This implies that the performance of the SMC becomes better as the SNR increases. This normally happens especially for mid-IBO values (here 10 dB). In brief, the SMC outperforms the predistorter in three cases, namely high-order constellation schemes, low-IBO regimes, or in mid-IBO regimes with high-SNR value.

Finally, the performance curves for 256-QAM are shown in Fig. 6.9. This constellation has the highest density of points compared to the others. As a result, the threshold at which the predistorter outperforms SMC is higher [compare 6.8(b) to 6.9(b)]. The reason that there is an error floor for the predistorter for low IBO
values is shown in Fig. 6.10. The black circle represents the saturation boundary meaning that the points outside of the circle lie in the saturation region. As seen in Fig. 6.10(a), for 16-QAM only four points are in the saturation region and since they are mapped to the circle there is no error floor for 16-QAM at IBO=9 [see Fig. 6.7(b)]. However, for 256-QAM, Fig. 6.10(b), there are quite a few symbols outside of the circle. Therefore, the SSPA actually maps them on the circle circumference without changing their phases. In this case increasing the SNR provides no advantage whatsoever, which results in the error floor in the case of the predistorter [see Fig. 6.9(a)].

Fig. 6.11 gives another interpretation of the above concept. It shows the percentage of the constellation points that are in the saturation region as a function of the input IBO of the amplifier for both the predistorter (black bars) and SMC (white bars). It clearly shows that the number of saturated points of the predistorter is always greater than or equal to the saturated points of SMC. Additionally, SMC is a soft detector, whereas the predistorter is a hard detector, so we would expect that SMC outperforms the predistorter in low IBO regimes.

In brief, Fig. 6.7 to Fig. 6.9 show that the performance of SMC is more robust to constellation scaling, and at the same time for low-IBO regimes and high-order constellation schemes it has a better performance than the predistorter while having
Figure 6.11: The percentage of the points in the saturation region vs. IBO value for the predistorter (black bars) and SMC (white bars), (a) 4-QAM, (b) 16-QAM, (c) 64-QAM, (d) 256-QAM.

no error floor. In other words,

- The SMC outperforms the predistorter almost everywhere for 64-QAM and 256-QAM.
- The SMC outperforms the predistorter in 16-QAM for low IBO values, mid-IBO values with high received SNR.
- The predistorter is at best 0.8 dB better than the SMC and only for a restricted operating region.

The final issue to be addressed is to see under what conditions the SMC receiver is preferred to the predistorter as a technique to compensate the distortions caused by the SSPA. A useful performance measure in the context of nonlinear power amplifiers is the total degradation (TD) as a function of the HPA output backoff (OBO) [103]. This measure is defined as follows:

\[
TD_{\text{dB}} = \text{SNR}_{\text{dB}} - \hat{\text{SNR}}_{\text{dB}} + \text{OBO}
\]  

(6.30)
Figure 6.12: Total degradation of different modulation schemes vs. OBO for both SMC and the predistorter for SER = $10^{-2}$ (a) 16-QAM (b) 64-QAM (c) 256-QAM (d) All.

where $\text{SNR}_{\text{dB}}$ is the required signal-to-noise ratio at the input of the receiver to obtain a fixed symbol error rate (SER) for a given value of OBO, and $\hat{\text{SNR}}_{\text{dB}}$ is the required signal-to-noise ratio to obtain the same SER in the absence of nonlinearity in an AWGN channel. Note that TD can be calculated for both the predistorter and SMC method, and is a reasonable way to compare the performance of these two methods. In other words, for a specific constellation and OBO value, the technique that gives the lowest TD is preferred.

Fig. 6.12 shows the TD corresponding to the SER = $10^{-2}$ for 16-QAM, 64-QAM, and 256-QAM. For a specific TD value, the parameter $\delta$ represents the difference in OBO between the SMC and the predistorter to achieve the same TD. The initial value of $\delta$ for all constellation values are also shown in the figure. Fig. 6.12(a) shows that the
predistorter outperforms SMC for OBO > 2.9 dB. This point is called the threshold point. Fig. 6.12(b) and Fig. 6.12(c) show that the threshold point moves to the higher values of OBO as the constellation scheme scales up, i.e., the threshold points for 64-QAM, and 256-QAM are about 4.7 dB and 5.9 dB, respectively. Note that for OBO values less than the threshold point, SMC by far outperforms the predistorter. For values above the threshold point, the difference between the performance of SMC and the predistorter for 64-QAM and 256-QAM is negligible. Therefore, for any constellation above 64-QAM, the technique of choice is SMC everywhere in the OBO region. Fig. 6.12(d), finally shows all the TD curves in the same figure for comparison. Note that if Fig. 6.12 is considered for smaller values of SER, then the performance advantage of SMC would be more evident meaning that the value of the initial $\delta$ would be higher.

In brief, the SMC outperforms the predistorter in most cases except in 16-QAM and for $2.9 \text{ dB} < \text{OBO} < 6 \text{ dB}$, where the difference in the total distortion between SMC and the predistorter is at most 0.8 dB. As far as the 64-QAM and 256-QAM constellation schemes are concerned, the results show that the SMC receiver can be used to advantage almost anywhere in the OBO region, where for low-OBO values SMC is, by far, better than the predistorter [see Fig. 6.9(b)], whereas for high-OBO values the performance loss is at most 0.2 dB in the worst case.

### 6.6.2 Unknown Parameters

In this case, at least one of the amplifier parameters ($A_o, \nu, \rho$) are assumed unknown so the task of SMC is to find the best estimate of that parameter and ultimately detect the symbol and compensate for the nonlinear distortion caused by the SSPA. One of the main motivations to use SMC methods is because the amplifier parameters might be known on average but due to manufacturing or environmental variations, they might change. Moreover, these parameters might not be known at all at the beginning. Therefore, we need an approach to estimate these parameters blindly or track their variations efficiently, since it directly affects the performance of the system. SMC methods are of interest as they do both tasks at the same time, meaning that they estimate the parameters and at the same time they detect the transmitted symbol.
Unknown $A_o$

One of the most common variations in amplifiers are the saturation voltage or the maximum output voltage, $A_o$ (see Fig. 6.2). In this case, the amplitude of the signal would be a function of $A_o$. Therefore, we have to look for a method that can first track the variations and detect the transmitted symbol simultaneously. In this case $p$, and $\nu$ are assumed to be known. The only information that the receiver has is the fact that $A_o$ has a certain degree of variation around a nominal value, $\bar{A}_0$. For simulation purposes, we chose $\bar{A}_0 = 1$, and the number of samples is 10. We randomly chose the value for $A_o$ using a uniform distribution with the mean equal to the nominal value and a variance of 0.3, so the signals are transmitted based on the chosen value of $A_o$. At the receiver, using the first prior density in (6.27), we draw 10 samples, based on which the likelihood function in (6.22) is calculated.

Unknown $\nu, p$

In this case, $A_o$ is assumed to be constant and the other two parameters are randomly chosen. The goal is to verify the effect of the individual parameters on the performance of SMC. The nominal values are $\bar{\nu} = 1$, and $\bar{p} = 2$. The value of each actual parameter is randomly chosen using a uniform distribution with the mean equal to the nominal value and variance of 0.3. At the receiver, using the last two prior densities in (6.27), we draw 10 samples, based on which the likelihood function in (6.23) is calculated.

Unknown $A_o, \nu, p$

In the case where all the parameters are unknown, the only thing the receiver knows is this nominal values and variance. We call the above receiver an adaptive SMC receiver as it adaptively estimates the unknown parameters of the SSPA, based on which they detect the transmitted symbol. Fig. 6.13 and 6.14 show the performance of all the above adaptive scenarios for 16-QAM and 64-QAM, respectively. This proves that an adaptive SMC approach is very robust in the sense that its performance is very close to the perfect SMC, where all the parameters are known. Thus, using the adaptive SMC, two main goals are achieved. First, the performance is much better than the predistorter in low-IBO regimes, and secondly, the performance of the system is robust to parameter variations. Fig. 6.13 shows the performance of different cases of adaptive SMC receivers for 16-QAM and IBO = 7 dB. The number of samples used
for all the adaptive receivers was 10. We also assumed that all the weights are equal. It shows that for different cases of adaptive SMC, where one or more parameters are unknown, performance results are nearly the same as the perfect SMC. In other words, the adaptive SMC is capable of joint blind estimation of all the parameters of the amplifier and detection of the transmitted symbols simultaneously. Note that the performance of the adaptive cases are all worse than that of the SMC receiver with known parameters, which is expected as there are more dimensions added to the unknown parameters.

Fig. 6.14 on the other hand, shows the performance of adaptive receivers for 64-QAM and IBO = 10 dB. It clearly shows the differences between various adaptive cases compared to the perfect SMC. As shown, the perfect SMC has the best performance, then the adaptive SMC with unknown $A_0$, then adaptive SMC with unknown $\nu, p$, and finally, the worst case is the adaptive SMC with all parameters unknown.

Fig. 6.15 shows the performance of our proposed sequential adaptive SMC receiver for 64-QAM and IBO = 10 dB. In this simulation we assume $J = 10$, $K = 5$, and the parallel resampling scheme proposed in [104]. The results show that employing the
sequential adaptive SMC receiver results in a performance improvement in all cases. This is because of the fact that the estimation from the previous time instants are used towards creation of more reliable values, which is intuitive as the parameters of the SSPA do not change too often.

In terms of implementation, adaptive SMC has some benefits including effective pipelined and parallel VLSI realization [101], [105]. This is possible because of the parallelism intrinsic to the algorithm and because it does not need many iterations to converge. This is a key advantage relative to the adaptive predistorter, because adaptive predistorters are normally implemented by look-up tables, which either have large convergence time [87], or results in additional hardware complexity to mitigate the large computational delay [106].

The candidate applications for our proposed SMC receiver might include OFDM systems where the peak-to-average-power-ratio is high, which may result in poor performance, especially for low-IBO values. Applying the SMC scenario in this context has two advantages. First, due to its soft estimation nature, the SMC receiver does not result in the performance loss, and secondly, it allows the use of low-IBO values,
6 Compensation of the Nonlinearity of Power Amplifiers Using Sequential Monte Carlo

Figure 6.15: Sequential adaptive vs adaptive receiver for 64-QAM for IBO = 10 dB.

which in turn is translated to higher power amplifier efficiency at the transmitter, rather than higher input backoffs normally used in other approaches like the predistorter.

6.7 Limitations of a Multi-carrier System

Based on the above simulations, the SMC method is a very efficient and yet simple-to-implement approach to use for HPA nonlinearity compensation. But it is important to investigate the effect of the SMC approach on the spectral spreading of the signal in multi-carrier systems such as OFDM. This investigation will help elucidate restrictions on the application of the SMC in terms of the range of the applicable OBO values that can be used, which are a result of the frequency smearing caused by the nonlinearity of the SSPA. Note that the result of this section can be applied to any single-carrier system with a spectral mask limitation.

Based on the results in Fig. 6.12(d), the SMC approach is more effective that the predistorter for a wide range of OBO values. However, for very low values of OBO,
the spectrum of the signal is spread increasing the adjacent channel interference level. In most standards, there is a spectral mask criteria that needs to be met, regardless of the compensation method used for the power amplifier. Thus, it is vital to investigate the range of the OBO values where the system meets the spectral mask requirements while using the SMC framework.

We employed an OFDM system, the IEEE802.11g system, with the center frequency at 2.4GHz. The spectral mask of such a system is shown in Fig. 6.16. In other words, the transmitted spectral products should be less than -30 dBr for $f_c - 22 \text{ MHz} < f < f_c - 11 \text{ MHz}$, and $f_c + 11 \text{ MHz} < f < f_c + 22 \text{ MHz}$. Moreover it should be less than -50 dBr for $f < f_c - 22 \text{ MHz}$, and $f > f_c + 22 \text{ MHz}$, where $f_c$ is the channel center frequency, and dBr is the relative power in dB to the peak power at the center frequency.

Spectral spreading is evaluated by simulation using pseudo-random modulated signals passed through the filters and SSPA. The power spectral density of the transmitted signal is determined using the fast Fourier transform (FFT). The spectrum spreading of the symmetric M-QAM signals into the neighboring frequency bands, which are amplified by the non-linear amplifier, depends on the OBO of the transmitted signal.

Fig. 6.17 shows the SSPA output signal spectrum for the 16-QAM constellation scheme, showing both the in-band and out-of-band portion of the signal. The simulated framework was an OFDM system with FFT size of 64 and the guard interval
was chosen to be 16. In each simulation 1000 OFDM symbols were simulated. The spectrums are shown for various OBO values. As is shown, the spectrum test fails for any OBO value less than 1.9 dB. For smaller values of OBO, either the first criterion (-30 dBr at ±11 MHz), or the second criterion (-50 dBr at ±22 MHz), as illustrated in Fig. 6.16, is not met. For instance, for OBO=1.5 dB the spectral products are less than -30 dBr for $f_c - 22 \text{ MHz} < f < f_c - 11 \text{ MHz}$ but is not less than -50 dBr for $f < f_c - 22 \text{ MHz}$.

Fig. 6.18 shows the result for the 64-QAM scenario. The threshold value of OBO for this case is around 3.2 dB. Therefore, the minimum OBO value at which the SMC demodulator satisfies the spectrum mask test is a function of the constellation scheme. For instance, in the case of 16-QAM, the lowest allowable OBO value for SMC demodulator is 1.9 dB, while it is around 3.2 dB for 64-QAM and 3.6 dB for 256-QAM, respectively. This is because of the fact that the 16-QAM signal has the lowest
peak-to-average-ratio (PAPR) and consequently its spectrum spreading due the non-linear amplification is lower than for 64-QAM, and 256-QAM signals. Therefore, it supports lower input/output back-offs.

Considering the results in Fig. 6.17, and Fig. 6.18 along with the simulation result from Fig. 6.12(d), determines the effective range of application of the SMC. In other words, the spectral mask imposes a lower bound on the minimum allowable OBO value that can be employed in the SMC receiver to avoid the adjacent channel interference. Fig. 6.19(a), and Fig. 6.19(b) show the difference in TD value between the predistorter and the SMC versus the OBO value for 16-QAM and 256-QAM, respectively. A negative difference means the TD value associated with the predistorter is lower than that of the SMC receiver, indicating the preferred point of the predistorter operation. On the other hand a positive value for the difference represents the areas where the SMC method outperforms the predistorter. In both figures there are three
6 Compensation of the Nonlinearity of Power Amplifiers Using Sequential Monte Carlo

Figure 6.19: The preferred operating region of the SMC and predistorter as a function of OBO considering the mask constraint for: (a) 16-QAM, (b) 256-QAM.

regions, namely (i) the SMC-preferred region (the whole gray area), which shows the areas in which the SMC outperforms the predistorter, (ii) permitted by mask (the gray dashed area), which reflects the limitation of the mask on the OBO value, and (iii) the predistorter-preferred region (the white area under the OBO axis), which shows the areas in which the predistorter has a better performance than the SMC. The permitted region by mask is in fact a subset of the SMC-preferred region. Note that the preferred-region of the SMC, which is permitted by the mask, widens while the preferred-region of the predistorter shrinks as the constellation level scales up (Fig. 6.19(b)). Moreover, the maximum gain achieved by employing the predistorter in its preferred region decreases by scaling the constellation (0.8 dB in 16-QAM as opposed to 0.2 dB in 256-QAM). Finally note that in any application with no mask constraint the SMC has a wide range of OBO values (Fig. 6.12).

6.8 Summary

The SMC method was studied and an adaptive SMC receiver was proposed for M-QAM schemes in the presence of solid-state power amplifiers. The performance results were compared with the predistorter method. It was shown that the SMC outperforms the predistorter in most cases except in 16-QAM and for $2.9 \, \text{dB} < \text{OBO} < 6 \, \text{dB}$, where the difference in the total distortion between SMC and the predistorter is at most 0.8 dB. As far as the 64-QAM and 256-QAM constellation schemes are
Concerned, the results show that the SMC receiver can be used to advantage almost anywhere in the OBO region, where for low-OBO values SMC is, by far, better than the predistorter [see Fig. 6.9(b)], whereas for high-OBO values the performance loss is at most 0.2 dB in the worst case. The effect of the SMC receiver on the spectral smearing was also studied where a lower bound was determined for the minimum allowable OBO value, which is a function of the constellation scheme and the spectral mask itself. The proposed SMC receiver is soft in nature, is more robust to the constellation scaling, and outperforms the predistorter for low-value OBOs.
7 Conclusions and Future Directions

7.1 Conclusions

Developing a scalable high-throughput MIMO detector for high-order constellation sizes has been a significant challenge in the literature. To address this issue in this thesis a novel on-demand scheme for the K-Best MIMO detector suitable for high-order constellations was proposed. It has the lowest latency reported to-date as well as the smallest number of visited nodes. It is applicable to the unbounded infinite lattice decoding problem and has the fixed-critical path independent of the constellation scheme or the lattice size. The key contribution was the introduction of a novel joint on-demand expansion and distributed sorting scheme operating in a pipelined fashion.

The detailed VLSI architecture for the proposed on-demand K-Best detector, whose critical path is independent of the constellation size, was realized, which achieves the highest throughput reported to-date. In 0.13 \( \mu \)m CMOS, it achieves 675 Mbps at 282 MHz satisfying the throughput requirements of next-generation WiMAX systems while consuming 200 pJ/bit, the lowest reported in the literature to-date.

Moreover, an efficient lattice-reduction (LR) aided implementation of the K-Best algorithm has been proposed for the lattice detection problem in a general infinite lattice framework. The proposed on-demand candidate generation along with the proposed distributed sorting scheme leads to an efficient implementation of the K-Best, whose complexity is independent of the constellation scheme. The proposed scheme does not require any a priori knowledge of the candidate displacement, is scalable in terms of the number of transmit antennas and its complexity grows sub-linearly with the constellation order. The main advantage of the joint implementation is a higher diversity and a better performance result at the receiver.

Furthermore, the SMC method was studied and an adaptive SMC receiver was proposed for M-QAM schemes in the presence of solid-state power amplifiers. The performance results were compared with the predistorter method. It was shown
that the SMC outperforms the predistorter in most cases except in 16-QAM and for 2.9 dB < OBO < 6 dB, where the difference in the total distortion between SMC and the predistorter is at most 0.8 dB. As far as the 64-QAM and 256-QAM constellation schemes are concerned, the results show that the SMC receiver can be used to advantage almost anywhere in the OBO region, where for low-OBO values SMC is, by far, better than the predistorter, whereas for high-OBO values the performance loss is at most 0.2 dB in the worst case. The effect of the SMC receiver on the spectral smearing was also studied where a lower bound was determined for the minimum allowable OBO value, which is a function of the constellation scheme and the spectral mask itself. The proposed SMC receiver is soft in nature, is more robust to the constellation scaling, and outperforms the predistorter for low-value OBOs. A scalable, high performance, low memory usage architecture was proposed for the parallel implementation of resampling. The proposed scheme has an execution time independent of the distribution of the weights.

7.2 Future Work

7.2.1 MIMO Detection

Technology Scaling

In the 0.13-µm CMOS detector presented in this work, it was observed that the static power due to the leakage current constitutes only less than 1% of the total power. As a result, in the power analysis in Chapter 4, the main focus was on the dynamic power consumption. However, it is well known that as the CMOS process technology scales to smaller geometries, the leakage power becomes a more dominant portion of the total power. Consequently, architectural and physical level leakage management techniques such as power-gating and employing multiple voltage domains and multiple threshold devices will be needed. Moreover, the main target of the VLSI architecture in Chapter 4 was to maximize the throughput. The alternatives to design efficient architectures for low-power applications remains open for research.
7 Conclusions and Future Directions

High-performance QR-decomposition

QR-decomposition is an essential signal processing core for the K-Best algorithm. In this thesis, it was assumed that the QR-decomposition is implemented ahead providing results to the K-Best receiver. However, decomposition of complex MIMO channel matrices with large dimensions leads to high computational complexity, and hence results in either large core area or low throughput. Considering the fact that the channel estimate for dynamic channels is assumed to be valid for only perhaps four consecutive symbol intervals, makes the implementation of a high-performance QR-decomposition an interesting and challenging topic for investigation both in suboptimal receivers such as BLAST and near-optimal detectors such as the K-Best detector.

Extension to 256-QAM

Since 256-QAM is considered as a possible optional modulation for the future standards, the ASIC implementation of the K-Best detector in Chapter 4 for a $4 \times 4$ 256-QAM is interesting. There is no hardware implementation in the literature because of the significant increase in the complexity of a K-Best detector for 256-QAM constellation. However, the on-demand nature of the design proposed in this thesis makes the realization of a 256-QAM system feasible.

Extension to Soft Detection

The proposed on-demand K-Best scheme presented in Chapter 3 applies only to the hard detection scenarios. The investigation of the extension of this scheme to the soft version so that a turbo receiver in coded MIMO systems can be realized is an interesting avenue to explore. On the algorithm side, the required value of $K$ for different constellations, the required number of children at the last stage, the effect of the incomplete candidate list on the LLR values and the final performance of the receiver needs to be investigated. On the implementation side, the effect of larger candidate list on the latency, throughput is an open area for research. An effective VLSI architecture for the soft version might reuse most of the cores in the hard detector and add more processing blocks in parallel to efficiently add more candidates to the list while not sacrificing the achieved throughput.
Complex Mode

The complex version of the on-demand K-Best scheme was proposed in Chapter 3. The efficient hardware implementation of the complex mode is an interesting topic. In fact, it was shown that the number of nodes to be searched to find the next best child is \( \mathcal{L} \leq \sqrt{M} \). However, a hardware implementation that achieves a high-throughput detection is challenging and is worth investigating further.

### 7.2.2 Lattice Reduction

There are several areas of further research in the field of lattice reduction. First, all the possible alternatives for the LR algorithm are required to be investigated to see the effect of different algorithms on the performance of the joint K-Best and LR implementation. In other words, it is not clear which LR algorithm (LLL, Seysen or Brun) is better for LR-aided K-Best detectors. secondly, the hardware implementation of the LR core has not been addressed in this thesis. In order to do so, the complexity of the different approaches as well as their joint implementation with K-Best detector is a possible extension. Moreover, the simulation results presented in Chapter 5 showed that LR needs to be jointly considered with K-Best algorithm for high-SNR regimes. Therefore, a configurable architecture that adaptively activates the LR core based on the SNR measurements is an interesting extension.

The integration of the real-mode K-Best and the real LR was discussed in Chapter 5. There are a few approaches for implementation of the LR in complex mode [71]. The integration of the complex LR algorithm with the complex on-demand K-Best scheme presented in Chapter 3 is an interesting area for investigation. Another possibility for integration is an investigation of efficient architectures for implementation of joint QR-decomposition and LR as the preprocessing blocks with the K-Best detector.

### 7.2.3 SSPA Compensation

The algorithm developed in Chapter 6 was not realized in hardware. Since it has several sub-blocks, efficient VLSI implementation of the proposed algorithm for high-throughput systems is a major challenge. In particular, the resampling core, which is the sequential part of the algorithm, is the bottleneck and needs to be customized for systems with strict QoS requirements. Extension of the proposed SMC scheme to the MIMO systems is another interesting road to investigate.
A Detailed Measurement Results

A.1 Test Results @ 80°C

This Appendix presents the test results from testing five working chips at 80°C. The detailed measurement results of all the chips in terms of the maximum operating frequency, operating current and power is also documented in Table A.1 to Table A.15 below.

Test results yield the plot shown in Fig. A.1 for maximum operating frequency and power dissipation as a function of supply voltage at 80°C. The results yield a clock rate of 250MHz while dissipating 104mW @ 1.2V producing a sustained data rate of 600Mbps.
Figure A.1: Measurement plots for maximum frequency and power dissipation vs. supply voltage ($V_{dd}$) at 80°C.
Table A.1: Measurement Results for Chip #1 @ 0°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)¹</td>
<td>13.6</td>
<td>9.8</td>
<td>7.7</td>
<td>5.67</td>
<td>4.7</td>
<td>3.85</td>
<td>3.45</td>
</tr>
<tr>
<td>f (MHz)²</td>
<td>73</td>
<td>102</td>
<td>129</td>
<td>179</td>
<td>211</td>
<td>259</td>
<td>287</td>
</tr>
<tr>
<td>I (mA)³</td>
<td>19.4</td>
<td>28.9</td>
<td>43</td>
<td>57.4</td>
<td>73.5</td>
<td>89.6</td>
<td>105.3</td>
</tr>
<tr>
<td>P (mW)⁴</td>
<td>13.6</td>
<td>23.2</td>
<td>38.8</td>
<td>57.4</td>
<td>81</td>
<td>107.5</td>
<td>137</td>
</tr>
</tbody>
</table>

¹ t: clock period. ² f: clock frequency. ³ I: core current. ⁴ P: core power @ supply voltage.

Table A.2: Measurement Results for Chip #1 @ 25°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)</td>
<td>13.6</td>
<td>9.8</td>
<td>7.7</td>
<td>5.7</td>
<td>4.8</td>
<td>3.98</td>
<td>3.6</td>
</tr>
<tr>
<td>f (MHz)</td>
<td>73</td>
<td>101</td>
<td>129</td>
<td>175</td>
<td>206</td>
<td>251</td>
<td>277</td>
</tr>
<tr>
<td>I (mA)</td>
<td>19.4</td>
<td>28.9</td>
<td>42.3</td>
<td>55.8</td>
<td>71</td>
<td>86.3</td>
<td>102</td>
</tr>
<tr>
<td>P (mW)</td>
<td>13.6</td>
<td>23.1</td>
<td>38.1</td>
<td>55.8</td>
<td>79.6</td>
<td>103.5</td>
<td>132.5</td>
</tr>
</tbody>
</table>

Table A.3: Measurement Results for Chip #1 @ 80°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)</td>
<td>13.7</td>
<td>9.9</td>
<td>7.9</td>
<td>5.9</td>
<td>5.0</td>
<td>4.15</td>
<td>3.75</td>
</tr>
<tr>
<td>f (MHz)</td>
<td>72</td>
<td>101</td>
<td>126</td>
<td>169</td>
<td>200</td>
<td>240</td>
<td>267</td>
</tr>
<tr>
<td>I (mA)</td>
<td>19.2</td>
<td>29</td>
<td>42</td>
<td>54.6</td>
<td>70.2</td>
<td>85.8</td>
<td>98</td>
</tr>
<tr>
<td>P (mW)</td>
<td>13.44</td>
<td>23.1</td>
<td>37.8</td>
<td>54.6</td>
<td>77</td>
<td>103</td>
<td>127.5</td>
</tr>
</tbody>
</table>
### Table A.4: Measurement Results for Chip #2 @ 0°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)</td>
<td>12.37</td>
<td>8.22</td>
<td>6.8</td>
<td>5.36</td>
<td>4.55</td>
<td>3.7</td>
<td>3.35</td>
</tr>
<tr>
<td>f (MHz)</td>
<td>81</td>
<td>121</td>
<td>147</td>
<td>186</td>
<td>219</td>
<td>265</td>
<td>297</td>
</tr>
<tr>
<td>I (mA)</td>
<td>21.7</td>
<td>34.7</td>
<td>47.1</td>
<td>59.6</td>
<td>75</td>
<td>90.4</td>
<td>109.2</td>
</tr>
<tr>
<td>P (mW)</td>
<td>15.2</td>
<td>27.8</td>
<td>42.5</td>
<td>59.6</td>
<td>82.5</td>
<td>108.5</td>
<td>141</td>
</tr>
</tbody>
</table>

### Table A.5: Measurement Results for Chip #2 @ 25°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)</td>
<td>12.4</td>
<td>8.8</td>
<td>7.15</td>
<td>5.5</td>
<td>4.65</td>
<td>3.85</td>
<td>3.5</td>
</tr>
<tr>
<td>f (MHz)</td>
<td>80</td>
<td>113</td>
<td>139</td>
<td>181</td>
<td>213</td>
<td>259</td>
<td>286</td>
</tr>
<tr>
<td>I (mA)</td>
<td>21.3</td>
<td>31</td>
<td>44.7</td>
<td>58.4</td>
<td>48</td>
<td>88</td>
<td>105.3</td>
</tr>
<tr>
<td>P (mW)</td>
<td>14.9</td>
<td>24.7</td>
<td>40</td>
<td>58.4</td>
<td>52.8</td>
<td>106</td>
<td>136.9</td>
</tr>
</tbody>
</table>

### Table A.6: Measurement Results for Chip #2 @ 80°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)</td>
<td>12.4</td>
<td>9.1</td>
<td>7.45</td>
<td>5.8</td>
<td>4.95</td>
<td>4.1</td>
<td>3.7</td>
</tr>
<tr>
<td>f (MHz)</td>
<td>80</td>
<td>110</td>
<td>134</td>
<td>169</td>
<td>202</td>
<td>243</td>
<td>268</td>
</tr>
<tr>
<td>I (mA)</td>
<td>21.3</td>
<td>31</td>
<td>43.3</td>
<td>55.6</td>
<td>71.7</td>
<td>87.9</td>
<td>99</td>
</tr>
<tr>
<td>P (mW)</td>
<td>14.9</td>
<td>24.7</td>
<td>38.9</td>
<td>55.6</td>
<td>78.9</td>
<td>105</td>
<td>129</td>
</tr>
</tbody>
</table>
Table A.7: Measurement Results for Chip #3 @ 0°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)</td>
<td>12.2</td>
<td>8.85</td>
<td>7.0</td>
<td>5.25</td>
<td>4.45</td>
<td>3.67</td>
<td>3.35</td>
</tr>
<tr>
<td>f (MHz)</td>
<td>81</td>
<td>112</td>
<td>141</td>
<td>190</td>
<td>225</td>
<td>271</td>
<td>297</td>
</tr>
<tr>
<td>I (mA)</td>
<td>21.7</td>
<td>31.1</td>
<td>44.75</td>
<td>58.4</td>
<td>75</td>
<td>91.3</td>
<td>109</td>
</tr>
<tr>
<td>P (mW)</td>
<td>15.2</td>
<td>24.9</td>
<td>40.2</td>
<td>58.4</td>
<td>82.5</td>
<td>109</td>
<td>141</td>
</tr>
</tbody>
</table>

Table A.8: Measurement Results for Chip #3 @ 25°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)</td>
<td>12.2</td>
<td>9.1</td>
<td>7.2</td>
<td>5.3</td>
<td>4.5</td>
<td>3.75</td>
<td>3.4</td>
</tr>
<tr>
<td>f (MHz)</td>
<td>81</td>
<td>108</td>
<td>139</td>
<td>190</td>
<td>225</td>
<td>265</td>
<td>291</td>
</tr>
<tr>
<td>I (mA)</td>
<td>21.4</td>
<td>30.1</td>
<td>44.1</td>
<td>58.1</td>
<td>73</td>
<td>87.8</td>
<td>107</td>
</tr>
<tr>
<td>P (mW)</td>
<td>15</td>
<td>24.1</td>
<td>36.6</td>
<td>58.1</td>
<td>80.2</td>
<td>105</td>
<td>139</td>
</tr>
</tbody>
</table>

Table A.9: Measurement Results for Chip #3 @ 80°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)</td>
<td>12.1</td>
<td>9.0</td>
<td>7.25</td>
<td>5.5</td>
<td>4.7</td>
<td>3.9</td>
<td>3.5</td>
</tr>
<tr>
<td>f (MHz)</td>
<td>82</td>
<td>111</td>
<td>137</td>
<td>183</td>
<td>212</td>
<td>256</td>
<td>284</td>
</tr>
<tr>
<td>I (mA)</td>
<td>21.4</td>
<td>30.9</td>
<td>43.7</td>
<td>56.6</td>
<td>73</td>
<td>86.2</td>
<td>105</td>
</tr>
<tr>
<td>P (mW)</td>
<td>15</td>
<td>24.7</td>
<td>39.3</td>
<td>56.6</td>
<td>80</td>
<td>103</td>
<td>136</td>
</tr>
</tbody>
</table>
### Table A.10: Measurement Results for Chip #4 @ 0°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)</td>
<td>12</td>
<td>8.48</td>
<td>6.9</td>
<td>5.41</td>
<td>4.63</td>
<td>3.85</td>
<td>3.55</td>
</tr>
<tr>
<td>f (MHz)</td>
<td>83</td>
<td>118</td>
<td>151</td>
<td>184</td>
<td>221</td>
<td>259</td>
<td>280</td>
</tr>
<tr>
<td>I (mA)</td>
<td>22.2</td>
<td>33.26</td>
<td>45.8</td>
<td>58.45</td>
<td>73.2</td>
<td>88.14</td>
<td>102.7</td>
</tr>
<tr>
<td>P (mW)</td>
<td>15.5</td>
<td>26.61</td>
<td>41.2</td>
<td>58.45</td>
<td>80.5</td>
<td>105.7</td>
<td>133.5</td>
</tr>
</tbody>
</table>

### Table A.11: Measurement Results for Chip #4 @ 25°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)</td>
<td>12.11</td>
<td>9.26</td>
<td>7.4</td>
<td>5.54</td>
<td>4.73</td>
<td>3.93</td>
<td>3.65</td>
</tr>
<tr>
<td>f (MHz)</td>
<td>82.51</td>
<td>107.9</td>
<td>135</td>
<td>180</td>
<td>211</td>
<td>253</td>
<td>273</td>
</tr>
<tr>
<td>I (mA)</td>
<td>22.1</td>
<td>30.38</td>
<td>44.2</td>
<td>58.1</td>
<td>72.7</td>
<td>87.4</td>
<td>100.7</td>
</tr>
<tr>
<td>P (mW)</td>
<td>15.52</td>
<td>24.3</td>
<td>39.8</td>
<td>58.1</td>
<td>80</td>
<td>104.8</td>
<td>131</td>
</tr>
</tbody>
</table>

### Table A.12: Measurement Results for Chip #4 @ 80°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)</td>
<td>12.11</td>
<td>9.34</td>
<td>7.45</td>
<td>5.58</td>
<td>4.8</td>
<td>4.02</td>
<td>3.73</td>
</tr>
<tr>
<td>f (MHz)</td>
<td>82.51</td>
<td>107</td>
<td>134</td>
<td>179</td>
<td>208</td>
<td>248</td>
<td>268</td>
</tr>
<tr>
<td>I (mA)</td>
<td>21.8</td>
<td>30.3</td>
<td>44.1</td>
<td>57.8</td>
<td>72.3</td>
<td>86.9</td>
<td>98.5</td>
</tr>
<tr>
<td>P (mW)</td>
<td>15.2</td>
<td>24.2</td>
<td>39.6</td>
<td>57.8</td>
<td>79.5</td>
<td>104.3</td>
<td>128</td>
</tr>
</tbody>
</table>
### Table A.13: Measurement Results for Chip #5 @ 0°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)</td>
<td>12.1</td>
<td>8.7</td>
<td>7</td>
<td>5.4</td>
<td>4.55</td>
<td>3.7</td>
<td>3.35</td>
</tr>
<tr>
<td>f (MHz)</td>
<td>82</td>
<td>114</td>
<td>141</td>
<td>184</td>
<td>219</td>
<td>268</td>
<td>298</td>
</tr>
<tr>
<td>I (mA)</td>
<td>22.5</td>
<td>31.9</td>
<td>44.9</td>
<td>57.9</td>
<td>74.4</td>
<td>90.9</td>
<td>108</td>
</tr>
<tr>
<td>P (mW)</td>
<td>15.7</td>
<td>25.5</td>
<td>40.4</td>
<td>57.9</td>
<td>81.8</td>
<td>109</td>
<td>141</td>
</tr>
</tbody>
</table>

### Table A.14: Measurement Results for Chip #5 @ 25°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)</td>
<td>12.2</td>
<td>9</td>
<td>7.25</td>
<td>5.45</td>
<td>4.6</td>
<td>3.76</td>
<td>3.4</td>
</tr>
<tr>
<td>f (MHz)</td>
<td>81</td>
<td>110</td>
<td>138</td>
<td>183</td>
<td>217</td>
<td>268</td>
<td>291</td>
</tr>
<tr>
<td>I (mA)</td>
<td>22.1</td>
<td>31</td>
<td>44.5</td>
<td>57.8</td>
<td>74</td>
<td>90.2</td>
<td>106</td>
</tr>
<tr>
<td>P (mW)</td>
<td>15.5</td>
<td>24.9</td>
<td>39.9</td>
<td>57.8</td>
<td>81.4</td>
<td>108.2</td>
<td>137.8</td>
</tr>
</tbody>
</table>

### Table A.15: Measurement Results for Chip #5 @ 80°C.

<table>
<thead>
<tr>
<th>Supply voltage</th>
<th>0.7 V</th>
<th>0.8 V</th>
<th>0.9 V</th>
<th>1.0 V</th>
<th>1.1 V</th>
<th>1.2 V</th>
<th>1.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>t (ns)</td>
<td>12.2</td>
<td>9.2</td>
<td>7.4</td>
<td>5.6</td>
<td>4.75</td>
<td>3.85</td>
<td>3.5</td>
</tr>
<tr>
<td>f (MHz)</td>
<td>81</td>
<td>108</td>
<td>135</td>
<td>177</td>
<td>211</td>
<td>259</td>
<td>287</td>
</tr>
<tr>
<td>I (mA)</td>
<td>22.1</td>
<td>31.1</td>
<td>44.5</td>
<td>57.8</td>
<td>73.3</td>
<td>90</td>
<td>104.5</td>
</tr>
<tr>
<td>P (mW)</td>
<td>15.5</td>
<td>24.9</td>
<td>40</td>
<td>57.8</td>
<td>81.0</td>
<td>108</td>
<td>135</td>
</tr>
</tbody>
</table>

157
Efficient Architectures for SMC Resampling

B.1 Introduction

The most challenging block in the SMC receiver is the resampling. In fact, a major problem with the SMC method is that the samples degenerate quickly. In other words, all the particles except for a very few are assigned negligible weights. This occurs due to the increase of the variance of the importance weights over time which has a negative effect on the accuracy of the algorithm ([100], [107]). The degeneracy implies that the performance of the particle filter will deteriorate. Degeneracy, however, can be reduced by using good importance sampling functions and resampling [108], [107], [109]. In other words, resampling converts the properly weighted sample set \( \{E^{(j)}_t, \omega^{(j)}_t\}_{j=1}^J \) to the updated set \( \{\tilde{E}^{(i)}_t, N_j\}_{j=1}^J \), where \( \sum_j N_j = J \), and \( N_j = 0 \) for some samples.

Resampling is a sequential computation in a fully parallel particle filtering algorithm. The challenge in its implementation comes from the fact that there is no concurrency between consecutive iterations as the samples of the new iterations depend on the previous ones. Moreover, if the resampler is intended to be implemented using parallel processing elements, it will generate extensive communication between processing elements (PEs). Finally, the communication between PEs depends directly on the distribution of the weights in the PEs and is not deterministic, thus can’t be calculated a priori. Therefore, the resampler should be designed for the worst case scenario, which requires more resources and limits the architecture speed.

To appreciate the challenges in the parallel implementation of resampling consider the example in Fig. B.1 where three PEs are involved, and the total number of samples is \( J = 60 \). After the sampling and weight calculation, replication factors are calculated in parallel. The total partial weights of each block along with an associated number of replications after resampling are shown in the figure. In order to proceed in a fully parallel way, all PEs should have an equal number of samples. Therefore, extra
particles from PE I should be routed to the others to make the sample loads balanced (sample routing). In general, with many PEs, sample routing and writing into the memory are non-deterministic and complex. In the following, both centralized and our proposed distributed implementation of resampling are discussed.

\section*{B.2 Centralized Implementation}

In the centralized implementation, the resampling is performed in a sequential manner due to the dependencies between samples and the way they are written into the memory. Therefore, it takes $J$ clock cycles to calculate the replication factors and in the worst case it takes $J$ cycles to write them all into the memory, where $J$ is the total number of samples. To understand the worst case scenario, consider the case where all the weights except the last one is zero. Therefore, the last sample has a weight equal to $J$. Writing this sample into the memory in order to get the next sampling done in parallel, takes $J$ cycles. Therefore, the resampling, in the worst case, takes $2J$ cycles.

One of the most popular centralized resampling methods is \textit{sequential importance sampling} (SIR). In this method, samples are drawn uniformly from the discrete set \{$\mathcal{E}_t^{(j)}\}_{j=1}^J$ with probabilities \{$\omega_t^{(j)}\}_{j=1}^J$. SIR can be implemented in $O(J)$ operations ([110]) instead of original $O(J\log J)$ operations in [111]. The variance of SIR can be
further reduced using some techniques [112]. However, one could do even better by employing Stratified sampling [113], [114]. The idea is to generate \( J \) points equally spaced in the interval \([0 : 1]\); and to set the number of offspring \( N_j \) for each particle to be equal to the number of points lying between the partial sums of weights \( q_{j-1} \) and \( q_j \), where \( q_j = \sum_{t=1}^{j} \omega_t^{(j)} \). This scheme can still be implemented in \( O(J) \) operations and has the minimum variance one can achieve in the class of unbiased schemes [115].

Standard algorithms used for centralized random resampling are different variants of stratified sampling such as residual resampling (RR) [116], branching corrections [117] and systematic resampling (SR) [93]. Systematic resampling is the most commonly used since it is the fastest resampling algorithm.

### B.3 Distributed Implementation

In the distributed implementation of resampling, the processing is broken down between a small number of PEs. In fact, two major steps should be done. The replication factor computation, and the sample routing between PEs. The replication factor calculation is shown to be parallelisable in an effective way [104]. However, there is no efficient way, presented in the literature, for dealing with the sample routing. In [94], two methods are proposed for sample routing, which are proportional allocation (RPA) and non-proportional allocation (RNA) schemes, where the sample space is divided into \( K \) disjoint strata and each stratum corresponds to a PE. The idea of proportional allocation means that more samples are drawn from the PE with larger weights. Two routing schemes based on a central unit controller are also considered. However, RPA is memory intensive and results in a non-deterministic routing scheme. Moreover, it requires a point-to-point network between PEs, which leads to a very complex central control unit to handle the routing. On the other hand, RNA has its own drawbacks including the performance loss, and being non-scalable to higher levels of parallelism in the system.

In our proposed scheme, a distributed implementation is proposed where the replication factor is assumed to be calculated according to [104] and a few small independent memory blocks are used along with a central control unit to handle the particle routing. Moreover, the routing scheme is fixed and its duration does not depend on the distribution of the weights in the PEs. Thus resampling is fully parallelized and can be overlapped with the sampling step of the next time slot.
B.4 Distributed Resampling Scheme

In total $J$ samples should be processed in each time interval to compute the target distribution. Considering $K$ parallel PEs in the architecture, there are $F = J/K$ samples in each PE. After the sampling and weight calculation steps, the samples and their associated weights are ready for the resampling. We assume that using the distributed algorithm proposed in [104], the replication factors are calculated in parallel before our scheme is applied. The next step is the routing scheme that balances the number of samples in the PEs.

There are two possible ways to implement the routing scheme. The first one is to use a $K$-port $J$-word memory, which enables PEs to write into the memory simultaneously. Another alternative is to use $K$ single-port $F$-word memories. The former option is too complex and expensive to implement using the current ASIC design methodologies while high-speed embedded single-port memories are very common in VLSI implementations. Moreover, as we will show, the main advantages of single-port memories in our architecture is that the resampling of the current time slot and sampling step of the next time slot can be pipelined. This results in a more efficient and higher speed architecture. In this section, we consider the distributed VLSI implementation of the resampling using $K$ independent $F$-word memory blocks represented by $\{Q_1, Q_2, ..., Q_K\}$. Our proposed distributed scheme is composed of three major steps, offset passing, access list derivation, and scheduling. All these steps will be addressed in detail in the sequel.

B.4.1 Offset Passing

Let $W_t^{(i)}$ denotes the summation of the weights of all the samples in the $i$-th PE (see Fig. B.1). These summations can be calculated during the weight calculation step. In a parallel implementation of the resampling, all PEs should be able to write into the memory simultaneously. In order to make that happen, each PE needs to know the memory block and the correct address that it should write into. In other words, each PE should be aware of the address offset caused by the other PEs. Knowing the offsets, it is guaranteed that the write addresses are orthogonal, thus, PEs can work in parallel without any address conflict or overlap. The offset passing idea is shown in Fig. B.2. As it can be seen, the offset passing is performed sequentially, thus, it takes $K$ clock cycles to communicate all the offsets. The first PE calculates
the number of its sample surplus and passes it to the next PE. This process is done all the way to the last PE. Note that \( b_i > 0 \) means that there are some extra samples in the previous PEs that should be handled by the \( i \)-th PE, while \( b_i < 0 \) means that there is a shortage of samples in the previous PEs. On the other side, \( c_i > 0 \) and \( c_i < 0 \) means that there is a surplus/shortage in the subsequent PEs. Notice that by the definition \( b_1 = b_{K+1} = c_1 = c_{K+1} = 0 \). The value of \( b_i \), and \( c_i \) are calculated as follows:

\[
\begin{align*}
    b_i &= W_t^{(i)} - F + b_{i-1} \\
    c_i &= W_t^{(i)} - F + c_{i+1}.
\end{align*}
\]
B Efficient Architectures for SMC Resampling

B.4.2 Access List Derivation

After calculation and passing the offsets to each PE, it potentially knows the list of memories into which it should write. We refer to this list as the “access list”. For the $i$-th PE, knowing $b_i, W_t^{(i)}$, and $c_i$ is enough to calculate the access list. Therefore, once these parameters are available in a PE, it can start deriving its own access list. This means that the access list derivation and offset passing steps can be pipelined, which reduces the number of cycles required to implement the resampling. The access list derivation can be split into two parallel sections that can be run simultaneously. We call them pre-section and post-section cores. For the $i$-th PE, the pre-section core and post-section core calculate the list and the number of accesses of the $i$-th PE to \(\{Q_1, \ldots, Q_{i-1}\}\), and \(\{Q_{i+1}, \ldots, Q_K\}\), respectively. The access to $Q_i$ can be calculated using the simple modulo arithmetic defined by (B.3).

It takes $K$ cycles to derive the access list for each PE, which results in $K^2$ cycles for all the PEs. However, using our proposed pipelined structure, it takes $2K$ cycles to derive all the access lists. This pipelined architecture is composed of the pre-section and post-section cores, which are shown in Fig. B.3, and Fig. B.5, respectively, where $a_i = W_t^{(i)}$. In these figures, $n_{i,j}^b$, and $n_{i,j}^c$ denote the number of accesses of the $i$-th PE to memory block $Q_j$ calculated by the pre-section, and post-section cores, respectively. Note that according to the architecture, $n_{i,j}^b = 0$ for $i > j$, and $n_{i,j}^c = 0$ for $i < j$. Moreover, as shown in the above figures, the pre-section core is run for PE

![Diagram](image_url)
Figure B.5: Post-section core for access list derivation.

Figure B.6: An example of the pre-section core for access list derivation.

1, ..., K − 1 while the post-section core is run for PE 2, ..., K. The detailed function of the i-th processing element is also shown in Fig. B.4. The post-section core is implemented using the same structure as the pre-section core. Since by definition $n^b_{i,j} = 0$, $\forall j$ when $b_i > 0$, and $n^{c}_{i,j} = 0$, $\forall j$ when $c_i > 0$, the pre-section and post-section cores are inactive for $b_i > 0$ and $c_i < 0$, respectively. The access of i-th PE to
memory block $Q_i$ is determined by:

$$n_{i,i} = \begin{cases} 
\min(\max(F - b_i, 0), a_i) & \text{if } b_i \geq 0 \\
\min(\max(a_i + b_i, 0), F) & \text{if } b_i < 0.
\end{cases}$$

(B.3)

As mentioned, offset passing and access list derivation can be pipelined. For instance, after passing its offset, PE 1 can start deriving its access list during which, PE 2 can start calculating its offset and so on.

Fig. B.6 illustrates the concept of the offset passing, access list derivation, and scheduling for pre-section core. There are four PEs each with a different number of particles. The first step is the offset passing after which, the data is ready for the access list derivation. For instance, PE III has to write three particles on the left-side PEs, and 2 particles on the right-side PEs. The access list derivation has a pipelined architecture. In Fig. B.6, the detailed snapshot of this pipelined implementation is shown where the objective is to find the access of the PE III to the previous PEs. As seen, the input is $-b_3 = 3$, $M = 5$, and $W_t^{(3)} = 11$. Based on our proposed architecture, for each PE $i$, the first $i - 1$ values are valid. Therefore, as shown, 0, and 3, were added to the access list matrix $A$ as $n_{3,1}^h$, and $n_{3,2}^h$, respectively.

**B.4.3 Scheduling**

The final step for the distributed resampling is scheduling based on the derived access list. It simply determines the routing scheme using which the samples should be routed to the memory blocks. Mathematically, the result of the access list derivation is a matrix:

$$A_{K \times K} = \begin{pmatrix} 
n_{1,1} & n_{1,2} & \cdots & n_{1,K} \\
n_{2,1} & n_{2,2} & \cdots & n_{2,K} \\
\vdots & \vdots & \ddots & \vdots \\
n_{K,1} & \cdots & \cdots & n_{K,K}
\end{pmatrix}$$

(B.4)

where $n_{i,j}$ shows the number of times that the $i$-th PE should write into the memory block $Q_j$, and is calculated as

$$n_{i,j} = n_{i,j}^h + n_{i,j}^c$$

for $i \neq j$. (B.5)

Note that $\sum_{j=1}^{K} n_{i,j} = W_t^{(i)}$, and $\sum_{i=1}^{K} n_{i,j} = F$. These equations imply that the
summation of each column of the matrix $\mathcal{A}$ is $F$, which is equal to the number of parallel writes into the memory. In other words, using this matrix, which is calculated by the access list derivation, all the writes into the memories are done simultaneously in $F$ cycles without any address conflict. In other words, at each cycle, one element of each column is chosen, which is equivalent to one write to the corresponding memory block at each cycle. This is an important feature of our architecture, which makes the resampling independent of the distribution of the sample weights in the PEs. Another nice feature of the architecture is that writing into the memory and sampling in the next time slot can be pipelined (Fig. B.7(b)). This means that as soon as the first write into the memory block is done, all PEs can start the sampling for the next time slot, as writing into the memory blocks are done simultaneously. The timing of the overall particle filtering algorithm using our distributed resampling scheme is shown in Fig. B.7(b), where $l_s$ and $l_w$ are the latencies for the sampling and writing sections, respectively. Compared to the sequential scenario, (Fig. B.7(a)), the execution time is much lower.
B. Efficient Architectures for SMC Resampling

Table B.1: Comparison of Resampling Schemes with \( J \) Samples and \( K \) PEs.

<table>
<thead>
<tr>
<th></th>
<th>Sequential</th>
<th>RPA [94]</th>
<th>RNA [94]</th>
<th>This work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory (Words)</td>
<td>( 2J )</td>
<td>( 3J + KJ )</td>
<td>( 4J )</td>
<td>( 2J + K^2 )</td>
</tr>
<tr>
<td>Execution time</td>
<td>( l_s + \frac{l_w}{K} + 2J )</td>
<td>( l_s + \frac{l_w}{K} + K + T_P )</td>
<td>( l_s + \frac{l_w}{K} + K + T_J )</td>
<td>( l_s + \frac{l_w}{K} + 4K + l_w )</td>
</tr>
<tr>
<td>Pipelineable</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Routing time</td>
<td>Not Applicable</td>
<td>Variable</td>
<td>Variable</td>
<td>Fixed</td>
</tr>
<tr>
<td>Scalable</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Perf. Loss</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

Table B.2: Memory Usage Breakdown for Parallel Implementation of Resampling.

<table>
<thead>
<tr>
<th></th>
<th>Sequential</th>
<th>RPA</th>
<th>RNA</th>
<th>This work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Samples (Words)</td>
<td>( J )</td>
<td>( J )</td>
<td>( J )</td>
<td>( J )</td>
</tr>
<tr>
<td>Weights</td>
<td>( J )</td>
<td>( J )</td>
<td>( J )</td>
<td>( J )</td>
</tr>
<tr>
<td>Replication factors</td>
<td>-</td>
<td>( J )</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Routing</td>
<td>-</td>
<td>( K^2F )</td>
<td>( 2KF )</td>
<td>( K^2 )</td>
</tr>
</tbody>
</table>

B.5 Performance Analysis And Simulation Results

Table B.1 shows the detailed comparison of our proposed scheme for the parallel implementation of the resampling with other alternatives. The results can be summarized as follows.

- **Memory usage**: A summary of the memory usage of all the approaches is shown in Table B.2. As shown, the memory usage for the sequential implementation determines the minimum achievable value. Our scheme requires an extra \( K^2 \)-word memory to save access list entries in (B.4). Therefore, it is the second best in the list.

- **Execution time**: In terms of the execution time, our scheme is the best and the sequential scheme is the worst in the list. The parameters \( l_s, l_w \) are negligible. However, \( T_P \), and \( T_N \) show the required variable times for the sample routing in RPA and RNA, respectively. In fact, these values can be very high for systems with a high level of parallelism while in our scheme it is replaced with a deterministic fixed term of \( 3K \), which is independent of the distribution of the weights.
Figure B.8: Performance comparison of various resampling schemes.

- **Pipelineable**: Determines whether resampling can be pipelined with the next sampling section or not. The proposed scheme in this work and RPA are pipelineable.

In brief, our proposed architecture has a fixed routing time and is scalable while maintaining the performance of the sequential method. Thus, our scheme outperforms other alternatives in terms of all the above issues at the cost of a minor increase in the control logic. For instance, for $J = 1024$, $K = 16$, and $F = 64$, the memory requirement and execution time of our scheme is at most $1/2$ and $2/3$ of the RPA and RNA, respectively. It is worth noting that even RPA and RNA need a control logic block for the sample routing.

Fig. B.8 shows the simulation results for different resampling schemes as they are applied to the SSPA framework with 64-QAM constellation scheme and IBO = 10 dB. The main body of the SMC core is considered to be the same for all. The figure shows that the performance of the various schemes closely agree. As was expected, the performance of the sequential scheme is the best. In terms of the BER performance, this work is very close to the RNA. However, considering the memory requirements, execution time, and scalability, our scheme outperforms all the other alternatives.
Moreover, the comparison between the execution time versus the number of PEs for both RNA and our proposed scheme is also shown in Fig. B.9. The results are for $J = 100$, $J = 500$, and $J = 1000$. Moreover, we assumed that the required time for the particle routing, $T_N$, for RNA is fixed, which might not be valid in general. Our proposed architecture has a lower execution time compared to the RNA, specially for lower levels of parallelism.

### B.6 Summary

A scalable, high performance, low memory usage architecture was proposed for the parallel implementation of resampling. The proposed scheme has an execution time independent of the distribution of the weights. It removes the need for a point-to-point network between PEs and has a simple central control unit. In our scheme, particle filtering is implemented using a fully parallel architecture. Moreover, by pipelining the resampling and sampling steps, a very high performance VLSI architecture was developed.
References


References


References


sampling for High-Speed Particle Filtering,” Proc. IEEE Int. Symp. Circuits

[102] ——, “VLSI architecture of a wireless channel estimator using Sequential
Monte Carlo methods,” IEEE Int. Workshop on Signal Proc. Advances in Wire-


filters,” IEEE International Conference on Acoustics, Speech, and Signal

[105] M. Shabany and P. G. Gulak, “VLSI implementation of Sequential Monte Carlo

[106] Y. Nagata, “Linear amplification technique for digital mobile communications,”
In proceedings of VTC’89, pp. 159–164, 1989.

[107] A. Kong, J. S. Liu, and W. H. Wong, “Sequential imputations and Bayesian
2002.

[108] G. Kitagawa, “Monte Carlo filter and smoother for non-Gaussian nonlinear
state space models,” Journal of Computational and Graphical Statistics, vol. 5,
pp. 1–25, 1996.

importance sampling and resampling,” in Sequential Monte Carlo Methods in


[113] G. Kitagawa, “Monte Carlo filter and smoother for non-Gaussian nonlinear
state space models,” Journal of Computational and Graphical Statistics, vol. 5,
pp. 1–25, 1996.

