

بسم الله الرحمن الرحيم





شبكة المعلومات الجامعية التوثيق الالكتروني والميكرو فيلم



جامعة عين شمس

التوثيق الإلكتروني والميكرو فيلم

قسم

نقسم بالله العظيم أن المادة التي تم توثيقها وتسجيلها
على هذه الأقراص المدمجة قد أعدت دون أية تغييرات



يجب أن

تحتفظ هذه الأقراص المدمجة بعيدا عن الغبار





IMPROVING CRYPTANALYSIS OF LATTICE-BASED CRYPTOGRAPHY USING GPU

Presented By

Mohamed Sidi Mohamed Esseissah

Supervised By

Assoc. Prof. Sameh S. Daoud

Assoc. Prof. Hatem M. Bahig

Dr. Ashraf M. Kamal

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENT FOR THE **Ph.D.** DEGREE
(COMPUTER SCIENCE)

SUBMITTED TO
DEPARTEMENT OF MATHAMETICS
FACULTY OF SCIENCE
AIN SHAMS UNIVERSITY
CAIRO, EGYPT

© Copyright by Mohamed Sidi Mohamed Esseissah, 2021

AIN SHAMS UNIVERSITY

Author: **Mohamed Sidi Mohamed Esseissah**

Title: **Improving Cryptanalysis of Lattice-Based
Cryptography Using GPU**

Division: **Computer Science**

Department: **Department of Mathematics**

Faculty: **Faculty of Science**

Degree: **Ph.D.**

Year: **2021**

Contents

Acknowledgements	xi
List of Abbreviations	xiii
List of Abbreviations	xiv
Abstract	xv
Summary	xvii
1 Lattices	1
1.1 An overview of Vector Spaces	2
1.2 Lattices	3
1.2.1 Gram-Schmidt Orthogonalization	7
1.2.2 Lattice Determinant	8
1.2.3 Computational Problems in Lattices	12
1.2.4 Successive Minima	15
1.2.5 The Hermite and Minkowski Upper Bounds	17
1.3 Approximation Algorithms for Solving SVP and CVP	18
1.3.1 Gaussian Lattice Reduction in Two Dimensions	19
1.3.2 Lattice Basis Reduction	23
1.3.3 LLL Algorithm	27
1.3.4 BKZ Algorithm	32

2	GPU	37
2.1	Computer Architecture	38
2.2	GPU Architecture	40
2.2.1	SMs and SPs	41
2.2.2	Memories	42
2.2.2.1	Registers	43
2.2.2.2	Shared Memory	44
2.2.2.3	Global Memory	46
2.2.2.4	Constant memory	47
2.2.2.5	Texture Memory	47
2.3	CUDA Programming Model	47
2.3.1	Handling Threads	51
2.3.2	CUDA Kernel	55
2.3.3	Warps	58
2.3.4	Occupancy	64
2.3.5	Global Memory Access Patterns	65
2.3.6	Warp Divergence	68
2.3.7	Improving performance using shared memory	74
3	Lattice-based cryptography	75
3.1	Introduction to Cryptosystem	75
3.2	NTRU Cryptosystem	78
3.2.1	Basic Notations and Definitions	79
3.2.2	NTRU Key Generation	79
3.2.3	NTRU Encryption Algorithm	80
3.2.4	NTRU Decryption Algorithm	80
3.2.5	Security of NTRU Cryptosystem	82
3.2.5.1	Brute Force Attack	83
3.2.5.2	Lattice-Based Attack	84
3.3	Regev Cryptosystem	87
3.3.1	Key Generation	87
3.3.2	Encryption Algorithm	88

3.3.3	Decryption Algorithm	88
3.3.4	Security of Regev Cryptosystem	90
4	Improving BDD Enumeration for LWE Problem Using GPU	95
4.1	Introduction	96
4.2	BDD Approach for LWE	97
4.3	Parallel Algorithms for BDD	101
4.4	The Proposed GPU-Based Enumeration Algorithm	102
4.4.1	Main Idea	102
4.4.2	Improved BDD Enumeration Algorithm	104
4.4.3	Generating GPU-Points	106
4.4.4	The Proposed Parallel Algorithm	108
4.5	Experimental Results	111
4.5.1	Platform Specification and Data Set	112
4.5.2	Implementation	113
4.5.3	Performance Analysis	115
5	Improving Shortest Vector Enumeration using GPUs	127
5.1	Introduction	128
5.2	Classical Enumeration Algorithm	130
5.3	Pruned Enumeration Algorithm	132
5.4	Extreme Pruning Enumeration	134
5.5	Parallel Algorithms for SVP	136
5.6	The Proposed GPU-based Pruned Enumeration	138
5.6.1	Pre-Processing Strategies	140
5.6.1.1	Strategy for Selecting Better Lattice Basis	141
5.6.1.2	Strategy for Determining Level α	141
5.6.2	Strategy for Improving Uses of GPU	143
5.7	Experimental Results	146
5.7.1	Platform specification and data set	146
5.7.2	Implementation of Pruned Enumeration	148

5.7.3 Implementation of Classical Enumeration 154

5.7.4 GPU Performance Analysis 154

6 Conclusion and Future Work 157

A Kernel Code for LWE 169

List of Tables

1.1	Comparing the three main families of SVP and CVP solvers	20
2.1	Access Time in Term of Memory Type	46
2.2	Resource Limitations Relating to Compute Capability . . .	56
2.3	The types of function qualifiers in CUDA	57
4.1	Average execution time (in seconds) and speedup for implementing Algorithm 4.2 Lindner-Peikert (sequential) using a single CPU and Algorithm 4.5 using a single GPU and two GPUs to solve the LWE problem	121
4.2	Average execution time (in seconds) and speedup for implementing Algorithm 4.2 pruned-enumeration using a single CPU and Algorithm 4.5 pruned-enumeration using single GPU and two GPUs to solve the LWE problem	122
4.3	Performance analysis of dimension $n = 50$ with $m = 90$. .	123
4.4	Performance analysis of dimension $n = 60$ with $m = 120$.	123
4.5	Performance analysis of dimension $n = 70$ with $m = 140$.	124
4.6	Performance analysis of dimension $n = 80$ with $m = 150$.	124
4.7	The effect of parameter α on the performance for dimension $n = 70$ with $m = 140$	125
4.8	The effect of parameter α on the performance for dimension $n = 80$ with $m = 150$	125
5.1	An example of selecting level α on which <i>GPU-points</i> are generated for dimension $n = 100$	150

5.2	An example of selecting level α on which <i>GPU-points</i> are generated for dimension $n = 90$	151
5.3	Average execution time (in seconds) and speedup for our implementations and Hermans et al.'s implementation [38]	152
5.4	Average execution time (in seconds) and speedup of our improvements using a single GPU and two GPUs	153
5.5	GPU performance analysis with strategy of Subsec. 5.6.1.2 for dimension 90	156
5.6	GPU performance analysis with strategies of Subsec. 5.6.1.2 and 5.6.2 for dimension 90	156

List of Figures

1.1	2-Dimensional Lattice	5
1.2	2-Dimensional Lattice	6
1.3	Successive minima in $\mathbb{Z}^2$, $\lambda_1 = \ b_1\ $, $\lambda_2 = \ b_2\ $	16
1.4	b_2^* is the orthogonal projection of b_2 onto the orthogonal complement of b_1	21
2.1	GPU and CPU Interconnection via PCI-E	41
2.2	Inside an SM	43
2.3	Register per SM	45
2.4	Grid and Threads Block Hierarchy	52
2.5	The logical perspective versus hardware perspective of CUDA programming	53
2.6	The logical perspective versus hardware perspective of warps in a threads block	60
2.7	Switching between States: Cycle 1	62
2.8	Switching between States: Cycle 2	63
2.9	Switching between States: Cycle 3	64
2.10	Access DRAM through L2 Cache	66
2.11	Aligned and coalesced memory accesses	67
2.12	Reduce an array using GPU	71
2.13	Neighbored pair approach	72
2.14	Interleaved pair approach	73
4.1	Our parallel Enumeration	103

5.1	Average execution time in seconds for Hermans et al. [38] and our improvements	149
5.2	Average execution time in seconds for our implementations using 1 GPU and 2 GPUs and Correia's implementation [19] using 1 core and 2 CPU cores of 32 threads	155

Acknowledgements

First of all, I would like to express my gratitude and thanks to ***my mother*** and ***my father***, who have been very supportive for me not only during my doctorate preparation but also throughout all my life. I would not have been able to pursue my Ph.D degree or complete this dissertation without their support and encouragement. May God bless my father's soul, forgive him, and reward him the highest place in paradise. Secondly, I am very thankful to ***my wife*** for her continuous support to complete my study. I would like to express my sincere gratitude to Dr. ***Hatem*** and Dr. ***Ashraf*** for their continuous support, valuable discussion, and supervision, I really appreciate their great effort for helping me to achieve my Ph.D degree. In addition, I am thankful to Dr. ***Sameh*** for his kind help and valuable discussion.