國立交通大學

資訊工程學系

博士論文

統合性的最佳化理論與架構在電子設計自動化應用之研究

A Unified Optimization Framework for Electronic Design Automation

研究生：余紹銘

指導教授：李毅郎　教授

：李義明　教授

中華民國九十六年十月

統合性的最佳化理論與架構在電子設計自動化應用之研究
A Unified Optimization Framework for Electronic Design Automation

研究生：余紹銘　　Student：Shao-Ming Yu
指導教授：李毅郎　博士　　Advisor：Dr. Yih-Lang Li
李義明　博士　　Advisor：Dr. Yiming Li

國立交通大學
資訊工程學系
博士論文

A Dissertation
Submitted to Department of Computer Science
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy
in

Computer Science

October 2007

Hsinchu, Taiwan

中華民國九十六年十月

統合性的最佳化理論與架構在電子設計自動化應用之研究

學生：余紹銘　　　　　　　　　指導教授：李毅郎 博士
李義明 博士

國立交通大學 資訊工程學系(資訊科學與工程研究所) 博士班

摘　　要

本研究之目的在於建立一個統合性的最佳化理論與架構，並且應用於電子設計自動化之領域。電子產業中有許多商用電腦輔助工具(CAD Tools)，此類工具多著重於模擬實務甚至提供客製化的功能，可應用於當前技術設計。在前瞻的理論發展與技術開發，研究學者往往需要重新撰寫最佳化暨模擬程式，但這過程是非常複雜且耗時。因此如何建立一個最佳化架構，能夠很方便地將研究議題導向成一個最佳化問題，且能整合不同的電腦模擬軟體與最佳化方法來協助前瞻技術的開發是一項重要的課題。

本研究基於上述概念，希望能開發出一套具備高適用性、高性能的統合性最佳化架構。此研究將藉由整合各種不同的生物演化觀點之最佳化技術、數值最佳化方法與C++程式語言之物件導向設計，而建立了一個統合性的最佳化架構，使之可以適用於處理工程最佳化問題，特別是電子資訊產業的問題。在此架構中，問題定義與最佳化理論方法將會是各自獨立的兩個部分，因此使用者只要藉由程式介面定義並撰寫他們的問題，即可使用已建立之最佳化方法來求解問題。同時使用者也有高自由度可以任意添加新的最佳化方法到此架構中，以達到較佳的延伸性。

在不考慮數學上的收斂特性下，本研究也同時提出了一個混和式的最佳化方法。在此混和式的方法中，先藉由生物演化觀點演算法在問題的全域範圍內尋求一個解答，再利用數值最佳化方法將此全域區間的解答做進一步的改善，之後將數值最佳化方法所找出的區域空間內最佳解，重新送回生物演化觀點演算法中繼續求解。在各類電子資訊產業所遭遇的問題都是非常複雜的，而且通常並不能確定是否每個問題都有最佳解，因此對於業界而言只需要取得一個能滿足所有設計條件的解答即可。在本研究所提

出的混和式方法中，藉由數值最佳化方法與生物演化觀點演算法的不斷交替使用，以及引入了各類電子資訊領域中的專業知識，可以針對複雜的電子設計問題，自動尋求出一個符合設計需求的解答。

在本研究中，進一步運用此概念整合不同的電腦輔助工具與自行研究的模擬程式，並且已經成功的應用在一個65奈米的互補式金氧半電晶體(Complementary Metal Oxide Semiconductor；CMOS)的製程參數設計問題上。在此應用中，藉由與一個自行研究開發的半導體隨機摻雜濃度擾動效應分析程式以及一些工程經驗的結合，可以針對一個CMOS電晶體的設計規格，找出一組符合需求的製程參數組，此參數組同時也能抑制電晶體製造中因為隨機摻雜濃度所造成的電特性擾動現象。藉由與實際實驗數據的比對結果，驗證了此方法的準確性與可行性。此概念也同時被應用於半導體參數萃取、超大型積體電路設計以及通訊系統中的天線設計最佳化等問題上，而且也都得到了良好的結果。此統合性最佳化架構之程式碼已經提供在公開的網路上(http://140.113.87.143/ymlab/uof/)。

A Unified Optimization Framework for Electronic Design Automation Application

Student: Shao-Ming Yu

Advisors: Dr. Yih-Lang Li
Dr. Yiming Li

Department of Computer Science
National Chiao Tung University

ABSTRACT

In the modern microelectronics industry, there are some kinds of computer-aided design tools (CAD tools) to assist engineers complete simulation jobs which can verify and estimate the performance of their designs. However, to satisfy the design targets, engineers must base on the simulation result to adjust the design parameters, and again feed the adjusted parameters to retrieve the improved result. Currently, such routine work mostly performed by engineers with expertise. Therefore, a well defined optimization platform can assist engineers to solve problems more efficiently.

This dissertation presents an object-oriented unified optimization framework (UOF) for general problem optimization. Based on biological inspired techniques, numerical deterministic methods, and C++ objective design, the UOF itself has significant potential to perform optimization operations on various problems. The UOF provides basic interfaces to define a general problem and generic solver, enabling these two different research fields to be bridged. The components in the UOF can be divided into problem and solver parts. These two parts work independently, allowing high-level code to be reused, and rapidly adapted to new problems and solvers. Without considering the mathematical convergence property, one hybrid intelligent technique for electronic design automation is also proposed and implemented in the UOF. In the proposed hybrid approach, an evolutionary method, such as genetic algorithm (GA), firstly searches the entire problem space to get a set of roughly estimated solutions. The numerical method, such as Levenberg-Marquardt (LM) method, then performs a local optima search and sets the local optima as the suggested values for the GA to perform further optimizations. The electronic design problems from the industry are very complicated and not always

guaranteed to have an optimal solution. Therefore, the designers or engineers only need to find one suitable solution which can meet all specifications. By integrating with empirical knowledge, the proposed hybrid approach can automatically search solutions to match the specified targets in the electronic design problems.

The purpose of the UOF is to assist the electronic design automation with various CAD tools. One application in 65nm CMOS device fabrication has been investigated. Integration of device and process simulation is implemented to evaluate device performances, where the developed approach enables us to extract optimal recipes which are subject to targeted device specification. Fluctuation of electrical characteristics is simultaneously considered and minimized in the optimization procedure. Compared with realistic fabricated and measured data, this approach can achieve the device characteristics, and can reduce the threshold voltage fluctuation at the same time. Other applications including device model parameter extraction, very large scale integration (VLSI) circuit design and the communication system antenna design are also implemented with the UOF and presented in this dissertation. The results confirm that UOF has excellent flexibility and extensibility to solve these problems successfully. The developed open-source project is available in the public domain (http://140.113.87.143/ymlab/uof/).

誌　謝

本論文得以順利完成，首先要感謝恩師　李毅郎教授給予我最大的自由度，讓我得以進行我感興趣的研究。其次我要感謝恩師　李義明教授多年來的悉心指導；感謝恩師於受業期間對學生論文研究之激勵，思緒慎密之牽引，觀念之啟迪，論文架構之匡正，研究方法之傳授及用字遣辭之推敲斟酌。更銘誌於心的是恩師在為學處世及待人接物之諄諄教誨，使學生在治學方法及處世態度上受益良多，而恩師在學術研究之嚴謹精神、半導體及資訊領域之專業知識與生活處世的積極態度，更足以為學生日後之表率。師恩細長，深切銘心，學生在此謹獻上最誠摯的感謝與敬意。

論文口試期間，承蒙交通大學資訊學院林一平院長、交通大學運輸科技與管理學系卓訓榮教授、元智大學通訊研究中心彭松村主任、清華大學資訊工程系林永隆研發長、清華大學電機資訊學院徐爵民院長及成功大學電機工程學系楊竹星教授撥冗細審，並惠予寶貴意見與殷切指正使本論文更臻完備。

論文進行期間，感謝交通大學電子資訊中心平行與科學計算實驗室所提供的協助，並感謝同窗好友的朝夕相伴，正凱、彥羽、至鴻及益廷的照顧幫忙，怡慧、素雲、大慶、宣銘、惠文、典燁及湯唯的互相砥礪，在此一併致謝。感謝台灣積體電路製造股份有限公司楊富量經理與黃俊仁經理提供研究上的協助。

學生能順利完成博士學業，全靠父母、家人及諸位朋友、同學的支持與忍耐，學生對此由衷感謝，謹在此將本論文獻給關心我的人！

本論文感謝行政院國家科學委員會（計畫編號 NSC-95-2221-E-009-336、NSC-96-2221-E-009-210）、卓越延續計畫（計畫編號 NSC-94-2752-E-009-003-PAE、NSC-95-2752-E-009-003-PAE、NSC-96-2752-E-009-003-PAE）、五年五百億計畫及台灣積體電路製造股份有限公司研究計畫之資助。

余紹銘　謹誌

中華民國九十六年十月 于風城交大

Contents

Abstract v

Acknowledgement ix

List of Tables xv

List of Figures xvii

1 Introduction **1**

- 1.1 Motivation 1
- 1.2 Objectives 2
- 1.3 Outline 5

2 The Conventional Optimization Framework **6**

- 2.1 GALIB 7
- 2.2 DESMO 8
- 2.3 NP-Opt 8

3 The Unified Optimization Framework 12
3.1 The Architecture of the UOF 13
3.1.1 The UOFProblem Class 15
3.1.2 The UOFEvaluator Class 16
3.1.3 The UOFInitializer Class 16
3.1.4 The UOFSolver Class 17
3.1.5 The UOFSolution Class 21
3.1.6 The Working Flow of the UOF 22
3.2 Implementation Examples 23
3.2.1 Simulation-based Problem 23
3.2.2 Genetic Algorithm Solver 24
3.2.3 Particle Swarm Optimization Solver 27
3.2.4 Basic Experiment on Traveling Salesman Problem 29
3.3 The Developed Parallelization Technique 31
3.3.1 Parallelization for Nanoscale Double-Gate MOSFETs Simulation . 34
3.3.2 Parallelization for Large-scale Protein Folding Dynamics 46
3.4 An Open-Source Project for Scientific Visualization 58
3.4.1 Layout and Functionality 59
3.4.2 Architecture and Visualization Procedure 60

3.4.3 Illustration Examples 68

4 Application of UOF in 65nm CMOS Device Fabrication 74

4.1 The Inverse Problems of the Semiconductor Devices 75

4.2 The TCAD Simulation for the Semiconductor Devices 77

4.3 The Optimization Methodology 82

4.4 The Achieved Results and Discussion 87

5 Application of UOF in VLSI Device Model Parameter Extraction 98

5.1 The Equivalent Circuit Model Parameter Extraction Problems 99

5.2 The Hybrid Optimization Methodology 101

5.3 The Empirical Rules for Equivalent Circuit Model Parameter Extraction . . 105

5.4 The Achieved Results and Discussion 108

5.5 Effects of Random Number Generations for Parameter Extraction 112

6 Application of UOF in VLSI Circuit Design 118

6.1 The Integrated-Circuits Design Problem 119

6.2 The Simulation-Based Hybrid Optimization Technique 122

6.3 The Computational Statistics Technique 125

6.3.1 Variables selection 127

6.3.2 Central Composite Design 129

6.3.3 Response Surface Model Construction 130
6.3.4 Optimization of characteristics 134
6.4 The Achieved Results and Discussion 136

7 Application of UOF in Communication System Antenna Design 150

7.1 The Antenna Geometry Design Problem 151
7.2 The Finite Element Method in Electromagnetics 153
7.3 The Optimization Method Using Genetic Algorithm 158
7.3.1 Problem Definition 161
7.3.2 Encoding Method 162
7.3.3 Fitness Evaluation 163
7.3.4 Selection Method 163
7.3.5 Crossover Procedure and Mutation Scheme 164
7.4 The Achieved Results and Discussion 164

8 Conclusions and Future Work 174

References 177

Appendix A

VITA 200

List of Tables

2.1 The found solutions of different methods in solving the minimization problem of Eq. 2.1. 11

3.1 The optimized result of the TSP. 31

3.2 A list of the achieved sequential and parallel time, efficiency, and speedup with respect to different number of nodes. It is performed on an 8-processors PC-based Linux cluster system. 43

3.3 The number of conformations of proteins with the length of L residues in a 2D square lattice model. 49

3.4 The achieved parallel speedup and efficiency of the parallel computing algorithm, where the tested case is a 17-mer with only hydrophobic residues. 53

4.1 A comparison list of the achieved results with respect to the two different extractions. The target specification is adopted from realistic fabricated and measured data. 88

4.2 A list of process recipe and device parameters for the device optimization w/ and w/o considering the fluctuation reduction. 90

5.1 Eight different random number generation schemes and their expressions. . 114

5.2 List of the root-mean-square (RMS) errors of the extracted results compared with the measured data for the four N-MOSFET devices. The device dimension is in μm. The oxide thickness of target devices are 3.36nm and the working temperature is settled at 298.15K. The chaotic random number generator is used in this simulation. 115

6.1 List of the residual of squares in the six response surface models. 140

6.2 List of the range of designing parameters and the optimized parameters for the current mirror amplifier circuit. 144

6.3 List of the specified targets and the extracted results for the current mirror amplifier circuit. 145

List of Figures

1.1 Illustration of a basic flow in the field of electronic design. 3

1.2 The basic contents of the proposed UOF. 5

2.1 The values of $f(x_1, x_2)$ respect to different x_1 and x_2 values. 11

3.1 An illustration of the architecture of UOF. Solid lines indicate inheritance and dashed lines indicate the ownership. 14

3.2 The class hierarchy of the UOFProblem and UOFSolver. The solid lines indicate inheritance. 14

3.3 The working flow and object inner connections in UOF from users' point of view. 22

3.4 The working flow of GetResult in TCADProblem. 25

3.5 The flowchart of TCADEvaluator. 26

3.6 The basic working flow of the developed parallelization approach. 33

3.7 The constructed 32-node PC-based cluster. 34

3.8 An illustration of the cross-section view for the n-type double-gate MOSFET device. 37

3.9 Adaptive finite volume solution algorithm for each decoupled semiconductor device equation. A is the system matrix, Z is the unknown vector, and F is the nonlinear vector. 38

3.10 Parallel domain decomposition for the two-dimensional double-gate MOSFET. The dash-lines indicate the boundary of partition domains. 39

3.11 The number of nodes (and elements) in Log scale versus the refinement levels. 42

3.12 The maximum difference versus the number of nodes. 44

3.13 The parallel speedup and efficiency versus the number of processors. . . . 45

3.14 A computational flowchart of the implemented parallel method. 52

3.15 The distribution of all computed eigenvalues for the sequence 0011101011 ($L = 10$) in (a) a single processor (b) and 32-node PC-based Linux cluster. . 54

3.16 The maximum norm error of the (a) first and (b) the $\mathbf{50^{th}}$ eigenvalues versus the number of iterations. 55

3.17 The eigenvalues of the transition matrix of a 17-mer with only hydrophobic residues. The dimension of the transition matrix is 2155667 by 2155667, and 50 eigenvalues have been computed. 56

3.18 The achieved load balancing versus the number of processors for the tested case of a 17-mer with only hydrophobic residues. 56

3.19 The achieved parallel efficiency versus the matrix size for different number of processors. 57

3.20 The achieved load balancing versus the matrix size for different number of processors. 57

3.21 (a) Visi's logo and (b) its layout, where four tabbed pages, Property, Display, Color, and Info. are designed in Control panel. 61

3.22 An architecture of the developed classes in Visi. 61

3.23 Filter functionalities of Visi and the corresponding implementation classes. 62

3.24 The procedure flow for manipulating Visi to visualize a source data, and the corresponding working flow in *QVTKFrame*. 64

3.25 Flowcharts for VTK data pipeline, and the list of derived classes for each base class which marked with dash-line. 65

3.26 Flowcharts for the developed Visi data pipeline, and the list of derived classes for CBaseActor class. The base class is marked with dash-line, while the function routines are marked with dot-line. 67

3.27 The tertiary structure of a catalytic antibody. (a) The structure data without applying filters and (b) is the structure after applied Glyph and Tube filters. 69

3.28 (a) The potential profile for a 2D MOSFET device, and (b) the zoom-in plot in the channel region of MOSFET device with contour lines and scalar bar. 69

3.29 The carpet plot for 2D potential profile by applying Warp-Scalar filter with different color mapping schemes. 70

3.30 (a) The surface potential plot of a 3D MOSFET device. (b) The transparent plot for potential profile and discrete implant doping atoms. 70

3.31 (a) X-Y and (b) X-Z cut planes of 3D MOSFET potential profile. 72

3.32 The carpet plots for (a) X-Y and (b) X-Z cut planes. 72

4.1 A working flow of the implement methodology for the reverse modeling problem in tuning fabrication parameters. 81

4.2 An illustration of the target I-V curves to be extracted and empirical knowledge. The important sections are pointed out in circles. The inset plot is a cross-section view of the simulated MOSFET with LDD doping profile. . . 85

4.3 An illustration of the initial doping profile for the explored 65 nm N-MOSFET. 88

4.4 The optimized 65 nm N-MOSFET doping profile for the device (a) without and (b) with considering the minimization of the threshold voltage fluctuation. 89

1896

4.5 The optimized doping profiles from the channel surface deep into the substrate. The 1-section is located at the center of device channel (x = 0). . . . 92

4.6 Doping profile from the source to drain along the channel direction 2 nm below the interface between the gate oxide and the silicon substrate. The inset of the figure shows the structure of the optimized 65 nm N-MOSFET. 93

4.7 Plots of band profile for the optimization with and without considering the threshold voltage fluctuation under the on-state; (a) is from the surface to the substrate and (b) is along the channel direction from the source to the drain which is about 2 nm below the channel surface. 94

4.8 Plots of band profile for the optimization with and without considering the threshold voltage fluctuation under the off-state; (a) is from the surface to the substrate and (b) is along the channel direction from the source to the drain which is about 2 nm below the channel surface. 95

4.9 The achieved accuracy of the extracted I-V curves for the explored 65 nm N-MOSFET. 96

4.10 The performance comparisons among three different evolutionary strategies. There are totally 31 process and device parameters to be optimized in the case of the 2D process and device simulations. The total time is about 70 hours on a PC-based Linux cluster with 16 processors. 97

5.1 An illustration of the equivalent circuit model in connecting the device fabrication and circuit design. 100

5.2 An illustration of the proposed hybrid intelligent computational methodology. 102

5.3 The sensitivities examination of the parameters to be extracted in the compact model. 106

5.4 The BSIM-4 extracted (solid-line) and measured (dot-lines) $I_{DS}-V_{DS}$ and $I_{DS}-V_{GS}$ curves of the 90 nm MOSFET (width = 10.0 μm), where V_{BS} = 0 V and V_{GS} varies from 0.4 to 1.4 V, and V_{DS} = 0.1 V and V_{BS} varies is 0 to -1.5 V. 107

5.5 The EKV extracted (solid-line) and measured (dot-lines) $I_{DS}-V_{DS}$ and $I_{DS}-V_{GS}$ curves are of the 0.18 μm MOSFET, where V_{BS} = -0.6V and, V_{GS} migrates from 0.4 to 1.4 V, and V_{DS} = 1.3V and V_{BS} migrates is 0 to -0.9 V. 108

5.6 (a) the score convergence of different extraction methods, and (b)the score convergence behavior w/ and w/o applying NN as a director during the evolutionary process, where the testing is with BSIM-4 model applying to four 0.13 μm MOSFETs. 110

5.7 The score convergence behavior of our proposed method and pure GA for different compact models. 111

5.8 The BSIM4-model extracted (solid-line) and measured (dot-lines) (a) I$_{DS}$-V$_{DS}$ and (b) I$_{DS}$-V$_{GS}$ curves of the MOSFET, where length = 90nm and width = 10μm. We notice that the chaotic random number generator is used in this simulation. 115

5.9 Comparison of the convergence score in GA with different random number generation schemes. In this extraction experiment, GA extracts a single N-MOSFET device. 116

5.10 Comparison of the convergence score in GA with different random number generation schemes. In this extraction experiment, GA extracts four N-MOSFET devices at the same time. 117

6.1 Functional blocks for the proposed simulation-based hybrid optimization approach. 122

6.2 The optimization flow for the hybrid optimization approach. 123

6.3 The explored current mirror amplifier circuit. The transistors MB and MS provide the current source for the differential amplifier. The transistors M1 and M2 form the differential amplifier, and transistors M3∼M6 make up the current mirror. 126

6.4 (a) The residual normal probability plot and (b) the residuals vs. predicted plot for the response $GAIN$. 141

6.5 (a) The residual normal probability plot and (b) the residuals vs. predicted plot for the response $1.0/Sqrt(FT)$. 142

6.6 (a) The residual normal probability plot and (b) the residuals vs. predicted plot for the response $PM^{1.57}$. 143

6.7 (a) The residual normal probability plot and (b) the residuals vs. predicted plot for the response $CMRR^{-2.34}$. 144

6.8 (a) The residual normal probability plot and (b) the residuals vs. predicted plot for the response $Sqrt(SR+)$. 145

6.9 (a) The residual normal probability plot and (b) the residuals vs. predicted plot for the response $1.0/Sqrt(SR-)$. 146

6.10 Scatter plot calculated from the response surface model versus values obtained from circuit simulator for (a) $GAIN$ and (b) $1.0/Sqrt(FT)$. 147

6.11 Scatter plot calculated from the response surface model versus values obtained from circuit simulator for (a) $PM^{1.57}$ and (b) $CMRR^{-2.34}$. 148

6.12 Scatter plot calculated from the response surface model versus values obtained from circuit simulator for (a) $Sqrt(SR+)$ and (b) $1.0/Sqrt(SR-)$. 149

7.1 An illustration of the problem domain and boundary conditions in the examined antenna. The problem domain is surrounded by the edges a, b, c and d; and all edges satisfy the third kind boundary conduction. The black rectangular frame near the upper line is the contact for input excitation. . . 156

7.2 The linear tetrahedral element. The labels for four nodes are local indices in FEM. 157

7.3 A flowchart for the proposed optimization approach in antenna design. In this procedure, after constructing the geometry of the new antenna, the corresponding input file for extern simulator is also generated and passed to the external simulator for further simulation and analysis. In this work, HFSS is applied as the external simulator to evaluate the performance of the new antenna. 159

7.4 The original geometry of examined antenna and the illustration of partitioned segments. The upper line presents the ground plane and the lower rectangle behaves as the radiation element. The black rectangular frame near the upper line is the contact for input excitation. The partitioned segments allow us to perturb the geometry without spatial limitation in a specific region. 165

7.5 The return loss (S11) of the original geometry. The antenna is a dual-band design but has only one clear band at 2.6GHz. The return loss on the frequency 5 GHz still has room for improvement. 166

7.6 An optimized geometry of the examined antenna after our optimization scheme. The geometry of radiation element is obviously tuned by GA to enhance the transmitting and receiving properties of the antenna. 167

7.7 The return loss (S11) of the optimized geometry. One can find that the return losses corresponding to dual operating frequencies are less than 20 dB after the proposed optimization scheme. 168

7.8 The gain pattern for the original antenna at 2.6GHz. The left object is the 3D gain pattern. The right one is the 2D gain pattern. The solid line presents the gain pattern in the xz-plane, and the dashed line is the gain pattern in the xy-plane. 169

7.9 The gain pattern for the original antenna at 4.9GHz. The left object is the 3D gain pattern. The right one is the 2D gain pattern. The solid line presents the gain pattern in the xz-plane, and the dashed line is the gain pattern in the xy-plane. 170

7.10 The gain pattern for the optimized antenna at 2.6GHz. The left object is the 3D gain pattern. The right one is the 2D gain pattern. The solid line presents the gain pattern in the xz-plane, and the dashed line is the gain pattern in the xy-plane. 171

7.11 The gain pattern for the optimized antenna at 4.9GHz. The left object is the 3D gain pattern. The right one is the 2D gain pattern. The solid line presents the gain pattern in the xz-plane, and the dashed line is the gain pattern in the xy-plane. 172

7.12 The fitness score versus the number of generations in the examination. The mutation rate in this examination is 0.01 and single-point cut crossover scheme is adopted. The proposed optimization scheme enhances the antenna property within ten generations in this examination. 173

8.1 The source code could be downloaded from http://140.113.87.143/ymlab/uof/ or http://sourceforge.net/projects/uof/. 176

Chapter 1

Introduction

1.1 Motivation

The modern microelectronics industry involves many optimization jobs that are subject to simulated results from certain simulators, such as device model parameter extraction [1–3], automatic design of integrated circuit (IC) [4], process simulation [5], reverse modeling [6] and antenna optimization [7, 11]. These terms need to feed specific parameters into some commercial simulators to obtain the corresponding simulated results. The engineers adjust the parameters according to the results, and again feed the adjusted parameters to retrieve the improved results. This circular procedure continues until the results match the requirements of the customers or supervisors. Currently, this routine work is typically performed by engineers with expertise. Therefore, a well-defined framework can assist engineers to complete their design works more easily with various optimization techniques.

A major obstacle to designing a general-purpose optimizer in programming language is the difficulty in defining a comprehensive interface for problems and solvers. Appropriate interfaces for problems and solvers produce objects that are heuristically easy to reuse again in a specific framework. Several optimization frameworks, such as GALIB [8], DESMO [9] and NP-Opt [10], have been proposed in the public domain. These packages have certain advantages, but still have room for improvement in terms of the connections between problem and solver for real-world applications, in particularly for the problems of electronic design.

1.2 Objectives

This dissertation implements a C++ unified optimization framework (UOF) for general problems and solvers. Based on the C++ design pattern techniques, the UOF enables users to define the problems and solvers from the fundamental level. The developed UOF is composed of two parts, the problem part and the solver part. The basic interface of problem-related classes contain initialization, evaluation and constraint procedures. The solver-related classes include the solver, solution, termination and information procedures. Separating each part into several classes increases the efficiency of reutilization. Without considering the mathematical convergence property, one hybrid intelligent technique for electronic design automation is also proposed and implemented in the UOF. In the proposed hybrid approach, an evolutionary method, such as genetic algorithm (GA), firstly searches the entire problem space to get a set of roughly estimated solutions. The numerical method, such as Levenberg-Marquardt (LM) method, then performs a local optima search and sets the local optima as the suggested values for the GA to perform further optimizations. Several applications are presented to demonstrate the flexibility and extensibility of UOF.

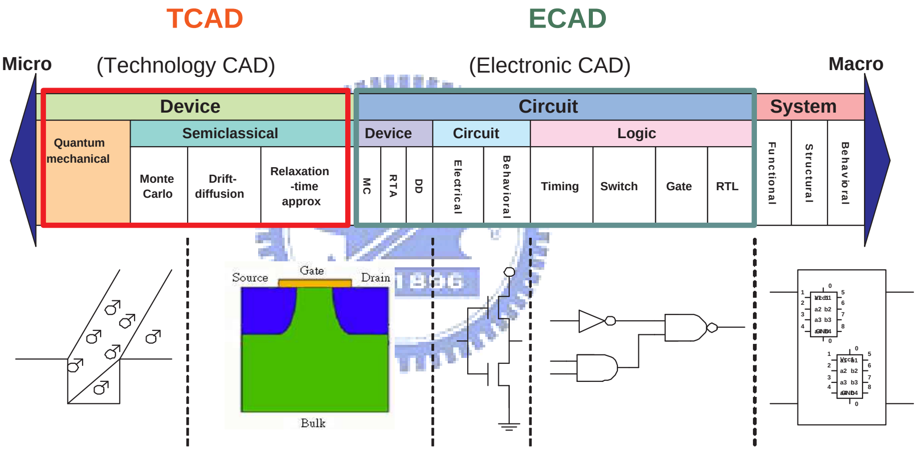


Figure 1.1: Illustration of a basic flow in the field of electronic design.

Figure 1.1 illustrates a basic flow of the electronic design. Toward the end of microelectronics industry, it is related to semiconductor device manufacturing and assisted by technology computer-aided design (CAD) tools. The major topic in the macroscopic side is the circuits design with the electronic CAD tools. The device compact models bridge these two areas, and the parameter extraction is the main problem in this part. These electronic design problems are very complicated and not always guaranteed to have an optimal solution. Therefore, the designers or engineers only need to find one suitable solution which can meet all specifications. The purpose of the UOF is to integrate the hybrid approach with various CAD tools and empirical knowledge to assist the designers or engineers in searching the solutions that can match their specified targets.

The applications from the micro side to macro side which include the reverse doping profile problems, device model parameter extraction, integrated circuit design and the antenna design optimization, as shown in Fig. 1.2, are implemented with the UOF. Now the UOF can directly link to the following CAD tools, ISE TCAD tool, HSPICE and HFSS. The ISE TCAD tool provides a series of process and device simulators for the semiconductor devices. HSPICE is a well-known circuits simulator for ICs design. HFSS is the industry-standard software for 3D electromagnetic field simulation of high-frequency and high-speed components. Most electronic design problems are time-consuming tasks, and one parallel environment is established for the UOF to speedup the optimization procedure. Several applications are implemented with the proposed parallel technique to illustrate the developed environment. Furthermore, one open-source project, Visi is also implemented for high-dimensional engineering and scientific data visualization.

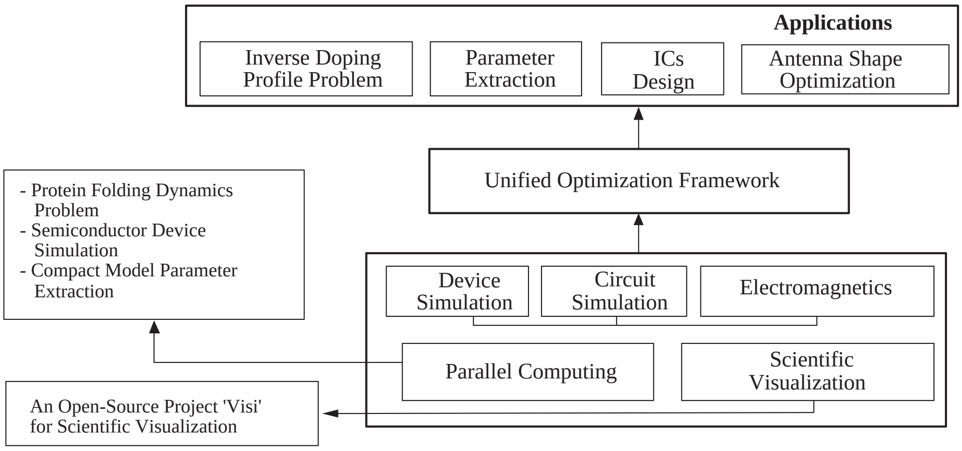

Applications
Inverse Doping Profile Problem
Parameter Extraction
ICs Design
Antenna Shape Optimization
Unified Optimization Framework
- Protein Folding Dynamics Problem
- Semiconductor Device Simulation
- Compact Model Parameter Extraction
Device Simulation
Circuit Simulation
Electromagnetics
Parallel Computing
Scientific Visualization
An Open-Source Project 'Visi' for Scientific Visualization

Figure 1.2: The basic contents of the proposed UOF.

1.3 Outline

This dissertation is organized as follows. The review of conventional optimization frameworks are given in chapter 2. Chapter 3 then introduces the architecture of the UOF. It explains the detailed coding methodologies for the problem and solver, and provides pseudocode for building problems and solvers. The architecture of the developed parallelization technique and one open-source project for scientific visualization are also presented. Chapters 4-7 present the applications of the UOF in the modern microelectronics industry, including 65nm CMOS device fabrication, VLSI device model parameter extraction, VLSI circuit design problem and communication system antenna design problem, respectively. Conclusions are finally drawn in Chapter 8, along with recommendations for future research.

Chapter 2

The Conventional Optimization Framework

Several optimization frameworks, such as GALIB, DESMO and NP-Opt, have been proposed in the public domain. GAlib contains a set of C++ genetic algorithm objects, which was developed at MIT. DESMO is a framework for “Discrete Event Simulation and MOdeling” developed at the University of Hamburg. NP-Opt is an object-oriented framework for optimization based on evolutionary computation techniques. These packages have certain advantages, but still have room for improvement in terms of the connections between problem and solver for real-world applications.

2.1 GALIB

GAlib [8] contains a set of C++ genetic algorithm objects, which was developed at MIT. This library includes tools for using genetic algorithms to do optimization in any C++ program using any representation and genetic operators. The programming interface for the library includes general library information, genetic algorithm, population, scaling, selection, genomes, and data structures. The users must define the representation, genetic operators and objective function for the problem when using GAlib, and users work primarily with two classes: a genome and a genetic algorithm. Each genome instance represents a single solution to the problem. The genetic algorithm object defines how the evolution should take place. The genetic algorithm uses the defined objective function to determine the fitness of each genome. It uses the genome operators and selection/replacement strategies to generate new individuals.

The GAlib is a well-known library for genetic algorithm, and provides basic components for users to define and solve their problems. However, the main drawback of GAlib is that it focuses on GA only and does not support other numerical or evolutionary optimization methods.

2.2 DESMO

A framework for Discrete Event Simulation and MOdeling (DESMO) [9] was developed at the University of Hamburg in the late 1980s. It has been continuously evolved over the last decade. Starting with Modula-2, the framework was ported to various programming languages including Smalltalk and C++. The latest version is DESMO-J which is based on a fully object oriented architecture and is completely implemented in Java. It presented bridges the gap between three different research areas: discrete event simulation, heuristic optimization methods and distributed systems technology. It bases on genetic algorithm to perform optimization works, and the fitness function comes from discrete event simulation. Since DESMO-J is a Java program, features of Java also apply to DESMO-J, and the Java's Remote Method Invocation (RMI) is employed for distributed simulation optimization.

2.3 NP-Opt

NP-Opt [10] is an object-oriented framework for optimization based on evolutionary computation techniques to address NP-hard problems. The purpose of NP-Opt is to provide a high code reutilization for general optimization works. Three different methods: genetic algorithm, memetic algorithm and multiple start are built in the NP-Opt. The framework also includes some refinements like population structure, fuzzy controlling of the evolutionary parameters and multiple populations. Five different types of recombination have been included so far, as well as two local search procedures. The class ”Framework” is the main control class of the NP-Opt framework. It provides all the possible control options to the user, such as problem type selection, instance to solve, method selection and execution parameters. It also defines all the variables used to control and run each problem or method, and includes the calling procedures to the classes linked to it. All code of NP-Opt was developed in Java, and it also provides a graphical user interface for users. To generate new problems in the NP-Opt, users must define the following: a representation for the problem, the method to evaluate a solution, the mutation scheme, the crossover operator, and the representation for the instance.

Many conventional optimization frameworks are based on evolutionary computation methods, and some even focus on genetic algorithm only. It is useful to develop optimization framework to include not only global evolutionary algorithms but also numerical deterministic methods. For more flexibility, it should allow user to mix different optimization methods for hybrid optimization procedures. It is also beneficial for user to easily define their problems and to build new optimization solvers by themselves.

Compared with these frameworks, the developed UOF contains not only evolutionary algorithms but also numerical deterministic methods. Furthermore, the hybrid approach which integrates both evolutionary and numerical algorithms is also can be easily implemented in the UOF. We use one minimization problem to compare the efficiency of the hybrid approach with the pure evolutionary or numerical algorithms. Consider a minimization problem in the following expression,

$$f(x_1, x_2) = 21.5 + x_1 sin(4\pi x_1) + x_2 sin(20\pi x_2), \tag{2.1}$$

where $-3.5 \leq x_1 \leq 12.1$ and $4.1 \leq x_2 \leq 5.8$. Figure 2.1 shows the values of $f(x_1, x_2)$ respect to different x_1 and x_2. The minimal value of f function is 3.8532 and occurs at $(x_1, x_2) = (11.8759, 5.7745)$.

Table 3.3 shows the found solutions by different methods. According to the results, the numerical method can not find the global minimal value without a good initial guess. The proposed hybrid approach can quickly get the optimal solution, and the evolutionary approach can also find a good solution, but it requires much longer time for iterations. The hybrid approach demonstrate excellent efficiency in solving this minimization problem.

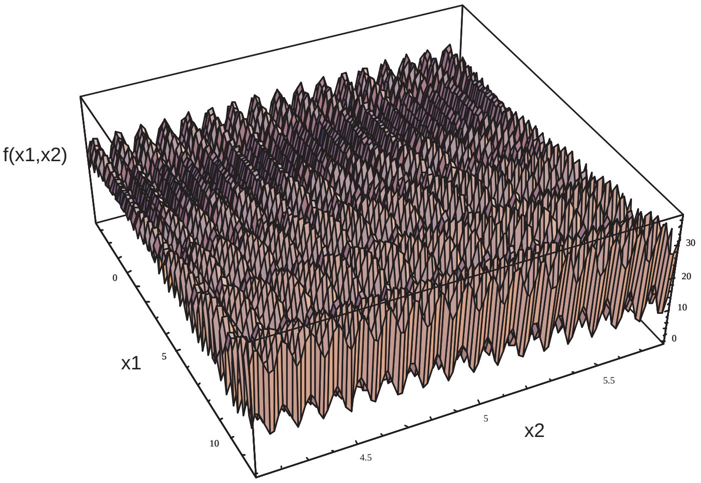


Figure 2.1: The values of $f(x_1, x_2)$ respect to different x_1 and x_2 values.

1896

Table 2.1: The found solutions of different methods in solving the minimization problem of Eq. 2.1.

# of iterations	Numerical method	GA	The hybrid approach
1	11.54	11.54	11.54
10	9.48	10.99	10.850
20	8.72	9.946	7.603
40	8.72	8.054	5.344
60	8.72	5.984	3.853
100	8.72	4.015	–

Chapter 3

The Unified Optimization Framework

The UOF is an object-oriented framework for general problem optimization. Based on biological inspired techniques, numerical deterministic methods, and C++ objective design, the UOF [34] itself has significant potential to perform optimization operations on various problems. The components of the UOF can be separated into problem and solver parts. The details of these two components and the architecture of the UOF are presented in this chapter. Several implementation examples are given to demonstrate the way to construct solvers or problems in the UOF. Furthermore, one developed parallelization technique are illustrated through some applications. One open-source project, Visi for high-dimensional engineering and scientific data visualization is also presented at the end of this chapter.

3.1 The Architecture of the UOF

Figure 3.1 illustrates the class diagram of UOF. The class UOFId is the base-class of every class here for run-time type information (RTTI) purposes. All members in UOF can be easily categorized into problem-related and solver-related classes. The UOFProblem class is the main class to define the problem; the UOFInitializer class is responsible for the initialization of the solution for each problem; the UOFEvaluator class provides the method for evaluating the result obtained by the UOFProblem derived classes; the UOFConstraint class defines the constraint of each parameter; the UOFSolver class takes the major character in solver relative category; the UOFSolution class stores possible solutions and several operators of the specified solver; the UOFTerminator class judges when and how to stop the optimization process of the solver, and the UOFInfo class logs the behavior of the solver during the solving process. All classes in both categories are abstract classes, and need to be implemented in derived classes.

Figure 3.2 shows the built-in class specializations of the most important two classes in current UOF. As shown in Fig. 3.2a, NPProblem, FunctionProblem, and ExtSimProblem are directly inherited from UOFProblem. The UOFProblem class has general interfaces for the user to define any problems. NPProblem provides convenient interfaces for defining NP-hard problems such as TSP and the single/parallel machine scheduling problem. Moreover, optimization problems of linear and non-linear continuous functions, such as DeJong

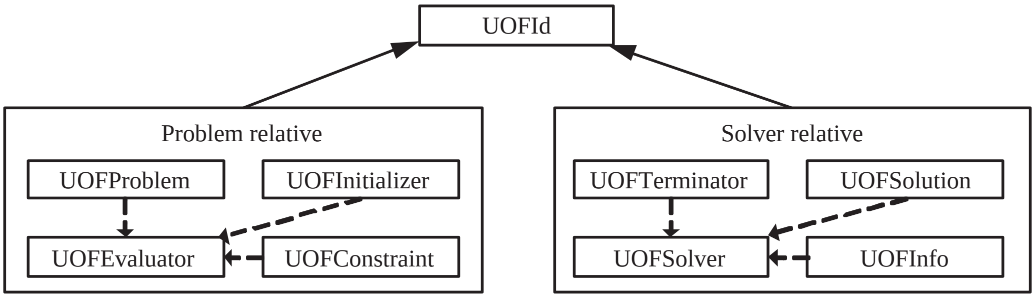


Figure 3.1: An illustration of the architecture of UOF. Solid lines indicate inheritance and dashed lines indicate the ownership.

functions [12], can be defined by FunctionProblem class. ExtSimProblem class has basic linkages to external software to perform simulations, such as commercial TCADs and ECADs. The solver-related classes illustrated in Fig. 3.2b can be separated into two categories, namely gradient-based and the population-based solvers. The fundamental abstract classes are described in detail in the following sections.

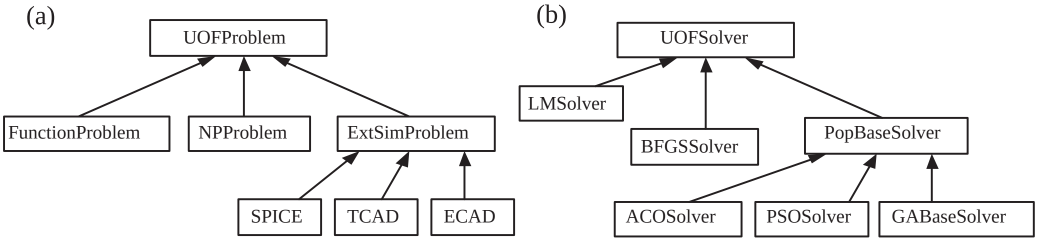


Figure 3.2: The class hierarchy of the UOFProblem and UOFSolver. The solid lines indicate inheritance.

3.1.1 The UOFProblem Class

The interface of UOFProblem class is defined as follows:

```
class UOFProblem : public UOFId
{
public:
    virtual double GetResult(void *inp)=0;
    virtual void GetResult(void *inp,void *out)=0;
    virtual void GetJacobianResult(void *inp,void *out)=0;
    virtual void GetExternalData(void *data)=0;
    virtual void SetConstraint(void *cnt)=0;
}
```

The GetResult function is the major function of UOFProblem class to calculate the results of given parameters. The function has two overloaded versions. One is a convenient version which returns a result in a ‘double’ format, and the other is the type-free version that returns data in any specified type. The GetJacobianResult function is used by the gradient-based solvers, which return the Jacobian matrix back to the solver.

3.1.2 The UOFEvaluator Class

The UOFEvaluator interface is declared as follows:

```
class UOFEvaluator : public UOFId
{
public:
    UOFEvaluator(UOFProblem & src) : m_pProblem(& src){}
    virtual double operator()(void* src) = 0;
    UOFProblem *m_pProblem;
}
```

The UOFEvaluator itself is a function object [21] that acts like a function pointer in C, but takes advantages of C++ object-oriented programming. It stores the pointer of UOFProblem, and performs the communications with UOFProblem in the operator() function. UOFEvaluator is like an agent that connects the UOFProblem and UOFSolver to reduce the coupling effects between UOFProblem and UOFSolver.

3.1.3 The UOFInitializer Class

The interface of UOFInitializer:

```
class UOFInitializer

{

public:

        virtual void operator()(void* src, UOFProblem* pT);

}
```

UOFInitializer performs solution initialization, which is called by UOFSolution object. The solution to be initialized should be transferred to void* type. Moreover, UOFInitializer also stores the UOFProblem pointer, and allows the user to retrieve extra information in UOFProblem object.

3.1.4 The UOFSolver Class

The interface of UOFSolver is defined as follows:

```
class UOFSolver : public UOFId

{

public:

        virtual void Initialization()=0;

        virtual void Solve()=0;

        virtual bool Configuration(const char *filename);
```

}

In this class, the Initialization function performs the initialization steps before the solving procedure. Configuration function allows the user to configure the solver through a script file, and the Solve function controls the entire optimization process.

The solvers of the UOF comprises two parts, the gradient-based and the population-based solver. The gradient-based solver includes the Broyden-Fletcher-Goldfarb-Shanno (BFGS) [13] and Levenberg-Marquardt (LM) methods [14]. Both of these are quasi-Newton methods and are popularly used in deterministic solving packages [15]. The population-based solver includes several heuristic methods such as genetic algorithm (GA) [16], particle swarm optimization (PSO) [17], ant colony optimization (ACO) [18] and simulated annealing method [19, 20].

In 1970, an alternative update formula was suggested independently by Broyden, Fletcher, Goldfarb, and Shanno. The method is now called the BFGS algorithm [13]. The BFGS method is derived from the Newton's method and is used to solve unconstrained nonlinear optimization problems. The LM method is a quasi-Newton method to accelerate the Gauss-Newton method. The Gauss-Newton method is the basic algorithm for solving the nonlinear optimization problem. Due to the nonlinear property of the problem, a gradient for each variable can be obtained. It starts from an initial guess, and follows the direction

1896

of the normal of the gradient to find the optimal solution. Therefore, the initial guess must be chosen carefully, or the solution may fell into a local optima. Unlike the Gauss-Newton method has the fixed steps toward the solution, LM optimization method detects that some regions with monotonic variation property can be speed up by increasing the step size. On the other hand, when the optimization process encounters a sensitive region, the step should be shorten to avoid skipping the optimum.

GA is a global search optimization method based on the mechanics of natural selection and natural genetics. It works with a coded of parameters string called chromosome instead of the solutions themselves. Each chromosome represents a solution set, and the fitness functions used to measure the survival scores of all chromosomes in the population. Then the GA will accord its selection scheme to select several chromosomes for copulation, discard unwanted chromosomes, and adopt the crossover scheme to produce the new generation. Then the GA will apply fitness function for the new population again and loop this cycle until certain stop criteria is achieved. The GA includes following parts, the problem definition, the gene encoding, fitness evaluation, selection, crossover, mutation. Particle swarm optimization is a population based stochastic optimization technique developed by Eberhart and Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling. The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles. The particle swarm optimization concept consists of, at each time step, changing the velocity of (accelerating) each particle toward its $pbest$ and $lbest$ locations (local version of PSO). Acceleration is weighted by a random term, with separate random numbers being generated for acceleration toward pbest and lbest locations. The ant colony optimization is also a population-based approach to the solution of combinatorial optimization problems. The basic ACO idea is that a large number of simple artificial agents are able to build good solutions to hard combinatorial optimization problems via low-level based communications. Annealing process is the way in which a metal cools and freezes into a minimum energy crystalline structure (the annealing process) which was proposed by Nicholas Metropolis in 1953. In 1985, Kirkpatrik proposed the simulated annealing (SA) optimization method which mimics the real annealing process in the system to find a optimal cooling process to get the minimal energy crystalline structure (optimal solution). First, they use SA to solve discrete problems such as T.S.P, Backpacking Problem, N.P. completeness problems, and so no. Recently, SA is widely used in: layout routing, parameters extraction ... etc.

3.1.5 The UOFSolution Class

UOFSolution is an abstract class to describe the solution format used in any UOF components. The class declaration is listed as below:

```
class UOFSolution : public UOFId
{
public:
    UOFSolution(UOFInitializer *init, UOFEvaluator *eval);
    UOFSolution(const UOFSolution&);
    UOFSolution& operator=(const UOFSolution&);
    virtual UOFSolution* clone() const;
    virtual void copy(const UOFSolution&);
    virtual double score();
}
```

The score function in UOFSolution invokes UOFEvaluator to obtain the simulation results of given parameters, and calculates the fitness score.

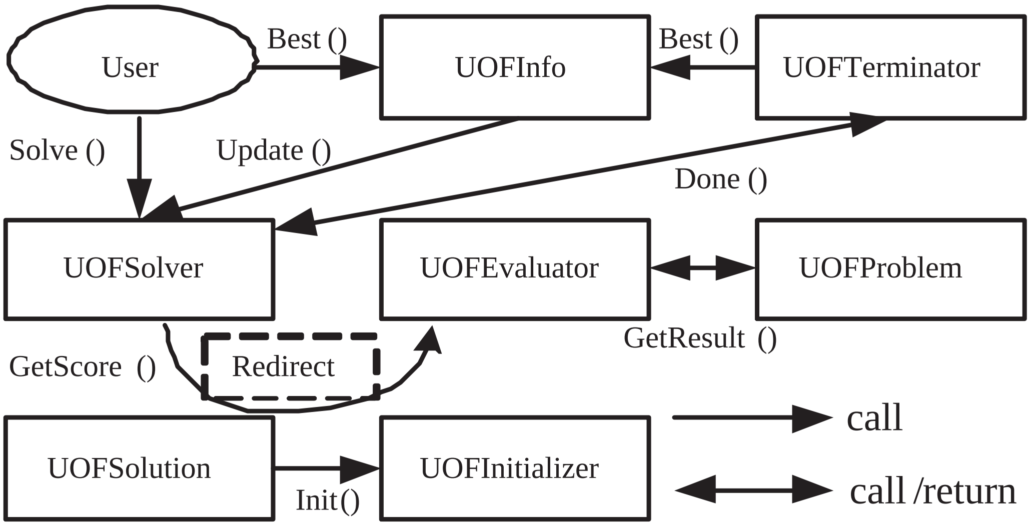


Figure 3.3: The working flow and object inner connections in UOF from users' point of view.

3.1.6 The Working Flow of the UOF

Figure 3.3 shows the working flow and object inner connections in the UOF from the user point of view. Users initially call the Solve function in the UOFSolver class to start the optimization operation. Meanwhile, the UOFInitializer initializes the UOFSolution objects. The Solve function invokes the GetScore function in UOFSolution class to calculate the results from the given parameters. The UOFSolution class redirects and passes the GetScore message to the UOFEvaluator class. The UOFEvaluator object passes the solution to UOFProblem, and calls GetResult to obtain the corresponding results. While the solver solves the problem, in each iteration the UOFInfo updates the current status of the solver, such as the best parameter set. Moreover, the UOFTeriminator also updates the current best solution from UOFInfo, and tells UOFSolver when to stop the optimization procedure. Finally, users can obtain the best solution from the UOFInfo object.

3.2 Implementation Examples

To apply the UOF with any specific problem, we first convert the problem as an optimization problem. The adjustable parameters, the evaluation criteria and the targets should be defined clearly. In the next step is to build the problem class for the specific problem. It is an easy way to duplicate one default problem class and alter the related variables and functions in the class accordingly. This section constructs of simulation-based problems and two solvers step by step given to demonstrate the capability and extensibility of UOF.

3.2.1 Simulation-based Problem

The major task in implementing a simulation-based problem in UOF is to encapsulate the I/O operations to external simulators within the subclass of the UOFProblem class. This section describes the specialization of UOFProblem class without loss of generality, taking the TCAD reverse modeling problem as an example. Three classes, TCADProblem, TCADEvaluator and TCADInitializer, are derived. TCADInitializer is responsible for generating a random initial parameter set. TCADEvaluator measures the error between the target data and the simulated result. TCADProblem provides the GetResult, GenerateIntermediateFiles, and ReadOutputFiles functions. Figure 3.4 shows the working flow of these methods. The Figure indicates that the GetResult function is rather simple. GetResult first calls the GenerateIntermediateFiles function to generate the input files required by the simulator, then runs the simulator, and finally calls the ReadOutputFiles to retrieve the simulated result. GenerateIntermediateFiles function generates the input files required by the simulator. These files are called m_InterMediateFiles. This procedure requires the m_InputFile data, and maps the value of each parameter into the m_InputFile to generate the m_InterMediateFiles. The ReadOutputFiles function parses the output file of the simulator to retrieve the necessary simulated result. Figure 3.5 shows the flowchart of TCADEvaluator. TCADEvaluator calls GetResult in TCADProblem to retrieve the simulated result, and returns the error between the simulated result and the target data to the caller.

3.2.2 Genetic Algorithm Solver

This subsection describes the construction of a genetic algorithm solver. The first step is to declare three classes named GABaseSolver, TerminateUponIteration and GA1DArraySolution, which are inherited from PopBaseSolver, UOFTerminator and PopSolution, respectively. In this experiment, the stopping criteria was set to reach the maximum iteration, therefore, so the TerminateUponIteration function would return true if the maximum iteration was

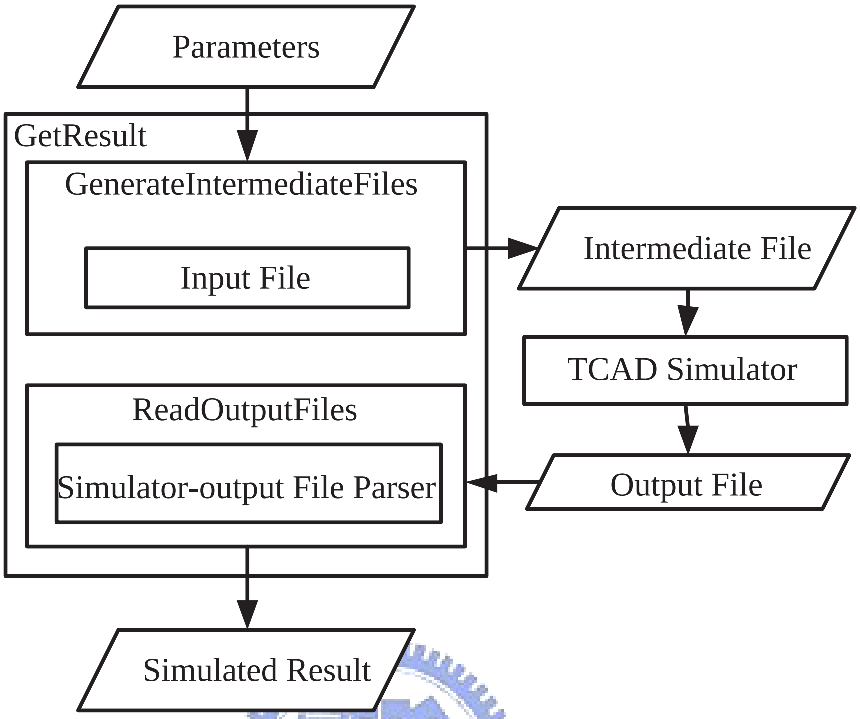


Figure 3.4: The working flow of GetResult in TCADProblem.

reached. The GA1DArraySolution class rewrites the GetScore function, and declares the GA crossover and mutation operator. The Solve function is the entry point of the solver called by the main function of the program. It first calls the Initialization, then performs Evolve to evaluate each individual and perform further genetic operators, namely selection, crossover, and mutation until TerminateUponIteration (in Done function) reports the "true" event. The pseudocode for Solve and Evolve functions are as follows.

```
void GABaseSolver::Solve()
```

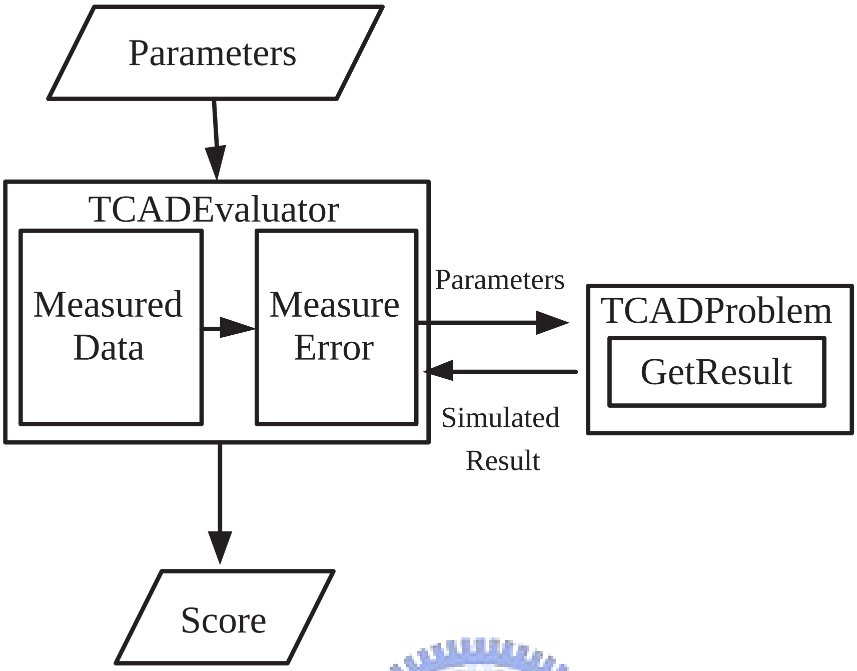


Figure 3.5: The flowchart of TCADEvaluator.

```
{
    Initialization();
    do{
      Evolve();
    }while(!Done());
}
void GABaseSolver::Evolve()
{
    1.Compute fitness score of each parameter set
      For each solution (in m_Pop array):
      Call m_Pop[a]→ score();
    2.Sort the fitness score:
      sort(m_Pop.begin(),m_Pop.end(),MinScore);
    3.Selection operation:
      (*m_pSlct)(m_Pop, selected);
    4.Crossover operation:
      (*m_pCros)(selected[p1], selected[p2], m_Pop[a], m_Pop[b]);
    5.Mutation operation:
      (*m_pMutr)(m_Pop[a]);
}
```

3.2.3 Particle Swarm Optimization Solver

The PSO, like the GA, is a population-based solver, meaning that the content are alike. The major difference between these two solvers occurs in the Evolve function. The PSOSolver performs a move operation defined in the PSOMove function of the PSO1DArraySolution to update the parameters after sorting the individuals. The pseudocode of the Evolve function is given below.

```
void GABaseSolver::Evolve()
{
    1.Compute fitness score of each parameter set
        For each solution (in m_Pop array):
        Call m_Pop[a]→ score();
    2.Sort the fitness score:
        sort(m_Pop.begin(),m_Pop.end(),MinScore);
    3. Move particles:
        (*m_pMove)(m_Pop[0], m_Pop[i], m_Vc, m_K1, m_K2);
}
void PSOMove::operator()
( PopSolution* bestp, PopSolution* currentp, double w, double k1, double k2)
{
    1.Convert bestp and currentp to PSO1DArraySolution type
    2.Compute the velocity of currentp;
    3.Update the position of currentp;
}
```

3.2.4 Basic Experiment on Traveling Salesman Problem

The TSP is a well-known discrete combinational problem. Its objective is to discover a minimized route for a salesman to visit each city only once and finally return to the starting point. Three classes, TSPProblem, TSPEvaluator, TSPInitializer, which are inherited from UOFProblem, UOFEvaluator, UOFInitializer, respectively, are defined to implement TSP in the UOF standard. The TSPInitializer is utilized to create a random string as a route to travel to each city. The TSPEvaluator simply returns the total length of the given route, which is derived from TSPProblem. Pseudocode for TSPInitializer, TSPEvaluator, and GetResult function in TSPProblem is given below.

```
class TSPInitializer : public UOFInitializer
{
    public:
        virtual void operator()(void* src, UOFProblem* pProblem)
        {
            1.Random generates the initial solution S
            2.Set S to src
        }
}
class TSPEval : public UOFEvaluator
{
    public:
        virtual double operator()(void* src)
        {
            1.Get total length of the route from TSPProblem:
            TotalLength = m_pProblem → GetResult(src);
            2.Return TotalLength
        }
}
double TSPProblem::GetResult(void* param)
{
    1.Convert param into proper data type
    2.Compute total_distance from param
    3.Return total_distance
}
```

Table 3.1: The optimized result of the TSP.

Method	Geometry	# of cities	Average # of Iteration	Percent of success run
GA	Circle	30	2231	80%
ACO	Circle	30	4	80%
GA	Circle	50	6562	30%
ACO	Circle	50	18	60%
GA	Matrix	9	24	100%
ACO	Matrix	9	4	100%
GA	Matrix	25	3600	30%
ACO	Matrix	25	26	100%

Four geometric distributions of cities were explored. The first and second distributions formed a circle, and contained 30 and 50 cities, respectively. The third and fourth distributions were 3×3 and 5×5 matrices, respectively. The GA and ACO were performed to compare the efficiency of each problem. Table 3.1 summarizes the result, revealing that the ACO method is better than GA method for solving this problem. The result also confirms that the UOF can solve the discrete combinational optimization problem.

3.3 The Developed Parallelization Technique

It is known that many engineering problems are quite complicated and time-consuming jobs which require huge resource for computations. Therefore, a parallelization technique can reduce the computing time and benefit the design flow in the electronic design. Based on the message passing interface (MPI) libraries and some partition techniques, we have developed one parallelization flow to speedup the simulation and optimization processes. Figure 3.6 shows a basic working flow of our parallelization approach. First we need to analyze the problems, try to partition the problem into sub-jobs, and then dispatch these sub-jobs to the processors in the cluster. Meanwhile, the data communications between processors should be carefully defined, and the communications are performed with the MPI libraries. Finally one processor in the cluster may be regard as the master sever to monitor and collect the necessary outputs in each processor of the PC-based Linux cluster. The benchmarks, such as speedup, efficiency, and maximum difference with respect to various numbers of processors can be adopted to evaluate the parallel performances. The speedup is the ratio of the code execution time on a single processor to that on multiple processors. The efficiency is defined as the speedup divided by the number of processors. In our PC-based Linux cluster, as shown in Fig. 3.7, each PC is equipped IBM eServer EM64T with 3.6 GHz CPU, 2 GB memory, and Intel 100 MBit fast Ethernet. All PCs in the constructed 32-nodes cluster system are connected with 100MBit 3Com Ethernet switch.

The major challenges in implementing parallelization are the way to partition the problems into sub-jobs. It is entirely dependent on the problems, and will affect the performance of the parallelization. Conventionally, there are two different ways to perform the partition.

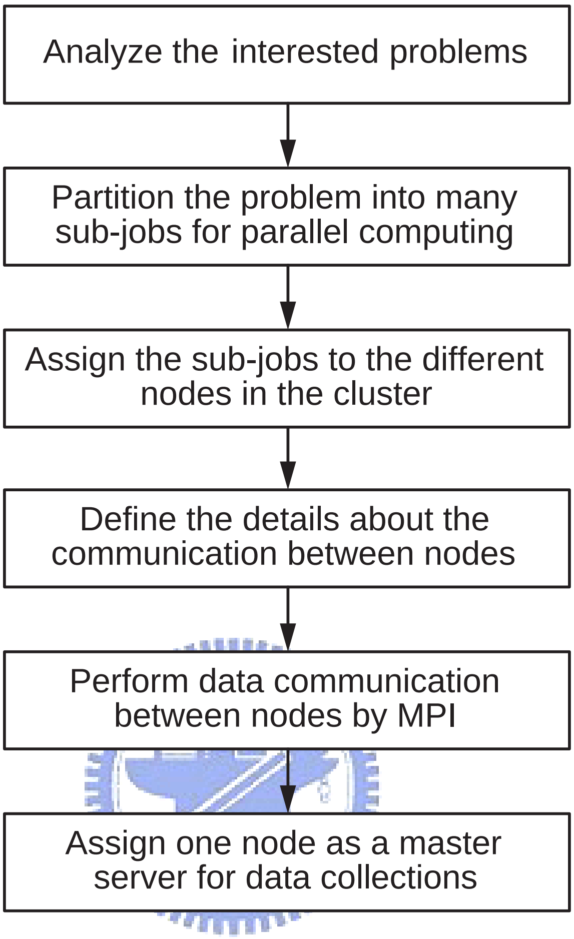

Analyze the interested problems
Partition the problem into many sub-jobs for parallel computing
Assign the sub-jobs to the different nodes in the cluster
Define the details about the communication between nodes
Perform data communication between nodes by MPI
Assign one node as a master server for data collections

Figure 3.6: The basic working flow of the developed parallelization approach.

One way is to partition the problem domain into sub-domains, and then perform the calculation or simulation on each sub-domain separately. For the parallelization of the semiconductor device simulation [61], we use a domain decomposition approach to partition the simulation domain into sub-domains, and then simulate the sub-domains on different processors in the cluster. Another method is to perform parallelization based on the used

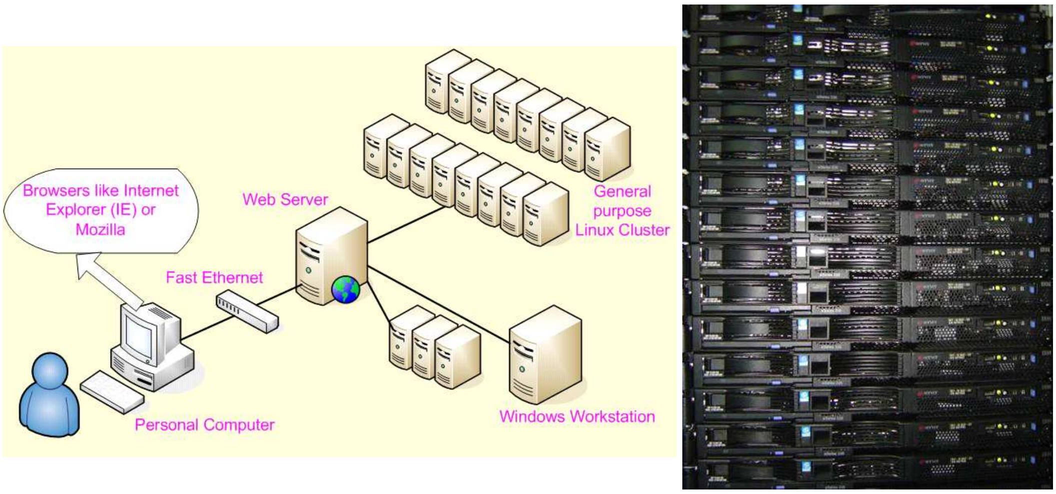

Browsers like Internet Explorer (IE) or Mozilla
Web Server
General purpose Linux Cluster
Fast Ethernet
Personal Computer
Windows Workstation

Figure 3.7: The constructed 32-node PC-based cluster.

algorithms. For example, according to the property of genetic algorithm, we can simply execute the GA on each processor simultaneously for the parallelization. In following, two applications of the developed parallelization method are presented to show the efficiency of the proposed method.

3.3.1 Parallelization for Nanoscale Double-Gate MOSFETs Simulation

In this section, we introduce the parallelization for semiconductor devices simulation which is based on the partition of the simulation domain. In this investigation, based on a posteriori error estimation, the triangular mesh generation, the adaptive finite volume method, the monotone iterative method, and the parallel domain decomposition algorithm, a set of two-dimensional quantum correction hydrodynamic (HD) equations is solved numerically for the double-gate metal-oxide-semiconductor field effect transistors (MOSFETs) on our constructed cluster system.

Classical HD model consists of at least five coupled partial differential equations (PDEs). The HD equations in semiconductor device simulation are as follows,

$$\triangle\phi = \frac{q}{\varepsilon_s}(n - p + D), \tag{3.1}$$

$$\frac{1}{q}\nabla \cdot J_n = R(n,p), \tag{3.2}$$

$$-\frac{1}{q}\nabla \cdot J_p = -R(n,p), \tag{3.3}$$

$$\nabla \cdot S_n = J_n \cdot E - n(\frac{w_n - w_0}{\tau_{nw}(T_n)}), \tag{3.4}$$

$$\nabla \cdot S_p = J_p \cdot E - p(\frac{w_p - w_0}{\tau_{pw}(T_p)}), \tag{3.5}$$

where ϕ is the electrostatic potential and n and p are classical electron and hole concentrations. T_n and T_p are electron and hole temperature (K). The electric field E (V/cm) is defined by $E = -\nabla\phi$, q is the elementary charge its unit is coulomb. ε_s is the semiconductor permittivity (F/cm). w_0 is the average carrier energy (eV) in the thermal equilibrium, and the net doping concentration is $D(x,y) = N_D^+(x,y) - N_A^-(x,y)$. R is the net recombination rate (cm^{-3}s^{-1}), and τ_{nw} and τ_{pw} are the electron and hole energy relaxation time

(s) approximations. The average carrier energy consists of the thermal energy and the drift energy

$$w_n = \frac{3}{2}k_B T_n + \frac{1}{2}m_n^* v_n^2, \tag{3.6}$$

$$w_p = \frac{3}{2}k_B T_n + \frac{1}{2}m_p^* v_p^2, \tag{3.7}$$

for electrons and holes, respectively. m_n^* and m_p^* are the electron and hole effective masses (Kg). v_n and v_p are the electron and hole mean velocities (cm/s). The carrier's currents and energy flux densities are given by

$$J_n = -q\mu_n n \nabla\phi + qD_n \nabla n + \mu_n k_B n \nabla T_n, \tag{3.8}$$

$$J_p = -q\mu_p p \nabla\phi + qD_p \nabla p + \mu_p k_B p \nabla T_p, \tag{3.9}$$

$$S_n = \frac{J_n}{-q} w_n + \frac{J_n}{-q} k_B T_n + Q_n, \tag{3.10}$$

$$S_p = \frac{J_p}{+q} w_p + \frac{J_p}{+q} k_B T_p + Q_p, \tag{3.11}$$

where μ_n and μ_p are the carrier mobility ($\mathrm{cm}^2/\mathrm{V}{-}\mathrm{s}$). The diffusion coefficients, D_n and D_p (cm^2/s), satisfy the Einstein relation. Qn and Qp are the heat flows. However, to simulate the nanoscale double-gate MOSFETs, the classical HD model is insufficient due to significant quantum confinement effects. Therefore, we have to consider the quantum mechanical effects in the classical HD model above. The quantum correction equation for the quantum corrected inversion-layer charge densities n_{QM} is

$$n_{QM} = a_0 n_{CL}(1 - exp(-a_1\xi^2(1 - \frac{1}{2}(\frac{\xi}{\xi_0})^2) - a_2\xi^3), \tag{3.12}$$

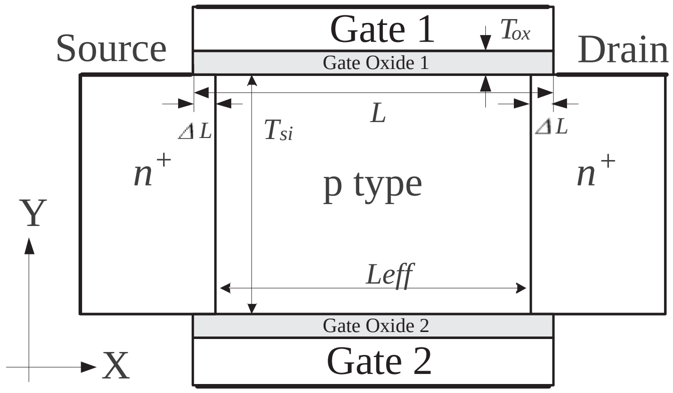


Figure 3.8: An illustration of the cross-section view for the n-type double-gate MOSFET device.

where n_{CL} is the classical electron density solved from the Poisson equation. $\xi = x/\lambda_{th}$ is dimensionless and $\lambda_{th} = (\frac{\hbar^2}{2m_0 k_B T})^{1/2}$ is the thermal wavelength ($\mathring{A}$), $\hbar$ is the reduced Planck constant (J$-$s), m_0 is the electron rest mass (Kg), k_B is the Boltzmann constant (J/K), T is the absolute temperature (K) and $\xi_0 = T_{si}/2\lambda_{th}$ is dimensionless. Together with the auxiliary equation (3.12), the conventional HD model forms a quantum correction HD model. The associated boundary condition for the model is the same with the conventional HD model.

We solve the above quantum correction HD model with a parallel adaptive computing technique for nanoscale double-gate MOSFETs, as shown in Fig. 3.8. The full set of quantum HD model is firstly decoupled with Gummel's method, and each decoupled PDE is solved sequentially. Figure 3.9 shows the adaptive computational procedure. For each

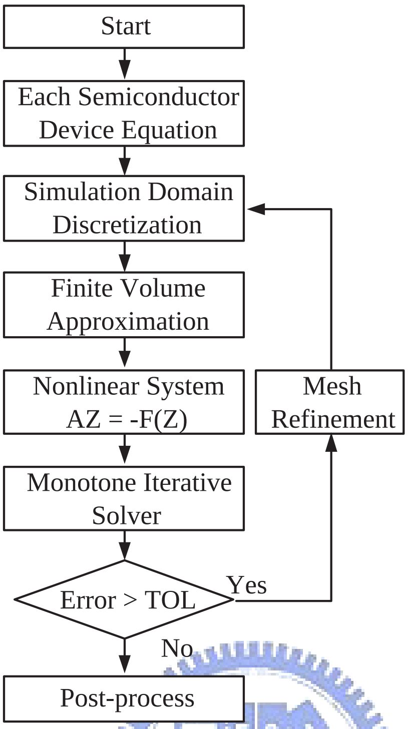

Start
Each Semiconductor Device Equation
Simulation Domain Discretization
Finite Volume Approximation
Nonlinear System AZ = -F(Z)
Mesh Refinement
Monotone Iterative Solver
Error > TOL
Yes
No
Post-process

Figure 3.9: Adaptive finite volume solution algorithm for each decoupled semiconductor device equation. A is the system matrix, Z is the unknown vector, and F is the nonlinear vector.

1896

decoupled semiconductor device equation, we first partition the solution domain into a set of finite volumes. Each decoupled PDE is then approximated by the finite volume method. After the assembling the nonlinear algebraic system, the monotone iterative solver is directly applied to solve the system of nonlinear equations. Once an approximate solution is computed, a posteriori error analysis is performed to assess the quality of the approximate solution. The error analysis will produce error indicators and an error estimator. If the

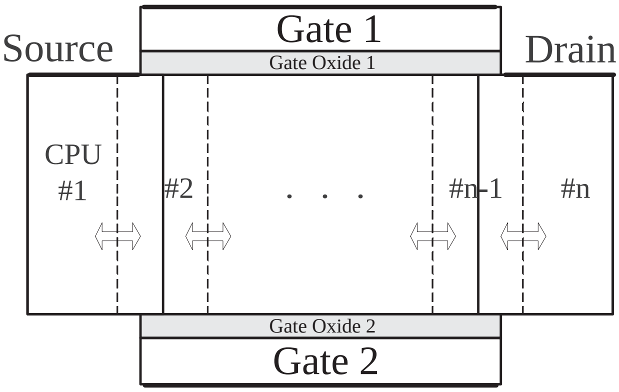


Figure 3.10: Parallel domain decomposition for the two-dimensional double-gate MOSFET. The dash-lines indicate the boundary of partition domains.

estimator is less than a preset error tolerance (TOL), the adaptive process will be terminated and the approximate solution can be post-processed for solving next equation or analyzing physical properties. Otherwise, a refinement scheme is employed to refine the current elements depending on the magnitude of the error indicator for that element. A newer partition of the domain is thus created and a new solution procedure is repeated.

In parallelization, a pre-processor will prepare the input data required for each client. In the configuration of MPI and Linux cluster, input data is prepared on the pre-processor and is sent to each client through TCP/IP, and then all clients do their own local jobs and exchange essential data with the neighbor clients. When each client completes its own jobs, it sends the results to a post-processor. The post-processor then estimates the solution error. If the estimator is larger than a TOL, the post-processor delivers the computed results to the pre-processor for the next mesh refinement. After the calculation and adaptation stages for error estimation as well as error indicators, a workload imbalance may exist due to the change of the number of meshes on individual client. As shown in Fig. 3.10, based on device structure and bias condition, the simulation domain is dynamically partitioned into several disjoint sub-domains. When a refined tree structure is created, the number of processors for the next computation will first be dynamically assigned and allocated following the total number of nodes. We apply the geometric dynamic graph partitioning method in x- or/and y-direction to partition the total number of nodes and assign those partitioned nodes to each processor. For the quantum correction HD model simulation, each partition sub-domain contains five nonlinear systems to be solved, where the systems have arisen from Gummel's decoupled and adaptive FV approximated device PDEs. Once previous results are given, the boundaries for partitioned sub-domains are totally separated. We solve the nonlinear systems with the MI method independently. When newer MI solutions of nonlinear systems are computed, we perform the boundary data exchange for the next Gummel's iteration loop. The parallel domain decomposition is shown in Algorithm 2 and a procedure for the corresponding parallel dynamic partition is shown in Algorithm 3. The communicate between processors is based on the MPI libraries.

```
**Begin** Parallel Domain Decomposition Algorithm
**While** (Error>Tolerance)
```

```
    For (all elements)
        Count the number of nodes and
        Do the dynamic partition algorithm
    End For
    For (all jobs)
        Solve this job
    End For
    For (all elements)
        Perform error estimation and mesh refinement
    End For
End While
End Parallel Domain Decomposition Algorithm
```

```
Begin Dynamic Partition of Domain Decomposition
For (all elements)
    Count the number $S$ of nodes sequentially
End For
Estimate an optimal number $N$ of CPUs with
the number of nodes empirically
```

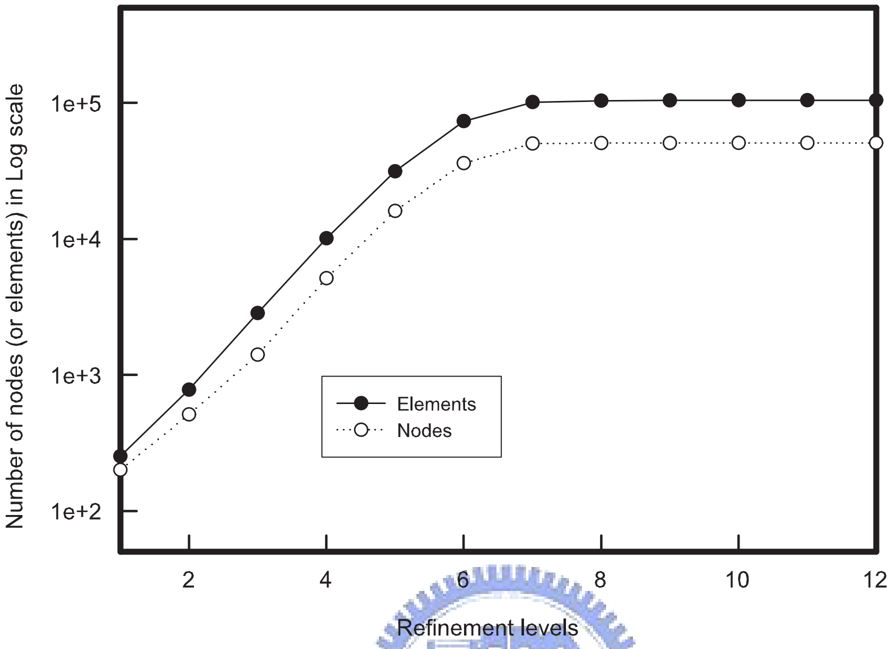


Figure 3.11: The number of nodes (and elements) in Log scale versus the refinement levels.

```
The number $M$ of jobs for each CPU $= S \ / \ N$
For $(i = 1$ to $N)$
    Assign M nodes to CPU$(i)$
End For
End Dynamic Partition of Domain Decomposition
```

The number of refined elements and nodes grows as the refinement levels increase. At

Table 3.2: A list of the achieved sequential and parallel time, efficiency, and speedup with respect to different number of nodes. It is performed on an 8-processors PC-based Linux cluster system.

Nodes	Sequential time (sec.)	Parallel time (sec.) of the 8-processors system	Speedup	Efficiency
1000	9.4	4.1	2.29	28.66 %
4000	87	26.1	3.30	41.67 %
8000	367	83.6	4.39	54.87 %
16000	794	179.2	4.43	55.39 %
32000	4041	735.6	5.49	68.67 %
64000	9403	1993.6	5.30	66.27 %
90000	27775	4487	6.19	77.38 %
250000	151016	20906	7.22	90.29 %

the beginning, the number of refined nodes (and elements) grows fast due to significant variations of solution gradients. After several refinements and solution processes, the increasing rate of the number of nodes (and elements) gradually becomes slow when the refinements increase. It eventually reaches to a saturated condition. Figure 3.11 reports the relationship of the number of nodes (and elements) versus the refinement levels. The increasing rate of the number of nodes (and elements) gradually becomes slow when the refinements increase. Therefore, it confirms the computational effectiveness of the adaptive computing method in the numerical simulation of the quantum correction transport model. The explored double-gate MOSFETs is shown in Fig. 3.8, where the length of the device is equal to 20nm.

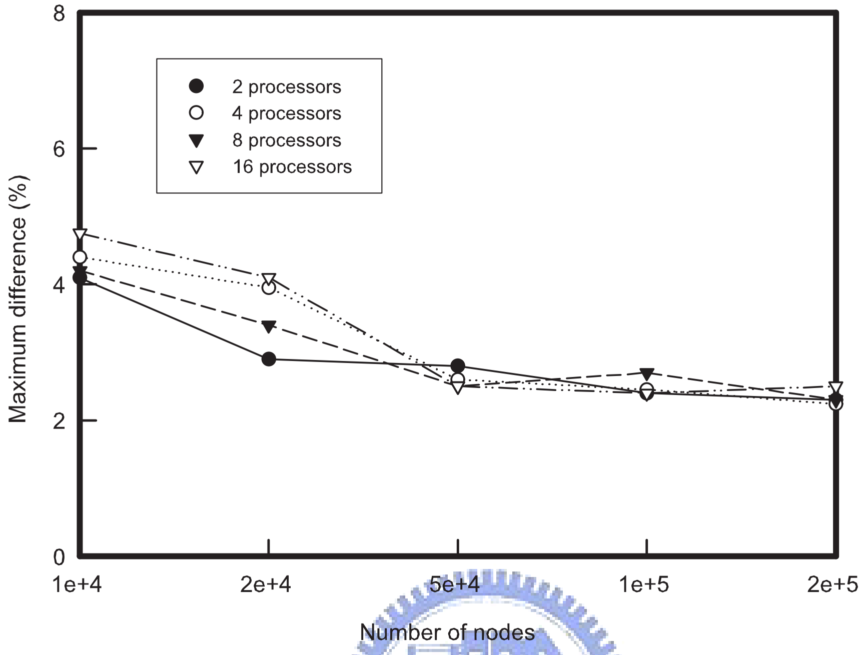


Figure 3.12: The maximum difference versus the number of nodes.

Parallelization of the adaptive simulation is performed on our Linux cluster. To verify the parallel performance of the domain decomposition method for the nanoscale double-gate MOSFETs simulation, the same device under $V_{G1} = V_{G2} = 1.0$ V and $V_{DS} = 0$ V is considered for the following cases. Table 1 reports the details of the achieved sequential and parallel time, efficiency, and speedup with respect to different number of nodes. It is performed on an 8 nodes PC-based Linux cluster system. In our numerical experience, a 7.22 speedup factor is obtained on the tested 8 nodes system. Figure 3.12 is the maximum

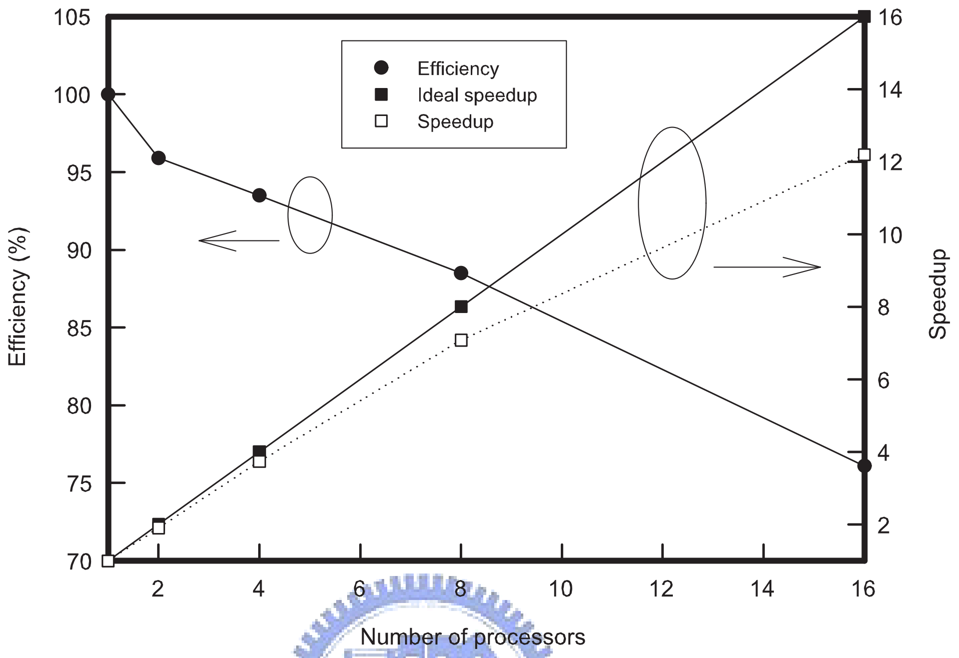


Figure 3.13: The parallel speedup and efficiency versus the number of processors.

difference versus the number of nodes. The maximum difference is defined as the maximum difference of the code execution time divided by the maximum execution time. For the simulation with 2, 4, 8, and 16 processors, the maximum difference decreases and tends to a stable value when the number of nodes increases. It shows a good dynamic load balancing for the domain decomposition. Figure 3.13 is the achieved speedup and efficiency, where the speedup is the ratio of the code execution time on a single processor to that on multiple processors. Efficiency is defined as the speedup divided by the number of processors. The speedup is about 12.2 for the simulation running on a 16 nodes system and 75% efficiency is maintained.

3.3.2 Parallelization for Large-scale Protein Folding Dynamics

In this section, one parallelization technique for the implicitly restarted Arnoldi algorithm that used to solve the eigenvalue problem for large-scale protein folding dynamics are presented. Dynamics of protein folding has recently been of great interest [131–133]. It is nowadays explored in different way, such as mass action models, all atoms, lattice, and off-lattice model, and methods between macroscopic and microscopic models. One of biopolymer folding dynamic core problems is finding an ensemble of transition state conformations or rate-limiting steps. For small molecules, numerical methods theoretically could find transition states, i.e. the saddle points of the potential energy surface. However, for proteins or RNA macromolecules, the potential energy surface of them is usually more complicated and has many pathways. Among approaches, a master equation is crucial in dynamic simulation of protein folding. Deriving from the Liouvillian, the master equation basically describes time evolution of a distribution of trajectories. Numerical solution of the master equation requires computing eigenvalues λ_N and eigenvectors for the corresponding N by N transition matrix. Those negative and larger eigenvalues determine the slowest dynamical relaxation processes. The size of transition matrix grows dramatically with respect to the length of studied protein, which results in a large-scale eigenvalue problem to be solved. Three different approaches, the QR [134], implicitly restarted Arnoldi [135–137], and Jacobi-Davidson [138] methods are compared in calculating all or first few larger nonpositive eigenvalues of the matrix arising from the master equation in protein folding problems. The implicitly restarted Arnoldi method presents its robust in calculating the desired eigenpairs of the master equation. Therefore, parallelization of the implicitly restarted Arnoldi method is implemented for protein folding dynamics with large sequences. The developed parallelization scheme principally partitions the operations of the matrix.

The master equation describing time evolution of trajectory (i.e., the transition of states) is expressed as

$$\frac{dP(t)}{dt} = AP(t). \tag{3.13}$$

To study the dynamics of Eq. (3.13), we have to compute the eigenvalues (in general, the first few largest nonpositive eigenvalue) of the matrix A, where $P(t)$ is the N-dimensional vector of the instantaneous probability of the N conformations. Depending on different free energy, A is a sparse and asymmetric N by N transition (or rate) matrix, where the matrix entries is defined as

$$A_{ij} = \begin{cases} k_{i\to j} & \texttt{, for } i \neq j \\ \sum_{l\neq i} k_{i\to l} & \texttt{, for } i = j \end{cases}, \tag{3.14}$$

where $k_{i \to j}$ is the rate constant for a protein changes its conformation from the i^{th} conformation to the j^{th} conformation, and only depends on the energy states between two conformations

$$k_{i \to j} = \begin{cases} 1 & \texttt{, for } E_i \geq E_j \\ e^{(E_i - E_j)/kT} & \texttt{, for } E_i < E_j \end{cases}. \tag{3.15}$$

where E_i and E_j are energy states. k is the Boltzmann constant and T means the temperature. For any initial conformations, solutions of Eq. (3.13) provides the dynamics of population

$$P(t) = \sum_{m=0}^{N=1} C_m^{(c)} n_m e^{-\lambda_m t}, \tag{3.16}$$

where the $-\lambda_m$ and n_m are the m^{th} eigenvalue and eigenvector. The overall folding kinetics is the linear combination by a coefficient $C_m^{(c)}$ of N possible protein folding conformations. Therefore, $C_m^{(c)}$ means contribution to overall kinetics from the mth mode. The eigenvalues can be ordered as $\lambda_0 < \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_{N-1}$. The eigenvector n_0 for the equilibrium mode gives the equilibrium population distribution. The eigenvector n_1 for the slowest mode indicates the intermediate state of protein folding process which takes longest time to fold to the next conformation.

Furthermore, people may interest in the largest nonpositive eigenvalue λ_1. If there is a large gap between the eigenvalues of slow modes group and a fast modes group, then the overall folding speed is limited by these slow modes. Therefore, this protein may fold fast in the beginning because of the group of fast modes; then there is a rate-determining process

Table 3.3: The number of conformations of proteins with the length of L residues in a 2D square lattice model.

Protein length (L)	Number of conformations (N)
4	5
5	13
6	36
7	98
8	272
9	740
10	2,034
15	296,806
16	802,075
17	2,155,667
20	41,889,578

because of the group of slow modes, and finally it fold as the ground state conformation with its function. Especially for the eigenvalue spectrum with extremely slow modes and a large gap from other modes, this mode is the bottleneck process of the whole folding process. It suggests that some eigenvectors with $\lambda_m << 0$ disappear quickly. On the other hand, any other eigenvectors with $\lambda_m \sim 0$ will determine the slowest dynamic relaxation processes. The overall folding processing time of the protein is approximated as $1/\lambda_1$. We use HP (hydrophobic and polar residues) model in a 2D square lattice. The number of conformations of proteins with L residues is listed in Table 3.3.

To calculate the first few larger nonpositive eigenvalues (or all eigenvalues) of the constructed matrix above, the implicitly restarted Arnoldi method is employed and implemented in this investigation. The implicitly restarted Arnoldi method is known as a direct algorithm for reducing a general matrix into upper Hessenberg form, and it could be more effective than subspace iteration methods for computing the dominant eigenvalues of a large, sparse, real, and asymmetric matrix. An algorithm for implementing the implicitly restarted Arnoldi method is shown below.

Arnoldi Algorithm:

Make an Arnoldi factorization $\mathbf{AV_m} = \mathbf{V_m H_m} + f_m e_m^T$

While (converge)

1. Compute the eigenvalues $\lambda_j, j = 1, 2, ..., m$;

2. Sorting eigenvalues into a wanted set $\lambda_j, j = 1, 2, ..., k$ and unwanted set $\lambda_j, j = k+1, k+2, ..., m$;

3. Perform $m - k = p$ steps of the QR iteration with the unwanted eigenvalues shift to obtain $\mathbf{H_m} Q_m = Q_m \mathbf{H_m^+}$;

4. $\mathbf{AV_m} Q_k = \mathbf{V_m} Q_k \mathbf{H_k^+} + f_k^+ e_k^T$, where $\mathbf{H_k^+}$ is the leading principle submatrix of order k for $\mathbf{H_m^+}$

5. Extend length k Arnoldi factorization to a length m factorization.

Parallelization scheme used in this work is to partition the operations of the matrix. For the Arnoldi factorization

$$\mathbf{AV_m} = \mathbf{V_m H_m} + f_m e_m^T, \tag{3.17}$$

where $\mathbf{V_m}$ is the set of Arnoldi vectors, $\mathbf{H_m}$ is the upper Hessenberg matrix, and f_m is the residual vector, it replicates $\mathbf{H_m}$ on every processor. $\mathbf{V_m}$ and f_m are partitioned by rows and are distributed to every processor in the cluster.

Figure 3.14 is a computational flowchart for the proposed parallel procedure of the solution method. The communication between processors is based on the MPI, where the initialization of the MPI environment has to be carried out firstly. We then perform an Arnoldi factorization, and replicate matrix $\mathbf{H}$ on each node. The next step is to partition the calculation of $\mathbf{V}$ matrix by the available number of processors, and distributes it to each node. Each processor performs the respective matrix operation, exchanges data to other processors. With this approach there are two communication points within the construction of the Arnoldi factorization: the computation of the 2-norm of the distributed vector f_m and the orthogonalization of f_m to $\mathbf{V_m}$ using classical Gram-Schmidt with DGKS correction. Typically the partitioning of $\mathbf{V}$ is comparable with the parallel decomposition of the matrix $\mathbf{A}$. For n-node PCs cluster, the $\mathbf{V}$ matrix is partitioned by blocks $\mathbf{V^T} = (\mathbf{V^{(1)T}}, \mathbf{V^{(2)T}}, ..., \mathbf{V^{(n)T}})$ with one block per processor and with $\mathbf{H}$ replicated on

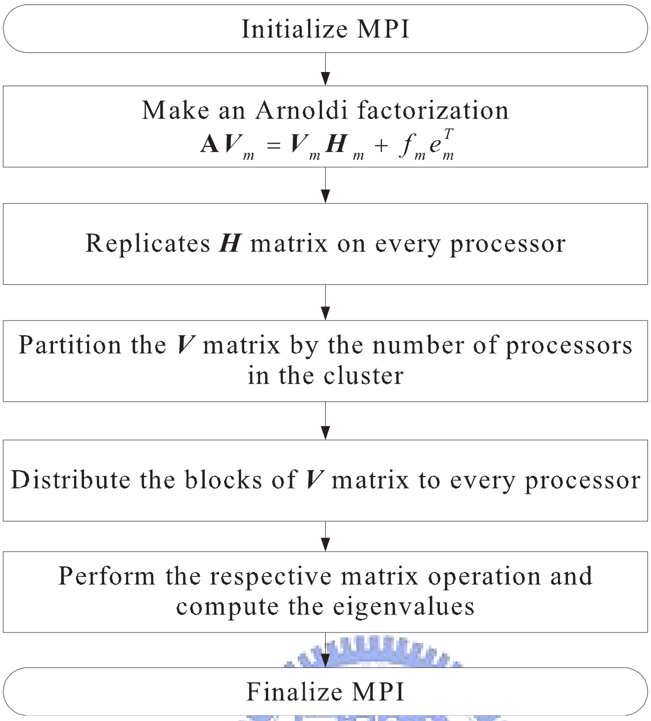

Initialize MPI
Make an Arnoldi factorization
$\mathbf{A}V_m = V_m H_m + f_m e_m^T$
Replicates H matrix on every processor
Partition the V matrix by the number of processors in the cluster
Distribute the blocks of V matrix to every processor
Perform the respective matrix operation and compute the eigenvalues
Finalize MPI

Figure 3.14: A computational flowchart of the implemented parallel method.

each node. Since $\mathbf{H}$ is replicated on each processor, all operations on the matrix $\mathbf{H}$ are replicated on each node and there is no communication overhead. We note that the parallel technique is that the partition of operations on the matrix can accelerate the calculations and reduce communications among processors.

To verify the stability and correctness of the parallel computational method, we test the sequence 0011101011 with 10 residues which has unique conformation in the ground state. To verify the correctness of the parallel solution technique, we plot the distributions

Table 3.4: The achieved parallel speedup and efficiency of the parallel computing algorithm, where the tested case is a 17-mer with only hydrophobic residues.

Processors	Simulation Time (sec.)	Speedup	Efficiency
1	34704	–	–
2	18964	1.83	91.50%
4	9859	3.52	88.00%
8	5119	6.78	84.75%
16	2902	11.96	74.75%
32	1496	23.20	72.50%

of all computed eigenvalues, shown in Figure 3.15, from a single processor PC and 32-node PC-based Linux cluster. Obviously, the computed eigenvalues are totally the same in a single PC and 32-node PC-based cluster. We also perform other verifications with larger sequences and have similar results. This indicates that the implemented parallel method provides a computationally efficient way to accurately calculate the eigenvalues in studying dynamics of protein folding. Figure 3.16 shows the maximum norm error of the smallest and fiftieth eigenvalue versus the number of iterations in a single PC and 32-node PC-based cluster. The convergence of the smallest eigenvalue is much faster than that of the fiftieth eigenvalue. It is also found that our parallel method has a similar behavior of convergence in 32-nodes cluster compared with the behavior in a single PC.

Table 3.4 summarizes the parallel speedup and efficiency of the implemented parallel technique. The tested cases is a 17-mer with only hydrophobic residues, and we compute

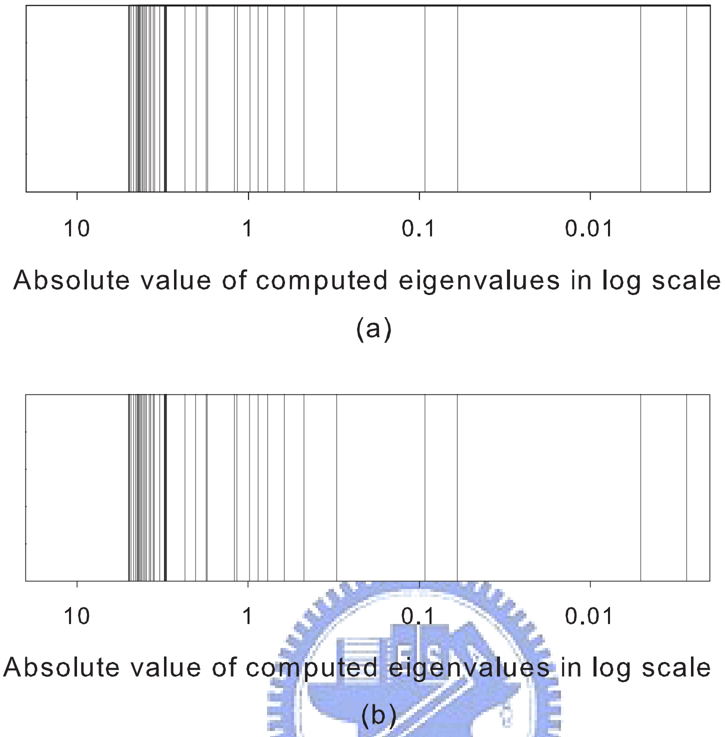


Figure 3.15: The distribution of all computed eigenvalues for the sequence 0011101011 ($L = 10$) in (a) a single processor (b) and 32-node PC-based Linux cluster.

the first 50 eigenvalues of the corresponding transition matrix. Figure 3.17 is the computed eigenvalues of the transition matrix of a 17-mer with only hydrophobic residues. The dimension of the transition matrix is 2155667 by 2155667, and 50 eigenvalues have been computed. We find that the number of processors increases, and the efficiency decreases, as shown in Tab. 3.4. However, a 23-times speedup is maintained and the efficiency is over 72% on the 32-nodes cluster for the 17-mer case. For the same tested case, as shown in Fig.

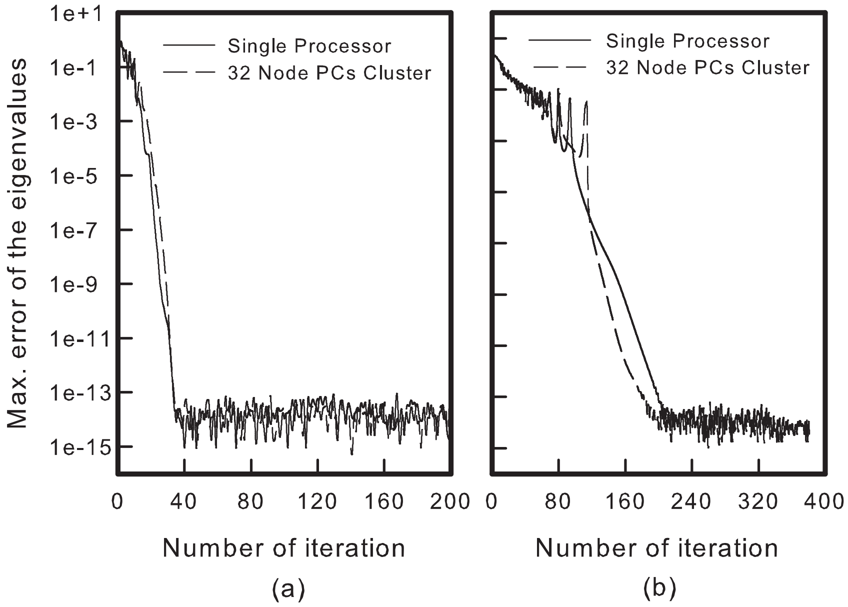


Figure 3.16: The maximum norm error of the (a) first and (b) the $\mathbf{50^{th}}$ eigenvalues versus the number of iterations.

1896

3.18, the variation of maximum difference is within 8% when the number of processors is increased from 2 to 32. The maximum difference is defined as the maximum difference of the code execution time divided by the maximum execution time. Figure 3.19 shows the achieved parallel efficiency versus the matrix size for different number of processors. The parallel efficiency can hold over 70% for parallelization on 2, 4 and 8 processors, which is almost independent of matrix size. It even achieves 90 % efficiency on 2 processors for all matrix sizes. The parallelization on 16 and 32 processors shows the efficiency increases when the matrix size increase. However, the efficiency is degraded for the cases

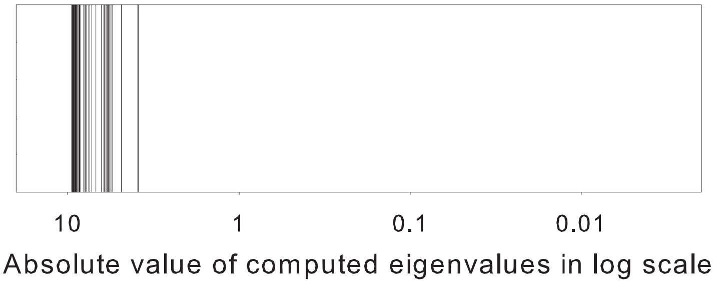


Figure 3.17: The eigenvalues of the transition matrix of a 17-mer with only hydrophobic residues. The dimension of the transition matrix is 2155667 by 2155667, and 50 eigenvalues have been computed.

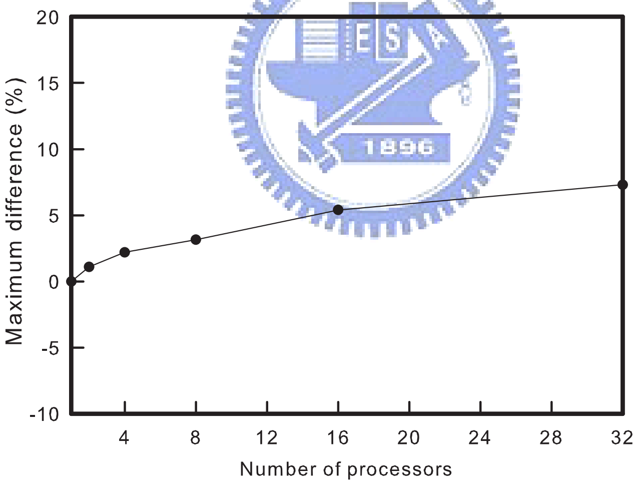


Figure 3.18: The achieved load balancing versus the number of processors for the tested case of a 17-mer with only hydrophobic residues.

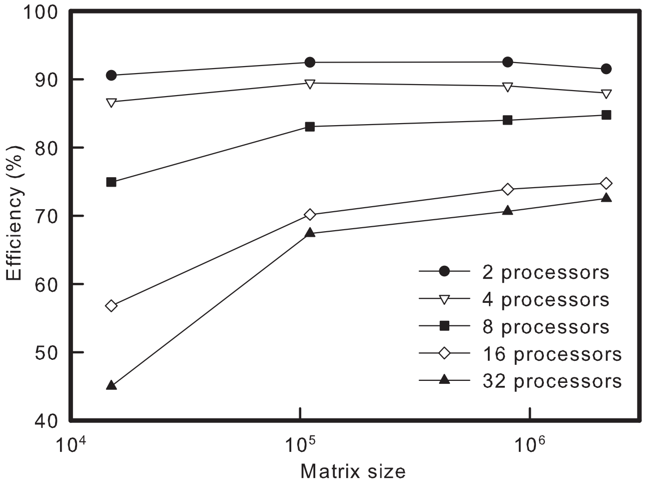


Figure 3.19: The achieved parallel efficiency versus the matrix size for different number of processors.

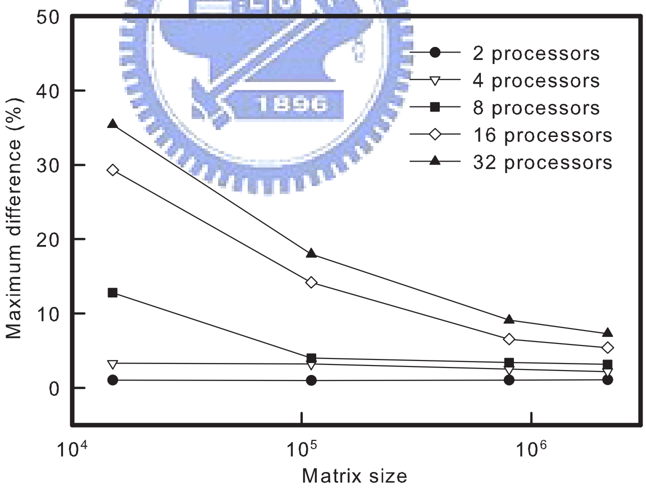


Figure 3.20: The achieved load balancing versus the matrix size for different number of processors.

with small matrix size due to the increased data communication time among the processors. According to the results, when the matrix size is less than 10^5, it is suitable to perform parallelization on the small number of processors (e.g., < 8) for maintaining optimal speedup and efficiency. As shown in Fig. 3.20, for the parallelization on 16 processors, the slope of efficiency is reduced when the matrix size is greater than 10^6, whereas the slope of efficiency of the parallelization on 32 processors still grows when the matrix size increases. The results imply that we may perform parallelization on 32 or more than 32 processors to gain an improved speedup without losing too much parallel efficiency when the matrix size is greater than 10^6. Figure 3.20 shows the maximum difference versus the matrix size for different number of processors. We notice that the maximum differences of the parallelization on 16 and 32 processors decrease significantly when the matrix size increases. We thus have an opportunity to achieve a better load balancing for those cases with large matrix size.

3.4 An Open-Source Project for Scientific Visualization

When visualizing a set of scientific or engineering results, people may be interested in a simple but vivid way to gain insight into the data in order to get the best possible impression of their content [139–141]. In this section, we introduce a new open-source project, Visi for high-dimensional engineering and scientific data visualization [139]. The main framework is based upon Qt (ver. 3.2.3) [142] and it integrates VTK (ver. 4.4) [143] as the rendering kernel. Visi is with state-of-the-art interactive user interface and graphics kernels. VTK is an open-source, cross-platform C++ library for 3D data visualization which has accumulated a large user base in the scientific computation research community. Moreover, Qt is a multi platform framework for building graphical user interface which supports OpenGL and has good performance by interacting with VTK *QVTKWidget*. Within an initialization procedure of Visi, a preliminary window will be activated by Qt, and the kernel of VTK is simultaneously embedded into the window, where the graphics resources are allocated. Visi working environment is now configured. Representation of visualization is established through an interactive interface so that the data will be rendered according to user's preference. This project is mainly developed by using C++, which significantly benefits the flexibility and extensibility of the programming.

3.4.1 Layout and Functionality

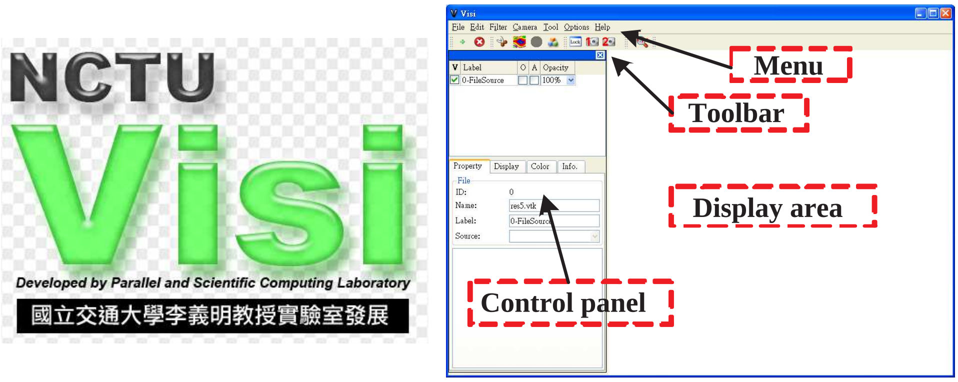
Figure 3.21(a) shows the logo of Visi developed by our laboratory and Fig. 3.21(b) depicts the execution window on Windows XP®. The layout includes four parts, **Menu**, **Toolbar**, **Control panel**, and **Display area**, where **Menu** contains all function shortcuts, and **Toolbar** gives a quick access to several most-used functions, such as ”adding data source” or

”performing CUT filter”. The detailed configurations of visualized objects in Visi are displayed in **Control panel**, shown in Fig. 3.21(b). It has multiple tabbed pages with Property, Display, Color, and Info. (abbrev. information) pages. Property page lists whole general file information and filter’s options. Display page shows the display-related functions for any visualized object, such as adjusting object’s opacity, adding/removing the outline box around the object, etc. In Color page, users can select different color suite to map to the input scalar data, or turn on/off the color bar. Properties for visualized objects, such as number of cells, points, or data ranges, are shown in Info. page. **Display area**, shown in Fig. 3.21(b), is designed to output visualized objects, where the real-time rendering engine depicts the visualized object. Visi provides various functions to inspect the objects in this area, for example, users can use mouse to rotate or zoom in/out the camera. The main features and the filters are briefed below.

3.4.2 Architecture and Visualization Procedure

An illustration of the architecture of developed classes in Visi is shown in Fig. 3.22. According to the functionality of the program, it can be separated into three categories: the maintenance-, GUI-, and visualization-related classes.

MainFrame class, shown in Fig. 3.22, is right the maintenance class in Visi which connects the human interface and the kernel of VTK. It includes two objects derived from


NCTU
Visi
Developed by Parallel and Scientific Computing Laboratory
國立交通大學李義明教授實驗室發展
Visi
File Edit Filter Camera Tool Options Help
V Label O A Opacity
0-FileSource 100%
Property Display Color Info.
File
ID: 0
Name: res5.vtk
Label: 0-FileSource
Source:
Menu
Toolbar
Display area
Control panel

(a) (b)

Figure 3.21: (a) Visi's logo and (b) its layout, where four tabbed pages, Property, Display, Color, and Info. are designed in Control panel.

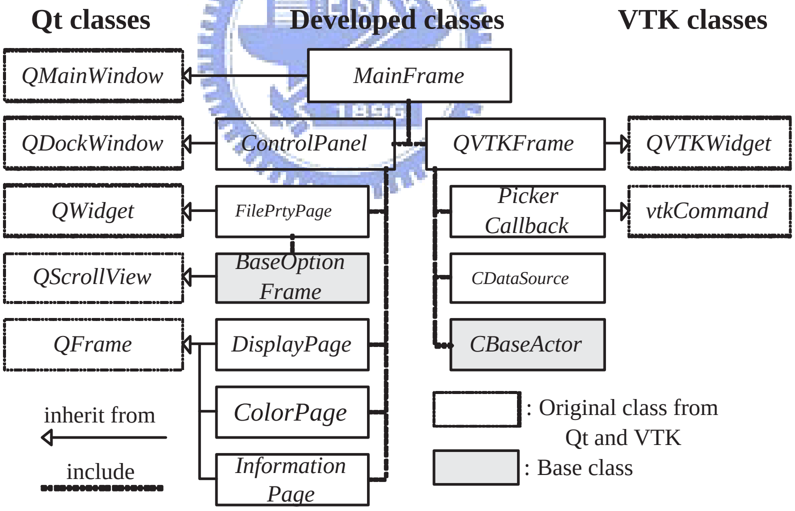


Figure 3.22: An architecture of the developed classes in Visi.

Filter	**Derived from *CBaseActor***	**Derived from *BaseOptionFrame***
Cut	***CCutActor***	***CutOptionFrame***
Clip	***CClipDatasetActor***	***ClipOptionFrame***
Contour	***CContourActor***	***ContourOptionFrame***
Threshold	***CThresholdActor***	***ThresholdOptionFrame***
Refinement	***CLoopSubdivisionActor***	***LoopSubdivisionOptionFrame***
Warp - Scalar	***CWarpScalarActor***	***WarpScalarOptionFrame***
SurfaceConstruction	***CDelauney 2DActor***	***Delauney2DOptionFrame***
Glyph	***CGlyphActor***	***GlyphOptionFrame***
Tube	***CTubeActor***	***TubeOptionFrame***

Figure 3.23: Filter functionalities of Visi and the corresponding implementation classes.

ControlPanel and *QVTKFrame* classes. Visualization-related category includes *QVTKFrame*, *PickCallback*, *CDataSource*, and *CBaseActor* classes. *CDataSource* is a general data reader to decode various data format for further processes. *CBaseActor* is the base class which provides a necessary interface for the basic visualized object. In Visi, all visualized objects can be regarded as actors which are derived from *CBaseActor*, shown in Fig. 3.23.

QVTKFrame manages all the graphics resources, such as the original VTK resource and developed visualized objects. It derived from *QVTKWidget* which provides an efficient way to embed into *MainFrame*. *PickCallBack* is a callback function object derived from *vtkCommand*. It provides a callback routine which is invoked by VTK kernel to get the information according to the location of mouse.

GUI-related group includes *ControlPanel*, *FilePrtyPage*, *BaseOptionFrame*, *DisplayPage*, *ColorPage*, and *InformationPage* classes. *ControlPanel* is a docking panel to display the property for each actor. *FilePrtyPage*, *DisplayPage*, *ColorPage*, and *InformationPage* are the property frames. It is noted that classes in GUI-related category are all derived from Qt, shown in Fig.3.22, such as *QDockWindow*, *QWidget*, *QScrollView*, and *QFrame*, respectively. *FilePrtyPage* is a sub-page to display the basic information of the actor. It includes the file and option frames. The file frame shows the file information, such as filename, path, and object id, etc. The option frame is a setting page which inherited from *BaseOptionFrame* for each filter. Figure 3.23 shows all derived classes of each filter. *DisplayPage* and *ColorPage* are used to adjust the appearance of the actor. *InformationPage* lists the geometric property of the actor (e.g., the object boundaries limits). We note that *BaseOptionFrame* and *CBaseActor* are base classes for filter construction.

Three steps are performed when visualizing the source data in Visi, shown in Fig. 3.24. For various kinds of the input data, *CDataSource* class firstly decodes the source. It then transforms the decoded data and generates a corresponding type of *CBaseActor* object. In the next step, users add proper Visi filter to *CBaseActor* object, and set the object properties in *ControlPanel*. *QVTKFrame* collects all objects above and invokes Visi engine, and then

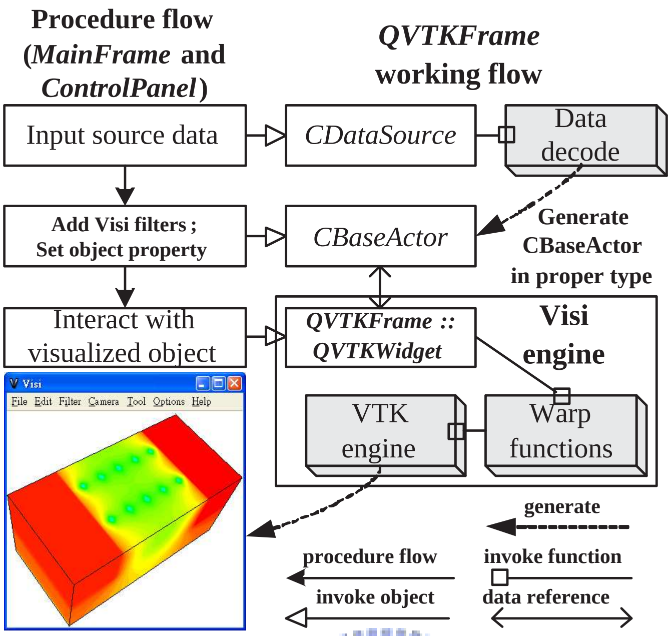


Figure 3.24: The procedure flow for manipulating Visi to visualize a source data, and the corresponding working flow in *QVTKFrame*.

it will call VTK engine to display the objects. Visi engine warps the original VTK classes including *VTKRender*, *VTKActor*, and *VTKPointPicker*, etc. In order to improve the flexibilities to fit our requirement in Visi, the redesign visualization procedure and developed visualized classes will be discussed later. In Fig. 3.24, *CDataSource*, *CBaseActor* and *QVTKFrame* are mainly handled by Visi, and *QVTKWidget* and other GUI-related classes are handled by Qt and VTK.

Based on VTK pipeline and developed data classes, Visi's visualization procedure exhibits much more extensibilities and flexibilities from the original VTK both in software development and tool interaction phases. VTK is a very large collection of C++ classes, and can be used to perform sophisticated scientific data processing and visualization tasks on a variety of data types. Most visualization systems are described through the analogy of a pipeline, where the data flows through a series of mathematical filter. A flowchart of VTK data pipeline is shown in Fig. 3.25.

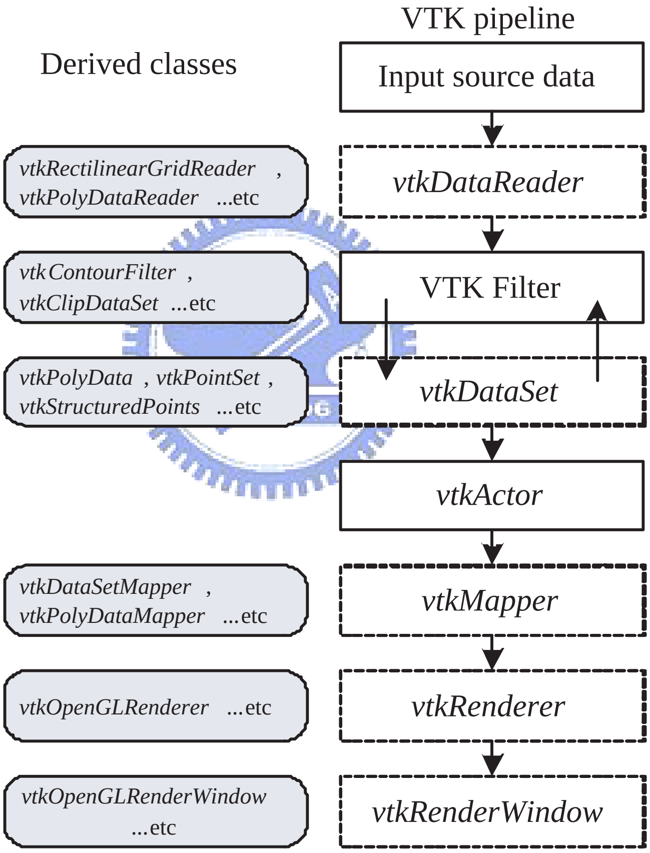


Figure 3.25: Flowcharts for VTK data pipeline, and the list of derived classes for each base class which marked with dash-line.

The original VTK library, shown in Fig. 3.25, provides various kinds of data readers (derived from *vtkDataReader*), filters (such as *vtkContourFilter* and *vtkClipDataSet*), data sets (derived from *vtkDataSet*), geometric mappers (e.g. *vtkDataSetMapper* and *vtkPolyDataMapper*), rendering objects (e.g. *vtkOpenGLRenderer*) and rendering window (derived from *vtkRenderWindow*). Through specific data reader and filter, the input source data can be processed and transformed into *vtkActor* object. Furthermore, the geometric mapper specifies interface between data (*vtkActor*) and graphics primitives. Once all *vtkActors* are ready, the *vtkRenderer* takes charge of rest display processes.

Major advantages of using VTK include greater modularity and ability to mix visualization metaphors. There are many filters, shown in Fig. 3.25, can be placed into the visualization pipeline; moreover, graphics generated by *vtkActor* can be combined with other ones. Particularly, those advantages are beneficial to software development level instead of application run-time level. Therefore, programmers can easily modify their own codes to manipulate advanced functions for high-dimensional data presentation. Unfortunately, it is somehow difficult for end-users to do such complicated operations in a run-time level; unless any developed applications can provide a convenient interactive interface and flexible visualization kernel.

In Visi, by introducing C++ programming techniques and developed data types, the improved data pipeline, shown in Fig. 3.26, which overcome the limitation above in the run-time level. Comparison shows that both VTK and Visi have similar architecture, shown in Fig. 3.25; however, the function blocks are replaced, such as *CDataSource* replaces *vtk-DataReader*, Visi filter replaces VTK filter, *CBaseActor* replaces *vtkDataSet* and *vtkActor*, and *QVTKFrame* takes care of the remain visualization process. In the following section, we discuss three major classes, *CDataSource*, *CBaseActor*, and *QVTKFrame*, respectively.

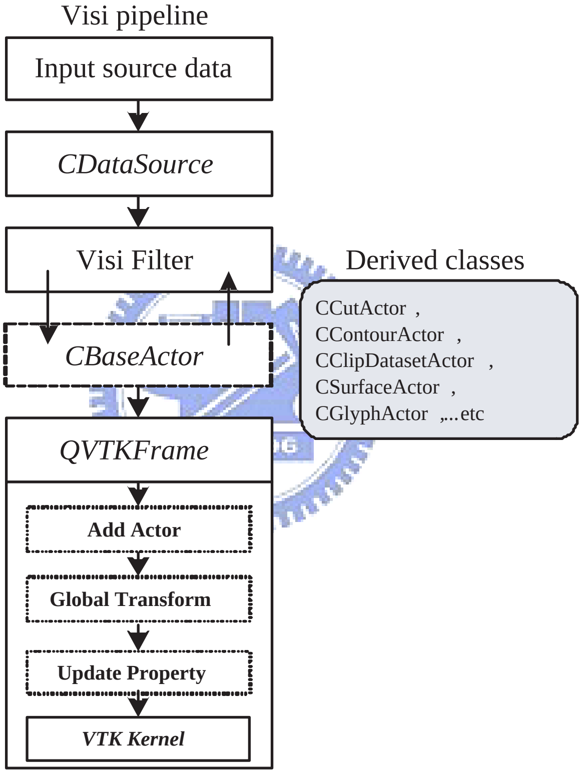


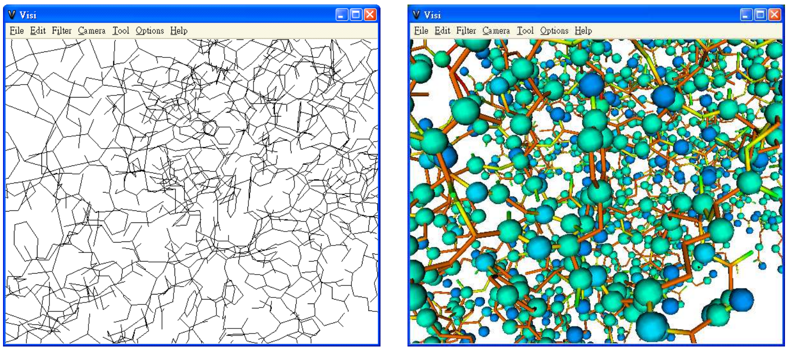
Figure 3.26: Flowcharts for the developed Visi data pipeline, and the list of derived classes for CBaseActor class. The base class is marked with dash-line, while the function routines are marked with dot-line.

3.4.3 Illustration Examples

In this section we apply Visi in several computational scientific and engineering fields to demonstrate the capabilities and flexibilities of presented project. Without loss of generality, all variables discussed in this section are dimensionless.

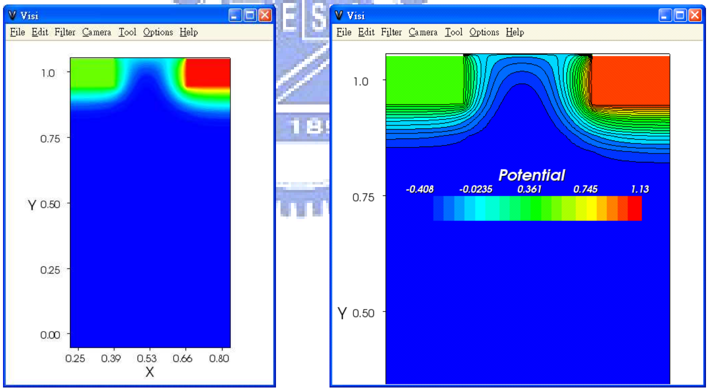
The protein data bank (pdb), in the bioinformatics, is a well-known and popular data format which is a repository for 3D structural data of proteins and nucleic acids. Currently, it is maintained by the research collaboratory for structural bioinformatics (RCSB). Through a VTK class, *vtkPDBReader*, Visi can directly load the file. A tertiary structure for catalytic antibody without applying any filters is shown in Fig. 3.27(a). In this third-level structure, the measured protein values include the atom types and peptide bonds. In Visi, different atom types, such as carbon, nitrogen, sulfur, and oxygen are presented as sphere in distinct colors by using Glyph filter, and the peptide bond can be express as a tube by applying Tube filter. The synthesis visualization image after applying Glyph and Tube filters is shown in Fig. 3.27(b). It is obvious that the latter image is much clear to express a protein sequence.

The Visi is also suitable for visualizations of 2D and 3D metal-oxide-semiconductor field effect transistor (MOSFET) simulation results. An electrostatic potential of a 2D n-type MOSFET device from technology computer-aided design (TCAD) simulation is shown in Fig. 3.28(a). In semiconductor device simulation, from the source side (green


Visi
File Edit Filter Camera Tool Options Help
Visi
File Edit Filter Camera Tool Options Help

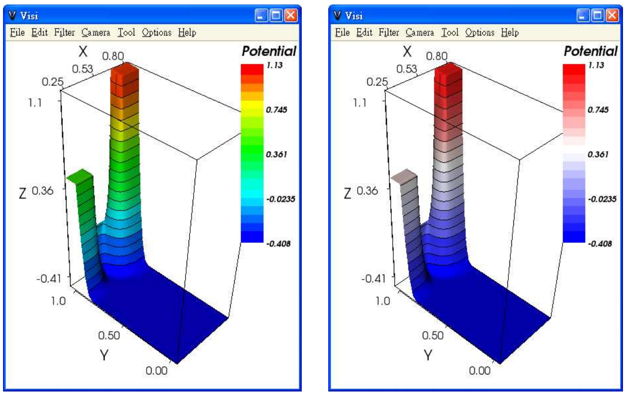
(a) (b)

Figure 3.27: The tertiary structure of a catalytic antibody. (a) The structure data without applying filters and (b) is the structure after applied Glyph and Tube filters.



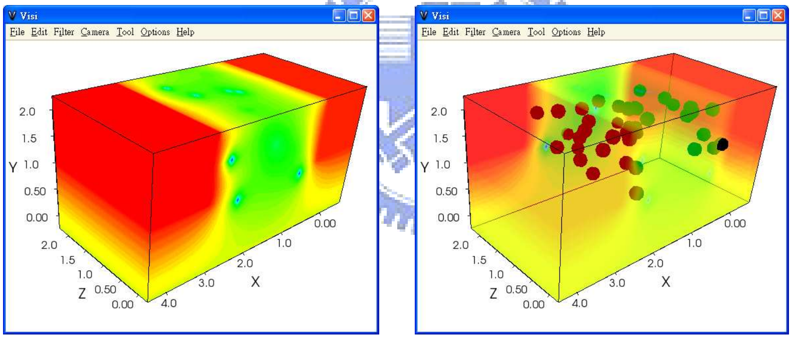
(a) (b)

Figure 3.28: (a) The potential profile for a 2D MOSFET device, and (b) the zoom-in plot in the channel region of MOSFET device with contour lines and scalar bar.



(a) (b)

Figure 3.29: The carpet plot for 2D potential profile by applying Warp-Scalar filter with different color mapping schemes.

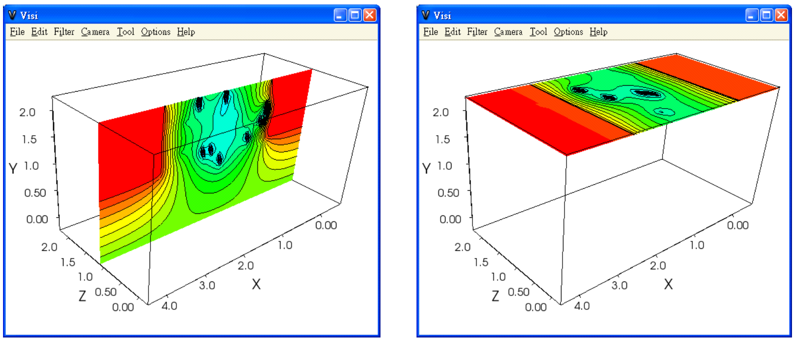


(a) (b)

Figure 3.30: (a) The surface potential plot of a 3D MOSFET device. (b) The transparent plot for potential profile and discrete implant doping atoms.

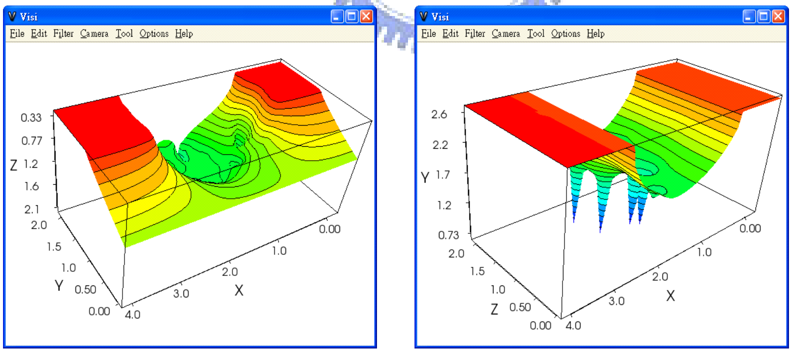
color) to the drain side (red color) horizontally, the potential's distributions and variations near the region of surface (i.e., $0.5 \leq y$) determine device characteristics. A zoom-in plot focusing on the upper half plane of the explored MOSFET with contours and scalar bar is shown in Fig. 3.28(b). With the help of Contour filter, the equal-potential lines of the computed solution profile are inspected easily. For the same 2D simulation data, the Warp-Scalar filter transforms 2D (x-y) with values to 3D (x-y-z) carpet plots, moreover, applied different color mapping scheme can express the original data vividly. Comparison of transformed carpet plots on potential profiles by applying Warp-Scalar filter with different color mapping schemes is shown in Fig. 3.29. For a 3D MOSFET simulation, it is difficult to exhibit the data in different angles, such as surface of simulation domain or inner cutting plane plot. The surface potential of a 3D MOSFET device with random-dopant-induced characteristic fluctuation is shown in Fig. 3.30(a). In this data set, discrete dopant particles are randomly distributed inside the device. Similarly, Glyph filter can be used to represent the random particles as a sphere, shown in Fig. 3.30(b). Visi combines two objects, surface plot for potential profile with 60% transparent and random particles, into a single visualized object. Benefitting from this feature, multiple objects can be integrated to represent a complex data. Furthermore, Cut filter extracts the simulated 3D potential profile from different cutting planes. For a fixed Z-position, X-Y cut plane and the corresponding contour are shown in Fig. 3.31(a). X-Z cut plane near the surface (i.e., $Y \approx 2.0$) is shown in Fig. 3.31(b). In those two figures, the contours indicate the regions have acute variations which are caused by random dopant particles. The 3D carpet surface plots generated by

Warp-Scalar filter from the extracted X-Y and X-Z plane are shown in Fig. 3.32.



(a) (b)

Figure 3.31: (a) X-Y and (b) X-Z cut planes of 3D MOSFET potential profile.



(a) (b)

Figure 3.32: The carpet plots for (a) X-Y and (b) X-Z cut planes.

In the following chapters, a serious of applications of UOF in the electronic design automation are presented. First we start from the 65nm CMOS device fabrication optimization. UOF was applied to search the optimal doping profile with respect to some targets, and this is a inverse problem. After the device fabrication, the equivalent circuit model is used to bridged the circuit design and chip fabrication. There are usually hundreds of parameters in the device model, and thus the equivalent circuit model parameter extraction is a complicated problem. In Chapter 5, the hybrid intelligent approach was explained for the parameter extraction problems. When a set of accurate parameters is extracted, it can be used for the circuit design. In Chapter 6, two circuit optimization methods are presented and compared. Finally, the design of antenna pattern applied for mobile broadcasting and the 802.11a WLAN is presented in the Chapter 7.

Chapter 4

Application of UOF in 65nm CMOS Device Fabrication

In this chapter, the utilization of UOF for inverse doping profile problems of the 65 nm complementary metal oxide semiconductor (CMOS) devices is presented. Fluctuation of electrical characteristics is simultaneously considered and minimized in the optimization procedure. Integration of device and process simulation is implemented to evaluate device performances, where the developed approach enables us to extract optimal recipes which are subject to targeted device specification. The quantum mechanical effects should be considered in the device simulation when the dimensions of the devices shrunk into nanometer scale. A set of Schrödinger-Poisson (SP) equations [52] has been applied to study the quantum effect in the inversion layers, but it is a time-consuming task in the application to realistic device characterization. Production of CMOS devices now enters the technology node of 65 nm; therefore, random-dopant-induced characteristic fluctuation should be minimized when a set of fabrication parameters is suggested. Verification of the optimization methodology is tested and performed for the 65 nm CMOS device. Compared with realistic fabricated and measured data, this approach can achieve the device characteristics; e.g., for the explored 65 nm n-type metal-oxide-semiconductor field effect transistor (MOSFET), the on-state current $> 0.35\ mA/\mu m$, the off-state current $< 1.5e - 11\ A/\mu m$, and the threshold voltage $= 0.43\ V$. Meanwhile, it reduces the threshold voltage fluctuation (vth $\sim 0.017\ V$). This approach provides an alternative to accelerate the tuning of process parameters and benefits manufacturing of nanoscale CMOS devices.

4.1 The Inverse Problems of the Semiconductor Devices

To pursue high performance CMOS devices for circuit designs, semiconductor manufacturing companies and foundries have to fabricate devices with different specifications. For example, logic, analog, and memory designs may require high threshold voltage, high speed, and low power devices. Process and device engineers experimentally split the process conditions, such as channel and source/drain implantations, and fabricate test samples in developing the required process modules. This empirical approach directs the engineer to realistic devices. However, as dimensions of MOSFETs enter the sub-65 nm era [40–44], proper use of computer simulation-and-optimization-aided manufacturing techniques can reduce the number of test runs. Diverse simulation and optimization methods including statistical schemes have been reported in optimal fabrication of CMOS devices [1,24,42,45–51,53,58–61]. The conventional methods use design of experiments (DOE) and external simulations to generate response surface model (RSM), and then analyze the statistical variation of process parameters to perform optimization of manufacturing process; and these methods have their advantages [46–49,60]. Fluctuation of electrical characteristics is evident in sub-65 nm technology [36,51,53–55,58,59]. Therefore, it will be a crucial investigation if the random-dopant-induced characteristics fluctuation can be simultaneously minimized in the solution procedure of the simulation-based optimization methodology.

Discovering the associated optimal configurations for a set of I-V curves of a given device forms an inverse problem [33]. This problem involves process and device simulations, and has sensitive parameters. Due to the above features, the problem currently plays an important role in technology development, as well as the performance diagnosis of nanoscale CMOS devices. In this application, the UOF with technology computer-aided design (TCAD) tools is adopted to deal with the inverse problems of nanoscale CMOS devices fabrication. The TCAD simulation integrates two-dimensional (2D) process simulation and device simulation. Threshold voltage fluctuations resulting from process variation effects and random dopant-induced effects should be taken into consideration and minimized in the optimization process. Here, we focus on the random dopant-induced fluctuations [50, 51, 58, 59]. The 2D TCAD simulation is sequentially performed to evaluate device performance using the developed optimization approach which enables us to extract optimal recipes subject to designed device specifications.

4.2 The TCAD Simulation for the Semiconductor Devices

The TCAD simulations, such as process and device simulations, are widely used for the analysis of semiconductor devices. The process simulation can generate the device geometry and doping profile according to the parameters of the fabrication processes. The output of process simulation is then used in the device simulation to estimate device characteristics. The drift-diffusion (DD) and hydrodynamic (HD) models play a crucial role in the development of semiconductor device simulator in the macroscopic point of view. The DD model [22, 23, 37, 62, 63] was derived from Maxwell's equation as well as charges' conservation law and has been successfully applied to study device transport behavior, in the past decades. It assumes local isothermal conditions and is still widely employed in semiconductor device design.

Classical drift-diffusion model consists of at least three coupled partial differential equations (PDEs) for, such as electrostatic potential and electron-hole densities. When device channel length is in the deep-submicron region, a set of the DD equations in semiconductor device simulation is solved:

$$\triangle\phi = \frac{q}{\varepsilon_s}(n - p + D), \tag{4.1}$$

$$\frac{1}{q}\nabla \cdot J_n = R(n, p), \tag{4.2}$$

$$\frac{1}{q}\nabla \cdot J_p = -R(n, p), \tag{4.3}$$

where ϕ is the electrostatic potential and its unit is volt. n and p are classical electron and hole concentrations (cm^{-3}). q is the elementary charge and its unit is coulomb. The net doping concentration is $D(x, y) = N_D^+(x, y) - N_A^-(x, y)$. R is the net recombination rate ($\mathrm{cm}^{-3}\mathrm{s}^{-1}$). The carrier's currents densities are given by

$$J_n = -q\mu_n n\nabla\phi + qD_n\nabla n + \mu_n k_B n\nabla T_n, \tag{4.4}$$

$$J_p = -q\mu_p p\nabla\phi + qD_p\nabla p + \mu_p k_B p\nabla T_p, \tag{4.5}$$

where μ_n and μ_p are the carrier mobility ($\mathrm{cm}^2/\mathrm{V}{-}\mathrm{s}$). The diffusion coefficients, D_n and D_p (cm^2/s), satisfy the Einstein relation.

The mobility model used in the device simulation, according to Mathiessen's rule [22, 64, 65], can be expressed as:

$$\frac{1}{\mu} = \frac{G}{\mu_{surf\_aps}} + \frac{G}{\mu_{surf\_rs}} + \frac{1}{\mu_{bilk}}, \tag{4.6}$$

where $G = exp(x/l_{crit})$, x is the distance from the interface and l_{crit} is a fitting parameter. The mobility consists of three parts: (1) the surface contribution due to acoustic phonon scattering

$$\mu_{surf\_aps} = \frac{B}{\mathbf{E}} + \frac{C(N_i/N_0)^{\tau}}{\mathbf{E}^{1/3}(T/T_0)^{K}}, \tag{4.7}$$

where $N_i = N_A + N_D$, $T_0 = 300K$, **E** is the transverse electric field normal to the interface of semiconductor and insulator, B and C are parameters which based on physically derived quantities, N_0 and τ are fitting parameters, T is lattice temperature, and K is the temperature dependence of the probability of surface phonon scattering; (2) the contribution attributed to surface roughness scattering is

$$\mu_{surf\_rs} = (\frac{(\mathbf{E}/\mathbf{E}_{ref})^{\Xi}}{\delta} + \frac{\mathbf{E}^3}{\eta})^{-1}, \tag{4.8}$$

where $\Xi = A + \frac{\alpha(n+p)N_{ref}^{v}}{(N_i+N_1)^{v}}$, $\mathbf{E}_{ref} = 1\ V/cm$ is a reference electric field to ensure a unitless numerator in μ_{surf_rs}, $N_{ref} = 1\ cm^{-3}$ is a reference doping concentration to cancel the unit of the term raised to the power v in the denominator of Ξ, δ is a constant that depends on the details of the technology, such as oxide growth conditions, $N_1 = 1\ cm^{-3}$, A, α and η are fitting parameters; (3) and the bulk mobility is

$$\mu_{bulk} = \mu_L(\frac{T}{T_0}^{-\xi}), \tag{4.9}$$

where μ_L is the mobility due to bulk phonon scattering and ξ is a fitting parameter.

The quantum mechanical effects [22, 45, 50–52] should be considered in the device simulation when the dimensions of the devices shrunk into nanometer scale. Various theoretical approaches have been presented to study the quantum confinement effects, such as full quantum mechanical model (e.g. nonequilibrium Green's function) and quantum corrections to the classical drift-diffusion (DD) or hydrodynamic (HD) transport models. A set of Schrödinger-Poisson (SP) equations has been applied to study the quantum effect in the inversion layers, but it is a time-consuming task in the application to realistic device characterization. Therefore, various quantum correction models, Hänsch, modified local density approximation (MLDA), effective potential (EP), density gradient (DG) and so on, have been proposed for classical DD or HD transport models. In this investigation, the density gradient was coupled with the DD model and solved for the quantum mechanical effects. The density gradient equation can be expressed as,

$$\vec{J_n} = -qn\mu_n \nabla\psi + qD_n \nabla n - qn\mu_n \nabla(2b_n \frac{\nabla^2 \sqrt{n}}{\sqrt{n}}) \tag{4.10}$$

where $b_n = \hbar^2/(12qm_n^*)$ and $m_n^* = m_k \times 9.11 \times 10^{-31}$ kg.

The threshold voltage fluctuation can be estimated by the perturbations technique [50, 51, 58, 59]. With the computed quantum corrected potential over entire device structure, the following perturbations to the Poisson equation are considered,

$$D(x_i, y_j) = \hat{D}(x_i, y_j) + \tilde{D}(x_i, y_j), \tag{4.11}$$

$$\phi_{QM}(x_i, y_j) = \hat{\phi}_{QM}(x_i, y_j) + \tilde{\phi}_{QM}(x_i, y_j), and \tag{4.12}$$

$$V_G = \hat{V_G} + \tilde{V_G},, \tag{4.13}$$

where D is the doping profile, ϕ_{QM} is the quantum corrected potential, and V_G is the applied gate voltage. Notation ˆ means that the expected value of D, ϕ_{QM}, and V_G, and ˜ is their fluctuations, respectively. The positions (x_i, y_j) for all (i, j) are the discretized mesh points. The fluctuation of the threshold voltage are computed with the randomly perturbed quantum potential. Therefore, the variance of the threshold voltage σ_{Vth} can be computed by assuming that the doping concentration is independent random variables for all (x_i, y_j).

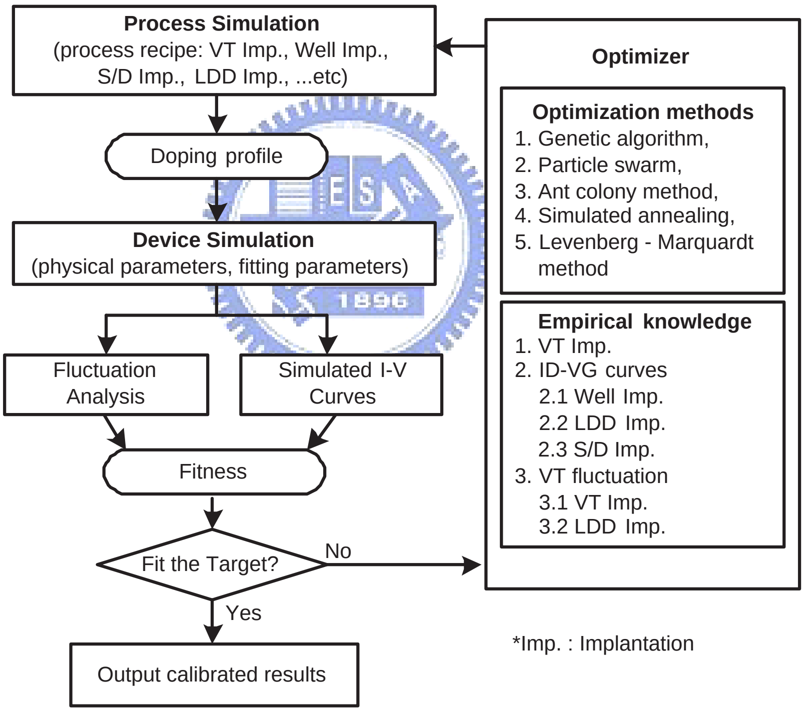


Figure 4.1: A working flow of the implement methodology for the reverse modeling problem in tuning fabrication parameters.

4.3 The Optimization Methodology

As shown in Fig. 4.1, a flowchart of the simulation-based optimization methodology illustrates the sequence of process and device simulations [22–24, 40, 41, 50, 61–63], fluctuation analysis [51, 53, 58, 59], and a optimization kernel [1, 6, 56, 57, 67–69]. For a given device specification, such as the on- and off-state currents and the threshold voltage, the simulator searches out a set of optimal settings to fit the prescribed target and then estimates the variation of the threshold voltage accordingly. Starting from an initial process recipe, a 2D process simulation is applied to generate the corresponding device structure and doping profile. Together with device parameters and physical models, the results of process simulation are used in the 2D device simulation to obtain the preliminary results. If we include the random-dopant induced characteristics fluctuation, we pass the physical results to perform the fluctuation analysis by solving the quantum correction model with perturbation and monotone iterative methods. At the same time, we calculate the error between the simulated result and the target to get the fitness (or the cost function). When the stopping tolerance is met, the solution procedure is terminated and the final results are outputted. Otherwise, the hybrid optimization is enabled to do the evolutionary searching process with respect to several specified constraints. The refined fabrication conditions as well as the physical model parameters are then used as inputs to the process simulation and device simulation is repeated. The iteration between the TCAD simulation and optimization is terminated when the simulated device's specification and the correspondingly computed tolerance of characteristic fluctuation meet the target. We note that to automatically search for the optimal recipes for device fabrication, the problem is now treated as a reverse modeling problem, which is a multidimensional minimization problem. It minimizes the errors between the specified (or measured) physical (and electrical) characteristics and the simulated results. The dimension of the optimization problem depends upon how many parameters are to be optimized; in general, it is about 30 process and device parameters. For ultra-small devices, 3D simulation should be considered to account for geometry effect. Figure 4.2 shows the target I-V curves to be optimized and several most concerned physical quantities empirically. Inset of Fig. 4.2 is a 2D cross-section view of the simulated 65nm MOSFET with LDD doping profile.

The developed evolutionary system for the semiconductor device fabrication contains two independent parts, the evolutionary core kernel and the external simulation programs, shown in Fig. 1. The former part mainly uses evolutionary algorithms, such as the genetic algorithm and the particle swarm method; and the external programs consist of the codes for device simulation and process simulation which can be replaced with any existing TCAD software. During the iterative procedure, the optimizer computes the fitness score for each setting (i.e., the process recipe) through a fitness function. The fitness function measures the error between simulated and target characteristics and the fluctuation of threshold voltage.

The fitness function used in this work is

$$fitness = weight_{ID}(\frac{log(I_D) - log(I_D^{target})}{log(I_D^{target})}) + weight_{\sigma vth}(\frac{\sigma vth}{V_T}). \tag{4.14}$$

where the I_D means the simulated data, the I_D^{target} is the specified target to be achieved, and σvth is the fluctuation of the threshold voltage (V_T). $weight_{ID}$ and $weight_{\sigma vth}$ are the weighted value for the I-V curves and σvth, respectively. In our work, we set $weight_{ID} = weight_{\sigma vth} = 1$, which means that the device performance and the fluctuation of the threshold voltage have the same weight. However, it can be adjusted according to different design purposes. To retrieve the simulated physical characteristics including I-V curves, the optimizer sends the setting to be evaluated to the external process and device simulation programs, and the external programs perform the simulations and generate the I-V curves. Physical-based empirical knowledge embedded in the evolutionary core kernel defines the relationship of the parameters and the device characteristics, as shown in Fig. 4.2.

According to engineering observation, parameters to be optimized are grouped into two categories, one is process-related and the other is device-related. The former part plays the important role of determining a device's preliminary characteristics. The I-V curves are physically divided into the linear, the off-state and the saturation regions. We first optimize process-related parameters by minimizing errors between simulation and target in the linear and off-state regions. To achieve error minimization, parameters relating to implantations

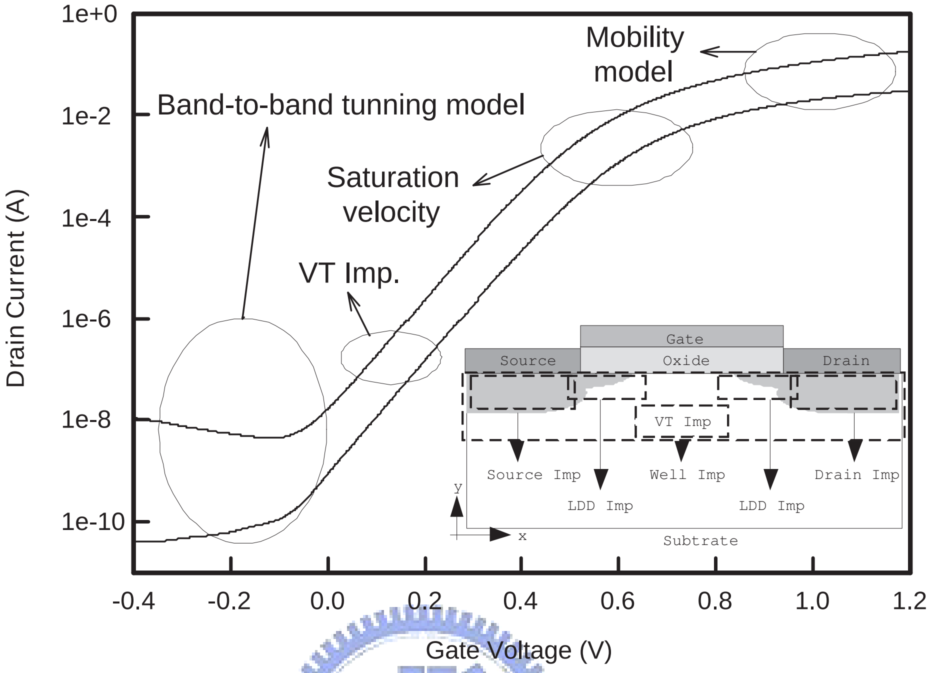


Figure 4.2: An illustration of the target I-V curves to be extracted and empirical knowledge. The important sections are pointed out in circles. The inset plot is a cross-section view of the simulated MOSFET with LDD doping profile.

of V_T, well, lightly doped drain (LDD), and source/drain are first computed simultaneously; allowing an accurate threshold voltage to be obtained. In the evolutionary part of the procedure, parameters coupled to band-to-band tunneling and saturation velocity are taken into consideration [22, 64–66]. Otherwise, parameters may possess unreasonable physical meanings, and then the optimization becomes meaningless. By minimizing errors in the saturation and off-state regions, device-related parameters are optimized with respect to the mobility model, band-to-band tunneling model, and saturation velocity [22, 64–66]. To reduce the fluctuation of the threshold voltage, we focus on V_T and LDD implantations. Finally, if necessary, the linear, off-state, and saturation regions are simultaneously optimized one more time. The optimization is terminated when errors are minimized for all I-V curves. We note that the device performance, in particular, for V_T and the linear region of I-V curves is significantly dominated by process-related parameters. Therefore, we put emphasis on the linear region and then the saturation region of I-V curves in the optimization procedure. During the optimization procedure, once a larger error within a certain region of I-V curves is observed, an empirical rule is employed to destroy the evolution, which may result in different mutation and is useful in the iteration loop of simulation and optimization.

Generally, the number of parameters to be optimized depends upon the device parameters selected in the TCAD simulation, and the process-related parameters that directly affect the device structure and doping profile. From empirical knowledge, we especially investigate the parameters for V_T, LDD and well implantations due to their significance in the threshold voltage and the linear region of the I-V curves. Moreover, the V_T and LDD implantations are crucial when considering the random-dopant-induced characteristics fluctuation. We also adjust the mobility and saturation velocity parameters in the device simulation for fine-tuning the I-V curves.

4.4 The Achieved Results and Discussion

In this section, the optimized parameters for on-target fabrication of the 65 nm CMOS devices will be investigated. For the sake of simplicity, we focus on the results of the 65 nm N-MOSFET. The device characteristics and the fluctuation tolerance of the threshold voltage are summarized in column 2 of Table 4.1. In this investigation, the on- and off-state currents, the threshold voltage as well as the device's fluctuation tolerance have been optimized simultaneously. With these four physical constraints, the coupled simulation and optimization methodology was self-consistently performed with the specified fitness function. The initial doping profile used in the optimization procedure is shown in Fig. 4.3. The simulated doping of the optimized device, both with and without the reduction of the threshold voltage fluctuation, are shown in Figs. 4.4a and 4.4b, respectively. Figure 4 shows a 1-D section in the center of the device channel of the 2-D doping profile distributions shown in Figs. 4.3 and 4.4. It can be observed that the minimization of the threshold voltage fluctuation (dotted line) results in a lower doping level compared with the optimization without (dashed line). This optimization automatically achieved the target parameters. Fig. 4.6 shows the horizontal doping profile 2nm below the channel surface between the source and drain. Together with the results, shown in Fig. 4.5, the higher doping level along the channel for the device when the threshold voltage fluctuations were also optimized maintains the same device characteristics as the one where dopant fluctuations were not

Table 4.1: A comparison list of the achieved results with respect to the two different extractions. The target specification is adopted from realistic fabricated and measured data.

	Target to be achieved	Result without σ_{vth} reduction	Result with σ_{vth} reduction
Ion (mA/μm)	> 0.35	0.35	0.39
$Ioff$ (mA/μm)	$<$ 1.5e-11	1.05e-11	1.13e-11
V_{th} (V)	$\sim$ 0.436	0.442	0.432
σ_{vth} (V)	$\sim$ 0.017	0.03	0.014
σ_{vth}/V_{th} (%)	3.89	6.78	3.24

considered.

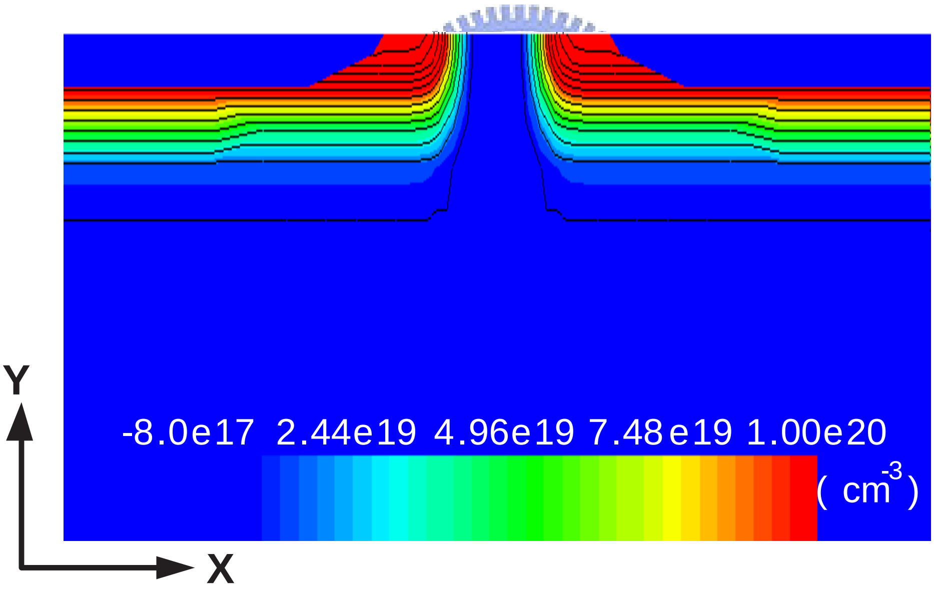


Figure 4.3: An illustration of the initial doping profile for the explored 65 nm N-MOSFET.

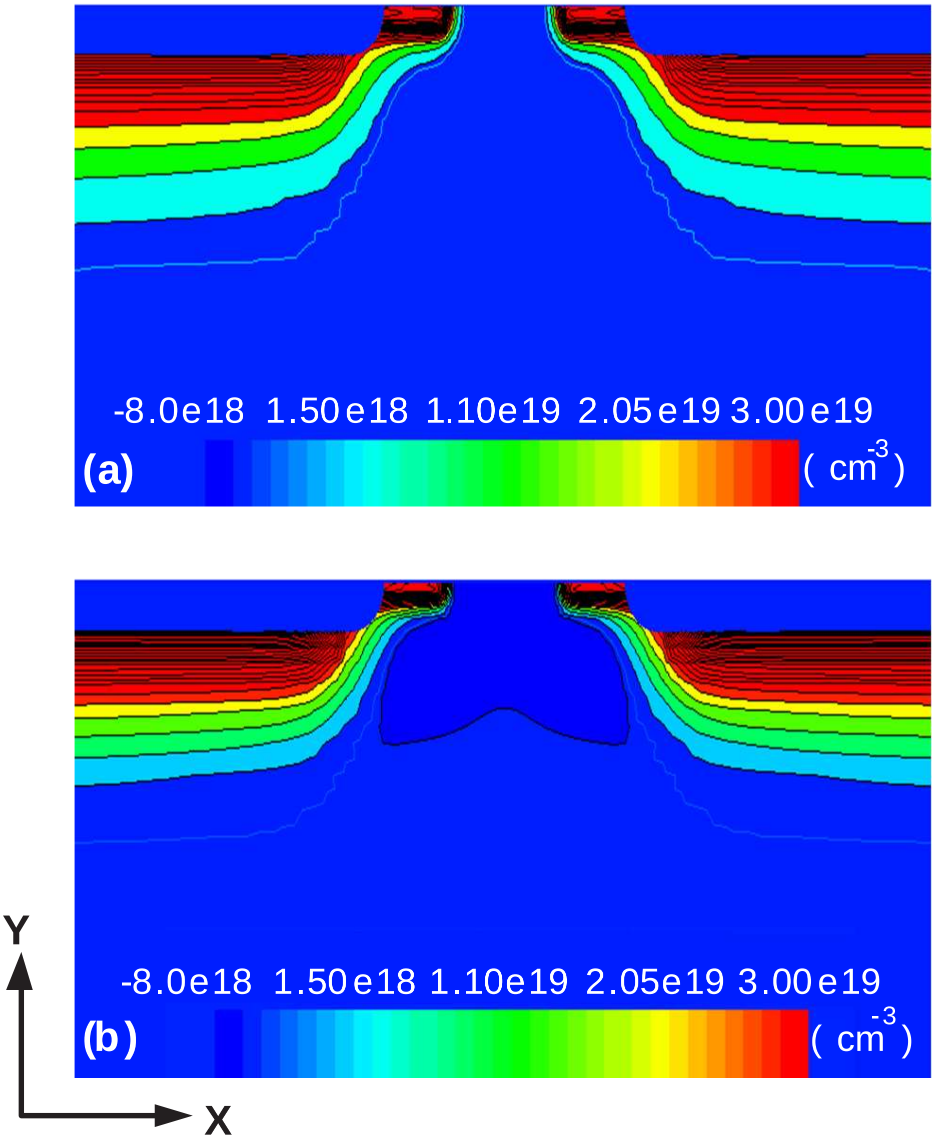


Figure 4.4: The optimized 65 nm N-MOSFET doping profile for the device (a) without and (b) with considering the minimization of the threshold voltage fluctuation.

Table 4.2: A list of process recipe and device parameters for the device optimization w/ and w/o considering the fluctuation reduction.

Parameters	Parameters Range	Result without σ_{vth} reduction	Result with σ_{vth} reduction
Core VT implantation	Energy: 20∼80 KeV	58	24
	Dose: 1e12∼2e13 cm^{-2}	2.7e12	1.3e13
N-LDD implantation	Energy: 10∼50 KeV	28	29
	Dose: 1e12∼5e13 cm^{-2}	3.1e13	3.8e13
P-WELL implantation	Energy: 200∼400 KeV	250	250
	Dose: 1e13∼4e13 cm^{-2}	2.4e13	2.4e13
Mobility model	B: 2e7∼8e7 cm/s	3.5e7	3.41e7
	C: 100∼500 $cm^{5/3}/(V^{2/3}S)$	160	170
Velocity saturation	V_{sat0}: 1e6∼1e8 cm/s	9e6	9e6
	V_{sat1}: 0.5∼1.0	0.81	0.82

The results of the optimized 65 nm N-MOSFET are summarized in Table 4.1. If the threshold voltage fluctuation minimization is not be activated in the optimization, there is 6.7% threshold voltage fluctuation, which significantly shifts the process away from the design window (3.89%). The band profiles for the devices in the on- and off-states are shown in Figs. 4.7a and 4.8a for both of the optimization cases. The optimization without minimization of the threshold voltage fluctuation has a little bit lower band edge than its counterpart due to a higher doping level (Figs. 4.7a and 4.8a). Figs. 4.7b and 4.8b, once again show the horizontal profile along channel for both optimization cases. For the optimization without minimization of fluctuation, both the on- and off-state bands (2 nm below the channel surface) are lower than that the optimization minimizing the fluctuations. Both the specified target and the optimized results are shown in Fig. 4.9. The results with and without considering the threshold voltage fluctuation are very close to the specified target. Nevertheless, the strategy of including the reduction of threshold voltage fluctuation successfully reduced the threshold voltage fluctuation and is shown in Table 4.1. A list of the process recipe and device parameters used for simulation, and the extracted parameters, with and without fluctuation reduction, are summarized in Table 4.2. It can be noted that there is a major difference between the core VT implantation and the result with and without σ_{vth} reduction. To verify the efficiency of the proposed method, three examinations are performed on our PC-based Linux cluster system with 16 processors. The fitness score versus the number of evolutionary generations is shown in Fig. 4.10. For the given target, it shows that the methodology with simultaneously considering the parameters of the process and device physics provides better computational efficiency.

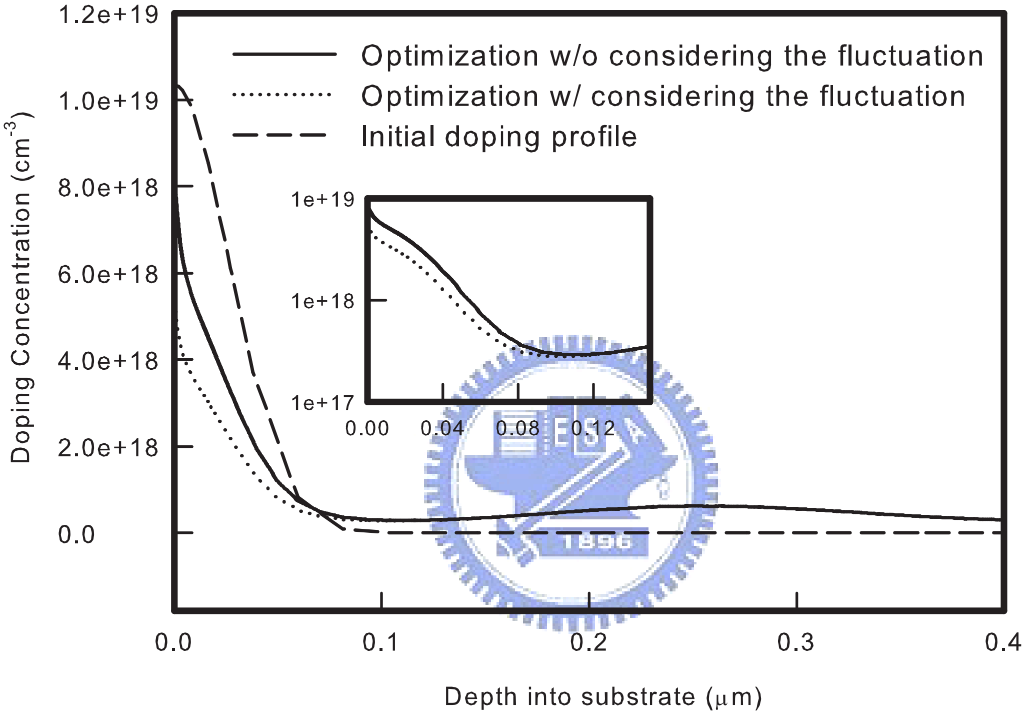


Figure 4.5: The optimized doping profiles from the channel surface deep into the substrate. The 1-section is located at the center of device channel (x = 0).

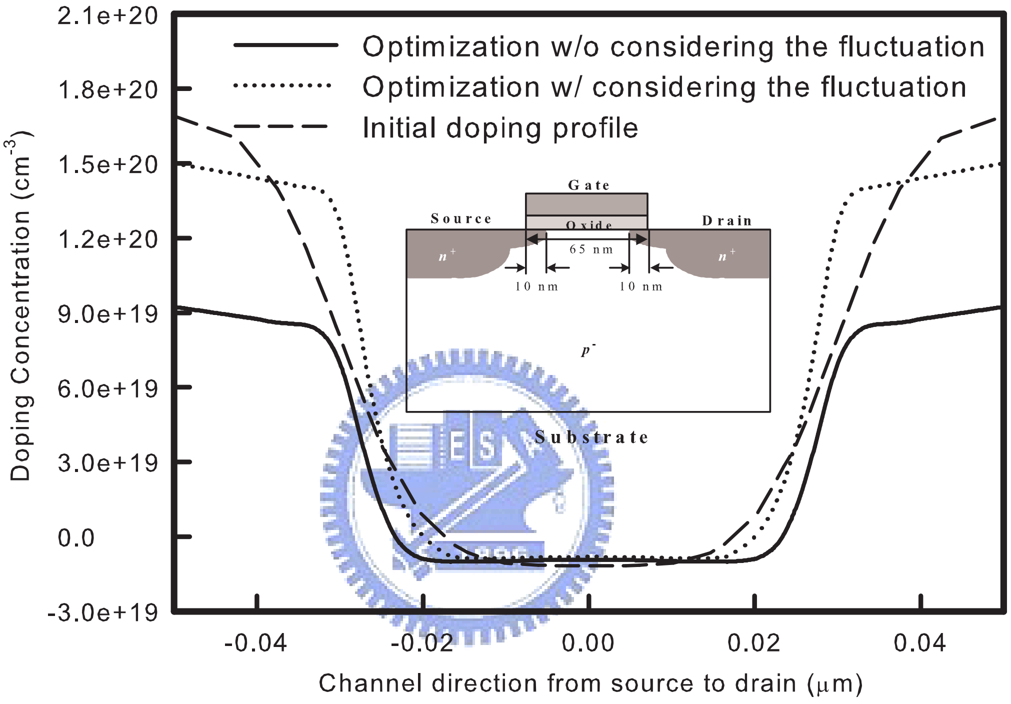


Figure 4.6: Doping profile from the source to drain along the channel direction 2 nm below the interface between the gate oxide and the silicon substrate. The inset of the figure shows the structure of the optimized 65 nm N-MOSFET.

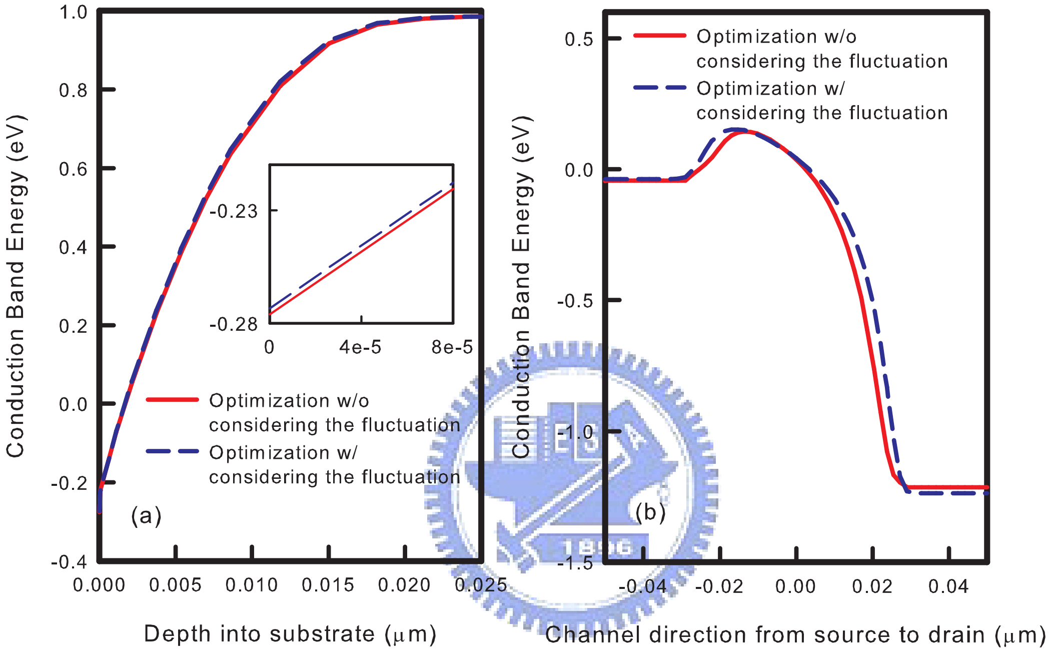


Figure 4.7: Plots of band profile for the optimization with and without considering the threshold voltage fluctuation under the on-state; (a) is from the surface to the substrate and (b) is along the channel direction from the source to the drain which is about 2 nm below the channel surface.

1896

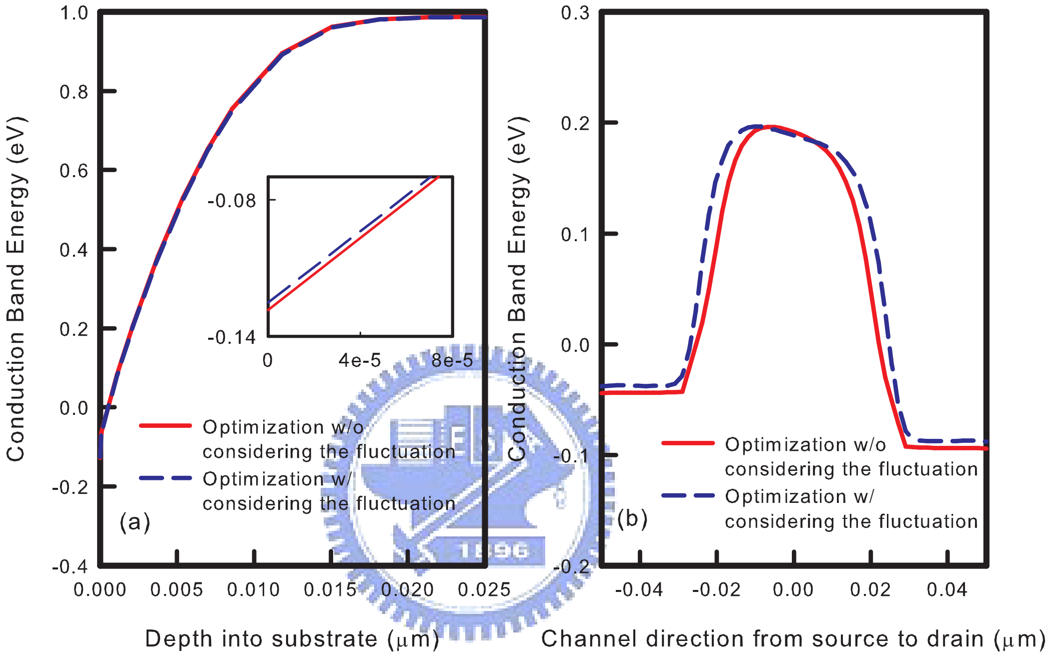


Figure 4.8: Plots of band profile for the optimization with and without considering the threshold voltage fluctuation under the off-state; (a) is from the surface to the substrate and (b) is along the channel direction from the source to the drain which is about 2 nm below the channel surface.

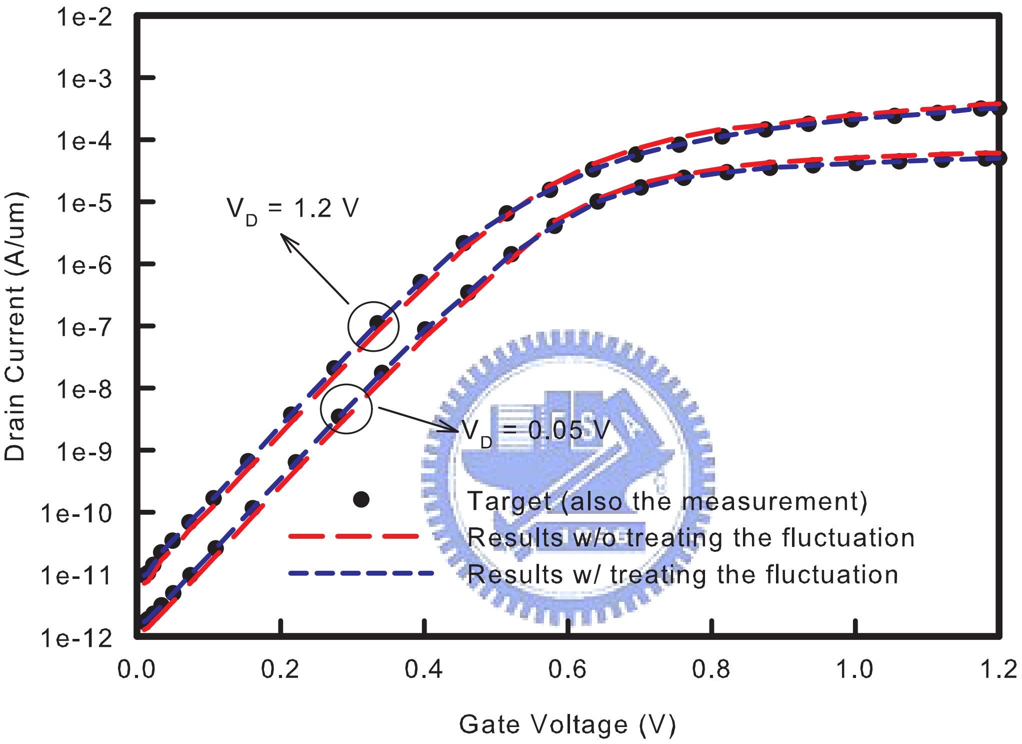


Figure 4.9: The achieved accuracy of the extracted I-V curves for the explored 65 nm N-MOSFET.

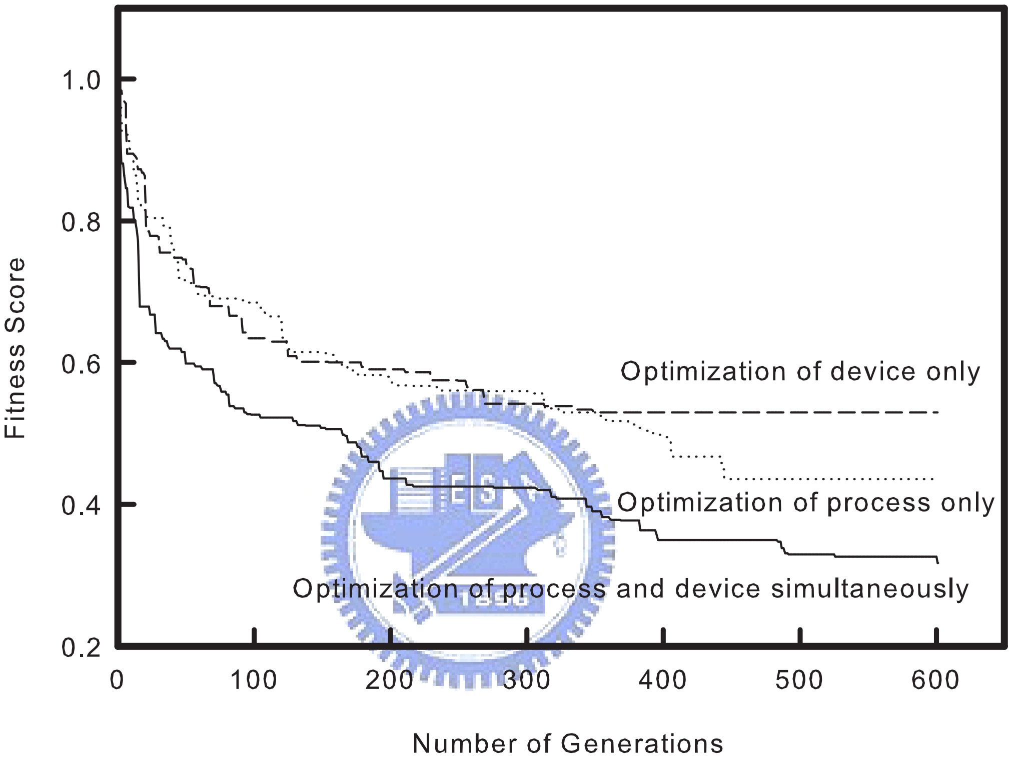


Figure 4.10: The performance comparisons among three different evolutionary strategies. There are totally 31 process and device parameters to be optimized in the case of the 2D process and device simulations. The total time is about 70 hours on a PC-based Linux cluster with 16 processors.

Chapter 5

Application of UOF in VLSI Device Model Parameter Extraction

Device model associated with a set of optimized parameters currently plays a central role in the connection between circuit design and chip fabrication communities, as shown in Fig. 5.1. An automatic model parameter extraction system that simultaneously integrates evolutionary and numerical optimization techniques for optimal characterization of very large scale integration (VLSI) devices has recently been advanced. In this chapter, a hybrid intelligent general solution method for compact model parameter extraction is proposed. This solution technique combines the genetic algorithm (GA), the neural network (NN), and the Levenberg-Marquardt (LM) method for optimal I-V curves characterization, optimization, and parameter extraction of deep-submicron metal-oxide-semiconductor field effect transistors (MOSFETs). This unified intelligent computing technique can extract a set of corresponding optimal parameters for compact models. The well-known BSIM-4 and EKV MOSFET compact models have been studied and implemented for automatic parameters extraction using the proposed intelligent architecture. The further investigation of the random number generations in the GA is also presented in this chapter.

5.1 The Equivalent Circuit Model Parameter Extraction Problems

The simulation program with integrated circuit emphasis (SPICE) models, such as BSIM, HiSIM, and PSP models characterize VLSI device's electrical characteristics (e.g., I-V curves), which are associated with a set of optimized parameters. However, in semiconductor device simulation, the setting for each construction parameter is always a complicated problem. The process that fits the simulation data as closely as possible to measured data is called the parameter extraction. For the problem of the SPICE model parameter extraction, it usually refers to several hundred I-V points. It is not only a time consuming task but also requires engineering expertise to find a proper configuration of parameters with reasonable physical meanings. Model parameter extraction techniques have been of great interest in

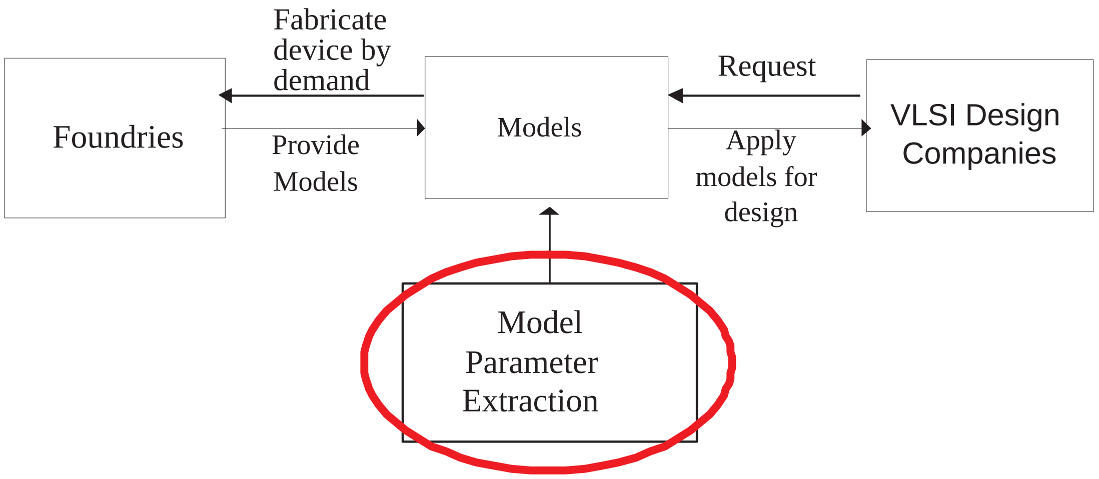


Figure 5.1: An illustration of the equivalent circuit model in connecting the device fabrication and circuit design.

1896

the last decade. Most of them have been proposed which include traditional numerical optimization methods and modern soft computing techniques [75–77] to solve such problems with huge searching space. The model parameter extraction problem can be regarded as a multidimensional optimization problem to minimize the errors between the measured and simulated results.Various compact models have been of great interest and studied for deep-submicron and sub-100 nanometer device simulation [70–74]. An equivalent circuit model and corresponding parameter extraction intrinsically characterizes the properties of designed and fabricated devices. The model parameter extraction problem can be regarded as a multidimensional optimization problem to minimize the errors between the measured and simulated results.

The solution technique proposed in this chapter mainly consider the GA, the NN, and the LM method in optimal I-V curves characterization, optimization, and parameter extraction of deep-submicron MOSFETs. This unified intelligent computing technique can extract a set of corresponding optimal parameters for compact models. The well-known BSIM-4 and EKV MOSFET compact models have been studied and implemented for automatic parameters extraction using the proposed intelligent architecture.

5.2 The Hybrid Optimization Methodology

The main goal of device model parameter extraction is to seek a set of parameters so that it minimizes the error between the simulated result and the measured data, where the extracted result is theoretically obtained through the equation

$$I_{DS}^{ex} = I_D(\vec{p}, \vec{v}, \vec{d}), \tag{5.1}$$

where I_{DS}^{ex} is the I-V functions (e.g., I-V points) to be optimized; the I_D is any selected compact model. Vectors $\vec{p}$, $\vec{v}$ and $\vec{d}$ are the parameter sets to be extracted, the bias condition for simulation, and the device geometry, respectively. The proposed methodology integrates the genetic algorithm and the neural network, and the Levenberg-Marquardt method [75–77]. GA is a global search optimization method based on the mechanics of natural selection and natural genetics. The NN is an adaptive learn network which has

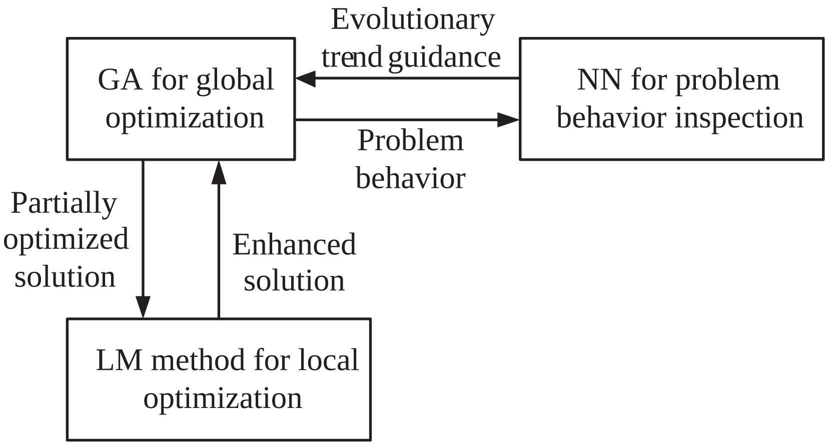

Evolutionary trend guidance
GA for global optimization
NN for problem behavior inspection
Problem behavior
Partially optimized solution
Enhanced solution
LM method for local optimization

Figure 5.2: An illustration of the proposed hybrid intelligent computational methodology.

the remarkable ability to derive meaning from complicated or imprecise data. It has been wildly used in various rages especially in pattern reorganizations and the image processing. In this work, we adopt Hamming net to guide GAs to search the better solutions. The LM method is a quasi-Newton method to accelerate the Gauss-Newton method.

The execution flowchart of the hybrid intelligent computational technique for the extraction task is shown in Fig. 5.2. As shown in this figure, GA searches the entire problem space to get the roughly estimated solution firstly. After a rough solution is obtained, LM method makes a local optima search and sets the local optima as the suggested values for GA to perform further optimizations. Meanwhile, the NN is applied to investigate the influence of parameters to the optimized functions, and guides GA to focus on some major parameters to obtain the better solutions instead of performing blind search. The NN compares the difference of measured and simulated curves features and suggest that GA should focus on the corresponding parameters. Typical genetic search methods are plagued by problems such as rapid decreases in the population diversity and disproportionate exploitation and exploration of the solution space with multiple dimensions. The results are frequent premature convergence and inefficient search. Although LM method is a modified Gauss-Newton method, it is still a local method which is easily trapped into local optima. However, comparing with the global optimization technique such as GA, LM method finds a solution rapidly. With the adoption of LM method, GA can save much unnecessary effort searching optima in local area. On the other hand, there are some major physical quantities influences the others. If these major physical quantities are intolerant, other characteristics will not be correct. Therefore, the parameters which affect these major quantities should be extracted firstly and the priority of model parameters should be considered. Besides, each physical quantity affects some specified I-V curves characteristics such that we can be conscious of the intolerance of physical quantities through investigating the characteristics of I-V curves. The information described above is built in our NN and under the guidance of NN, GA emphasizes the most important parameters and corrects physical quantities one by one. The hybrid optimization algorithm is shown below.

```
Begin Hybrid Optimization Algorithm
```

```
    Initialize parameters extraction environment
    Begin GA optimization
      Initialize GA
      While EstimatedError(BestSolution) $>$ ToleranceError
        GA performs ParametersExtractionOptimization
        GA obtain BestSolution
        LM ParametersExtractionOptimization(BestSolution)
        NN ModelInspection(BestSolution)
      End While
    End GA optimization
End Hybrid Optimization Algorithm
```

1896

This unified framework exhibits very effective and robust automatic optimization function for different compact models' parameter extraction. Taking the well-known compact models, BSIM and EKV models as examples, there are more than one hundred parameters in BSIM-4 and 30 parameters in EKV model have been involved in the model parameter extraction. We investigate the accuracy and efficiency of the proposed hybrid intelligent computational technique by considering both BSIM-4 and EKV MOSFET compact models. Shown in the next section, numerical results confirm the proposed approach is superior to the others pure numerical or evolutional methods.

5.3 The Empirical Rules for Equivalent Circuit Model Parameter Extraction

For the model parameter extraction problems, there are some empirical knowledge can be applied to accelerate the parameter extraction process. These empirical knowledge plays an important role in the parameter extraction, and lack of the experience usually make the extraction task more difficult. In general, the empirical rules are resulted from the characteristics of the model and the physical or mathematical meanings of the parameters. Based on our experience and the investigation of the models, some empirical extraction strategies are proposed.

According the physical point of view, the parameters can be divided into several categories, which include threshold voltage, mobility model, drain current, subthreshold region, drain-source resistance, channel geometry, output resistance, capacitance, process related and temperature modeling. Sensitivity of the parameters to be extracted is one of important issues for assisting parameter extraction. The sensitivity examination of parameters can point out what kind of parameters affects behavior of convergence significantly. According to this information, we firstly extract those most sensitive parameters. When these parameters are firstly decided, all parameters will be extracted simultaneously. To perform the sensitivity analysis, the developed system extracts single parameters category meanwhile

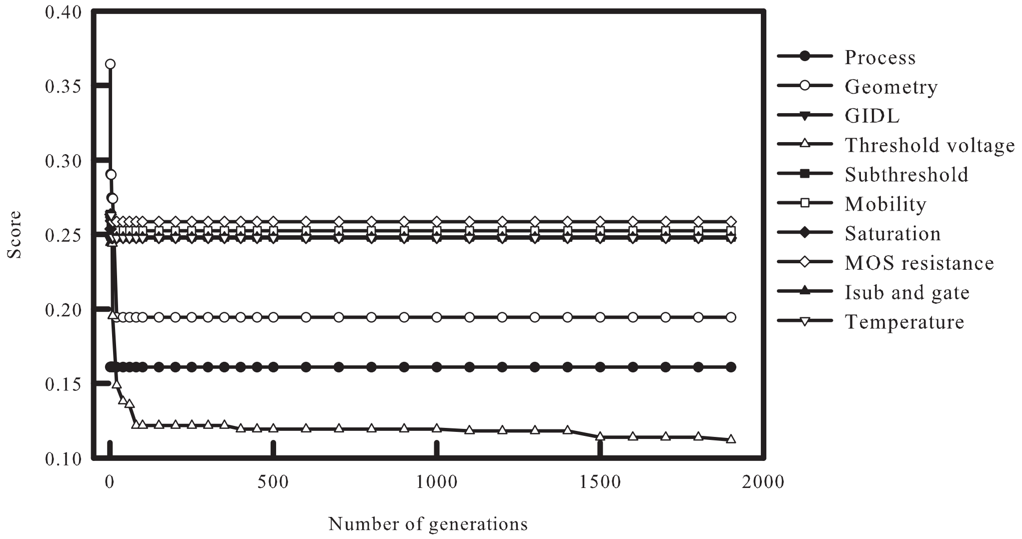


Figure 5.3: The sensitivities examination of the parameters to be extracted in the compact model.

locks other parameters. The expected result should show that varying certain parameters category would make notable progress while some others would not. Figure 5.3 shows the sensitivities examination of the extracted parameters in the compact model and reveals that the parameters related to the threshold voltage would make the most improvement. The parameters in each category can be further divided into three types: parameters with physical meaning, global parameters and local parameters. For example, the parameters with physical meaning in category of threshold voltage include Vth0 and VBM. The global parameters are K1, K2, DVT1 and DSUB. The local parameters includes K3, K3B, W0, ... etc. The parameters with physical meaning can be decided firstly, and then adjust the

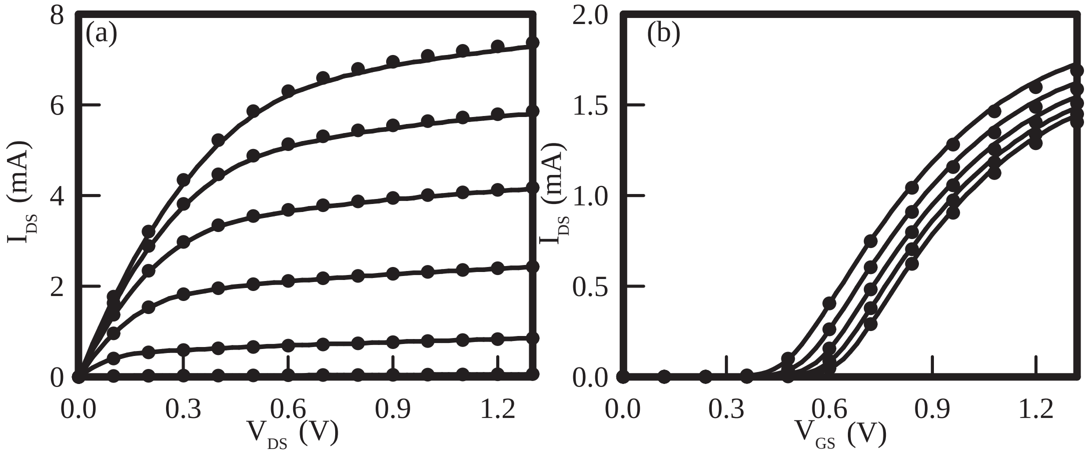


Figure 5.4: The BSIM-4 extracted (solid-line) and measured (dot-lines) $I_{DS} - V_{DS}$ and $I_{DS} - V_{GS}$ curves of the 90 nm MOSFET (width = 10.0 μm), where V_{BS} = 0 V and V_{GS} varies from 0.4 to 1.4 V, and V_{DS} = 0.1 V and V_{BS} varies is 0 to -1.5 V.

global and local parameters to fit the targets. In some category, the parameters with physical meaning can be can be obtained from measurement data, such as the Vsat in the drain current category. For some categories, they may have only one or two types of parameters. In the channel geometry category, there are only local parameters, but it still can have some rules for parameter extraction. To extract the parameters in the channel geometry category, the parameters Wint and Lint should be extracted until the extracted results has better shape, then start to extract other parameters.

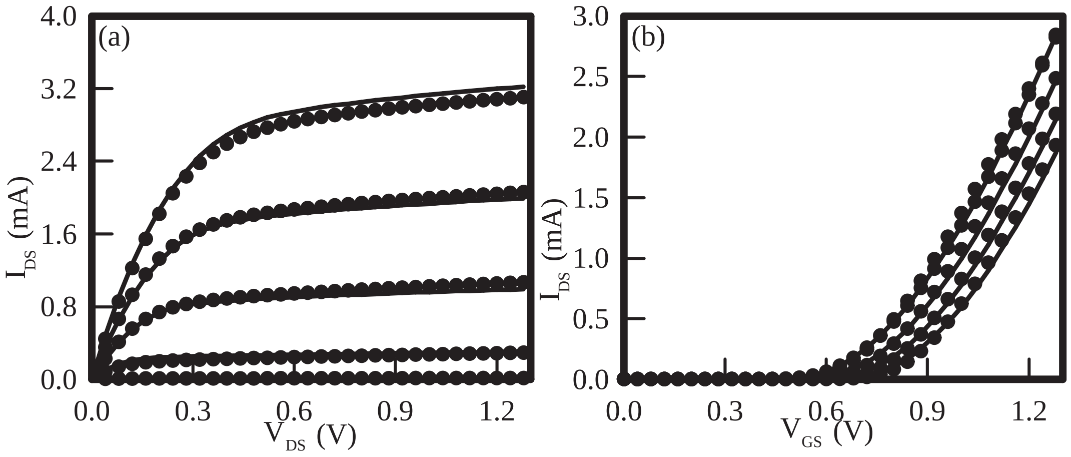


Figure 5.5: The EKV extracted (solid-line) and measured (dot-lines) $I_{DS} - V_{DS}$ and $I_{DS} - V_{GS}$ curves are of the 0.18 μm MOSFET, where V_{BS} = -0.6V and, V_{GS} migrates from 0.4 to 1.4 V, and V_{DS} = 1.3V and V_{BS} migrates is 0 to -0.9 V.

1896

5.4 The Achieved Results and Discussion

The achieved accuracies of the extracted model parameters are given in Figs. 5.4 and 5.5 for both BSIM-4 and EKV models. In Fig. 5.4a represents the $I_{DS} - V_{DS}$ curves and Fig. 5.4b stands for $I_{DS} - V_{GS}$ curves; the width and channel length of the target device is equal to 10 μm and 90 nm. On the other hand, the width and channel length of the target device is equal to 10 μm and 180 nm in Fig. 5.5;similarly, Fig. 5.5a represents the $I_{DS} - V_{DS}$ curves and Fig. 5.5b stands for $I_{DS} - V_{GS}$ curves. The errors between measured and extracted I-V curves are less than 3$\%$. All figures shown above have demonstrated the accuracy of the proposed method. In the next part, the efficiency of the proposed method is discussed.

Figure 5.6 is a fitness score comparison with different extraction methods for BSIM-4 model. As shown in Fig. 5.6, without the guidance of NN, the methods of GA only and GA+LM spent a lot of time to reduce the fitness score down to 0.5, while the ones with NN can easily shot this problem. Comparing the GA+LM and GA+NN+LM method in Fig. 5.6, the NN detects the differences between measured and extracted curves, and suggest a better extraction direction to GA to perform the fine tune task among the curves and its corresponding relative parameters. The results indicate that the evolutionary process with the guidance of NN really shows the better convergence behavior, and they confirm the great efficiency of the method. The result confirms the efficiency of the method. Fig. 5.7 shows a comparison of the score convergence behavior of our proposed method with pure GA for different two compact models. The EKV compact model can more quickly achieve a lower score than the BSIM-4 model due to less parameters, but it's final results are not better than the results of BSIM-4 model after lots of generations. This figure shows our method has superiority and robustness for both BSIM-4 and EKV compact models. We note that, shown in Figs. 5.6 and 5.7, when the other methods appear to be saturated, our method can continuously improve the fitness score. Experiments in this work preliminarily confirm that the proposed method solves complicated multidimensional optimization problem effectively and provides an alternative for deep-submicron and nanoscale device model parameter extraction.

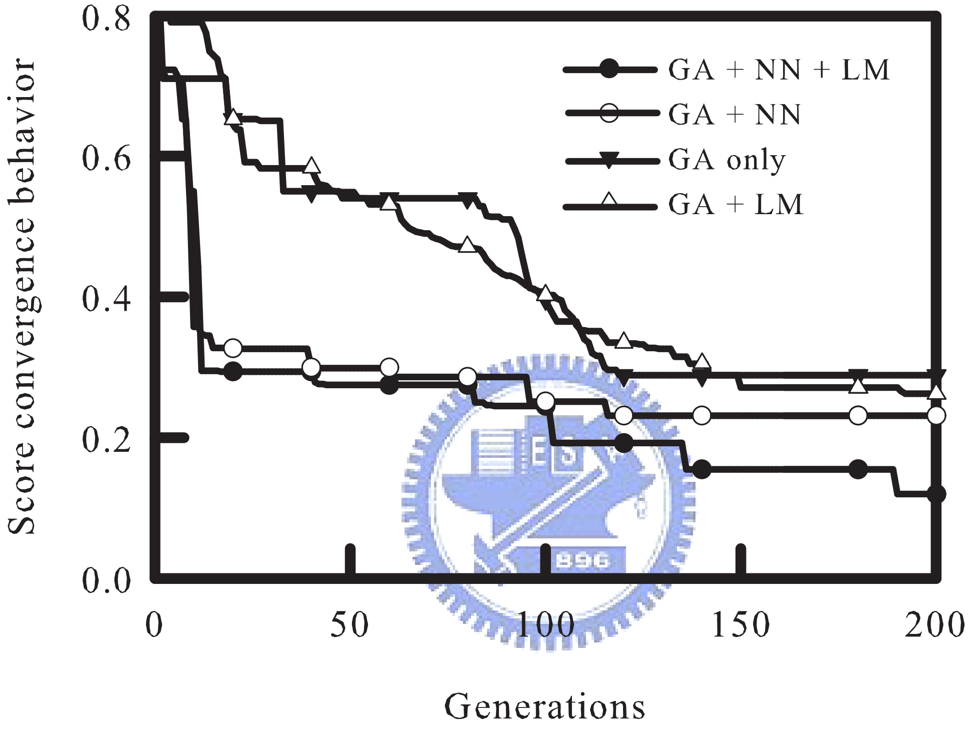


Figure 5.6: (a) the score convergence of different extraction methods, and (b)the score convergence behavior w/ and w/o applying NN as a director during the evolutionary process, where the testing is with BSIM-4 model applying to four 0.13 μm MOSFETs.

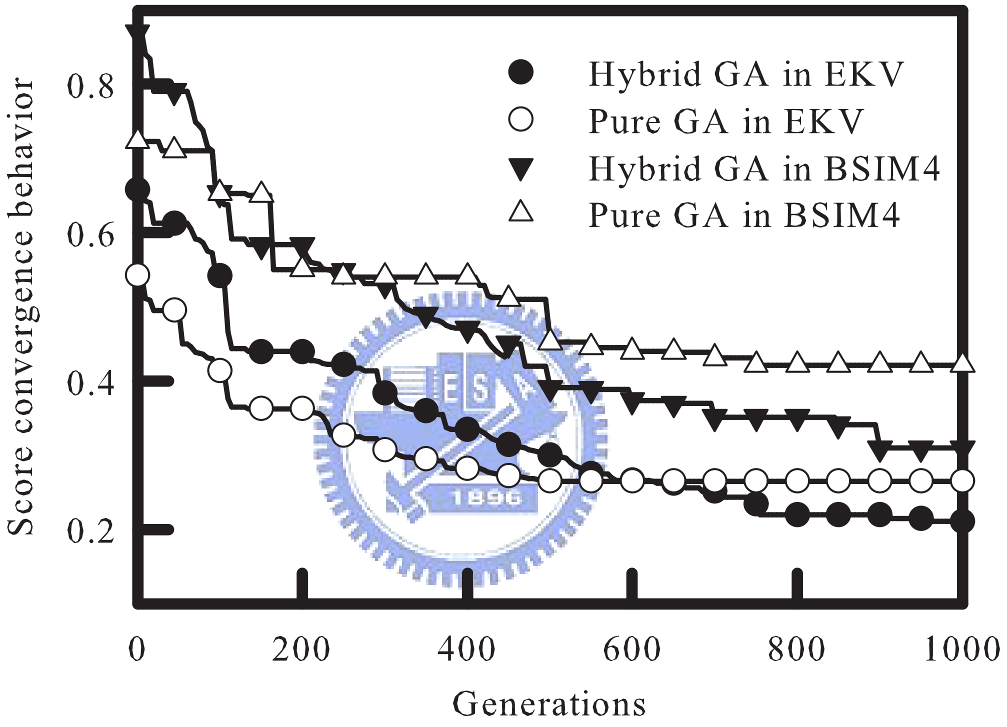


Figure 5.7: The score convergence behavior of our proposed method and pure GA for different compact models.

5.5 Effects of Random Number Generations for Parameter Extraction

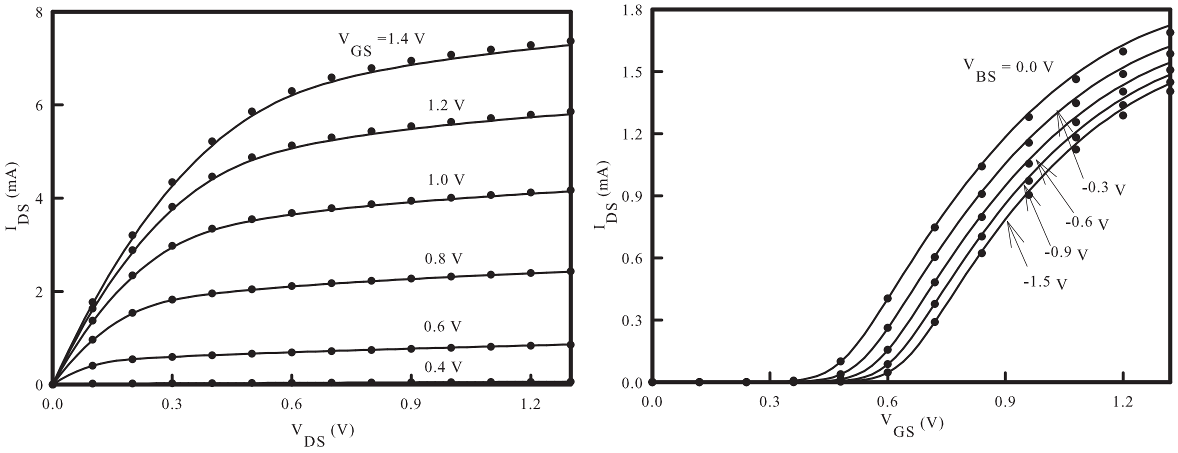
We notice that, GA is a global search optimization method based on the mechanics of natural selection and natural genetics. It works with a coded of parameters string called chromosome instead of the solutions themselves. Each chromosome represents a solution set, and the fitness functions used to measure the survival scores of all chromosomes in the population. However, the qualities of the random number generation scheme will affect the efficiency of GA. We thus examine eight different random number generation schemes in our intelligent extraction system. The selected random number generation formulations, as listed in Tab. 5.1, includes Logistic Map, Tent Map, Gauss Map, Sinusoidal Iterator, Lozi Map, Chua's Oscillator, C/C++ Random Generator, and Chaotic Random Number Generator [78–85]. The variables X_1, X_2, X_3, ... in Tab. 5.1 represent the random number sequence generated by the aforementioned methods. Among all schemes, the C/C++ random generator is a kind of linear congruent generator and it uses the $Rand$ function to generate random number. The $Rand$ function returns a value of the integer type (usually a two-byte quantity), the maximum RANDMAX value is often not so large. The other random number generation methods can generate chaotic sequences. The Logistic Map [81–83] is the simplest method for generating chaotic sequences. The Tent Map [82, 84] has similar form to the Logistic Map, which makes a little modification. The Sinusoidal Iterator [82, 84] uses sinusoidal function to generate random number sequence. Lozi Map [82, 84] is a two-dimension map similar to the henon map but use $-a \times |X_k|$ to replace $-a \times X_k^2$. The chua's oscillator [82, 83] is a well known chaotic oscillator. Apply suitable sampling times on Chua's oscillator, we can derive discrete-time chaotic time series form the signals. The Chaotic Random Number Generator [85] is one of the popular methods, which can generate random numbers with better randomness and longer cycle lengths in random maps. The randomness and cycle length of random numbers are the criterions of the random number generators. For better randomness and cycle length, the generated random number distribution can aid the solution searches of GA.

To validate the function and accuracy of the proposed intelligent method, parameters of BSIM4 (BSIM version 4) model for 90 nm n-typed metal-oxide-semiconductor field effect transistors (N-MOSFETs) are extracted in our system and compared with the measured data, as shown in Fig. 5.8. The width and channel length of the target device is equal to 10 μm and 90 nm. The errors between measured and extracted I-V curves are less than 3%. For the model parameter extraction of multiple devices, we examine the following four N-MOSFET devices: (length / width) = (1.2 μm/ 10 μm), (10 μm / 10 μm), (10 μm / 1.2 μm) and (0.13 μm / 1.2 μm). Table 5.2 shows the extraction result of the four devices. As shown in this table, RMS error of curves is strictly within 4% for all cases.

Table 5.1: Eight different random number generation schemes and their expressions.

Generation Scheme	Expression
Logistic Map	$X_{k+1} = aX_k(1 - X_k)$, where $X_0 = 0.2027$ and $a = 1$
Tent Map	$X_0 = 0.27$, if $X < 0.7$, $X_{k+1} = X_k/0.7$ otherwise, $X_{k+1} = (1/0.3)X_k(2 - X_k)$
Gauss Map	$X_{k+1} = (1/X_k)$ mod 1, where $0 < X < 1$
Sinusoidal Iterator	$X_{k+1} = sin(\pi X_k)$, $X_0 = 0.7$
Lozi Map	$X_{k+1} = b * (1 + b * X_k - a * \lvert X_k \rvert)$, where $a = 1.7$ and $b = 0.5$
Chua's Oscillator	$X(t+1) = a(Y(t) - L(X(t)))$, $Y(t+1) = X(t) - Y(t) + Z(t)$, $Z(t+1) = -bY(t) - cZ(t)$, $L(v) = m_1 v + 0.5(m_0 - m_1)(\lvert v+1 \rvert - \lvert v-1 \rvert)$, where $a = 9$, $b = 14.286$, $c = 0$, $m_0 = -0.142857$, and $m_1 = 0.285714$
C/C++ Random Generator	$X_{k+1} = (aX_k + c)$ mod M, where M=RAND_MAX, and a, c are positive
Chaotic Random Number Generator	$Z_n = ((Y_{n-j}$ rotr $r_3) + (Y_{n-k}$ rotr $r_1))$ mod $2^{b/2}$ $Y_n = ((Z_{n-j}$ rotr $r_4) + (Z_{n-k}$ rotr $r_2))$ mod $2^{b/2}$ $X_n = Y_n + Z_n * 2^{b/2}$, where $k > j > i > 0$, $b > 4 \geq 0$. Y rotr r means the bits of Y rotated r places to the right.

Figure 5.9 demonstrates the behavior of the convergence score for a single MOSFET device using BSIM4 model parameter extraction. As shown in this figure, the most well-known C/C++ random generator or Gauss map do not have outstanding performance. Similarly, Fig. 5.10 shows the convergence score in the four devices' extraction; it is obvious that C/C++ random generator even has the worst performance. It is found that the chaotic random number generator has superior convergence behavior for both single and multiple



(a) (b)

Figure 5.8: The BSIM4-model extracted (solid-line) and measured (dot-lines) (a) I_{DS}-V_{DS} and (b) I_{DS}-V_{GS} curves of the MOSFET, where length = 90nm and width = 10μm. We notice that the chaotic random number generator is used in this simulation.

Table 5.2: List of the root-mean-square (RMS) errors of the extracted results compared with the measured data for the four N-MOSFET devices. The device dimension is in μm. The oxide thickness of target devices are 3.36nm and the working temperature is settled at 298.15K. The chaotic random number generator is used in this simulation.

Device Geometry ($\mu m/\mu m$)	Error of I_{DS}-V_{DS}	Error of I_{DS}-V_{GS}
L/W (1.2/10)	3.59%	3.31%
L/W (10/10)	2.92%	3.15%
L/W (1.2/10)	3.21%	3.51%
L/W (1.2/0.13)	3.24%	3.34%

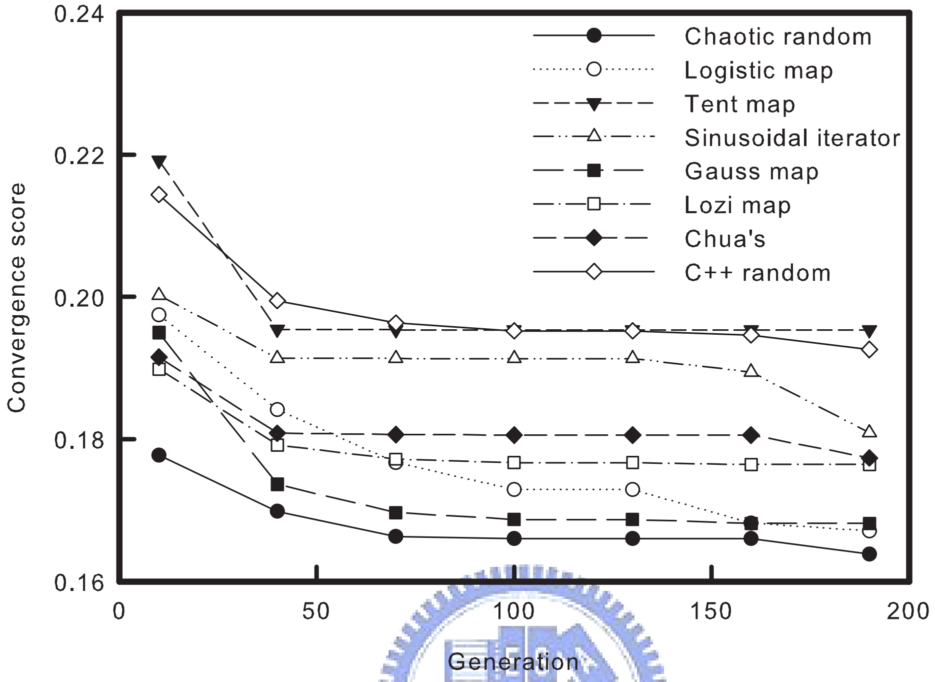


Figure 5.9: Comparison of the convergence score in GA with different random number generation schemes. In this extraction experiment, GA extracts a single N-MOSFET device.

1896

devices extraction cases. It generates the random numbers with better distribution which keeps the diversity of the extraction system, thus the best performance of the convergence score is reached. According to the numerically achieved results, it shows that the random number generation will affect the performance of GA and better generation method can improve the search for solutions in the problem of semiconductor device model parameter extraction.

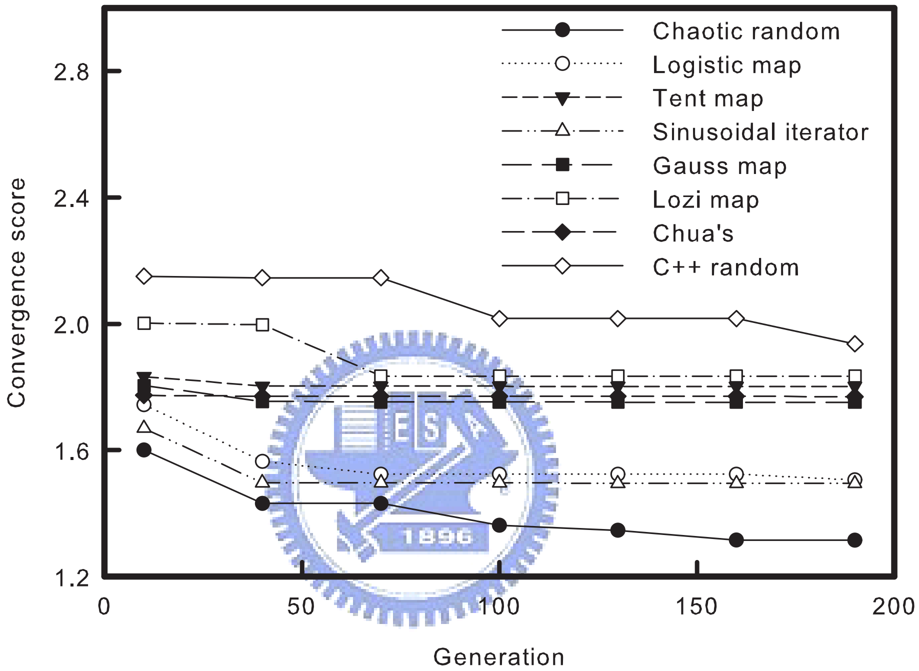


Figure 5.10: Comparison of the convergence score in GA with different random number generation schemes. In this extraction experiment, GA extracts four N-MOSFET devices at the same time.

Chapter 6

Application of UOF in VLSI Circuit Design

In this chapter, we introduce two approaches for design optimization of integrated circuits (ICs). The first one is a simulation-based approach which is based on evolutionary algorithms, numerical methods and circuit simulation. One of evolutionary algorithms, such as genetic algorithm (GA), will enable us to search solution globally, the numerical methods, such as Levenberg-Marquardt (LM) method, will enhance the results of GA by performing the local optimization and the circuit simulation is used to evaluate the fitness of each individual in the GA. Another one uses a computational statistics technique for the circuit design. Integration of a well-known circuit simulation software and central composite design method enables us to construct a second-order response surface model (RSM) for each concerned constraint. After construction of RSMs, we verify the adequacy and accuracy using the normal residual plots and their residual of squares. The constructed models are further employed for design optimization of ICs. One current mirror amplifier ICs with 0.18 μm CMOS devices are examined in this chapter. By considering the voltage gain, cutoff frequency, phase margin, common-mode rejection ratio and slew-rate, six designing parameters including the width and length of different transistors are selected and optimized to fit the targets. We also compare the optimized results of the computational statistics approach and the results of the simulation-based approach.

1896

6.1 The Integrated-Circuits Design Problem

It is known that integrated circuits (ICs) design nowadays plays a crucial role for microelectronics industry; in particular, for highly competitive consumer products [1, 2, 29, 31, 57, 86–90]. In modern ICs design flow and chip implementation, IC designers perform a series of functional examination and analysis of the characteristics by circuit simulation tools to match specifications. To meet specified electrical characteristics and performance of designed product, designers in general have to tune parameters of the passive and active devices ranging from resistors, capacitors, inductors, line width, line length, to transistor size, etc [1, 2, 31, 38, 57, 60, 89, 90]. It thus requires experienced designers to accomplish such complicated works. Diverse approaches have been proposed to reduce this designing cycle, which includes numerical optimization techniques and evolutionary algorithms; and have demonstrated their merit and validity [1, 2, 31, 46, 47, 57, 60, 89–96, 98]. Furthermore, integration of circuit simulation tool, design of experiment, and response surface methodology may also provide a cost-effective way to advanced IC design optimization and sensitivity analysis of performance.

In this investigation, two different approaches are implemented and compared for the circuit design optimization. One is a simulation-based approach which is based on the integration of evolutionary algorithms, numerical methods and circuit simulation. First of all, preliminary parameters as well as the netlist for circuit simulation are loaded. A circuit simulation tool will be performed for the circuit simulation and then the results are used in the evaluation of specification. Once the specification meets the aforementioned constraints, the optimized parameters will be outputted. Otherwise, we activate an evolutionary method, such as genetic algorithm, for the global optimization; in the meanwhile, a gradient-based method, such as the Levenberg-Marquardt method, searches the local optima according to the global searched results. The numerical optimization method does significantly accelerate the evolution process. We repeatedly call circuit simulator to compute and evaluate newer results until the specification is matched. Another approach uses a computational statistics technique [35] for the ICs design optimization. Based on HSPICE circuit simulator [1, 57, 90, 107], a central composite design (CCD), and a second-order response surface model (RSM), the circuit performances can be systematically optimized with respect to different specified constraints. The investigated current mirror amplifier IC with 0.18 μm CMOS devices has the specifications that includes the voltage gain within 50$\sim$100db, the cut-off frequency (FT) within 20$\sim$70MHz, the common-mode rejection ratio (CMRR) within 60$\sim$85db, the slew-rate (SR)+ within 20$\sim$80V/uS, and the range of SR- is 20$\sim$70V/uS. We firstly use the circuit simulator and the central composite design to construct the second-order response surface models. Seventy-seven experimental runs of circuit simulations are completed to generate the necessary data for construction of the quadratic response models. For validating the constructed model, the model adequacy checking and the accuracy verification are necessary [91, 97, 99–102]. With the second-order RSMs [60, 91, 97, 99, 101], some optimization approaches, such as the least squares method and desirability function approach, can be used to extract the optimal parameters to fit the specifications. In the examined current mirror amplifier IC, six parameters including the width and length of different transistors are selected and optimized to satisfy the circuit specifications.

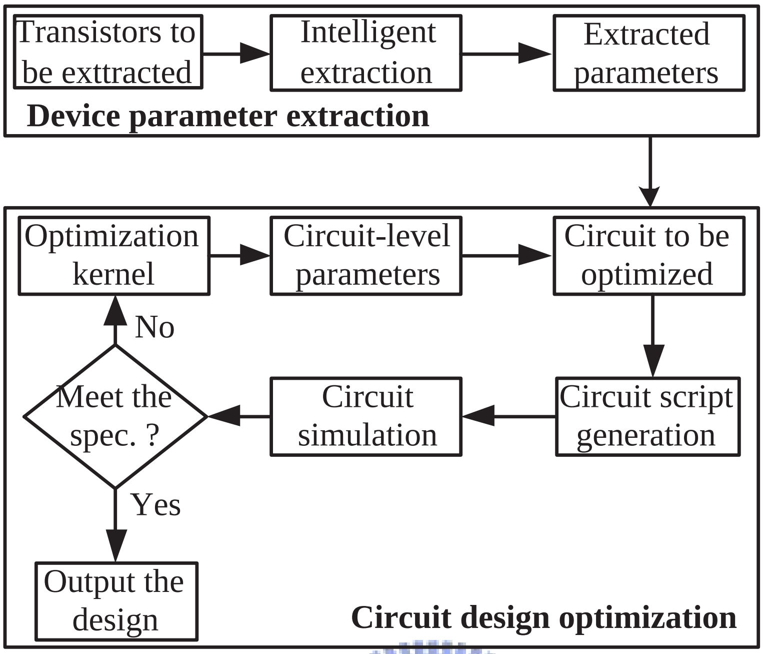


Figure 6.1: Functional blocks for the proposed simulation-based hybrid optimization approach.

1896

6.2 The Simulation-Based Hybrid Optimization Technique

Modern IC design is a real-world challenge to solve multidimensional optimization problems by using either conventional numerical method or soft computing techniques. The idea of the simulation-based hybrid optimization technique takes a evolutionary algorithm to perform global search, and while the evolution seems to be saturated, the gradient-based method is then enhancing the searching behavior to perform the local search.

Two stages, shown in Fig. 6.1, are performed in the optimal design of ICs. The model

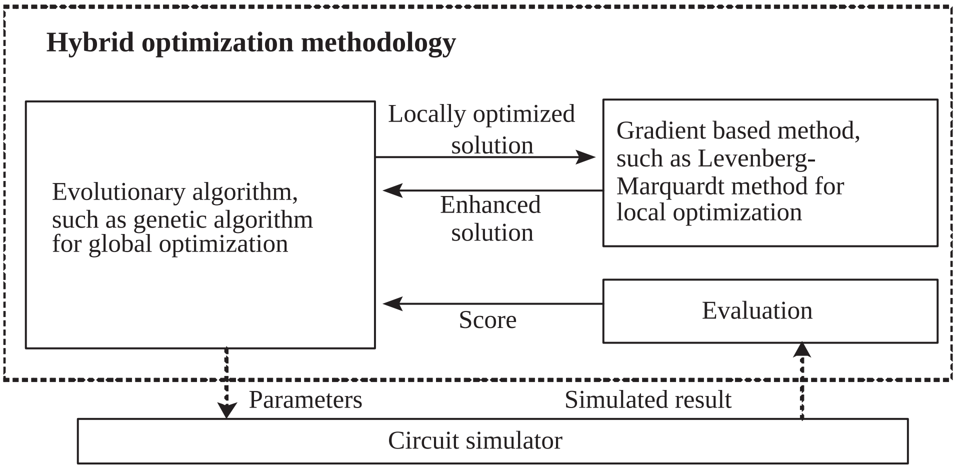

Hybrid optimization methodology
Evolutionary algorithm, such as genetic algorithm for global optimization
Locally optimized solution
Enhanced solution
Gradient based method, such as Levenberg-Marquardt method for local optimization
Score
Evaluation
Parameters
Simulated result
Circuit simulator

Figure 6.2: The optimization flow for the hybrid optimization approach.

parameters of CMOS transistors are first optimized. For a set of given measured current-voltage (I-V) data of CMOS devices and a selected equivalent circuit model. With the optimized device parameters, we then move to the part of circuit design optimization. Based on the developed architecture, the UOF can easily integrate well-known circuit simulation tools, such as HSPICE, ELDO, SPECTRE, SmartSPICE in the optimization flow. For a circuit to be optimized, we automatically parse and generate the corresponding netlist of the circuit for the circuit simulation and result evaluation. If the result meets the target, we output the final optimized data. If the error between the target and result does not meet the stopping criterion, the established optimization kernel will enable the circuit parameter extraction in a global sense, where the number of parameters to be extracted depends upon the specification that we want to be achieved. According to the optimized results, the netlist is updated and the next optimization is repeated. As shown in Fig. 6.2, during the optimization process, the global-searched candidates will be inputted into a circuit simulator to retrieve the simulation results, where a set of differential equations is solved numerically. According to the specified targets, the results are evaluated to measure the fitness score, which guides the evolutionary process. Once a solution is obtained, the gradient-based numerical method solves the problem locally. The local optima are right the initial values of evolutionary algorithm for further optimization. After this optimization process, the optimal parameters can be extracted with respect to the specifications.

For the circuit design optimization, the empirical knowledge and experience of the explored circuits can assist the circuit design task. Well understanding of the purposes, characteristics and behaviors of the circuits can help us to generate some empirical rules in the optimization process. For example, the explored current mirror amplifier circuit, as shown in Fig. 6.3, can be divided into five different components, including transistor pairs (MB,MS), (M1,M2), (M3,M5), (M4,M6) and (M7,M8). The transistors MB and MS provide the current source for the differential amplifier. The transistors M1 and M2 form the differential amplifier, and transistors M3 $\sim$ M6 make up the current mirror. The current mirror amplifier circuit is a symmetrical circuit, the sizes of M1 and M2 are the same, the transistors M3, M4, M5 and M6 have the same width and length, and the size of transistors M7 and M8 are also the same. The transistor MB is a component for providing voltage source, and its size is usually fixed. The MS is a current source and the magnitude of current is related to the channel width of MS and MB. To avoid the process variation, the channel length of the input pair M1 and M2 are also fixed. Therefore, we can select the partial parameters for optimization, which include width of MS, width of M1 and M2, length and width of M3, M4, M5 and M6, and the size of transistors M7 and M8. It is better for for us to only adjust these selected parameters without extracted all parameters (the length and width of all transistors) for some specifications. If we extract all parameters simultaneously, it may damage some properties of the current mirror amplifier circuit and also makes the optimization process more complicated. By integrating this kind of empirical knowledge into the proposed optimization approach can significantly improve the optimization process.

6.3 The Computational Statistics Technique

In this section, one computational statistics approach for circuit design is presented, and the procedure of the proposed methodology is given by:

```
The Computational Statistics Approach
{
    Step 1. Variables selection by screening design or empirical check
    Step 2. Central composite design
```

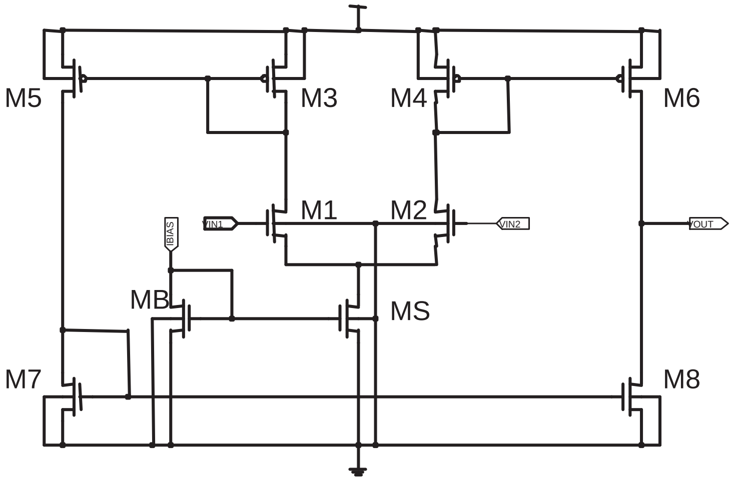


Figure 6.3: The explored current mirror amplifier circuit. The transistors MB and MS provide the current source for the differential amplifier. The transistors M1 and M2 form the differential amplifier, and transistors M3$\sim$M6 make up the current mirror.

Step 3. *Model construction*

Step 4. *Accuracy and adequacy verifications*

Step 5. *If not satisfies the verifications in Step* 4

perform transformation of the parameters, then goto Step 1

Step 6. *Optimization of characteristics*

Step 7. *Check the results with target*

If results not achieve the target, repeat Steps 1$\sim$6

Step 8. *Output the optimized results*

}

The first step in our approach is variables selection by using a screening design or empirical check so that we can determine the significant designing parameters. After determining the important factors, we will execute the central composite design by running the HSPICE circuit simulator. A second-order response surface model is then established between circuit performance (i.e., responses) and circuit parameters (i.e., factors), and the optimization for the circuit design then can be followed. In the following subsections, we state each procedure respectively.

6.3.1 Variables selection

Variables selection is a step to find the few significant factors from a list of many potential ones. Alternatively, we use a screening design or empirical check to identify significant main effects, rather than interaction effects, the latter being assumed an order of magnitude less important. To determine factor's significance, two-level fractional factorial design or Plackett-Burman design is ideally suited for screening design [96]. Two-level fractional factorial design can reasonably assume that high-order interactions are negligible. We can run only a fraction of the complete factorial experiment to obtain information on the main effects and low-order interactions. For example, in one-half fraction of the 2^3 design (2^{3-1}

design), A and BC are aliases, B and AC are aliases, C and AB are aliases, where A, B, and C are factors. When designs with resolution III, main effects are aliases with two-factor interactions and two-factor interactions may be aliased with each other. Sometimes designs with resolution IV are also used for screening designs. In this design main effects are aliased with, at worst, three-factor interactions. This is better from the confounding viewpoint, but the designs require more runs than a resolution III design.

Plackett-Burman design is two-level fractional designs [98] for studying up to $k = N - 1$ variables in N runs, where N is a multiple of 4. In a Plackett-Burman design, main effects are heavily confounded with two-factor interactions in general. For example, N = 12, every main effect is partially aliased with every two-factor interaction. Each main effect is partially aliased with 45 two-factor interactions. And the plus and minus signs are:

$$K = 11, N = 12 + + - + + + - - - + - \,. \tag{6.1}$$

When we analyze data from screening designs, the use of an error mean square obtained by pooling high order interactions is inappropriate occasionally. To overcome this problem a half-normal probability plot of the estimates of the effects is suggested. The half-normal plot consists of the point:

$$(\Phi^{-1}(0.5 + 0.5[i - 0.5]/I), |\theta|_{(i)}|), \tag{6.2}$$

for i = 1, . . ., I. The Φ is the cumulated density function of the standard normal distribution. If factors are unimportant, the effects with mean zero and variance σ^2 will tend to fall along a straight line on this plot, whereas important factors will not lie along the straight line [91,97,99].

In our current mirror amplifier IC, six variables are selected from eight parameters according to the empirical check. The selected parameters include W1, W3, L3, W7, L7, and Ws (i.e., the width or length of different transistors).

6.3.2 Central Composite Design

Response models are constructed relating the characteristics of circuits using data generated from statistical experimentation. Mathematically, response surface models may be represented as second-order polynomials:

$$Y = \beta_0 + \Sigma_{i=1}^{k}\beta_i x_i + \Sigma_{i=1}^{k}\beta_{ii} x_i^2 + \Sigma_{i=1}^{k}\Sigma_{i\neq j}^{k}\beta_{ij} x_i x_j + \varepsilon, \tag{6.3}$$

where k is the number of input factors, x_i is the ith input factor, β_i is the ith regression coefficient, and ε represents model error.

Many applications of response surface models involve constructing and checking the adequacy of a second-order model. The central-composite design (CCD) is perhaps the most common experimental design used to generate second-order response models. These designs combine a two-level full factorial or fractional factorial design of n_f runs with $2k$ axial runs and n_c center runs, where k represents the number of control factors [91,97,99]. The axial points represent new extreme values for each factor in the design. There are three varieties of CCD which include central composite circumscribed (CCC), central composite inscribed (CCI), and face centered cube (CCF) design [96].

Central-composite designs include five input levels for each control factor (0 ; ± 1 ; ± α). Level 0, the nominal factor level, represents the base processing conditions. The cube levels (± 1) are selected to reflect the design space of interest. These values are typically set to a multiple of the factor's standard deviation or a percentage of its nominal value. The precise value of α depends on certain properties desired for the design and on the number of factors involved. To maintain rotatability, the value of α depends on the number of experimental runs in the factorial portion of the central composite design:

$$\alpha = [n_c]^{1/4}. \tag{6.4}$$

6.3.3 Response Surface Model Construction

It is necessary to develop an approximate model for the true response surface. If n observations are collected in an experiment, the model for them takes the form:

$$y = X\beta + \varepsilon, \tag{6.5}$$

where

$$y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \quad X = \begin{pmatrix} 1 & x_{11} & x_{12} \ldots & x_{1k} \\ 1 & x_{21} & x_{22} \cdots & x_{2k} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & x_{n2} \cdots & x_{nk} \end{pmatrix}, \quad \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{pmatrix}, and \quad \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix}.$$

In general, y is an $n \times 1$ vector of the observations, X is an $n \times k$ matrix of the levels of the independent variables, β is a $k \times 1$ vector of the regression coefficients, and ε is an $n \times 1$ vector of random errors.

We want to find the least squares estimators, $\hat{\beta}$, that minimizes

$$L = \sum_{i=1}^{n} \varepsilon_i^2 = \varepsilon^T \varepsilon = (y - X\beta)^T (y - X\beta). \tag{6.6}$$

As the result of our calculation, the least squares estimator of β is

$$\hat{\beta} = (X^T X)^{-1} X^T y. \tag{6.7}$$

The fitted regression model is

$$\hat{y} = X\hat{\beta}. \tag{6.8}$$

The difference between the responses y_i and the fitted value $\hat{y}_i$ is a residual, say $e_i = y_i - \hat{y}_i$, The vector of residual is denoted by:

$$\mathbf{e} = y - \hat{y}. \tag{6.9}$$

To check the normality assumption, we prepare a normal probability plot of the residual values. If the assumption holds, this plot will resemble a straight line. If the assumption is violated, a non-linear data transformation (e.g., $y' = log(y)$) may be applied and new models are generated in an attempt to improve model adequacy [99]. A second plot showing the residual values versus the predicted response values is used to verify if the variance of the original observation is constant. A random scattering of the residual values indicates that no correlation exists between the observed variance and the mean level of the response [60,91,97].

To develop an estimator of this parameter, we consider the sum of squares of the residuals, say

$$SS_E = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} e_i^2 = e^T e. \tag{6.10}$$

Equation (6.10) is called the **error** or **residual of squares**, and it has $n - k$ degrees of freedom associated with it. It can be shown that

$$E(SS_E) = \sigma^2(n-k), \tag{6.11}$$

so an unbiased estimator of σ^2 is given by

$$\hat{\sigma}^2 = \frac{SS_E}{n-k}. \tag{6.12}$$

To determine if there is a linear relationship between the response variable y and a subset of the regressor variables $x_1, x_2, \cdots, x_k$ we test for significance of regression. The appropriate hypotheses are [99]:

$$\begin{aligned} H_0 &: \beta_1 = \beta_2 = \cdots = \beta_k = 0, \\ H_1 &: \beta_j \neq 0 \; for\ at\ least\ one\ j. \end{aligned} \tag{6.13}$$

If we reject H_0, it implies that at least one of the regressor variables $x_1, x_2, \cdots, x_k$ contributes significantly to the model. The test procedure involves partitioning the total sum of squares due to residual:

$$SS_T = SS_R + SS_E. \tag{6.14}$$

where SS_R is the sum of squares due to regression. A relatively simple procedure is performed to check for model significance in relation to random error. This test involves calculating the test statistic:

$$F_0 = \frac{MS_R}{MS_E} = \frac{SS_R/k}{SS_E/(n-k-1)} = \frac{\frac{1}{k}\sum_{j=1}^{n}(\hat{y}_i - \overline{y})^2}{\frac{1}{n-k-1}\sum_{j=1}^{n}(y_i - \hat{y}_i)^2}, \tag{6.15}$$

where $\overline{y}$ is the average of measured response values. y_i, $\hat{y}_i$, and n are the ith measured response, the ith predicted response, and the number of simulated runs, respectively [99]. If this statistic exceeds the corresponding value of the F distribution value ($F_{\alpha,k,n-k-1}$), the response model is considered significant in relation to random error.

A second statistic, the coefficient of multiple determination R^2 is defined as:

$$R^2 = \frac{SS_R}{SS_T} = 1 - \frac{SS_E}{SS_T} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2}. \tag{6.16}$$

R^2 measures the amount of reduction in variability of the response y achieved, using the input factors $x_1, x_2, \ldots, x_k$. From Eq. (6.14) we see that R^2 varies from zero to one [60,91,97,99]. However, a large value of R^2 does not necessarily imply that the regression model is good one. Adding a variable to the model will always increase R^2, regardless of whether the additional variable is statistically significant or not. About this problem, some regression model builders prefer to use an adjusted R^2 statistic defined as

$$R^2_{adj} = 1 - \frac{SS_E/(n-k-1)}{SS_T/(n-1)} = 1 - \frac{n-1}{n-k-1}(1-R^2). \tag{6.17}$$

In general, the adjusted R^2 statistic will not always increase as variables are added to the model. In fact, if unnecessary terms are added, the value of R^2_{adj} will often decrease.

6.3.4 Optimization of characteristics

After the construction of models, we can use several techniques, such as the normality assumption and plot of residuals versus predicted value, to verify the adequacy of the response surface model. We further apply some numerical methods to perform the design optimization of our circuits. A useful approach to optimization of multiple responses is to use the simultaneous optimization technique popularized by Derringer and Suich [103]. Their procedure makes use of desirability functions. The general approach is to first convert each response y_i into an individual desirability function d_i that varies over the range $0 \le d_i \le 1$, where if the response y_i is at its goal or target, then d_i = 1, and if the response is outside an acceptable region, d_i = 0. When multiple response are transformed into individual desirability, the individual desirability is then combined using geometric mean to maximize the overall desirability D :

$$D = (d_1 \times d_2 \times \ldots \times d_m)^{1/m}, \tag{6.18}$$

where m is the number of responses [103]. By the equation, if any d_i is equal to zero, then the overall desirability is zero.

According to the specification for the responses, response is to be maximized, minimized, or achieved a target value. For the ith response y_i is a maximum value,

$$d_i = \begin{cases} 0, & \hat{y}_i < L_i \\ (\frac{\hat{y}_i - L_i}{T_i - L_i})^s, & L_i \leq \hat{y}_i \leq T_i \cdot \\ 1, & \hat{y}_i > T_i \end{cases} \tag{6.19}$$

For the response y_i is a minimum value,

$$d_i = \begin{cases} 1, & \hat{y}_i < T_i \\ (\frac{U_i - \hat{y}_i}{U_i - T_i})^s, & T_i \leq \hat{y}_i \leq U_i \cdot \\ 0, & \hat{y}_i > U_i \end{cases} \tag{6.20}$$

For the response is achieved a target value,

$$d_i = \begin{cases} 0, & \hat{y}_i < L_i \\ (\frac{\hat{y}_i - L_i}{T_i - L_i})^s, & L_i \leq \hat{y}_i \leq T_i \\ (\frac{U_i - \hat{y}_i}{U_i - T_i})^t, & T_i \leq \hat{y}_i \leq U_i \\ 0, & \hat{y}_i > U_i \end{cases}, \tag{6.21}$$

where the weight s and t determine how important it is close to the target value. When the weight $s = 1, t = 1$ the desirability function is liner. Choosing $s > 1$, $t > 1$ means more important to be close the target value with the function that is concave, and choosing $0 < s < 1$, $0 < t < 1$ means this less important with the function that is convex. L_i, U_i, and T_i are the lower, upper, and target value, respectively.

6.4 The Achieved Results and Discussion

Operational amplifier plays an indispensable role of analog integrated circuits. We explore the design optimization of a current mirror amplifier [86–88, 106] with 0.18 μm CMOS devices to examine the validity of the proposed method. Figure 6.3 shows the studied current mirror amplifier circuit. In this experiment, the specifications are including the ranges of voltage gain, cut-off frequency, phase margin, common-mode rejection ratio and slew-rate; and the adjustable parameters are related to the size of transistors used in the designed IC. The gain is a measure of the circuit's ability, and it is defined as the mean ratio of the signal output to the signal input of a system. The term cut-off frequency represents a boundary in the system response. CMRR measures the tendency of the device to reject input signals. The slew rate represents the maximum rate of change of signal at any point in a circuit. The phase margin is the additional phase required to bring the phase of the loop gain to -180 degrees. These are important factors to measure the ability and performance of the current mirror amplifier IC. According to the results of the variable selection, we selected six factors which includes width of transistor M1 (W1), width and length of M3 (W3 and L3), width and length of M7 (W7 and L7) and width of MS (Ws) for our next design, and the face centered cube design (CCF) is also used. To generate the necessary data for construction of the quadratic response models, 77 experimental runs (1 center point, 12 axial points, and 26 cube points) are completed by the running HSPICE circuit simulations. We choose the CCF design from the three type of the CCD (CCC, CCI, and CCF), because the CCF design is more suitable for designing the response surface models. Following equations are the constructed model of each response, and the variables, A, B, C, D, E and F are the factors of W1, W3, L3, W7, L7 and Ws, respectively.

$$
\begin{aligned}
GAIN = {} & 61.3078 - 0.33473 \times E^2 + 1.415906 \times A + 0.285844 \times A \times F \\
& +0.013121 \times B + 0.069125 \times B \times F + 0.839219 \times C - 0.04759 \times C \times D \\
& -0.16788 \times D + 0.240156 \times C \times E + 1.906719 \times E + 0.176656 \times C \times F \\
& -0.87466 \times F + 0.054844 \times D \times E - 1.09336 \times A^2 + 0.098156 \times D \times F \\
& -0.14331 \times C^2 - 0.23459 \times E \times F. \qquad (6.22)
\end{aligned}
$$

$$
\begin{aligned}
1.0/Sqrt(FT) = {} & 0.0001651 + 8.46 \times 10^{-8} \times E^2 - 1.25 \times 10^{-5} \times A \\
& +3.07 \times 10^{-8} \times A \times B + 5.3 \times 10^{-7} \times B + 6.96 \times 10^{-7} \times A \times C \\
& +4.632 \times 10^{-6} \times C + 3.86 \times 10^{-8} \times A \times E + 2.344 \times 10^{-8} \times D \\
& -5.7 \times 10^{-7} \times A \times F + 3.123 \times 10^{-7} \times E + 3.28 \times 10^{-7} \times B \times C \\
& -2.625 \times 10^{-5} \times F - 6.8 \times 10^{-8} \times B \times F + 1.421 \times 10^{-5} \times A^2 \\
& -5 \times 10^{-8} \times C \times E + 6.348 \times 10^{-7} \times C^2 - 2.2 \times 10^{-7} \times C \times F. \qquad (6.23)
\end{aligned}
$$

$$
\begin{aligned}
(PM)^{1.57} = 721.3923 - 0.80638 \times E^2 - 55.5958 \times A - 6.95119 \times A \times C \\
-15.3572 \times B - 1.8323 \times A \times E - 132.616 \times C - 11.4127 \times A \times F \\
-1.42565 \times D - 3.27457 \times B \times C - 18.9915 \times E2.38953 \times C \times E \\
-27.2108 \times F - 4.90147 \times C \times F + 25.36517 \times A^2 - 0.91534 \times D \times E \\
+0.880197 \times B^2 - 0.82643 \times E \times F + 9.737562 \times C^2. \qquad (6.24)
\end{aligned}
$$

$$
\begin{aligned}
(CMRR)^{-2.34} = 6.6662 \times 10^{-5} + 3.713 \times 10^{-7} \times A \times D - 1.092 \times 10^{-5} \times A \\
-8.542 \times 10^{-7} \times A \times E - 9.3073 \times 10^{-7} \times B - 5.454 \times 10^{-7} \times A \times F \\
+2.13336 \times 10^{-6} \times C - 3.928 \times 10^{-7} \times B \times C - 5.7487 \times 10^{-7} \times D \\
-1.065 \times 10^{-7} \times B \times E + 1.36801 \times 10^{-6} \times E - 3.767 \times 10^{-7} \times B \times F \\
+1.61471 \times 10^{-5} \times F - 1.027 \times 10^{-7} \times C \times D + 1.08167 \times 10^{-7} \times B^2 \\
+2.357 \times 10^{-7} \times C \times E - 1.1113 \times 10^{-7} \times C^2 + 7.426 \times 10^{-7} \times C \times F \\
-2.2466 \times 10^{-7} \times E^2 - 4.531 \times 10^{-7} \times D \times F - 1.4178 \times 10^{-7} \times A \times B \\
+1.025 \times 10^{-6} \times E \times F + 3.63478 \times 10^{-7} \times A \times C. \qquad (6.25)
\end{aligned}
$$

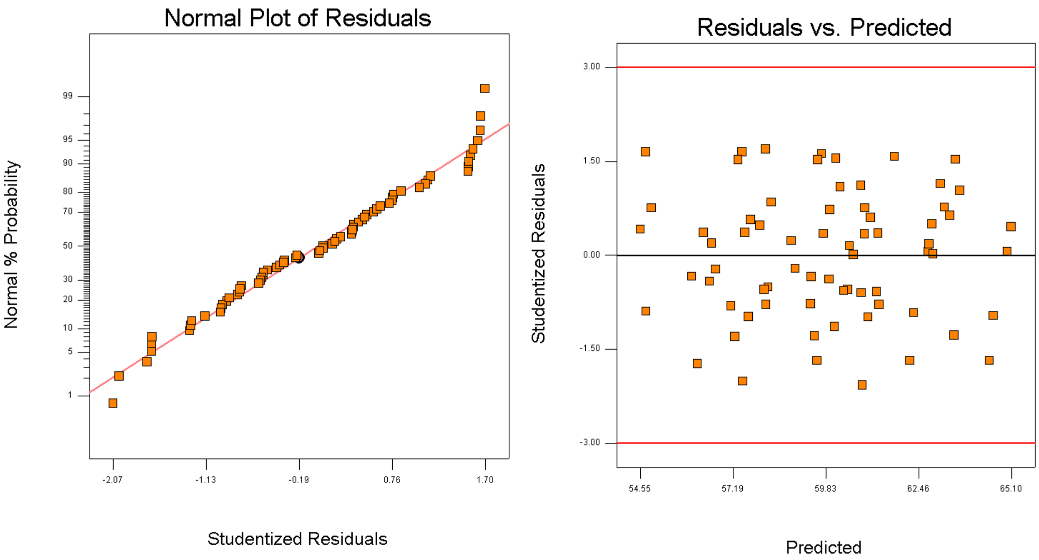
Table 6.1: List of the residual of squares in the six response surface models.

Response	Residual of squares
$GAIN$	0.9993
$1.0/Sqrt(FT)$	0.9999
$PM^{1.57}$	0.9999
$CMRR^{-2.34}$	0.9997
$Sqrt(SR+)$	0.9999
$1.0/Sqrt(SR-)$	0.9984

$$\begin{aligned} Sqrt(SR+) &= 6498.221 + 22.16759 \times C^2 - 15.1833 \times A - 6.28594 \times A \times C \\ &+4.293962 \times B - 3.11049 \times A \times F - 90.0353 \times C + 7.492849 \times B \times C \\ &+1805.042 \times F + 8.133772 \times B \times F + 59.15891 \times A^2 - 83.5258 \times C \times F. \quad (6.26) \end{aligned}$$

$$\begin{aligned} 1.0/Sqrt(SR-) &= 0.000163577 - 4.3 \times 10^{-7} \times A \times B + 9.28055 \times 10^{-7} \times A \\ &+4.58 \times 10^{-7} \times A \times C - 6.0608 \times 10^{-7} \times B + 3.17 \times 10^{-7} \times A \times F \\ &+2.31514 \times 10^{-6} \times C - 9.3 \times 10^{-7} \times B \times C - 3.5869 \times 10^{-5} \times F \\ &-1 \times 10^{-6} \times B \times F + 1.44356 \times 10^{-5} \times A^2 + 9.77 \times 10^{-7} \times C \times F. \quad (6.27) \end{aligned}$$

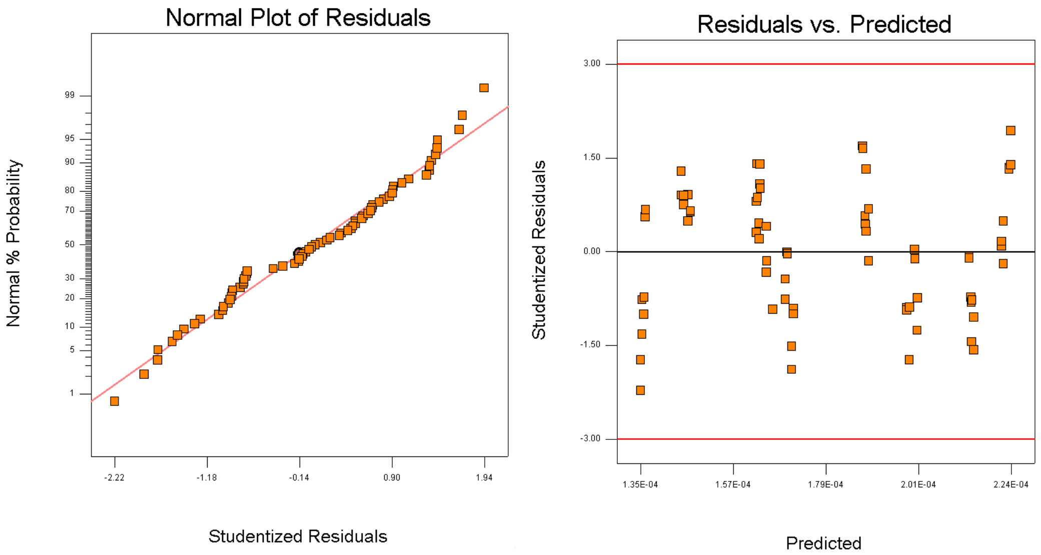
The $SR+$ and $SR-$ in Eqs. (6.26) and (6.27) are the positive and negative slew-rate. After we construct the response models, we need to verify the adequacy of the models. We



(a) (b)

Figure 6.4: (a) The residual normal probability plot and (b) the residuals vs. predicted plot for the response $GAIN$.

notice that the transformation of FT, PM, CMRR and SR- by BOX-COX transformation or experience [103] can improve the model adequacy. Table 6.1 shows the residual of squares of each response surface model. Figure 6.4 shows the (a) residual normal probability plot and (b) the residuals vs. predicted plot for the response GAIN, and Figs. 6.5-6.9 show the model adequacy checking for the $1.0/Sqrt(FT)$, $PM^{1.57}$, $CMRR^{-2.34}$, $Sqrt(SR+)$, and $1.0/Sqrt(SR-)$. Figures 6.10-6.12 show the scatter plots of values calculated from the response surface models versus values that obtained from the HSPICE circuit simulator for the models of all responses. The results show that there is a high linearity between actual

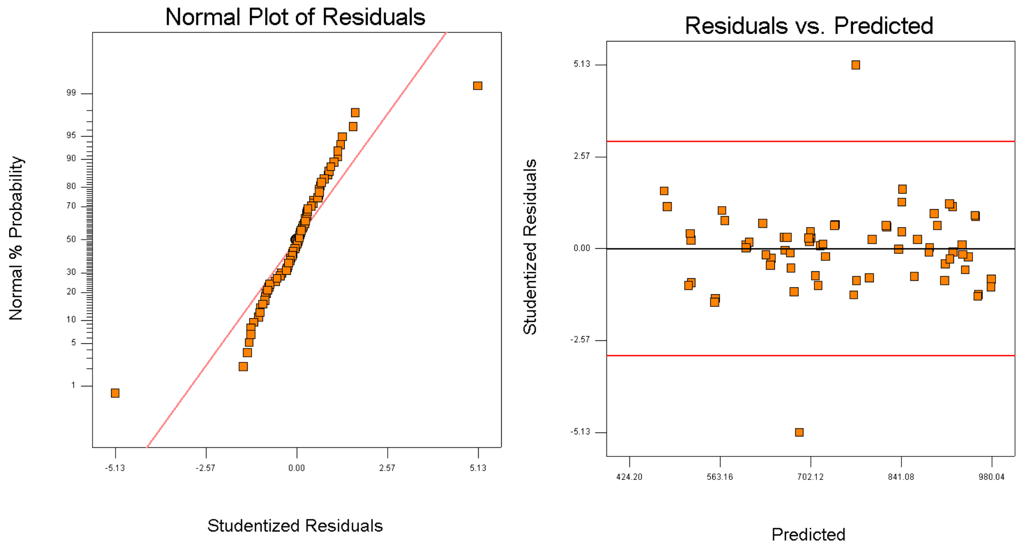


(a) (b)

Figure 6.5: (a) The residual normal probability plot and (b) the residuals vs. predicted plot for the response $1.0/Sqrt(FT)$.

and predicted values and thus confirm the accuracy of the constructed models. The Design-Expert software is employed in this examination, and the proposed statistics method is now developed in our unified optimization framework.

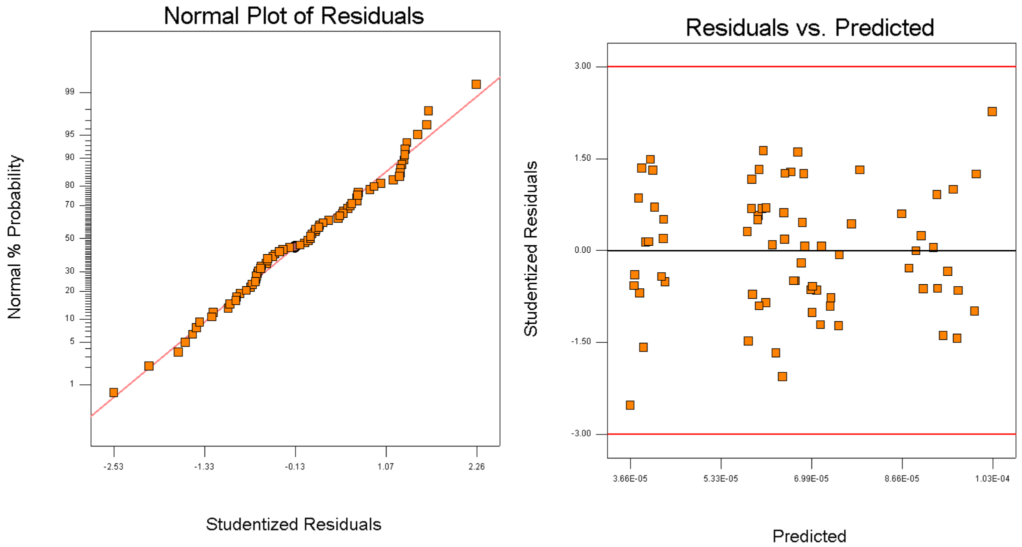
Based on the response models, we can extract the factors to satisfy the targets. In this examination, we also apply the simulation-based hybrid approach to extract the parameters. Table 6.2 lists the extracted parameters of both methods, and the range of each parameter for constructing the responses. Table 6.3 summaries the list of targets and the extracted results for the current mirror amplifier circuit. It shows that the extracted parameters can satisfy all the specified targets for both methods. As shown in Tab. 6.3, the simulation-based



(a) (b)

Figure 6.6: (a) The residual normal probability plot and (b) the residuals vs. predicted plot for the response $PM^{1.57}$.

approach can obtain better results of some specifications including FT, CMRR, SR+ and SP-, with a little lower Gain value, by comparing to the results of the statistics approach. This is due to the limitation of the parameters' range for the statistics approach. Since the accuracy of the responses can be promised only when the used parameters are within the default range for constructing the responses. In this examination of the simulation-based approach, the parameters are not limited to within the range, so it can extract parameters outside the range for the better result. However, the simulation-based approach usually spends more time for extracting the parameters, and the statistics approach can obtain the results more efficiently.

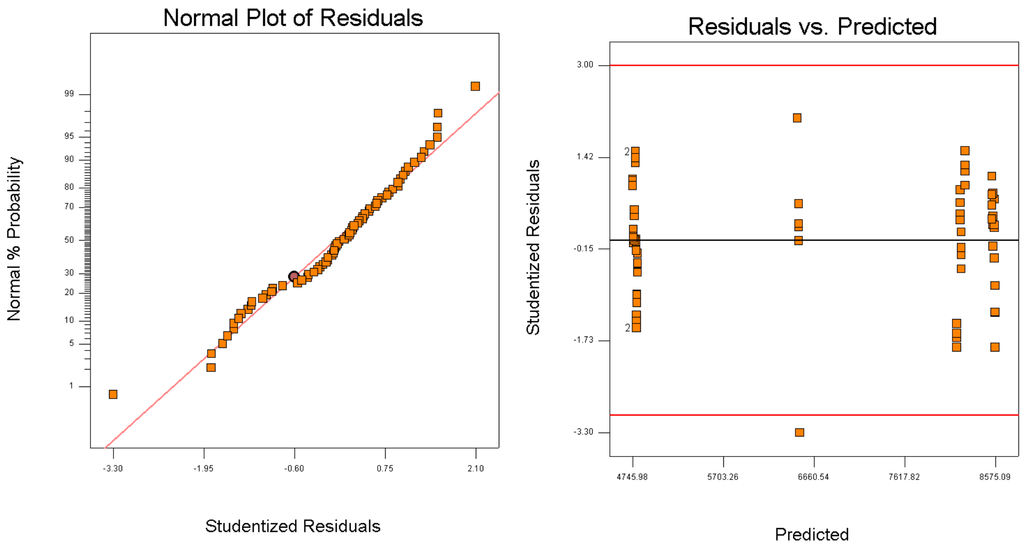


(a) (b)

Figure 6.7: (a) The residual normal probability plot and (b) the residuals vs. predicted plot for the response $CMRR^{-2.34}$.

Table 6.2: List of the range of designing parameters and the optimized parameters for the current mirror amplifier circuit.

Parameter	Range for constructing the RSM	Results of statistics approach	Results of simulation-based approach
A: W1 (μm)	40 $\sim$ 100	100	14.5483
B: W3 (μm)	60 $\sim$ 80	80	44.3709
C: L3 (μm)	1.2 $\sim$ 2.5	1.21	1.32914
D: W7 (μm)	14 $\sim$ 20	18.24	79.6026
E: L7 (μm)	1.2 $\sim$ 2.5	2.47	2.23311
F: WS (μm)	60 $\sim$ 160	66.16	14.55

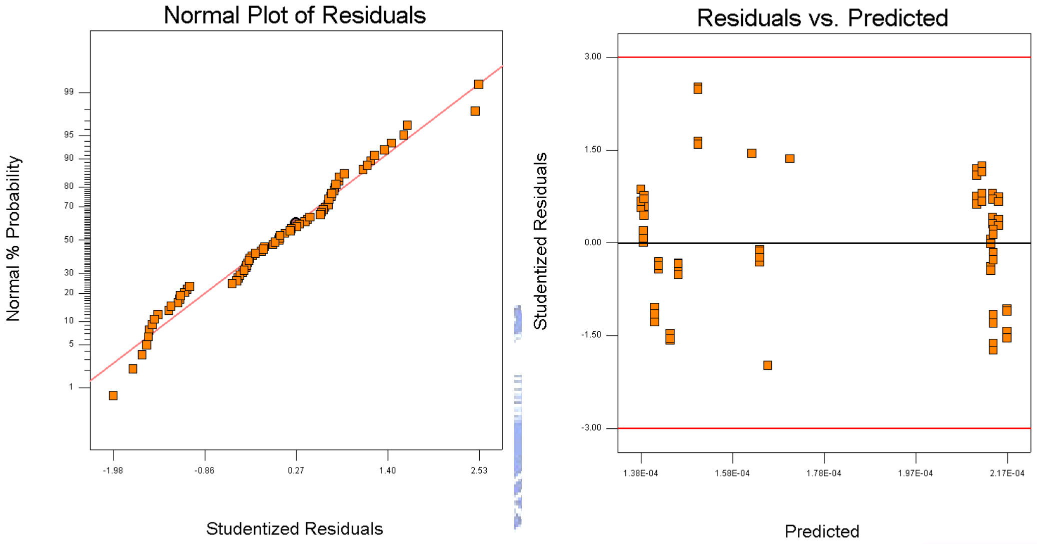


(a) (b)

Figure 6.8: (a) The residual normal probability plot and (b) the residuals vs. predicted plot for the response $Sqrt(SR+)$.

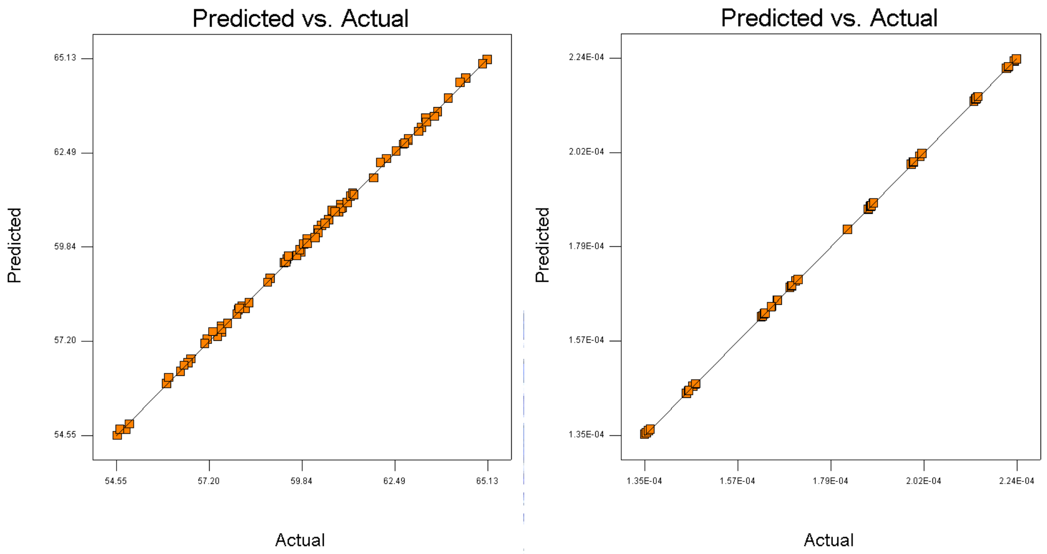
Table 6.3: List of the specified targets and the extracted results for the current mirror amplifier circuit.

Specification	Goal	Range	Results of statistics approach	Results of simulation-based approach
Gain(db)	maximize	50∼ 100	61.54	55.358
FT(Mhz)	maximize	20 ∼ 90	43.87	63.128
PM (degree)	in the range	55 ∼ 70	70	56.26
CMRR (db)	maximize	60 ∼ 85	64.01	66.132
SR+(V/us)	maximize	20 ∼ 100	56.27	99.405
SR-(V/us)	maximize	20 ∼ 100	39.40	88.155



(a) (b)

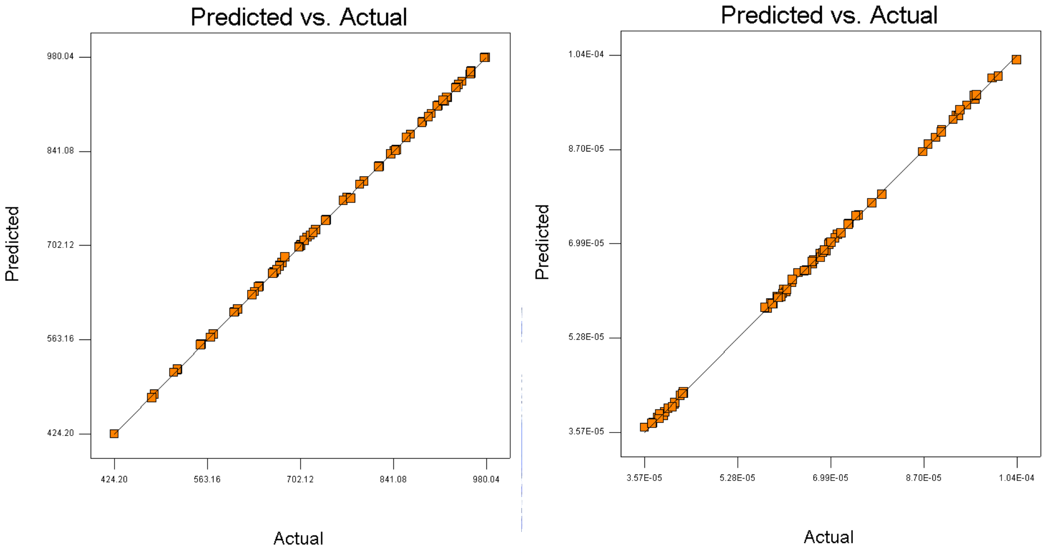
Figure 6.9: (a) The residual normal probability plot and (b) the residuals vs. predicted plot for the response $1.0/Sqrt(SR-)$.



(a)

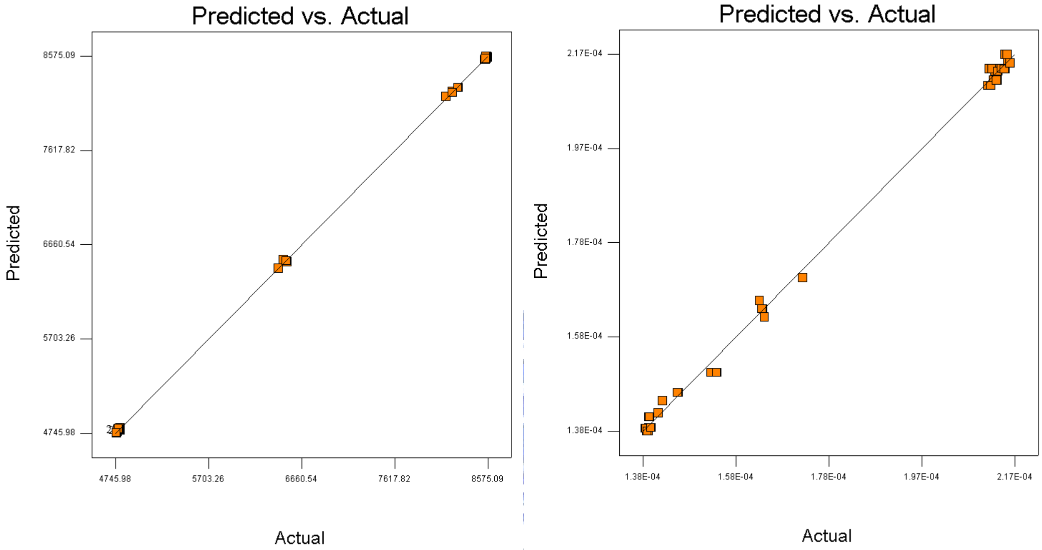
(b)

Figure 6.10: Scatter plot calculated from the response surface model versus values obtained from circuit simulator for (a) $GAIN$ and (b) $1.0/Sqrt(FT)$.



(a) (b)

Figure 6.11: Scatter plot calculated from the response surface model versus values obtained from circuit simulator for (a) $PM^{1.57}$ and (b) $CMRR^{-2.34}$.



(a) (b)

Figure 6.12: Scatter plot calculated from the response surface model versus values obtained from circuit simulator for (a) $Sqrt(SR+)$ and (b) $1.0/Sqrt(SR-)$.

Chapter 7

Application of UOF in Communication System Antenna Design

In this chapter, an investigation for the design of antenna pattern applied for mobile broadcasting and the 802.11a WLAN is presented. A Z-shaped antenna is optimized with respect to the return loss by the UOF. Inspired by the scenario of GA for optical proximity correction in our earlier work [32, 122], we firstly partition the edges of the antenna into segments, and then adjust the movements of each segment to construct the geometry of new antenna. The external simulator is then called to simulate and evaluate the performance of the new antenna. If the simulated results meet the specific constraints, the optimized antenna is outputted. Otherwise, the evolutionary algorithms will enable us to search solution again. The preliminary results confirm the robustness and efficiency of the proposed optimization method.

7.1 The Antenna Geometry Design Problem

A trend of portable device integration has risen last decade due to the massive growth of wireless communications, and products are expected to have multiple wireless service. As a result, a single, small antenna with the ability to operate effectively over each required bands is desired. Modern antenna applications have special specifications, such as small size, multi-band or broadband requirement, beam-forming, system gain and so on in communication systems [7, 11, 108–119]. In the past years, in order to achieve the desired specifications based on well-known geometry, how to adjust the geometry to get the best design is a state-of-art and needs more experience through trial-and-error process. Recently, the optimization scheme, genetic algorithm [1–3, 16, 68, 69, 121], is considered to optimize the antenna with required specifications and makes design procedure automatic [7, 11, 108–119]. Based on pre-selected design parameters, such as the geometry dimension [108, 110, 115–119], locations and values of loads [108, 118], the number of elements of fractals [109, 112, 113], the number of arrays [110, 119], the existence of discretized elements [111,114], and so on, GA adjusts those parameters automatically by communicating with full-wave electromagnetic solvers to extract the evolved antenna. The GA-based optimized dual-band antenna has been studied in [111, 114, 116, 117], and could be divided into two different approaches. One approach starts from a rectangular antenna, and then discretize the antenna geometry into uniform grids, and apply GA to decide whether each grid is reserved or removed from the geometry [111, 114]. The final reserved grids form the new antenna geometry. The other approach defines some dimensions of geometry, such as the patch length, width, and so on, for a given antenna and automatically tunes those values [116, 117]. As far as we know, the notch and slit-loaded provide extra resonant frequencies and introduce new band into original antenna [120]. Hence the approach in [111, 114] can work well, but their ability depends on the grid's size.

Based on our experience and technique of using GA to optical proximity correction (OPC) [32,122], we implement an optimization method to enhance the receiving and transmitting abilities of the given antenna. OPC is the process of modifying the geometries of the layouts to compensate for the non-ideal properties of the lithography process. Given the shapes desired on the wafer, the mask is modified to improve the reproduction of the critical geometry. In the our developed OPC approach, original layout patterns are divided into small edge and corner segments which are to be moved during OPC. The movements of those segments are then optimized with respect to the calculated exposed results using the GA algorithm. According to the this idea, we propose a similar approach for the antenna shape design. The proposed method partitions the edges of the antenna into some sections and then those sections can be shifted in the normal direction of the edges to improve the return loss of the antenna. This approach makes the variation of geometry has more freedom. It can extend the geometry without limited boundaries and can construct geometry without specific form. The involved simulator solves the Maxwell's equations by finite element method. The Maxwell's equations govern the performance in electromagnetics and the simulated results are applied to evaluate the new generated antenna. Z-shaped antenna is used in our examination which has radiation elements and ground plane on the same plane and it has only one clear band. To enhance the other band, we try to add the discontinuities for current on the patch, i.e., change the geometry of the radiation element. The result shows the evolved antenna satisfies the specification.

7.2 The Finite Element Method in Electromagnetics

In this section, we state the Maxwell's equations which are fundamental in electromagnetics [124]. The Maxwell's and continuity equations in the differential form are

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}, \tag{7.1}$$

$$\nabla \times \mathbf{H} = \frac{\partial \mathbf{D}}{\partial t} + \mathbf{J}, \tag{7.2}$$

$$\nabla \cdot \mathbf{D} = \rho, \tag{7.3}$$

$$\nabla \cdot \mathbf{B} = 0, \tag{7.4}$$

from Eqs. (7.2), (7.3), and (7.4), we can derive

$$\begin{aligned} \nabla \cdot (\nabla \times \mathbf{H}) &= \nabla \cdot \frac{\partial \mathbf{D}}{\partial t} + \nabla \cdot \mathbf{J} \\ &= \frac{\partial \rho}{\partial t} + \nabla \cdot \mathbf{J}, \end{aligned} \tag{7.5}$$

and $\nabla \cdot (\nabla \times \mathbf{H}) = 0$, then we can get

$$\nabla \cdot \mathbf{J} = -\frac{\partial \rho}{\partial t}, \tag{7.6}$$

where **E** is electric field intensity, **D** is electrical flux density, **H** is magnetic field intensity, **B** is magnetic flux density, **J** is electric current density, and is electric charge density. The bold face denotes vector. Consider the time-harmonic e^{jwt}, Eqs. (7.1), (7.2), and (7.6) can be rewritten in the following forms

$$\nabla \times \mathbf{E} = -jw\mathbf{B}, \tag{7.7}$$

$$\nabla \times \mathbf{H} = jw\mathbf{D} + J, \tag{7.8}$$

and

$$\nabla \cdot \mathbf{J} = -jw\rho, \tag{7.9}$$

Based on the time-harmonic case with constitutive relations, the vector wave equations can be derived as

$$\nabla \times (\frac{1}{\mu_r}\nabla \times \mathbf{E}) - k_0^2\varepsilon_r\mathbf{E} = -jk_0Z_0\mathbf{J}, \tag{7.10}$$

$$\nabla \times (\frac{1}{\varepsilon_r}\nabla \times \mathbf{H}) - k_0^2\mu_r\mathbf{H} = \nabla \times (\frac{\mathbf{J}}{\varepsilon_r}), \tag{7.11}$$

Now we introduce how to apply the finite element method (FEM) to solve Maxwell's equations for the antenna electromagnetic problem. To derive the FEM formulations, first we consider the boundary value problem in 3D with the general form

$$-\frac{\partial}{\partial x}(\alpha_x\frac{\partial\phi}{\partial x}) - \frac{\partial}{\partial y}(\alpha_y\frac{\partial\phi}{\partial y}) - \frac{\partial}{\partial z}(\alpha_z\frac{\partial\phi}{\partial z}) + \beta\phi = f, \forall(x,y,z) \in V, \tag{7.12}$$

and the boundary conditions are $\phi = p$ on S_1 and

$$(\alpha_x\frac{\partial\phi}{\partial x}\hat{x} + \alpha_y\frac{\partial\phi}{\partial y}\hat{y} + \alpha_z\frac{\partial\phi}{\partial z}\hat{z}) \cdot \hat{n} + \gamma\phi = q, \tag{7.13}$$

on S_2. ϕ denotes the unknown function, V is the problem domain, S_1 is the surface with Dirichlet boundary condition, S_2 is the surface with the boundary condition of the third kind, and other parameters are known when converting the physical problem into the boundary value problem. In our examined antenna case, as shown in Fig. 7.1, the problem domain are the gray areas surrounded by the edges a, b, c and d; and the boundary conditions of all the edges, such as absorbing boundary condition (ABC) [124–126], perfectly matched layer (PML) [127, 128], etc., satisfy the third kind boundary condition. The black rectangular frame near the upper line is the contact for input excitation, and it is the Dirichlet boundary condition.

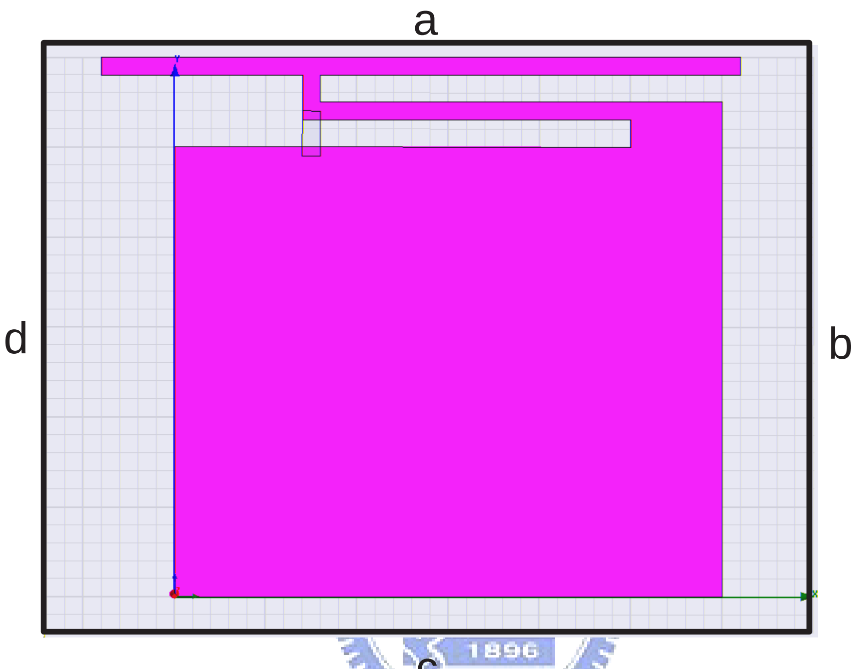


Figure 7.1: An illustration of the problem domain and boundary conditions in the examined antenna. The problem domain is surrounded by the edges a, b, c and d; and all edges satisfy the third kind boundary conduction. The black rectangular frame near the upper line is the contact for input excitation.

The corresponding variational problem for above boundary value problem is

$$\frac{\partial F(\phi)}{\partial \phi} = 0, \tag{7.14}$$

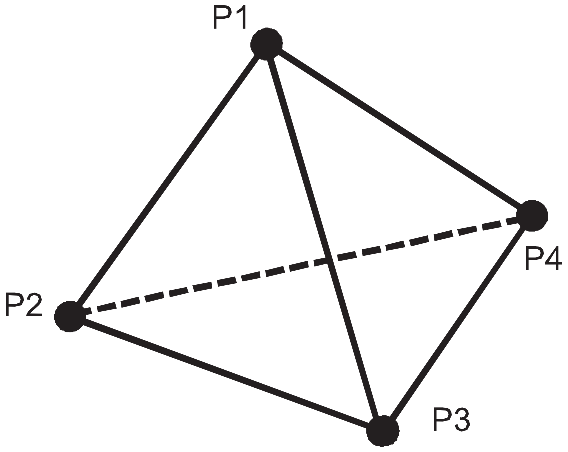


Figure 7.2: The linear tetrahedral element. The labels for four nodes are local indices in FEM.

where

$$\begin{aligned} F(\phi) = \frac{1}{2}\int\int\int_V [\alpha_x(\frac{\partial\phi}{\partial x})^2 + \alpha_y(\frac{\partial\phi}{\partial y})^2 + \alpha_z(\frac{\partial\phi}{\partial z})^2 + \beta\phi^2]dV \\ + \int\int_{S_2}(\frac{\gamma}{2}\phi^2 - q\phi)dS - \int\int\int_V f\phi dV, \end{aligned} \tag{7.15}$$

and $F(\phi)$ is named functional. The unknown function can be obtained by solving Eq. (7.15). To compute the solution, FEM is applied. FEM first discretizes the 3D problem domain using tetrahedral elements and calculates the expression of the unknown function on each element. For example, Fig. 7.2 shows a linear tetrahedral element and it mimics the unknown function as

$$\phi^e(x, y, z) = a^e + b^e x + c^e y + d^e z. \tag{7.16}$$

After discretization, Eq. (7.16) can be transferred into the matrix form such as

$$\begin{aligned}\{\frac{\partial F(\phi)}{\partial \phi}\} &= \sum_{e=1}^{M}\{\frac{\partial \bar{F}^e(\phi^e)}{\partial \phi^e}\} + \sum_{s=1}^{M_s}\{\frac{\partial \bar{F}_b^s(\phi^s)}{\partial \phi^s}\} \\ &= \sum_{e=1}^{M}([\bar{k}^e]\{\bar{\phi}^e\} - \{\bar{b}^s\}) + \sum_{s=1}^{M_s}([\bar{k}^s]\{\bar{\phi}^s\} - \{\bar{b}^s\}) = \{0\}. \end{aligned} \tag{7.17}$$

In above expression, $\{A\}$ denotes a column vector, $[A]$ denotes a matrix, A^e is the entry corresponding to discretized element, A^s is the entry corresponding to surface, and the bar over entries in the equation denotes the global representation which transfers from the local representation without bar. Finally, the solution will be obtained by solving the matrix Eq. (7.17).

Comparing Eq. (7.12) with Eqs. (7.10) and (7.11), the parameters in Eq. (7.12) can be extracted. Also, the boundary conditions continuity of fields between two materials in the electromagnetic problem can also relate to Eqs. (7.16) and (7.17). For open structures, in order to truncate the problem into the finite domain, ABC and PML are used to model the boundary conditions and to avoid the reflection at artificial boundaries [123–128]. The radiation pattern and the return loss are then evaluated to show the antenna property [129].

7.3 The Optimization Method Using Genetic Algorithm

As a designer drew a blueprint of an antenna for specific purpose, a fine tune process should be performed for further improvement. We already know the geometry of the antenna

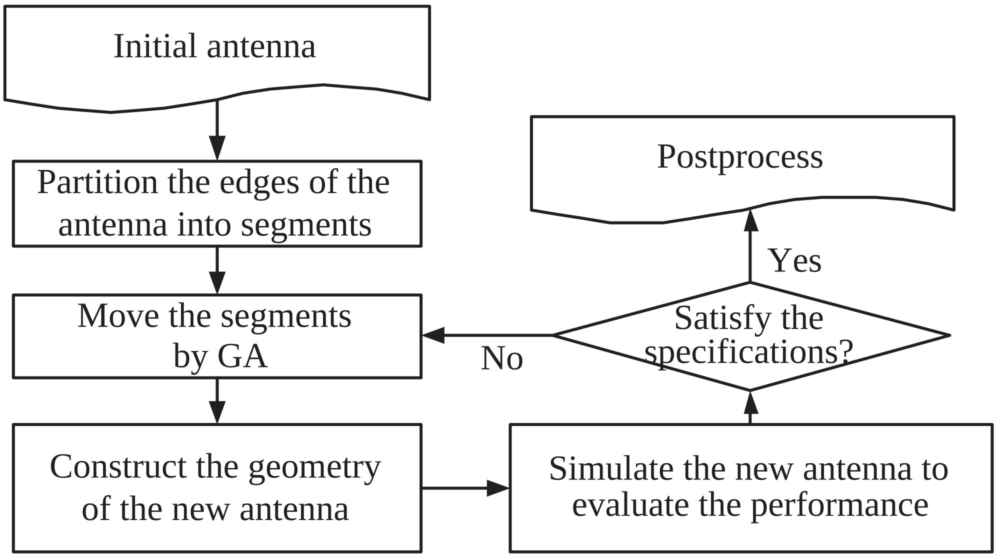

Initial antenna
Partition the edges of the antenna into segments
Move the segments by GA
Construct the geometry of the new antenna
Simulate the new antenna to evaluate the performance
Satisfy the specifications?
No
Yes
Postprocess

Figure 7.3: A flowchart for the proposed optimization approach in antenna design. In this procedure, after constructing the geometry of the new antenna, the corresponding input file for extern simulator is also generated and passed to the external simulator for further simulation and analysis. In this work, HFSS is applied as the external simulator to evaluate the performance of the new antenna.

affects its performance. However, the major difficulty is how the antenna shape affects its performance. For an antenna designer, when one fine tunes the shape, it is just like making a wild guess. To automate the search for the desired antenna shape, Fig. 7.3 shows the flowchart for the proposed optimization approach, and we have implemented this optimization procedure in the UOF [68, 69]. As shown in Fig. 7.3, the edges of the initial antenna are firstly partitioned into several segments. For this antenna design problem, the parameters in GA are the shift offset of each segment, shown in Fig. 7.4, encoded as a string of discrete steps for evolution. Each segment from 1 to 15, shown in Fig. 7.4, can shift inward or outward for within a specified range, while the value of parameter 16 to 20 indicates the width of the top line. The GA is adopted to decide the movement or thickness of each segment. In the next step, the geometry of new antenna is constructed according to the evolutionary result from GA, and then generates the input file for the external simulator. In this examination, the HFSS [130] is applied to simulate the new generated antenna and evaluate the corresponding performance. Once the specification is matched or the predefined deadline is reached, the procedure stops and outputs the final result.

There are several components in the GA [1–3, 68, 69], such as the problem definition, encoding method, fitness evaluation, selection method, crossover procedure, and mutation scheme have to be performed for an evolutionary process. A computational procedure of applying GA in the optimal antenna design problem is given below.

```
GA with antenna design problem
Begin
    Partition edges of antenna into segments
    For j = 1 to Number of segments
      GeneEncode( Segment[j] )
    End For
    While ErrorNorm not satisfies the stop criteria
        Evolution()
        UpdateSegment()
        ErrorNorm = return_loss(optimized) - return_loss(original)
    End While
    Output optimized antenna
End
```

According to this procedure, the geometry of antenna is evolved continuously until the stop criteria are matched. The following sections state each step of GA in the implementation of the antenna design problem with genetic algorithm.

7.3.1 Problem Definition

The goal in the antenna design problem is to obtain an optimized antenna whose geometry produces the better property. In this examination, the return loss (S11) is considered as the criterion for the evolution. It means that the GA will find out the best configuration of the antenna shape, and the extracted return loss of the optimized antenna is less than the return loss of the original antenna. The relationship between the original and the optimized antenna can be written as follows: Return_loss(opimized) $<$ Return_loss(original) where

Return_loss(opimized) is the return loss of the optimized antenna and Return_loss(original) means the return loss of the original antenna.

7.3.2 Encoding Method

Encoding method is the procedure that encodes the target parameters into genes. For the proposed method, we encode the movements or the width of each segment into genes. All unknowns to be extracted are floating-point numbers. We transform these continuous floating-point numbers into discrete steps (Psteps) through step function as shown in Eq. (7.18) instead of real numbers, and we encode the discrete steps as genes on chromosomes. The discrete steps show the strongly combinatorial properties, and we have found this representation has better results in crossover and mutation.

$$P_{value} = P_{min} + Psteps\frac{P_{max} - P_{min}}{R_{esolution}}. \tag{7.18}$$

The P_{value} is the real value of the movement or width which is a float-point number. P_{min} and P_{max} are the minimum and maximum values of each parameter, and Resolution is the specific resolution in the range between P_{min} and P_{max}.

7.3.3 Fitness Evaluation

The fitness evaluation is to evaluate the fitness score for each chromosome. The fitness score can be seen as the accommodation status of each chromosome in the current environment, and it usually presents the differences between the target and the chromosome. The fitness evaluation function in this examination can be written as follows:

$$F = \sum_{i=f_{start}}^{f_{end}} (return\_loss(optimized)_i - return\_loss(original)_i), \tag{7.19}$$

where $Return\_loss(opimized)_i$ and $Return\_loss(original)_i$ are the return losses of the original and optimized antenna on the frequency i. f_{start} and f_{end} are the starting and ending frequencies which designates the range of the operating frequencies. The return loss value describes the reduction in the amplitude of the reflected energy, as compared to the forward energy. It is an important parameter to measure the performance of the antenna and can be extracted by the external simulator, such as HFSS.

7.3.4 Selection Method

After fitness score for each chromosome is obtained, a selection method selects chromosomes that will stay in the population and breed next generation. There are many selection schemes, such as ranking selection, roulette wheel selection, and tournament selection. The ranking selection selects chromosomes with the rule of first-rate score. The roulette wheel selection gives each chromosome a different chosen rate by the average score and the fitness scores of each chromosome; and the tournament selection chooses several pairs of chromosomes and selects the better one of each pair. In this work, the ranking selection is chosen for its simplicity.

7.3.5 Crossover Procedure and Mutation Scheme

When selection has been carried out, we will perform the crossover procedure. Crossover procedure mates two chromosomes selected by the selection method to generate new chromosomes. To generate offspring, the crossover operator gives a few cuts on the parent chromosomes and exchanges the genes. In this examination, the single-point cut crossover scheme is adopted. After the crossover procedure, a certain rate of the newborn chromosomes mutates into another different chromosomes. The mutation rate is typically less than 1%. The mutation scheme may act in different ways. In the proposed method, it raises up the mutation rate when the behavior tends to saturation situation and decreases the mutation rate when the population achieves to high diversity. After the above steps are complete, the GA evaluates the next population and stops till certain stop criteria is reached.

7.4 The Achieved Results and Discussion

Figure 7.4 shows the original shape of the examined antenna. It is a Z-shaped antenna and has radiation elements and ground plane on the same plane. As shown in Fig. 7.4,

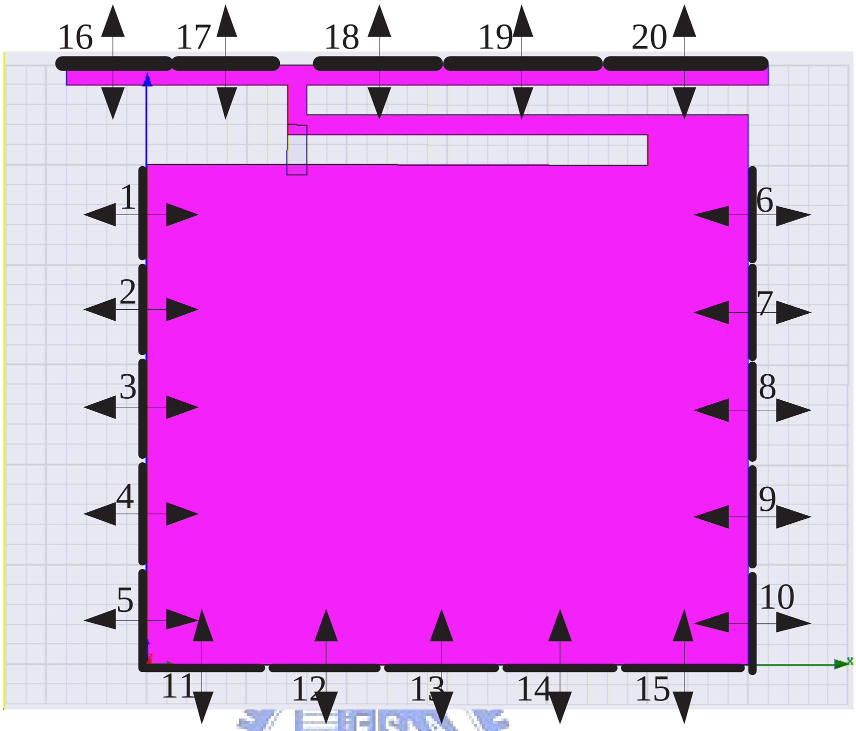


Figure 7.4: The original geometry of examined antenna and the illustration of partitioned segments. The upper line presents the ground plane and the lower rectangle behaves as the radiation element. The black rectangular frame near the upper line is the contact for input excitation. The partitioned segments allow us to perturb the geometry without spatial limitation in a specific region.

the top line is the ground plane, the bottom rectangle represents the radiation element, and the segments 1∼20 are the partitions for evolution. Figure 7.5 shows the return loss (S11) of the original antenna. The dual operating frequencies are set to 2.6 and 5 GHz. The frequency 2.6 GHz is referred as the ”S-band” used for mobile broadcasting and the

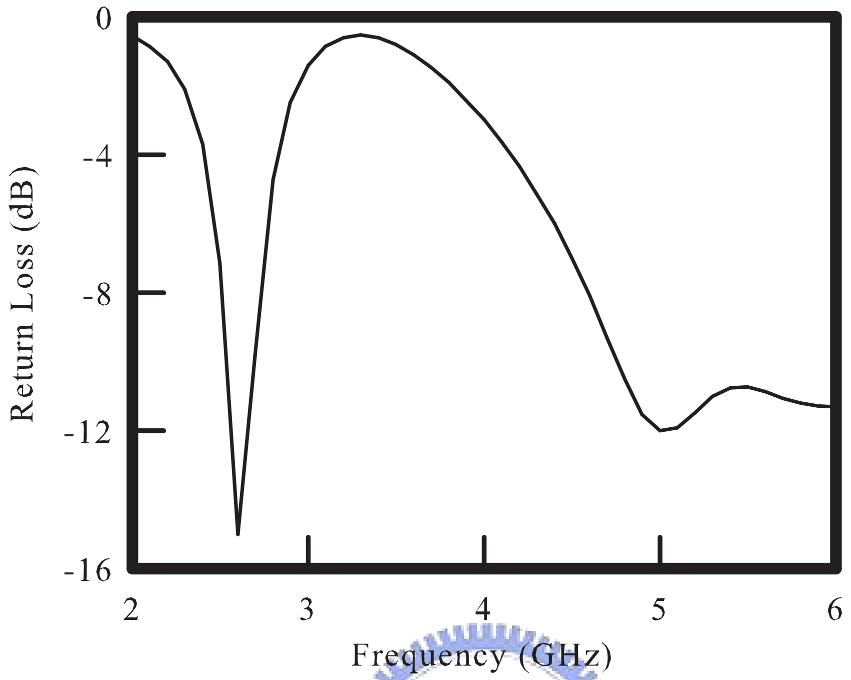


Figure 7.5: The return loss (S11) of the original geometry. The antenna is a dual-band design but has only one clear band at 2.6GHz. The return loss on the frequency 5 GHz still has room for improvement.

frequency 5 GHz is used by the 802.11a WLAN protocol. Our target is to improve the return loss for both operating frequencies 2.6 and 5 GHz. Figure 7.6 shows the optimized geometry, and Fig. 7.7 illustrates the corresponding result. Compared with Fig. 7.5, the return loss on the frequencies of 2.6 and 5 GHz is down to 20 dB for the optimized antenna which has a significant improvement.

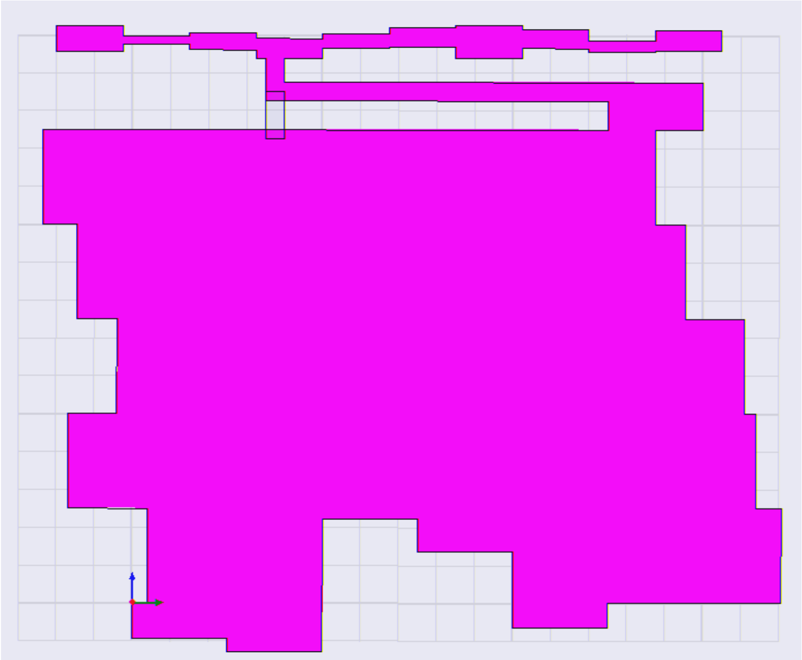


Figure 7.6: An optimized geometry of the examined antenna after our optimization scheme. The geometry of radiation element is obviously tuned by GA to enhance the transmitting and receiving properties of the antenna.

1896

As shown in Fig. 7.4, the original Z-shape antenna locates in the xy-plane and its normal vector is in z direction. For the radiation property of the desired antenna, we want to hold the omidirectional radiation in the xz-plane after the proposed optimization scheme. As shown in Figs. 7.8 and 7.9, the original antenna has omidirectional gain pattern in the xz-plane at 2.6GHz and 4.9GHz. After optimization, the gain patterns are still omidirectional in the xz-plane at the dual operating frequencies, as shown in Figs. 7.10 and 7.11. From Figs. 7.8-7.11, we notice that the gain of the optimized antenna is close to that of

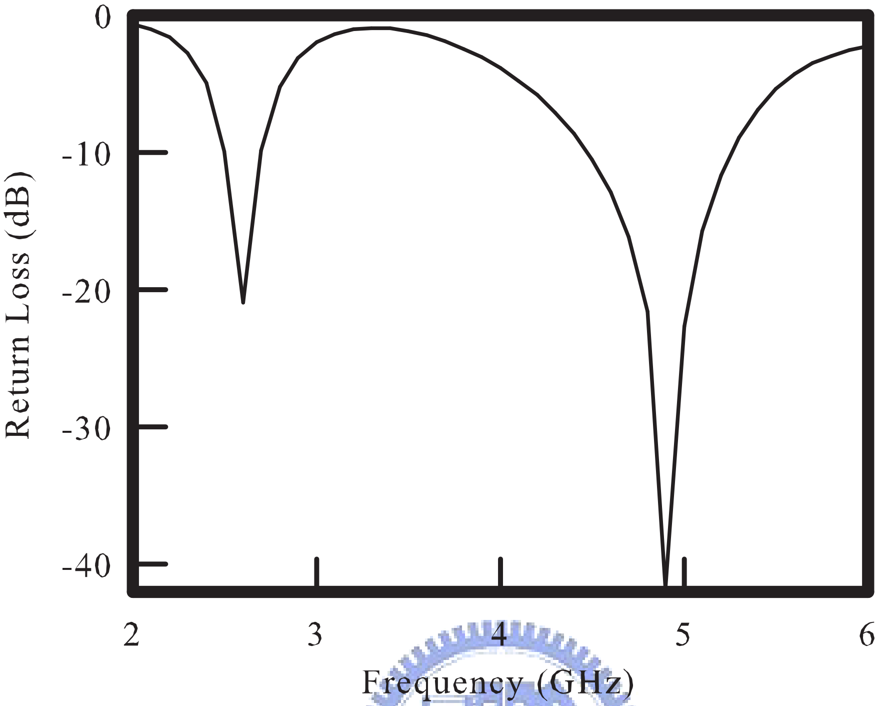


Figure 7.7: The return loss (S11) of the optimized geometry. One can find that the return losses corresponding to dual operating frequencies are less than 20 dB after the proposed optimization scheme.

the original one at both operating frequencies but the return losses are obviously improved after the optimization. The fitness score versus the number of evolutionary generations is shown in Fig. 7.12, and one improved antenna is given after ten evolutionary generations in our examination.

In this application, one simulation-based optimization method for the design of antenna is presented. This method minimizes the return loss under the frequencies 2.6 and 5 GHz.

The receiving and transmitting properties of the optimized antenna are simulated and the resulting gain pattern is still omidirectional in the xz-plane. The gain is close to the original antenna. The optimized antenna has gain pattern as good as the original one but the operating frequencies are enhanced for dual band design. To validate this method, more antennas with different operating frequencies will be examined.

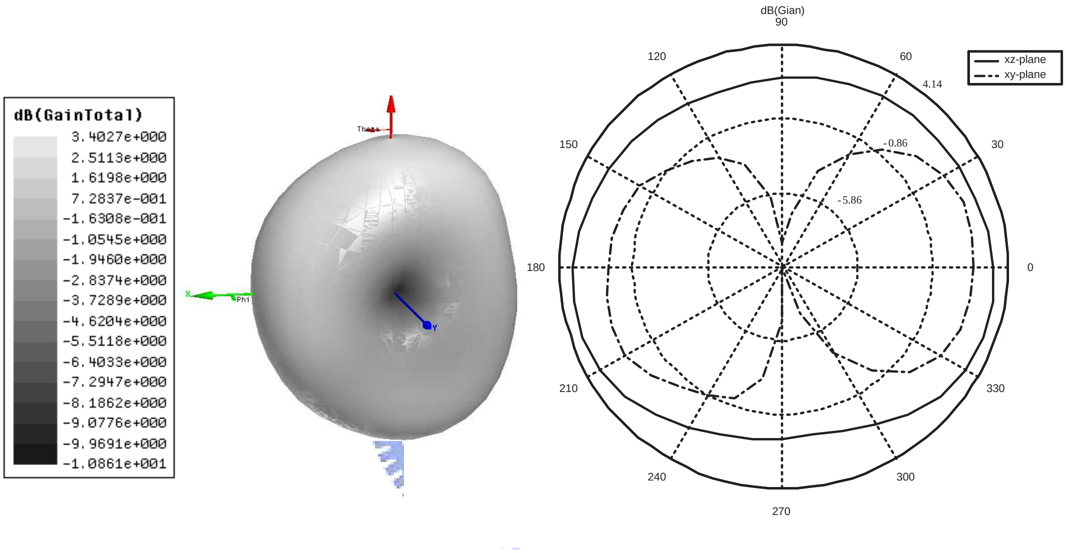


Figure 7.8: The gain pattern for the original antenna at 2.6GHz. The left object is the 3D gain pattern. The right one is the 2D gain pattern. The solid line presents the gain pattern in the xz-plane, and the dashed line is the gain pattern in the xy-plane.

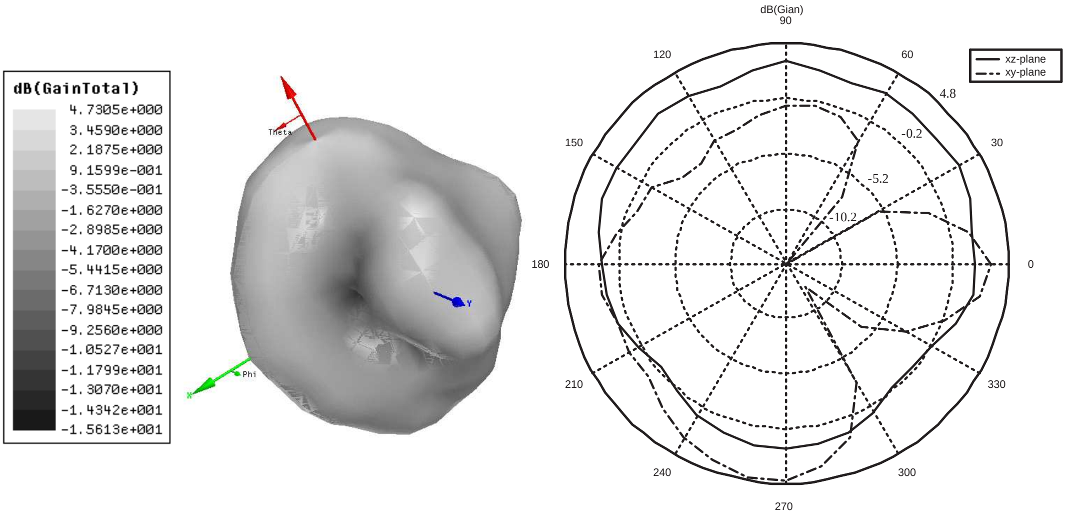


Figure 7.9: The gain pattern for the original antenna at 4.9GHz. The left object is the 3D gain pattern. The right one is the 2D gain pattern. The solid line presents the gain pattern in the xz-plane, and the dashed line is the gain pattern in the xy-plane.

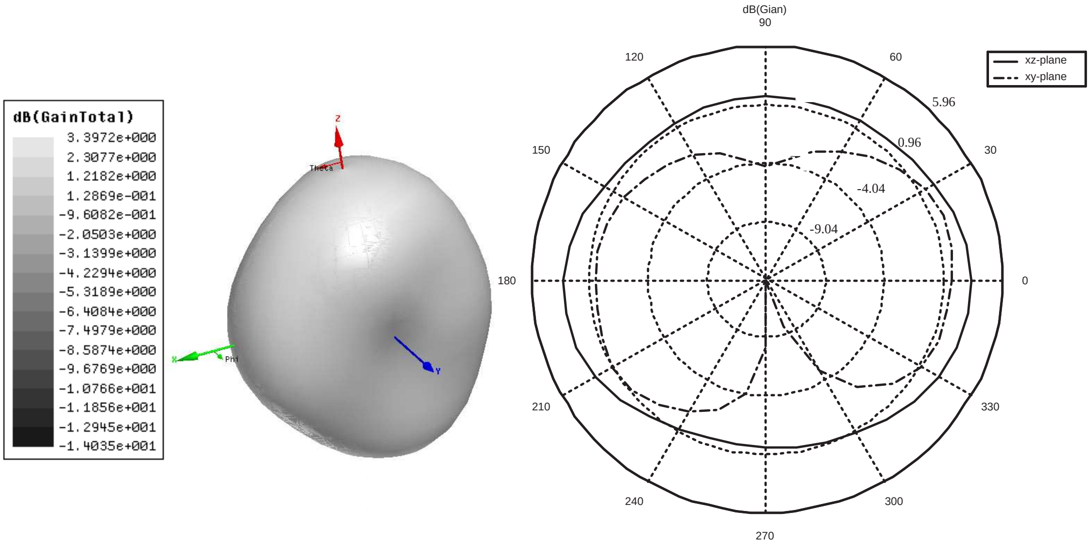


Figure 7.10: The gain pattern for the optimized antenna at 2.6GHz. The left object is the 3D gain pattern. The right one is the 2D gain pattern. The solid line presents the gain pattern in the xz-plane, and the dashed line is the gain pattern in the xy-plane.

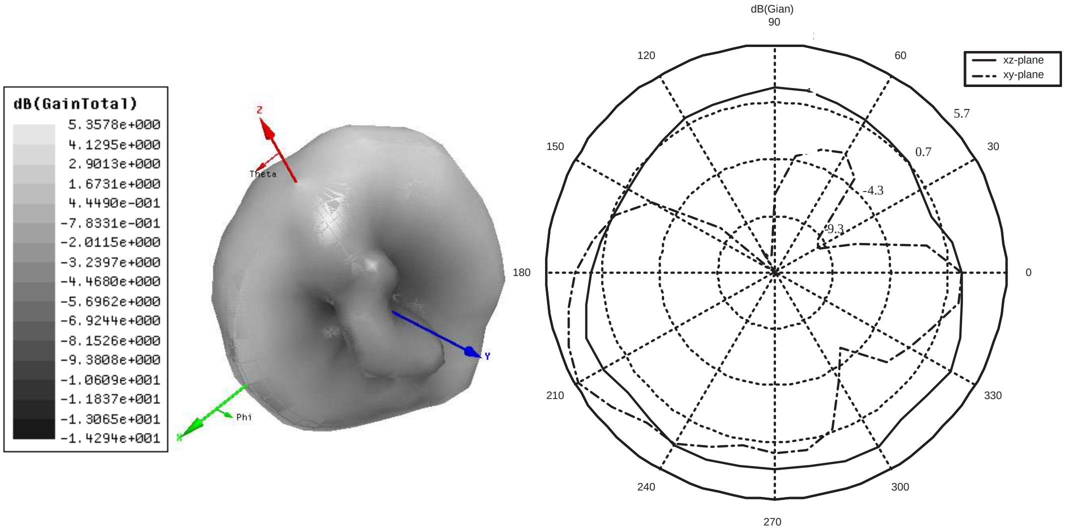
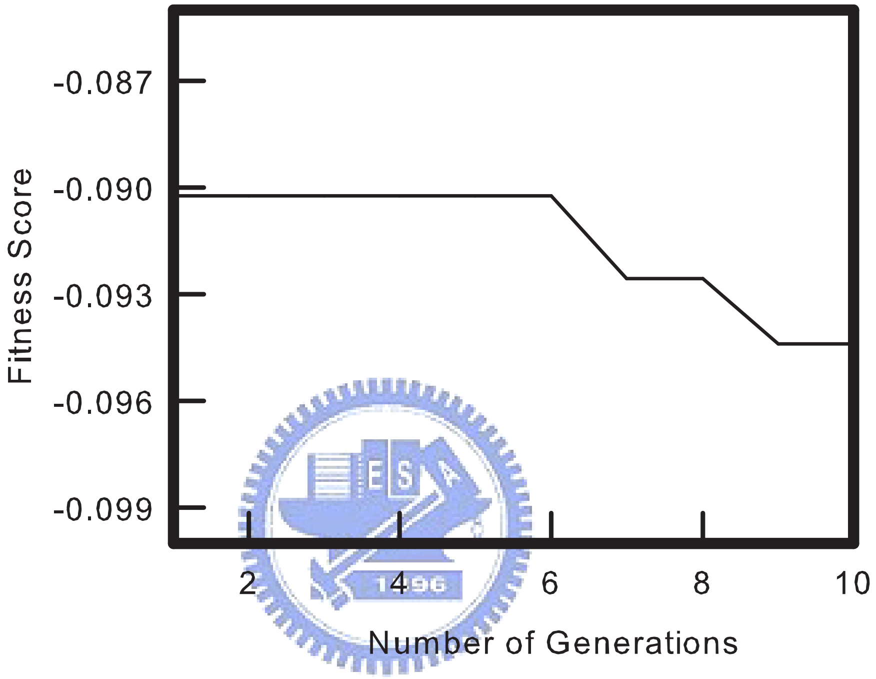


Figure 7.11: The gain pattern for the optimized antenna at 4.9GHz. The left object is the 3D gain pattern. The right one is the 2D gain pattern. The solid line presents the gain pattern in the xz-plane, and the dashed line is the gain pattern in the xy-plane.



1896

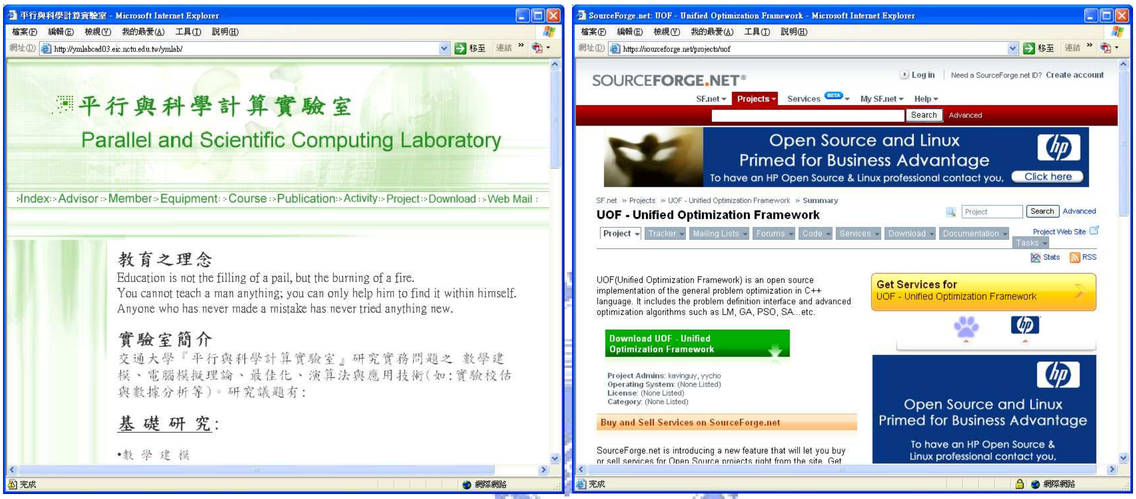
Figure 7.12: The fitness score versus the number of generations in the examination. The mutation rate in this examination is 0.01 and single-point cut crossover scheme is adopted. The proposed optimization scheme enhances the antenna property within ten generations in this examination.

Chapter 8

Conclusions and Future Work

This study develops an object-oriented UOF for general problem optimization. The UOF is based on biological inspired techniques, numerical deterministic methods and C++ objective design, and is demonstrated to be an alternative to optimization operations on various problems. The UOF bridges the gap between problems and solvers. The high level code reutilization allows the adaptation to new problem and solver quickly. The hybrid approach which integrates both evolutionary and numerical algorithms is implemented in the UOF, and successfully customizes UOF to deal with the following electronic design applications: 65nm CMOS device fabrication, the equivalent circuit model parameter extraction, the VLSI circuit design problem, and the antenna shape optimization. In the utilization of UOF for inverse doping profile problems of the 65 nm CMOS devices, the fluctuation of electrical characteristics is simultaneously considered and minimized in the optimization procedure. Integration of device and process simulation is implemented to evaluate device performances, where the developed approach enables us to extract optimal recipes which are subject to targeted device specification. For equivalent circuit model parameter extraction problems, the hybrid intelligent method is applied to perform parameter extraction and demonstrated excellent results. In the application of VLSI circuit design, two different approaches including the simulation-based method and the computational statistics approach are proposed and compared by the examination of the current mirror amplifier circuit design. In the final application, an investigation for the design of antenna pattern applied for mobile broadcasting and the 802.11a WLAN is presented. A Z-shaped antenna is optimized with respect to the return loss by the UOF. Experimental results of these applications confirm capability of the proposed UOF. The proposed UOF can benefit the electronic design automation, and significantly help for engineers to complete their optimization tasks. The developed open-source project is available in the public domain, as shown in Fig. 8.1 (http://140.113.87.143/ymlab/uof/ or http://sourceforge.net/projects/uof/).

In the future, more applications, such as the electrostatic discharge (ESD) circuit design and the advanced design of phase change memories, can be examined and implemented by the UOF. The graphical user interface can be included in the UOF to provide more convenient ways for users to utilize UOF.


平行與科學計算實驗室 - Microsoft Internet Explorer
檔案(F) 編輯(E) 檢視(V) 我的最愛(A) 工具(T) 說明(H)
網址(D) http://ymlabcad03.eic.nctu.edu.tw/ymlab/
移至 連結
平行與科學計算實驗室
Parallel and Scientific Computing Laboratory
Index Advisor Member Equipment Course Publication Activity Project Download Web Mail
完成
網際網路
SourceForge.net: UOF - Unified Optimization Framework - Microsoft Internet Explorer
網址(D) https://sourceforge.net/projects/uof
SOURCEFORGE.NET®
Log in
Need a SourceForge.net ID? Create account
SF.net Projects Services BETA My SF.net Help
Search Advanced
Open Source and Linux
Primed for Business Advantage
To have an HP Open Source & Linux professional contact you,
Click here
SF.net » Projects » UOF - Unified Optimization Framework » Summary
Project Search Advanced
Project Tracker Mailing Lists Forums Code Services Download Documentation Tasks
Project Web Site
Stats RSS
Get Services for
UOF - Unified Optimization Framework
Download UOF - Unified Optimization Framework
Buy and Sell Services on SourceForge.net
Open Source and Linux
Primed for Business Advantage
To have an HP Open Source & Linux professional contact you,

教育之理念

Education is not the filling of a pail, but the burning of a fire.
You cannot teach a man anything; you can only help him to find it within himself.
Anyone who has never made a mistake has never tried anything new.

實驗室簡介

交通大學『平行與科學計算實驗室』研究實務問題之 數學建模、電腦模擬理論、最佳化、演算法與應用技術(如:實驗校估與數據分析等)。研究議題有:

基礎研究:

•數學建模

UOF - Unified Optimization Framework

UOF(Unified Optimization Framework) is an open source implementation of the general problem optimization in C++ language. It includes the problem definition interface and advanced optimization algorithms such as LM, GA, PSO, SA...etc.

Project Admins: kavinguy, yycho
Operating System: (None Listed)
License: (None Listed)
Category: (None Listed)

SourceForge.net is introducing a new feature that will let you buy or sell services for Open Source projects right from the site. Get

Figure 8.1: The source code could be downloaded from http://140.113.87.143/ymlab/uof/ or http://sourceforge.net/projects/uof/.

Bibliography

[1] Y. Li, Y.-Y Cho, "Intelligent BSIM4 Model Parameter Extraction for Sub-100 nm MOSFET Era," Jpn. J. App. Phy., vol. 43, pp. 1717-1722, 2004.

[2] Y. Li, Y.-Y. Cho, C.-S. Wang, K.-Y. Huang, "A Genetic Algorithm Approach to InGaP/GaAs HBT Parameters Extraction and RF Characterization," Jpn. J. App. Phy., vol. 42, pp. 2371-2374, 2003.

[3] Y. Li, C.-T. Sun, C.-K. Chen, "A Floating-Point Based Evolutionary Algorithm for Model Parameters Extraction and Optimization in HBT Device Simulation," in: Advances in Soft Computing - Neural Networks and Soft Computing, Edited by L. Rutkowski and J. Kacprzyk, Physica-Verlag, 2003, pp. 364-369.

[4] S.-J. Jou, K.-F. Liu, C. Su, "Circuits Design Optimization Using Symbolic Approach," Proc. IEEE Int. Symp. Cir. and Sys., pp. 1396-1399, 1995.

[5] J.W. Herrmann, B.F. Conaghan, L. Henn-Lecordier, P. Mellacheruvu, M.-Q. Nguyen, G.W. Rubloff, R.Z. Shi, “Understanding the impact of equipment and process changes with a heterogeneous semiconductor manufacturing simulation environment,” Proc. Winter Sim. Conf., pp. 1491-1498, 2000.

[6] T. Binder, C. Heitzinger, S. Selberherr, “A Study on Global and Local Optimization Techniques for TCAD Analysis Tasks,” IEEE Trans. CAD, vol. 23, pp. 814-822, 2004.

[7] A. J. Kerkhoff, R. L. Rogers, H. Ling, “Design and Analysis of Planar Monopole Antennas Using a Genetic Algorithm Approach,” IEEE Trans. Antenn. and Propa., vol. 52, pp. 2709-2718, 2004.

[8] M. Wall, GAlib a C++ Library of Genetic Algorithm Components. Mass. Inst. of Technol., Cambridge., 2000. (online available: http://lancet.mit.edu/ga)

[9] B. Gehlsen, B. Page, “A framework for distributed simulation optimization,” Proc. 2001 Winter Sim. Conf., pp. 508-514, 2001.

[10] A. Mendes, P. França, P. Moscato, “NP-Opt: an optimization framework for NP problems,” Proc. IV SIMPOI/POMS, pp. 82-89, 2001.

[11] W.-C. Liu, “Design of a Multiband CPW-fed Monopole Antenna Using a Particle Swarm Optimization Approach,” IEEE Trans. Antenn. and Propa., vol. 53, pp. 3273-3279, 2005.

[12] K. A. De Jong, An analysis of behavior of a class of genetic adaptive systems, Doctoral Dissertation, University of Michigan, Dissertation Abstracts International, 1975.

[13] R. H. Byrd, P. Lu, J. Nocedal, C. Zhu, “A limited memory algorithm for bound constrained optimization,” SIAM J. Sci. Comput. vol. 16, pp. 1190-1208, 1995.

[14] S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, Numerical Recipes in C, Cambridge University Press, pp. 542-549, 1988.

[15] P. Mendes, D. Kell, “Non-linear optimization of biochemical pathways: applications to metabolic engineering and parameter estimation,” Bioinfo., vol. 14, pp. 869-883, 1998.

[16] J. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, Mich., 1975.

[17] J. Kennedy, R. C. Eberhart, “Particle swarm optimization,” Proc. IEEE Int. Conf. Neural Networks, pp. 1942-1948, 1995.

[18] M. Dorigo, L. M. Gambardella, “Ant colony system: A cooperative learning approach to the traveling salesman problem,” IEEE Trans. Evol. Comput., vol. 1, pp. 53-66, 1997.

[19] J. Kim, S. Pramanik, M. J. Chung, “Multiple Sequence Alignment using Simulated Annealing,” Comp. Applic. Biosci, vol. 10, pp. 419-426, 1994.

[20] M. Ishikawa, T. Toya, M. Hoshida, K. Nitta, A. Ogiwara, M. Kanehisa, “Multiple Sequence Alignment by Parallel Simulated Annealing,” Comp. Applic. Biosci., vol. 9, pp. 267-273, 1993.

[21] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison Wesley Professional, 1994.

[22] S. Selberherr, Analysis and Simulation of Semiconductor Devices, New York: Springer-Verlag, 1984.

[23] Y. Li, H.-M. Chou, J.-W. Lee, “Investigation of Electrical Characteristics on Surrounding-Gate and Omega-Shaped-Gate Nanowire FinFETs,” IEEE Trans. Nanotech., vol. 4, pp. 510-516, 2005.

[24] Y. Li, S.-M. Yu, “A Parallel Adaptive Finite Volume Method for Nanoscale Double-gate MOSFETs Simulation,” J. Comput. and Appl. Math., vol. 175, pp. 87-99, 2005.

[25] K.-Y. Huang, Y. Li, C.-P. Lee, “A Time Domain Approach to Simulation and Characterization of RF HBT Two-Tone Intermodulation Distortion,” IEEE Trans. Microwave Theo. and Tech., vol. 51, pp. 2055-2062, 2003.

[26] Y. Li, “A Parallel Monotone Iterative Method for the Numerical Solution of Multidimensional Semiconductor Poisson Equation,” Comput. Phy. Comm., vol. 153, pp. 359-372, 2003.

[27] Y. Li, T.-S. Chao, and S. M. Sze, “A Novel Parallel Approach for Quantum Effect Simulation in Semiconductor Devices,” Int. J. Model. & Sim., vol. 23, pp. 94-102, 2003.

[28] R. Gupta, D. J. Allstot, “Parasitic-aware design and optimization of CMOS RF integrated circuits,” Proc. IEEE Int. Microwave Symp., pp. 1867-1870, 1998.

[29] P. Vancorenland, G. Van der Plas, M. Steyaert, G. Gielen, W. Sansen, “A layout-aware synthesis methodology for RF circuits,” Proc. IEEE/ACM Int. Conf. CAD, pp. 358-362, 2001.

[30] E. Seevinck, F.J. List, J. Lohstroh, “Static-noise margin analysis of MOS SRAM cells,” IEEE J. Solid-State Cir., vol. 22, pp. 748-754, 1987.

[31] Y. Li, C.-S. Lu, “Characteristic Comparison of SRAM Cells with 20 nm Planar MOSFET, Omega FinFET and Nanowire FinFET,” Proc. IEEE Nanotech. Conf., pp. 339-342, 2006.

[32] S.-M. Yu, Y. Li, "A Pattern-Based Domain Partition Approach to Parallel Optical Proximity Correction in VLSI Designs," Proc. 19th IEEE Int. Parall. and Dist. Proce. Symp., 2005.

[33] Y. Li and S.-M. Yu, "A Coupled-Simulation-and-Optimization Approach to Nanodevice Fabrication with Minimization of Electrical Characteristics Fluctuation,"IEEE Trans. on Semiconductor Manufacturing. (to appear)

[34] Y. Li, S.-M. Yu and Y.-L. Li, "Application of A Unified Optimization Framework to Electronic Design Automation," Mathematics and Computers in Simulation. (to appear)

[35] Y. Li, Y.-L. Li and S.-M. Yu, "Design Optimization of a Current Mirror Amplifier Integrated Circuit Using a Computational Statistics Technique," Mathematics and Computers in Simulation. (to appear)

[36] Y. Li, S.-M. Yu and H.-M. Chen, "Process-Variation- and Random-Dopants-Induced Threshold Voltage Fluctuations in Nanoscale CMOS and SOI Devices," Microelectronic Engineering, Vol. 84, No. 9-10, Sep.-Oct. 2007, pp. 2117-2120.

[37] S.-C. Lo, Y. Li, and S.-M. Yu, "Analytical Solution of Nonlinear Poisson Equation for Symmetric Double-Gate Metal-Oxide-Semiconductor Field Effect Transistors," Mathematical and Computer Modelling, Vol. 46, No. 1-2, July 2007, pp. 180-188.

[38] S.-M. Yu, J.-W. Lee, and Y. Li, "An Optimal Silicidation Technique for Electrostatic Discharge Protection Sub-100 nm CMOS Devices in VLSI Circuit," Microelectronic Engineering, Vol. 84, No. 2, Feb. 2007, pp. 213-217.

[39] Y. Li and S.-M. Yu, "Parallel Numerical Solution to Large-Scale Eigenvalue Problem in Master Equation of Protein Folding Kinetics,"Second IEEE Workshop on High Performance Computing in Medicine and Biology (HiPCoMB 2006), Vienna University of Technology, Vienna, Austria, April 18-20, 2006.

[40] N. Sano, A. Hiroki, K. Matsuzawa, "Device modeling and simulations toward sub-10 nm semiconductor devices," IEEE Transactions on Nanotechnology, vol. 1, pp. 63-71, 2002.

[41] G. Pei, J. Kedzierski, P. Oldiges, M. Ieong and E.C.-C. Kan, "FinFET design considerations based on 3-D simulation and analytical modeling," IEEE Transactions on Electron Devices, vol. 49, pp. 1411-1419, 2002.

[42] S. Xiong and J. Bokor, "Sensitivity of Double-Gate and FinFET Devices to Process Variations," IEEE Transactions on Electron Devices, vol. 50, no. 11, pp. 2255-2261, 2003.

[43] F.-L. Yang, D.-H. Lee, H.-Y. Chen, C.-Y. Chang, S.-D Liu, C.-C. Huang, T.-X. Chung, H.-W. Chen, C.-C. Huang, Y.-H. Liu, C.-C. Wu, C.-C. Chen, S.-C. Chen, Y.-T. Chen,

Y.-H. Chen, C.-J. Chen, B.-W. Chan, P.-F. Hsu, J.-H. Shieh, H.-J. Tao, Y.-C. Yeo, Y. Li, J.-W. Lee, P. Chen, M.-S. Liang and C. Hu, “5nm-gate nanowire FinFET,” in Proceeding of Symposium on VLSI Technology, 2004, pp. 15-17.

[44] V. A. Sverdlov, T. J. Walls, and K.K. Likharev, “Nanoscale silicon MOSFETs: a theoretical study,” IEEE Transactions on Electron Devices, vol. 50, no. 9, pp. 1926-1033, 2003.

[45] S. M. Ramey, and D. K. Ferry, “Modeling of quantum effects in ultrasmall FD-SOI MOSFETs with effective potentials and three-dimensional Monte Carlo,” Physica B, vol. 314, pp. 350-353, March 2002.

[46] A.R. Alvarez, B.L. Abdi, D.L. Young, H.D. Weed, J. Teplik, and E.R. Herald, “Application of Statistical Design and Response Surface Methods to Computer-Aided VLSI Device Design,” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 7, no. 2, pp. 272-288, 1988.

[47] D.S. Boning and P.K. Mozumder, “DOE/Opt: A System for Design of Experiments, Response Surface Modeling, and Optimization Using Process and Device Simulation,” IEEE Transactions on Semiconductor Manufacturing, vol. 7, no. 2, pp. 233-244, 1994.

[48] G.J. Gaston and A.J. Walton, "The Integration of Simulation and Response Surface Methodology for the Optimization of IC Processes," IEEE Transactions on Semiconductor Manufacturing, vol. 7, no. 1, pp. 22-33, 1994.

[49] D.S. Boning, M.B. McIlrath, Jr. P. Penfield and E.M. Sachs, "A General Semiconductor Process Modeling Framework," IEEE Transactions on Semiconductor Manufacturing, vol. 5, no. 4, pp. 266-280, 1992.

[50] Y. Li and S.-M. Yu, "A Unified Quantum Correction Model for Nanoscale Single- and Double-Gate MOSFETs under Inversion Conditions," Nanotechnology, vol. 15, no. 8, pp. 1009-1016, 2004.

[51] Y. Li and S.-M. Yu, "Quantum correction simulation of random dopant-induced threshold voltage fluctuations in nanoscale metal-oxide-semiconductor structures," in Proceeding of IEEE Nanotechnology Conference, 2005, pp. 527-530.

[52] Y. Li and S.-M. Yu, "A Numerical Iterative Method for Solving Schrödinger and Poisson Equations in Nanoscale Single, Double and Surrounding Gate Metal-Oxide-Semiconductor Structures," Computer Physics Communications. Vol. 169, No. 1-3, July 2005, pp. 309-312.

[53] A. Asenov, G. Slavcheva, A.R. Brown, J.H. Davies and S. Saini, "Increase in the Random Dopant Induced Threshold Fluctuations and Lowering in Sub-100 nm MOSFETs

Due to Quantum Effects: A 3-D Density-Gradient Simulation Study," IEEE Transactions on Electron Devices, vol. 48, no. 4, pp. 722-729, 2001.

[54] K. A. Bowman, X. Tang, J.C. Eble and J.D. Menldl, "Impact of extrinsic and intrinsic parameter fluctuations on CMOS circuit performance," Solid-State Circuits, vol. 35, no. 8, pp. 1186-1198, 2000.

[55] X. Tang, V. K. De and J. D. Meindl, "Intrinsic MOSFET Parameter Fluctuations Due to Random Dopant Placement," IEEE Transactions on VLSI Systems, vol. 5, no. 4, pp. 369-376, 1996.

[56] D. A. Van Veldhuizen, J. B. Zydallis, and G. B. Lamont, "Evolutionary computing and optimization: Issues in parallelizing multiobjective evolutionary algorithms for real world applications," in Proceeding of ACM symposium on applied computing, 2002, pp. 595.

[57] Y. Li, "An automatic parameter extraction technique for advanced CMOS device modeling using genetic algorithm," Microelectronic Engineering, vol. 84, pp. 260-272, 2007.

[58] Y. Li and S.-M. Yu, "Comparison of Random Dopant-Induced Threshold Voltage Fluctuations in Nanoscale Single-, Double-, and Surrounding-Gate Field Effect Transistors," Japanese Journal of Applied Physics, vol. 45, no. 9A, pp. 6860-6865, 2006.

[59] Y. Li and S.-M. Yu, “Study of Threshold Voltage Fluctuations of Nanoscale Double Gate Metal-Oxide-Semiconductor Field Effect Transistors Using Quantum Correction Simulation,” Journal of Computational Electronics, vol. 5, no. 2-3, pp. 125-129, 2006.

[60] Y. Li and Y.-S. Chou, “A Novel Statistical Methodology for Sub-100 nm MOSFET Fabrication Optimization and Sensitivity Analysis,” in Extend Abstract of 2005 International Conference on Solid State Devices and Materials, 2005, pp. 622-623.

[61] Y. Li and S.-M. Yu, “A Two-Dimensional Quantum Transport Simulation of Nanoscale Double-Gate MOSFETs using Parallel Adaptive Technique,” IEICE-Transactions on Info and Systems, vol. E87-D, pp. 1751-1758, 2004.

[62] Y. Li and H.-M. Chou, “A Comparative Study of Electrical Characteristic on Sub-10 nm Double Gate MOSFETs,” IEEE Transactions on Nanotechnology, vol. 4, no. 5, pp. 645-647, 2005.

[63] T.-W. Tang, X. Lin and Y. Li, “Discretization Scheme for the Density-Gradient Equations and Effect of Boundary Conditions,” Journal of Computational Electronics, vol. 1, no. 3, pp. 389-393, 2002.

[64] C. Lombardi, S. Manzini, A. Saporito and M. Vanzi, “A Physically Based Mobility Model for Numerical, Simulation of Nonplanar Devices,” IEEE Transactions on

Computer-Aided Design of Integrated Circuits and Systems, vol. 7, pp. 1164-1171, 1988.

[65] M. N. Darwish, J. L. Lentz, M. R. Pinto, P. M. Zeitzoff, T. J. Krutsick and H. H. Vuong, "An Improved Electron and Hole Mobility Model for General Purpose Device Simulation," IEEE Transactions on Electron Devices, vol. 44, pp. 1529-1537, 1997.

[66] A. Schenk, "Rigorous theory and simplified model of the band-to-band tunneling in Silicon," Solid-State Electronics, vol. 36, pp. 19-34, 1993.

[67] Y. Li, C.-K. Chen and Y.-Y. Cho, "A Unified Optimization Framework for Microelectronics Industry," in Proceeding of the 8th Annual Conference on Genetic and Evolutionary Computation, 2006, pp. 1875-1876. UOF project [Online]. Available: http://140.113.87.143/ymlab/

[68] Y. Li and S.-M. Yu, "A Unified Optimization Framework for Real World Problems," in Lecture Series on Computer and Computational Sciences, edited by T. Simos et al., Recent Progress in Computational Sciences and Engineering, Brill Academic Publishers, vol. 7, pp. 816-819, 2006.

[69] Y. Li, S.-M. Yu and Y.-L. Li, "Application of A Unified Optimization Framework to Electronic Design Automation," to appear in Mathematics and Computers in Simulation.

[70] C. Hu, “BSIM model for circuit design using advanced technologies,” Digest of Technical Papers 2001 Symposium on VLSI Circuits, pp. 5-10, 2001.

[71] M. Bucher, C. Lallement, C. Enz, F. Theodoloz and F. Krummenacher, The EPFL-EKV MOSFET Model Equations for Simulation, Swiss Federal Institute of Technology (EPFL), 1999.

[72] BSIM 4.2.1 MOSFET Model User’s Manual, U.C. Berkeley, CA, 2001.

[73] P. Bendix, Tech. Proc. Int. Conf. Modeling and Simulation of Microsystems, 649 (2002).

[74] X. Xi, K. Cao, J. He, H. Wan, M. Chan and C. Hu, “Symmetry realization of BSIM model with dynamic reference method for circuit simulation,” The 60th Annual Device Research Conf. Dig., pp. 65-66, 2002.

[75] J. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, Mich., 1975

[76] D.E. Goldberg, Genetic Algorithm Search, Optimization and Machine Learning, Reading, MA: Addison-Wesley, 1989.

[77] C.-A. Hung and S.-F. Lin, “Adaptive Hamming Net: A Fast-Learning ART 1 Model Without Searching,” Neural Networks, vol. 8, no. 4, pp. 605-618, 1995.

[78] H.Niederreiter, Random Number Generation and Quasi-Monte Carlo Methods Philadelphia, PA: SIAM, 1992.

[79] R. S. Maddocks, S. Matthews, E. W.Walker, and C. H.Vincent, “A compact and accurate generator for truly random binary digits,” J. Physics E, vol. 5, no. 5, pp. 542-544, 1972.

[80] T. Stojanovski and L. Kocarev, “Chaos-based random number generators-part I: analysis,” IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, vol. 48, pp. 281-288, 2001.

[81] S. C. Phatak and S. Suresh Rao, “Logistic map: A possible random-number generator,” Phys. Rev. E, vol. 51, pp. 3670-3678, 1995.

[82] R. Caponetto, L. Fortuna, S. Fazzino and M.G. Xibilia, “Chaotic sequences to improve the performance of evolutionary algorithms,” IEEE Transactions on Evolutionary Computation, vol. 7, pp. 289-304, 2003.

[83] T. S. Parker and L. O. Chua, Practical Numerical Algorithms for Chaotic System. Berlin, Germany: Springer-Verlag, 1989.

[84] H. Peitgen, H. Jurgens, and D. Saupe, Chaos and Fractals. Berlin, Germany: Springer-Verlag, 1992.

[85] Agner Fog, “Chaotic Random Number Generators with Random-Cycle Lengths,” December 2000. [Online]. Available: http://www.anger.org/random/theory

[86] S. Liao, Microwave Circuit Analysis and Amplifier Design, Prentice-Hall, pp. 123-160, 1988.

[87] G. Gonzalez, Microwave Transistor Amplifiers, Prentice-Hall, pp. 139-193, 1984.

[88] N. Dye,H. Granberg, Radio Frequency Transistors Principles and Practical Applications, Butterworth- Heinemann, pp.204-231, 1993.

[89] Y. Li, C.-S. Lu, W.-W. Lo, M.-J. Tsai, and T.-Y. Wu, “Sensitivity Analysis of Static Noise Margin in SRAM Cells with 65 nm CMOS devices,” Proc. Conf. Scientific Computing in Electrical Engineering, pp. 21-22, 2006.

[90] Y. Li, S.-M. Yu, and J.-W. Lee, “Quantum Mechanical Corrected Simulation Program with Integrated Circuit Emphasis Model for Simulation of Ultrathin Oxide Metal-Oxide-Semiconductor Field Effect Transistor Gate Tunneling Current,” Jpn. J. App. Phy., vol. 44, pp. 2132-2136, 2005.

[91] W.-W. Lo: Masters Thesis, National Chiao Tung University, Hsinchu, Taiwan, 2006.

[92] A. N. Lokanathan and J. B. Brockman, “A methodology for concurrent processcircuit optimization,” IEEE. Trans. CAD., vol. 18, pp. 889-902, 1999.

[93] S.Williams, K. Varahramyan, andW. Maszara, “Statistical Optimization and Manufacturing Sensitivity Analysis of 0.18 μm SOI MOSFETs, Microelectronic Engineering,” vol. 49, pp. 245-261, 1999.

[94] A. V. Nenarokomov and T. D. Fadale, “Uncertainties in parameter estimation: the optimal experiment design,” Int. J. Heat and Mass Transfer, vol. 43, pp. 3331-3339, 2000.

[95] M. S. Phadke, Quality Engineering Using Robust Design, AT&T Bell Laboratories, 1989.

[96] Engineering Statistics Handbook [Online]. Available: http://www.itl.nist.gov/div898/handbook/

[97] Y.-S. Chou: Masters Thesis, National Chiao Tung University, Hsinchu, Taiwan, 2005.

[98] C. F. J. Wu and M. Hamada, Experiments: Planning, Analysis, and Parameter Design Optimization, New York: John Wiley & Sons, Inc., 2000.

[99] R. H. Myers and D. C. Montgomery, Response Surface Methodology: Process and Product Optimization Using Designed Experiments, New York: John Wiley & Sons, Inc., 2002.

[100] C. Daniel, “Use of Half-normal Plots in Interpreting Factorial Two-level Experiments,” Technometrics, vol. 1, pp. 311-341, 1959.

[101] G. E. P. Box and Norman R. Draper, Empirical Model-Building and Response Surfaces , New York, Wiley, 1987.

[102] J. H. Dodgson, “A Graphical Method for Assessing Mean Squares in Saturated Fractional Designs,” Journal of quality technology, vol. 35, pp. 206-212, 2003.

[103] D. C. Montgomery, Design and Analysis of Experiments, New York: John Wiley & Sons, Inc., 1997.

[104] Carroll, J.and Kai Chang, Statistical computer-aided design for microwave circuits, IEEE Trans. Microwave Theory and Techniques, vol. 44, pp. 24-32, 1996.

[105] J. S. Ramberg, S. M. Sanchez, P. J. Sanchez, and L. J. Hollick, “Designing simulation experiments: Taguchi methods and response surface metamodels,” in Proc. Winter Simulation Conference, pp. 167-176, 1991.

[106] H. Lee and P.K.T. Mok, “A CMOS current-mirror amplifier with compact slew rate enhancementcircuit for large capacitive load applications,” Proc. IEEE Int. Symp. Circuits and Systems, pp. 220-223, 2001.

[107] Gordon W. Roberts and Adel S. Sedra, Spice for Microelectronic circuits, Saunders College Pub., 1992.

[108] A. Boag, A. Boag, E. Michielssen, and R. Mittra, “Design of electrically loaded wire antenna using genetic algorithm,” IEEE Transactions on Antennas and Propagation, vol. 44, pp. 687-695, 1996.

[109] N. Cohen, “Fractal coding in genetic algorithm (GA) antenna optimization,” IEEE Antennas and Propagation Society International Symposium, vol. 3, pp. 1692-1695, 1997.

[110] W. Comisky, J. R. Koza, and J. Yu, “Automatic synthesis of a wire antenna using genetic programming,” Late Breaking Papers at the 2000 Genetic and Evolutionary Computation Conference, Las Vegas, Nevada. pp. 179-186.

[111] F. Castellana, F. Bilotti, and L. Vegni, “Automated dual band patch antenna design by a genetic algorithm based numerical code,” IEEE Antennas and Propagation Society International Symposium, vol. 4, pp. 696-699, 2001.

[112] J. P. Gianvittorio and Y. Rahmat-Smaii, “Fractal antennas: A novel antenna miniaturization technique, and applications,” IEEE Antennas and Propagation Magazine, vol. 44, pp. 20-36, 2002.

[113] D. H. Werner and S. Ganguly, “An overview of fractal antenna engineering research,” IEEE Antennas and Propagation Magazine, vol. 45, pp. 38-57, 2003.

[114] F. J. Villegas, T. Cwik, Y. Rahmat-Samii, and M. Manteghi, “A parallel electromagnetic genetic-algorithm optimization (EGO) application for patch antenna design,” IEEE Transaction on Antennas and Propagation, vol. 52, pp. 2424-2435, 2004.

[115] J. D. Lohn, D. S. Linden, G. S. Hornby, A. Rodriguez-Arroyo, S. E. Seufert, B. Blevins, and T. Greenling, “Evolutionary design of a single-wire circularly-polarized X-band antenna for NASA’s space technology 5 mission,” IEEE Antennas and Propagation Society International Symposium, vol. 2B, pp. 267-270, 2005.

[116] N. Jin and Y. Rahmet-Samii, “Parallel particle swarm optimization and finite difference time-domain (PSO/FDTD) algorithm for multiband and wide-band patch antenna designs,” IEEE Transactions on Antennas and Propagation, vol. 53, pp. 3459-3468, 2005.

[117] H. T. Chou, Y. C. Hou, and W. J. Liao, “A dual band patch antenna design for WLAN and DSRC applications based on a genetic algorithm optimization,” Electromagnetics, vol. 27, pp. 253-262, 2007.

[118] J. R. Koza, S. H. Al-Sakran, L. W. Jones, G. Manassero, “Automated synthesis of a

fixed-length loaded symmetric dipole antenna whose gain exceeds that of a commercial antenna and matches the theoretical maximum," Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, pp. 2074-2081, 2007.

[119] E. E. Altshuler and D. S. Linden, "Wire-antenna designs using genetic algorithms," IEEE Antennas and Propagation Magazine, vol. 39, no. 2, Apr. 1997.

[120] P. Bhartia, I. Bahl, R. Garq, and A. Ittipiboon, Microstrip antenna design handbook, Norwood, MA: Artech House, 2000.

[121] R. Salomon, "Evolutionary algorithms and gradient search: similarities and differences," IEEE Transactions on Evolutionary Computation, vol. 2, pp. 45-55, 1998.

[122] S.-M. Yu and Y. Li, "A computational intelligent optical proximity correction for process distortion compensation of layout mask in subwavelength era," Abstracts of the 10th International Workshop on Computational Electronics, pp. 179-180, 2004.

[123] J. Jin, The Finite Element Method in Electromagnetics, 2nd ed., John Wiley & Sons: NY, 2002.

[124] B. Engquist and A. Majda, "Absorbing boundary conditions for the numerical simulation of waves," Proceedings of the National Academy of Sciences, vol. 74, no. 5, May, 1977.

[125] J. M. Jin and N. Lu, “Application of adaptive absorbing boundary condition to finite element solution of three-dimensional scattering,” IEE Proceedings-Microwaves, Antennas and Propagation, vol. 143, pp. 57-61, Feb. 1996.

[126] J. M. Jin, J. L. Volakis, and V. V. Liepa, “An engineer’s approach for terminating finite element meshes in scattering analysis,” IEEE Antennas and Propagation Society International Symposium, vol. 2, pp. 1216-1219, June 1991.

[127] J. P. Berenger, “A perfectly matched layer for the absorption of electromagnetic waves,” Journal of computational physics, vol. 114, pp. 185-200, 1994.

[128] J. P. Berenger, “Three-dimensional perfectly matched layer for the absorption of electromagnetic waves,” Journal of computational physics, vol. 127, pp. 363-379, 1996.

[129] W. L. Stutzman, Antenna Theory and Design, NY: John Wiley & Sons, 1998.

[130] http://www.ansoft.com/products/hf/hfss/

[131] E. Alm, A. M. AV, T. Kortemme, D. Baker, Simple physical models connect theory and experiment in protein folding kinetics, J. Mol. Biol., 322 (2002) 463-476.

[132] S. Altmeyer, R. Fuchslin, J. McCaskill, Folding stabilizes the evolution of catalysts, Artificial Life, 10 (2004) 23-38.

[133] S. W. Englander, Protein folding intermediates and pathways studied by hydrogen exchange, Annual Rev. Biophy. and Biomol. Struct., 29 (2000) 213-238.

[134] D. S. Watkins, Qr-like algorithms for eigenvalue problems, J. Comput. Appl. and Math., 123 (2000) 67-83.

[135] A. Stathopoulos, S. Yousef, W. Kesheng, Dynamic thick restarting of the davidson, and the implicity restarted arnoldi methods, SIAM J. Sci. Comput., 19 (1998) 227-245.

[136] R. B. Lehoucq, D. Sorensen, Deflation techniques for an implicitly re-started arnoldi iteration, SIAM J. Matr. Anal., 17 (1996) 789-821.

[137] D. C. Sorensen, Implicitly-restarted arnoldi/lanczos methods for large scale eigenvalue calculations, in: Parall. Numer. Alg.: Proc. ICASE/LaRC Workshop, Edited by D. E. Keyes, A. Sameh, and V. Venkatakrishnan, Kluwer, 1995.

[138] D. R. Fokkema, G. L. G. Sleijpen, H. A. V. Vorst, Jacobi-davidson style qr and qz algorithms for the reduction of matrix pencils, SIAM J. Sci. Comput., 20 (1998) 94.

[139] Visi - a VTK- and QT-Based Open-Source project for data visualization. Available: http://140.113.87.143/ymlab/visi

[140] A. Buja, D. Cook, and D. F. Swayne, "Interactive high-dimensional data visualization," Journal of Computational and Graphical Statistics, vol. 5, pp. 78-99, 1996.

[141] B. Field, S. O'Neill, V. Interrante, and T. W. Jones, "FieldVis: A Tool for Visualizing Astrophysical Magnetohydrodynamic Flow," IEEE Computer Graphics and Applications, vol. 27, no. 1, pp. 9-13, 2007.

[142] Qt sets the standard for high-performance, cross-platform application development. Available: http://www.trolltech.com/products/qt

[143] Document for VTK file format. Available: http://public.kitware.com/VTK/pdf/file-formats.pdf

[144] F. Stern, "Iteration methods for calculating self-consistent fields in semiconductor inversion layers," Journal of Computational Physics, vol. 6, pp. 56-67, 1970.

[145] J. A. Lopez-Villanueva, I. Melchor, F. Gamiz, J. Banqueri, J. A. Jimenez-Tejada, "A model for the quantized accumulation layer in metal-insulator-semiconductor structures," Solid-State Electronics, vol. 38, pp. 203-210, 1995.

VITA

Name: Shao-Ming Yu, 余紹銘

Permanent address: 4F., No. 210, Ren-ai St., Sanchong City,
Taipei County Taiwan

Degree and date to be conferred: Doctor of Philosophy of Computer Science, October, 2007

Date of birth: December 8st, 1978

Place of birth: Hsinchu, Taiwan

Collegiate institutions attended	Degree	Date of graduate
Department of Computer and Information Science National Chaio Tung University, Hsinchu, Taiwan	BS	June, 2002
Department of Computer and Information Science National Chaio Tung University, Hsinchu, Taiwan	MS	July, 2004
Department of Computer Science National Chaio Tung University, Hsinchu, Taiwan	Ph. D	October, 2007

Ph. D dissertation title: A Unified Optimization Framework for Electronic Design Automation Application

Publication Lists:

[1] Yiming Li, **Shao-Ming Yu** and Yih-Lang Li, “Parallel Solution of Large-Scale Eigenvalue Problem for Master Equation in Protein Folding Dynamics,” submitted for publication in *Journal of Parallel and Distributed Computing*. (to appear)

[2] Yiming Li and **Shao-Ming Yu**, “A Coupled-Simulation-and-Optimization Approach to Nanodevice Fabrication with Minimization of Electrical Characteristics Fluctuation,” *IEEE Trans. on Semiconductor Manufacturing*. (to appear)

[3] Yiming Li, **Shao-Ming Yu** and Yih-Lang Li, “Application of A Unified Optimization Framework to Electronic Design Automation,” *Mathematics and Computers in Simulation*. (to appear)

[4] Yiming Li, Yih-Lang Li and **Shao-Ming Yu**, “Design Optimization of a Current Mirror Amplifier Integrated Circuit Using a Computational Statistics Technique,” *Mathematics and Computers in Simulation*. (to appear)

[5] Yiming Li, **Shao-Ming Yu** and Hung-Ming Chen, “Process-Variation- and Random-Dopants-Induced Threshold Voltage Fluctuations in Nanoscale CMOS and SOI Devices,” *Microelectronic Engineering*, Vol. 84, No. 9-10, Sep.-Oct. 2007, pp. 2117-2120.

[6] Shih-Ching Lo, Yiming Li, and **Shao-Ming Yu**, “Analytical Solution of Nonlinear Poisson Equation for Symmetric Double-Gate Metal-Oxide-Semiconductor Field Effect Transistors,” *Mathematical and Computer Modelling*, Vol. 46, No. 1-2, July 2007, pp. 180-188.

[7] **Shao-Ming Yu**, Jam-Wem Lee, and Yiming Li, “An Optimal Silicidation Technique for Electrostatic Discharge Protection Sub-100 nm CMOS Devices in VLSI Circuit,” *Microelectronic Engineering*, Vol. 84, No. 2, Feb. 2007, pp. 213-217.

[8] Yiming Li and **Shao-Ming Yu**, “Comparison of Random Dopant-Induced Threshold Voltage Fluctuations in Nanoscale Single-, Double-, and Surrounding-Gate Field Effect Transistors,” *Japanese Journal of Applied Physics*, Vol. 45, No. 9A, 2006, pp. 6860-6865.

[9] Yiming Li and **Shao-Ming Yu**, “A Numerical Iterative Method for Solving Schrödinger and Poisson Equations in Nanoscale Single, Double and Surrounding Gate Metal-Oxide-Semiconductor Structures,” *Computer Physics Communications*. Vol. 169, No. 1-3, July 2005, pp. 309-312.

[10] Yiming Li, **Shao-Ming Yu**, and Jam-Wem Lee, “Quantum Mechanical Corrected Simulation Program with Integrated Circuit Emphasis Model for Simulation of Ultrathin Oxide Metal-Oxide-Semiconductor Field Effect Transistor Gate Tunneling Current,” *Japanese Journal of Applied Physics*, Vol. 44, No. 4B, April 2005, pp. 2132-2136.

[11] Yiming Li and **Shao-Ming Yu**, “A Parallel Adaptive Finite Volume Method for Nanoscale Double-gate MOSFETs Simulation,” *Journal of Computational and Applied Mathematics*. Vol. 175, No. 1, March 2005, pp. 87-99.

[12] Yiming Li, **Shao-Ming Yu**, Chih-Hong Hwang, and Yih-Lang Li, “Temperature Dependence on the Contact Size of GeSbTe Films for Phase Change Memories,” accepted by the 12th IEEE International Workshop on Computational Electronics (IEEE IWCE-12), University of Massachusetts Amherst, Massachusetts, U.S.A, Oct. 8-10, 2007.

[13] Yiming Li, **Shao-Ming Yu**, Yi-Ting Kuo and Yih-Lang Li, Simulation-Based Evolutionary Method in Antenna Design Optimization for WLAN and Wireless Communication Applications,” accepted by International Conference of Computational Methods in Sciences and Engineering (ICCMSE 2007), Corfu, Greece, Sep. 25-30, 2007.

[14] Yiming Li, **Shao-Ming Yu** and Yih-Lang Li, “A Simulation-Based Hybrid Optimization Technique for Analog and Digital Integrated Circuits Design Automation,” International Conference on Modeling and Simulation of Microsystems (MSM 2007), Silicon Valley, California, U.S.A, May 20-24, 2007.

[15] Yiming Li, **Shao-Ming Yu** and Yih-Lang Li, “A Simulation-Based Hybrid Optimization Technique for Low Noise Amplifier Design Automation,” Computational Science - ICCS 2007, Edited by Y. Shi et al., *Lecture Notes in Computer Science* , Springer Berlin / Heidelberg, Vol. 4490, 2007, (ISBN: 978-3-540-72589-3).

[16] Yiming Li and **Shao-Ming Yu**, “A Unified Optimization Framework for Real World Problems,” Lecture Series on Computer and Computational Sciences, edited by T. Simos et al., *Recent Progress in Computational Sciences and Engineering*, Brill Academic Publishers, vol. 7, pp. 816-819, 2006, (ISBN-10: 90 04 15542 2, ISBN-13: 978 90 04 15542 2).