Wenjian Yu
Xiren Wang

Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits

TSINGHUA UNIVERSITY PRESS
Springer

Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits

Wenjian Yu • Xiren Wang

Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits

TSINGHUA UNIVERSITY PRESS

Springer

Wenjian Yu
Department of Computer Science
and Technology
Tsinghua University
Beijing, China

Xiren Wang
Cadence Design Systems
San Jose, CA, USA

ISBN 978-3-642-54297-8 ISBN 978-3-642-54298-5 (eBook)
DOI 10.1007/978-3-642-54298-5
Springer Heidelberg New York Dordrecht London

Jointly published with Tsinghua University Press, Beijing
ISBN: 978-7-302-35151-1 Tsinghua University Press, Beijing

Library of Congress Control Number: 2014937680

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Foreword

This book is about three-dimensional RC extraction techniques for microelectronic designs. The authors are from the Parasitic Parameter Extraction Group, Department of Computer Science and Technology, Tsinghua University, which has made many significant contributions to the field since 1991. Some of their approaches, e.g., direct boundary element method (BEM) for C extraction (2001–2006) and hierarchical block BEM (2004), have been incorporated in industrial tools.

Resistance and capacitance (RC) extraction is an essential step to model the interconnect wires and substrate coupling in integrated circuits. The parasitic plays a significant role in the system performance. Advances of fabrication processes and new materials with various dielectric permittivities call for accurate and efficient extraction tools to handle complex geometries. Although RC extraction has been a research topic in the electronic design automation community for about 25 years, larger designs and faster project turnaround have kept pushing the demand for better extraction tools.

The authors cover the state-of-the-art techniques of RC field solvers, mainly the boundary element method (direct or indirect) with accelerating techniques and the fast floating random walk methods. These subjects are relatively new and of large impact theoretically and practically. The content also reflects the research activities of the authors in the last 10 years.

This book presents a systematic introduction to and treatment of the key concepts of the extraction. To the best of my knowledge, it is the first time for a monograph dedicated to the advanced RC extraction techniques. Various field-solver techniques are explained in details, with examples to illustrate the advantages and disadvantages of each algorithm. Readers are encouraged to consider the computational complexity, physical theory, numerical stability, robustness of the algorithm for general cases, and applicability for software development and maintenance. The presentation brings insights of suitable solvers for specific extraction problems.

San Diego, CA, USA — Chung-Kuan Cheng

Preface

The main goal of writing this book was to present a methodological and algorithmic perspective on the field-solver-based parasitic extraction of integrated circuits (ICs). Specifically, we present advanced techniques based on three-dimensional (3-D) boundary element method and floating random walk method for the problems of resistance and capacitance (RC) calculation. With the feature size scaling down and mixed-signal interference in modern ICs, the research of parasitic extraction has gained much concern in recent years and promoted the utilization of field-solver methods for tackling the challenge of accuracy.

Now, the field solver which directly solves the electrostatic equations is becoming more and more important for the RC extraction of ICs. It is a necessary supplement, or even a replacer, of the existing parasitic extraction methodology. On accurately capturing the complex interconnect geometry and the substrate coupling in mixed-signal IC, the field-solver method has distinct advantages. The major obstacle for its application is the excessive computational expense. The complexity of interconnect structure and even tighter performance margin for designing nanometer-technology ICs have urged the extensive usage of field solvers. The random process variations also add significance to this request. All these have pushed the related research for 20 years. Various accelerating approaches have been proposed to reduce the computational expense while preserving accuracy. Until recently, the achievements of these works have been applied successfully in industrial tools. They are daily used for settling the sign-off timing and verification issues in various IC designs. These achievements in field-solver-based RC extraction are the object of this book. We hope we have succeeded in providing a unique and comprehensive treatment on them.

The works presented in this book are mostly from research projects undertaken by the Parasitic Extraction Group, Tsinghua University, China. Chapter 3 and Chaps. 5, 6, and 7 are contributed by Xiren Wang, mostly from his Ph.D. work at the Parasitic Extraction Group. The remaining chapters are written by Wenjian Yu, based on his research work. Many of those original publications can be found at http://learn.tsinghua.edu.cn:8080/2003990088/index.htm.

We want to emphasize that the book is by no means intended to be comprehensive. The absence of coverage of related works should by no means diminish their value and contribution. Many academic groups and experts from industry have made significant contributions in the field, and the reader is encouraged to investigate their works. Key contributors to progress in RC field-solver techniques include: Jacob White (MIT), Weiping Shi (TAMU), Lawrence T. Pileggi (CMU), Ali Niknejad (UC Berkeley), Dan Jiao (Purdue Univ.), Vikram Jandhyala (Univ. Washington), Luca Daniel (MIT), Yannick L. Le Coz (RPI), Sheldon X.-D. Tan (UC Riverside), Lei He (UCLA), Ranjit Ghapurey (UT Austin), Charlie Chung-Ping Chen (National Taiwan Univ.), Wayne Dai (UCSC), Nick van der Meijs (TU Delft), Luis Miguel Silveira (Technical University of Lisbon), Ibrahim Elfadel (Masdar Institute of Science and Technology), Nasser Masoumi (Univ. Waterloo), Madhav P. Desai (Indian Institute of Technology, Bombay), Angelo Brambilla (Politecnico di Milano), Alkiviades A. Hatzopoulos (Aristotle Univ. Thessaloniki), Ruben Specogna (Univ. Udine), Xuan Zeng (Fudan Univ.), Zeyi Wang (Tsinghua Univ.), Wei Hong (Southeast Univ.), Junfa Mao (Shanghai Jiaotong Univ.), Martin Bachtold (Swiss Federal Institute of Technology), Sharad Kapur (Integrand Software Inc.), Joel R. Phillips (Cadence Inc.), Xiaoning Qi (Intel), and Zhuoxiang Ren (Mentor Graphics Inc.).

Beijing, China — Wenjian Yu
San Jose, CA, USA — Xiren Wang

Acknowledgment

The contents of this book mainly come from the research works done in the Parasitic Extraction Group at the Department of Computer Science and Technology, Tsinghua University.

It is a pleasure to record our gratitude to many colleagues and students who have contributed to this book. First of all, the authors would like to thank Prof. Zeyi Wang of Tsinghua University. He was the Ph.D. thesis advisor of both authors and had been the leader of the Parasitic Extraction Group until 2006. The quasi-multiple medium approach for accelerating the boundary element method (BEM) originates from his idea, and several works on fast BEM approaches were also under his guidance. Many students in this group have excellent contributions to the works in this book, including Deyan Liu, Mengsheng Zhang, Lei Zhang, Wangyang Zhang, Chao Hu, Qingqing Zhang, Hao Zhuang, Zhi Liu, Gang Hu, and Chao Zhang. Special thanks are also given to Prof. Xianlong Hong of Tsinghua University for his great support during these years and Dr. Jiangchun Gu and Dr. Jinsong Hou graduated from the group, who had provided a solid basis for our research works.

Wenjian Yu is grateful to Prof. Zhiping Yu and Prof. Zuochang Ye at the Institute of Microelectronics of Tsinghua University and Prof. Zuying Luo of Beijing Normal University for much helpful technical discussion. Thanks are also given to Dr. Jinjun Xiong of IBM Watson Research Center and Dr. Rong Jiang of Cadence Inc. for their insights from industry, which inspired the work on variation-aware extraction in this book. Wenjian Yu would also like to thank Prof. Chung-Kuan Cheng at the University of California at San Diego and Prof. Sheldon X.-D. Tan at the University of California at Riverside for their help and encouragement of writing such a book.

Xiren Wang is appreciative to the members in the Parasitic Extraction Group of Tsinghua University, where he had stayed for 5 years and got the Ph.D. degree. He is also grateful to Prof. Vikram Jandhyala at the University of Washington, who had been his postdoc advisor. Discussion with Dr. Dipanjan Gope at the University of Washington, and now Indian Institute of Science, was largely beneficial and really appreciated.

The authors would like to thank the National Natural Science Foundation of China and the Beijing Natural Science Foundation for their financial support for

this book. The works presented in this book have been funded in part by the National Natural Science Foundation of China (NSFC) grants under No. 90407004, No. 60401010, and No. 61076034 and the Beijing Natural Science Foundation grant under No. 4132047. Wenjian Yu would like also to acknowledge the funding support from the Tsinghua University Initiative Scientific Research Program.

Last but not least, Wenjian Yu and Xiren Wang would like to thank their families, respectively, for understanding and support during the many hours it took to compose this book.

Contents

1 Introduction 1
- 1.1 The Need for Parasitic Extraction 1
- 1.2 The Methods for RC Extraction and Field Solver 2
- 1.3 Book Outline 4
- 1.4 Summary 6

2 Basic Field-Solver Techniques for RC Extraction 7
- 2.1 Problem Formulation 7
- 2.2 Overview of the Numerical Methods 10
- 2.3 Indirect Boundary Element Method 11
- 2.4 Direct Boundary Element Method 14
- 2.5 Floating Random Walk Method 16
- 2.6 Summary 18

3 Fast Boundary Element Methods for Capacitance Extraction (I) 19
- 3.1 Basics of Indirect Boundary Element Methods 19
- 3.2 Fast Multipole Methods 20
  - 3.2.1 Introduction 20
  - 3.2.2 Multipole Expansions 22
  - 3.2.3 Local Expansions 23
  - 3.2.4 Fast Multipole Algorithm 24
- 3.3 Low-Rank Matrix Compression-Based Fast Iterative Solvers 26
  - 3.3.1 Why Compression? 26
  - 3.3.2 Matrix Compression Can Reduce the Complexity to Linear 27
  - 3.3.3 Compression Possible? 28
  - 3.3.4 Basics of Matrix Compression Using SVD and QR 30
  - 3.3.5 Compression Without Building Entire Matrix Beforehand 32

3.4 Matrix Compression by Adaptive Cross Approximation 34
3.4.1 Adaptive Cross Approximation (ACA) 35
3.4.2 Recompression of Adaptive Cross Approximation 36
3.5 Summary 37

4 Fast Boundary Element Methods for Capacitance Extraction (II) 39
4.1 Direct Boundary Element Method for Multi-dielectric Capacitance Extraction 40
4.2 The Quasi-multiple Medium Approach 43
4.2.1 Basic Idea 43
4.2.2 Decomposition of Dielectrics and Boundary Element Partition 45
4.2.3 Algorithm Description and Analysis 47
4.3 Equation Organization and Solving Techniques 49
4.3.1 Organization of the Coefficient Matrix 49
4.3.2 Extended Jacobi and MN Preconditioners 51
4.4 Numerical Results 54
4.4.1 The Comparison with GIMEI 54
4.4.2 The Comparison with ODDM 55
4.4.3 The Results for Structures from Real Design 57
4.4.4 The Comparison with FastCap 59
4.5 Efficient Techniques for Handling Floating Metal Fills 61
4.5.1 Basic Idea 64
4.5.2 Equation Formation and Solution 65
4.5.3 Numerical Results 66
4.6 Summary 70

5 Resistance Extraction of Complex 3-D Interconnects 71
5.1 Analytical Resistance Formulation 72
5.2 Field Solver for Interconnect Resistance 72
5.2.1 Resistance Network of Multiterminal Regions 73
5.2.2 Resistance Calculation Using Direct BEM 74
5.3 Fast BEM Solver Using Linear Boundary Elements 74
5.3.1 Physics-Based Nonuniform Virtual Cutting 75
5.3.2 Discarding Conductors Not in the Path of Direct Current 80
5.3.3 Dividing Elements Only in One Direction When Possible 80
5.3.4 Linear Boundary Elements for Straight Conductors 81
5.3.5 Efficiency Summary 82
5.4 Analytical QBEM Extraction 83
5.4.1 General Analytical QBEM Algorithm 83
5.4.2 Distinguish Between Regular and Irregular Subregions 84

5.4.3 Compute the Resistance Network of the Whole Region 84
5.4.4 Numerical Result and Analysis 85
5.5 Summary 86
Appendix 5.A 86

6 Substrate Resistance Extraction with Boundary Element Method ... 91
6.1 Field Solver for Substrate Resistance 92
6.2 Efficient Field-Solver Techniques 95
6.2.1 Nonuniform Meshing 95
6.2.2 Numerical Reduction of Linear Equation System 96
6.2.3 Quasi-multiple Medium Technique to Sparsify Matrix 99
6.3 Numerical Experiments 100
6.3.1 Simple One-Layer Substrate 100
6.3.2 The 52-Contact Structure with Three Doping Profiles 102
6.3.3 Test Structure with Lateral Resistivity Variation 104
6.4 Summary 106

7 Extracting Frequency-Dependent Substrate Parasitics 107
7.1 Field Solver for Substrate Capacitance and Resistance 108
7.2 Direct Boundary Element Method for Substrate Impedance Extraction 109
7.3 The Two-Step Approach 110
7.3.1 Frequency-Dependent Entries in Matrix A 111
7.3.2 Perturbed Equation System and Its Efficient Solution 112
7.4 Efficient Technique for Solving the Real-Valued System 114
7.5 Overall Algorithm Flow and Discussion 115
7.6 Numerical Results 116
7.6.1 Substrate with 52 Contacts 116
7.6.2 More Numerical Experiments 118
7.7 Summary 119

8 Process Variation-Aware Capacitance Extraction 121
8.1 Motivation 121
8.2 The Incremental BEM for Variation-Aware Capacitance Library Building 124
8.2.1 Basic Idea 125
8.2.2 Modification of the Coefficient Matrix and the Solving Technique 126
8.2.3 Numerical Results 127
8.3 Preliminaries of Variation-Aware Statistical Capacitance Extraction 128
8.3.1 Grid-Based Process Variation Model 128
8.3.2 The Hermite Polynomial Collocation Method 130

8.4 Chip-Level Statistical Capacitance Extraction Considering Spatial Correlation 133
8.4.1 Intra-window Capacitance Extraction with the Grid-Based Variation Model 134
8.4.2 Calculation of Inter-window Capacitance Covariance 137
8.4.3 Complexity Analysis of the Inter-window Calculation ... 139
8.4.4 Statistical Model of Full-Path Capacitance 142
8.5 Experiments of Statistical Capacitance Extraction 144
8.5.1 Simple Cases with Parallel-Line Structure 145
8.5.2 A Large Case with Multilayered Structure 147
8.6 Summary 148
Appendix 8.A. Complete Proof of Theorem 8.3 148

9 Statistical Capacitance Extraction Based on Continuous-Surface Geometric Model 153
9.1 The Continuous-Surface Model for Geometric Variation 154
9.1.1 Three Geometric Variation Models 154
9.1.2 The Reasonable CSV Model for On-Chip Interconnect 156
9.1.3 The Comparison of Three Geometric Variation Models 158
9.2 Efficient Statistical Extraction Techniques 161
9.2.1 The Weighted PFA for Variable Reduction 162
9.2.2 Parallel Statistical Capacitance Extraction 163
9.2.3 Calculating the Inter-window Covariance of Capacitance 165
9.3 Fast Approaches to Model the Line-Edge Roughness 167
9.3.1 Background 167
9.3.2 The Adjoint Field Technique for Sensitivity Calculation 169
9.3.3 Two Efficient Approaches 170
9.3.4 Numerical Results 173
9.3.5 More Analysis Results and Discussion 176
9.4 Summary 177

10 Fast Floating Random Walk Method for Capacitance Extraction 179
10.1 The Basic Floating Random Walk Algorithms 180
10.1.1 Numerical Technique to Calculate Multi-dielectric Surface Green's Function 183
10.2 A Multi-dielectric FRW Algorithm with the Precharacterized Probabilities and Weight Values 184
10.2.1 The Basic Idea 184
10.2.2 The Details of the Precharacterization Procedure 185
10.2.3 The FRW Algorithm with Multi-dielectric GFTs and WVTs 189

10.3 The Techniques for Variance Reduction 190
10.3.1 Background 190
10.3.2 The Importance Sampling with the Weight Values Averaged 192
10.3.3 The Comprehensive Variance Reduction Scheme 195
10.4 The Space Management Technique and Parallel Implementation 198
10.4.1 The Space Management Technique 198
10.4.2 The Parallel Implementation 200
10.5 Numerical Results 201
10.5.1 Test Cases 201
10.5.2 Validating the Multi-dielectric FRW Algorithm 202
10.5.3 Validating the Variance Reduction Techniques 205
10.5.4 Comparing with the Fast Boundary Element Method 205
10.5.5 Validating the Efficiency of Parallel Computing 207
10.6 Summary 208

11 FRW-Based Solver for Chip-Scale Large Structures 209
11.1 Motivation 209
11.2 Basic Operations of Space Management and Accelerating Techniques 210
11.2.1 Basic Operations 211
11.2.2 Improving the Candidate Checking with Distance Limit 212
11.2.3 Incomplete Candidate List 215
11.2.4 Reducing the Time for Inquiring the Candidate List 216
11.3 Space Management Structures and Approaches 218
11.3.1 The Improved Octree-Based Approach 219
11.3.2 Two Grid-Based Approaches 220
11.3.3 The Hybrid Approach Using Grid and Octree 221
11.4 Numerical Results 223
11.4.1 Test Cases 223
11.4.2 Validating the Three Accelerating Techniques 224
11.4.3 Evaluating Different Space Management Approaches 226
11.4.4 RWCap2 with the Hybrid Approach Using Grid and Octree 229
11.4.5 The Results for Multi-dielectric Cases 230
11.5 Summary 231

References 233

Index 243

Chapter 1
Introduction

1.1 The Need for Parasitic Extraction

In very large-scale integration (VLSI) circuits, electromagnetic coupling among interconnect wires is becoming increasingly important. With the introduction of deep sub-micronmeter (DSM) or nanometer semiconductor technologies, the on-chip interconnect wire could no longer be considered as equipotential link. The parasitic effects introduced by the wires display a scaling behavior that differs from that of active devices such as transistors, and these effects tend to gain importance as device dimensions are reduced and circuit speed is increased. In fact, they have dominated some of the relevant metrics of digital integrated circuits, such as speed and reliability. A typical recursive design flowchart of digital integrated circuit (IC) is shown in Fig. 1.1, where a post-layout step termed *parasitic extraction* precedes *gate-level simulation*. Each step in the design flow corresponds to numerous computer-aided design (CAD) tools, which guarantee the feasibility of designing modern VLSI circuits. The task of parasitic extraction is to model the electromagnetic effect of the wires with parasitic components of capacitance (C), resistance (R), and inductance (L), so that the gate-level simulation, including timing analysis, can be performed.

In microwave or analog integrated circuits, the electromagnetic coupling among conductors also greatly influences circuit performance. This coupling effect is sometimes utilized to construct compact circuit components. But under most circumstances, it is regarded as a parasitic effect that must be modeled accurately for the verification of circuit's validity and performance. With the increase of working frequency and advancement of the silicon technologies, the discrepancy between the analog IC and the VLSI digital IC becomes reduced. Therefore, the electromagnetic modeling and accurate extraction of the interconnect parasitics have become a widely concerned research topic. Among the three parasitic parameters, capacitance has attracted the most attention because it greatly influences time delay, power consumption, and signal integrity [3, 32]. Because of the equivalence of electrostatic field and steady-state current field, the method for extracting capacitances can

W. Yu and X. Wang, *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*, DOI 10.1007/978-3-642-54298-5_1,

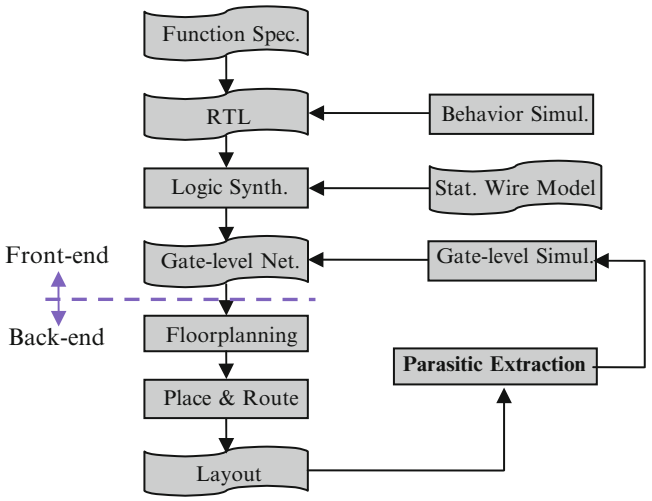

Fig. 1.1 A typical recursive design flow of digital IC (Reprinted with permission from Yu and Wang [180] © 2005 John Wiley & Sons)

be applied to extract the resistances. In most cases, the resistance extraction of interconnect wire is much easier. If the signal frequency is not very high, the inductive effect is not prominent. For example, the parasitic inductance is often ignored in the design of digital IC.

Except for the parasitic effect on interconnect wire, the modeling of substrate coupling in mixed-signal IC also attracts a lot of attention. With the increase of integration, the digital components and analog components in a circuit are often built on a common silicon substrate. This gives rise to the problem called *substrate noise coupling*. The digital components may inject current noise into the substrate, which travels throughout the substrate and severely impacts the sensitive analog components. The substrate coupling affects some circuit metrics, like the quality factor of spiral inductor in microwave circuit. If the signal frequency is not very high, the substrate coupling can be modeled with a resistance network among contacts. With the increase of frequency, more comprehensive model with both capacitance and resistance effects should be adopted. The calculation of the substrate coupling parameters is also regarded as a parasitic extraction problem. Efficient modeling and extraction of substrate coupling are necessary, or even critical, for the design of mixed-signal IC.

1.2 The Methods for RC Extraction and Field Solver

An accurate and general method for calculating the capacitance parameters needs simulating the electrostatic field among conductors, because capacitance is the coefficient relating the electric potential (voltage) and the electric charge. This method

is often referred to as *field solver*. For the problems of resistance extraction and inductance extraction (or impedance extraction), there are different field solvers which simulate corresponding steady current field and electromagnetic field. However, efficient field solver for capacitance calculation is difficult, if feasible, due to the large quantity of and complicated interconnect wires we are facing in ICs. That means huge expense on memory and computing time.

To obtain a good trade-off between accuracy and efficiency, modern capacitance extraction tools utilize special techniques for the capacitance extraction in IC design, which is usually divided into two major steps [79]:

1. Technology precharacterization. Given a description of the process cross sections, tens of thousands of test structures are enumerated and simulated with two-dimensional (2-D) and/or three-dimensional (3-D) field solvers. The resulting data are collected either to fit some empirical formulas or to build lookup tables (either type is called a "pattern library") [3, 32, 33, 55]. This first step should be performed only once per process technology. The challenge in this step includes the handling of increasingly complex technology features, such as low-k dielectric, air bubble, nonvertical conductor cross sections, conformal dielectric, and shallow trench isolations.
2. Capacitance extraction with pattern-matching approach. While performing full-chip or full-path (specified critical signal path) extraction, the whole structure is chopped into small- or medium-sized pieces firstly. Each piece is regarded as a pattern structure with geometric parameters. Then, the geometric parameters are matched to some entries in the precharacterized pattern library. Finally, the capacitance values of the piece can be obtained through table lookup or analytical formulas. With more kinds of patterns and more geometric parameters per pattern, the computational expense of technology precharacterization and the complexity of the pattern-matching procedure will largely increase. On the other hand, if there are few patterns and each is described by very few parameters, it becomes difficult to be accurate. So, the definition of geometric patterns is crucial and relates both steps in modern parasitic extraction tool. Some techniques can be used to reduce the complexity of geometric patterns by considering the shielding effect in electrostatic field. For example, it is often assumed that the conductors two layers away from the main conductor of interest can be described as a big plane [33, 55].

There are two major sources of error in the pattern-matching approach. The first one is called the *pattern mismatch*, where extracted geometry parameters do not have an exact match in the pattern library. The other one is due to the layout decomposition, which is analyzed in Dengi and Rohrer [37] and can be revamped by the approach proposed by Shi and Yu [134]. However, with the increase of geometric and material complexity in the advanced process technology, the error of pattern-matching-based capacitance extraction will be even larger.

For the resistance of interconnect wire, the simple method of "square-counting" is fast and accurate in most time. However, to model some complex 3-D structure, the field-solver method should be used to simulate the steady current field in the

body of conductor wires. This scenario also occurs in the problem of modeling substrate coupling noise. It is almost impossible to apply the pattern-matching approach in the extraction of substrate parasitics. So, the field-solver method is the major choice.

With the field-solver method, we model the 3-D geometry accurately and thus are able to achieve the highest precision. So, the field solver provides the accuracy criterion to other capacitance extraction methods. Actually, the field solver is indispensable in the technology precharacterization step and will play more important role in the capacitance extraction of nanometer-technology IC design.

As VLSI technology scales into the nanometer regime, one profound change in the chip design is that engineers cannot put the design precisely into the silicon chips. This is because the manufacture process variations start to play a big role, whose influence on the circuit performance, yield, and reliability becomes significant [131, 184]. Specifically, the geometric and dielectric variations in interconnect structure largely affect the parasitic capacitance [175] and therefore the timing delay and other signal integrity metrics. Although traditional corner-based analysis and design approaches apply guard bands to consider parameter variations, it leads to too conservative designs, which may largely degrade the design quality [131]. As a result, it is imperative to develop new statistical techniques to consider the impacts of various process uncertainties on the parasitic extraction [186].

Another challenge is brought by the multigate devices, such as FinFETs and Tri-gate devices, in the coming sub-10-nm-technology nodes, which are the necessity for continuing the CMOS technology. To accurately capture the parasitic capacitances related with the multigate devices, which is essential for performance verification, a 3-D-technology CAD (3-D-TCAD) transport analysis-based approach should be added to the capacitance field solver [17, 18]. This largely complicates the problem. Although preliminary results have been achieved, further research along this direction is definitely largely demanded in the future.

In this book, we focus on the numerical methods for the parasitic extraction problems on IC interconnects and mixed-signal substrate modeling. It is mainly devoted to the methods and techniques for developing efficient 3-D field solvers for capacitance and resistance extraction. The statistical methods for the variation-aware capacitance extraction are also discussed, in both chip-level and local visions. The key field-solver techniques for RC extraction will be exposed in detail. Some of them have been widely used in IC industry, but not systematically presented ever in literature.

This book does not by any means intend to be comprehensive. The absence of coverage of related works should not diminish their value and contribution.

1.3 Book Outline

Since the early 1990s, the 3-D field solver for parasitic extraction has become an important research topic. A number of fast algorithms have been proposed for the problems of capacitance extraction, resistance extraction, inductance/impedance

extraction, substrate extraction, variation-aware capacitance extraction, etc. Most of them had been firstly presented on the Design Automation Conference (DAC), the International Conference on Computer-Aided Design (ICCAD), the Asia and South Pacific Design Automation Conference (ASP-DAC), and the Design, Automation, and Testing in Europe Conference (DATE). They also constitute the topics of several book chapters [75, 115, 180].

In this book, we systematically present the important techniques for fast RC field solver, along with some recent advancements in this area. Specially, large effort has been paid on the fast techniques based on boundary element method (BEM), which are the foundation of several state-of-the-art capacitance solvers. We also show in detail how the direct boundary element method can be applied to the calculation of interconnect resistance and the substrate parasitics. With specific accelerating techniques, efficient algorithms and prototypes are then developed for the industrial problems, respectively. As the strong competitor of the fast BEM algorithms, the floating random walk (FRW) algorithm for capacitance extraction has distinct advantages. In the last chapters of this book, we will present the efficient FRW algorithms for interconnect capacitance extraction.

It should be pointed out that the inductance/impedance extraction is important, but not included in this book. The modeling of inductive effect is indispensable in the design of radio-frequency (RF) microwave circuits and some analog circuits. However, for the digital IC, the inductive effect of interconnects has much less significance [107].

Below, we briefly introduce the following chapters in this book:

- In Chap. 2, the problem formulation and numerical methods in field solvers for RC extraction are briefly introduced. The methods are mainly indirect BEM, direct BEM, and the FRW method.
- In Chap. 3, we present the fast computing techniques for the capacitance extraction based on indirect BEM.
- In Chap. 4, we propose the quasi-multiple medium (QMM) techniques for the direct BEM-based capacitance extraction. Numerical experiments are presented to demonstrate their advantages.
- In Chap. 5, we turn to the topic of resistance extraction. Efficient techniques based on direct BEM are proposed for calculating the resistances of complex 3-D on-chip interconnects [159, 161].
- In Chap. 6, the problem of modeling and extracting substrate coupling effect is firstly introduced. Under the assumption of low signal frequency, BEM techniques are proposed to calculate the substrate resistances.
- In Chap. 7, we firstly give the problem formulation of frequency-dependent parasitic modeling for substrate coupling. Then, a two-step approach is proposed to extract the frequency-dependent substrate parasitics [177].
- In Chap. 8, we turn to the topic of capacitance extraction considering process variations. After presenting a fast approach for building capacitance library, we propose the techniques for chip-level statistical capacitance extraction considering spatially correlated random variations.

- In Chap. 9, we focus on variational capacitance modeling for detailed geometric variations of local interconnects. A continuous-surface geometric model is proposed, followed by relevant statistical extraction techniques and their application to modeling random line-edge roughness (LER).
- The last two chapters are devoted to the efficient techniques based on FRW method for capacitance extraction. In Chap. 10, we propose an approach to handle structures with multilayered dielectrics and a novel variance reduction technique to quicken the convergence of FRW procedure.
- In Chap. 11, the space management techniques for chip-scale large structures are proposed. This enables fast FRW-based solver for interconnect structures with over one million conductors. Prospects of the FRW method and the field solvers for capacitance extraction are finally given.

1.4 Summary

In this chapter, the motivations for IC parasitic extraction and the field-solver methods are firstly described. We have presented a short survey of current methods and challenges for the parasitic extraction. After briefly reviewing the research history of the field-solver methods for RC extraction, we present the aim of this book. Then, we briefly introduce all the chapters in the book. They can be divided into five parts:

- Introduction and fundamentals (Chaps. 1 and 2)
- Fast BEM algorithms for capacitance extraction (Chaps. 3 and 4)
- BEM-based techniques for resistance/substrate parasitic extraction (Chaps. 5, 6, and 7)
- Variation-aware interconnect capacitance extraction (Chaps. 8 and 9)
- Fast FRW algorithms for capacitance extraction (Chaps. 10 and 11)

Numerical examples are provided throughout the book to help the reader gain more insights into the discussed methods. Our treatment of those methods does not mean to be comprehensive, but we hope it can guide circuit designers and CAD software developers to understand the major ideas and limitations of their existing tools. It would be grateful if this book could help readers to apply those methods and to develop next-generation CAD tools for the design of emerging nanometer ICs.

Chapter 2
Basic Field-Solver Techniques for RC Extraction

Because 3-D numerical methods accurately model the realistic geometry, they possess the highest precision. The field solver based on 3-D numerical methods is not only used as a library-building tool in industrial RC extraction but also of increasing importance for the modeling and analysis of critical nets or key signal integrity issues. The major challenge for applying field solver is due to its low computational speed. Therefore, a lot of research has been devoted to improve the computational efficiency of 3-D capacitance extraction methods. With these works and the widely spreading parallel computing techniques, it is now possible to directly use the field solver in chip-scale extraction tasks. This is strongly demanded because of the increased accuracy requirement of parasitic extraction under the nanometer process technologies.

In this chapter, the principle and basic methods of 3-D field solver are to be introduced, with the emphasis on capacitance extraction. They are the basis of other RC extraction problems and more cutting-edge techniques for capacitance extraction. Those problems and techniques will be presented in detail, with the following chapters of this book.

2.1 Problem Formulation

As it is well known, the capacitor is a kind of circuit elements commonly used in electric or electronic equipments. It is usually composed of two conductors insulated from each other. When charged, the two surfaces of the conductors facing each other carry equal and opposite charges: Q and $-Q$, respectively (see Fig. 2.1). The electric potential difference between the two conductors $\phi_1 - \phi_2$ is called the voltage of the capacitor and is always denoted by V. Experiments, and theoretical analyses show that, for a capacitor, Q is always proportional to V and thus the ratio Q/V is a constant determined by the structure of the capacitor. This ratio is called the *capacitance* of the capacitor and denoted by C: $C = Q/V$.

W. Yu and X. Wang, *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*, DOI 10.1007/978-3-642-54298-5_2,

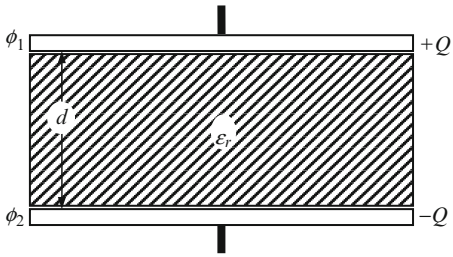

Fig. 2.1 A parallel plate capacitor

The SI unit of capacitance is faraday (F). It is the capacitance of a capacitor that has one coulomb on one of its conductor polar when the potential difference is 1 V. Other commonly used units of capacitance are μF (10^{-6} F), pF (10^{-12} F), and fF (10^{-15} F).

The capacitance of some simple capacitor can be calculated easily. For example, for the parallel plate capacitor shown in Fig. 2.1, we have

$$C = \frac{\varepsilon_0 \varepsilon_r S}{d}, \tag{2.1}$$

where ε_0 is the dielectric constant of free space (supposing that the plate's dimension is much larger than d). The dielectric constant takes the value

$$\varepsilon_0 = \frac{1}{4\pi \times 9 \times 10^9} = 8.85 \times 10^{-12}\ C^2/N \cdot m^2. \tag{2.2}$$

ε_r is the relative permittivity of the insulating material, S is the area of the facing plate, and d the distance between two parallel plates.

Actually, the capacitor has more generalized forms than that described above, which consists of two insulated conductors. The capacitance of a single conductor (conductor 1) is defined as if another conductor (conductor 2) was located at an infinite distance away to form a joint capacitor (conductors $1 + 2$). For example, the capacitance of an isolated conductor sphere with radius of R can be calculated as $C = 4\pi\varepsilon_0 R$.

There are many conductor interconnect wires in an integrated circuit (IC). The wires are insulated by some dielectric such as oxide SiO_2. The capacitance between any two wires reflects the electrostatic coupling between them. Calculating these capacitances with high accuracy is very important for the analysis of the circuit's performance.

For an N-conductor system, such as the interconnect wires in an IC, an $N \times N$ capacitance matrix $[C_{ij}]_{N\times N}$ is defined by

$$Q_i = \sum_{j=1}^{N} C_{ij}\phi_j, \quad i = 1, 2, \ldots, N, \tag{2.3}$$

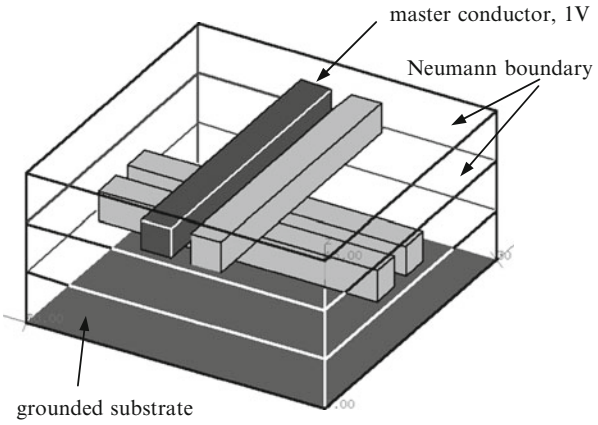

Fig. 2.2 A structure involving 2×2 crossover interconnect wires (Reprinted with permission from Yu and Wang [180] © 2005 John Wiley & Sons)

where C_{ij} $(i \neq j)$ is the coupling capacitance between conductor i and j and C_{ii} is called the self-capacitance or total capacitance of conductor i. Q_i is the induced charge on conductor i; ϕ_j is the electric potential of conductor j (usually the known bias voltage). Figure 2.2 shows typical crossover wires in VLSI circuit, where the coupling capacitance between any two conductors needs to be calculated.

Accurate modeling of the wire capacitances in a state-of-the-art IC is not a trivial task. The capacitance of a interconnect wire is a function of its shape, environment, distance from the substrate, and distance to surrounding wires. This usually calls for solving the electrostatic field in a region involving multiple dielectrics. Take the structure with three dielectric layers in Fig. 2.2 as an example. With one conductor (called *master conductor*) setting 1 V and others (called *environment conductors*) 0 V, the electric potential ϕ is governed by the *Laplace equation* for each homogenous dielectric region [70]:

$$\nabla^2 \phi = \frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} + \frac{\partial^2 \phi}{\partial z^2} = 0. \tag{2.4}$$

Taking into account the boundary conditions of the whole simulated region and those at the dielectric interfaces, the electrostatic field (potential and field intensity) can be solved. After that, we can obtain the charges of conductors. Note that the charge of an environment conductor equals to the capacitance between it and the master conductor, due to (2.3).

According to the boundary assumptions for the whole simulated region, there are several different models for capacitance extraction:

- The first one is called the infinite-domain model, because in it the electrostatic field spreads to the infinite, resulting in an infinite problem space.

- The second is called finite-domain model, where the electrostatic field is restricted within a finite domain, with the Neumann condition on the outer boundary [164]: $\partial\phi/\partial n = 0$. This means the electric field is not able to spread out of the finite simulated region. The Neumann condition is also called reflective boundary condition and was introduced as the "magnetic wall" in Hong et al. [61].
- The third boundary assumption employed in capacitance extraction problem is with a Dirichlet condition at the outer boundary [83]: $\phi = 0$. This is the default setting when employing the floating random walk method (see Sect. 2.5).

It should be pointed out that the infinite-domain model is ideal for simulating isolated structures, but for the IC capacitance extraction, it is not the case due to the influence of neighboring conductors. On the other hand, the finite-domain model considers a part cut from actual layout of IC. However, it assumes a condition not generally existing. Now, the three models of capacitance extraction are all used, for different extraction scenarios using different numerical methods. Although they produce different capacitance results for a given conductor system, the difference would be negligible if suitable size of the outer boundary is set. In the following discussions, the numerical methods will be introduced along with their suitable extraction models.

The background and problem formulations for resistance extraction, substrate parasitic extraction, and variation-aware capacitance extraction will be presented in Chaps. 5, 6, and 8, respectively.

2.2 Overview of the Numerical Methods

Basically, the methods for 3-D field solver are classified as the domain discretization method, the boundary integral equation method, semi-analytical approaches, and the stochastic method. The domain discretization method includes the finite difference method (FDM) [129, 144], the finite element method (FEM) [29, 31, 118, 137, 149, 169], and the method of the measured equation of invariance (MEI) [73, 88, 142]. The boundary integral equation method includes the method of moment [60, 124], indirect boundary element method (BEM) [13, 14, 27, 28, 54, 77, 78, 86, 98–100, 105, 106, 111, 132, 133, 136, 146, 164, 170, 171, 193–196], and direct BEM [4–6, 37, 46, 58, 63, 72, 89, 179, 181]. The semi-analytical approaches are combination of the analytical formulas and some traditional numerical methods [21, 61, 151, 152, 199, 200]. The stochastic method is based on the theory of random walk method [22, 83].

FDM and FEM discretize the whole 3-D simulated domain, thus producing a linear algebra system with large order. Hence, the computational speed of these methods is largely limited. However, since both methods are relatively well established, they are still used in the industry as a reference tool with accurate values calculated under fine grids. For example, the famous software of 2-D/3-D capacitance extraction "Raphael" utilizes FDM, and the "Q3D" of Ansoft Corp. is based on FEM.

Since the 1990s, the boundary integral equation method (i.e., BEM) has begun to replace the domain discretization method, thanks to its high efficiency. In both indirect and direct BEMs, only boundary of 3-D domain is discretized, and a smaller system of linear equations is obtained. Problems encountered with complex boundary can also be effectively handled with BEM, with accuracy comparable to FEM. Thus, BEM with rapid computing techniques has become the focus in the research of 3-D capacitance field solver.

Below, the basic method of indirect BEM is introduced. The principles of direct BEM and FRW method are discussed in the following two subsections, respectively. As for other field-solver methods, the reader is referred to the relevant references or [180].

2.3 Indirect Boundary Element Method

The indirect boundary method can be regarded as a variation of the method of moment (MoM). Because only domain boundary needs to be discretized, the indirect BEM involves much fewer unknowns than FDM and FEM. However, it leads to a dense coefficient matrix, whose formulation and solution introduce many difficulties. The innovation of the fast multipole method (FMM), the low-rank matrix compression method, and the hierarchical method had made the indirect BEM more applicable. Now, indirect BEM with a fast computing technique has become a major choice for 3-D field solver. Several commercial capacitance solvers have been developed based on the advanced indirect BEM approaches [78, 147].

The indirect BEM is also called the *equivalent charge method*, whose boundary integral equation involves the surface charge density $\sigma(\boldsymbol{r}')$ as an unknown function:

$$\phi(\boldsymbol{r}) = \int_{\Gamma} G(\boldsymbol{r}, \boldsymbol{r}')\, \sigma(\boldsymbol{r}')\, \mathrm{d}a', \quad (\boldsymbol{r} \in \Gamma), \tag{2.5}$$

where $G(\boldsymbol{r}, \boldsymbol{r}')$ is the Green's function and $\phi(\boldsymbol{r})$ stands for the electrostatic potential at point $\boldsymbol{r}$. For free space, $G(\boldsymbol{r}, \boldsymbol{r}') = 1/\|\boldsymbol{r} - \boldsymbol{r}'\|$; Γ is the boundary surface. After solving the surface charge density $\sigma(\boldsymbol{r}')$, the charge on conductor i can be calculated with

$$Q_i = \int_{S_d(i)} \sigma(\boldsymbol{r}')\, \mathrm{d}a', \tag{2.6}$$

where $S_d(i)$ is the surface of conductor i. We discretize the surfaces of m conductors into n constant elements (or panels), where charge density is assumed to be element-wise constant. Then, the potential at the center of the kth panel $\boldsymbol{r}_k$ can be expressed as a sum of the contributions of all panels:

$$\phi_k = \sum_{j=1}^{n} \int_{\Gamma_j} \frac{\sigma_j(\boldsymbol{r}')}{\|\boldsymbol{r}' - \boldsymbol{r}_k\|} \mathrm{d}a', \tag{2.7}$$

where $\sigma_j(\boldsymbol{r}')$ is the surface charge density of panel j (Γ_j). Substituting the known boundary conditions, we obtain a dense linear equation system:

$$\boldsymbol{P}\boldsymbol{q} = \boldsymbol{b}, \tag{2.8}$$

where the coefficient matrix $\boldsymbol{P}$ is dense and nonsymmetric. The Krylov subspace iterative method, such as GMRES algorithm [126], is usually used to solve the equation.

For a problem involving multiple dielectrics, the polarization charge density on dielectric interface needs to be introduced, which contributes to the potential distribution together with the free charge density on conductor surfaces. Therefore, the problem becomes equivalent to that in the free space, and the simple free-space Green's function is used to form Eq. (2.8). Except for Eq. (2.5) on each conductor panel, the normal derivative of the potential satisfies

$$\varepsilon_a \frac{\partial \phi_+(\boldsymbol{r})}{\partial \boldsymbol{n}_a} = \varepsilon_b \frac{\partial \phi_-(\boldsymbol{r})}{\partial \boldsymbol{n}_a}, \tag{2.9}$$

where $\boldsymbol{r}$ is on the interface of two dielectrics with permittivities ε_a and ε_b, respectively. Here $\boldsymbol{n}_a$ denotes the normal to the dielectric interface at $\boldsymbol{r}$ that points into dielectric a. $\phi_+(\boldsymbol{r})$ is the potential at $\boldsymbol{r}$ approached from the side of the interface ε_a, and $\phi_-(\boldsymbol{r})$ is the analogous potential for the b side. With (2.9), a discretized linear equation with variables of charge densities on conductors and dielectric interfaces can be formed for each dielectric interface panel. They are combined with (2.8) to obtain a linear equation system with the same number of equations and unknown variables.

For the multi-dielectric problem, the so-called *total-charge Green's function approach* presented above involves more unknowns at dielectric interfaces. Another choice to deal with the problem is to employ the multilayered Green's function. Then, only the free charge density on conductor surfaces needs to be considered as unknown function. However, to evaluate the Green's function for the multilayered medium, infinite summations are involved, which are computationally expensive. Oh et al. [105] derived a closed-form expression of the Green's function for the multilayered medium by approximating the Green's function using a finite number of images in the spectral domain. This greatly reduces the computational task. Li et al. [86] presented for the first time general analytical formulas for the static Green's functions for shielded and open arbitrarily multilayered media. Zhao et al. [193] proposed an efficient scheme for the generation of multilayered Green's functions using a generalized image method. The multilayered Green's function is much more complicated than the free-space Green's function and only applicable to the simple stratified structure of multiple dielectrics. While for more complex structures, such as the conformal dielectric, the deduction of the Green's function may be impossible.

More research work has been conducted to accelerate the capacitance extraction using the total-charge Green's function approach. In 1991, Nabors et al. successfully

applied the multipole accelerated method, proposed earlier by Greengard and Rokhlin [57], to 3-D capacitance extraction with the indirect BEM. In the MPA method, calculation of the interaction between charges [i.e., the coefficients in (2.8)] is divided into two parts: the near-field computation and far-field computation. For the near-field computation, the coefficients are calculated directly; for the far-field computation, the multipole expansion and local expansion are used to accelerate the computation. Therefore, the CPU time of forming and solving (2.8) with iterative equation solver is greatly reduced. In the successive works [98, 100], the adaptive, preconditioned MPA method was developed. Corresponding software prototype FastCap is shared on the MIT's website [97] and has become a popular academic tool of capacitance extraction. Other works on the capacitance extraction using the multipole accelerated indirect BEM can be found in Beattie and Pileggi [14] and Pan et al. [106].

In 1998, a fast hierarchical algorithm for 3-D capacitance extraction was presented on the *Design Automation Conference* [132] and was later presented in a journal article [133]. Similar to the fast multipole algorithm, it is based on indirect BEM and also originated from the fast computation of the *N-body problem* [9]. For the singular integral kernel of $1/\|\boldsymbol{r}-\boldsymbol{r}'\|$, it can achieve high speedup of computation. And only $O(N)$ operations are needed for each iteration while solver the linear equation system. For other weaker singular kernel, the efficiency of this method may be reduced. In 1997, Kapur et al. proposed an accelerating approach based on the *singular-value decomposition* (SVD) [77]. It is independent on the kernel and based on the Galerkin method using the pulse function as the base function. It needs $O(N)$ times operation to construct the coefficient matrix and $O(N\log N)$ operations to perform an iteration in the equation solution. Besides, the precorrected fast Fourier transform (pFFT) algorithm [111] has the same computational complexity, while it is based on the collocation method for discretization. Other fast computing approaches for indirect BEM include those based on wavelet transformation [136], multiscale method [146], and the second kind of integral equation [110].

The aforementioned works on capacitance extraction with indirect BEM all handle the infinite-domain model. In 1996, Wang et al. improved the multipole accelerated indirect BEM to make it suitable to handle the finite-domain problem [164]. And a parallel multipole accelerated 3-D capacitance simulator based on nonuniformed cube partition was proposed. In Beattie and Pileggi [13], a window technique is combined with the indirect BEM to reduce its computational expense, along with an error bound analysis for the capacitance results.

The SVD-based fast approach [77] employs a low-rank matrix compression technique. This idea has been further developed [15, 16, 78] and evolved to several commercial capacitance solvers. The recently developed $\boldsymbol{H}$-matrix-based fast approaches [27, 28] can be regarded as another way for matrix compression, which however enable solving the obtained linear system with direct linear equation solver. Several enhancements have also been proposed for the fast hierarchical algorithm [133]. In Yan et al. [170], a sparsification technique was proposed for the coefficient matrix generated from indirect BEM, which also facilitates an efficient

preconditioning technique for the iterative solution of the linear equation system. Then, a matrix reduction technique was proposed in Yan et al. [171] to gain further computational speedup. In Zhou et al. [196], the total-charge Green's function approach was combined with the multilayered Green's function approach to efficiently handle the realistic interconnect structure with multilayer and conformal dielectrics. To extend the application of capacitance field solver, a divide-and-conquer method was proposed in Shi and Yu [134]. It invokes the fast hierarchical BEM solver and extends it to the extraction task with a whole critical net.

More recently, the parallel computing techniques have been proposed for the multipole accelerated fast capacitance solver, on platforms of multi-core CPU [54] or single-instruction-multiple-data (SIMD) graphic processing units (GPUs) [194].

In Chap. 3, the technical details of fast multipole method and low-rank matrix compression in the indirect BEM capacitance solvers will be presented.

2.4 Direct Boundary Element Method

The direct BEM is based on the *direct boundary integral equation (direct BIE)* and suitable for solving 3-D Laplace equation with various boundary conditions [24]. Specifically, the direct BEM is very efficient for handling the finite-domain model of capacitance extraction.

Within the finite domain that is involved in capacitance extraction (see Fig. 2.2), electric potential ϕ fulfills the following Laplace equation with mixed boundary conditions [181]:

$$\begin{cases} \varepsilon_i \nabla^2 \phi = 0, & \text{in } \Omega_i \quad (i = 1, \ldots, M) \\ \phi = \phi_0, & \text{on } \Gamma_u \\ q = \partial\phi / \partial \boldsymbol{n} = 0, & \text{on } \Gamma_q, \end{cases} \tag{2.10}$$

where the whole domain $\Omega = \underset{M}{\cup} \Omega_i$ and Ω_i stands for the space possessed by the ith dielectric. Γ_u stands for the Dirichlet boundary (conductor surfaces), where ϕ is known as the bias voltages; Γ_q represents the Neumann boundary (outer boundary of the simulated region), where electric flux q is supposed to be zero. Here $\boldsymbol{n}$ denotes the unit vector outward normal to the boundary. At the dielectric interface, the compatibility equation (2.9) holds.

With the fundamental solution as the weighting function, the Laplace equations in (2.4) are transformed into following direct BIEs by the Green's identity [24]:

$$c_s u_s^i + \int_{\partial\Omega_i} q^* u^i \, \mathrm{d}\Gamma = \int_{\partial\Omega_i} u^* q^i \, \mathrm{d}\Gamma, \quad (i = 1, \ldots, M), \tag{2.11}$$

where u_s^i stands for the electric potential at collocation point s (in dielectric region i) and c_s is a constant dependent on the boundary geometry near to the point s.

$u^* = 1/4\pi r$ is the fundamental solution of 3-D Laplace equation, whose derivative along the outward normal direction $\boldsymbol{n}$ is $q^* = \partial u^*/\partial \boldsymbol{n} = -\left(\overrightarrow{\boldsymbol{r}}, \overrightarrow{\boldsymbol{n}}\right)/4\pi r^3$. r is the distance from the collocation point s to the point on Γ. $\partial\Omega_i$ is the boundary that surrounds dielectric region i. Note that the fundamental solution u^* is actually the free-space Green's function in (2.5).

Employing the collocation method after discretizing the boundary, like that in the indirect BEM, we get a system of linear equations [181]:

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{f}. \tag{2.12}$$

Finally, with the preconditioned Krylov iterative equation solver, the normal electric field intensity on the conductor surface is obtained [181].

In the direct BEM, both variables of potential and field intensity are involved. So, two kinds of integral kernels are dealt with. Though this is more complex than the indirect BEM, the direct BEM has its own advantages. Firstly, it is suitable for the capacitance extraction within the finite domain since two variables are included. On the second, because the variables in each BIE are within a same dielectric region, it has a so-called *localization character*. This character leads to a sparse linear system for problem with multiple dielectrics.

In the direct BEM, a great deal of time and memory are consumed in forming and solving the system of discretized BEM equations. Fukuda et al. [46] solved the problem of 2-D capacitance extraction using the direct BEM. In 1997, Bachtold et al. extended the multipole method to handle the "potential boundary integral" (whose kernel is $1/r^3$) in the direct BEM [4]. An adaptive boundary meshing scheme with an error indictor was also proposed. What they discussed was the model of multiple dielectrics within infinite domain. Hou et al. proposed an adaptive 3-D BEM solver for capacitance extraction [62], whose idea is to automatically meet the specified accuracy requirement by gradually refining the boundary meshes. In 2000, Gu et al. extended the fast hierarchical method used in the indirect BEM, making it feasible to the direct BEM-based capacitance extraction [58].

Yu et al. proposed a quasi-multiple medium (QMM) method, based on the localization character of direct BEM [181]. The QMM method exploits the sparsity of the resulting coefficient matrix when handling the multi-dielectric problem. Together with the efficient equation organization and iterative solving technology, the QMM-accelerated method greatly reduced the computing time and memory usage. In Yu et al. [181], an efficient analytical/semi-analytical integration scheme was also proposed to accurately calculate the boundary integrals under the VLSI planar process technology. The QMM method has been successfully applied to the actual 3-D multi-dielectric capacitance extraction [179, 181]. For finite-domain and multi-dielectric problems, the QMM-based method has shown 10× speedup and memory saving over the multipole accelerated indirect BEM (FastCap 2.0) with comparable accuracy [179]. Based on the direct BEM, techniques were recently proposed for fast 2-D capacitance extraction of nanometer VLSI interconnects [188].

There is another kind of approach for capacitance field solver, called *global approach*. It does not solve the resulting linear system in the usual way. The global approach discretizes the field equation and converts them to a circuit network of resistors or capacitors. Finally, with circuit reduction or matrix computation, the whole resistance or capacitance matrix can be obtained directly. In 1997, Dengi et al. proposed a global approach (called *macromodel method*) for 2-D interconnect capacitance extraction based on direct BEM [36, 37]. Lu et al. successfully extended the idea of boundary element macromodel to the 3-D case and developed a rapid *hierarchical block boundary element method (HBBEM)* for interconnect capacitance extraction [89]. HBBEM is suitable for extracting the whole capacitance matrix of an N-conductor system, with faster computational speed than the QMM-accelerated BEM. It have been extended and applied for the capacitance extraction problem in analog integrated circuits [176]. Besides, based on HBBEM, an IBM in-house tool (*CSurf*) has been developed for extracting the capacitances of interconnect and packaging structures [72].

In Chap. 4, the major techniques of QMM-accelerated BEM for capacitance extraction will be presented. Moreover, the fast direct BEM techniques have been developed for the resistance extraction and substrate extraction problems, which will be presented in Chaps. 5, 6, and 7.

2.5 Floating Random Walk Method

The FRW algorithm for capacitance extraction, presented in a 2-D version, was firstly proposed in 1992 [83]. Its basic idea is to convert the calculation of conductor charge to the Monte Carlo (MC) integration performed with floating random walks. This can be illustrated with the following equation:

$$\phi(\boldsymbol{r}) = \oint_S P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right)\phi\left(\boldsymbol{r}^{(1)}\right) \mathrm{d}\boldsymbol{r}^{(1)}, \tag{2.13}$$

where $\phi(\boldsymbol{r})$ is the electric potential at point $\boldsymbol{r}$ and S is a closed surface surrounding $\boldsymbol{r}$. $P(\boldsymbol{r}, \boldsymbol{r}^{(1)})$ is called the surface Green's function. For a fixed $\boldsymbol{r}$, $P(\boldsymbol{r}, \boldsymbol{r}^{(1)})$ can be regarded as the probability density function (PDF) for selecting a random point $\boldsymbol{r}^{(1)}$ on S. In this sense, $\phi(\boldsymbol{r})$ can be estimated by the mean value of $\phi(\boldsymbol{r}^{(1)})$, providing sufficient large number of sample points $\boldsymbol{r}^{(1)}$ on S are evaluated. If S is the surface of a homogeneous 3-D cube centered at $\boldsymbol{r}$, $P(\boldsymbol{r}, \boldsymbol{r}^{(1)})$ only depends on the relative position of $\boldsymbol{r}^{(1)}$ and is not related to the size of cube [69, 83]. More importantly, this surface Green's function can be derived analytically [69] and pre-calculated and stored as the discrete probabilities for jumping to the discretized cells of the cube surface.

In the situation that $\phi(\boldsymbol{r}^{(1)})$ is unknown, we apply (2.13) recursively to obtain the following nested integral formula:

$$\phi(\boldsymbol{r}) = \oint_{S^{(1)}} P^{(1)}\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \oint_{S^{(2)}} P^{(2)}\left(\boldsymbol{r}^{(1)}, \boldsymbol{r}^{(2)}\right) \cdots \oint_{S^{(k+1)}} P^{(k+1)}\left(\boldsymbol{r}^{(k)}, \boldsymbol{r}^{(k+1)}\right) \phi\left(\boldsymbol{r}^{(k+1)}\right) \mathrm{d}\boldsymbol{r}^{(k+1)} \cdots \mathrm{d}\boldsymbol{r}^{(2)} \mathrm{d}\boldsymbol{r}^{(1)}, \tag{2.14}$$

where $S^{(i)}$, $(i = 1, \ldots, k+1)$ is the surface of the ith cube centered at $\boldsymbol{r}^{(i-1)}$. $P^{(i)}(\boldsymbol{r}^{(i-1)}, \boldsymbol{r}^{(i)})$, $(i = 1, \ldots, k+1)$, are the surface Green's functions relating the potentials at $\boldsymbol{r}^{(i-1)}$ to $\boldsymbol{r}^{(i)}$. This can be interpreted as a floating random walk procedure: for the ith hop of a walk, the maximum conductor-free cube centered at $\boldsymbol{r}^{(i-1)}$ is constructed, and then a point $\boldsymbol{r}^{(i)}$ is randomly selected on the cube surface according to the discrete probabilities obtained with $P^{(i)}(\boldsymbol{r}^{(i-1)}, \boldsymbol{r}^{(i)})$. Note that, to obtain the probabilities, we only need to consider the normalized unit-size cube for the ith cube and the corresponding positions of $\boldsymbol{r}^{(i-1)}$ and $\boldsymbol{r}^{(i)}$ in the unit-size cube. The walk terminates after k hops if the potential at point $\boldsymbol{r}^{(k)}$ is known, e.g., it is on a conductor surface in the problem of capacitance extraction. With the surface Green's function and derived sampling probabilities for a unit-size cube calculated in advance [69], the major cost of random walk is for geometric operations. After performing many walks, the mean value of these estimates approximates $\phi(\boldsymbol{r})$ very well.

For extracting the capacitance, each walk starts from a point on a Gaussian surface enclosing the master conductor and terminates on a conductor surface after some successive hops. For each hop of a walk, a conductor-free cube centered at current location is constructed, and the hop reaches a random point on the cube's boundary. The FRW algorithm does not rely on assembling any linear equation system and has several computational advantages over the deterministic methods: lower memory usage, more scalability for large structures, tunable accuracy, and better parallelism.

The 3-D FRW algorithm for capacitance extraction has been developed and applied to the design and analysis of VLSI circuits [69, 75, 82]. In 2005, Batterywala et al. proposed several techniques to reduce the variance of MC procedure in the FRW-based capacitance extraction [12] and further reduce the total computing time. The FRW algorithm was also extended to handle the floating dummy-fills [11]. In 2008, a technique based on the FRW algorithm was proposed to enable fast incremental variational capacitance extraction [43]. A general FRW algorithm was also proposed in El-Moselhy et al. [43] for arbitrary dielectric configuration, where the whole problem domain was covered by a set of cubic transition subdomains for which the transition probability is numerically calculated online, rather than offline. This technique largely reduces the number of hops for a FRW walk, with the overhead of calculating and storing the transition probabilities for a lot of transition domains. The general FRW algorithm can be very time-consuming for a large-scale 3-D problem. A hierarchical FRW (HFRW) algorithm was later proposed for a fabric-aware extraction problem [44, 45], where the topological variation rather than the common non-topological variation was considered. Note that the HFRW is not suitable for the general problem of capacitance extraction, because an arbitrary structure cannot be regarded as the composition of predefined "motif" structures. Different from most of FRW-based capacitance solvers which employ

the cubic transition domain to suit the Manhattan geometry of VLSI interconnects, the technique using spherical transition domains was investigated in Brambilla et al. [22, 23].

Compared with the BEM-based techniques, there are much fewer literatures devoted to the 3-D FRW algorithms for multi-dielectric capacitance extraction. To fill in the gap between the theory and application of the FRW-based capacitance solver, a program set called RWCap [125, 187] has been developed since 2012. An approach to precisely handle the multilayered dielectrics, a novel variance reduction scheme for acceleration, and a parallel implementation on the multi-core/multi-CPU platform were presented in Yu et al. [187]. Efficient techniques were also proposed to parallelize the FRW-based capacitance extraction on GPUs [189]. The RWCap was further enhanced with a comprehensive space management technique, which facilitates efficient capacitance extraction of chip-scale large interconnect structures [190].

The advanced FRW techniques in RWCap will be introduced in the last two chapters of this book.

2.6 Summary

In this chapter, the problem formulation for the 3-D capacitance field solver is firstly described. Then, we present a survey on the numerical methods for capacitance extraction. The principles of indirect BEM and direct BEM are introduced, which form the basis for the techniques presented in Chaps. 3 and 4. They also provide a necessary background for the materials in Chaps. 5, 6, 7, 8, and 9. Finally, the FRW method and its state of the art are briefly introduced. Detailed treatment of this method will be given in the last two chapters of this book.

Chapter 3
Fast Boundary Element Methods for Capacitance Extraction (I)

As discussed in Chap. 2, the capacitance extraction is reduced to a linear system solution. Linear system solution methods are well studied, like Gaussian elimination [53]. However, such traditional methods usually require too much computational resources in terms of CPU time and memory. For capacitance extraction, how to solve in fast ways has been a hot topic of researches.

In the past years, multiple fast solvers were proposed for iterative solvers like GMRES [126]. In such solvers, the dominant consumer of time is matrix-vector product, so to expedite matrix-vector product is the core. For direct BEM, the quasi-multiple medium method [181] is a fast solver, since it can enhance the matrix sparsity; Chap. 4 will offer more details. The hierarchical method in Lu et al. [89] is to solve the system in a hierarchical way, where at each level small matrices are handled. For indirect BEM, the fast multipole methods [98–100], low-rank matrix compression methods [77, 78], and adaptive cross approximation (ACA) techniques [15, 16, 80, 120] are widely employed.

Below, we firstly introduce the basic idea of fast solvers under the context of indirect BEM. Then, these fast-solver methods are introduced one by one.

3.1 Basics of Indirect Boundary Element Methods

Indirect boundary element methods solve for the unknown electrical *charges* or their equivalents on conductor surface. For simplicity, suppose the enclosing mediums are homogeneous; otherwise, some multilayered Green's function [172] is needed. The surface is discretized into small panels, each of which is associated with an unknown charge σ. Then the electrical potential can be expressed as a sum of all panels' contribution:

W. Yu and X. Wang, *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*, DOI 10.1007/978-3-642-54298-5_3,

$$u = \sum_{j=1}^{n} \frac{\sigma_j (x')}{\|x' - x_k\|} \, \mathrm{d}s' \tag{3.1}$$

where σ_j is the unknown of panel $\mathbf{\Gamma}_j$. With the boundary condition that potential u is known on conductor surface, a linear system is produced:

$$\boldsymbol{P\sigma} = \boldsymbol{b} \tag{3.2}$$

where $\boldsymbol{P}$ is a dense matrix.

The traditional direct solvers like Gaussian elimination [53] are computationally expensive, since its memory complexity is $O(n^2)$ and CPU time complexity is $O(n^3)$. Obviously it's feasible only for small problems, or n is small. Otherwise, iterative solvers like GMRES solver [126] are good alternatives.

Iterative solvers work in a better-and-better way. It's offered a "guess" solution $\boldsymbol{x}_0$; then it computes the residual or how far the current "solution" is from the true one; in turn it will update the solution to $\boldsymbol{x}_{i+1}$, after getting the matrix-vector product $\boldsymbol{P}^*\boldsymbol{x}_i$. If the new $\boldsymbol{x}_{i+1}$ is not accurate enough, it repeats to update again, with $\boldsymbol{i} = \boldsymbol{i} + 1$, until accuracy requirement is met. For capacitance extraction, matrix $\boldsymbol{P}$ is usually dense and of large scale; computing the matrix-vector product is the most time-consuming part and may also be the most memory-consuming part, especially when the entire $\boldsymbol{P}$ is directly stored in memory.

A special property of iterative solvers is that it only asks for matrix-vector products, but not explicitly for the matrix $\boldsymbol{P}$ itself. This opens the door for fast solvers. Fast multipole methods don't generate and store the whole matrix before iterative solvers start. Instead, it generates the products on the fly. Compression-based fast solvers like IES3 [77] do store the matrix but only after it's remarkably compressed. We will soon see that compression means to squeeze "water" (redundant numerical info) out of "sponge" (matrix), which can save considerable memory and also CPU time.

3.2 Fast Multipole Methods

3.2.1 Introduction

As an iterative solver, the fast multipole method takes advantage of the fact that iterative solvers only require matrix-vector products, but not the matrix itself. When source points and evaluation points are well separated, their complete interactions can be significantly approximated in some way to arbitrary accuracy. Suppose the number of evaluation points and charge (source) points are m and n, respectively. Brute-force computation of the complete interactions will need $O(mn)$ entries, each starting from a charge point and ending at an evaluation point. In Fig. 3.1, for simplicity, only the interaction from a single charge point (center of a patch) is present. If all interactions were plotted, the full graph will be crowded.

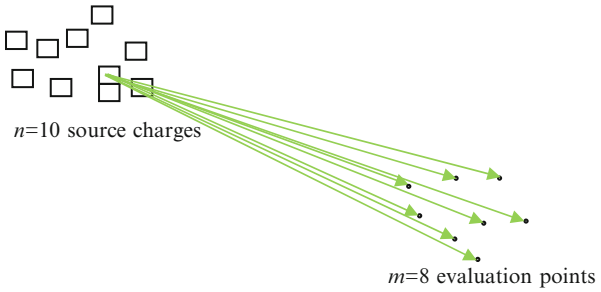

Fig. 3.1 Direct interaction between 10 source patches and 8 evaluation points. For simplicity, only interactions from a single source were depicted

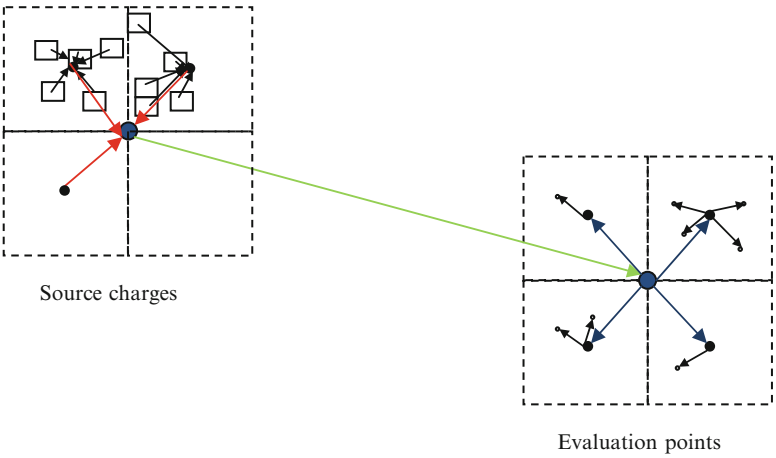

Fig. 3.2 Schematic fast multipole algorithm. Interactions between well-separated sources and evaluation points are done indirectly through two-level "centers." Sources map their charges to fine-level centers, which will in turn map the collected "charges" to upper level. The upper level conducts electrical potential on upper-level evaluation centers, which then pass the potential to fine-level evaluations

However, fast multipole method can reduce the interaction computation to order $O(m+n)$. A schematic diagram is in Fig. 3.2. In order to get product $\boldsymbol{Px}$, it takes $\boldsymbol{x}$ as electrical charges and $\boldsymbol{Px}$ correspondingly as the electrical potential resulted from these charges. Instead of interacting directly with evaluation points, the sources first "collect" their charges to a (finest-level) "center." This center can also do the same thing to the upper-level "center." The top-level center will apply its collected charge to the top-level evaluation point. This point will then pass the potential it received to its children "centers," which will caste the potential to final evaluation points. Also refer to Fig. 3.6.

This schematic description is of course not accurate, but it introduces two important processes: sources collect their charges to some other point, which can be called multipole expansion process, and finest-level evaluations receive potential from upper-level parent's center, which can be called local expansion process.

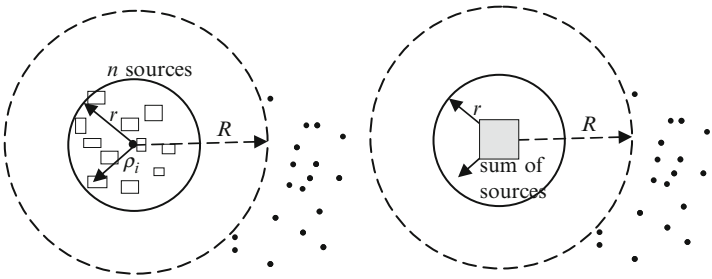

Fig. 3.3 Multipole expansion of order 0 for n source patches and evaluation points

3.2.2 Multipole Expansions

Suppose the charges are physically close to each other but considerably far away from the evaluation points. For a group of charge patches as shown in Fig. 3.3, the potential observed at evaluation point j can be computed by

$$\Psi\left(R_j,\theta_j,\phi_j\right)\approx\sum_{p=0}^{l}\sum_{m=-p}^{p}\frac{1}{R_j^{p+1}}M_p^m Y_p^m\left(\theta_j,\phi_j\right) \tag{3.3}$$

where l is the order of multipole expansion; R_j, θ_j, and ϕ_j are the spherical coordinates with respect to the center of source charges; and subscript j means for the jth evaluation point. Y_p^m is the spherical harmonics [70]. The *multipole coefficient* herein is

$$M_p^m=\sum_{i=1}^{n}q_i\rho_i^p Y_p^{-m}\left(\alpha_i,\beta_i\right) \tag{3.4}$$

where q *means the charge* and ρ, α, and β are the spherical coordinates again, but this time they are for the charges, with $i=1{:}n$ referring to all source charges.

Ideally, the expansion order l is infinite; then, there is no loss of accuracy. However, it's computationally accurate enough with a pretty low order provided that they are well enough separately. The expansion error with order of l is bounded by

$$\left|\Psi\left(R_j,\theta_j,\phi_j\right)-\sum_{p=0}^{l}\sum_{m=-p}^{p}\frac{1}{R_j^{n+1}}M_p^m Y_p^m\left(\theta_j,\phi_j\right)\right|<K_1\left(\frac{r}{R}\right)^{l+1} \tag{3.5}$$

where K_1 is a constant (independent of l), R is the radius of the charge sphere, and all the evaluation points have distances no less than r to the sphere center.

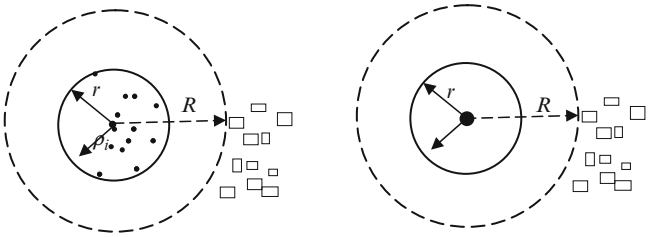

Fig. 3.4 Local expansion of grouped evaluation points and charge points

It's clear that if $r \ll R$, even a small order l (e.g., $l=1$ even $l=0$) can introduce enough accuracy. Specially, when $l=0$, multipole expansion is similar to aggregating all charge patches to a single but larger patch, i.e., the sphere center, as depicted in Fig. 3.2 right.

Equation (3.3) reveals that if all source charges are close to each other and far away from evaluation points, it's at complexity of $O(n+m)$ to evaluate all interactions between n_2 charges and m evaluation points, in two steps:

Step 1: Compute multipole coefficients as in (3.4), at cost $O(n)$.
Step 2: Compute potentials by (3.3), for $j=1{:}m$, at cost $O(m)$.

One may ask the question that what the computational complexity is if all evaluation points are close to each other (and well separately from the source charges). This can be answered by the local expansion below.

3.2.3 *Local Expansions*

Now consider the case that evaluation points are much close to each other but very far from charge points, as shown in Fig. 3.4. The potential at jth evaluation points with spherical coordinate (r_j, θ_j, and ϕ_j) is approximated to order l is

$$\Psi\left(R_j,\theta_j,\phi_j\right) \approx \sum_{p=0}^{l}\sum_{m=-p}^{p} L_p^m Y_p^m\left(\theta_j,\phi_j\right) R_j^p; \tag{3.6}$$

the local expansion herein is

$$L_p^m = \sum_{i=1}^{n} \frac{q_i}{\rho_i^{p+1}} Y_p^{-m}\left(\alpha_i,\beta_i\right). \tag{3.7}$$

Here ρ, α, and β are the spherical coordinates of charges with respect to the enclosing sphere of evaluation points. Then the expansion error is

$$\left|\Psi\left(R_j,\theta_j,\phi_j\right)-\sum_{p=0}^{l}\sum_{m=-p}^{p}L_p^m Y_p^m\left(\theta_j,\phi_j\right)R_j^p\right|<K_2\left(\frac{r}{R}\right)^{l+1}. \tag{3.8}$$

Const K_2 is independent of l, r, and R; r is the enclosing radius, and R is the minimum distance from the sphere center to the charge patches.

Similarly, it can be done in linear time to compute all the interactions between n charges and m evaluation points:

Step 1: Compute local expansion (3.7), at time cost $O(n)$
Step 2: Compute all potential (3.6) for $j = 1{:}m$, at time cost $O(m)$.

The total computation complexity is $O(m+n)$, which scales linearly.

3.2.4 *Fast Multipole Algorithm*

The above two expansions require charge points well separated to evaluation points; otherwise, accuracy will request too high order to be practical. In order to take full advantage of expansions, one can divide the whole geometry into many small zones. Then a small zone becomes well separated from some zones, but it is still close to some others. For the nearby zones, fast multipole can't be applied, and direct brute-force computation is needed. Refer to [99] for details.

One can go a step farther to hierarchically divide the problem zone into several levels; at each level, there will be faraway zones, where the fast multipole method can be applied. So the global algorithm is:

(a) Hierarchically divide the problem into cubes for 3-D problems (or squares for 2-D problems).
(b) At each level, identify near field and far field of each cube.
    1. Near field is the cubes that share edges or points with this cube.
    2. Far field is the other cubes apart from above that are the children of the near fields of the cube's father.
(c) Iterative solution of linear system.
    1. Start with an initial guess of solution vector $\boldsymbol{x}_0$.
    2. Improve iteratively the guess by getting the matrix-vector product.
        (i) The product is equivalent to the potential at evaluation points.
        (ii) For near field, the potential is directly computed.
        (iii) For far field, get the potential by multipole expansions and local expansions.

Figure 3.5 is an example for a typical 2-D problem. It's straightforward to extend it to 3-D cases.

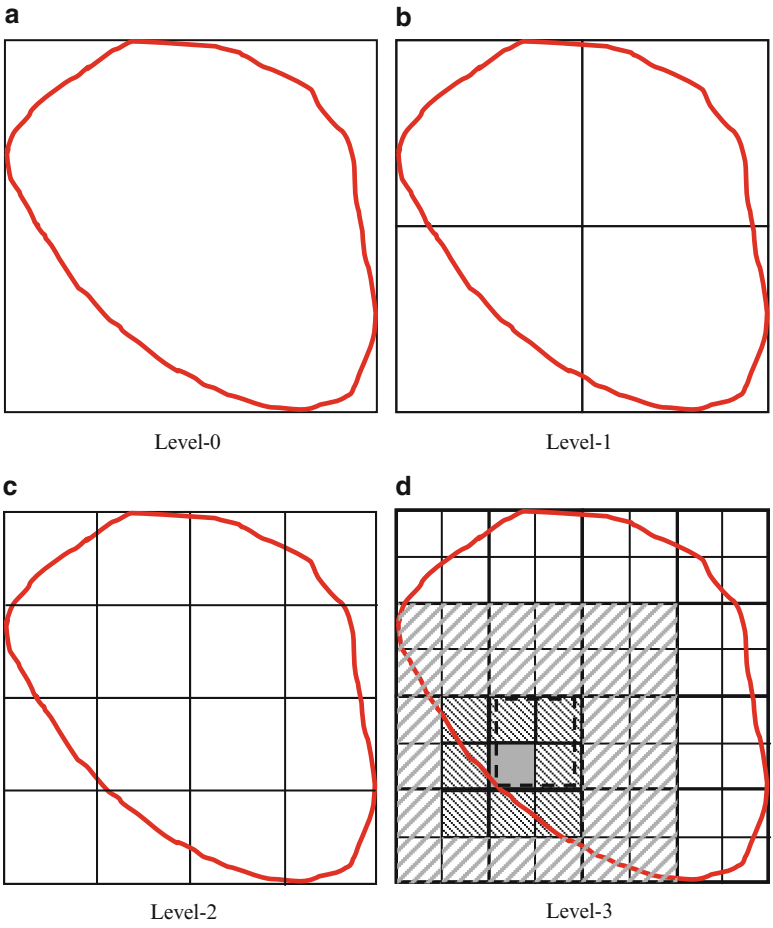

Fig. 3.5 A two-dimensional problem with irregular boundary and its squares up to level 3. The *square* in *gray* has 8 near fields (in dense shield) and 27 far fields (in sparse shield). The level-2 box in *dashed line* is the parent of the finest-level squares

The problem has an irregular boundary, while the smallest square that can enclose the entire boundary is actually the level-0 square, which is the parent for level-1 squares. Each of the four level-1 squares covers ¼ of the area and in turn has four children further, which compose the level 2. This procedure repeats until some criteria are met, for example, until the square has at least 10 charges.

It's easy to distinguish near fields and far fields of any squares in the figure. Level-1 squares don't have far fields. Let's see the level-3 square in gray. It has 8 near-field neighbors, each of which shares at least a common boundary point. Its parent box (in dashed line) also has 8 near-field neighbors; the nine level-2 boxes (including itself) have 36 level-3 squares. Among these 36 squares, apart from the gray box itself and its near-field neighbors, the left 27 squares are the far fields.

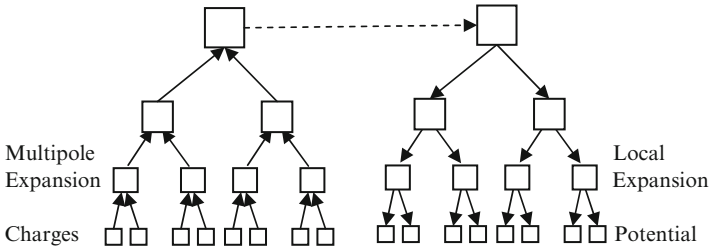

Fig. 3.6 Source charges contribute to the potential at evaluation points indirectly by multilevel multipole expansions and local expansions

Figure 3.6 illustrates how the charges *transfer* their contributions to the potential at evaluation points hierarchically. The charges contribute to the multipole expansions at the upper level; then the upper level will transfer to even upper levels. Then the upper-level boxes transfer to some other upper-level evaluation boxes, which in turn deliver to the finest-level evaluation points.

Since the major matrix-vector product is conducted with linear efforts, the fast multipole methods as a whole also have a linear complexity. However, this method may have its limitations. For example, it's not easy to be parallelized, which is a big shortage in the multi-core era.

Another kind of fast iterative method is based on matrix compression. It's really simple to be parallelized, as will be discussed later.

3.3 Low-Rank Matrix Compression-Based Fast Iterative Solvers

Recall that when evaluation points and source points are well separated, each source point looks similar to each other, and each can be somehow represented by the enclosing circle's center; refer to Fig. 3.3. From the view point of matrix, this means that the matrix columns corresponding to these source points are similar and can be roughly represented by a single column. Actually the representation can be of arbitrary accuracy, which is called low-rank compression.

3.3.1 Why Compression?

As discussed before, what's needed in the iterative solve is the matrix-vector product, in terms of

$$\boldsymbol{Px} = \boldsymbol{P}_{m\times n} * \boldsymbol{x}_n. \tag{3.9}$$

Table 3.1 Time scales with $O(mn)$ and $O(m+n)$

Size $m=n$	$O(mn)$ (days)	$O(mn)$ (years)	$O(m+n)$ (days)	$O(m+n)$ (years)
100	1	0.003	1	0.003
1,000	100	0.274	10	0.027
10,000	10,000	27.397	100	0.274
100,000	1,000,000	2,739.726	1,000	2.740
1,000,000	100,000,000	273,972.603	10,000	27.397

From now on, subscripts are used to indicate the matrix dimensions, i.e., $\boldsymbol{P}_{m\times n}$ means matrix $\boldsymbol{P}$ is of dimension $m \times n$. So the computation cost to get this product is $O(mn)$, since all the matrix entries are used at least one time.

Suppose we can have some approximation to matrix $\boldsymbol{P}$:

$$\boldsymbol{P}_{m\times n} \approx \boldsymbol{L}_{m\times r} * \boldsymbol{R}_{r\times n}, \tag{3.10}$$

where the middle-size r is called rank, usually $r \ll \min(m,n)$. Why bother to get the approximation? Because it can save a lot of computation:

$$\boldsymbol{P}_{m\times n}\boldsymbol{x}_{n\times 1} \approx \boldsymbol{L}_{m\times r}\boldsymbol{R}_{r\times n}\boldsymbol{x}_{n\times 1} = \boldsymbol{L}_{m\times r}\left(\boldsymbol{R}_{r\times n}\boldsymbol{x}_{n\times 1}\right). \tag{3.11}$$

By computing $\boldsymbol{Rx}$ first and then computing $\boldsymbol{L}(\boldsymbol{Rx})$, (3.11) can be done at cost of $O(r(m+n))$ or simply $O(m+n)$, because r is usually much less than m or n and can be seen as a constant in most cases.

3.3.2 *Matrix Compression Can Reduce the Complexity to Linear*

We've mentioned linear complexity many times, but why linear complexity is preferred? It's because we care about the algorithms' ability to scale well for large-sized problems. As m and n increases 10x, $O(mn)$ increases 100x or 10-squared times. This is why it is called squared complexity. By contrary, $O(m+n)$ only increases 10x, so it's called linear complexity, meaning its increasing is linear to m and n.

Suppose we can solve a problem with 100 unknowns in 1 day. Table 3.1 lists the time request for problem of larger sizes. For simplicity, it's assumed that $m=n$.

This table tells us that if the problem size increases 100x to 10,000, $O(mn)$ algorithms will require 27 years, while linear $O(m+n)$ algorithms only request in less than 4 months. This really makes a lot of senses for our limited life! And this can easily answer the question why everyone will like linear algorithms, if possible.

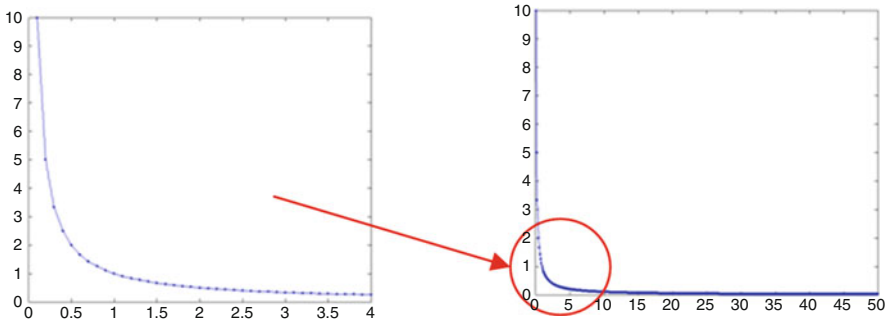

Fig. 3.7 The curve of $1/r$, with smaller and smaller slope as r increases

3.3.3 Compression Possible?

Let's see the matrix $\boldsymbol{P}$ in (3.2). Entry P_{ij} for evaluation point i and charge j is in the form of

$$\boldsymbol{P}_{ij} = \oint_{S_j} \int_{S_i} \frac{1}{\|x' - x_k\|} \mathrm{d}S_i S_j$$

where x' is on the surface of S_j and x_k is on the surface of S_i. $\|x' - x_k\|$ is the distance r between the two points. When they are faraway, its inverse $\frac{1}{\|x'-x_k\|}$ will decrease very slowly.

Figure 3.7 shows the inversion curve $1/r$, $r \in (0, 50)$. It's obvious that at the beginning, especially when r is smaller than 1, the curve decreases rapidly. While as r increases, the slope of the curve becomes smaller and smaller; when r is large enough, say, $r = 10$, the curve almost stagnates with a close-to-zero slope. Then one might think that a straight line is good enough to approximate the curve when r is large, no matter how many r values are present.

Then a natural question is, what about the above matrix $\boldsymbol{P}$? Does it also exhibit the same smoothing property?

Let's do some experiments here. Select random triangular patches whose centers are where the source charges are located; all the patches are enclosed by a cube of unit size. Similarly, select other random patches, whose center are the evaluation points; they are also enclosed by a unit cube. The observation cube is separated also by a unit size; refer to Fig. 3.8. Its matrix $\boldsymbol{P}$ is in Fig. 3.9.

This 100×100 matrix looks not smooth at all, but with some reordering scheme it may be. There are so many peaks and bottoms, in a messed way. However, this matrix can be remarkably *compressed*. Let's start with the basics of compression.

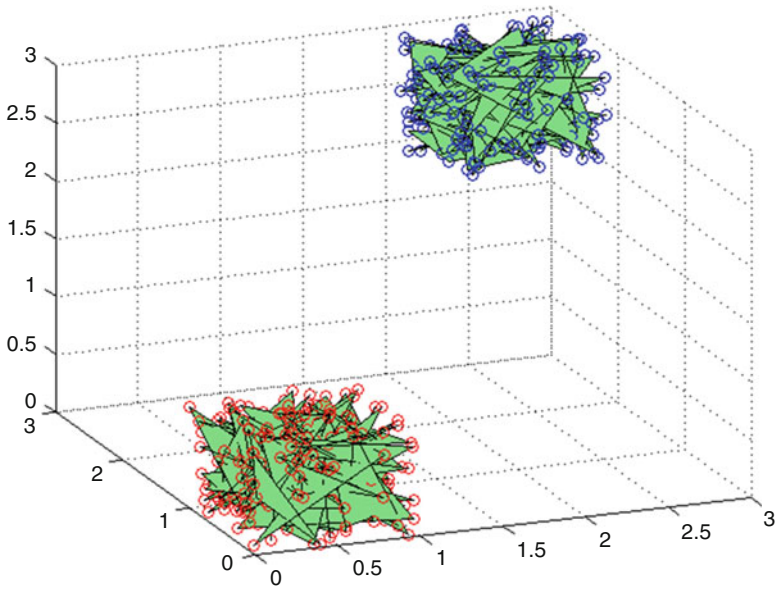

Fig. 3.8 100 source charges are located in the center of the patches at *bottom left*, while evaluation points are the centers of 100 *upright* patches

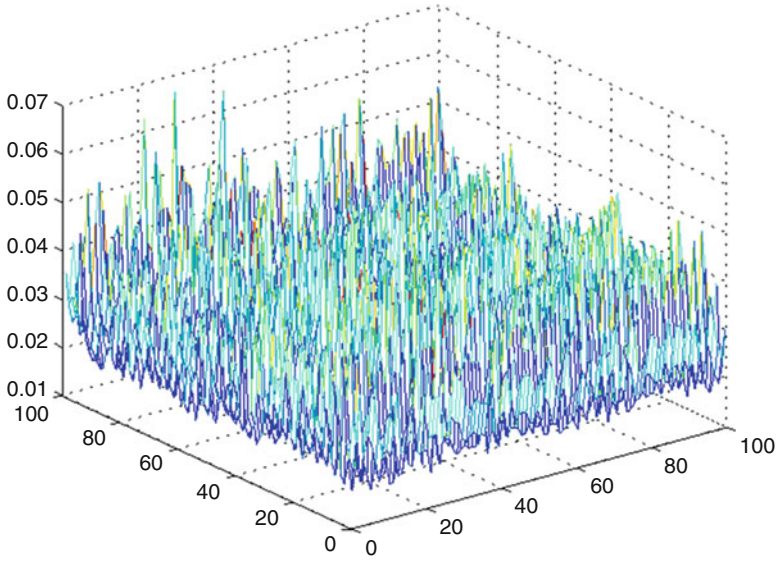

Fig. 3.9 Matrix $\boldsymbol{P}$ for the charges and evaluation points in previous figure

3.3.4 Basics of Matrix Compression Using SVD and QR

Singular-value decomposition (SVD) [53] is a powerful tool to analyze matrices. The SVD theory says that for any matrix of size $m \times n$ ($m \geq n$) (if $m < n$, simply transpose the matrix before decomposition), there is a unique decomposition:

$$\boldsymbol{P}_{m\times n} = \boldsymbol{U}_{m\times m}\boldsymbol{S}_{m\times n}\boldsymbol{V}_{n\times n}, \tag{3.12}$$

where $\boldsymbol{U}_{m\times m}$ and $\boldsymbol{V}_{n\times n}$ are both unitary, i.e., $\boldsymbol{U}_{m\times m}\boldsymbol{U}^{\mathbf{T}}_{m\times m} = \boldsymbol{I}_{m\times m}$, with $\boldsymbol{I}_{m\times m}$ being a unit matrix of size m. $\boldsymbol{S}_{m\times n}$ is a diagonal matrix, whose diagonal values are called singular values, $s_1 \geq s_2 \geq s_3 \geq \cdots \geq s_n \geq 0$, in form of

$$\boldsymbol{S}_{m\times n} = \begin{bmatrix} s_1 & & & & \\ & s_2 & & & \\ & & \ddots & & \\ & & & \ddots & \\ & & & & s_n \\ 0 & & \cdots & & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & & \ddots & & 0 \end{bmatrix}.$$

where the empty boxes mean zero.

We can define the rank-k approximation $\boldsymbol{U}_{m\times k^*}\boldsymbol{S}_{k\times k^*}\boldsymbol{V}_{k\times n}$, which is the truncated version of SVD in (3.12). A nice property is that its approximation error is S_{k+1},

$$\|\boldsymbol{P}_{m\times n} - \boldsymbol{U}_{m\times k}\boldsymbol{S}_{k\times k}\boldsymbol{V}_{k\times n}\| = s_{k+1}.$$

And it's also guaranteed that *this approximation is the best for all rank-k approximations*. In other words, matrix compression will get the best compression through SVD. While in reality, SVD involves too much computation, and a practical alternative is QR decomposition [53]. The rank-k approximation using QR can be expressed as $\boldsymbol{P}_{m\times n} \approx \boldsymbol{Q}_{m\times k^*}\boldsymbol{R}_{k\times n}$, usually with the close-to-optimal accuracy. Note that matrix Q is unitary, i.e., $\boldsymbol{Q}^*\boldsymbol{Q}^{\mathbf{T}} = \boldsymbol{Q}^{\mathbf{T}*}\boldsymbol{Q} = \boldsymbol{I}$; matrix $\boldsymbol{R}$ is upper triangular, i.e., all entries below main diagonal are zeros.

Let's see the singular values of matrix $\boldsymbol{P}$ in Table 3.2.

Note that the unitized singular values decrease below 1e-8 from the sixth one. According to SVD theories, using the larger values may be accurate *enough*. Let's see the approximations and error distributions with rank 30, rank 10, and rank 4, in Figs. 3.10, 3.11, and 3.12.

These figures tell us that rank-30 approximation is almost the same as the original matrix, without any visual discrepancy. Without looking at the error distribution, one

Table 3.2 Singular values of a $1/r$ matrix with random charge and evaluation points

No. of SV	Singular values	Unitized SV
1	2.8952	1
2	0.042169	0.014565
3	0.00070918	0.00024495
4	9.9916e-006	3.4511e-006
5	1.0845e-007	3.7458e-008
6	1.3703e-009	4.733e-010
7	1.5264e-011	5.2721e-012
8	1.1842e-013	4.0901e-014
9	9.4656e-016	3.2694e-016
10	7.9288e-016	2.7386e-016
11	2.0143e-016	6.9574e-017
12	4.6611e-017	1.6099e-017
...	...	...
30	2.9313e-017	1.0125e-017
31	2.8564e-017	9.8659e-018

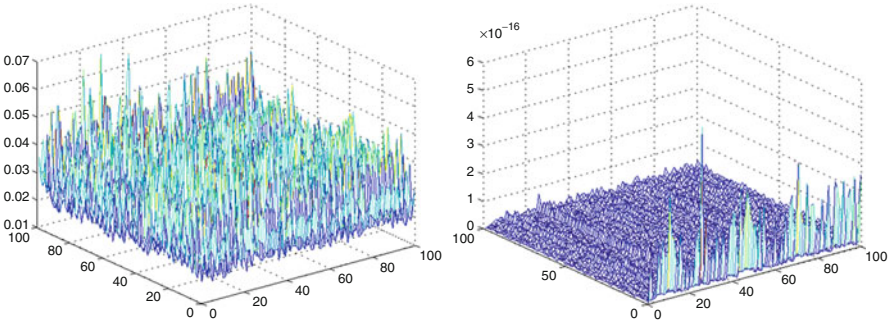

Fig. 3.10 Approximated matrix of rank 30 and its error to original matrix

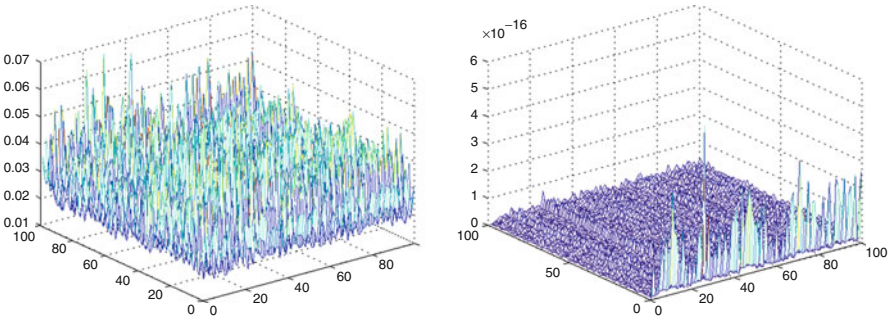

Fig. 3.11 Approximated matrix of rank 10 and its error to original matrix

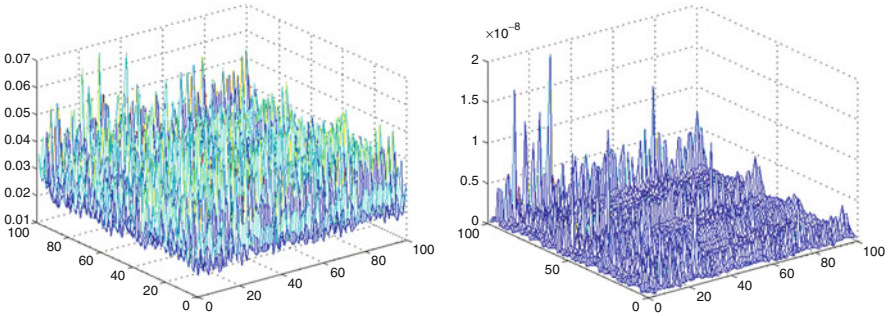

Fig. 3.12 Approximated matrix of rank 4 and its error to original matrix

might also think rank-4 approximation is very *accurate*. In many computation cases, like in capacitance extraction programs, rank-4 approximation is already accurate enough.

Although this case is only for kernel $1/r$, the low-rank property is also widely observed for full-wave simulations with kernel $e^{-jkr/r}$ [99, 100], which is very similar to $1/r$ at not-very-high frequencies or when the problem size is not comparable to wavelength.

Now we have agreed that compressed matrix is better, but here is a question: if we have to build the matrix in full first (anyway, this is an $O(mn)$ task), what's the point to do extra work to compress it? This is a really good and important question. The answer is don't build the matrix in full but sample only a few representative rows and columns. Based on the samples, a satisfactory approximation can be obtained.

3.3.5 *Compression Without Building Entire Matrix Beforehand*

Step 1: Randomly select a column and a row to be inserted to column index set I_c and row index set I_r, respectively.

Step 2: Now suppose we have selected N_c columns and N_r rows, with column and row index sets I_c and I_r, respectively. The columns comprise a matrix $\boldsymbol{P_c}$, and the rows comprise a matrix P_r. From the columns that are not selected yet, select the *best* one which maximizes some distance from $\boldsymbol{P_c}$:

$$c_i = \min_{\tilde{c} \in \boldsymbol{I_c}} \left\| \boldsymbol{P_c}(:, c) - \boldsymbol{P_c}(:, \tilde{c}) \right\|, \quad c_{\text{new}} = \min_{c_i \notin \boldsymbol{I_c}} \left\| c_i \right\|. \tag{3.13}$$

And then insert c_{new} into $\boldsymbol{I_c}$. Similarly, the new row index is also a maximizer:

$$r_i = \min_{\tilde{r} \in \boldsymbol{I_r}} \left\| \left\| \boldsymbol{P}_r(r, :) - \boldsymbol{P}_r(\tilde{r}, :) \right\| \right\|, \quad r_{\text{new}} = \min_{r_i \notin \boldsymbol{I_c}} \left\| r_i \right\|. \tag{3.14}$$

which is inserted to index set $\boldsymbol{I}_r$.

Step 3: Repeat step 2, until some criteria is met, for example, all the min values above are smaller than a tolerance.
Step 4: QR decompose $\boldsymbol{P}_c$ to get the $\boldsymbol{Q}$ matrix, as directed in (3.17).
Step 5: Solve (3.20) for $\boldsymbol{R}$, as directed in (3.21), (3.22), and (3.23).

Let's see why the $\boldsymbol{Q}$ in step 4 and $\boldsymbol{R}$ in step 5 satisfy $P \approx Q^* R$. With I_c and I_r being the selected indexes of columns and of rows, the set of corresponding matrix columns and rows are $\boldsymbol{P_c}$ and $\boldsymbol{P_r}$:

$$\boldsymbol{P} = \boldsymbol{Q} * \boldsymbol{R}, \tag{3.15}$$

$$\boldsymbol{P} * \boldsymbol{S_c} = \boldsymbol{Q} * \boldsymbol{R} * \boldsymbol{S_c}, \tag{3.16}$$

$$\boldsymbol{P_c} = \boldsymbol{Q} * (\boldsymbol{R} * \boldsymbol{S_c}), \tag{3.17}$$

where $\boldsymbol{S_c}$ is the selection matrix, whose diagonal elements are I_c.

Here is an example to better understand "selection matrix." If matrix $\boldsymbol{S_c}$ has two columns, with only two nonzero values at (2,1) and (2,4), then right multiplying $\boldsymbol{S_c}$ means to screen three columns or to *select* column 2 and column 4:

$$\begin{bmatrix} \alpha_2 & \alpha_4 \end{bmatrix} = \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \tag{3.18}$$

where α_i means column vectors.

Equation (3.17) *shows that the entire matrix* $\boldsymbol{P}_{m\times n}$ *and the sampled columns* $(\boldsymbol{P_c})_{m\times c}$ ($\boldsymbol{c}$ = *number of columns*) *share the same* $\boldsymbol{Q}$. *In other words, it suffices to compute the QR decomposition of* $(\boldsymbol{P_c})_{m\times c}$, *whose* $\boldsymbol{Q}$ *is actually also the* $\boldsymbol{Q}$ *of* $\boldsymbol{P}$.

If $\boldsymbol{P}$ were fully generated beforehand, one can easily get $\boldsymbol{R} = \boldsymbol{Q}^{\mathbf{T}*}\boldsymbol{P}$, T denoting transpose. But even if $\boldsymbol{P}$ were available, $\boldsymbol{R}$ can be obtained much quicker:

$$\boldsymbol{P_r} = \boldsymbol{S_r} * \boldsymbol{P}, \tag{3.19}$$

$$\boldsymbol{P_r} = \boldsymbol{S_r} * \boldsymbol{Q} * \boldsymbol{R}, \tag{3.20}$$

$$\boldsymbol{P_r} = (\boldsymbol{S_r} * \boldsymbol{Q}) * \boldsymbol{R}. \tag{3.21}$$

In (3.21), selected rows $\boldsymbol{P_r}$, row selection matrix $\boldsymbol{S_r}$, and $\boldsymbol{Q}$ (as in (3.17)) are already known; the desired $\boldsymbol{R}$ can be easily solved. A practical way is

$$\boldsymbol{P_r} = \boldsymbol{M} * \boldsymbol{R}, \tag{3.22}$$

$$\boldsymbol{P}_r = \boldsymbol{Q}_M \boldsymbol{R}_M * \boldsymbol{R}, \tag{3.23}$$

$$\boldsymbol{P}_r = \boldsymbol{Q}_M \boldsymbol{R}_M * \boldsymbol{R}, \tag{3.24}$$

where $\boldsymbol{M} = \boldsymbol{S}_r{}^* \boldsymbol{Q}$ and $\boldsymbol{Q}_M$ and $\boldsymbol{R}_M$ are the QR decomposition of $\boldsymbol{M}$.

Equation (3.21) *shows that the entire matrix* $\boldsymbol{P}_{m\times n}$ *and the sampled rows* $(\boldsymbol{P}_r)_{r\times n}$ ($\boldsymbol{r}$ = *number of rows*) *share the same* $\boldsymbol{R}$.

Then in short,

$$\boldsymbol{P} = \boldsymbol{Q} * \boldsymbol{R}, \tag{3.25}$$

where $\boldsymbol{Q}$ comes from (3.17) and $\boldsymbol{R}$ is from (3.24). Now it's obvious that the matrix $\boldsymbol{P}$ doesn't have to be entirely explicitly computed, but a limited number of rows and columns are sufficient to get its approximated QR decomposition, with arbitrary error tolerance. Usually the number of rows and columns are two times tolerance rank.

On performance side, computations in Eqs. (3.17) and (3.24) are limited to $O(m)$ or $O(n)$, which can be easily verified by the reader.

Note that the row and column sampling algorithm in step 2 is not the best. It's found to be prone to large approximation error [156]. More interesting sampling ideas can be found in [156].

Another good improvement over the compression is to reorganize the far-field cubes into the so-called merged interaction list (MIL) [56]. This makes sense because for the cubes that only contain few sources, compression is losing its efficiency because the rank is equal or comparable to the source number. Combining multiple cubes will then enhance the total compression ratio and generate a smaller number of matrix blocks, which both will benefit the matrix solution later.

This compression-based fast solver is inherently suitable for parallel computing, because all the matrix blocks are independent. Its parallel version includes distributed memory version (for computer clusters) [157], share-memory version (for multi-core CPU computers), and hybrid version (for clusters with multi-core computer nodes) [158].

Similar to the above QR decomposition-based compression technique, there is another compression technique called adaptive cross approximation (ACA). It's different in terms that it selects rows and columns in a very simply way.

3.4 Matrix Compression by Adaptive Cross Approximation

This algorithm tries to find a good approximation decomposition $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{V}^{\mathbf{T}}$. Note that this $\boldsymbol{U}$ is a general matrix; for example, it's not unitary or orthonormal. It doesn't request that the entire matrix $\boldsymbol{A}$ be formed beforehand, either. The difference

between ACA and the QR-based compression is that it has totally new sampling procedure, but usually it will produce a higher rank, where recompression [16] is helpful to reduce it.

3.4.1 Adaptive Cross Approximation (ACA)

This algorithm is to find appropriate $\boldsymbol{U}$ and $\boldsymbol{V}$, so that $\boldsymbol{P}_{m\times n} \approx \boldsymbol{U}_{m\times k^*} \boldsymbol{V}^{\mathbf{T}}_{n\times k}$, where rank k is much smaller than m and n. Suppose the previous k rows and columns are selected, and now the core is to find next $(k+1)$ row and column, in the following order:

$$\boldsymbol{R}_k\left(:, i_{k+1}\right) = \boldsymbol{P}\left(:, i_{k+1}\right) - \sum_{t=1}^{k} \boldsymbol{U}\left(i_{k+1}, t\right) * \boldsymbol{V}^{\mathrm{T}}\left(:, t\right), \tag{3.26}$$

$$j_{k+1} = \max_j \left|\boldsymbol{R}_k\left(i_{k+1}, j\right)\right|, \tag{3.27}$$

$$\boldsymbol{V}\left(:, k+1\right) = \frac{\boldsymbol{R}_k\left(i_{k+1}, :\right)}{\boldsymbol{R}_k\left(i_{k+1}, j_{k+1}\right)}, \tag{3.28}$$

$$\boldsymbol{U}\left(:, k+1\right) = \boldsymbol{P}\left(:, j_{k+1}\right) - \sum_{t=1}^{k} \boldsymbol{V}\left(j_{k+1}, t\right) * \boldsymbol{U}\left(:, l\right), \tag{3.29}$$

$$i_{k+2} = \max_{i\neq i_{k+1}} \left|\boldsymbol{U}\left(i, k+1\right)\right|. \tag{3.30}$$

Now let $k = k+1$, go to (3.26), until $\|\boldsymbol{U}(:,k)\|_F \|\boldsymbol{V}(:,k)\|_F \leq \in \|\boldsymbol{A}_k\|_F$. The last F-norm can be efficiently computed in an incremental way

$$\begin{aligned}\left\|\boldsymbol{A}_k\right\|_F^2 = \left\|\boldsymbol{A}_{k-1}\right\|_F^2 + 2\sum_{t=1}^{k-1}\left(\boldsymbol{U}\left(:, t\right), \boldsymbol{U}\left(:, k\right)\right)\left(\boldsymbol{V}\left(:, t\right), \boldsymbol{V}\left(:, k\right)\right) \\ + \left\|\boldsymbol{U}\left(:, k\right)\right\|_F^2 \left\|\boldsymbol{V}\left(:, k\right)\right\|_F^2.\end{aligned}$$

Note that only a small number of rows and columns are involved in the above procedure, and the computation is simply multiplication and addition. Therefore, the complexity is also linear $O(m+n)$.

An interesting property of ACA is that matrix $\boldsymbol{UV}^{\mathbf{T}}$ is exact at the selected columns and rows, $(\boldsymbol{UV}^{\mathbf{T}})(i_{k+1}, :) = \boldsymbol{A}(i_{k+1}, :)$, $(\boldsymbol{UV}^{\mathbf{T}})(:, j_{k+1}) = \boldsymbol{A}(:, j_{k+1})$.

Compared with the unitary matrix $\boldsymbol{Q}$ and upper-triangular matrix $\boldsymbol{R}$ in QR decomposition, $\boldsymbol{U}$ and $\boldsymbol{V}$ are general matrices. In addition, the *rank* revealed by this algorithm is usually larger than that in the above QR approximation, not to mention than that in SVD decompositions. To get the best performance for matrix-vector products, it's beneficial to reduce the rank in some way, which is called recompression. Note that recompression is a plus, not a must for ACA.

Table 3.3 Comparison of the fast field solvers

	Fast multipole methods	Matrix compression-based methods
Kernel dependency	Yes	No
	Even for capacitance extraction, a multilayered Green function may vary with the geometry or the physical parameters; then, the fast multipole code has to be considerably revised	The compression only requires some columns and rows of the matrix, but it doesn't have to know what the kernel is. It can work similarly for various kernels even beyond the field of capacitance extraction, for example, for full-wave simulations, provided that the matrix is of low ranks
Far-field matrix memory	Matrix is not computed or stored in memory explicitly	Matrix is computed once (in compressed form) and stored in memory.
Parallel computing	Hard	Very easy
	It's essentially serial computing. For each iteration in the iterative solve, it has to update multipole moments first, pass it from bottom to up, and then compute the local expansions, in sequential	Matrix is inherently partitioned into many parts. Each part can be compressed independently, and more importantly, each part can do its own matrix-vector product simultaneously in iterative solvers

3.4.2 Recompression of Adaptive Cross Approximation

Since $\boldsymbol{U}$ and $\boldsymbol{V}$ above are general matrices, they may include redundant information. So they themselves can be compressed with controllable accuracy, just to reduce the rank and then speed up the matrix-vector product later.

Let's start from tolerance-controlled QR decomposition of $\boldsymbol{U}$ and similarly of $\boldsymbol{V}$:

$$\boldsymbol{U} \approx \boldsymbol{Q}_U \boldsymbol{R}_U, \tag{3.31}$$

where $\text{rank}(\boldsymbol{Q}_U) \leq \text{rank}(\boldsymbol{U})$.

$$\boldsymbol{V} \approx \boldsymbol{Q}_V \boldsymbol{R}_V, \tag{3.32}$$

$$\boldsymbol{A} = \boldsymbol{U} * \boldsymbol{V}^{\mathrm{T}} \approx \boldsymbol{Q}_U \boldsymbol{W} \boldsymbol{Q}_V^{\mathrm{T}}, \tag{3.33}$$

where $\boldsymbol{W} = \boldsymbol{R}_U \boldsymbol{R}_V^{\mathrm{T}}$. Then we can do SVD of $\boldsymbol{W}$:

$$\boldsymbol{W} \approx \boldsymbol{U}_W \boldsymbol{S}_W \boldsymbol{V}_W, \tag{3.34}$$

$$\boldsymbol{A} \approx (\boldsymbol{Q}_W \boldsymbol{U}_W \boldsymbol{S}_W) * \left(\boldsymbol{V}_W \boldsymbol{Q}_V^{\mathrm{T}}\right) = \boldsymbol{U}_1 \boldsymbol{V}_1, \tag{3.35}$$

where the two new matrices $\boldsymbol{U}_1 = \boldsymbol{Q}_U \boldsymbol{U}_W \boldsymbol{S}_W$, $\boldsymbol{V}_1 = \boldsymbol{Q}_V \boldsymbol{V}_W^{\mathrm{T}}$. It's easy to verify that $\mathrm{rank}(\boldsymbol{U}_1) \leq \mathrm{rank}(\boldsymbol{U})$ and $\mathrm{rank}(\boldsymbol{V}_1) \leq \mathrm{rank}(\boldsymbol{V})$.

The recompression loses some accuracy in (3.31), (3.32), and (3.34). This error is built on the base of compression error of ACA itself. On the other hand, the benefit is that matrix-vector product will be faster with the reduced rank, which makes more sense for more iterations in the iterative solver.

3.5 Summary

In this chapter, two kinds of fast field solvers are discussed, mainly for the capacitance extraction purposes. Nowadays, the multi-core CPUs, even many-core CPUs, are more and more popular, then it makes sense to talk about the parallel computing of the two methods. The detailed comparison of the two fast methods based on indirect BEM is given in Table 3.3.

Chapter 4
Fast Boundary Element Methods for Capacitance Extraction (II)

In the capacitance extraction of VLSI interconnects, a finite region governed by the Laplace equation with mixed boundary conditions is often considered [164]. The Neumann boundary condition is assumed to the simulated region, as in several existing works [37, 46, 61, 164, 199, 200]. For this problem setting, the direct boundary element method (BEM) is more suitable for extracting the capacitance than indirect BEM. This is because there are two variables, the electric potential and its normal derivative, in the direct boundary integral equation (BIE) [24, 76]. Compared with the semi-analytical approaches proposed by W. Hong et al. [61, 199, 200], the direct BEM can deal with more complicated 3-D structure of the interconnects. Note that the semi-analytical approaches have some limitations on the geometry they handle [142]. However, the direct BEM generally leads to a nonsymmetric coefficient matrix, and the matrix for single-dielectric region is dense. This causes much computational expense in forming and solving the resulted linear equation system [24].

In this chapter, a novel quasi-multiple medium (QMM) technique is proposed to accelerate the direct BEM computation. It utilizes the localization character of direct BEM to transfer the coefficient matrix into a highly sparse block matrix. With the technology of storing sparse block matrix and efficient preconditioned iterative equation solver, the QMM technique can greatly reduce the CPU time and memory usage of large-scale direct BEM computation. Numerical experiments are carried out to validate the efficiency of the QMM technique. The comparisons with the overlapped domain decomposition method (ODDM) [200], geometry-independent measured equation of invariance (GIMEI) method [142], and fast multipole BEM [100] show the advantages of the QMM-accelerated BEM. Then, our method is extended to efficiently handle the structure with floating dummy-fills and demonstrates large speedup over Raphael of Synopsys Inc. [144] and a fast solver called PASCAL [84].

It should be pointed out that the QMM technique is mainly used in the capacitance model with finite region, not the infinite-domain (open-space) model

W. Yu and X. Wang, *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*, DOI 10.1007/978-3-642-54298-5_4,

considered by other BEMs [100, 133, 165]. And, in some methods for capacitance extraction (including ours) and commercial softwares (such as Raphael and SpiceLink[1]), the finite-domain model is considered as the default setting.

4.1 Direct Boundary Element Method for Multi-dielectric Capacitance Extraction

For an interconnect structure with N_C conductors embedded in M dielectric layers, we set the jth conductor (master conductor) to 1 V and the rest to 0 V. By solving the electrostatic field, we can obtain the self- and coupling capacitances of the jth conductor. This procedure can be repeated N_C times for getting the whole capacitance matrix.

Within the 3-D domain of the ith dielectric denoted by Ω_i, the electric potential u is governed by the Laplace equation with mixed boundary conditions:

$$\begin{cases} \nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = 0, & \text{In } \Omega_i \quad (i = 1, \ldots, M) \\ u = u_0, & \text{On } \Gamma_u \\ q = \frac{\partial u}{\partial \boldsymbol{n}} = 0, & \text{On } \Gamma_q, \end{cases} \tag{4.1}$$

where q is the normal electric field intensity on the outer boundary Γ_q. Along the interface of two adjacent dielectrics a and b, the compatibility equation holds:

$$\begin{cases} \varepsilon_a \cdot \partial u_a / \partial \boldsymbol{n}_a = -\varepsilon_b \cdot \partial u_b / \partial \boldsymbol{n}_b \\ u_a = u_b, \end{cases} \tag{4.2}$$

where ε_a and ε_b stand for the permittivities of dielectric a and b, respectively. With the fundamental solution u^* as the weighting function, the Laplace equations in (4.1) are transformed into the following direct BIEs by the Green's identity [24]:

$$c_s u_s + \int_{\partial\Omega_i} q^* u \mathrm{d}\Gamma = \int_{\partial\Omega_i} u^* q \mathrm{d}\Gamma, \quad (i = 1, \ldots, M), \tag{4.3}$$

where u_s is the electric potential at source point s, c_s equals to 1/2 if s is on a planar boundary, and q^* is the derivative of u^* along the outward normal direction of boundary $\partial\Omega_i$.

Employing boundary element partition and evaluating the direct BIE at collocation points (i.e., source points), one for an element, the discretized BIEs for the ith dielectric are achieved:

[1]The early version of Q3D^{TM} of Ansoft Corp.

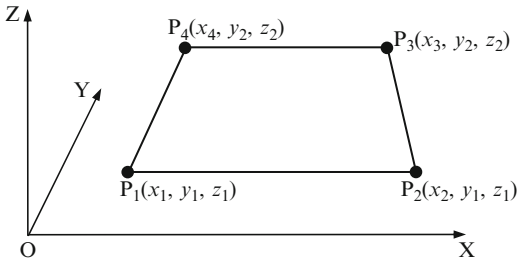

Fig. 4.1 The trapezoid element $P_1P_2P_3P_4$, whose hemlines are parallel to the X-axis

$$c_k u_k + \sum_{j=1}^{N_i} \left(\int_{\Gamma_j} q^*_{(k)} \mathrm{d}\Gamma \right) u_j = \sum_{j=1}^{N_i} \left(\int_{\Gamma_j} u^*_{(k)} \mathrm{d}\Gamma \right) q_j, \quad (k = 1, \ldots, N_i), \tag{4.4}$$

where N_i is the number of the boundary elements in dielectric i and Γ_j is the jth element.

The evaluation of integrals in (4.4) costs a major part of the computational time of boundary element algorithms, in particular for 3-D analysis [25, 46]. The integrals can be classified as the singular integrals and non-singular integrals. When the source point is on the same element where the integral is taken, i.e., $k = j$ in (4.4), it is singular integral; otherwise, it is non-singular integral. For the singular integral, the analytical integral method adopting local polar coordinates is effective [66]. The Gauss-Legendre integration scheme with adaptive determination of integration points is employed to calculate the non-singular integral [25, 46]. But for the nearly singular integrals, when the source point is close to the element where the integral is taken, the order of Gauss-Legendre integration is still very high. So, reducing the calculating time of the nearly singular integrals becomes very important for the application of direct BEM.

We propose a semi-analytical method to deal with these nearly singular integrals. Consider the boundary elements in the discretization of VLSI geometries, which are rectangle, parallelogram, trapezoid, or triangle. Because rectangle, parallelogram, and triangle are special trapezoids, here we consider the trapezoid only. Generally, the hemlines of the trapezoid are parallel to one coordinate axis. Without loss of generality, assume that it is X-axis, as shown in Fig. 4.1.

For the 2-D integral taken on the trapezoid element in Fig. 4.1,

$$I = \int_{\Gamma_j} f\,(x, y, z)\,\mathrm{d}\Gamma, \tag{4.5}$$

making transformation adopting X and Y as local coordinate axes, we get

$$I = \sqrt{1 + K^2} \int_{y_1}^{y_2} \int_{X_1}^{X_2} f\,(x, y, z)\ \mathrm{d}x\mathrm{d}y, \tag{4.6}$$

where $K = (z_2 - z_1)/(y_2 - y_1)$, $X_1 = (y - y_1) \cdot (x_4 - x_1)/(y_2 - y_1) + x_1$, and $X_2 = (y - y_1) \cdot (x_3 - x_2)/(y_2 - y_1) + x_2$.

If the inner integral can be calculated by analytical integration, and F is the primitive function of f on variable x, (4.6) can be written as

$$I = \sqrt{1+K^2}\int_{y_1}^{y_2} \left[F\left(X_2, y, z\right) - F\left(X_1, y, z\right)\right] \mathrm{d}y, \tag{4.7}$$

where $z = K \cdot (y - y_1) + z_1$.

In the discretized BIE (4.4), the integral kernels are $q^*_{(k)} = \partial u^*/\partial n = -d/4\pi r^3$ and $u^*_{(k)} = -1/4\pi r$. So, the integration in our method is on the kernels $1/r$ and $1/r^3$, but omitting the constants. We will analyze both kernels as follows:

For kernel $f_u(x, y, z) = 1/r$, we get

$$I_u = \sqrt{1+K^2}\int_{y_1}^{y_2} \ln\left[\frac{X_2 - x_s + \sqrt{(X_2 - x_s)^2 + (y - y_s)^2 + (z - z_s)^2}}{X_1 - x_s + \sqrt{(X_1 - x_s)^2 + (y - y_s)^2 + (z - z_s)^2}}\right] \mathrm{d}y. \tag{4.8}$$

For kernel $f_q(x, y, z) = 1/r^3$, we get

$$I_q = \sqrt{1+K^2}\int_{y_1}^{y_2} \left\{\frac{X_2 - x_s}{\left[(y-y_s)^2 + (z-z_s)^2\right]\sqrt{(X_2-x_s)^2+(y-y_s)^2+(z-z_s)^2}} - \frac{X_1 - x_s}{\left[(y-y_s)^2 + (z-z_s)^2\right]\sqrt{(X_1-x_s)^2+(y-y_s)^2+(z-z_s)^2}}\right\} \mathrm{d}y. \tag{4.9}$$

In the above two expressions, (x_s, y_s, z_s) is the source point. Using one-dimensional (1-D) Gauss-Legendre integration, the value of I_u and I_q can be obtained from (4.8) and (4.9).

In the 1-D Gauss-Legendre integration, the number of integration points can be dynamically determined according to the value range of y. In actual 3-D interconnect capacitance extraction, most of the boundary elements are rectangle element perpendicular to coordinate axis. For this case, i.e., $K = 0$, $X_1 = x_1$, and $X_2 = x_2$, the analytical integral formula can be further deducted. If many integration points are required for a non-singular integral, the analytical formula can be used to calculate it. Otherwise, the semi-analytical formula is used. Our semi-analytical and analytical integral method not only improves the accuracy of the non-singular integrals but also increases the computational speed of them.

After calculating the integrals, a matrix equation for each dielectric is formed:

$$\boldsymbol{H}^i \cdot \boldsymbol{u}^i = \boldsymbol{G}^i \cdot \boldsymbol{q}^i, \quad (i = 1, \ldots, M), \tag{4.10}$$

where $\boldsymbol{u}^i$ is the column vector of electric potential on boundary of dielectric i and $\boldsymbol{q}^i$ is the column vector of normal electric field intensity. $\boldsymbol{H}^i$ and $\boldsymbol{G}^i$ are the corresponding coefficient matrixes, respectively. Both vectors of $\boldsymbol{u}^i$ and $\boldsymbol{q}^i$ have order of N_i.

Matrix equations (4.10) can be put together utilizing the compatibility equations (4.2). Then we reorganize the equation system, such that all unknown variables are collected in a left-hand side vector, while a corresponding right-hand side vector is obtained by multiplying matrix entries with the known values of u and q. This gives

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{f}. \tag{4.11}$$

The coefficient matrix $\boldsymbol{A}$ is a large nonsymmetric one for 3-D problem. The Krylov subspace iterative methods are efficient to solve them. A preconditioned GMRES algorithm is used here [126]. After solving (4.11), the self- and coupling capacitances can be evaluated by the integral of normal electric field intensity on the conductor surfaces [46, 100].

4.2 The Quasi-multiple Medium Approach

In this section, we firstly present the basic idea of the quasi-multiple medium approach. The strategies for decomposing dielectric layers and element partition are given later. Lastly, we present the algorithm description of QMM-accelerated BEM, along with the analysis of its computational complexity.

4.2.1 Basic Idea

From (4.4) we can see that, in each discretized BIE, all discretized variables are on the boundary elements of one dielectric region. So, there are direct interactions among the boundary elements in the same dielectric, which result in nonzero coefficients in the overall equation (4.11). We call this the *localization character* of direct BEM.

In the linear system (4.11), coefficient matrix $\boldsymbol{A}$ reflects the distribution of interactions among all boundary elements. If there is the direct interaction between two elements, nonzero entries are formed by the integrals taken on one of the elements with the source point on the other. Otherwise, when the source point and the discrete variable are on the elements without direct interaction, i.e., not involved in a same dielectric, a corresponding zero entry is in matrix $\boldsymbol{A}$. For a problem with multiple dielectrics, the localization of direct BEM makes matrix $\boldsymbol{A}$ sparse, from which we could benefit while storing and solving the linear equation system (4.11).

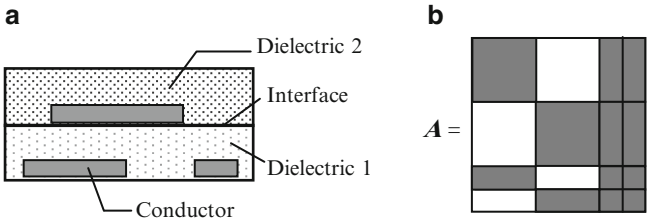

Fig. 4.2 (**a**) A 2-D problem with two dielectrics and (**b**) the corresponding coefficient matrix $\boldsymbol{A}$, where the *gray* blocks stand for nonzero entries

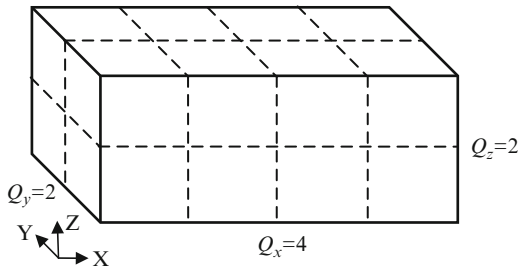

Fig. 4.3 A single dielectric with Cartesian coordinate system is cut into $Q = Q_x \times Q_y \times Q_z$ fictitious mediums

In Fig. 4.2, we show a typical structure with two dielectrics and the corresponding matrix $\boldsymbol{A}$ generated by the direct BEM, where the nonzero entries and the location of discrete variables are indicated.

The QMM approach takes full advantage of the localization character of direct BEM. A single dielectric with permittivity ε is regarded as a composition of Q fictitious medium blocks, whose permittivities are all the same as ε, as shown in Fig. 4.3. Thus, the problem with single medium is transferred into a problem with some fictitious mediums. Because of the localization character, the dense coefficient matrix for single-medium problem is converted into a sparse one for the problem with multiple mediums.

With suitable decomposition of the single dielectric, the resulting coefficient matrix $\boldsymbol{A}$ will become one with much sparsity so that computational speedup is available. With the storage technique of sparse-blocked matrix and iterative equation solvers such as GMRES algorithm, the computing time and memory usage for the original single-medium problem will be greatly reduced. We call this the *quasi-multiple medium* (QMM) approach.

Therefore, the QMM approach includes the following three main points. Firstly, a single dielectric is regarded as a composition of some fictitious mediums. Secondly, a suitable strategy of decomposition is considered to make the resulted BEM coefficient matrix with much sparsity. Lastly, the technique of storing sparse matrix and iterative equation solver are used to benefit from the matrix sparsity.

It should be pointed out that the QMM approach adds some unknowns to the overall problem, which are introduced on the additional fictitious interfaces of quasi-multiple mediums. With suitable decomposition of dielectric regions,

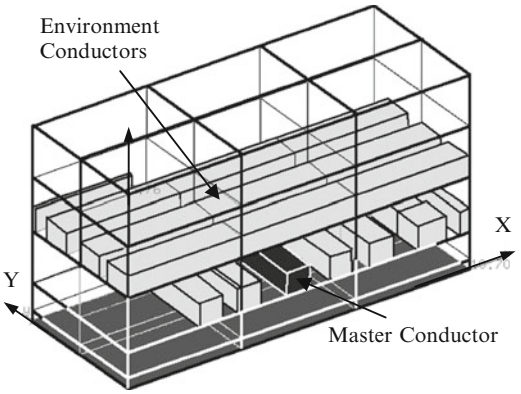

Fig. 4.4 A typical 3-D interconnect capacitor with five dielectrics is cut into 3×2 structures (Reprinted with permission from Yu and Wang [180] © 2005 John Wiley & Sons)

these unknowns would account for a little percentage of total unknowns, since most boundary elements are located on conductor surfaces. So, compared with conventional BEM, the nonzero entries of matrix $\boldsymbol{A}$ are much fewer if applying the QMM approach. Since the Krylov subspace iterative methods are usually used for 3-D capacitance extraction, fewer nonzero matrix entries mean less memory usage and computing time by using sparse matrix data structure. Actual cases of 3-D capacitance extraction verified this analysis.

4.2.2 Decomposition of Dielectrics and Boundary Element Partition

In order to decrease the additional efforts brought by the QMM decomposition, we adopt a simple strategy. Since every dielectric layer is cuboid, and each surface of it parallels to one of the three coordinate planes in the 3-D Cartesian coordinate system, we use two groups of planes parallel to the *YOZ* and *ZOX* planes, respectively, to cut all dielectric layers into pieces (see Fig. 4.4). Thus, in the top view of the 3-D interconnect structure, each original dielectric is decomposed into an array of $m \times n$ fictitious medium blocks. We call (m, n) the *QMM cutting number*.

The conductor distribution of actual 3-D interconnect structure differs in thousands of ways, and the cutting position is not as important as the number Q of fictitious mediums for the QMM's efficiency. So, a strategy of uniformly spacing cutting is adopted. In Fig. 4.4, we show a five-layered interconnect structure to which a 3×2 QMM cutting is performed.

Every dielectric layer is decomposed into $Q = m \times n$ fictitious medium blocks. Neither of small and large values of Q can bring the best speedup of QMM computation. Moderate values of m and n should be chosen. An empirical formula which relates the cutting numbers to the dimensions of the extracted interconnect

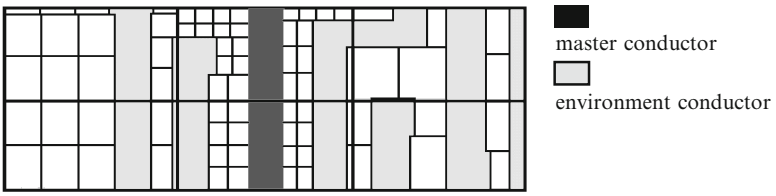

Fig. 4.5 Boundary element partition of one-layer interface in the structure is shown in Fig. 4.4

structure can be obtained from a great deal of calculation for actual interconnect structures. Another approach to automatically determine the QMM cutting numbers can be found in Yu and Wang [179], which selects the optimal value by estimating the overall computational time with the number of nonzero matrix entries.

In applications of BEM, the partition of boundary elements is very important. It affects both speed and accuracy of BEM computation. Here, we adopt a strategy of nonuniform density partitioning. So, we can partition the boundaries into fewer elements without loss of accuracy.

There are two kinds of boundary surfaces in actual interconnect structure. Some surfaces can be treated as trapezoid planes without holes, and the other can be treated as planes with some polygon holes. Using the scan-line algorithm, a surface with holes can be further treated as a composition of smaller trapezoids [59]. Hence, both kinds of boundary surfaces consist of the trapezoids, which are called *mother elements* and are to be further divided into the boundary elements.

According to the electrostatic analysis, the electric field intensity on conductor surfaces, especially that of the master conductor, is often the largest in the simulated region. Besides, the electric field intensity at boundary elements near the master conductor is larger too. So, for each mother element to be partitioned, the mesh number along two directions should be deferent according to its type, position, and size. The larger the estimated electric field intensity, the more densely it should be partitioned.

For the additional fictitious surfaces introduced by QMM, we also use different partition density according to the above electrostatic analysis. For each dielectric layer, the partition density of fictitious surfaces is different. In the dielectric layer containing the master conductor, fictitious surfaces are partitioned more densely. While in the dielectric layers far from the master conductor, the partition density can be much less.

In the QMM-accelerated BEM, the interface of dielectric layer is cut into small pieces, and some fictitious surfaces (which may be surfaces with holes) are produced. So the partition of boundary element becomes more complex than that without QMM acceleration. In Fig. 4.5, the partition of the bottom surface of the dielectric layer in Fig. 4.4, which includes the master conductor, is shown. This complexity brings much difficulty to the more detailed discussion about the influence of the QMM cutting number on the total computing time.

4.2.3 Algorithm Description and Analysis

An interconnect structure cut from the real layout is a stratified structure and has many conductors embedded in M-stratified dielectrics. Once the original dielectric layers are decomposed into fictitious medium blocks with the QMM approach, we use the direct BEM to simulate the new multi-dielectric structure. The major steps of our QMM-accelerated algorithm are listed as follows:

1. Read the data describing a 3-D interconnect structure.
2. Set element-partitioning gaps for each boundary surface.
3. **For** i=1 to M
   Decompose the ith dielectric into fictitious mediums;
   For j=1 to ConductorNumberInLayer[i]
   If the jth conductor **intersects** the interfaces of fictitious mediums
   Decompose conductor j according to the decomposition of dielectric i;
   Set containing relationship of conductor blocks and fictitious medium blocks;
   EndIf
   EndFor
   EndFor
4. Organize medium blocks and conductors blocks into new object lists.
5. Partition all boundary surfaces of the new multi-dielectric structure.
6. Calculate integrals in (4.4) and form the Eq. (4.11).
7. Solve (4.11) with preconditioned GMRES and output the capacitance results.

The QMM approach has been applied in the extraction of actual 3-D multilayered interconnect capacitance. It produces the overall coefficient matrix with much sparsity. The computational time and memory usage of QMM for actual capacitance extraction will be analyzed as follows:

The total computational time for 3-D interconnect capacitance extraction with direct BEM can be expressed as follows:

$$t = t_{\text{gen}} + t_{\text{sol}} + t_{\text{aux}}, \tag{4.12}$$

where t_{gen} is the time spent in generation of the coefficients in (4.11), t_{sol} is the time spent in solution of (4.11), and t_{aux} stands for the time spent in other supplementary procedures, including input of structure data and partition of boundary elements. Generally speaking, the sum of t_{gen} and t_{sol} accounts for more than 90 % of the total CPU time t.

Only nonzero matrix entries need to be computed and stored. So,

$$t_{\text{gen}} \propto Z, \tag{4.13}$$

where Z stands for the number of nonzero entries of matrix $\boldsymbol{A}$. In the phase of equation solution, the main manipulation of each iteration is once matrix-vector multiplication. So, we have

$$t_{\text{sol}} \propto Z \cdot k, \tag{4.14}$$

where k stands for the number of iterations.

For 3-D capacitance extraction, the coefficient matrix $\boldsymbol{A}$ is usually a non-symmetric sparse matrix with a large order, e.g., larger than 1,000. A good preconditioning matrix should also be selected for GMRES algorithm to quicken convergence. Properly organizing the discretized BIEs, the diagonal preconditioner can bring quick convergence to the GMRES solver, whose results will be shown in Sects. 4.4.1, 4.4.2, and 4.4.3. The number of iterations k is much less than the parameter Z in (4.14). Therefore, the number of nonzero matrix entries Z is very significant for the total computing time.

If we ignore the influence of t_{aux} and assume the k does not change much while using the QMM approach, we will find out that the fewer nonzero entries there is, the less CPU time will be taken. In formulation, the speedup ratio of the BEM computation with QMM acceleration is expressed as

$$R_{\text{speedup}} = \frac{t}{t'} \approx \frac{Z}{Z'}, \tag{4.15}$$

where Z and Z' stand for the numbers of nonzero entries of matrix $\boldsymbol{A}$ in the BEM computation without QMM acceleration and that with QMM acceleration, respectively. This expression reveals that the ratio of number of nonzero matrix entries approximately equals to the speedup ratio of the QMM approach. So, when the QMM approach is applied to actual 3-D capacitance extraction, its efficiency is mainly determined by the reduction of the nonzero matrix entries.

In our implementation of BEM computation, the memory usage consists of two main parts. One is the memory needed to store the coefficient matrix, denoted by Mem_A; the other is used to store the orthogonal basis vectors in the GMRES algorithm, denoted by Mem_V. With the technology of storing sparse matrix, we approximately have

$$\text{Mem}_A \propto Z. \tag{4.16}$$

In the GMRES algorithm, a new orthogonal basis vector is constructed in each iterative step. So, we have

$$\text{Mem}_V \propto N \cdot k, \tag{4.17}$$

where N is the number of all unknowns and k is the number of iterations. Since double-precision arithmetic is required for only the work comprising the orthogonalization process, we store the matrix $\boldsymbol{A}$ in single precision and the basis

vectors in double precision. This storing scheme of mixed precision results in less memory storage than the wholly double-precision version while high computational accuracy is preserved [85].

Using the QMM technique, Mem_A is reduced by the same ratio with the reduction of nonzero matrix entries. On the other hand, Mem_V is increased because more unknowns are involved. Usually Mem_A is much larger than Mem_V. So, if the unknowns are not increased much, the total memory usage will be reduced while using the QMM approach. This is verified by actual examples of 3-D capacitance extraction, for which several times of reduction in memory could be found while using the QMM approach. It should also be pointed out that the increase of memory usage would be possible in the case with unsuitable QMM decomposition, where too many fictitious interfaces of quasi-multiple mediums caused a great increase of unknowns.

4.3 Equation Organization and Solving Techniques

In this section, the efficient techniques for generating and solving the linear equation system (4.11) from the QMM-accelerated BEM are proposed. With them, we guarantee the nearly linear relationship between the solution time and the number of nonzero matrix entries (4.14), which is important to the overall efficiency of the QMM technique.

4.3.1 Organization of the Coefficient Matrix

Organization of the coefficient matrix $\boldsymbol{A}$ in multi-dielectric BEM computation involves the sorting order of unknowns and source points and the storage structure. The order of unknowns determines the arrangement of matrix columns, whereas the order of source points determines the arrangement of matrix rows. We make the order of source points consistent with that of unknowns, so that the diagonal entries of the matrix are obtained by the singular integrals. Because the singular integral results in a nonzero entry with larger absolute value, the diagonal preconditioner can bring quick convergence to the GMRES solver.

How to arrange the unknowns or source points, which determine the distribution of nonzero entries in the matrix $\boldsymbol{A}$, is very important for QMM-accelerated BEM. Using the QMM approach, the regions of dielectrics are at least several times more than the original structure without fictitious cutting. For example, a three-dielectric structure contains 12 dielectric regions while 2×2 QMM cutting is applied. If the unknowns were arranged without serious consideration, the nonzero entries would disperse in the coefficient matrix. Also, the nonzero matrix blocks increase much faster than the dielectric regions. This would cause a lot of additional CPU time

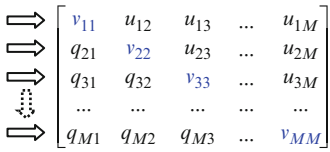

Fig. 4.6 Matrix expression of the unknown order

spent on switching manipulation among matrix blocks and locating of nonzero entries, in each matrix-vector multiplication. The efficiency of the QMM approach would be weakened.

The arrangement of unknowns in multi-region BEM computation is discussed in Fukuda et al. [46] and Kane [76] for the direct equation solver. With reference to them, we propose a matrix expression of the unknown order suitable for arbitrary multi-dielectric structure. By this arrangement, the number of nonzero blocks is deceased to the least, and their distribution is so regular that an efficient storage structure can be easily found to save the additional CPU time.

The matrix expression of the unknown order is introduced below. For the ith dielectric region, unknowns in discretized BIEs can be classified into three types:

1. u on the Dirichlet boundary and q on the Neumann boundary, denoted by v_{ii}
2. u on dielectric interface, denoted by u_{ij} (the jth dielectric shares an interface with dielectric i)
3. q on dielectric interface, denoted by q_{ij} (the meaning of j is the same as that in 2)

Because of the compatibility of u and q along interfaces (see (4.2), u_{ij} and u_{ji} can be represented only by u_{ij} $(i<j)$, while q_{ij} and q_{ji} by q_{ij} $(i>j)$. The order of unknowns follows the rules below and is expressed by a matrix in Fig. 4.6:

- All possible $M \times M$ types of unknowns are arranged in a $M \times M$ matrix (M is the number of dielectric regions).
- Entries on the main diagonal are of the type v, while entries in upper triangle are of type u and lower triangle are of type q.
- Subscript of each matrix entry is the same with its row-column position.
- From left to right in the first row, and so on, row by row (i.e., follow the arrow lines), we get the order of all unknowns.

Using this order of unknowns and the corresponding order of source points, the coefficient matrix $\boldsymbol{A}$ for the two-dielectric problem in Fig. 4.2 is shown in Fig. 4.7a, where the nonzero entries are distributed more regularly than that in Fig. 4.2b. Figure 4.7b shows the nonzero block distribution for a three-dielectric structure applied 2×2 QMM cutting, under our matrix organization. In this case, there are 50 nonzero blocks after merging. While by another arrangement, the number would be 404. It could be proved that our method produces the fewest nonzero blocks in the coefficient matrix. According to the regular distribution of nonzero matrix entries, a length-varied 2-D array is designed to store the coefficient matrix (shown in Fig. 4.8). It has M rows, and the cells in the ith row are one more than the number of interfaces related to dielectric i. Each cell is a MAT_BLOCK

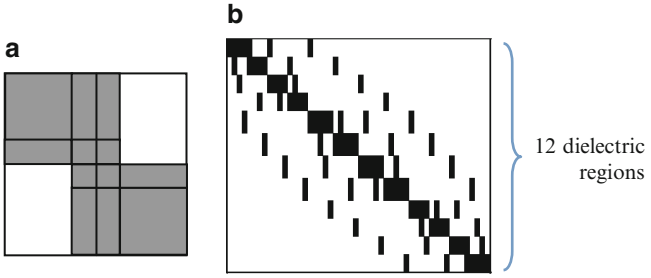

Fig. 4.7 The distribution of the nonzero matrix entries for (**a**) the two-dielectric problem in Fig. 4.2 and (**b**) a three-dielectric structure that applied 2 × 2 QMM cutting

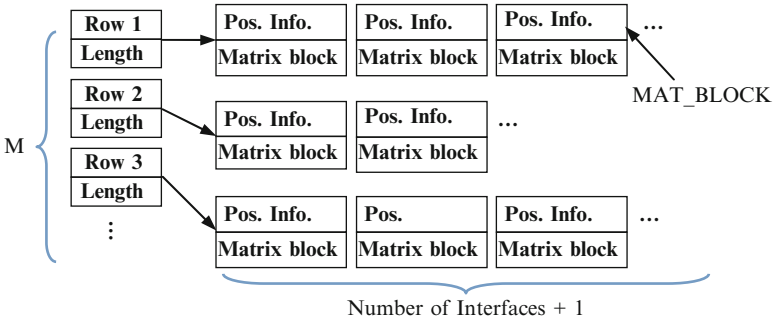

Fig. 4.8 Storing structure of the coefficient matrix

structure, which includes a 2-D array to store a nonzero matrix block and its position information. Experiments reveal that our organization of the coefficient matrix effectively reduces the additional manipulations in the equation solution for QMM-accelerated BEM computation and ensures the nearly linear relationship between the CPU time spent in equation solution and the number of nonzero matrix entries.

4.3.2 Extended Jacobi and MN Preconditioners

The organization of matrix $\boldsymbol{A}$ results in nonzero diagonal entries with larger absolute value. For this reason, the Jacobi (or named diagonal) preconditioner can bring quick convergence to the iterative GMRES solution. In this subsection, we will discuss two easily computed preconditioners which bring even faster convergence than the Jacobi preconditioner, for the 3-D capacitance extraction.

For Eq. (4.11), a preconditioned solution is equivalent to using GMRES to solve $\boldsymbol{APy} = \boldsymbol{f}$ for the unknown vector $\boldsymbol{y}$, from which the original unknown vector $\boldsymbol{x}$ is computed by $\boldsymbol{x} = \boldsymbol{Py}$. This is called the *right preconditioning*. An ideal

preconditioner should firstly well approximate to $\boldsymbol{A}^{-1}$ so that it can improve the condition of the linear system. It should also be easily computed and with great sparsity in order not to increase much computation for constructing and using it in the iterations. To some extent, Vavasis had proposed a good idea for constructing such a preconditioner [150], which is briefly introduced below.

Each row of the preconditioner $\boldsymbol{P}$ is generated separately. Let the ith column of $\boldsymbol{P}^{\mathrm{T}}$ be denoted by $\boldsymbol{p}_i$, i.e., $\boldsymbol{P}^{\mathrm{T}} = (\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_N)$. Ideally, we would like to have

$$\boldsymbol{PA} = \boldsymbol{I} \iff \boldsymbol{A}^{\mathrm{T}} \boldsymbol{p}_i = \boldsymbol{e}_i, \tag{4.18}$$

where $\boldsymbol{e}_i$ is the ith column of the identity matrix. Note that each column (or row) of matrix $\boldsymbol{A}$ corresponds to a discretized unknown (or source point) and further to a boundary element. Therefore, we use the number of row or column as the index of its corresponding source point, unknown, and element. With some strategy we may determine a small list L of indices drawn from $\{1, 2, \ldots, N\}$, where N is the number of unknowns. L denotes the unknowns having the most impact on the current unknown i. Then, (4.18) can be reduced to

$$\overline{\boldsymbol{A}}^{\mathrm{T}} \overline{\boldsymbol{p}}_i = \overline{\boldsymbol{e}}_i, \tag{4.19}$$

where the bars over the variables indicate that all the rows and columns except for those with indices in L are removed. After solving (4.19), we expand $\overline{\boldsymbol{p}}_i$ back to the corresponding entries in the row i of $\boldsymbol{P}$. Repeating the above procedure for all rows, we get the whole sparse matrix $\boldsymbol{P}$.

For example, if the L has three indices, and the first one is the current row i, then Eq. (4.19) will be

$$\begin{bmatrix} a_{l_1 l_1} & a_{l_2 l_1} & a_{l_3 l_1} \\ a_{l_1 l_2} & a_{l_2 l_2} & a_{l_3 l_2} \\ a_{l_1 l_3} & a_{l_2 l_3} & a_{l_3 l_3} \end{bmatrix} \begin{bmatrix} p_{l_1} \\ p_{l_2} \\ p_{l_3} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \tag{4.20}$$

where a_{ij} means the entry of $\boldsymbol{A}$ on the ith row and jth column and $l_1 = i$.

Two strategies for selecting the set L are proposed to construct our preconditioners. The first one is called *extended Jacobi (EJ) preconditioner*. Actually, the Jacobi preconditioner uses the $L = \{i\}$ for each row. However, it does not consider all effect of the singular integrals for a multi-region BEM analysis. For the boundary element Γ_{I} on the interface of medium region i and j, the two unknowns on it are denoted by $u_{ij}(\Gamma_{\mathrm{I}})$ and $q_{ji}(\Gamma_{\mathrm{I}})$. Note that the source point on element Γ_{I} presents twice in the matrix $\boldsymbol{A}$, for the discrete BIE of region i and region j, respectively. Therefore, the singular integral on element Γ_{I} has four positions in matrix $\boldsymbol{A}$. Two of them are not on the main diagonal. For example, a typical structure with three dielectrics and the corresponding sparse matrix $\boldsymbol{A}$ generated with the technique in Sect. 4.3.1 are shown in Fig. 4.9. The small circles in Fig. 4.9b denote the singular integrals not on the matrix main diagonal. The EJ preconditioner is based on the above observation, and let L contain two indices of the row itself and the other

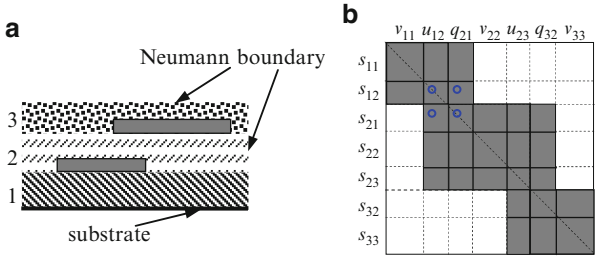

Fig. 4.9 (**a**) Problem with three dielectrics. (**b**) Corresponding coefficient matrix $\boldsymbol{A}$, where the *gray* blocks stand for nonzero entries and the type of discretized variables is labeled beside the matrix columns

occurrence of the source point on the same element, for the row corresponding to an interface element. Otherwise, L only contains the index of the current row. The EJ preconditioner is a little more complex than the Jacobi (for some rows, a 2×2 equation is solved), but it accelerates the convergence remarkably.

In the EJ preconditioner, mno "neighbor" boundary element is considered. To bring more faster convergence to GMRES iteration, a *mesh neighbor* MN(n) preconditioner is proposed, where n stands for the number of neighbor elements. The geometry distance of two elements is not calculated, since the matrix $\boldsymbol{A}$ is stored explicitly and its entry value can be used to judge the neighborhood. For each pair of elements, the maximum absolute value of matrix entries (more than one in a row, if the interface element is involved) representing the interactions between them is called "gravitation" here. Comparing each nonzero entries on row i, the n elements that have the maximum "gravitations" to the current element can be selected. These n elements are then considered as the most neighboring to row i's source element, and the indices of their variables are added to L. Because the index for the current element must be selected and one element may contain two variables, the L has the maximum degree of $2(n + 1)$ in the MN(n) preconditioner.

The difference of our MN(n) preconditioner to other MN-like preconditioners (such as that in Nabors and White [100]) is that we use the explicitly stored matrix entries to judge the neighborhood to avoid the relative complex calculation of 3-D distance. So, our method has less computational expense for a little n and is adapted to the 3-D finite-domain capacitance extraction with multiple dielectrics very well.

Since the number of GRMES iterations is fewer in 3-D capacitance extraction (the relative error norm of 10^{-2} or 10^{-3} is usually set), and the coefficient matrix we have is very sparse, the simplicity of preconditioning is very important. Therefore, some traditional preconditioners such as that using the *incomplete LU decomposition* (ILU) and blocked diagonal preconditioner are out of our consideration. This demonstrates the significance of the proposed preconditioners.

More than 100 structures of VLSI interconnects have been computed. From them we find out that the EJ and MN(1) both have high efficiency. To compare with GMRES solver using the Jacobi preconditioner, the new solvers using these two preconditioners both can reduce 30 % or more computational time. For the problems

with larger order (10^4 or more), the MN(1) preconditioner seems to have better performance.

4.4 Numerical Results

The QMM-accelerated BEM has been developed into a solver called QBEM [114]. In this section, the capacitances of some 3-D interconnect structures are extracted. We firstly compare QBEM with the GIMEI [142] and ODDM [200] in the first two subsections. Then, three large 3-D cases cut from real design are used to depict the speedup brought by the QMM technique. Finally, QBEM is compared with FastCap 2.0 [97]. Raphael is a widely used commercial software [144], including a finite difference solver (RC3) with advanced nonuniform meshing scheme. The result of Raphael under very dense mesh is often used as the golden value by the industry and used to validate the accuracy of QBEM here.

In the following first three subsections, the Jacobi preconditioner is used in QBEM, and the stopping criteria of the GMRES is set to be 1.0×10^{-3}. With the experiments in the last subsection, we evaluate the efficiency of the preconditioning techniques in Sect. 4.3.2.

4.4.1 The Comparison with GIMEI

The test structures are taken from Sun et al. [142] and shown in Fig. 4.10, where a 1×1 cross is immersed in five dielectric layers with a ground plane at the very bottom of the structure. The height of each dielectric layer is 1 μm. Each metal line has the width of 1 μm, and the two lines have the same length z μm. And, they overlapped each other, in the middle of the other line. The lower metal is numbered one, while the higher is numbered two. The dielectric permittivities are all the same. It is worth noting that in Sun et al. [142], the dielectric permittivity given for this structure, 3.9, is impossible. By calculating the structures with FastCap [97, 100] and Raphael, it is found that the permittivity should be 1.0, not 3.9.

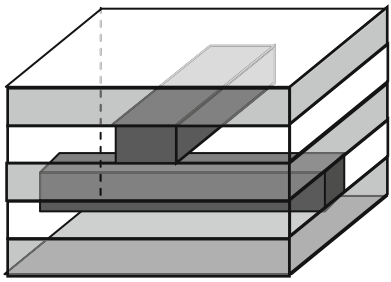

Fig. 4.10 A 1×1 cross over a ground plane

Table 4.1 Capacitance matrix calculated by the GIMEI, FastCap, Raphael, and our method (in unit of 10^{-18} F)

Conductor Length z	C_{11}				C_{22}			
	GIMEI	FastCap	Raphael	QMM	GIMEI	FastCap	Raphael	QMM
4	230	226	232.2	221.1	180.6	176	181.5	176.4
5	260	265	272.1	257.9	203.5	205	208.4	202.5
7	348.7	341.4	348.8	333.1	260.1	253.7	258.6	251.9
10	440.3	451.6	460.2	441.7	326.8	324.2	329.2	326.6

Table 4.2 Comparison of CPU time and memory usage for GIMEI, FastCap, and our method

Conductor Length z	CPU time (s)				Memory (MB)			
	GIMEI	FastCap	QMM	GIMEI/QMM	GIMEI	FastCap	QMM	GIMEI/QMM
4	2.8	24.37	4.98	0.56	3.5	22	0.68	5.1
5	3.23	26.06	5.34	0.60	3.7	24.5	0.68	5.4
7	6.5	65.62	5.61	1.12	5.6	60	0.72	7.8
10	9.16	93.24	6.52	1.40	6.5	78	0.90	7.2

With the line length z taking the values of 4, 5, 7, and 10 μm, the structures are computed by GIMEI, FastCap, Raphael, and QMM-accelerated BEM. The results of GIMEI are obtained from Sun et al. [142]. Because, at the present, our method can only handle problem with finite Neumann boundary, four Neumann boundaries are added far around the crossover. The simulated region defined by the finite Neumann boundaries has a length of 30 μm and a width of 30 μm, and the crossover is placed at its center. This makes the accurate value of capacitance close to that in the infinite region, which is handled by GIMEI. In order to get the capacitance matrix, the program of QMM-accelerated BEM is run twice with two settings of bias voltages. 3×3 QMM cutting is applied. Table 4.1 shows the results of capacitance C_{11}, C_{22} computed by different methods. The discrepancy between the results obtained with our method and other methods is within 5 %. On a SunSparc workstation 20, the CPU time and memory size used by GIMEI, FastCap, and QMM-accelerated BEM are listed in Table 4.2 (because the computing environment of Raphael is different, the data of Raphael are not listed). The CPU time consumed by GIMEI and our method is on the same order, and the memories used by GIMEI are about six times more. With length z increased, the computing time of the GIMEI increases more than two times, whereas that of QMM-accelerated BEM increases only 30 % or so. The QMM-accelerated BEM uses an order of magnitude of less computing time and memory usage than FastCap.

4.4.2 The Comparison with ODDM

The test structure is shown in Fig. 4.11. The size of every straight line is $1 \times 1 \times 13$, and the gap between conductor 3 and 4, as well as conductor 5 and 6, is 3. The

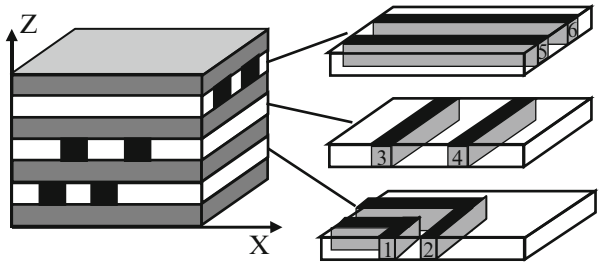

Fig. 4.11 Four crossovers above two bends embedded in seven dielectric layers

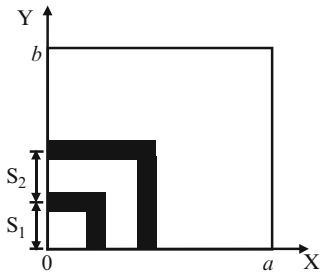

Fig. 4.12 Top view of the layer with bends in Fig. 4.11, $a = b = 13$, $S_1 = 3.5$, $S_2 = 3$

distance between the straight line and the Neumann boundary is 4. The size of the cross section of every bend is 1×1; other geometric parameters of the bends are shown in Fig. 4.12. Counted from the bottom, the thickness of every dielectric layer is 1, 1, 2, 1, 1, 1, and 1. All length parameters above are in unit of μm. The relative permittivity of every layer is 2, 3, 3, 4, 4, 5, and 5.

We have calculated the capacitance matrix by QMM-accelerated BEM, while the corresponding results of SpiceLink and ODDM are provided by Zhu et al. [200]. Only the diagonal entries of the capacitance matrices are listed in Table 4.3. The discrepancy among the results obtained with three methods is within 2 %. In the QMM-accelerated BEM, 3×3 cutting is performed, and the program is run six times with different settings of bias voltages. There are boundary elements from 2,277 to 2,575 in these six computations. On a SunSparc workstation 20, the CPU time and memory size used by these methods are given in Table 4.3. Therefore, the CPU time used by SpiceLink is 20 times more than that used by QMM-accelerated BEM. The CPU time consumed by ODDM is about two times that consumed by our method.

With another experiment where a smaller structure is simulated [181], we found out that when the interconnect structure becomes complicated with embedded conductors increased from two to six, the computing time of the ODDM increases by approximately nine times, whereas that of QMM-accelerated BEM increases only by three times. The time consumed by our method is less than that by

Table 4.3 The diagonal entries of the capacitance matrices calculated with the SpiceLink, ODDM, and our method (in unit of 10^{-18} F)

	C_{11}	C_{22}	C_{33}	C_{44}	C_{55}	C_{66}	Time (s)	Memory (MB)
SpiceLink	0.669	1.29	1.6	1.54	2.53	2.53	1,327	75.9
ODDM	0.68	1.29	1.57	1.52	2.54	2.54	122	2.7
QMM	0.682	1.31	1.6	1.54	2.53	2.53	58.4	3.80

Table 4.4 Comparison between BEM without QMM and BEM with QMM for the number of nonzero matrix entries and iterations

	BEM without QMM		BEM with QMM		
	Nonzero entry	Iteration	Nonzero entry	Iteration	Ratio of nonzero entries
1	13423574	24	1658476	26	8.1
2	18968008	25	3250417	27	5.8
3	26479962	22	3275984	24	8.1

ODDM for the more complicated example. Therefore, the QMM-accelerated BEM is superior to the ODDM in CPU time, especially for fairly large and complex structures. The memory used by QMM-accelerated BEM is very close to that used by ODDM for larger interconnect structure, as shown in Table 4.3.

4.4.3 The Results for Structures from Real Design

We have also compared the BEM without QMM acceleration and BEM with QMM acceleration for three large 3-D examples using five-metal-layer technology. All these examples have conductors distributed from layer 2 to layer 5 and include many crossovers and bends. The first example has 34 pieces of conductors, while the second and the third have 53 and 142 pieces of conductors, respectively. By assigning the QMM cutting number to be (1, 1), we attain the conventional BEM, i.e., BEM without QMM acceleration. The cutting numbers in the QMM-accelerated BEM are different for the three examples. They are (3, 7), (3, 5), and (6, 3), respectively. In these real structures, the master conductors are specified. So, only one setting of bias voltages is used for each example. In Table 4.4, the number of nonzero coefficient matrix entries and GMRES iteration number are listed for these cases, when using QMM acceleration or not.

This experiment is carried out on a Sun Ultra Enterprise 450 server, and the computational results are listed in Table 4.5. The corresponding results of Raphael are also listed for comparison. From the data, we can see that the BEM with QMM is about six times faster than that without QMM. The speedup ratios of BEM with QMM to BEM without QMM are close to the ratios of nonzero entries in Table 4.4. So, the analysis in Sect. 4.2.3 is verified. It also can be found out that the BEM with QMM uses about 1/5 to 1/3 of memory than that BEM without QMM uses. Though the boundary elements and GMRES iteration number increases while using

Table 4.5 Comparison among Raphael, BEM without QMM, and BEM with QMM

	Raphael		BEM without QMM				BEM with QMM					Speedup	
	Cap.	Time	Element	Mem	Cap.	Time	Element	Mem	Cap.	Time	Error (%)	Non-QMM	Raphael
1	20.19	560	5551	54.2	19.52	138.9	6669	13.0	19.69	18.1	0.9	7.7	31
2	31.38	828	7382	76.5	30.97	191.4	8963	21.4	31.24	35.5	0.9	5.4	23
3	27.09	1603	8713	106	26.61	272.5	9928	22.0	27.12	35.1	1.9	7.8	46

Cap.—self-capacitance of the master conductor (unit in 10^{-12} F)
Time—CPU time (unit in second)
Mem—memory usage (unit in Mbyte)

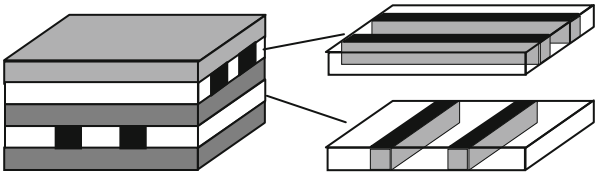

Fig. 4.13 A 2×2 crossover embedded in five dielectric layers

QMM acceleration, the QMM approach greatly reduces CPU time and memory usage of BEM computation. The BEM with QMM acceleration has a large speedup ratio compared to Raphael, which is more than 20 for the three examples, and the discrepancies of capacitance between both methods are within 2 %.

4.4.4 The Comparison with FastCap

The test structures are $k \times k$ bus crossing conductors embedded in five-layered dielectrics ($k = 2$ to 5). The 2×2 bus example is shown in Fig. 4.13. Each bus in the $k \times k$ example is scaled to $1 \times 1 \times (2k + 5)$ (unit in μm). The distance between the conductors in the same layer is 1, and the distance between the Neumann boundary and its neighboring conductor is 3. The thickness and relative permittivity of every layer are 1 and 3.9, respectively. All crossovers have a ground plane at the very bottom. The conductors are numbered from one side to the other side 1, 2 ..., k (bottom layer) and then $k+1$, ..., $2k$ (top layer). The total capacitance of conductor 1 and its couplings with the other conductors are computed; they form a column of the whole capacitance matrix, which is denoted by vector $\boldsymbol{C}_1$ here. These finite-domain and multi-dielectric problems can be easily handled by Raphael and QBEM. In the input of FastCap, 0 is assigned to the permittivity of outer space to handle the finite-domain model, and each interface between dielectric layers is specified to make the comparisons equitable. FastCap with default expansion order 2 is denoted by FastCap(2), while a faster program FastCap(1) has the expansion order 1. Besides, to make FastCap only compute the capacitances related with conductor 1, the "-rs" option is used to remove other conductors from solving [97]. The following experiments are carried on a SUN Ultra Enterprise 450 server with UltraSparc II processors at 248 MHz.

The above crossover problems are computed by Raphael with 0.25×10^6 grid points and 10^6 grid points, FastCap(1), FastCap(2), and QBEM. Our GMRES solver reduces the 2-norm of the residual to 1 % of the initial residual, the same condition used in FastCap. The number of panels per edge for each conductor is specified individually, so as to make FastCap compute a linear equation system with similar order as that in QBEM. Using the result $\boldsymbol{C}_1$ of Raphael with 10^6 grids as standard,

Table 4.6 Comparisons of FastCap, Raphael, and QBEM for the crossover problem

Test problems	2×2	3×3	4×4	5×5
FastCap(1)				
Time (s)	7.9	9.2	10.0	12.5
Memory (MB)	17.9	17.9	19.1	23.7
Panel	1080	1284	1487	1804
Error	1.6 %	2.1 %	3.4 %	2.9 %
FastCap(2)				
Time (s)	11.5	15.1	17.5	24.3
Memory (MB)	26.4	28.4	30.7	38.5
Panel	1080	1284	1487	1804
Error	2.1 %	2.3 %	2.6 %	3.0 %
Raphael (0.25M grids)				
Time (s)	78.8	67.1	88.9	81.9
Memory (MB)	47	45	48	48
Error	0.3 %	0.4 %	0.5 %	0.8 %
QBEM				
Time (s)	1.0	1.3	1.6	1.5
Memory (MB)	1.7	2.7	2.1	2.1
Panel[a]	1184	1431	1502	1558
Error	2.7 %	2.5 %	1.0 %	1.2 %
Sp. to FC1[b]	8	9	6	8
Mem_R to FC1[c]	11	7	9	11
Sp. to FC2[b]	12	12	11	16
Mem_R to FC2[c]	16	11	15	18

[a]The panels on the interface between the fictitious medium blocks are not counted
[b]Sp. to FC1/2 means the speedup ratio to FastCap(1/2)
[c]Mem_R to FC1/2 means the memory reduction compared with FastCap(1/2)

the error of capacitance vector $\boldsymbol{C}_1^{'}$ computed by another program is estimated in the 2-norm: $\|\boldsymbol{C}_1^{'} - \boldsymbol{C}_1\|/\|\boldsymbol{C}_1\|$.

Table 4.6 compares the QBEM solver, FastCap, and Raphael. The following is a summary of the comparison:

1. Using the Raphael's result under 1M grids as criterion, the errors of FastCap(1), FastCap(2), and QBEM are all within 3 %. The error of our method is relatively small.
2. The FastCap uses almost the same (even less) number of panels as our QBEM and also uses the nonuniform partition (dense near the master). So, under the same-scale discretization, the speedup of QBEM to FastCap(2) is from 12 to 16 and 6 to 9 to FastCap(1).
3. The QBEM uses 1/18 to 1/11 of the memory used by FastCap(2). Compared with FastCap(1), the QBEM's memory usage is from 1/11 to 1/7.
4. Compared with Raphael of 0.25M grids, QBEM has over 55X runtime improvement and 17X memory saving.

In the above computations with QBEM solver, the EJ preconditioner is used. The QMM cutting numbers are (4, 4), (5, 5), (3, 3), and (3, 3), respectively, for the four crossover cases. The detailed results for the 4×4 crossover are listed in Table 4.7.

For the 4×4 crossover problem, different preconditioners discussed in Sect. 4.3.2 are used for comparison. Related data are listed in Table 4.8. From it we can see that MN(1) and MN(2) consume much time in constructing and using the preconditioner for the problem (with 2,435 variables), so the reduction of iterative number does not bring effective speedup of equation solution. The EJ preconditioner is a little more complex than the Jacobi, but it reduces six steps in iteration. Therefore, it achieves the least computational time of equation solution, which is about 30 % less than that using the Jacobi preconditioner. More experiments have also shown that the iteration number decreases gradually for preconditioners in order: Jacobi, EJ, MN(1), and MN(2). And the EJ or MN(1) has the best overall performance, achieving much faster equation solution than the Jacobi. More efficiency validation of the EJ and MN(1) preconditioners can be found in Yu et al. [182].

The QBEM solver and FastCap are all of the boundary integral method. So, for a same finite-domain multi-dielectric problem, N_p boundary elements not including that on the fictitious medium interfaces will guarantee the same accuracy for both methods. The FastCap utilizes the boundary element method of the total-charge Green's function, which produces a dense matrix with N_p^2 nonzero entries. With the multipole approach, not all matrix entries need to be computed, and the matrix-vector multiplication is accelerated. The direct BEM used in the QBEM solver has the character of resulting in a sparse matrix for a multi-region problem. With the QMM approach, the degree of the matrix is a little more than N_p (adding elements on the fictitious interfaces), but the sparsity is greatly enlarged. So, the nonzero matrix entries are much less than N_p^2, and the matrix-vector multiplication is also accelerated. Since both methods have almost the same number of iterative steps, the QMM approach has shown the same or better efficiency than the multipole approach on matrix sparsification. Furthermore, the careful processing of the integrals and the characters of direct BEM make the equation forming fast and matrix-vector multiplication more convenient than FastCap, which has much auxiliary cost on the cube partition and multipole expansion (see Table 4.7). With above analysis and experiment results, we can see the QBEM solver is superior to the multipole accelerated BEM for the actual finite-domain capacitance extraction.

4.5 Efficient Techniques for Handling Floating Metal Fills

The *chemical-mechanical polishing* (CMP) is a necessary manufacturing step for VLSI circuit, by which the wafer is polished with a rotating pad and slurry to achieve the planarized surfaces [74, 139]. Because the dielectric thickness strongly

Table 4.7 Comparison for the 4×4 crossover problem (capacitance in unit of 10^{-12} F, time in unit of second, and memory in unit of MB)

	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}	C_{16}	C_{17}	C_{18}	Panel	Ele_N	Iter	Mem	Tgen	Tsol	Time
QBEM	2422	−620	−8.7	−2.0	−224	−143	−143	−224	1502	1896	11	2.1	1.02	0.29	1.6
FastCap(2)	2352	−573	−4.1	−3.2	−214	−143	−143	−214	1487	1487	9	30.7	13.4	4.0	17.5
FastCap(1)	2345	−547	5.6	2.3	−220	−142	−145	−219	1487	1487	13	19.1	6.9	2.9	10.0
Raphael(1M)	2408	−601	−9.2	−1.9	−224	−150	−150	−224	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Panel means the number of "true" elements not including that on the fictitious interfaces; Ele_N means the number of all elements including those on the fictitious interfaces; Iter is the number of iterative steps; Mem is the memory usage; Tgen is the time for forming the linear equation system; Tsol is the time for equation solution; Time is the total time of capacitance extraction

Table 4.8 Comparison of four preconditioners for the 4×4 crossover problem

Jacobi		EJ		MN(1)		MN(2)	
Iter	Tsol(s)	Iter	Tsol(s)	Iter	Tsol(s)	Iter	Tsol(s)
17	0.42	11	0.29	11	0.44	10	0.52

depends on the pattern density of underlying metal layer, a widely used method for reducing the variation of dielectric thickness in CMP process is to insert dummy metals (this procedure is also called "area fill") [30, 74, 138, 139]. These dummies are situated between signal lines to increase the pattern density and at the same time influence the electric characteristics of interconnects in different ways depending on whether they are grounded or on a floating state. In the *application-specific integrated circuit* (ASIC) design, the floating dummy-fills are preferred due to the short design period and considerable area to be filled. Such floating dummy-fills have a strong impact on interconnect capacitance and therefore signal delay and cross talk [74, 138, 139].

Nowadays, the area fill synthesis considering the impact on circuit performance has become an important problem of the *design for manufacturability* (DFM) [30]. For the electric characterization or optimization of design rules for the dummy-fills, a huge number of simulations are required for the structures involving floating dummy-fills. For the case with floating conductors, a conventional field solver (such as Raphael) treats them as normal electrodes and extracts the full capacitance matrix. Then, the capacitance matrix is reduced by considering that the total charge on each floating conductor is equal to zero [144]. Obviously, the cost of CPU time for extracting the full capacitance matrix is prohibitive while including a lot of floating dummies.

In Park et al. [109] and Cueto et al. [35], two algorithms of capacitance extraction were proposed with specific consideration of the floating dummy-fills. The algorithm in Cueto et al. [35] is based on a so-called fictitious domain method, which uses the Lagrange multiplier λ to consider the conductor boundary and replaces the complex shape domain of potential computation with a simple one. However, no other algorithm was compared with the algorithm based on fictitious domain method, and the computational results in Cueto et al. [35] did not show high efficiency (it costs 1 h for a structure including 130 floating metals). The algorithm in Park et al. [109] is based on the finite difference method (implemented in a solver called PASCAL [84]).

In this section, we extend the QMM-accelerated BEM to efficiently handle the structure with floating dummy-fills. Our method is similar to that in Park et al. [109], but based on the direct BEM. We modify the direct BEM equations and propose efficient solution technique for handling the metal fills. Numerical experiments show a significant advantage of our technique over Raphael and PASCAL.

4.5.1 Basic Idea

If there is no floating dummy in the interconnect structure for capacitance extraction, potential u is known on all conductor surfaces. This boundary condition is utilized to form (4.11). After solving (4.11), the self-capacitances and coupling capacitances of the master conductor can be evaluated by the integral of the normal electric field intensity q on the conductor surfaces.

If some conductors in the simulated structure are changed to be floating (without setting known voltage), (4.11) cannot be solved because there are more unknowns (potential u on these floating conductors) than equations. So, our task is to supply additional equations about the floating dummies and make the enlarged system of linear equations solvable. Our method of capacitance extraction for the structure involving floating dummies includes the following steps:

1. Set the jth interconnect conductor to 1 V and the rest interconnects to 0 V.
2. For each dielectric region, discretize its boundary (including dummy surface, since it is also a part of boundary) and formulate the discretized BIEs.
3. Put the discretized BIEs for all regions together utilizing (4.2), and substitute the known boundary conditions on conductor surface and the outer boundary.
4. Supply some equations about the floating dummies so that the total number of equations is equal to the number of unknowns (including the additional u unknowns of dummies).
5. Solve the generated system of linear equations, and get the charges on interconnect conductors which equal to capacitances.
6. Repeat steps 1–5 with different voltage settings to get all capacitances among the interconnects.

Now, we discuss how to supply the equations about the floating dummies. To calculate the total charge of a floating dummy, we have

$$\int_{\Gamma_f} \sigma \cdot \mathrm{d}\Gamma = \int_{\Gamma_f} \varepsilon \cdot q \cdot \mathrm{d}\Gamma = \overline{Q}_f, \tag{4.21}$$

where Γ_f is the surface of the dummy conductor and σ is the surface charge density. ε is the permittivity of the dielectric surrounding surface Γ_f, and q is the normal electric field intensity. Here, the charge $\overline{Q}_f$ is usually zero for each dummy as its initial electric state. Since the dummy surface is discretized into elements, (4.21) is actually used with its discretized form which involves discretized q unknowns on dummy surface.

If there are N_f floating dummies, we now have N_f new equations of (4.21), one for each dummy. On the other hand, as we have discussed there are also N_f additional u unknowns of dummies, one for each dummy (because of the equipotential property). So, the extended equation system from (4.11) can be solved, and we then obtain the interconnect capacitances.

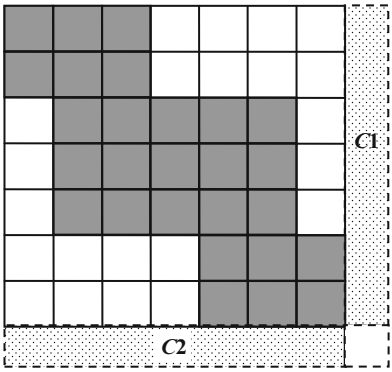

Fig. 4.14 Matrix population corresponding to the structure in Fig. 4.9 and the regions with *dashed-line* contour represent the added entries when considering the floating conductors

4.5.2 Equation Formation and Solution

As discussed above, more unknowns and equations are introduced for the case involving floating dummy-fills. Putting the equations (4.21) and the unknowns of potential on floating dummies at the end of (4.11), we have the new linear system with the form:

$$\begin{bmatrix} \boldsymbol{A} & \boldsymbol{C}1 \\ \boldsymbol{C}2 & \boldsymbol{O} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{u}_f \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \overline{\boldsymbol{q}} \end{bmatrix}, \quad \text{i.e.,} \quad \boldsymbol{A'x'} = \boldsymbol{f'}, \tag{4.22}$$

where $\boldsymbol{A}$, $\boldsymbol{x}$, and $\boldsymbol{f}$ are obtained from the linear system (4.11) with the floating character of dummies not considered. $\boldsymbol{u}_f$ is the vector consisting of the unknown potential of dummies, and $\overline{\boldsymbol{q}}$ is made up of the right-hand side of (4.21) (actually a zero vector). $\boldsymbol{C}1$ and $\boldsymbol{C}2$ are corresponding sub-matrices of coefficients, and $\boldsymbol{O}$ is a zero sub-matrix. For the problem with three dielectrics in Fig. 4.9, if some conductors are changed to be floating, the modification of the overall coefficient matrix can be illustrated as the regions with dashed-line contour in Fig. 4.14.

Below we give the details of calculating the sub-matrices $\boldsymbol{C}1$ and $\boldsymbol{C}2$, for a problem involving multi-dielectric regions and multiple floating dummies. $\boldsymbol{C}1$ corresponds to the coefficients of the new unknowns u_f and can be expressed as

$$(\boldsymbol{C}1)_{\boldsymbol{ij}} = \int_{\partial\Omega_{f,j}^{(i)}} q_{(i)}^{*} \mathrm{d}\Gamma, \tag{4.23}$$

where $q_{(i)}^{*}$ means the normal derivative of u^* in (4.3), which is related with the ith collocation point. $\partial\Omega_{f,j}^{(i)}$ means the part of surface of the jth floating dummy, which is within the same dielectric region with the collocation point i. $\boldsymbol{C}2$ corresponds to the coefficients in (4.21) and can be expressed as

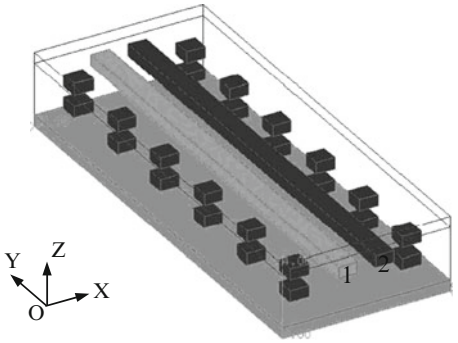

Fig. 4.15 Two conductor lines surrounded with 24 square floating dummies located at two layers

$$(\boldsymbol{C}2)_{ij} = \begin{cases} \int_{\Gamma_j} \varepsilon \cdot \mathrm{d}\Gamma, & \text{when column } j \text{ corresponds to a } q \text{ unknown on dummy } i \\ 0, & \text{otherwise,} \end{cases} \tag{4.24}$$

where Γ_j stands for the jth boundary element and ε is the permittivity of the surrounding dielectric.

For capacitance extraction without floating dummies, the extended Jacobi (EJ) preconditioner for GMRES algorithm is proposed in Sect. 4.3.2. It is an approximation to $\boldsymbol{A}^{-1}$ and makes the convergence rate at least 30 % faster than using the Jacobi preconditioner while introducing little overhead. However, since there is a zero diagonal block in $\boldsymbol{A}'$, the EJ preconditioner cannot be used directly. Assume an EJ preconditioning matrix $\boldsymbol{P}$ is constructed for the original coefficient matrix $\boldsymbol{A}$, then we construct the new preconditioner as follows:

$$\boldsymbol{P}' = \begin{bmatrix} \boldsymbol{P} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{I} \end{bmatrix}, \tag{4.25}$$

where $\boldsymbol{I}$ stands for the identity matrix. Since the dimension of $\boldsymbol{I}$ (equal to the number of dummies) is much smaller than the total number of discretized unknowns, $\boldsymbol{P}'$ still approximates $(\boldsymbol{A}')^{-1}$ to some extent. Therefore, the new preconditioner should also improve the rate of convergence for the GMRE equation solver. Numerical experiments verified this analysis.

4.5.3 Numerical Results

The techniques have been implemented in QBEM. Some typical structures of floating dummy-fills are calculated to verify their efficiency. All experiments in this subsection are run on a Sun Ultra V880 server with 750 MHz frequency.

Three test cases are shown in Figs. 4.15, 4.16, and 4.17, respectively. The first one involves two parallel conductor lines surrounded with 24 square floating dummies

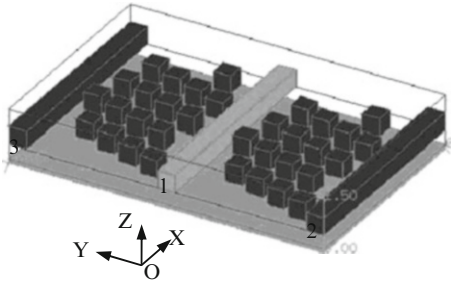

Fig. 4.16 Three conductor lines and 34 floating dummies of the dot-array type

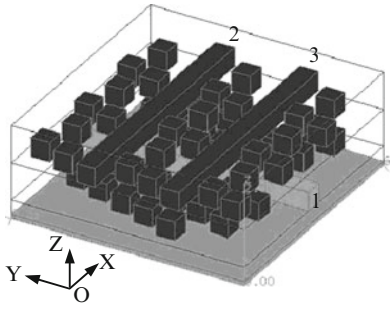

Fig. 4.17 Three conductor lines and 53 floating dummies of the dot-array type located at two layers

located at two layers. The size of each conductor lines is $1 \times 23 \times 0.6$. The size of each dummy metal is $1 \times 1 \times 0.6$, and they all are within a window of size $9 \times 23 \times 4$, above a grounded substrate. The second one is a typical structure of floating dummy-fills, which are arranged as a dot array in the oblique alignment and is often adopted for minimizing changes in the coupling capacitances and maximizing the uniformity of the pattern density [74, 109]. The size of each parallel conductor line is $6 \times 0.5 \times 0.5$. The size of each dummy metal is $0.5 \times 0.5 \times 0.5$, and they are all within a window of size $6 \times 9 \times 1.5$. In case 3 (Fig. 4.17), the dummy insertion rule is similar to that in the second case (Fig. 4.16), as dot array in the oblique alignment. Different from case 2, the conductor lines and dummies are located at two layers, and the former constructs a 1×2 crossover. In this case, two conductor lines in the same layer are of size $6.5 \times 0.5 \times 0.5$, while the third one is of $0.5 \times 6.25 \times 0.5$. The size of each dummy is $0.5 \times 0.5 \times 0.5$, and all are within a window of size $6.5 \times 6.25 \times 2.5$. All length parameters above are in unit of μm. The relative permittivities of layers in the same case are different. They have values of 1.9, 2.9, and 3.9.

In each of the three cases, conductor 1 is set to be the master conductor, and its self-capacitance and coupling capacitances with other conductors are computed using Raphael and our method. In the computation with Raphael, the dummy-related capacitances are calculated by setting each dummy as master conductor one after another, and the wanted capacitances are outputted after performing circuit reduction [144]. In the computations with our method, 2×1, 1×3, and 4×2 QMM cuttings are applied, respectively, for the three cases. The capacitance results and other detailed data for these three cases are listed in Table 4.9.

Table 4.9 Comparison of computational results with our method and Raphael (capacitance in unit of 10^{-18} F)

Case	Method	C_{11}	C_{12}	C_{13}	Ele_N	Var_N	Iter	Time (s)	Memory (MB)	Speedup
1	QBEM	2446	−1091	N/A	654	792	37	1.1	1.7	299
	Raphael	2470	−1088	N/A	[a]85×10^3		N/A	328.4	18	N/A
2	QBEM	766.1	−11.4	−10.4	1,006	1172	55	1.9	2.7	1,330
	Raphael	753.2	−11.5	−11.5	[a]281×10^3		N/A	2,527	54	N/A
3	QBEM	732.1	−161.1	−141.5	1,789	2501	52	3.0	5.9	2,311
	Raphael	741.6	−163.0	−161.4	[a]385×10^3		N/A	6,932	72	N/A

[a]The value is the number of default grids generated by Raphael automatically

Table 4.10 Comparison of computational time of our method and the conventional method for case 1

	Extraction time related to (s)			Total time (s)	Speedup over master extraction	Speedup over matrix extraction
	Conductor 1	Conductor 2	24 dummies			
Conventional method	0.84	0.84	25.07	26.75	1	1
Our method	1.13	1.13	N/A	2.26	**22.9**	**11.8**

From Table 4.9, we can see that the errors of capacitances computed with our method are all within 3 % (using Raphael's result as criterion), while the speedup ratio to Raphael ranges from three hundred to several thousands. Our method uses less than 1/10 of the memory used by Raphael. It should also be pointed out that the electric potential of dummy metal is also calculated and it is very close to Raphael's result as well (with discrepancy within 5 %). The number of unknowns in our method is also listed in Table 4.9, which is larger than the element number because each interfacial element has two unknowns.

To demonstrate the efficiency of the new preconditioner proposed in last subsection, the problem is also solved without a preconditioner and with a modified Jacobi preconditioner. Experiment results show that without a preconditioner, the GMRES algorithm cannot converge within 300 steps for the last two problems and converges with 230 steps for the first problem. While using the modified Jacobi preconditioner, the iterative numbers for the three examples are 39, 62, and 62, respectively. Here the modified Jacobi preconditioner is constructed also as (4.25). So, it is clear that the modified EJ preconditioner performs very well for the capacitance extraction involving floating dummies.

To demonstrate the advantage of our method over the conventional one, the whole capacitance matrix of case 1 is calculated using both methods. With our method, QBEM ran twice setting the two signal lines as master, respectively. With the conventional method, QBEM without the presented modification ran totally 26 times setting all conductors including dummies as master, one for each time. The details of computational time are listed in Table 4.10. Without considering the time to eliminate the floating nodes, we found out the speedup ratio of our method is 22.9 and 11.8, for single-master extraction and whole-matrix extraction, respectively.

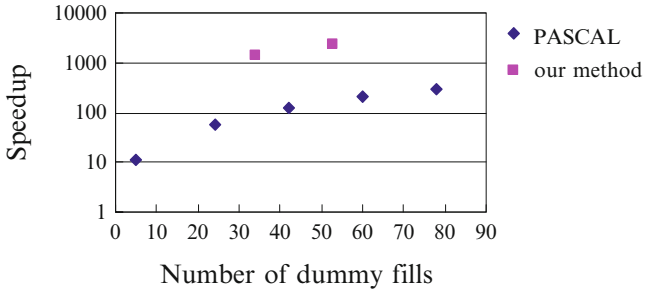

Fig. 4.18 The speedup ratio of PASCAL and our method to Raphael for structures involving different number of dummies

The conventional method treats the floating dummies as normal electrodes, which firstly extracts the capacitances related with these dummies, and then reduces them to get capacitances of interconnect. For a structure involving N_c interconnect (signal) conductors and N_f floating dummies, the conventional method invokes the 3-D field solver for $(N_f + 1)$ times to calculate the capacitances related with the master conductor. In contrast, the 3-D field solver is invoked only once in our method, although the degree of the linear system to be solved increases for N_f due to the additional unknowns of dummy potential. Usually in the field solution, the number of discretized unknowns is much larger than N_f. So the increased computational expense of our method in once field solution is very little. Ignoring both the increased expense of our method and the time of network reduction in a straightforward method, the speedup of our method can be expected to be $(\boldsymbol{N_f} + 1)$ for the capacitance extraction with one master conductor. To calculate the whole capacitance matrix of the interconnects, the 3-D field solver needs to be invoked for $(N_f + N_c)$ times in the conventional method, while in our method only N_c times are needed. Similarly, the speedup ratio of our method is about $(N_c + N_f)/N_c$ when extracting the whole capacitance matrix. For case 1, the expected speedup ratios for the single-master extraction and matrix extraction are 25 and 13, respectively. This is consistent with the results shown in Table 4.10.

In Park et al. [109], a fast method to extract interconnect capacitance considering the floating dummy-fills was implemented as software PASCAL, whose performance was also compared with Raphael. They used several cases of dummies located as dot array in the oblique alignment (similar to those in Figs. 4.16 and 4.17) to make comparison, which showed that PASCAL has a speedup ratio varied from 11 to 290 (see Table 2 in [109]), as the number of dummy metals increases. Since we cannot obtain the PASCAL for direct comparison in the same computational environment, only a relative comparison is carried out based on the information provided by Park et al. [109]. Figure 4.18 shows the speedup ratios of two methods to Raphael for similar structures involving oblique alignment of dot dummies. From it we can expect that our method is about ten times as fast as the PASCAL.

4.6 Summary

In this chapter, efficient techniques based on direct BEM are presented for the problem of capacitance extraction with multiple dielectrics. The major idea is called quasi-multiple medium (QMM), which enables a largely sparse linear equation system to be generated. Along with efficient formulation and solution approaches, i.e., preconditioned GMRES iterative solver, the enhanced direct BEM solver exhibits large advantages over other capacitance solvers based on FDM, indirect BEM, and semi-analytical approaches. Besides, the BEM solver is extended to handle the floating dummy-fills in actual IC layouts by introducing a floating condition into the direct BIE and adopting an efficient preconditioning technique.

Except that the QMM cutting is empirically determined, an optimal approach can also be employed to maximize the sparsity of generated coefficient matrix [179]. The test cases presented are all with multilayered dielectrics. Actually, the presented method is able to handle more general dielectric configurations, including complex conformal dielectrics and embedded air gaps [183]. These functions have been implemented in QBEM [114].

Chapter 5
Resistance Extraction of Complex 3-D Interconnects

In the deep submicron process technology, multiple metal layer technology is widely used. The integrity and also the clock frequency of the VLSI circuits increase quickly. Besides capacitance extraction, fast and accurate extraction of interconnect resistance is also important for VLSI designs [143]. The interconnect resistance affects the timing, power consumption, and signal integrity, among others.

One-dimensional (1-D) and two-dimensional (2-D) extraction methods for interconnect resistance have been developed. They include heuristic method [62], which is not suitable for handling multiterminal-resistive regions today, and numerical methods based on solving the Laplace equation. The numerical methods include finite difference method (FDM) [52], finite element method (FEM) [96], and boundary element method (BEM) [143, 163]. BEM has the advantages of fewer variables, better accuracy, and higher ability to deal with complicated interconnect structures. Therefore, it has received considerable attention [143, 181]. Also note that BEM is only in 2-D in Sun et al. [143] and Wang and Wu [163].

Three-dimensional (3-D) extraction methods become more and more necessary for realistic VLSI circuits nowadays. For example, via layers are widely adopted to connect different layers. Figure 5.1 shows multiple via layers. The existence of these layers makes the 3-D effects not negligible anymore. For these cases, it will be hard for 2-D extraction approaches to get accurate resistance data. Considering the circuits and technology are still developing so quickly, the importance of 3-D extraction will increase along time.

This chapter presents efficient extraction methods based on 3-D BEM. Multiple techniques are proposed to improve the efficiency, including sparsifying linear system, reducing system size by adopting various boundary elements, and combining numerical methods with analytical formulations. The improvements are based on physical observations and numerical experiments, so they keep high accuracy during the process of pursuing high speed. This chapter is organized as follows. Section 5.1 describes the basics of interconnect resistance calculation. Section 5.2

W. Yu and X. Wang, *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*, DOI 10.1007/978-3-642-54298-5_5,

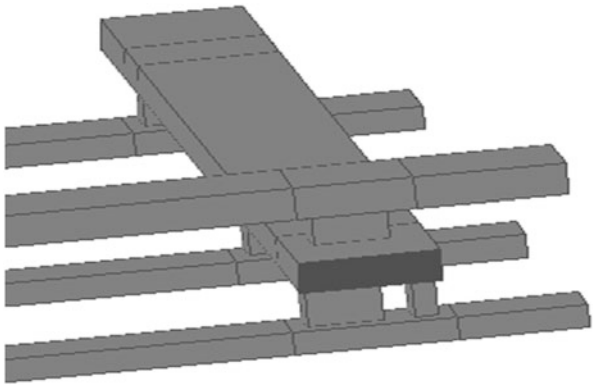

Fig. 5.1 3-D view of a part of realistic circuit design

formulates the field-solver techniques based on BEM. It's followed by improvement techniques. Section 5.4 combines analytical formulation with BEM extraction. Numerical experiments will demonstrate the efficiency and accuracy terms.

5.1 Analytical Resistance Formulation

The resistance of a straight conductor, which has two parallel ports of the same size, is very easy to be calculated with the analytical resistance formula:

$$R = \rho \frac{l}{S} = \frac{l}{\sigma S},$$

where l is the length along the current direction, S is the cross-sectional area, and σ is the conductivity (see Fig. 5.2).

Note that this equation only fits for such conductors. For real 3-D interconnect structures, their equivalent resistance network can be obtained using field-solver techniques including direct BEM.

5.2 Field Solver for Interconnect Resistance

For simplicity, terminals of resistive regions, including ports and contacts, are all called *ports*, and only cuboids (parallel to x or y axis) are here. Multiterminal interconnect structures have equivalent resistance network, where there is a resistor between each pair of ports.

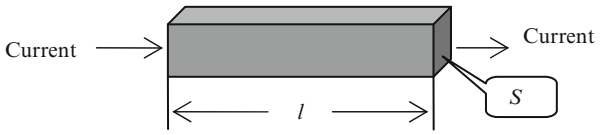

Fig. 5.2 A straight conductor wire

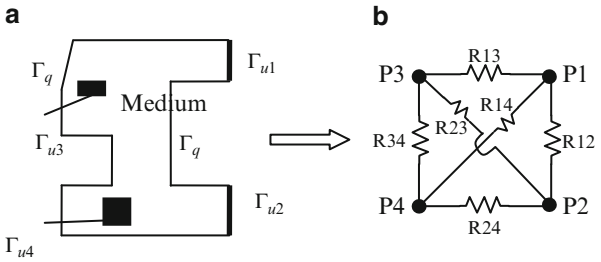

Fig. 5.3 A resistance region (**a**) and its resistance network (**b**)

5.2.1 Resistance Network of Multiterminal Regions

A 2-D resistance region and its equivalent full-graph resistance network are shown in Fig. 5.3, where Γ_u denotes port surfaces. To get all resistors, let the voltage of port j (called main port) be 1 V and voltages of the other ports be 0 V. Then the resistor, R_{jk}, between port j and port k is

$$R_{jk} = \frac{\Delta V_{jk}}{I_k} = \frac{V_j - V_k}{I_k} = \frac{1}{I_k}, \tag{5.1}$$

and the I_k in (5.1), the current flowing into port k, is [163]

$$I_k = \int_{\Gamma_k} \sigma \cdot \frac{\partial u}{\partial \boldsymbol{n}} \mathrm{d}\Gamma, \tag{5.2}$$

where Γ_k is the surface of port k, σ is the conductivity, u is the electric potential, and $\boldsymbol{n}$ is the unit normal vector of the region boundary. The computation above repeats with different main port every time, and the resistance network of the region is obtained. It is shown in the following text that $\partial u/\partial \boldsymbol{n}$ can be obtained using direct boundary element method and in turn the resistance network becomes available through (5.2) and (5.1).

5.2.2 Resistance Calculation Using Direct BEM

Generally speaking, to get the electric field distribution of a multiterminal-resistance region needs solving the Laplace equation in the region [143, 163],

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = 0, \tag{5.3}$$

with the two mixed boundary conditions of

$$u = u_0 \quad (\text{on } \Gamma_u) \tag{5.4}$$

and

$$q = q_0 \quad \left(\text{on } \Gamma_q\right), \tag{5.5}$$

where u is the potential, Γ_u is port surfaces, q is $\partial u/\partial \boldsymbol{n}$, Γ_q is the surfaces of the conductors in the region excluding Γ_u, and the boundary $\Gamma = \Gamma_u + \Gamma_q$. The application of direct boundary element method transforms the Eq. (5.3) to a boundary integral equation [24, 163]:

$$c_s u_s + \int_\Gamma q^* u \mathrm{d}\Gamma = \int_\Gamma u^* q \mathrm{d}\Gamma, \tag{5.6}$$

where u_s is the electric potential at the source point s, c_s is a constant depending on the boundary geometry at the point s, u^* is the fundamental solution of the Laplace equation, and q^* is $\partial u^*/\partial \boldsymbol{n}$. Dividing the boundary of the region into elements and substituting boundary conditions (5.4) and (5.5), we get a linear system [24, 163]:

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{f}, \tag{5.7}$$

where $\boldsymbol{x}$ includes discrete u and q variables on the boundary. Solving the linear system (5.7), we can get the resistance network through (5.2) and (5.1).

The technology of quasi-multiple medium in Chap. 4 is applicable to make the system (5.7) more sparse and thus to accelerate its solution in iterative solvers. Computing complexity of this quasi-multiple medium BEM (QBEM) is between $O(N)$ and $O(N^2)$ generally [178], where N is the number of elements. Therefore, if N is reduced, then the extraction will be faster. The following text will try to reduce it in multiple ways.

5.3 Fast BEM Solver Using Linear Boundary Elements

Although BEM itself has built-in advantages like few variables, it still fails to meet computational resource requirement of CPU time and memory when solving large problems. Multiple techniques can expedite the solution of (5.7). They will

be described in details, and their individual and total effects can be demonstrated by multiple experiments. The original and improved QBEMs are compared with a commercial software tool Raphael [144] (from Synopsys Inc., with FDM), as well as direct boundary element method (DBEM, without quasi-multiple medium technique applied). Two test cases are sram.qbem and clk.qbem, which come from realistic circuits. They have one and two via layers, respectively. All programs run on Sun Ultra Enterprise 450 (CPU frequency 248 MHz, memory 4G), and the comparison between conductance matrices is listed in Tables 5.1, 5.2, 5.3, and 5.4. If not specialized, QBEM in the first three tables is only with one kind of improvement, and QBEM in the last table is with all. In these tables, the element number of Raphael is its default grid number; the running time of Raphael is also its default running time. The running time of QBEM is the unit for the "time ratio" column, and it's similarly true for "mem ratio" (memory ratio) and "ele ratio" (element ratio) columns. Max error means the maximum of the absolute diagonal values in relative error matrices where the references are from Raphael with much denser meshes (2.5M/2M grids for the first and second cases, respectively).

5.3.1 Physics-Based Nonuniform Virtual Cutting

When QMM is applied in capacitance extraction, the region is virtually cut [178, 181] evenly. For resistance extraction, cutting averagely may divide the wire interface into multiple parts, introducing more variables, which is bad for speed. Figure 5.4 is an example, where the thin lines are cutting traces.

More can be done for nonuniform cutting. It's well known that when the voltages of interconnect conductors are of low frequency, say, direct current, the current flowing through corner sections is not uniform, but it becomes nearly uniform some distance away from the corners [155]. Accordingly, a new cutting method (called WHOLE, which means it preserves the interfaces and corners) is shown in Fig. 5.5. The only cutting lines are d distance away from corners; numerical experiments show that the extraction efficiency is best when $d = w/2$ where w is the width of corner conductor. Cutting the primitive conductors creates many *virtual conductors*. Some of them may be straight, who will have uniform current distribution inside. This observation is the basement for the improvement of linear boundary elements, to be discussed later. Now, the accelerating effect of WHOLE may not be obvious because lots of long virtual conductors are not cut.

For the two cases, the different QBEMs are listed in Table 5.1. The average scheme (denoted by avg) is to cut the primitive conductors into some virtual parts, every one of which has the length less than two times its width.

Both QBEMs use less computation resource than DBEM (native direct boundary element); this is to say, QMM algorithm is useful to enhance efficiency. The WHOLE cutting has produced fewer elements than average cutting, partly because the interfaces are not divided. But the improvement of performance is

Table 5.1 Results of two of the quasi-cutting schemes

	sram.qbem					clk.qbem				
	#Ele	Mem (M)	Time (s)	Speedup	Err (%)	#Ele	Mem (M)	Time (s)	Speedup	Err (%)
QBEM (WHOLE)	839	0.52	5.90	1.00	5.224	4276	9.94	204.30	1.00	3.708
QBEM (avg)	870	0.51	4.71	0.80	5.470	4639	8.22	128.53	0.63	5.279
Direct BEM	835	0.68	8.65	1.47	4.395	4536	17.08	386.28	1.90	2.615
Raphael	63168	14.00	101.73	17.24	1.048	336398	89.00	6,118.76	29.95	1.578

Table 5.2 Results of dividing elements in two directions and in only one direction

	sram.qbem					clk.qbem				
	#Ele	Mem (M)	Time (s)	Speedup	Err (%)	#Ele	Mem (M)	Time (s)	Speedup	Err (%)
QBEM (1 dir)	587	0.19	2.91	1.00	1.854	3233	5.48	63.91	1.00	2.060
QBEM (2 dirs)	870	0.51	4.71	1.62	5.470	4639	8.22	128.53	2.01	5.279
Direct BEM	835	0.68	8.65	2.97	4.395	4536	17.08	386.28	6.04	2.615
Raphael	63168	14.00	101.73	34.96	1.048	336398	89.00	6,118.76	95.74	1.578

Table 5.3 Pure constant element versus coupling constant-linear element method

	sram.qbem						clk.qbem					
	#Ele	Mem	Time	Ele ratio	Speedup	Err	#Ele	Mem	Time	Ele ratio	Speedup	Err
QBEM (coupled)	287	<0.10	1.44	1.00	1.00	0.668	1173	1.52	28.69	1.00	1.00	1.049
QBEM (constant)	870	0.51	4.71	3.03	3.27	5.470	4639	8.22	128.53	3.95	4.48	5.279
Direct BEM	835	0.68	8.65	2.91	6.01	4.395	4536	17.08	386.28	3.87	13.46	2.615
Raphael	63168	14.00	101.73	220.10	70.65	1.048	336398	89.00	6,118.76	286.78	213.27	1.578

Table 5.4 Performance combining of our four kinds of improvements

	sram.qbem						clk.qbem					
	#Ele	Mem	Time	Mem ratio	Speedup	Err	#Ele	Mem	Time (s)	Mem ratio	Speedup	Err
QBEM	230	<0.10	1.11	1.00	1.00	0.791	598	0.43	8.56	1.00	1.00	−1.008
Direct BEM	835	0.68	8.65	6.80	7.79	4.395	4536	17.08	386.28	39.72	45.13	2.615
Raphael	3168	14.00	101.73	140.00	91.65	1.048	336398	89.00	6,118.76	206.98	714.81	1.578

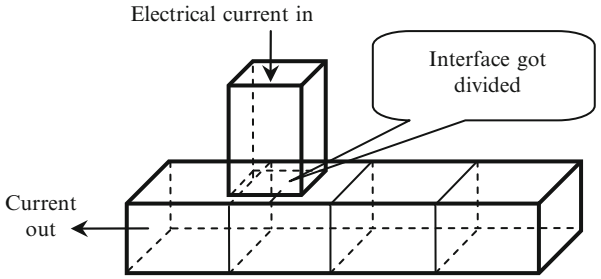

Fig. 5.4 Average cutting may cut interfaces and introduce more unknowns

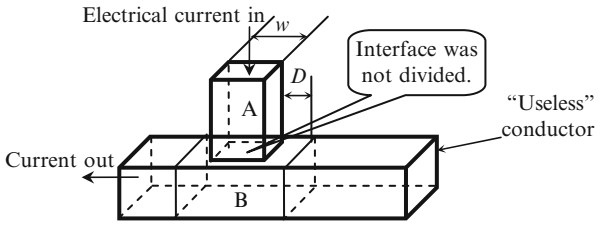

Fig. 5.5 New cutting strategy. A and B are corner conductors

not finalized yet. Combined with the improving measures to come, the WHOLE cutting has much better performance; refer to Table 5.4. The WHOLE cutting is the foundation for those to come.

5.3.2 Discarding Conductors Not in the Path of Direct Current

According to the electric theories, when a conductor is not in the path of direct current, there is no direct current flowing through it, and it is of equivalent potential. Such conductors will only introduce more unknowns to the numerical extractor, while getting rid of it will not affect accuracy. They are called *useless conductors*, with an example in Fig. 5.5.

Undoubtedly, if there are waste conductors in interconnect structures, removing them from numerical extraction can reduce the size of question and in turn improve the efficiency of resistance extraction.

5.3.3 Dividing Elements Only in One Direction When Possible

In QBEM capacitance extraction, conductor surfaces are meshed into 2-D elements, in both directions of the surface. In resistance extraction, some surfaces don't have

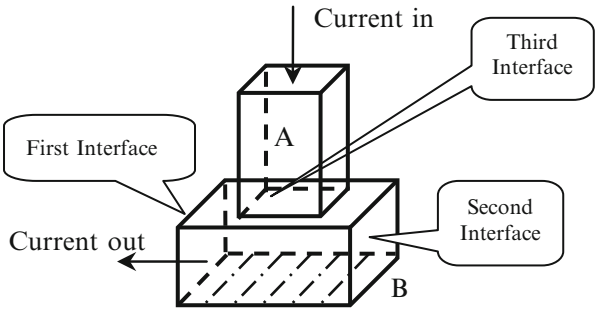

Fig. 5.6 Dividing elements in one direction on bottom surface of corner conductor B

to be discretized in two directions. One of such surfaces is the four surfaces of a straight conductor which are parallel to the direction of inner current, and another kind is one or some of corner conductor surfaces.

In addition, when the bias voltage of 1 V is applied to the ends, a straight conductor will have uniform current distribution, and it can be seen as a set of tiny equipotential bodies along the current direction. The current flows perpendicularly from one tiny body to another, so it is only in this direction that the potential and its derivative are varying. So in the BEM solver, the four surfaces of straight conductors are only discretized along the current direction. Obviously this can save many elements.

Even some surfaces on complicated conductors can also be meshed in one direction. For example, the bottom surface of conductor B in Fig. 5.6 has two perpendicular interfaces with the same normal vector, so the surface may be divided only in one direction. The other surfaces of B must be divided in two directions, though.

The one-direction meshing and two-direction meshing have different performances, as listed in Table 5.2. The table tells that the former has higher speed and better accuracy, because it's more physically reasonable.

Even so, if the straight conductor is very long, there will be a large number of one-direction elements. Then we can turn to linear elements, which are the topic of next section.

5.3.4 Linear Boundary Elements for Straight Conductors

Realistic interconnect structures may have very complicated geometry. However, some portions of the structures may be very simple. For example in Fig. 5.7 (left), the sections connecting different layers of wires and vias are complicated, while the other sections are just like the straight line in Fig. 5.2. Refer to the right half of this figure for the two different subregions.

When the bias voltage is applied on the straight conductors, the electric potential varies linearly along the current direction. Assume the current flow is along the x direction; the potential is a linear function of x: $u = ax + b$, where a and b are

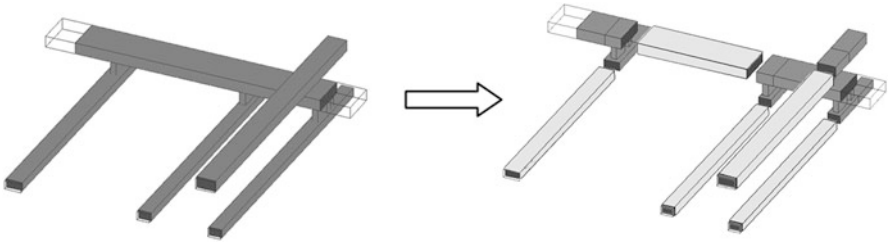

Fig. 5.7 An interconnect structure is cut into regular subregions (in *light color*) and irregular subregions (in *deep color*)

unknown constants. Then for this long conductor, a single linear boundary element is enough to simulate the linear variation of potential, which will only introduce two unknowns (a and b) to the linear system (5.7). On the contrary, constant elements will introduce too many u or q variables.

Therefore, the introduction of linear elements will be helpful to reduce system size. There is considerable difficulty in the formulation of integrals, as in the Appendix 5.A. Let's see the numerical performance in Table 5.3. The number of elements in the constant-linear coupling method is much smaller than in the constant-only element method. Beyond that, the speed and accuracy of the coupling method are much higher, with much less memory usage. To get better accelerating effect, we prefer to obtain as many straight conductors as possible after cutting, and this is why we did not cut those long conductors further.

Although linear elements are more efficient, it can't totally replace constant ones. They can only be meaningful for straight conductors, and constant elements are more widely applicable to general conductors.

5.3.5 *Efficiency Summary*

Here, all the improvement techniques above are combined together, denoted by QBEM in Table 5.4. Direct BEM means no improvement applied. The data is similar to what's observed in the previous experiments, while the total improvement here becomes even better: the speedups become 91x and 714x over Raphael.

Let's summarize all the improvement techniques. QMM-aided BEM technique is further customized for resistance extraction in the following four aspects: advanced scheme of cutting conductors, discarding useless conductors, dividing elements only in one direction when possible, and using linear boundary elements for straight conductors. Experiments show that the four kinds of improvements can greatly enhance QBEM in memory usage, speed, and accuracy.

5.4 Analytical QBEM Extraction

In realistic interconnect cases, there are many occurrences of straight wires; global power/ground nets and clock networks may be good examples. Considering the potential will be in linear distribution inside the wires, it's a good idea to simulate such wires using linear boundary elements, as discussed in previous subsection. It did bring considerable efficiency improvement since it reduces the unknown number and system size. However, for large scales of interconnects, the improved QBEM may still fail for memory limitations.

On the contrary, since the long conductors act like the conductor in Fig. 5.2, their resistance can be analytically obtained, while the other sections remain in the domain of QBEM. Then we can combine the two methods together, to result in analytical QBEM algorithm. The main advantage here is that each QBEM only solves a small part, which only requires limited resources. In addition, since each QBEM is independent of the others, it's straightforward to transfer its serial computing to parallel computing.

5.4.1 General Analytical QBEM Algorithm

The method cuts the whole interconnect structure into regular subregions and irregular subregions. A regular subregion is a straight conductor, and its resistance network is analytically obtained, where the computational resource requirement is almost zero. The irregular subregion may be composed of multiple layers of wires and wires, provided that all of them are connected. Its resistance is a network (as in Fig. 5.3) and will be calculated using QBEM. Then we can combine these networks to get the final resistance network. Because the computation complexity of QBEM is easily higher than linear $O(n)$, the computational resources required by all irregular subregions are still much less than that for QBEM that solves the interconnect structure as a whole. As observed in numerical experiments, the speedup may be 50X.

There are a few steps for this algorithm:

1. Generate regular and irregular subregions.
2. Compute regular resistance by analytical formulation.
3. Numerically solve resistance networks of irregular subregions.
4. Combine the resistance from 2 and 3 to get the final network.

The coming emphases are how to distinguish between regular and irregular subregions and how to combine part resistance networks to get the network of the whole region.

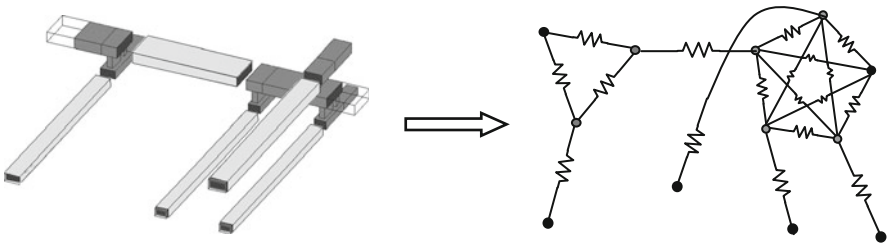

Fig. 5.8 Subregions and KCL network. *Solid dots* are primitive ports

5.4.2 Distinguish Between Regular and Irregular Subregions

When the voltages of interconnect conductors are of low frequency, the current flowing through corner sections is not uniform, but it becomes nearly uniform some distance away from the corners [178]. Accordingly, we can distinguish between regular and irregular subregions. Firstly, find the corners according to space relations of geometry shapes; secondly, cut the conductors into some virtual parts half a width away from the corners; the straight conductors are regular subregions, and the conductors left compose irregular subregions.

Adding one *virtual port* to some of the cutting places, we can see irregular subregions as independent resistance regions, and their resistance network, RNets, can be calculated independently. The resistance of regular subregions, R's, can be obtained analytically. RNets and R's constitute a net $\overline{G}$ corresponding to their space connection (see Fig. 5.8). The task left is to solve for the resistance network G of the whole region from $\overline{G}$.

5.4.3 Compute the Resistance Network of the Whole Region

Generally speaking, the resistance network of every irregular subregion is a full-graph G_i whose nodes respond to ports (including virtual ports) and sides respond to resistors between these ports. The resistance of regular subregions, R_j's, can be seen as sides connecting different G_i's and primitive ports. In this way, R_j's and G_i's constitute a net $\overline{G}$ (like the net in Fig. 5.6). Now, we must transform $\overline{G}$ equivalently to the network G of the whole region. The detailed algorithm is below:

1. Let one of the primary ports be the main port (volt = 1 V) and the others be the assist port (volt = 0 V).
2. Assume there are k irregular subregions, whose equivalent resistance networks are G, respectively ($i = 1, \ldots, k$). G_i has n_i ($n_i \geq 3$) nodes and $n_i(n_i - 1)/2$ sides. Give a volt variable and a current variable to each node and side of G_i, respectively, and the number of variables given to G_i is $n_i + n_i(n_i - 1)/2$.

Table 5.5 Comparison of the three methods

Method	AQBEM	QBEM	Raphael
Element num	$335 + 138$	3855	336398
Memory (MB)	0.27	8.22	89
Time (s)	2.97	128.53	6118.76
Max error	-1.83 %	7.31 %	1.58 %
Speedup	2,060.19	47.60	1

3. Assume there are l regular subregions and give a current variable to the sides in $\overline{G}$ corresponding to them. In total, l variables are given.
4. In G_i ($i = 1, \ldots, k$), for each node there is a node current equation, viz., KCL (Kirchhoff's Current Law) equation; because voltage fall between the two nodes of each side is equal to the current flowing through the side times the resistor value between the nodes, there is a voltage fall equation for the side. The number of equations obtained in G_i is also $n_i + n_i(n_i - 1)/2$.
5. There are l voltage fall equations for the corresponding sides of regular regions in $\overline{G}$, where the voltages set in 1 are of some use.
6. Solving the $\sum_{i=1}^{k} \frac{n_i\,(n_i + 1)}{2} + l$ equations formed in 4 and 5, we can get the values of the variables given in 2 and 3. Listing the KCL equations of primitive ports, we get the values of current flowing through primitive ports, and the reciprocals of them compose a column of the final resistance matrix corresponding to the main port.
7. Letting different primitive ports be the main port and repeating 2–6, we can get the resistance matrix (and resistance network) of the whole region.

Generally speaking, because the variables in the above algorithm are much fewer than the variables in 3-D numerical solution of irregular subregions, the time spent on the algorithm is very little.

5.4.4 Numerical Result and Analysis

The analytical QBEM method presented above is called AQBEM. In Table 5.5, it's compared with the commercial software Raphael (version: 2000.2, with FDM) and the direct boundary element method without the analytical method coupled (abbr. QBEM). All programs run on Sun Ultra Enterprise 450.

Note that Raphael adopts its default mesh; max error is for the maximum inside the resistance matrix in reference to Raphael with a mesh of 2M grids. The then table tells that the analytical BEM method has a speedup of 2060 over Raphael, with comparable accuracy and 0.5 % of the memory.

The above experiment is only for a small test case. For large cases, even if Raphael would not fail for memory overflow, AQBEM will gain more speedup due to its divide-and-conquer nature and Raphael's volume-based discretization. This is nice since realistic interconnects are usually pretty large.

5.5 Summary

Interconnect resistance extraction is important for VLSI designs now. The field solver of direct boundary element method can be improved significantly. In this chapter, two methods are proposed. The first method tries to reduce the mesh elements by introducing a linear boundary element to replace many constant elements for a straight segment of interconnect. This is based on the linear potential distribution along some long and straight conductor parts. The other tries to divide and conquer the problem to many parts; complicated parts are for the above improved boundary element methods, while straight parts are simply for analytical formulations. The combination of numerical and analytical methods offers even higher computing efficiency, with comparable accuracy. Parallel computing for this divide-and-conquer method is very easy, since each part is independent.

Appendix 5.A

Here, we give the derivation of the analytical integral formulas on the boundary elements with linear variety of the u and q. For simplicity, we assume that the integral point (x, y, z) is in the *XOY* plane, the z-coordinate of the plane is z_{xoy}, and $\boldsymbol{n}$, the unit normal vector of the plane, is $\{0, 0, 1\}$. The potential u of the integral plane changes linearly along the x-axis:

$$u = ax + b, \tag{5.A.1}$$

where a and b are unknown constants. Let the *source* point s have the coordinate (x_s, y_s, z_s).

Equation (5.3) can be written as

$$c_s u_s + \int_{\Gamma u} q^* u \mathrm{d}\Gamma + \int_{\Gamma q} q^* u \mathrm{d}\Gamma = \int_{\Gamma u} u^* q \mathrm{d}\Gamma + \int_{\Gamma q} u^* q \mathrm{d}\Gamma, \tag{5.A.2}$$

where Γ_u is the port surfaces, Γ_q is the other surfaces of the conductors in the region, and Γ, the whole boundary of the region, is equal to $\Gamma_u + \Gamma_q$. The boundary conditions are $u = 0$ or $u = 1$ on Γ_u and $q = 0$ on Γ_q (here, Γ_q must be a shape of a rectangle). Then we can get

$$\int_{\Gamma q} u^* q \mathrm{d}\Gamma = 0 \tag{5.A.3}$$

and

$$\int_{\Gamma u} q^* u \mathrm{d}\Gamma = C_1, \tag{5.A.4}$$

where C_1 is a constant only depending on Γ_u (exactly on the main port surfaces; see the Eq. (5.A.22)). Now, (5.A.2) is

$$C_1 + c_s u_s + \int_{\Gamma q} q^* u \mathrm{d}\Gamma = \int_{\Gamma u} u^* q \mathrm{d}\Gamma. \tag{5.A.5}$$

Note that linear elements are only on the conductor surfaces Γ_q. So what we need to do is to calculate the potential integral Int$U$:

$$\mathrm{Int}U = \int_{\Gamma q} q^* u \mathrm{d}\Gamma = \int_{\Gamma q} q^* (ax + b)\, \mathrm{d}\Gamma. \tag{5.A.6}$$

For simplicity, only the corresponding indefinite integrals are written in the following text with integral constants (C) omitted, and we will give the entire formulas finally.

$$q^* = \frac{\partial u^*}{\partial \boldsymbol{n}} = \nabla\left(\frac{1}{4\pi r}\right) \cdot \boldsymbol{n} = -\frac{1}{4\pi r^2}\left\{\frac{\partial r}{\partial x}, \frac{\partial r}{\partial y}, \frac{\partial r}{\partial z}\right\} \cdot \{0, 0, 1\} = -\frac{1}{4\pi r^2}\frac{\partial r}{\partial z}, \tag{5.A.7}$$

$$r = \sqrt{(x - x_s)^2 + (y - y_s)^2 + (z - z_s)^2}, \tag{5.A.8}$$

$$\frac{\partial r}{\partial z} = \frac{(z - z_s)}{r}. \tag{5.A.9}$$

Substituting (5.A.8) and (5.A.9) into (5.A.7), we get

$$q^* = -\frac{z - z_s}{4\pi r^3} = \frac{z_s - z_{xoy}}{4\pi r^3} = \frac{d}{4\pi r^3}, \tag{5.A.10}$$

where the constant $d = z_s - z_{xoy}$. Then

$$\mathrm{Int}U = \frac{d}{4\pi}\int_{\Gamma q} \frac{x}{r^3}\mathrm{d}\Gamma \cdot a + \frac{d}{4\pi}\int_{\Gamma q} \frac{1}{r^3}\mathrm{d}\Gamma \cdot b. \tag{5.A.11}$$

Let the coefficient of a be $d/4\pi * I_a$ where

$$I_a = \int_{\Gamma q} \frac{x}{r^3}\mathrm{d}\Gamma = \iint \frac{x}{r^3}\mathrm{d}x\mathrm{d}y \tag{5.A.12}$$

and the coefficient of b be $d/4\pi * I_b$ where

$$I_b = \int_{\Gamma q} \frac{1}{r^3}\mathrm{d}\Gamma = \iint \frac{1}{r^3}\mathrm{d}x\mathrm{d}y. \tag{5.A.13}$$

Firstly, consider I_b in the two steps of I_{b1} and I_{b2}:

$$I_{b1} = \int \frac{1}{\left(\sqrt{(x-x_s)^2+(y-y_s)^2+(z-z_s)^2}\right)^3}\mathrm{d}x = \int \frac{1}{\left(\sqrt{x^2+y^2+d^2}\right)^3}\mathrm{d}x$$
$$= \frac{x}{(y^2+d^2)\sqrt{x^2+y^2+d^2}}, \tag{5.A.14}$$

and

$$I_{b2} = \int \frac{x}{(y^2+d^2)\sqrt{x^2+y^2+d^2}}\mathrm{d}y = \frac{1}{d}\arctan\left(\frac{x}{d}\cdot\frac{y}{\sqrt{x^2+y^2+d^2}}\right). \tag{5.A.15}$$

Substituting (5.A.14) and (5.A.15) into (5.A.13), we can get I_b.
Secondly, transform coordinate for I_a:

$$I_a = \iint \frac{x+x_s}{\left(\sqrt{x^2+y^2+d^2}\right)^3}\mathrm{d}x\mathrm{d}y = \iint \frac{x}{\left(\sqrt{x^2+y^2+d^2}\right)^3}\mathrm{d}x\mathrm{d}y + x_s I_2, \tag{5.A.16}$$

where I_{a2} has the solution of the same format as I_b. Let

$$I_{a1} = \iint \frac{x}{\left(\sqrt{x^2+y^2+d^2}\right)^3}\mathrm{d}x\mathrm{d}y, \tag{5.A.17}$$

and we consider its indefinite integral in two steps:

$$I_{a11} = \int \frac{x}{\left(\sqrt{x^2+y^2+d^2}\right)^3}\mathrm{d}x = -\frac{1}{\sqrt{x^2+y^2+d^2}}. \tag{5.A.18}$$

$$I_{a12} = \int \frac{1}{\sqrt{x^2+y^2+d^2}}\mathrm{d}y = \log\left(y+\sqrt{x^2+y^2+d^2}\right). \tag{5.A.19}$$

Substituting (5.A.18) and (5.A.19) into (5.A.17), we can get I_{a1}. Then $I_a = I_{a1} + x_s I_{a2}$.

Finally, we write the entire formulas of I_a and I_b (for simplicity, the integral plane Γ_q is seen as a whole):

$$I_a = I_{a1} + x_s I_{a2} = \log\left(\frac{\left(y_2+\sqrt{x_1^2+y_2^2+d^2}\right)\left(y_1+\sqrt{x_2^2+y_1^2+d^2}\right)}{\left(y_2+\sqrt{x_2^2+y_2^2+d^2}\right)\left(y_1+\sqrt{x_1^2+y_1^2+d^2}\right)}\right) + \frac{x_s}{d}\left[\arctan\left(\frac{x_2y_2}{d\sqrt{x_2^2+y_2^2+d^2}}\right)+\arctan\left(\frac{x_1y_1}{d\sqrt{x_1^2+y_1^2+d^2}}\right) - \arctan\left(\frac{x_1y_2}{d\sqrt{x_1^2+y_2^2+d^2}}\right)-\arctan\left(\frac{x_2y_1}{d\sqrt{x_2^2+y_1^2+d^2}}\right)\right]. \tag{5.A.20}$$

$$I_b = \frac{1}{d}\left[\arctan\left(\frac{X_2Y_2}{d\sqrt{X_2^2+Y_2^2+d^2}}\right)+\arctan\left(\frac{X_1Y_1}{d\sqrt{X_1^2+Y_1^2+d^2}}\right) - \arctan\left(\frac{X_1Y_2}{d\sqrt{Y_1^2+Y_2^2+d^2}}\right)-\arctan\left(\frac{X_2Y_1}{d\sqrt{X_2^2+Y_1^2+d^2}}\right)\right], \tag{5.A.21}$$

where $x_1/x_2 = X_1/X_2 - x_s$, $y_1/y_2 = Y_1/Y_2 - y_s$, $d = z_s - z_{xoy}$, and X_1/X_2, Y_1/Y_2 are the boundary coordinates of Γ_q (as supposed, Γ_q is a shape of a rectangle).

We also write the formula of C_1:

$$C_1 = \int_{\Gamma u} q^* u \mathrm{d}\Gamma = \frac{1}{4\pi}\left[\arctan\left(\frac{X_2Y_2}{d\sqrt{X_2^2+Y_2^2+d^2}}\right)+\arctan\left(\frac{X_1Y_1}{d\sqrt{X_1^2+Y_1^2+d^2}}\right) - \arctan\left(\frac{X_1Y_2}{d\sqrt{Y_1^2+Y_2^2+d^2}}\right)-\arctan\left(\frac{X_2Y_1}{d\sqrt{X_2^2+Y_1^2+d^2}}\right)\right], \tag{5.A.22}$$

where $d = z_s - z_{xoy}$ and X_1/X_2, Y_1/Y_2 are the boundary coordinates of $\Gamma_{u\mathrm{Main}}$ (a rectangle surface of the main port).

Chapter 6
Substrate Resistance Extraction with Boundary Element Method

Realistic chips are composed of transistors, metal interconnects, and also substrates or basement materials that provide physical and electric supports to the chips. The popular silicon process starts from a nearly pure silicon wafer, which will be the foundation to grow active wells (p-wells and n-wells), and then in turn many layers of metal wires or interconnects. The foregoing chapters have proposed field solvers for interconnects, and now let's go on to silicon substrate.

At frequency of several gigahertz, a substrate behaves mainly resistively [50, 135]. So, the substrate coupling is modeled with resistor connecting the contacts on its top surface. The substrate resistance can be calculated with a device simulator, such as MEDICI [145], where complex physical effects, such as drift and diffusion, are involved. Therefore, such simulators run very slowly. A practical method may be the curve fitting with data obtained from device simulator or measurement. Although this approach is fast in some sense, their application is very limited [34]. Other methods are based on analytical formulas [91], which are efficient for some special cases, but hardly suitable for general ones. A lot of numerical methods are also proposed to calculate the resistance with high accuracy. They can be classified into the finite element method (FEM), the finite difference method (FDM) [153], and the method of Green's function (also called boundary element method [BEM]) [34, 49, 51, 92, 102, 135, 168]. FEM and FDM, with discretization of the entire three-dimensional (3-D) substrate volume, produce a large coefficient matrix, which limits the size of problem that they can handle. The methods based on the Green's function [34, 49, 51, 92, 102, 135, 168] only discretize the contact surfaces and therefore employ the fewer variables. However, for the multilayered structure, the Green's function is composed of several infinite series which converge very slowly. In Niknejad et al. [102], a numerically stable method was proposed to calculate the Green's function with discrete cosine transform (DCT) for acceleration. However, it is not actually stable, and the further remedy was presented in Xu et al. [168]. It should be pointed out that these Green's function-based methods are limited to the multilayered structure where each layer is of uniform resistivity. For more complicated structures such as those containing lateral resistivity variations,

W. Yu and X. Wang, *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*, DOI 10.1007/978-3-642-54298-5_6,

the corresponding Green's function can hardly be deduced [49, 128]. One way to deal with the lateral variation is to use a combined boundary element and finite element method [128], where the top part of substrate with inhomogeneous material is discretized with FEM mesh. Therefore, this method still requires larger computational resources.

In fact, there are lots of realistic substrates with layout-dependent doping profiles, such as the oxide wells, trenches, sinkers, buried diffusions, etc. [40]. There are also special structures like Faraday shields and junction shields [1] for noise reduction, which are buried components with different resistivity. For these non-stratified substrates, the derivation of the Green's function becomes very difficult if not impossible. On the other hand, simply neglecting the special components may induce large errors for the simulation.

Different from the Green's function-based method, there is another kind of boundary integral method called direct boundary element method (DBEM) [24]. The DBEM obtains the direct boundary integral equation (BIE) by adopting Green's identity and using the free-space Green's function as weighting function [24]. So, it avoids the difficulties of deducing the structure-dependent Green's function and also employs much fewer variables than the FDM or FEM. The DBEM has been successfully applied to capacitance and resistance extraction for various interconnect structures [159, 181].

This chapter will introduce DBEM to substrate resistance extraction and also several improvement techniques for efficient purposes. Firstly, a nonuniform boundary element partition scheme is presented as a basis of efficient DBEM simulation. Secondly, many inessential unknowns are removed from the linear system before solution but without loss of accuracy. Finally, quasi-multiple medium (QMM) [181] is helpful to make the coefficient matrix much sparser so as to expedite matrix reduction and final equation solution greatly. Numerical experiments also illustrate the accuracy and efficiency of the improved DBEM. It is compared with the DCT-accelerated Green's function method, the eigen-decomposition-based method [34], and an analytical integration method [92]. The results of Raphael [144] and IE3D [92] are also given for reference. Then, it's concluded that the new method is superior to the methods in Costa et al. [34] and Masoumi et al. [92] in CPU time and has speedup of 100x to Raphael and IE3D while preserving high accuracy. Especially, our method demonstrates its versatility in a substrate case with lateral resistivity variations, while it is beyond the ability of the other methods.

6.1 Field Solver for Substrate Resistance

A typical substrate example is shown in Fig. 6.1. It consists of three medium layers, denoted by Ω_1, Ω_2, and Ω_3; they are stratified with different height and resistivity. Contacts are on the very top surface, which connect outer circuits such as active wells and wires. And there is possibly a grounded plane on the very

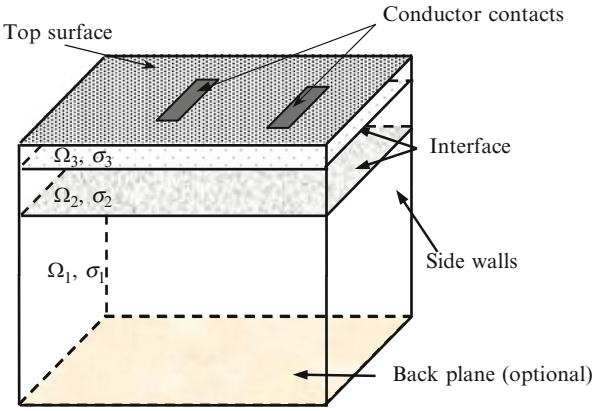

Fig. 6.1 A typical substrate structure

bottom; otherwise, it can be taken as a nature boundary. Another example with more complicated topology will come later in Fig. 6.9.

For substrate resistance extraction, one contact j (called master) is set with voltage 1 V and the others with 0 V. Then, the resistance R_{jk} between contact j and contact k ($k \neq j$) can be obtained by the current flowing through contact k [163]:

$$\frac{1}{R_{jk}} = \int_{\Gamma_{Ck}} \sigma \cdot \frac{\partial u}{\partial n} \mathrm{d}\Gamma = \int_{\Gamma_{Ck}} \sigma \cdot q \mathrm{d}\Gamma, \tag{6.1}$$

where Γ_{Ck} is the surface of contact k and σ is the conductivity (or reciprocal of resistivity ρ). u is the electric potential, and q is the normal electric field intensity on the boundary.

Direct boundary element method (DBEM) can solve (6.1) for q. This procedure is repeated by enforcing different master contacts, and finally, the complete resistance matrix is obtained. The details are as follows.

The electric potential u of the steady current field fulfills Laplace equation in each homogeneous medium:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = 0 \quad \text{in region } \Omega_i. \tag{6.2}$$

With the Green's identity and using the free-space Green's function as the weighting function, Laplace equation (6.2) can be transformed into a direct boundary integral equation (BIE); refer to Chap. 2 for more details. After discretizing the region boundary into quadrilateral elements with constant interpolation, we get the discretized BIE [24, 159, 163, 181]:

$$c_s u_s + \sum_{j=1}^{N_i} \int_{\Gamma_{ij}} q^*_{(s)} u \mathrm{d}\Gamma = \sum_{j=1}^{N_i} \int_{\Gamma_{ij}} u^*_{(s)} q \mathrm{d}\Gamma, \quad \text{for region } \Omega_i \tag{6.3}$$

where c_s is a constant depending on the boundary geometry at the collocation point s. $u^*_{(s)}$ is the free-space Green's function related with point s, and $q^*_{(s)}$ is its normal derivative on the boundary. The entire boundary of region Ω_i is partitioned into N_i elements; Γ_{ij} represents the jth element. After calculating the boundary integrals in (6.3) with the efficient approach proposed in Yu et al. [181], a linear equation with discretized unknowns of u and q is obtained. Evaluating (6.3) at collocation points, one for an element on the boundary of region Ω_i, a linear equation system is formed:

$$\boldsymbol{H}^{(i)} \cdot \boldsymbol{u}^{(i)} = \boldsymbol{G}^{(i)} \cdot \boldsymbol{q}^{(i)}, \quad \text{for region } \Omega_i . \tag{6.4}$$

Besides, u and q along the interface of two mediums a and b are both unknown, and they fulfill the compatibility equations:

$$\begin{cases} \sigma_a \cdot q_a = -\sigma_b \cdot q_b \\ u_a = u_b . \end{cases}, \quad \text{on interface } \Gamma_{\mathrm{I}} \tag{6.5}$$

The matrix equations (6.4) for all mediums can be put together with (6.5), where only pair u_a and q_a (or u_b and q_b) remains and the other is represented. Apply the boundary conditions (u on the contact surface is either 0 or 1, and q on the Neumann boundaries is 0; see Fig. 6.1), and reorganize the equations to an overall linear system:

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \tag{6.6}$$

where unknown vector $\boldsymbol{x}$ is the unknown u and q and the right-hand-side vector $\boldsymbol{b}$ corresponds to the specified bias voltage setting. To calculate the complete resistance network, the vector $\boldsymbol{b}$ has different values and (6.6) comes with multiple right-hand sides as follows:

$$\boldsymbol{A}\boldsymbol{X} = \boldsymbol{B}. \tag{6.7}$$

$\boldsymbol{B} = [\boldsymbol{b}\ \boldsymbol{b}_2\ \boldsymbol{b}_3\ \ldots]$. Solve this system for q and in turn for resistances through (6.1).

According to (6.3), only the elements on the single region's boundary have direct interactions. Therefore, the matrix $\boldsymbol{A}$ in (6.6) and (6.7) becomes sparse for multi-region problems. More details are in Yu et al. [181]. An efficient preconditioned GMRES solver [179] is applied. However, solving the linear system (6.6) consumes considerable time since too many unknowns are involved.

Note that the organization of matrix $\boldsymbol{A}$ can affect the performance of iterative solver remarkably. For capacitance extraction, an effective arrangement of the unknowns and collocation points, as well as the storage scheme for matrix $\boldsymbol{A}$, is in Yu et al. [181], which is also applied here for substrate resistance extraction.

6.2 Efficient Field-Solver Techniques

It becomes harder and harder, in terms of CPU time and memory, to solve larger and larger systems (6.7) and/or with denser and denser matrix $\boldsymbol{A}$. Here are three techniques for fast computation. Firstly, a nonuniform boundary meshing scheme reduces the system size. Then, a matrix reduction technology can numerically cut the linear system size before the expensive solution, without loss of accuracy. Finally, the quasi-multiple medium (QMM) technique is improved to make the matrix sparser, in order to speed up the matrix-vector product in iterative solvers.

6.2.1 Nonuniform Meshing

It's obvious that the size of system (6.6) is equal to the number of elements. How to generate elements affects both computational speed and accuracy. A simple scheme is uniform partition, while nonuniform meshing produces fewer elements while preserving desirable accuracy [181]. The substrate boundaries can be partitioned with a heuristic estimation of current distribution. This can be done in two stages. The first step is to estimate current flowing directions. The other is to mesh different kinds of boundary separately and distinctively:

1. *Forecasting current flowing directions.* If no back plane is present, current will flow horizontally between contacts; otherwise, considerable current will flow vertically toward the back plane. However, when the layer just above the back plane has high resistivity or many contacts are close to the current injector, the downward current will be less.
2. *Dividing Dirichlet boundary.* The contact surfaces should be meshed densely.
3. *Dividing sidewalls.* When a sidewall surface is near to contacts, the element size should be smaller. Otherwise, it can be larger.
4. *Dividing Neumann boundary on top surface.* The element mesh should be dense wherever close to contacts.
5. *Dividing medium interfaces.* If the layer connecting contacts is of higher resistivity than the lower layers, most current may flow downward to the back plane, and thus the interface portion just below the contacts should be partitioned into smaller elements; otherwise, most current may flow laterally, and the entire interface between this layer and lower layer should become smaller elements.

This scheme is based on experience from experiments and literatures [91, 141] and is adjustable for general complicated structures. A simple example of nonuniform elements is shown in Fig. 6.2.

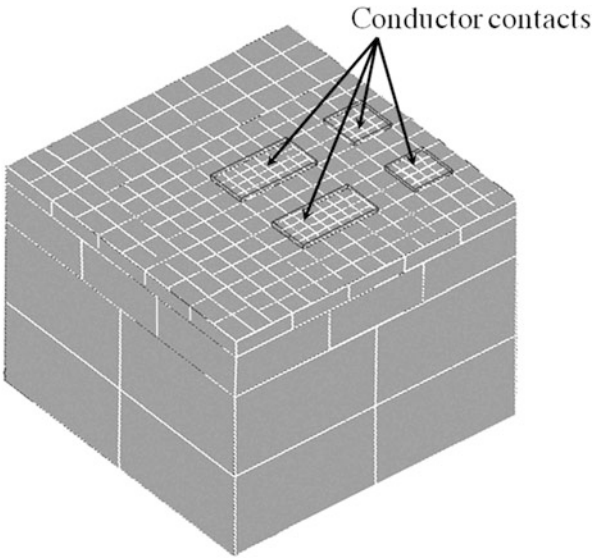

Fig. 6.2 A 3-D view of a three-layered substrate with nonuniform element mesh

6.2.2 *Numerical Reduction of Linear Equation System*

The idea is to discard some u variables, since only q variables are needed in (6.1). Based on the special unknown ordering [181], the first step is to further classify and reorder the unknowns for the top layer where all contacts reside. For example, in Fig. 6.1, the unknowns of type v_{33} (both q and u unknowns in medium Ω_3 excluding those on interface) include three subtypes and can be ordered as

$$v_{\mathrm{O}} \rightarrow q_{\mathrm{C}} \rightarrow u_{\mathrm{T}}, \tag{6.8}$$

where q_{C} represents q unknowns on contact surfaces (Γ_{C} in Fig. 6.1), u_{T} denotes u unknowns on the Neumann boundary of top surface (Γ_{T} in Fig. 6.1), and v_{O} denotes the other unknowns on the sidewalls in v_{33}.

After the subtype permutation in (6.8) is combined into the unknown ordering scheme in Yu et al. [181], the nonzero entry distribution of matrix $\boldsymbol{A}$ for the substrate in Fig. 6.1 is shown in Fig. 6.3a. Only the types of collocation points and unknowns related with Ω_3 are labeled there, separated by dashed lines.

Figure 6.3a verifies that the unknowns related with region Ω_3 account for a large part of total unknowns. This is generally true for substrate cases, because most boundary elements are located on or close to contacts in the nonuniform meshes.

Now, consider the discretized BIEs related with Ω_3, which form the nonzero submatrices $\boldsymbol{A}_{33}$, $\boldsymbol{A}_{3\mathrm{T}}$ ($=\boldsymbol{A}_{3\mathrm{T}1} \cup \boldsymbol{A}_{3\mathrm{T}2}$), $\boldsymbol{A}_{\mathrm{T}3}$, and $\boldsymbol{A}_{\mathrm{TT}}$ in Fig. 6.3a. These discretized BIEs can be expressed as

$$\begin{bmatrix} \boldsymbol{A}_{33} & \boldsymbol{A}_{3\mathrm{T}} \\ \boldsymbol{A}_{\mathrm{T}3} & \boldsymbol{A}_{\mathrm{TT}} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{x}_3 \\ \boldsymbol{u}_{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_3 \\ \boldsymbol{b}_{\mathrm{T}} \end{bmatrix}, \tag{6.9}$$

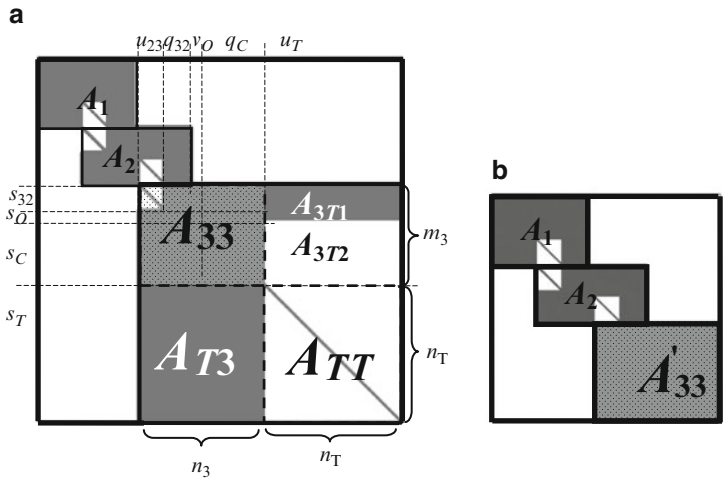

Fig. 6.3 (**a**) Original distribution of nonzero entries in matrix $\boldsymbol{A}$ for Fig. 6.1; (**b**) reduced

where $\boldsymbol{x}_3$ is the vector consisting of unknowns u_{23}, q_{32}, v_O, and q_C; refer to Fig. 6.3a. The second row of (6.9) is for the collocation points s_T, which are located on the top surface excluding on the contacts. Therefore, coefficient matrix $\boldsymbol{A}_{TT}$ is square. Because $\boldsymbol{u}_T$ is useless for calculating the resistance, let's get rid of it to reduce the equation as follows:

$$\boldsymbol{A}'_{33} \cdot \boldsymbol{x}_3 = \boldsymbol{b}'_3, \tag{6.10}$$

where

$$\boldsymbol{A}'_{33} = \boldsymbol{A}_{33} - \boldsymbol{A}_{3T}\boldsymbol{A}_{TT}^{-1}\boldsymbol{A}_{T3} \tag{6.11}$$

and

$$\boldsymbol{b}'_3 = \boldsymbol{b}_3 - \boldsymbol{A}_{3T}\boldsymbol{A}_{TT}^{-1}\boldsymbol{b}_T. \tag{6.12}$$

After (6.10) is combined with the Eq. (6.4) for the other mediums, a new global linear equation system is obtained, whose coefficient matrix is shown in Fig. 6.3b. Suppose the dimensions of matrices $\boldsymbol{A}_{33}$, $\boldsymbol{A}_{3T}$, $\boldsymbol{A}_{T3}$, and $\boldsymbol{A}_{TT}$ are $m_3 \times n_3$, $m_3 \times n_T$, $n_T \times n_3$, and $n_T \times n_T$, respectively. n_T is the number of elements on the Neumann boundary of top surface. m_3 is the number of other elements in region 3, including those on contacts, sidewalls, and the interface between regions 2 and 3. n_3 is a little larger than m_3; their difference is the number of elements on the region interface. If n_T is large, the effect of matrix reduction in (6.10) becomes important, which can greatly reduce the time for solving the final linear system.

Generally, the procedure of reduction in (6.11) has a computational complexity of $O(n_T^3 + n_T^2 n_3 + n_T m_3 n_3)$. This is not acceptable if n_T is very large. However, with the following three theorems, it can be greatly reduced.

Theorem 6.1 *$\boldsymbol{A}_{TT}$ is a diagonal matrix.*

Proof Consider the discretized direct BIE (6.3). For a s_T-type collocation point on the Neumann boundary of top surface (Γ_T in Fig. 6.1), the coefficient for a discretized unknown of type u_T is

$$\int_{\Gamma_{Tj}} q^*_{(s_T)} \mathrm{d}\Gamma = \int_{\Gamma_{Tj}} \frac{\partial u^*_{(s_T)}}{\partial \boldsymbol{n}} \mathrm{d}\Gamma = \int_{\Gamma_{Tj}} \frac{-1}{4\pi r^2} \cdot \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{n}} \mathrm{d}\Gamma, \tag{6.13}$$

where Γ_{Tj} is the jth element on the top surface. r is the distance between the collocation point and the integral point(s) on Γ_{Tj}, and $\boldsymbol{n}$ is the normal vector of Γ_{Tj}. If the collocation point is just located on Γ_{Tj}, the integral (6.13) is singular, of course not zero [181]. Otherwise, if s_T is not on Γ_{Tj}, $\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{n}} = 0$ because vector $\boldsymbol{r}$ (within the same plane as Γ_{Tj}) is perpendicular to $\boldsymbol{n}$. Since the unknowns of type u_T and collocation points of type s_T are in the same order, nonzero coefficients (i.e., the integral with collocation point just on Γ_{Tj}) are only on the diagonal. In other words, $\boldsymbol{A}_{TT}$ is diagonal. ■

Theorem 6.2 *$\boldsymbol{b}_T$ in (6.9) is a zero vector.*

Proof The right-hand side $\boldsymbol{b}$ is formed by summing up all known items in (6.3). In the discretized BIEs for medium Ω_3, the only nonzero known quantity is the preset bias voltage (1 V) of the master contact, whose coefficient is in the same expression of (6.13), except that the integral surface becomes contact surfaces. Since collocation points of type s_T are located on the same plane with the contact, but not on the contact itself, the coefficients become always zero. So, the $\boldsymbol{b}_T$ in (6.9) and (6.12) is zero. ■

Theorem 6.3 *In the discretized BIEs with collocation point of type s_C, the coefficients of u_T-type unknown are zero. In other words, the matrix $\boldsymbol{A}_{3T2}$ in Fig. 6.3*a *is zero.*

Proof Collocation points of type s_C are located on contact surfaces, which are on the same plane as the elements where variables of u_T are located. So, as analyzed in the proof of *Theorem 6.1*, the corresponding coefficient (6.13) is zero. Therefore, matrix $\boldsymbol{A}_{3T2}$ in Fig. 6.3a becomes a zero matrix. And we can say that the sub-matrix $\boldsymbol{A}_{3T}$ is partly sparse. ■

These three theorems explain the distribution of nonzero matrix entries shown in Fig. 6.3 and are helpful to analyze the cost of our matrix reduction technology. The numbers of multiplication operations needed in (6.11) and (6.12) are:

- $\boldsymbol{A}_{3T}\boldsymbol{A}_{TT}^{-1}$: less than $m_3 \times n_T$, because $\boldsymbol{A}_{TT}$ is diagonal and $\boldsymbol{A}_{3T}$ is sparse.

- $(\boldsymbol{A}_{3\mathrm{T}}\boldsymbol{A}_{TT}^{-1})\cdot\boldsymbol{A}_{\mathrm{T}3}$: less than $m_3\times n_{\mathrm{T}}\times n_3$, because $\boldsymbol{A}_{3\mathrm{T}}\boldsymbol{A}_{TT}^{-1}$ has the same sparse pattern as $\boldsymbol{A}_{3\mathrm{T}}$.
- $(\boldsymbol{A}_{3\mathrm{T}}\boldsymbol{A}_{TT}^{-1})\cdot\boldsymbol{b}_{\mathrm{T}}$: no operation needed, because $\boldsymbol{b}_{\mathrm{T}}$ is a zero vector.

Then briefly, we have the following theorem for total complexity:

Theorem 6.4 *Reducing system through* (6.11) and (6.12) *only needs less than* $O(m_3\times n_{\mathrm{T}}\times(1+n_3))$ *multiplication operations. The number of addition operations is similarly bounded.*

Clearly, this cost is thus much less than the previously estimated for general $O(n_T^3+n_T^2n_3+n_{\mathrm{T}}m_3n_3)$.

For substrate resistance extraction, there are many contacts, and all resistances among them need to be calculated. Therefore, by discarding the unknowns of type u_{T} as described above, the computational time for solving (6.10) with multiple right-hand sides will be greatly reduced, since the matrix reduction is conducted only one time but the iterative solver will run many times. Numerical experiments verify this. For example, a substrate case with two layers involves 1,511 unknowns after boundary element discretization, while only 779 unknowns were left after reduction technology. The reduction itself consumes only 2.20 s, but the average solving time is reduced from 3.54 to 0.48 s per right-hand-side (RHS) vector. Then, it's clear that for 100 RHS vectors, the total time is reduced from 354 to 50.2 s. Suppose there are 1,000 RHS vectors; the time will be reduced from 3,540 to 482.20 s.

6.2.3 *Quasi-multiple Medium Technique to Sparsify Matrix*

The localization character of DBEM is revealed by (6.3), where the variables in each BIE are within the same medium region. This character results in a blocked sparse coefficient matrix $\boldsymbol{A}$ for a multi-region problem. A quasi-multiple medium (QMM) method [181] is able to enlarge the matrix sparsity by cutting a medium into some fictitious medium blocks. The QMM can greatly reduce the CPU time and memory for interconnect capacitance/resistance extraction [159, 181]. Refer to Chap. 4 for more details.

For substrate resistance extraction, QMM technique is also helpful. Because there are relatively few boundary elements in the lower layers, only the top layer of substrate, with the contacts, is decomposed into $Q_x\times Q_y$ fictitious medium blocks vertically. Correspondingly, the nonzero sub-matrices for region 3 in Fig. 6.4a are replaced by many smaller nonzero sub-matrices. And the sparseness of matrix $\boldsymbol{A}$ is obviously enhanced.

The matrix reduction technology in the previous section can be easily extended to handle the system after QMM cuts the top layer. The elapsed time of matrix reduction becomes much less than that without QMM. For an actual substrate structure similar to that in Fig. 6.1, after QMM is applied ($Q_x=4$, $Q_y=2$), the

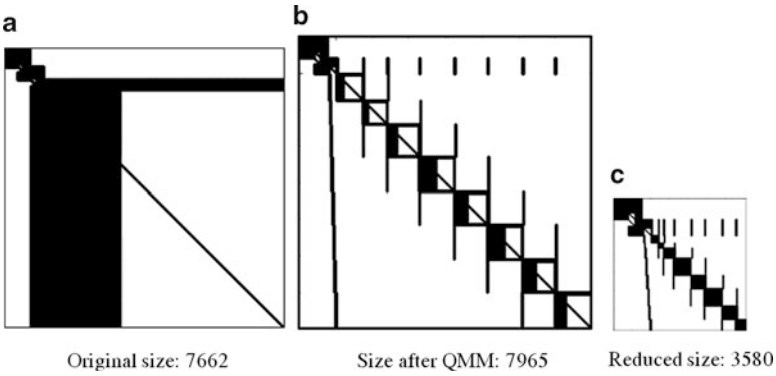

Fig. 6.4 Matrix for a typical substrate: (**a**) original, (**b**) after QMM, and (**c**) after reduction

nonzero entry distribution of matrix $\boldsymbol{A}$ turns from Fig. 6.4a into Fig. 6.4b. The original subblock for region Ω_3 is replaced by sparse 8×8 blocks. For the matrix in Fig. 6.4b, the CPU time for matrix reduction is only 3.1 s. And then, 2.8 s is needed for the preconditioned GMRES solver per right-hand side. But if not using the QMM technique, 580 s is needed to reduce 4,492 top-surface unknowns, while solving for one right-hand side costs 10.3 s. Therefore, the QMM technique improves both matrix reduction and equation solution.

6.3 Numerical Experiments

The presented method SubDBEM is implemented in C++. Some typical substrate structures demonstrate the efficiency of our algorithms. All experiments are on a Sun Fire V880 server with a frequency of 750 MHz. The first test case is a simple structure for purposes of accuracy and its convergence. The second case is a typical substrate with 52 contacts, where SubDBEM is compared with other Green's function-based methods. The last case is a substrate with lateral resistivity variation for the purpose of versatility. In the following text, all time data are in unit of second.

6.3.1 Simple One-Layer Substrate

The test case is a single-layer substrate as found in Ghapurey and Meyer [51] and Masoumi et al. [92] or Fig. 6.5. There is a contact on the center of the top surface and a grounded plane at the very bottom. The bulk height is 100 μm, and the resistivity is 10 $\Omega \cdot$cm. In Ghapurey and Meyer [51], the contact-to-ground resistance was calculated with multiple methods. It's 345 Ω by the Green's function method and

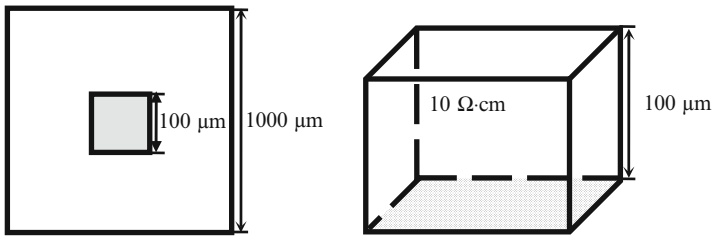

Fig. 6.5 The top view and 3-D view of a simple one-layer substrate structure (not scaled)

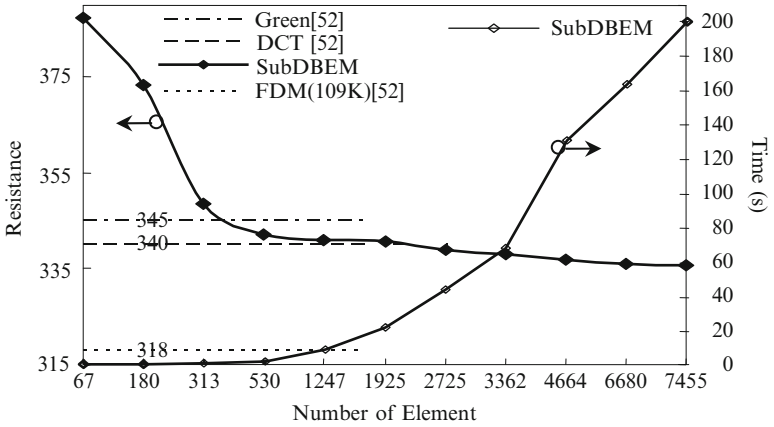

Fig. 6.6 Resistance values and running time versus element numbers

becomes 340 Ω when DCT is applied. The converged resistance from FDM is 318 Ω, with 109, 520 mesh points [51]. SubDBEM computes with different meshes, but QMM technique is not applied here. The converging curve is in Fig. 6.6.

Figure 6.6 says that the resistance value decreases with the element number but decreases more and more slowly before final convergence. If the value under the finest mesh (7,455 elements) is taken as reference, the values under 67 and 530 element meshes have errors of 15.4 and 1.9 %, respectively. At the same time, compared with the results of the Green's function method, DCT-accelerated method, and FDM, the converged value has little discrepancy. The computational time of SubDBEM versus element number is also in Fig. 6.6. For example, with middle size of 530 elements, the computational time is 1.4 s. The times for the latter three are 190 s, 76 s, and 674 s, respectively [51], on an IBM S6000 workstation with 166-MHz CPU. So, the speedups against the Green's function method, DCT-improved Green's function method, and FDM are **30x**, **12x**, and **106x**, respectively.

Although SubDBEM with denser meshes gives more reliable results, a trade-off should be made between computational accuracy and speed. In the following tests, adequate meshes are used which still produce substrate resistance with reasonable accuracy.

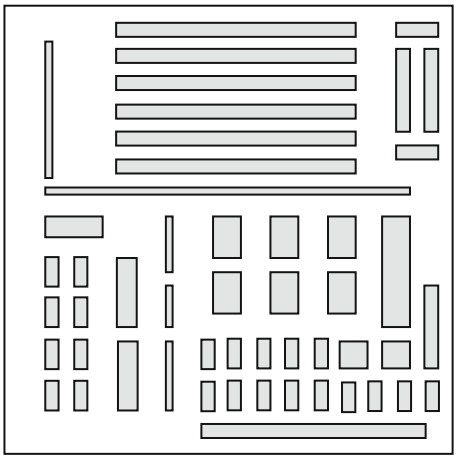

Fig. 6.7 Top view of a typical substrate

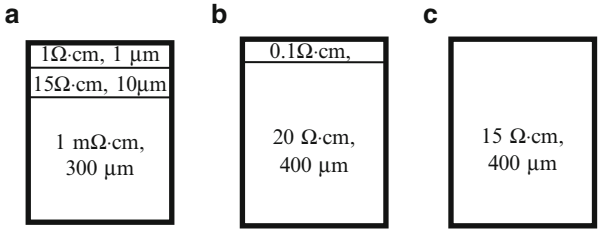

Fig. 6.8 (**a**) Low-resistivity profile, (**b**) high-resistivity profile, and (**c**) single-layer profile

6.3.2 The 52-Contact Structure with Three Doping Profiles

In Costa et al. [34] and Masoumi et al. [92], a complicated substrate from a mixed-signal circuit was used to demonstrate their extraction efficiency. It has 52 contacts on a 128 $\mu m \times 128$ μm substrate, as shown in Fig. 6.7. This structure can be processed with any of the three typical doping profiles in Fig. 6.8 [34, 91, 92].

This first test case is the substrate with low-resistivity process. The DCT-accelerated Green's function method and the eigen-decomposition-based method in Costa et al. [34] are denoted by "DCTGreen" and "EigenDec," respectively. Their results are obtained from Costa et al. [34]. The pure direct boundary element method without any of improvement techniques presented in this chapter is denoted by DBEM. Raphael is a 3-D FDM field solver from Synopsys [144]. All the programs ran on the same machine. Table 6.1 lists the different results, where $R_{i,j}$ is the resistance between contacts i and j. Because DBEM gives the same results as SubDBEM, they share the same row. QMM cutting of top layer is 4×2 for this case and also for the next high-resistivity case.

In Table 6.1, the discrepancy between SubDBEM and methods in Costa et al. [34] is within 5 %, except that for $R_{20,42}$. This is reasonable, because the current

Table 6.1 Resistance ohms for 52-contact substrate with low-resistivity profile

	$R_{1,2}$	$R_{16,19}$	$R_{19,2}$	$R_{20,42}$	$R_{21,49}$	$R_{37,45}$	$R_{37,50}$
DBEM/SubDBEM	4663	5791	6890	9.0×10^7	670	2651	89773
DCTGreen	4690	5850	6627	7.8×10^7	659	2633	89315
EigenDec	4685	5712	6643	7.8×10^7	658	2540	94327
Raphael[a]	4673	4341	4130	9.8×10^7	650	2476	46520

[a]Set dense 9×10^6 grids to get converged results

Table 6.2 Efficiency parameters for low-resistivity substrate

	SubDBEM	QBEM	RBEM	DBEM	Raphael[a]	DCTGreen	EigenDec
# unknowns	**7965**	7902	7662	7662	1.6×10^6	2647	17764
Memory (MB)	**35**	38	226	192	382	138	22.5
# iteration	**2433**	2461	2349	2310	N/A	1238	1868
Total time (s)	**160.4**	705	1,176	4,322	$>1 \times 10^5$	30,965.5	4,994.9
Workstation	Sun Ultra V880 (750 MHz)					Sun UltraSparc 1	

[a]With default 1.4×10^6 grids

Table 6.3 Performance parameters for high-resistivity substrate

	SubDBEM	Raphael	DCTGreen	EigenDec
# unknowns	**7261**	8×10^5	2647	17764
Memory (MB)	**31**	209	145	26
GMRES iteration	**2247**	N/A	8030	2930
Total time (s)	**112.6**	$>7 \times 10^4$	123,630	8,405.6
Workstation	Sun Ultra V880 (750 MHz)		Sun UltraSparc 1	

between contacts 20 and 42 is much smaller than others by several magnitudes. In addition, SubDBEM is closer to Raphael, with only 9 % discrepancy.

The relevant computational parameters are in Table 6.2. In order to demonstrate the efficiency of each of the improvement techniques presented in this chapter, more "half-done" programs were also included here. RBEM means DBEM plus the reduction technique, and QBEM means DBEM combined with QMM technique. Since it's not possible to compare our method and those in Costa et al. [34] directly on the same computer, the relative performance is estimated according to CPU parameters. Sun UltraSparc 1 workstation has a CPU with frequency of 143 MHz or 167 MHz. Therefore, Table 6.2 shows that the pure DBEM has no advantage over either DCTGreen or EigenDec without any improvement technique. However, SubDBEM has a speedup of **37x** over DCTGreen and nearly **6x** to EigenDec. The memory consumed by DCTGreen, EigenDec, and SubDBEM is on the same order or much less than that by Raphael or the pure DBEM.

If the high-resistivity profile in Fig. 6.8b is used, the computational parameters are listed in Table 6.3. No resistance result is available in Costa et al. [34], so the performance data is the focus here. DCTGreen and EigenDec ran on Sun UltraSparc 1 [34]. Assuming its CPU is of frequency 143 MHz, SubDBEM is more than **200** times faster than the DCTGreen and **14** times faster than the EigenDec. While

Table 6.4 Resistance ohms for the 52-contact substrate with single-layer profile

	$R_{3,4}$	$R_{10,19}$	$R_{25,50}$	$R_{33,34}$	$R_{48,50}$
SubDBEM	60.211	349.924	449.039	62.002	999.179
Method of [92]	60.352	353.551	437.771	61.921	1005.637
IE3D	61.333	365.343	450.343	62.734	1020.987
Raphael[a]	52.960	244.861	371.576	56.302	759.842

[a]Set dense 6×10^6 grids to get converged results

on memory usage side, SubDBEM is much more superior to the DCTGreen and consumes roughly the same as EigenDec.

It's interesting to compare the two tables (Tables 6.2 and 6.3). DCTGreen and EigenDec consume more CPU time for high-resistivity profile. This is because the number of GMRES iterations increases a lot. However, SubDBEM performs reversely better. The main reason is that there is only one medium interface here, which means a smaller number of boundary elements and fewer GMRES iterations.

For the single-layer profile in Fig. 6.8c, no computational result was given in Costa et al. [34]. But it was discussed in Masoumi et al. [92], where a fast-convergent Green's function and an analytical solution for the double surface integrals were proposed. Below, it will be compared with SubDBEM. The QMM technique in SubDBEM means two cuttings: first horizontal cutting of two layers (with heights 20 μm and 380 μm, respectively) and second vertical cutting of 2×4 on resultant top layer.

In Table 6.4, some mutual resistances by SubDBEM and Raphael are listed, as well as those from Masoumi et al. [92]. IE3D is a 3-D full-wave, method of moment (MoM), integral equation electromagnetic simulator, whose computational results are directly from Masoumi et al. [92].

SubDBEM has less than 5 % discrepancy against IE3D and has similar accuracy to the method in Masoumi et al. [92]. Raphael has worse accuracy for this case. The CPU time of SubDBEM is only 51.7 s, which is more than **940** times faster than IE3D (73,364 s), on a PC with 500-MHz frequency [92]. On a Sun Ultra 12 workstation, the method with analytical Green's function costs 441.2 s [92]. Although we have no exact data of the working frequency of Sun Ultra 12 workstation, we learn from the website of SUN Inc. that it will be above 140 MHz. Then, SubDBEM is even a bit faster than the analytical method in Masoumi et al. [92].

Beyond the advantages of speed and accuracy, SubDBEM can also handle substrates with arbitrary doping profiles, which is the topic of the next section.

6.3.3 Test Structure with Lateral Resistivity Variation

In most literatures of substrate resistance extraction, the stratified geometry is taken for granted, partially because the Green's function cannot be deduced for

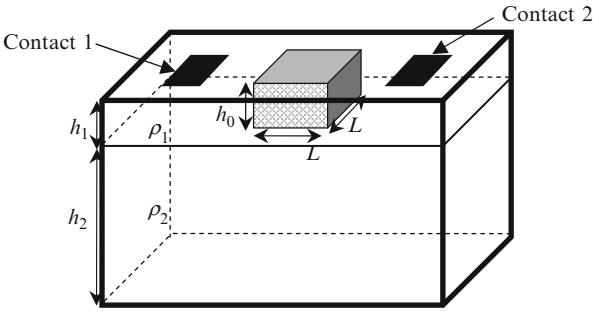

Fig. 6.9 A substrate with lateral resistivity variation

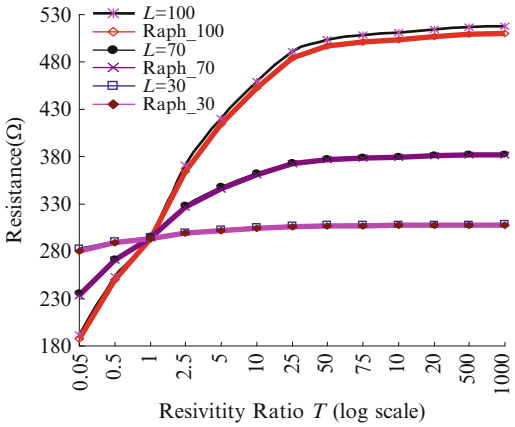

Fig. 6.10 Resistance versus the resistivity of lateral variation region

complicated doping geometries. On the contrary, SubDBEM has no limitation on geometries. To demonstrate this, a substrate with lateral resistivity variation is shown in Fig. 6.9.

It has two medium layers, with resistivity $\rho_1 = 0.1\ \Omega \cdot \text{cm}$ and $\rho_2 = 35\ \Omega \cdot \text{cm}$, respectively. The height is $h_1 = 5\ \mu\text{m}$ for upper layer and is $h_2 = 400\ \mu\text{m}$ for lower layer. The center of upper layer is an $L \times L$ square region with different resistivity, which is T times of ρ_1. The height of this resistivity-variation region is $h_0 = 5\ \mu\text{m}$. The dimension of whole substrate is $200\ \mu\text{m} \times 200\ \mu\text{m}$. On the top surface, there are two $10\ \mu\text{m} \times 10\ \mu\text{m}$ contacts located to the left and right sides of the resistivity-variation region. The resistance between these contacts varies with L and T. For $L = 100\ \mu\text{m}$, $70\ \mu\text{m}$, or $30\ \mu\text{m}$, the resistance values are depicted in Fig. 6.10, where the resistivity ratio T varied. SubDBEM has a 3×3 QMM cutting on the top layer so as to make the resistivity-variation region the central fictitious region.

It's learned from Fig. 6.10 that SubDBEM keeps closely up with Raphael, illustrating its high accuracy for such complicated case. Besides, the resistance increases with the resistivity of the lateral variation region and finally converges. This is reasonable, because the block region behaves more and more as an obstacle

in the path of current flow as its resistivity gets larger and larger. When the resistivity is large enough, the current will not flow through this region at all, so that the resistance between two contacts reaches a converged value. Also, the resistance increases more quickly for larger L. This can be explained similarly, because the larger the resistivity-variation region, the more it influences the current flow. The speedup of SubDBEM against Rapheal (with default grid) is 818.8 s/17.5 s = **47**.

6.4 Summary

The DBEM is presented and improved to extract 3-D substrate resistance. Many methods require stratified structure of substrates, while DBEM uses only the free-space Green's function, so it is able to handle arbitrary doping profiles. DBEM discretizes the entire boundary of a substrate and thus produces more unknowns. To improve the performance, three techniques are proposed. With nonuniform element partition and matrix reduction technique, the number of unknowns in DBEM is greatly reduced. And with the enhanced sparseness brought by the QMM technique, the final linear system solution can be performed much faster. The combination of these techniques improves the computational efficiency remarkably, especially for the structure involving many contacts and lateral resistivity variation.

Numerical results demonstrate the high accuracy. The experiments on a typical 52-contact structure show that SubDBEM is several times faster than the Green's function method improved by the eigen-decomposition technique and tens of times faster than the DCTGreen's function method. An example with lateral resistivity variation is also simulated to show the versatility of it.

Chapter 7
Extracting Frequency-Dependent Substrate Parasitics

As discussed in the previous chapter, the lossy nature of widely used silicon (Si) substrate may introduce considerable coupling effects, which are increasingly important for the design of mixed-signal and RF circuits [34, 102]. At frequencies lower than gigahertz, the coupling is often modeled with a resistance network, which is the topic of the previous chapter. At higher frequencies, both ohmic and displacement current will not be negligible anymore and should be modeled with frequency-dependent resistance and capacitance [102].

Similarly to resistance extraction, there are multiple kinds of numerical methods for frequency-dependent parameter extraction. The Green's function-based methods discretize the contact surfaces but only feasible for the multilayer structures. Even for multilayer mediums, the Green's function may be composed of several nested infinite series, and then converges very slowly. The techniques based on eigen-decomposition and discrete cosine transform (DCT) have been proposed to accelerate calculation of Green's function [34, 102]. For more complicated geometry structures such as those containing lateral resistivity variations, the corresponding Green's function can hardly be deduced [162].

The direct boundary element method (DBEM) discretizes the boundary of a homogeneous region and converts the Laplace equation to the boundary integral equation (BIE) with free-space Green's function [181]. DBEM is suitable for solving three-dimensional (3-D) electrostatic problem within finite domain [179, 181]. For multi-region problems, the BIE is derived for each homogeneous region and then coupled through the interface compatibility equations. Therefore, DBEM produces blocked sparse linear equation system. A quasi-multiple medium (QMM) technique can decompose a physical medium into fictitious regions, so as to generate a sparser linear system [181]. The DBEM can be utilized for substrate resistance extraction, as discussed in the previous chapter. With a matrix reduction technique and the QMM technique, the DBEM well outperforms the Green's function-based methods in Costa et al. [34] and Ghapurey and Meyer [51]. More importantly,

W. Yu and X. Wang, *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*, DOI 10.1007/978-3-642-54298-5_7,

the DBEM has no demand of the layered-geometry substrate. It is able to handle realistic substrates, such as those with noise reduction components, where layout-dependent doping profile makes lateral resistivity variation.

DBEM can also compute frequency-dependent impedance parameters, including both resistance and capacitance. As we will see later, this can be done by expanding the resistance extraction method presented in the last chapter using frequency-dependent medium permittivity. For multiple-frequency computation, one can simply repeat the extraction trivially. Obviously this is not the most efficient way. A two-step approach is presented for higher efficiency. The first step is to generate a real-valued linear equation system corresponding to resistance extraction. The second is to expand the resultant solution by Sherman-Morrison-Woodbury formula using frequency information. The new method will be compared with a Green's function-based method [102, 103] to demonstrate its efficiency and high accuracy.

7.1 Field Solver for Substrate Capacitance and Resistance

Strictly speaking, both electric and magnetic effects should be considered during the process of substrate coupling extraction, but the latter is usually negligible because the lightly doped Si substrate is widely used [47]. Here we ignore the magnetic effect. Figure 7.1a shows a 3-D substrate including two mediums, Ω_1 and Ω_2. Each medium has its own conductivity σ and permittivity ε. The mediums are usually different from each other. The coupling effects between contacts 1 and 2 in Fig. 7.1a can be represented by the equivalent model in Fig. 7.1b. Generally, the equivalent capacitance and resistance in Fig. 7.1b are both frequency dependent [102] due to frequency-dependent property of the mediums.

The admittance matrix $Y(\omega)$ in Fig. 7.1 is obviously dependent on angular frequency ω. Its real part represents resistive effects, while its imaginary part is

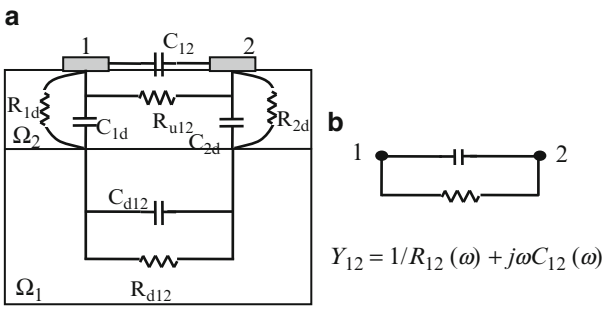

Fig. 7.1 A conceptual illustration of the frequency-dependent model. (**a**) Cross-sectional view of a lossy substrate with two contacts. (**b**) Equivalent frequency-dependent parameters for the coupling between contacts 1 and 2

for capacitive effects. The admittance can be calculated by setting voltage 1 V on a single contact (say, contact i) and 0 V on others. Then, the current flowing through contact k equals to the value of Y_{ik}:

$$Y_{ik} = \int_{\Gamma_{Ck}} (\sigma_2 + j\omega\varepsilon_2)\, q \mathrm{d}\Gamma, \tag{7.1}$$

where Γ_{Ck} is the surface of contact k and q is the normal electric field intensity on the contact boundary. Note that the value of q is usually frequency dependent. Suppose the values of q on boundary are numerically obtained, the corresponding resistance and capacitance can be obtained with

$$R\,(\omega) = 1/\mathrm{real}(Y), \quad C\,(\omega) = \mathrm{imag}(Y)/j\omega. \tag{7.2}$$

The conductors in Fig. 7.1a are often approximated as planar contacts with zero thickness, because conductor thickness is usually much smaller than its lateral dimensions. Whether the contacts are modeled as 2-D flatten contact or 3-D conductor of nonzero thickness makes little difference for DBEM here, since the modeling only affects the boundary conditions of simulated region. However, the 3-D modeling will bring considerable difficulty to Green's function-based methods, even makes it theoretically impossible. In addition, realistic substrate structures may be more complicated than Fig. 7.1a, with non-stratified medium topology or dielectric material (whose σ is zero) involved. For these kinds of mediums, it's very computationally expensive, or even impossible, to find a special Green's function. On the contrary, DBEM can easily handle them; refer to the end of the previous chapter.

7.2 Direct Boundary Element Method for Substrate Impedance Extraction

Here the substrate was abstracted as in Fig. 7.2. Since only the quasi-state electric field and steady current field are considered, the electric potential u fulfills the Laplace equation in each homogenous medium. With the DBEM, a linear equation system is formed for each medium region [181]:

$$\boldsymbol{H}^{(i)} \cdot \boldsymbol{u}^{(i)} = \boldsymbol{G}^{(i)} \cdot \boldsymbol{q}^{(i)}, \quad \text{for region } \Omega_i, \tag{7.3}$$

where vectors $\boldsymbol{u}^{(i)}$ and $\boldsymbol{q}^{(i)}$ are the discretized u and q unknowns on the ith medium's boundary, respectively.

On the interface of medium a and b, u and q unknowns fulfill the compatibility equations:

$$\begin{cases} (\sigma_a + j\omega\varepsilon_a)\, q_a = -(\sigma_b + j\omega\varepsilon_b)\, q_b, & \text{on interface } \Gamma_{\mathrm{I}}. \\ u_a = u_b \end{cases} \tag{7.4}$$

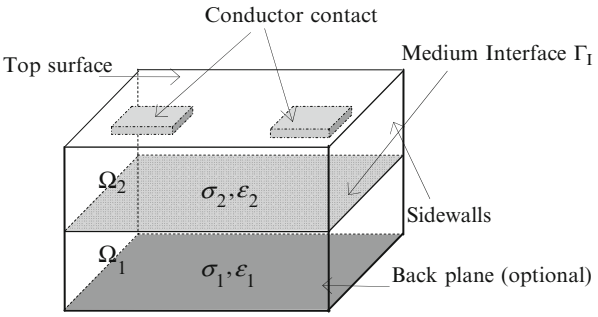

Fig. 7.2 A typical multiple-layer substrate and its boundaries

The matrix equations (7.3) for all regions can be combined together with (7.4). Using the boundary conditions (see Fig. 7.2) and organizing the equations as in Yu et al. [181], a linear equation system with multiple right-hand side (RHS) vectors is obtained for extracting the admittance matrix [162]:

$$\boldsymbol{A}\boldsymbol{X} = \boldsymbol{B}. \tag{7.5}$$

Here $\boldsymbol{B}$ consists of RHS vectors for specified voltage settings, and $\boldsymbol{X}$ is comprised of the discretized unknowns of u and q.

Because Eq. (7.4) includes complex-valued numbers, the linear system (7.5) is also complex valued. Therefore, the solution (u and q) depends on frequency. To acquire comprehensive understanding of the coupling effects, it is necessary to solve (7.5) at many frequencies. A trivial approach is to repeat at each frequency [160]. A much better method will come later, whose running time is almost independent of the number of frequency points.

Note the above formulation is actually the generalization of resistance extraction or capacitance extraction. If frequency is set to zero, the $j\omega\varepsilon$ items disappear in (7.2) and (7.4), and it is for resistance extraction, which is already discussed in the previous chapter. Similarly, if the σ items are dropped, i.e., the substrate is totally lossless, it is equal to capacitance extraction. In both cases, the linear system is real valued and frequency independent.

7.3 The Two-Step Approach

Let's first examine the distribution of frequency-dependent entries in matrix $\boldsymbol{A}$ and then introduce a two-step approach based on equation perturbation. To get most out of the new approach, efficient solving techniques will follow.

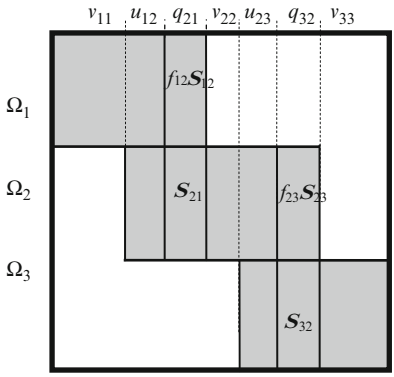

Fig. 7.3 Nonzero entry (*in gray*) of matrix $\boldsymbol{A}$ for a three-layer substrate. Only the blocks $f_{12}\boldsymbol{S}_{12}$ and $f_{23}\boldsymbol{S}_{23}$ depend on frequency

7.3.1 Frequency-Dependent Entries in Matrix A

Without loss of generality, take the substrate in Fig. 7.2 as an example. Its matrix $\boldsymbol{A}$ can be found in Fig. 7.3, where gray area means nonzero entries and white means zeros. Various types of unknowns (u or q) are marked for columns, where u_{12}/q_{21} are on interface of mediums Ω_1 and Ω_2.

Now it's clear that matrix $\boldsymbol{A}$ is a blocked sparse. Suppose the interface unknown q_{21} is for medium Ω_2, while its counterpart for medium Ω_1 can be represented by (7.4). Thus, the matrix block $\boldsymbol{S}_{21}$ has real-valued entries calculated through panel integral, and the block directly above $\boldsymbol{S}_{21}$ can be expressed as $f_{12}\boldsymbol{S}_{12}$, where $f_{12} = -(\sigma_2 + j\omega\varepsilon_2)/(\sigma_1 + j\omega\varepsilon_1)$ and $\boldsymbol{S}_{12}$ also consist of panel integrals. It's similar for the interface between Ω_2 and Ω_3, whether the coefficients of q_{32} are a real-valued matrix block $\boldsymbol{S}_{32}$ and a scalar-matrix product $f_{23}\boldsymbol{S}_{23}$.

Because only (7.4) introduces complex value $j\omega$, all entries in matrix $\boldsymbol{A}$ except those in blocks $f_{12}\boldsymbol{S}_{12}$ and $f_{23}\boldsymbol{S}_{23}$ are real values. In other words, the frequency dependency is only caused by the scalars f_{12} and f_{23}.

On the other hand, the quasi-multiple medium (QMM) [162] technique is helpful to introduce a sparser matrix, which may result in considerable performance benefits. If QMM cuts the first top of medium into 2×2 fictitious mediums, its matrix $\boldsymbol{A}$ turns into Fig. 7.4. It's obvious that the block-sparse structure is reserved and improved.

Note that the complex-valued blocks inside Fig. 7.3 now turn into more small blocks. Because the value of $\sigma + j\omega\varepsilon$ for a fictitious medium region is the same as the original medium it belongs to, the additional interface brought by QMM cutting does not introduce more frequency-dependent or complex-valued entries to matrix $\boldsymbol{A}$. Comparing Fig. 7.3 and Fig. 7.4, it's found that QMM cutting only makes the frequency-dependent block $f_{23}\boldsymbol{S}_{23}$ dispersed horizontally to be several strips. The matrix reduction technique (condense matrix $\boldsymbol{A}$ before iterative solution) proposed in Chap. 6 does not change the above observation, because it only equivalently discards the u unknowns on top surface. In other words, the reduction technique only condenses the equations for top medium Ω_3.

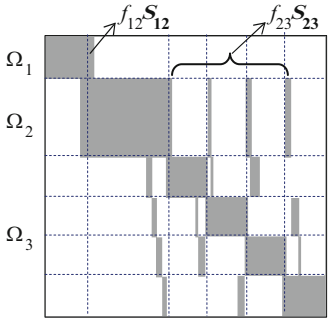

Fig. 7.4 After 2 × 2 QMM cuts top medium, matrix $\boldsymbol{A}$ becomes a 6 × 6 blocked matrix

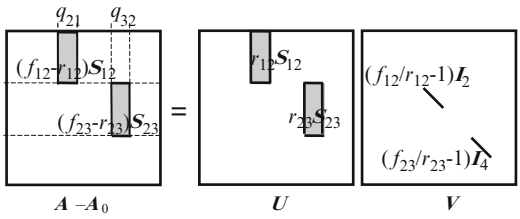

Fig. 7.5 Matrix $\boldsymbol{A}$ is equal to $\boldsymbol{A}_0 + \boldsymbol{UV}$. U is real valued. $\boldsymbol{I}_2$ and $\boldsymbol{I}_4$ are identity matrices

Even after the matrix reduction technique is applied, one has to solve for a complex linear system (7.5) at each frequency. Solving the complex system at a single-frequency point is already multiple times computationally expansive than solving the real system. What's more, solving at many frequency points are even more expensive. The following section will introduce a two-step approach for much more efficient multiple-frequency computation.

7.3.2 *Perturbed Equation System and Its Efficient Solution*

Equation (7.5) can be regarded as a perturbed equation system, with coefficient matrix expanded from a frequency-independent real-valued matrix $\boldsymbol{A}_0$. For the matrix $\boldsymbol{A}$ in Fig. 7.3, $\boldsymbol{A}_0$ is defined as

$$\boldsymbol{A}_0 \equiv \boldsymbol{A}, \quad \text{if } f_{12} = r_{12} \text{ and } f_{23} = r_{23} \tag{7.6}$$

where r_{12} and r_{23} are real constant values (e.g., -1). Then,

$$\boldsymbol{A} = \boldsymbol{A}_0 + \boldsymbol{UV} \tag{7.7}$$

where $\boldsymbol{U}$ is a sparse matrix with nonzero entries being a subset of those of $\boldsymbol{A}_0$ and $\boldsymbol{V}$ is a sparse diagonal matrix (see Fig. 7.5). Note that matrix $\boldsymbol{U}$ is frequency independent. Let

$$\boldsymbol{A}_0\boldsymbol{X}_0 = \boldsymbol{B} \tag{7.8}$$

be a real-valued linear system. Below we consider the relationship between the solutions of (7.5) and (7.8).

The Sherman-Morrison-Woodbury formula [53] claims

$$\left(\boldsymbol{A}_0 + \boldsymbol{U}\boldsymbol{V}^{\mathrm{T}}\right)^{-1} = \boldsymbol{A}_0^{-1} - \boldsymbol{A}_0^{-1}\boldsymbol{U}\left(\boldsymbol{I} + \boldsymbol{V}^{\mathrm{T}}\boldsymbol{A}_0^{-1}\boldsymbol{U}\right)^{-1}\boldsymbol{V}^{\mathrm{T}}\boldsymbol{A}_0^{-1}, \tag{7.9}$$

provided $\boldsymbol{A}_0$ and $(\boldsymbol{I} + \boldsymbol{V}^{\mathrm{T}}\boldsymbol{A}_0^{-1}\boldsymbol{U})$ are invertible. Since $\boldsymbol{V}$ is a diagonal matrix and $\boldsymbol{A} = \boldsymbol{A}_0 + \boldsymbol{U}\boldsymbol{V}$,

$$\boldsymbol{X} = \boldsymbol{A}^{-1}\boldsymbol{B} = \boldsymbol{A}_0^{-1}\boldsymbol{B} - \left(\boldsymbol{A}_0^{-1}\boldsymbol{U}\right)\left(\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U}\right)^{-1}\boldsymbol{V}\left(\boldsymbol{A}_0^{-1}\boldsymbol{B}\right). \tag{7.10}$$

Therefore, the solution of system (7.5) can be derived from the solution to (7.8). In other words, the frequency-dependent parameters can be obtained based on the frequency-independent solution $\boldsymbol{X}_0$. Since both $\boldsymbol{X}_0$ and $\boldsymbol{A}_0^{-1}\boldsymbol{U}$ are frequency independent, they can be calculated one time and reused for whatever frequencies.

Therefore, the main task for solving (7.5) is to compute $(\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U})^{-1}\boldsymbol{V}\boldsymbol{X}_0$. Note that $\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U}$ is complex valued and is of the same dimension as $\boldsymbol{A}$. Trivially computing it will make no sense.

However, $\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U}$ is also highly sparse and has special sparse pattern. Now these properties are taken advantages, in order to generate efficient algorithms to get $(\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U})^{-1}\boldsymbol{V}\boldsymbol{X}_0$. The following is how.

In $\boldsymbol{A} - \boldsymbol{A}_0$, the possible nonzero entries are at the positions corresponding to interface unknowns q; refer to Fig. 7.5. For a simulated structure including N_I "physical" interfaces, there are exactly N_I nonzero blocks in $\boldsymbol{A} - \boldsymbol{A}_0$ if all r parameters in (7.6) are set to be -1. Here "physical" interfaces denote those of different σ and/or ε. A "nonphysical" interface is between two mediums of the same σ and ε, for example, the fictitious interface induced by QMM. The corresponding f parameter in Fig. 7.3 will be constant -1, which does not contribute any nonzero entry to matrix $\boldsymbol{A} - \boldsymbol{A}_0$. The nonzero blocks in $\boldsymbol{A} - \boldsymbol{A}_0$ occupy m columns, where m is the number of elements on all the N_I "physical" interfaces. Because matrix $\boldsymbol{U}$ has the same sparse pattern as $\boldsymbol{A} - \boldsymbol{A}_0$, $\boldsymbol{A}_0^{-1}\boldsymbol{U}$ has exactly m nonzero columns. As shown in Fig. 7.5, the row indexes of nonzero diagonal entries in $\boldsymbol{V}$ are the same as the column indexes of nonzero entries in $\boldsymbol{A}_0^{-1}\boldsymbol{U}$.

As a result, $\boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U}$ is a sparse matrix with $m \times m$ nonzero entries in $N_I \times N_I$ blocks. Figure 7.6 shows the matrix $\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U}$ for the three-medium example, where $N_I = 2$. As it's shown later, this is the most important feature that can be used to reduce the computation of $(\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U})^{-1}\boldsymbol{V}\boldsymbol{X}_0$.

Let $\boldsymbol{M}_{ik}$ $(1 \le i, k \le N_I)$ denote the nonzero blocks in $\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U}$ and form a compact matrix $\boldsymbol{M} = [\boldsymbol{M}_{ik}]_{N_I \times N_I}$. Now let's prove that inversing $\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U}$ is exactly equal to inversing $\boldsymbol{M}$.

Theorem 7.1 *Computing* $(\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U})^{-1}$ *equals to inversing* the $m \times m$ *matrix* $\boldsymbol{M}$.

Proof Let $\boldsymbol{W} = \boldsymbol{M}^{-1}$. Express $\boldsymbol{W}$ as the blocked form $\boldsymbol{W} = [\boldsymbol{W}_{ik}]_{N_I \times N_I}$. Then, fill $\boldsymbol{W}_{ik}$ $(1 \le i, k \le N_I)$ into an identity matrix $\boldsymbol{I}$ according to the sparseness pattern of $\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U}$. In this way, $(\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U})^{-1}$ is generated by only inverting $\boldsymbol{M}$.

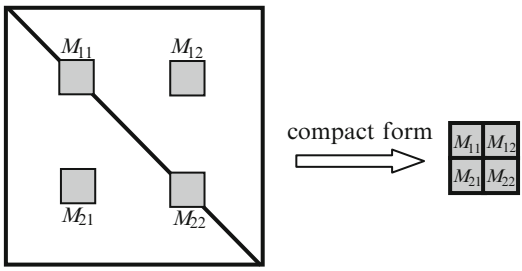

Fig. 7.6 Matrix $\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U}$ and its compact form for the example in Fig. 7.3

This can be verified with the example in Fig. 7.6 ($\boldsymbol{I}_1$, $\boldsymbol{I}_3$, and $\boldsymbol{I}_5$ denote three identity matrices of appropriate sizes):

$$\begin{bmatrix} \boldsymbol{I}_1 & & & \\ & \boldsymbol{M}_{11} & \boldsymbol{M}_{12} & \\ & & \boldsymbol{I}_3 & \\ & \boldsymbol{M}_{21} & \boldsymbol{M}_{22} & \\ & & & \boldsymbol{I}_5 \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{I}_1 & & & \\ & \boldsymbol{W}_{11} & \boldsymbol{W}_{12} & \\ & & \boldsymbol{I}_3 & \\ & \boldsymbol{W}_{21} & \boldsymbol{W}_{22} & \\ & & & \boldsymbol{I}_5 \end{bmatrix} = \boldsymbol{I}, \tag{7.11}$$

For general cases, $(\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U})^{-1}$ can be constructed similarly. ■

Notice that what's really needed is $(\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U})^{-1}\boldsymbol{V}\boldsymbol{X}_0$, not $(\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U})^{-1}$ itself. If the number of conductors is N_c, the dimension of $\boldsymbol{X}_0$ in (7.5) is $N \times N_c$. It's easy to firstly calculate $\boldsymbol{V}\boldsymbol{X}_0$, since $\boldsymbol{V}$ is a diagonal matrix. Then, solve the linear equation with coefficient matrix $\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U}$ and right-hand side vectors $\boldsymbol{V}\boldsymbol{X}_0$. According to Theorem 7.1, $(\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U})^{-1}\boldsymbol{V}\boldsymbol{X}_0$ is actually solved indirectly by inverting the much smaller matrix $\boldsymbol{M}$, instead of the big $\boldsymbol{I} + \boldsymbol{V}\boldsymbol{A}_0^{-1}\boldsymbol{U}$ itself. This result is then multiplied by $\boldsymbol{A}_0^{-1}\boldsymbol{U}$ ($N \times N$ matrix with m nonzero columns) to get desired $\boldsymbol{X}$.

Now the most time-consuming part becomes solve real-valued system (7.8). Let's see how to accelerate it.

7.4 Efficient Technique for Solving the Real-Valued System (7.8)

Because vectors $\boldsymbol{B}$ in (7.5) do not include complex numbers, (7.8) is a real-valued linear equation system. It can be regarded as resistance extraction equation, when all mediums are replaced by normal conductors.

Since both $A_0^{-1}B$ and $A_0^{-1}U$ are needed in (7.10), they are regarded as a single linear system with enlarged RHS vectors $\boldsymbol{P} = [\boldsymbol{B}\ \boldsymbol{U}]$. Blocked Gauss method can perform better than the normal preconditioned GMRES algorithm [162].

For brief introduction, let's consider a 2×2 blocked equation:

$$\begin{bmatrix} \boldsymbol{D}_{11} & \boldsymbol{D}_{12} \\ \boldsymbol{D}_{21} & \boldsymbol{D}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{bmatrix}. \tag{7.12}$$

Equivalently,

$$\begin{cases} \left(\boldsymbol{D}_{22} - \boldsymbol{D}_{21}\boldsymbol{D}_{11}^{-1}\boldsymbol{D}_{12}\right)\boldsymbol{x}_2 = \boldsymbol{b}_2 - \boldsymbol{D}_{21}\boldsymbol{D}_{11}^{-1}\boldsymbol{b}_1 \\ \boldsymbol{D}_{11}\boldsymbol{x}_1 = \boldsymbol{b}_1 - \boldsymbol{D}_{12}\boldsymbol{x}_2. \end{cases} \tag{7.13}$$

In other words, $\boldsymbol{x}_2$ and $\boldsymbol{x}_1$ can be solved in sequence. In this procedure, smaller linear system with coefficient matrix $\boldsymbol{D}_{11}$ and $\boldsymbol{D}_{22} - \boldsymbol{D}_{22}\boldsymbol{D}_{22}{}^{-1}\boldsymbol{D}_{21}$ (or updated $\boldsymbol{D}_{22}$) is solved. For a general $N_b \times N_b$ blocked equation, a similar deduction can be applied. Note that $\boldsymbol{D}_{22}{}^{-1}$ in (7.13) is purely symbolic. The matrix inverse is never computed explicitly, instead $\boldsymbol{D}_{22}{}^{-1}\boldsymbol{C}$ is obtained by solving equation $\boldsymbol{D}_{11}\boldsymbol{x} = \boldsymbol{c}$, where $\boldsymbol{c}$ is either a column vector or a matrix including multiple columns.

Compared with the original Gauss elimination, the blocked Gauss method is advantageous in that LU factorization is not done for the whole matrix but for smaller diagonal block, like $\boldsymbol{D}_{11}$. If many blocks in the coefficient matrix are zero, or include a few of nonzero columns, the blocked Gauss method will introduce a lot of efficiency benefits. At the same time, for a linear system with multiple RHS vectors, the blocked Gauss method would perform better than the iterative equation solver, because the LU factorization is done only once. On the contrary, iterative solvers have to repeat for each RHS. So, the more RHS vectors the problem contains, the more advantage the blocked Gauss method will show.

With the QMM technique and nonuniform element partition presented in Wang et al. [162], matrix $\boldsymbol{A}_0$ becomes a blocked sparse matrix with small diagonal blocks. Thus, the blocked Gauss method has high efficiency for solving the original linear system.

7.5 Overall Algorithm Flow and Discussion

The algorithm flow of the two-step approach for frequency-dependent substrate extraction is as follows:

Step One

1. Make QMM cutting, and form the discretized DBEM equations for the imaginary structure. All r parameters in (7.6) set to -1.
2. Generate and condense the system (7.8) with the reduction technique [162].
3. Generate $\boldsymbol{U}$ from $\boldsymbol{A}_0$; solve for $\boldsymbol{X}_0 = \boldsymbol{A}_0^{-1}\boldsymbol{B}$, $\boldsymbol{X}_U = \boldsymbol{A}_0^{-1}\boldsymbol{U}$ by blocked Gauss.

Step Two

4. For each frequency, extract the Y parameters.

   (4a) Form the sparse diagonal matrix $\boldsymbol{V}$, and calculate $\boldsymbol{V}\boldsymbol{X}_0$.

(4b) Collect and assemble a small matrix $\boldsymbol{M}$ from $\boldsymbol{I} + \boldsymbol{V}\boldsymbol{X}_U$.
(4c) Solve a linear system with matrix $\boldsymbol{M}$ and the right-hand sides composed of corresponding rows from $\boldsymbol{V}\boldsymbol{X}_0$.
(4d) Reorganize the above solution to get $(\boldsymbol{I} + \boldsymbol{V}\boldsymbol{X}_U)^{-1}\boldsymbol{V}\boldsymbol{X}_0$ as Theorem 7.1 directs.
(4e) Calculate $\boldsymbol{X} = \boldsymbol{X}_0 - \boldsymbol{X}_U \cdot (\boldsymbol{I} + \boldsymbol{V}\boldsymbol{X}_U)^{-1}\boldsymbol{V}\boldsymbol{X}_0$ and in turn the $\boldsymbol{Y}$ matrix.

Now let's find the memory complexity. The first step runs only once for a given substrate, where the major work is to solve a real-valued linear system with an $N \times N$ coefficient matrix and $N_c + m$ RHS vectors, where m is the number of elements on "physical" interfaces. Since the nonzero entries of $\boldsymbol{X}_U$ are in m columns, the extra memory usage is $O(N(N_c + m))$. The item (2) in the first step is optional and only performed for the substrate model with planar contacts. In the second step, a $m \times m$ complex-valued linear system with N_c RHS vectors is solved for each frequency. Since m is usually small, the frequency-dependent equation can be solved with LU factorization.

Note that the two-step approach is based on the exact Sherman-Morrison-Woodbury formula; therefore, it does not suffer any loss of accuracy.

Actually, the two-step approach has more applications. Since only the f in DBEM matrix involve the physical properties of mediums (see Fig. 7.3), the stored $\boldsymbol{X}_0$ and $\boldsymbol{X}_U$ can also be utilized for the extraction problem where only σ or ε parameters vary. This equals to fast extraction for structures with only frequency and/or medium parameters changed.

7.6 Numerical Results

The proposed method is called as subRCbem in C++. The efficiency will be demonstrated first, and then more validation cases follow. All experiments are run on a Sun Fire V880 server with CPU of 750 MHz, unless indicated explicitly.

7.6.1 Substrate with *52 Contacts*

Now let's examine the typical 52-contact substrate again. Its structure is described in Sect. 6.3.1, where the layout size is 128 μm $\times$ 128 μm. The layout is combined with a low-resistivity (LR) and a high-resistivity (HR) substrate profiles (see Figs. 3 and 4 in [102]). The extraction results of subRCbem are compared with those of ASITIC [101]. In ASITIC, considering the Neumann boundary condition at the magnetic walls, the Green's function is expanded in cosine series. Thus, the fast Fourier transform (FFT) can be applied to accelerate the summation of cosine series [102]. The method implemented in ASITIC is described in Niknejad et al. [102] and Niknejad and Meyer [103], and here it is referred to as the DCT-accelerated Green's function method.

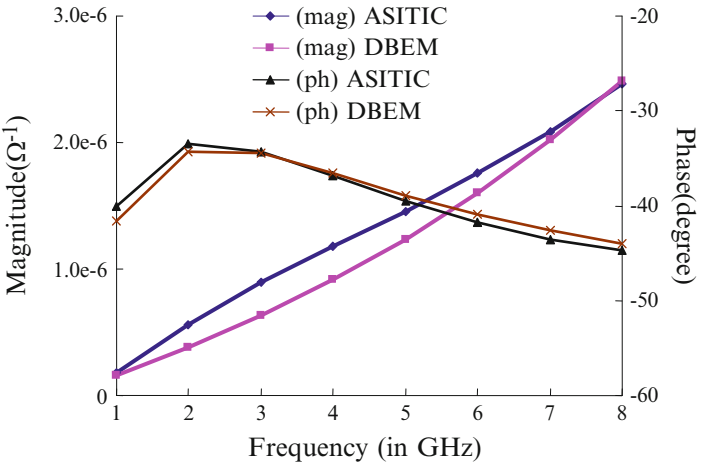

Fig. 7.7 Magnitude and phase of coupling admittance between contacts 26 and 52

Table 7.1 Efficient parameters for calculating HR profile example

Program	subRCbem	ASITIC		
# Contacts	52	52		
# Panels	7252	483	860	4352[a]
Memory (MB)	60	24	32	310
PreCal Time (s)	421	556	571	656
Time (s)	9.0	1930	9000	N/A

[a]Parameters corresponding to the default mesh setting in ASITIC

For this substrate with HR profile, the whole admittance (Y) matrix is extracted at frequencies from 1 to 8 GHz. The coupling admittances between contacts 26 and 52 (two close contacts at the top-right corner) are depicted in Fig. 7.7. Both results obtained with ASITIC and subRCbem show the same trends of magnitude and phase. The discrepancy between them is within 4 % for phase and 15 % for magnitude. These are also true for the other Y parameters.

The relevant computational parameters are shown in Table 7.1. "PreCal Time" means the CPU time in the first step of subRCbem or the preprocess for generating the Green's function in ASITIC. Both preprocessing times are comparable. The "Time" means the CPU time for a single-frequency point. subRCbem takes only 9.0 s at each frequency, once the first step was performed. On the contrary, the computational time of ASITIC with its default mesh is too long to be obtained. Coarser meshes must be set for ASITIC. Even with only 483 panels, ASITIC takes 1,930 s to finish a single frequency, which is 214 times more than subRCbem. Note that the comparison is similar when the substrate is with LR profile.

Let's analyze why subRCbem is so superior. Main reason is that only a small linear equation system is solved at each frequency. With the techniques of QMM, matrix reduction, and blocked Gauss solver, the preprocessing of subRCbem also

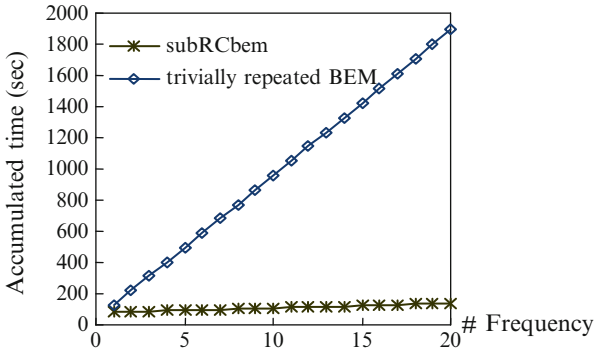

Fig. 7.8 Elapsed CPU time of subRCbem and repeated BEM for multiple frequencies

runs very fast, even though the number of unknowns is pretty large. On the contrary, ASITIC needs to build and solve (for each frequency) a complex-valued linear system, preceded by extensive computation of Green's function.

To further demonstrate the advantage, we compare subRCbem with the trivially repeated BEM method [160]. The repeated BEM is also based on DBEM and employs the technique of QMM and matrix reduction, but without the two-step algorithm. Actually, it's already optimized in term that it reuses the frequency-independent entries of matrix $\boldsymbol{A}$ but solves the whole linear system (7.5) at each frequency. For 20-frequency computing on a Linux sever with 3-GHz CPU, the accumulated CPU time is shown in Fig. 7.8. The elapsed CPU time of repeated BEM increases dramatically along with number of frequency points, because the linear system solution is very computationally expensive but repeated at each frequency. It is interesting that the time of subRCbem for 20 frequencies is even less than that of the trivially repeated BEM for the first frequency. One reason is that the real-valued linear system in subRCbem is solved efficiently with the blocked Gauss method, while the trivially repeated BEM has to solve a complex-valued linear system. As we know, complex floating numbers operate multiple times slower than real ones.

7.6.2 More Numerical Experiments

Another test case is a single-contact substrate in Niknejad et al. [102]. The DCT-accelerated Green's function method was also employed. The comparison between subRCbem and the results from [102] is in Table 7.1. It's clear that the maximum discrepancy is only 2.0 %, which demonstrates good accuracy matching.

The last case is a substrate with noise reduction components, where two contacts are surrounded by two components with buried materials, respectively. The effect of noise reduction is evaluated with the resistivity of buried region changing. Since the first step in subRCbem is only related with geometry parameters and need

not be performed again while the resistivity changed, subRCbem consumes less computational time for this simulation task. With the resistivity of buried region decreasing from 0.12 $\Omega \cdot$m to 5×10^{-6} $\Omega \cdot$m, computational results show that the contact coupling impedance increases for orders of magnitude. This means, with the conductive ability increasing, the noise reduction effect becomes prominent.

7.7 Summary

In order to efficiently compute coupling impedance of substrate at multiple to many frequencies, a two-step approach based on DBEM is presented. The first step basically equals to extracting substrate resistance, which is frequency independent. The other is to modify the resistance solution to get impedance parameters at any frequency, in a quick and exact way. Since the first step is done only once and the second step is very quick, the overall efficiency is remarkable. What's more, it doesn't suffer any accuracy loss due to its basement on the exact Sherman-Morrison-Woodbury formula. Numerical results show that this method is hundreds of times faster than a DCT-accelerated Green's function method [102] while preserving excellent accuracy.

Chapter 8
Process Variation-Aware Capacitance Extraction

Interconnect parasitic extraction, as a critical step in circuit design and verification, is now facing the new challenges of process variations. The geometric variation of interconnects has a great impact on parasitic parameters, which in turn greatly affects circuit performance. It is estimated that RC extraction without considering the process variations can cause error from 15 to 30 % in circuit performance [116]. Therefore, process variations must be taken into account during the parasitic extraction.

In this chapter, we present two works of capacitance extraction which take the process variation into account. The first work accelerates the library-building procedure for full-chip capacitance extraction. This is very useful to the corner-based methodology considering the process variation and applicable to the current commercial extraction flow. The second work models the spatially correlated random process variations. With the statistical Hermite polynomial collocation techniques, our method significantly accelerates the Monte Carlo method for the chip-level capacitance extraction. The statistical capacitance extraction possesses higher accuracy than the corner-based methodology. For the variations with larger magnitude, the statistical extraction method should be used to avoid the overpessimistic results got from the corner-based method.

8.1 Motivation

Although a lot of field-solver techniques [180] have been proposed for accurate capacitance extraction, they are still time-consuming and cannot replace the pattern-matching-based full-chip extraction methodology. For full-chip capacitance extraction, the geometric patterns of interconnects for a given technology are firstly generated and extracted with accurate field solvers. Then, the analytical formulas or lookup tables are generated to store the pattern capacitances [3]. This is the library-building procedure [67]. At the stage of layout capacitance extraction, the

W. Yu and X. Wang, *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*, DOI 10.1007/978-3-642-54298-5_8,

interconnect structures are decomposed into small structures, each of which is enclosed by a window. A procedure of pattern matching is carried out for each window to get the capacitances with the analytical formulas or lookup tables. With the capacitances of intra-window structures, the distributed RC circuit can be generated for the subsequent timing analysis. Because of the locality property of the electrostatic field, this methodology has moderate accuracy and is very efficient for large-scale extraction tasks. Since a large amount of patterns are needed, the library-building procedure is extremely time-consuming.

Recently, the variation of the process parameters, such as line width, thickness, spacing, etc. [87, 90], caused by the uncertainties in silicon fabrication becomes large. To take them into account, the capacitances corresponding to the structures with the minimal value and maximal value (also called "corner") of a process parameter need to be calculated. This *multi-corner* consideration increases the number of patterns for extraction for several tens of times. Therefore, how to improve the field-solver algorithm to reduce the time for library building becomes crucial. In Sect. 8.2, we propose an incremental boundary element method (BEM) to tackle this problem.

Generally, the process variations on chip can be classified into systematic variations and random variations [166]. The systematic variations are often pattern-dependent and can be modeled with some deterministic methods [116, 197], while the random variations need a stochastic modeling methodology. The random variation is mainly due to process uncertainty and results in geometric variations of on-chip interconnect and dielectric. Another important character of on-chip variation is the spatial correlation [166]. The spatially correlated multivariate Gaussian distribution is often assumed for the geometry parameter variations [41, 71, 198, 201, 202].

For the statistical variation-aware parasitic extraction, a straightforward approach is the Monte Carlo method, which suffers from huge computational cost with thousands of stochastic samplings. A non-sampling method named FastSies was proposed in Zhu et al. [198, 201] for variation-aware capacitance extraction. However, this method only produces the mean value and standard deviation (Std) of capacitance and is focused on the off-chip rough-surface effect. Another work combines the Neumann expansion and Hermite expansion to model the variation-aware impedance [41], mainly for off-chip interconnects with rough surface. These statistical methods for off-chip capacitance extraction cannot be directly applied to on-chip interconnect. The rough-surface effect is not prominent for on-chip interconnect. The surface of on-chip interconnect is relatively smooth, and the process variation has stronger spatial correlation. On the other hand, since the capacitance extraction is followed by statistical circuit analysis, an explicit linear or quadratic expression of the statistical capacitance is often necessary.

A number of statistical capacitance extraction methods have been proposed for on-chip interconnects. Based on a 3-D BEM capacitance solver, a conversion method was proposed in Jiang et al. [71] to generate a quadratic model for the on-chip capacitance variation. Another method, the spectral stochastic collocation method (SSCM), was proposed in Zhu et al. [202] with the same on-chip

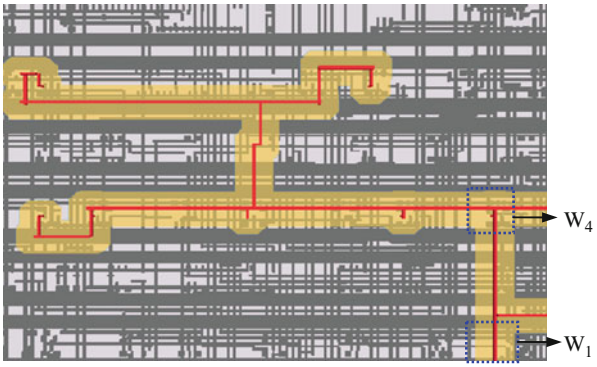

Fig. 8.1 A portion of the chip-level interconnect layout, where a net and its neighbors for capacitance extraction are *highlighted*. W_1 and W_4 are two of the extraction windows

interconnect variation model. In Shen et al. [130], a spectral stochastic method using the Hermite polynomial chaos is proposed to generate the quadratic capacitance model. It only considers the first-order variation of potential coefficients, so that it is efficient but not promising on the accuracy. Methods in Jiang et al. [71], Shen et al. [130], and Zhu et al. [202] model the process variations as the fluctuation of conductor surface panels. With this detailed variation model, the number of variables d can be very large even after applying variable reduction techniques. The methods generating quadratic capacitance models may have less computational advantage compared to the Monte Carlo method, because their computational time is $O(d^2)$ times of that for running 3-D field solver. Another deficiency of the existing methods is that the adopted variation model does not consider the variations of wire spacing, interlayer dielectric (ILD) thickness, etc. In Bi et al. [20], Labun [81], Qu et al. [117], and Ren et al. [119], the methods for sensitivity calculation of on-chip capacitance were proposed, most of which corresponds to a linear statistical model. The method in Labun [81] is based on analytical formulas, while the others are based on BEM or FEM. And, neither of them considers the spatial correlation.

It should be pointed out that except [81], all above methods are only able to handle small structures. This constraint makes them not suitable for the chip-level task of capacitance extraction. In actual chip-level capacitance extraction, each signal net and its neighbor regions are partitioned into small window structures (e.g., W_1 and W_4 in Fig. 8.1), and each window is extracted separately [180]. With the extracted capacitances, the distributed parasitic circuit is generated for the subsequent timing analysis or circuit simulation. If the total capacitance of a critical net is needed, the capacitances obtained within the windows along it need to be summed up [6, 134]. For each window, a field-solver or table-lookup method is employed to extract the capacitances. To consider the process variations in chip-level capacitance extraction, not only the statistical capacitance model within the window is needed, but the spatial correlation of capacitances from different windows should be considered. None of the existing methods considers the spatial

correlation of capacitances for actual chip-level extraction. The total or coupling capacitances of the whole net is called the *full-path capacitance*, in this chapter.

In Sects. 8.3 and 8.4, we present the statistical capacitance extraction techniques to consider the on-chip spatially correlated process variations, based on a grid-based variation model. A chip-level statistical extraction framework is proposed based on the window partitioning strategy. For each window, the technique of Hermite polynomial collocation (HPC), which is a generalized version of SSCM in Zhu et al. [202], is used to generate the linear or quadratic stochastic capacitance model. Then, the technique for calculating capacitance covariance between different windows is proposed to consider the spatial correlation. Finally, the algorithm for statistical modeling of full-path capacitances is proposed, which produces the mean value, the standard deviation (Std), and the explicit expression of capacitance.

8.2 The Incremental BEM for Variation-Aware Capacitance Library Building

The procedure of library building for capacitance extraction involves various interconnect patterns describing the basic 3-D geometry structures. Figure 8.2 shows a typical pattern, in which there are three metal layers and the line width (W), spacing (S), and the density of conductors in the upper and lower layer (D) are the pattern parameters. When changing these parameters, for example, W can have values 0.2 μm, 0.4 μm, and 0.8 μm, and D has values between 0 and 1, a series of patterns are generated. A 3-D numerical method is then employed to calculate the capacitance of these interconnect patterns.

Below, we take the pattern in Fig. 8.2 as an example to present the incremental BEM for variation-aware library building.

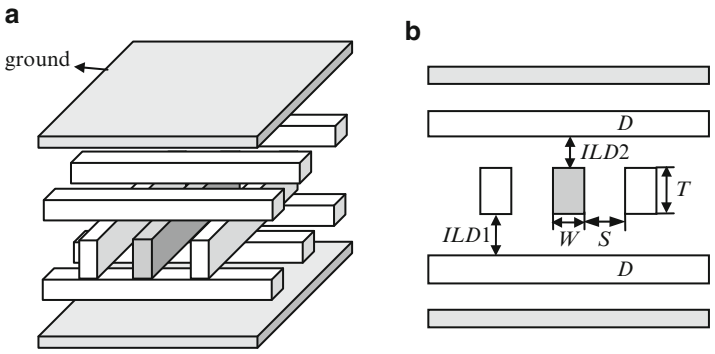

Fig. 8.2 A typical interconnect pattern. (**a**) 3-D view and (**b**) cross-sectional view

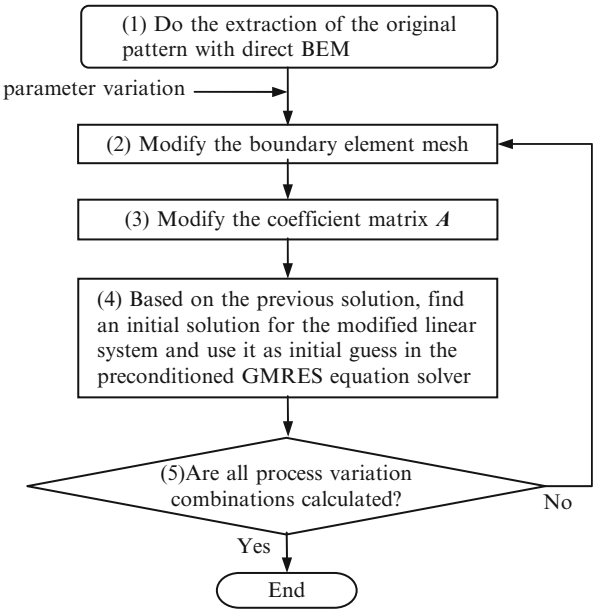

Fig. 8.3 Steps of incremental method for extracting the pattern capacitances considering process variation

8.2.1 Basic Idea

Three parameters with variation are considered. They are the line width (W), spacing (S), and thickness (T), each of which varies from -10 % to 10 % with respect to the nominal value. If three corner values (with variation ratio -10 %, 0, 10 %) are considered, there are $3^3 = 27$ patterns generated from each original pattern. Here we assume the three conductors in the active layer vary to the same extent. If the conductors are considered individually, much more patterns will need to be calculated in the variation-aware library-building procedure. Below, an incremental method based on direct BEM [179, 181] is presented for this task.

Generally, there are five steps in the 3-D capacitance extraction with direct BEM:

1. Load the process and layout data of a pattern.
2. Generate the 3-D geometry structures of the pattern.
3. Generate the boundary element mesh.
4. Form the linear equation system $\boldsymbol{Ax} = \boldsymbol{f}$.
5. Solve the system and get the result capacitance.

If the process variation is considered, the calculation steps for the patterns derived from an original pattern are shown in Fig. 8.3. The difference between the proposed method and the straightforward method which just repeats the computation for all derived patterns is on steps (3) and (4).

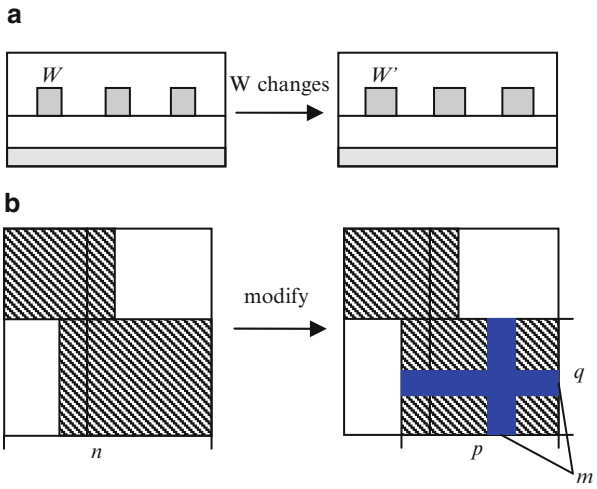

Fig. 8.4 An example to show process variation and corresponding modification on the coefficient matrix. (**a**) Line-width changes (cross-section view). (**b**) Corresponding modification on matrix

8.2.2 *Modification of the Coefficient Matrix and the Solving Technique*

With the sorting approach proposed in Yu et al. [181], the discretized boundary integral equations for all medium regions form a sparse coefficient matrix with regular distribution of nonzero blocks. The entry in the matrix is an integral of a source point-related function, on a boundary element. So, if the source point and boundary element are not changed, the corresponding coefficient will not change. As shown in Fig. 8.4a, only a small portion of geometry changes in the varied pattern. Therefore, the coefficient matrix for the original pattern can be reused with just modifying the entries corresponding to the changed boundary elements. With suitable arrangement, these modified matrix entries are located in the blue regions in Fig. 8.4b, where n is the dimension of the matrix and p and q are the numbers of columns and rows of the downright nonzero block, respectively. m is the width of the blue bands, whose value equals to the number of changed boundary elements. It is revealed that the modification of coefficient matrix is very limited, and a lot of time will be saved if only the changed matrix entries are calculated for the varied patterns.

Solving linear system $\boldsymbol{Ax} = \boldsymbol{f}$ is another time-consuming step. And, the value of initial guess largely affects the convergence rate of the preconditioned GMRES solver. An effective strategy is proposed to choose the initial value, which consists of two steps:

- Choose the solution of the linear system of the original pattern as $\boldsymbol{x}_0^{'}$.
- According to the difference between the original pattern and the varied pattern, make a heuristic modification to $\boldsymbol{x}_0^{'}$. This gives the initial value $\boldsymbol{x}_0$ for solving the linear equation system corresponding to the varied pattern.

Table 8.1 The parameters for the three patterns in experiments

Pattern no.	Pattern parameters (μm)	# boundary elements	# conductors
1	W0.20_S0.21_D0.5	3,997	47
2	W0.80_S0.80_D0.7	5,923	61
3	W1.20_S4.00_D1.0	9,157	85

Table 8.2 Computational results of the three patterns with variations on single parameter

Pattern no.	Parameter variation	Time t_0 (s)	Time t_1 (s)	Ratio of changed boundary elements (%)	Speedup t_0/t_1	Error (%)
1	$W + 10\ \%$	9.56 (38)	2.36 (11)	8.4	**4.05**	−0.11
2	$W + 10\ \%$	19.71 (41)	4.1 (11)	4.8	**4.81**	−0.08
3	$W + 10\ \%$	61.59 (54)	11.78 (13)	3.5	**5.23**	−0.15
1	S-10 %	9.6 (38)	2.34 (11)	5.6	**4.10**	−0.05
2	S-10 %	19.84 (41)	3.86 (10)	3.1	**5.14**	0.07
3	S-10 %	61.8 (54)	12.42 (14)	2.0	**4.98**	−0.12
1	$T + 10\ \%$	9.55 (38)	2.05 (12)	2.0	**4.66**	−0.09
2	$T + 10\ \%$	19.94 (42)	3.81 (11)	1.2	**5.23**	0.09
3	$T + 10\ \%$	61.77 (55)	11.12 (13)	0.9	**5.55**	0.01

The same problem was considered in Ye et al. [173], where a generalized Krylov recycling method was proposed to accelerate the iterative GMRES solution procedure. The results had shown that the method in Ye et al. [173] achieved a speedup of 5X–30X for solving multiple-related linear equation systems arisen from building the capacitance libraries.

8.2.3 Numerical Results

Three patterns with structure in Fig. 8.2 are used for numerical experiments. The pattern parameters are shown in Table 8.1; for example, "W0.20_S0.21_D0.5" means the line width is 0.2 μm, the line space is 0.21 μm, and the density in the upper and lower layer is 0.5.

For each pattern, only one of the three parameters (W, S, T) varies and the other two kept unchanged. The corresponding computational results are shown in Table 8.2, where t_0 is the computing time for the original pattern and t_1 is the time

for the pattern with variation. From the table we can see, the computing time of our method is about 1/5 of that by computing directly for the varied pattern. At the same time, the computational error is less than 0.2 %. The numbers in the brackets are the iteration number, which shows the initial value in our method accelerates the GMRES convergence greatly.

In practice, a series of patterns with the variation of several parameters need to be considered. For each of the three parameters (W, S, T), if five corners of variation (-10 %, -5 %, 0 %, 5 %, 10 %) are considered, there are 125 patterns derived from the original pattern. How to arrange the order of calculating these patterns is important. Our strategy is to make the difference between two sequential patterns as small as possible. For example, the variation combinations on three parameters can be ordered as a sequence like (0 %, 0 %, 0 %) $\rightarrow$ (5 %, 0 %, 0 %) $\rightarrow$ (10 %, 0 %, 0 %) $\rightarrow$ (10 %, 5 %, 0 %) $\rightarrow$ (10 %, 10 %, 0 %) ....

We have done the experiment of calculating the series of varied patterns, based on the second pattern in Table 8.1. Among the whole 125 patterns in the sequence, only the first one is calculated with the original BEM, while the others are calculated with the incremental BEM. The total time spent by our method is 425 s, while 2463 s are needed if simply repeating the original BEM. It should be pointed out that the speedup ratio in this experiment is larger than those in Table 8.2, because the variation gap (5 %) is smaller which results in fewer elements changed between two adjacent computations.

8.3 Preliminaries of Variation-Aware Statistical Capacitance Extraction

In this section, we firstly introduce the grid-based stochastic model for the on-chip process variations. Then, the Hermite polynomial collocation method for statistical capacitance extraction is presented.

8.3.1 Grid-Based Process Variation Model

Figure 8.2b shows the cross-sectional view of an interconnect structure for extraction. This structure can be characterized with several geometric parameters, and most of them are subject to process variation. From the capacitance extraction point of view, the variability of these geometric parameters is the cause of capacitance variation. So, we refer to these parameters as variation sources. In this example, the wire width, wire spacing, wire thickness, and ILD thickness are the variation sources. A geometric parameter F can be expressed as

$$F = F_0 + F_s + F_r, \tag{8.1}$$

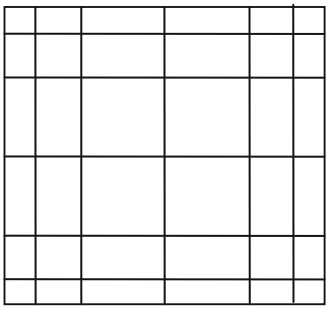

Fig. 8.5 A nonuniform grid for on-chip variation modeling

where F_0, F_s, and F_r denote the nominal value, systematic variation, and random variation of F, respectively. With the systematic variation determined [116, 197], F can be regarded as the sum of a nominal value and a zero-mean random variable F_r, which is the main factor considered.

For the random variation, an important feature is spatial correlation [166]. One model for variation with spatial correlation involves a set of grid cells imposed on the whole chip area. The chip is dissected into grid cells of different sizes, and the random process variation is considered the same within each cell. The way of gridding can be decided with the knowledge of manufacturing process. For example, Fig. 8.5 shows a gridding scheme, where the center grid cells are coarser than those at boundary [166]. The random variable F_r may have different Std at different grid cells and is spatially correlated among the cells. The spatial correlation is characterized by the covariance between two variables from the same source. As in Jiang et al. [71], Zhu et al. [198], Zhu and White [201], and Zhu et al. [202], we assume the joint spatial variation of the same physical parameter follows a multivariate Gaussian process with respect to their physical locations on the chip. Then, the correlation function between cell i and cell j is

$$\rho_{i,j} = \exp\left(-\left|\overrightarrow{r}_i - \overrightarrow{r}_j\right|^2/\eta^2\right), \tag{8.2}$$

where η is the correlation length and the positions $\overrightarrow{r}_i$ and $\overrightarrow{r}_j$ are of the center points of cell i and cell j, respectively. The covariance between variables F_i in cell i and F_j in cell j is given by

$$\mathrm{cov}\left(F_i, F_j\right) = \rho_{i,j}\sigma_{F_i}\sigma_{F_j}, \tag{8.3}$$

where σ_{F_i} and σ_{F_j} are the Std of F_i and F_j, respectively. Other models of the geometry variation and the technique to extract the spatial correlation can be found in Xiong et al. [166]. Note that different variation source may have different variation Std and correlation length, as well as the grid setting.

Different extraction windows, such as W_1 and W_4 in Fig. 8.1, often involve geometry variables from the same source. So, there exists covariance of capacitances from different windows, which is the result of the spatial correlation.

8.3.2 The Hermite Polynomial Collocation Method

For statistical circuit analysis, it is desirable to express the variational capacitance as a stochastic function with respect to random variables. For various applications, the first-order (linear) or second-order (quadratic) capacitance expression is employed. In Jiang et al. [71], a conversion method was proposed for statistical capacitance extraction. This method is based on the system equations derived from BEM [99]:

$$\boldsymbol{P}\boldsymbol{q} = \boldsymbol{v}, \tag{8.4}$$

where $\boldsymbol{P}$ is the potential coefficient matrix and $\boldsymbol{q}$ and $\boldsymbol{v}$ are panel charge distribution and potential vectors, respectively. The conversion method first formulates a second-order expression for the coefficient matrix $\boldsymbol{P}$ and then converts it to the expression of $\boldsymbol{q}$ by solving the perturbation equation:

$$(\boldsymbol{P} + \Delta\boldsymbol{P})(\boldsymbol{q} + \Delta\boldsymbol{q}) = \boldsymbol{v}. \tag{8.5}$$

In Jiang et al. [71], the second-order Taylor expansion is used for the stochastic representation of $\boldsymbol{P}$, and finally the quadratic capacitance expression is obtained. The conversion method includes invoking $O(d^2)$ times of nonstatistical capacitance extraction for the quadratic model, where d is the number of independent variables.

Different from the Taylor expansion, the homogenous chaos expansion expands a stochastic function $f(\boldsymbol{\xi})$ with

$$f(\boldsymbol{\xi}) = a_0\Psi^0 + \sum_{i_1=1}^{d} a_{i_1}\Psi^1(\xi_{i_1}) + \sum_{i_1=1}^{d}\sum_{i_2=1}^{i_1} a_{i_1 i_2}\Psi^2(\xi_{i_1}, \xi_{i_2}) + \ldots, \tag{8.6}$$

where $\boldsymbol{\xi} = [\xi_1, \xi_2, \ldots, \xi_d]^{\mathrm{T}}$ is a set of independent random variables and Ψ^i stands for an orthogonal polynomial with the ith order. If the random variables follow the Gaussian distribution $N(0,1)$, the Hermite polynomials are used, whose general expression is [113]

$$\Psi^k(\xi_{i_1}, \ldots, \xi_{i_k}) = \exp\left(\frac{1}{2}\sum_{j=1}^{k}\xi_{i_j}^2\right)(-1)^k\frac{\partial^k}{\partial\xi_{i_1}\ldots\partial\xi_{i_k}}\exp\left(-\frac{1}{2}\sum_{j=1}^{k}\xi_{i_j}^2\right) \tag{8.7}$$

Specifically, the Hermite polynomials below order 3 are

$$\Psi^0 = 1, \quad \Psi^1(\xi_i) = \xi_i, \quad \Psi^2(\xi_i, \xi_j) = \xi_i \xi_j - \delta_{i,j}, \tag{8.8}$$

where $\delta_{i,j}$ is the Kronecker delta function. The Hermite polynomial expansion is guaranteed to converge for any Gaussian random process with finite second-order moments [48]. Moreover, the Askey principle [167] shows that the expansion has the optimal convergence rate. In El-Moselhy and Daniel [41], the Hermite polynomial expansion is used to substitute the Taylor expansion for representing the coefficient matrix. Based on the Hermite polynomial expansion, a spectral stochastic collocation method (SSCM) was proposed for the quadratic capacitance model [202], which does not deal with the expansion of the coefficient matrix $\boldsymbol{P}$. The SSCM has the same order of computational complexity as the conversion methods but achieves higher accuracy.

The SSCM in Zhu et al. [202] combines the Hermite polynomial expansion and the spectral collocation method. It is based on a variation model considering the fluctuation of conductor surface panels, and thus the BEM-based nonstatistical capacitance extraction and the techniques reducing the number of random variables are involved. Here, we adopt the idea of combining the Hermite polynomial expansion and the spectral collocation method, which is referred as Hermite polynomial collocation (HPC) method hereafter.

Suppose the statistical capacitance $C(\boldsymbol{\xi})$ is expanded with the Hermite polynomials. If the terms with order larger than k are truncated, and the polynomials are consistently labeled in ascending order, we have

$$C(\boldsymbol{\xi}) = \sum_{i=1}^{M} c_i \Psi_i(\boldsymbol{\xi}), \tag{8.9}$$

where Ψ_i is the ith Hermite polynomials after the ordering and the number of terms

$$M = \binom{d+k}{k} = \frac{(d+k)!}{k!d!}. \tag{8.10}$$

Specifically, the zero-order Hermite polynomial is $\Psi_1 = \Psi^0 = 1$.

The Gaussian-weighted inner product is defined as

$$< X, Y >_{\mathcal{P}} = E(XY) = \int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} X(\boldsymbol{\xi}) Y(\boldsymbol{\xi}) \mathcal{P}(\boldsymbol{\xi}) \mathrm{d}\xi_1 \dots \mathrm{d}\xi_d, \tag{8.11}$$

where $E(\cdot)$ denotes the mathematical expectation and $\mathcal{P}(\boldsymbol{\xi})$ is the *probability density function* (PDF) of the following multivariate Gaussian distribution with independent $N(0,1)$ components:

$$\mathcal{P}(\boldsymbol{\xi}) = \left(\frac{1}{\sqrt{2\pi}}\right)^d \exp\left(-\frac{1}{2}\sum_{i=1}^{d}\xi_i^2\right). \tag{8.12}$$

Based on the definition in (8.11), the Hermite polynomials are orthogonal to each other:

$$< \Psi_i(\boldsymbol{\xi}), \Psi_j(\boldsymbol{\xi}) >_{\mathcal{P}} = h_i \delta_{i,j}. \tag{8.13}$$

Here h_i is a positive constant, and

$$h_i = < \Psi_i(\boldsymbol{\xi}), \Psi_i(\boldsymbol{\xi}) >_{\mathcal{P}} = E\left(\Psi_i^2(\boldsymbol{\xi})\right). \tag{8.14}$$

With the spectral collocation method, the coefficients in (8.9) are evaluated with Zhu et al. [202]

$$c_i = \frac{< C(\boldsymbol{\xi}), \Psi_i(\boldsymbol{\xi}) >_{\mathcal{P}}}{< \Psi_i(\boldsymbol{\xi}), \Psi_i(\boldsymbol{\xi}) >_{\mathcal{P}}} = \frac{< C(\boldsymbol{\xi}), \Psi_i(\boldsymbol{\xi}) >_{\mathcal{P}}}{h_i}, \quad i = 1, \ldots, M. \tag{8.15}$$

According to (8.11),

$$< C(\boldsymbol{\xi}), \Psi_i(\boldsymbol{\xi}) >_{\mathcal{P}} = \int_{-\infty}^{\infty} \ldots \int_{-\infty}^{\infty} C(\boldsymbol{\xi}) \Psi_i(\boldsymbol{\xi}) \mathcal{P}(\boldsymbol{\xi}) \, \mathrm{d}\xi_1 \ldots \mathrm{d}\xi_d. \tag{8.16}$$

The integral can be calculated with the technique of Gaussian-Hermite quadrature [113]. For one-dimensional (1-D) integral, the Gaussian-Hermite quadrature utilizes the formula

$$\int_{-\infty}^{+\infty} f(x)\mathcal{P}(x)\mathrm{d}x = \sum_{j=0}^{l} w_j^{(l)} f\left(x_j^{(l)}\right), \tag{8.17}$$

where $x_j^{(l)}$ and $w_j^{(l)}$ are the level-l collocation points and corresponding weights, respectively, which can be obtained with the method in Press et al. [113]. The level-l Gaussian-Hermite quadrature achieves degree $2l+1$ of exactness. We define the set of collocation points for d-dimensional polynomial as Θ_d^k and the corresponding weight as W_d^l, and then for 1-D integral, we have $\Theta_1^l = \{x_0^{(l)}, \ldots, x_l^{(l)}\}$ and $W_1^l = \{w_0^{(l)}, \ldots, w_l^{(l)}\}$. A straightforward approach to compute Θ_d^k and W_d^l is to use collocation points obtained from the direct tensor product of Θ_1^l's for all dimensions. However, it suffers from the "curse of dimensionality," i.e., the number of collocation points increases exponentially with respect to the dimension number d.

To improve the computational efficiency, the number of collocation points can be reduced with the sparse-grid quadrature [104]. For 1-D integral, the sparse-grid technique is the same as the Gaussian-Hermite quadrature. While for the

d-dimensional integral (8.16), the collocation point set Θ_d^k for the sparse-grid technique is constructed from a combination of tensor products of Θ_1^l's with level l less than k:

$$\Theta_d^k = \bigcup_{k+1-d \le |i| \le k} \left(\Theta_1^{i_1} \times \cdots \times \Theta_1^{i_d} \right), \tag{8.18}$$

where $\times$ indicates the tensor product, $|i| = \sum_{j=1}^{d} i_j$, and k is the level of accuracy. The corresponding weights are

$$w_{j_1,\ldots,j_d}^{(i_1,\ldots,i_d)} = (-1)^{k-|i|} \binom{d-1}{k-|i|} \prod_{m=1}^{d} w_{j_{i_m}}^{(i_m)}, \tag{8.19}$$

where $w_{j_{i_m}}^{(i_m)}$ is the Gaussian-Hermite weight for 1-D integral and $0 \le j_{i_m} \le i_m$, $m = 1, 2, \ldots, d$. It is proved that the level-k sparse-grid quadrature has degree $2k+1$ of exactness, the same as the Gaussian-Hermite quadrature [104], and the number collocation points

$$N_c \approx \frac{2^k}{k!} d^k = O\left(d^k\right), \tag{8.20}$$

which has much lower complexity than the multidimensional Gaussian-Hermite quadrature.

Once the collocation points and weights are determined, (8.15) becomes

$$c_i = \frac{\sum_{j=0}^{N_c-1} w_j C\left(\boldsymbol{\xi}^{(j)}\right) \Psi_i\left(\boldsymbol{\xi}^{(j)}\right)}{h_i}, \tag{8.21}$$

where $\boldsymbol{\xi}^{(j)}$ and w_j stand for the collocation points and corresponding weights for the d-dimensional sparse-grid quadrature, respectively. Since $\boldsymbol{\xi}^{(j)}$ is a deterministic quantity, $C(\boldsymbol{\xi}^{(j)})$ can be calculated with a nonstatistical capacitance solver.

8.4 Chip-Level Statistical Capacitance Extraction Considering Spatial Correlation

For chip-level capacitance extraction, the window technique is necessary to limit problem size. In this section, the techniques for statistical chip-level extraction are proposed. They are based on the grid-based spatially correlated variation model and assemble the covariance of capacitances from different windows to obtain the statistical full-path capacitance. Both linear and quadratic capacitance models are considered.

8.4.1 Intra-window Capacitance Extraction with the Grid-Based Variation Model

The HPC method is employed to generate the statistical capacitances for each intra-window structure. With the grid-based variation model, we ignore the detailed random fluctuation of conductor surfaces, so that unlike [71] and [202], the BEM is not necessary to be used. In the HPC-based extraction, the $C(\boldsymbol{\xi}^{(j)})$ can be obtained with any nonstatistical capacitance solver, even the most efficient table-lookup method [3]. This is a prominent advantage of the HPC technique for the grid-based variation model.

We further give three remarks regarding the sparse-grid-accelerated HPC technique used in this work.

Remark 8.1 If the approximation (8.9) is an order-k polynomial, the integrand in (8.16) is a polynomial with order $2k$. Thus, the sparse-grid quadrature of level k is accurate enough for generating the order-k model of statistical capacitance, due to its degree $2k+1$ of exactness. For the linear and quadratic capacitance model, the level 1 and level 2 sparse grids are therefore employed, and they have $2d+1$ and $2d^2+3d+1$ collocation points, respectively.

Remark 8.2 The numbers of collocation points for the d-dimensional Gaussian-Hermite quadrature are 2^d and 3^d for the linear and quadratic capacitance models, respectively. It is easy to find out that if the number of variables d is less than 3, the number of collocation points for sparse-grid quadrature is larger than that for Gaussian-Hermite quadrature. For example, if $d=2$, the point numbers are 5 and 14 for level 1 and 2 sparse-grid quadrature, respectively. In contrast, the Gaussian-Hermite quadrature only needs 4 and 9 points. In this case, the Gaussian-Hermite quadrature is preferred.

Remark 8.3 The set of sparse-grid points generated with (8.18) and (8.19) may contain some identical points with different weights. For example, the level 2 sparse grid for 3 variables contains 4 instances of point (0, 0, 0). Realizing this can reduce the times of invoking nonstatistical capacitance extraction.

To use the HPC method for the intra-window extraction, a preprocessing step must be performed in advance, to convert the correlated variables to independent variables. This is because the generation of variation grid is irrelevant to the partition of extraction window. So, a window may overlap more than one grid cell, which produces correlated variables.

Theorem 8.1 *For a set of correlated Gaussian variables $\boldsymbol{\zeta}$, if the covariance matrix $\Delta n(\boldsymbol{\zeta}) = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}$, then $\boldsymbol{\xi} = \boldsymbol{L}^{-1}\boldsymbol{\zeta}$ is a set of independent variables with $N(0,1)$ distribution.*

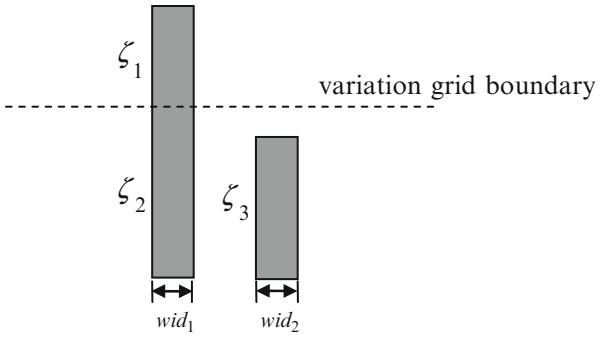

Fig. 8.6 An example for variable preprocessing

Proof According to the definition of covariance matrix,

$$\begin{aligned}
\Delta n(\boldsymbol{\xi}) &= E\left\{[\boldsymbol{\xi} - E(\boldsymbol{\xi})][\boldsymbol{\xi} - E(\boldsymbol{\xi})]^{\mathrm{T}}\right\} \\
&= E\left\{\left[\boldsymbol{L}^{-1}\boldsymbol{\zeta} - E\left(\boldsymbol{L}^{-1}\boldsymbol{\zeta}\right)\right]\left[\boldsymbol{L}^{-1}\boldsymbol{\zeta} - E\left(\boldsymbol{L}^{-1}\boldsymbol{\zeta}\right)\right]^{\mathrm{T}}\right\} \\
&= E\left\{\boldsymbol{L}^{-1}[\boldsymbol{\zeta} - E(\boldsymbol{\zeta})][\boldsymbol{\zeta} - E(\boldsymbol{\zeta})]^{\mathrm{T}}\boldsymbol{L}^{-\mathrm{T}}\right\} \\
&= \boldsymbol{L}^{-1} \cdot E\left\{[\boldsymbol{\zeta} - E(\boldsymbol{\zeta})][\boldsymbol{\zeta} - E(\boldsymbol{\zeta})]^{\mathrm{T}}\right\} \cdot \boldsymbol{L}^{-\mathrm{T}} \\
&= \boldsymbol{L}^{-1} \cdot \Delta n(\boldsymbol{\zeta}) \cdot \boldsymbol{L}^{-T} \\
&= \boldsymbol{L}^{-1}\boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}\boldsymbol{L}^{-\mathrm{T}} \\
&= \boldsymbol{I}.
\end{aligned}$$

Since the covariance matrix of $\boldsymbol{\xi}$ is the identity matrix, $\boldsymbol{\xi}$ is a set of independent variables with $N(0,1)$ distribution. ■

The Cholesky factorization can be performed to decompose the covariance matrix of process variations, which is always symmetric positive definite in nature. With the Cholesky factor $\boldsymbol{L}$, we can relate a set of correlated Gaussian variables $\boldsymbol{\zeta}$ to a set of variables $\boldsymbol{\xi}$ with $N(0,1)$ distribution as stated in Theorem 8.1.

In practice, the variables are divided into subsets according to the variation source they belong to. Each subset of variables is processed separately. With the Cholesky factorization performed for each subset, the resulting lower triangular matrices are then combined along the diagonal to get the overall factor matrix $\boldsymbol{L}$.

Figure 8.6 gives an example for a simple layout, where we assume the widths of two wires are the only variation source and the variation grid yields three variables.

Suppose wid_1 and wid_2 have the Std of σ_{w1} and σ_{w2}, respectively, and the random variables ζ_1 and ζ_2 have correlation coefficient $\rho_{1,2}$. With the Cholesky factorization, we have

$$\text{cov}\,(\zeta_1, \zeta_2) = \sigma_{w1}^2 \begin{bmatrix} 1 & \rho_{1,2} \\ \rho_{1,2} & 1 \end{bmatrix} = \boldsymbol{L}_1 \boldsymbol{L}_1^{\mathrm{T}},$$

where $\boldsymbol{L}_1 = \sigma_{w1} \begin{bmatrix} 1 & 0 \\ \rho_{1,2} & \sqrt{1-\rho_{1,2}^2} \end{bmatrix}$. Therefore,

$$\boldsymbol{\zeta} = \begin{bmatrix} \zeta_1 \\ \zeta_2 \\ \zeta_3 \end{bmatrix} = \begin{bmatrix} \sigma_{w1} & 0 & 0 \\ \sigma_{w1}\rho_{1,2} & \sigma_{w1}\sqrt{1-\rho_{1,2}^2} & 0 \\ 0 & 0 & \sigma_{w2} \end{bmatrix} \begin{bmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \end{bmatrix} = \boldsymbol{L}\boldsymbol{\xi}. \tag{8.22}$$

With (8.22), we translate the HPC collocation point for the independent variable $\boldsymbol{\xi}$ to the value of actual geometry variables $\boldsymbol{\zeta}$. Then, with (8.21) and a nonstatistical capacitance solver, the coefficients of the Hermite polynomial expansion are obtained. Suppose the kth capacitance inside window i is represented by

$$C_{i,k} = \sum\nolimits_{j=1}^{M_i} c_{i,k,j} \Psi_j \left(\boldsymbol{\xi}_{(i)}\right), \tag{8.23}$$

where $\boldsymbol{\xi}_{(i)}$ denotes the vector of independent variables for window i. The mean value and variance (the square of Std) of $C_{i,k}$ are

$$\begin{cases} E\,(C_{i,k}) = c_{i,k,1} \\ D\,(C_{i,k}) = E\left[(C_{i,k})^2\right] - \left[E\,(C_{i,k})\right]^2 = \sum\nolimits_{j=2}^{M_i} h_j^2 c_{i,k,j}^2 \end{cases}. \tag{8.24}$$

The algorithm for statistical capacitance extraction within a window is as follows.

Algorithm 8.1 Intra-Window Capacitance Extraction (W_i)

Preprocess variables within window W_i, to get the Cholesky factor matrix $\boldsymbol{L}_i$, relating the physical variables $\boldsymbol{\zeta}_{(i)}$ to a set of independent variables $\boldsymbol{\xi}_{(i)}$;
Generate sparse-grid collocation points $\{\boldsymbol{\xi}^{(j)}\}$ for $\boldsymbol{\xi}_{(i)}$.
For each $\boldsymbol{\xi}^{(j)}$
  Convert $\boldsymbol{\xi}^{(j)}$ to physical variable values $\boldsymbol{\zeta}^{(j)}$ with (8.22) and construct a sample geometry structure.
  Solve the sample structure to get sample capacitances $\boldsymbol{C}_i(\boldsymbol{\xi}^{(j)})$.
EndFor
For each desired capacitance $C_{i,k}$
  Evaluate the coefficients $c_{i,k,j}$ in (8.23) using (8.21).
  Evaluate the mean value and variance of $C_{i,k}$ with (8.24).
EndFor

8.4.2 Calculation of Inter-window Capacitance Covariance

The capacitances from different windows may be correlated, due to the spatial correlation of geometry parameters. Suppose the intra-window capacitance is expressed by (8.23), the covariance of capacitances can be evaluated according to the linearity of covariance:

$$\operatorname{cov}\left(C_{i,k}, C_{j,l}\right) = \sum_{p=1}^{M_i} \sum_{q=1}^{M_j} c_{i,k,p} c_{j,l,q} \operatorname{cov}\left[\Psi_p\left(\boldsymbol{\xi}_{(i)}\right), \Psi_q\left(\boldsymbol{\xi}_{(j)}\right)\right], \tag{8.25}$$

where $C_{i,k}$ and $C_{j,l}$ are represented by M_i and M_j Hermite polynomials, respectively. $\boldsymbol{\xi}_{(i)}$ and $\boldsymbol{\xi}_{(j)}$ are the variable sets in windows i and j, respectively. Now, the problem becomes calculating the covariance between two Hermite polynomials.

The Hermite polynomials with order below three are given in (8.8). The zero-order polynomial Ψ_1 is a constant and orthogonal to any other polynomial. So, the covariances between Ψ_1 and other polynomials are all 0. Thanks to the symmetry of covariance, the polynomial pairs in (8.25) are of six types if the quadratic model is considered:

$$\begin{cases} \operatorname{cov}\left(\xi_{(i)a}, \xi_{(j)b}\right) \\ \operatorname{cov}\left(\xi_{(i)a}^2 - 1, \xi_{(j)b}^2 - 1\right) \\ \operatorname{cov}\left(\xi_{(i)a}\xi_{(i)c}, \xi_{(j)b}\xi_{(j)d}\right) \\ \operatorname{cov}\left(\xi_{(i)a}, \xi_{(j)b}^2 - 1\right) \\ \operatorname{cov}\left(\xi_{(i)a}, \xi_{(j)b}\xi_{(j)d}\right) \\ \operatorname{cov}\left(\xi_{(i)a}^2 - 1, \xi_{(j)b}\xi_{(j)d}\right) \end{cases}. \tag{8.26}$$

Here the subscripts a and c indicate different variables for window i, and b and d indicate different variables for window j. Except for $\operatorname{cov}(\xi_{(i)a}, \xi_{(j)b})$, each type of covariance should be expressed with the covariances of variable pairs. Note that if only the linear capacitance model is considered, only the first term in (8.26) is present.

Before introducing the technique for calculating the covariances in (8.26), we present the following theorem.

Theorem 8.2 *If Ψ_p and Ψ_q are two Hermite polynomials of random variables with $N(0,1)$ distribution, and neither of their orders is 0, then*

$$\operatorname{cov}\left(\Psi_p, \Psi_q\right) = E\left(\Psi_p \Psi_q\right). \tag{8.27}$$

Proof According to the definition of covariance,

$$\operatorname{cov}\left(\Psi_p, \Psi_q\right) = E\left(\Psi_p \Psi_q\right) - E\left(\Psi_p\right) E\left(\Psi_q\right).$$

Due to the orthogonality of Hermite polynomials,

$$E\left(\Psi_p\right) = E\left(\Psi_p \Psi_1\right) = <\Psi_p, \Psi_1>_{\mathcal{P}} = 0, \quad p > 1.$$

So, (8.27) is proved. ■

The following theorem converts the covariances in (8.26) to the covariances of single variables.

Theorem 8.3 *If $\boldsymbol{\xi}_{(i)}$ and $\boldsymbol{\xi}_{(j)}$ are the independent variable sets for windows i and j, respectively, then*

$$\operatorname{cov}\left(\xi_{(i)a}^2 - 1, \xi_{(j)b}^2 - 1\right) = 2\operatorname{cov}\left(\xi_{(i)a}, \xi_{(j)b}\right)^2, \tag{8.28}$$

$$\begin{aligned}\operatorname{cov}\left(\xi_{(i)a}\xi_{(i)c}, \xi_{(j)b}\xi_{(j)d}\right) &= \operatorname{cov}\left(\xi_{(i)a}, \xi_{(j)b}\right)\operatorname{cov}\left(\xi_{(i)c}, \xi_{(j)d}\right) \\ &\quad + \operatorname{cov}\left(\xi_{(i)a}, \xi_{(j)d}\right)\operatorname{cov}\left(\xi_{(i)c}, \xi_{(j)b}\right),\end{aligned} \tag{8.29}$$

$$\operatorname{cov}\left(\xi_{(i)a}, \xi_{(j)b}^2 - 1\right) = 0, \tag{8.30}$$

$$\operatorname{cov}\left(\xi_{(i)a}, \xi_{(j)b}\xi_{(j)d}\right) = 0, \tag{8.31}$$

$$\operatorname{cov}\left(\xi_{(i)a}^2 - 1, \xi_{(j)b}\xi_{(j)d}\right) = 2\operatorname{cov}\left(\xi_{(i)a}, \xi_{(j)b}\right)\operatorname{cov}\left(\xi_{(i)a}, \xi_{(j)d}\right), \tag{8.32}$$

where a and c are different variable indices for window i and b and d are different indices for window j.

Proof According to Theorem 8.2, the covariance equals to the mean value of two polynomials' product. Then, utilizing the properties of $N(0,1)$ distribution and handling the special situation where variables from different windows are identical, the theorem can be proved. The complete proof is given as Appendix 8.A. ■

With Theorem 8.3, the covariances of Hermite polynomials in (8.25) are all expressed in terms of the variable covariances. Two variables in the independent variable sets $\boldsymbol{\xi}_{(i)}$ and $\boldsymbol{\xi}_{(j)}$, say $\xi_{(i)a}$ and $\xi_{(j)b}$, can be independent or correlated. If the two variables are associated with different physical variation sources, they are independent (covariance is 0). Otherwise, they are correlated. On the other hand, if the two variables represent the same physical variable, they are identical and their covariance is just 1. This scenario is illustrated with Fig. 8.7, where $\xi_{(1)2}$ is identical to $\xi_{(2)1}$ since they are both converted from ζ_2 in the preprocessing step.

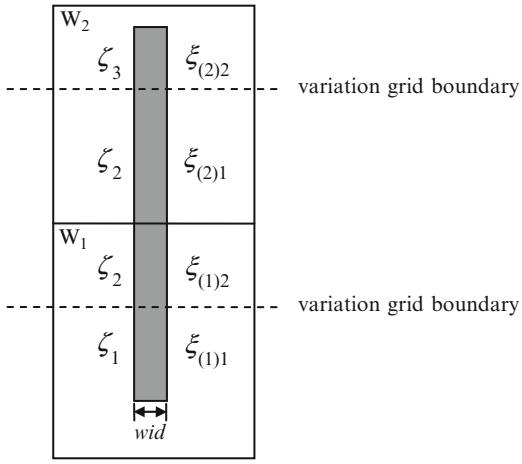

Fig. 8.7 An example to show that variables from different windows may be identical. The physical variables for width variation are labeled to the *left* of the wire, while the converted independent variables are to the *right*. $\xi_{(1)2}$ in W_1 is identical to $\xi_{(2)1}$ in W_2, since they both equal to ζ_2/σ_2

The relationship between the independent variables and physical variables are

$$\boldsymbol{\xi}_{(i)} = \boldsymbol{L}_i^{-1}\boldsymbol{\zeta}_{(i)}, \boldsymbol{\xi}_{(j)} = \boldsymbol{L}_j^{-1}\boldsymbol{\zeta}_{(j)}, \tag{8.33}$$

where $\boldsymbol{L}_i$ and $\boldsymbol{L}_j$ are the Cholesky factor matrices for window i and j, respectively. For two correlated variables $\xi_{(i)a}$ and $\xi_{(j)b}$, we then have

$$\begin{aligned}\operatorname{cov}\left(\xi_{(i)a}, \xi_{(j)b}\right) &= \operatorname{cov}\left(\sum_s g_{i,a,s}\zeta_{(i)s}, \sum_t g_{j,b,t}\zeta_{(j)t}\right) \\ &= \sum_s \sum_t g_{i,a,s} g_{j,b,t} \operatorname{cov}\left(\zeta_{(i)s}, \zeta_{(j)t}\right),\end{aligned} \tag{8.34}$$

where $g_{i,a,s}$ and $g_{j,b,t}$ denote the matrix entries of $\boldsymbol{L}_i^{-1}$ and $\boldsymbol{L}_j^{-1}$, respectively. Since both $\zeta_{(i)s}$ and $\zeta_{(j)t}$ are physical variables, $\operatorname{cov}(\zeta_{(i)s}, \zeta_{(j)t})$ can be easily calculated according to (8.3) in Sect. 8.3.1.

8.4.3 Complexity Analysis of the Inter-window Calculation

To calculate the inter-window capacitance covariance in (8.25), there are two steps of work. The first step is calculating and storing the covariances of two variables from different windows with (8.34). The other step is evaluating (8.25) with the stored variable covariances and Theorem 8.3. Below we analyze the computational complexity for both steps in turn.

Assume there are p variation sources. For example, $p = 6$ for the configuration shown in Fig. 8.2b. Suppose window i overlaps α_i variation grids for each variation source, on average. Therefore, the total variable number in window i is $d_i = p\alpha_i$, and

each variation source corresponds to α_i variables. Accordingly, the total variable number in window j is denoted by d_j, and $d_j = p\alpha_j$, where α_j is the average number of variation grids overlapped by window j.

Calculating and storing the covariances of two variables from window i and j involve all nonzero pairs of $\{\text{cov}(\xi_{(i)a}, \xi_{(j)b}),\ 1 \le a \le d_i \text{ and } 1 \le b \le d_j\}$. According to the independence of variation source, the number of covariances is at most $p\alpha_i\alpha_j$. Each one is calculated with (8.34). The $\boldsymbol{L}_i$ and $\boldsymbol{L}_j$ are block diagonal matrices due to the independence of variation source, and the blocks in them are of size α_i and α_j, respectively. Therefore, the twofold summation in (8.34) at most involves α_i and α_j nonzero items, respectively. The time complexity for calculating (8.34) is about $O(\alpha_i\alpha_j)$. The total time complexity for calculating and storing the variables covariances is about $O(p\alpha_i^2\alpha_j^2)$.

We remark that the inversing of $\boldsymbol{L}_i$ and $\boldsymbol{L}_j$ needed for (8.34) can be performed on $2p$ matrix blocks, due to the block diagonal structure of the both matrices. The time complexity for inversing $\boldsymbol{L}_i$ and $\boldsymbol{L}_j$ is $O(p\alpha_i^3 + p\alpha_j^3)$, which is controlled by the complexity of calculating the variable covariances.

The second step is calculating (8.25). Due to Theorem 8.3 and the independence of variation sources, not all items in the twofold summation in (8.25) should be calculated. We only consider the covariances of polynomial pairs with nonzero value. They can be classified into the following four groups, according to the type of the two polynomials:

1. Two polynomials are in the forms of $\xi_{(i)a}$ and $\xi_{(j)b}$.
   The number of this kind of nonzero covariances in (8.25) is about $p\alpha_i\alpha_j$.
2. Two polynomials are in the forms of $\xi_{(i)a}^2 - 1$ and $\xi_{(j)b}^2 - 1$.
   According to (8.28), this type of covariance is nonzero if and only if $\text{cov}(\xi_{(i)a}, \xi_{(j)b})$ is nonzero. The number of this kind of nonzero covariances is about $p\alpha_i\alpha_j$.
3. Two polynomials are in the forms of $\xi_{(i)a}^2 - 1$ and $\xi_{(j)b}\xi_{(j)d}$ or $\xi_{(i)a}\xi_{(i)c}$ and $\xi_{(j)b}^2 - 1$.
   According to (8.32), this type of covariance is nonzero if and only if all the involved variables are associated with the same variation source. The number of this kind of nonzero covariances is about $p\alpha_i\alpha_j^2/2 + p\alpha_i^2\alpha_j/2$.
4. Two polynomials are in the forms of $\xi_{(i)a}\xi_{(i)c}$ and $\xi_{(j)b}\xi_{(j)d}$.
   According to (8.29), if the involved variables are all from the same variation source, this type of covariance is nonzero. Otherwise, it is nonzero only if $\{\xi_{(i)a}, \xi_{(j)b}\}$ and $\{\xi_{(i)c}, \xi_{(j)d}\}$ are of the same variation source, respectively, or $\{\xi_{(i)a}, \xi_{(j)d}\}$ and $\{\xi_{(i)c}, \xi_{(j)b}\}$ are of the same variation source, respectively. The number of this kind of nonzero covariances is about $\frac{p\alpha_i^2\alpha_j^2}{4} + \frac{p^2\alpha_i^2\alpha_j^2}{2}$.

Below we summarize the computational complexity for linear and quadratic capacitance models, respectively:

1. *Linear Capacitance Model:* In the second step, all the polynomials are in the forms of $\xi_{(i)a}$ and $\xi_{(i)b}$. Thus, the overall computational complexity is that of the first step, $O(p\alpha_i^2\alpha_j^2)$.

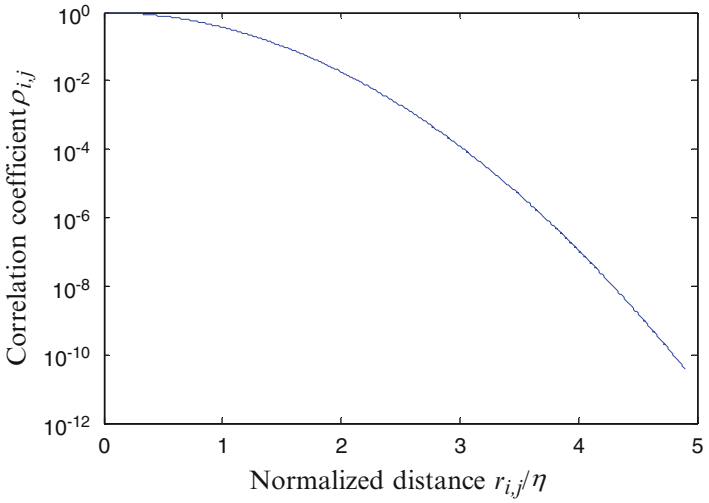

Fig. 8.8 The correlation coefficient decreases exponentially with the normalized distance

2. *Quadratic Capacitance Model:* In the second step, the number of nonzero covariances of polynomial pairs is dominated by $O(p^2\alpha_i^2\alpha_j^2)$. Each polynomial covariance is calculated with the simple expression of (8.28), (8.29), or (8.32), where the variable covariances have been calculated in advance. So, the total time complexity for the second step is $O(p^2\alpha_i^2\alpha_j^2)$, which is also the overall computational complexity.

For comparison, we consider the product of variable numbers in both windows, which is $d_i d_j = p^2\alpha_i\alpha_j$. Assume that α_i and α_j are small numbers, which means the extraction window is not much larger than variation grid cell. It is realized that the computational complexity of calculating an inter-window covariance approximates $O(d_i d_j)$, for both linear and quadratic models. With the sparse-grid-accelerated HPC method, generating the statistical capacitance model for window i involves $O(d_i)$ and $O(d_i^2)$ times of nonstatistical extraction, for linear and quadratic models, respectively. Because each nonstatistical capacitance extraction has time complexity much higher than d_i operations, the inter-window calculation is much faster than the intra-window capacitance extraction.

The above analysis only considers a pair of windows. Due to the fast attenuation of the correlation function, we can ignore the covariance of two variables physically far away from each other. This will reduce the window pairs considered in the inter-window calculation. Figure 8.8 shows the attenuation of the typical correlation function (8.2), with the variable distance increasing. According to Fig. 8.8, the correlation coefficient decays to about 10^{-4} when the distance between two grid cells is 3η. Therefore, it is safe to discard the window pairs with distance larger than 3η, which results in a sparse covariance matrix for the distributed capacitances.

8.4.4 Statistical Model of Full-Path Capacitance

For full-path capacitance extraction, the total capacitance of a critical net and its coupling capacitances to neighbor nets are calculated by assembling the capacitances from related windows [134]. To emphasize the treatment of statistical capacitance model, we only consider a simple assembling technique, i.e., the full-path capacitance is simply the sum of related window capacitances. This assumption does not prevent the proposed method from collaborating with other windowing techniques.

Along a critical net k, we generate the extraction windows. We use $S(k)$ to denote the set of these windows' indices, and $S(l) \subseteq S(k)$ is its subset indicating the windows containing a neighbor net l. For each window, Algorithm 8.1 is performed to obtain the total capacitance of net k and its coupling capacitances. Suppose in window i, the local index of net k's total capacitance is $I(i,k,k)$, while the local index of the coupling capacitance between net k and net l is $I(i,k,l)$. With our assumption, the coupling capacitance between net k and l is

$$\tilde{C}_{k,l} = \sum_{i \in S(l)} C_{i,I(i,k,l)}. \tag{8.35}$$

If $l = k$, (8.35) becomes the total capacitance of net k:

$$\tilde{C}_{k,k} = \sum_{i \in S(k)} C_{i,I(i,k,k)}. \tag{8.36}$$

The mean value and variance of capacitance $\tilde{C}_{k,l}$ can be calculated with

$$E\left(\tilde{C}_{k,l}\right) = \sum_{i \in S(l)} E\left(C_{i,I(i,k,l)}\right), \tag{8.37}$$

$$\begin{aligned} D\left(\tilde{C}_{k,l}\right) = D\left(\sum_{i \in S(l)} C_{i,I(i,k,l)}\right) &= \sum_{i \in S(l)} D\left(C_{i,I(i,k,l)}\right) \\ &+ 2 \sum_{i,j \in S(l), i \neq j} \operatorname{cov}\left(C_{i,I(i,k,l)}, C_{j,I(j,k,l)}\right). \end{aligned} \tag{8.38}$$

Thus, substituting the mean values and covariances calculated with (8.24) and (8.25), we obtain the mean value and variance of the full-path capacitance.

Considering the attenuation of correlation function, we can ignore the covariance item representing two windows far from each other. This will reduce the number of capacitance covariances and thus the expense of calculating (8.38).

To obtain the explicit expression of capacitance, the random variables in the related windows must be merged to get an overall independent variable set $\boldsymbol{\xi}$. Without loss of generality, we assume a full-path capacitance:

$$\tilde{C}_x = \sum_{i=1}^{N_w} C_i, \tag{8.39}$$

where C_i stands for statistical capacitance expression in window i and shows how to obtain the explicit capacitance expression for $\tilde{C}_x$.

For each window i, we have

$$\boldsymbol{\zeta}_{(i)} = \boldsymbol{L}_i \boldsymbol{\xi}_{(i)}, \tag{8.40}$$

where $\boldsymbol{\zeta}_{(i)}$ is the physical variables and $\boldsymbol{L}_\mathrm{i}$ is the Cholesky factor matrix. Suppose all $\{\boldsymbol{\zeta}_{(i)}, 1 \le i \le N_w\}$ are merged to form an overall physical variable set $\boldsymbol{\zeta}$. The two quantities have the following relationship:

$$\boldsymbol{\zeta}_{(i)} = \boldsymbol{J}_i \boldsymbol{\zeta}, \tag{8.41}$$

where $\boldsymbol{J}_i$ is a $d_i \times N$ adjacent matrix, which is very sparse, and N is the dimension of $\boldsymbol{\zeta}$. With Theorem 8.1, the overall independent variable set can be generated, and we have

$$\boldsymbol{\zeta} = \boldsymbol{L}\boldsymbol{\xi}, \tag{8.42}$$

where $\boldsymbol{L}$ is the Cholesky factor and $\Delta n(\boldsymbol{\zeta}) = \boldsymbol{L}\boldsymbol{L}^\mathrm{T}$. With (8.40) and (8.42), we derive

$$\boldsymbol{\xi}_{(i)} = \boldsymbol{L}_i^{-1} \boldsymbol{J}_i \boldsymbol{L} \boldsymbol{\xi}. \tag{8.43}$$

Note that N may be less than the sum of the dimensions d_i of $\boldsymbol{\zeta}_{(i)}$, if there are some variables shared by different windows.

In window i, the intra-window statistical capacitance C_i can be written in the general second-order form:

$$C_i = c_i + \boldsymbol{a}_i^\mathrm{T} \boldsymbol{\xi}_{(i)} + \boldsymbol{\xi}_{(i)}^\mathrm{T} \boldsymbol{A}_i \boldsymbol{\xi}_{(i)}, \tag{8.44}$$

where c_i is the zero-order coefficient, $\boldsymbol{a}_i$ is the vector with the first-order coefficients, and $\boldsymbol{A}_i$ is the matrix with the second-order coefficients. Substituting (8.43) into (8.44) yields

$$C_i = c_i + \boldsymbol{b}_i \boldsymbol{\xi} + \boldsymbol{\xi}^\mathrm{T} \boldsymbol{B}_i \boldsymbol{\xi}, \tag{8.45}$$

where

$$\boldsymbol{b}_i = \boldsymbol{a}_i^\mathrm{T} \boldsymbol{L}_i^{-1} \boldsymbol{J}_i \boldsymbol{L} \quad \text{and} \quad \boldsymbol{B}_i = \boldsymbol{L}^\mathrm{T} \boldsymbol{J}_i^\mathrm{T} \boldsymbol{L}_i^{-\mathrm{T}} \boldsymbol{A}_i \boldsymbol{L}_i^{-1} \boldsymbol{J}_i \boldsymbol{L}. \tag{8.46}$$

And the quadratic model of $\tilde{C}_x$ is then obtained:

$$\tilde{C}_x = \sum_{i=1}^{N_w} c_i + \left(\sum_{i=1}^{N_w} \boldsymbol{b}_i\right)\boldsymbol{\xi} + \boldsymbol{\xi}^{\mathrm{T}}\left(\sum_{i=1}^{N_w} \boldsymbol{B}_i\right)\boldsymbol{\xi}. \tag{8.47}$$

Note that the second-order terms in (8.47) will be dropped if only the linear model is needed. Because the matrices $\boldsymbol{L}_i^{-1}$ has been generated for inter-window calculation, the Cholesky factorization for $\boldsymbol{L}$ is only performed once, and the dimensions of matrices in (8.46) are not very large, it will not consume much time to calculate (8.46) and (8.47). Once the explicit expression of the statistical full-path capacitance $\tilde{C}_x$ is obtained, its PDF or the *cumulative distribution function* (CDF) can be derived with the technique of characteristic function [71].

Finally, we give the algorithm description for the statistical full-path capacitance extraction.

Algorithm 8.2 Full-Path Capacitance Extraction (Net k)

Partition extraction windows along net k, and get $S(k)$.
For each window W_i, $i \in S(k)$,
  Run Algorithm 8.1: *Intra-Window Capacitance Extraction* (W_i).
EndFor
For each neighbor net l of net k, or $l=k$,
  For each pair of $i,j \in S(l)$,
    Compute $\mathrm{cov}(C_{i,I(i,k,l)}, C_{j,I(j,k,l)})$ with (8.25) and the technique in Sect. 8.4.2.
  EndFor
EndFor
For each desired full-path capacitance of net k,
  Get the mean value and variance of full-path capacitance with (8.37) and (8.38).
  If desired, derive the explicit capacitance expression with (8.47).
EndFor

8.5 Experiments of Statistical Capacitance Extraction

Several interconnect structures with window partition are tested with the proposed method for statistical chip-level capacitance extraction. For simplicity, the windows are partitioned to be nonoverlapping, and the variation grid is the same for different variation sources. The Monte Carlo (MC) simulation with 10,000 samples is performed, whose results are regarded as the golden values, as in Jiang et al. [71]. The proposed method is implemented with MATLAB, which generates the deterministic sample structures and calculates the statistical intra-window capacitances and full-path capacitances. The FastCap 2.0 [97, 99] is used to extract the sample structures for each window. All experiments are carried out on a Linux server with 3GHz CPU.

Table 8.3 The computational results for the first case

			Total cap err.		Coupling cap err.	
Model	Collocation points	Time (s)	Mean (%)	Std (%)	Mean (%)	Std (%)
Linear	7	4.18	−0.10	−1.00	−0.06	−1.31
Quadratic	25	15.0	−0.03	−0.66	0.04	−0.70

8.5.1 Simple Cases with Parallel-Line Structure

The first case, containing two parallel lines dissected by ten windows evenly, is used to demonstrate the efficiency of the proposed method and the necessity of considering the capacitance covariance among windows. Each line is of 40 μm length and 1 μm width and height, and the spacing between them is 2 μm. The variation grid is assumed to coincide with the window boundaries for this case. The physical variables are set to be one variable for line height and two variables for the line widths. The Stds of variables are all set to 0.2 μm, while the correlation length is set to 8 μm for all variation sources. Since each window overlaps one grid cell, there are three random variables for intra-window extraction. The full-path capacitances of one line are calculated by summing up capacitances from ten windows, and both linear and quadratic statistical models are considered for intra- and inter-window calculation. The relative errors, compared with the MC simulation, on the mean value and Std of the capacitances are listed in Table 8.3. The computing time for each model and the number of sparse-grid collocation points in HPC method for extracting each window are also listed. Note that the number of collocation points for quadratic model is evaluated considering the Remark 8.3 in Sect. 8.4.1.

From Table 8.3, we can see that both linear and quadratic models achieve high accuracy. Both errors on mean and Std are less than 2 %. Since this case includes ten windows, 10,000-times MC simulation for the whole structure involves invoking FastCap for 100,000 times. The total CPU time of MC simulation is 5,867 s. This means, to generate the quadratic model, the proposed method is 391× faster. The speedup ratio for the linear model to the MC simulation is about 1,404. With the proposed method, calculating the linear and quadratic models need 70 and 250 runs of FastCap, respectively. The ratio of times of invoking FastCap approximates the actual CPU time ratio of the proposed method to MC simulation, because the computing time for other steps in the proposed method is much less than that for running FastCap.

The CDF of total capacitance, computed by the proposed method (with linear and quadratic models), and MC simulation are shown in Fig. 8.9a. And, a zoom-in view is in Fig. 8.9b to distinguish between the linear model and quadratic model. In both figures, we also show the CDF of the statistical capacitance without considering the capacitance covariances among windows, which is labeled as “direct sum.” It is obvious that the capacitance variance is remarkably underestimated in the “direct

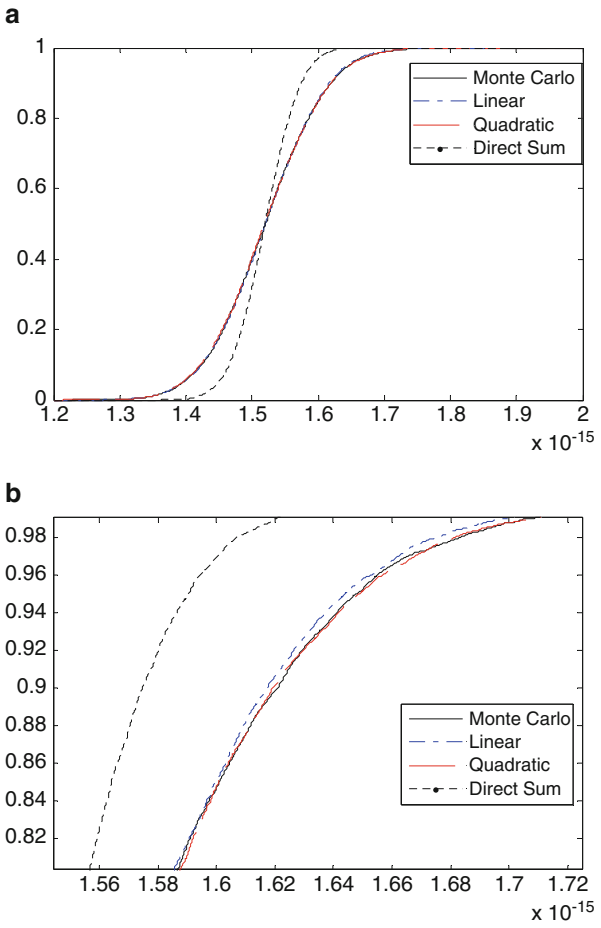

Fig. 8.9 (**a**) The cumulative distribution function (*CDF*) curves for the total capacitance. (**b**) A zoom-in view of the CDF curves

sum" approach. So, considering and calculating the correlation among the window capacitances are very important.

The second case has the same structure as the first one, but the extraction window overlaps two variation grid cells like that shown in Fig. 8.7. This makes the number of random variables increase to six variables in each window. The computational time and error of both linear and quadratic models are listed in Table 8.4. The results still show good accuracy of the proposed method for both models. The CPU time of the MC simulation is 6515 s. Due to the increase of variable number, the speedup ratios of the proposed method to the MC simulation become 762 and 115 for the linear and quadratic model, respectively.

Table 8.4 The computational results for the second case

			Total cap err.		Coupling cap err.	
Model	Collocation points	Time (s)	Mean (%)	Std (%)	Mean (%)	Std (%)
Linear	13	8.49	−0.12	−0.90	−0.10	−1.74
Quadratic	85	56.5	0.06	−0.44	0.06	−0.32

Table 8.5 The computational results for the third case

			Total cap err.		Coupling cap err.	
Model	Collocationpoints	Time (s)	Mean (%)	Std (%)	Mean (%)	Std (%)
Linear	2	1.19	0.04	−3.45	0.08	−3.09
Quadratic	3	1.80	−0.02	−0.69	−0.07	−0.83

Table 8.6 The computational results for the fourth case

	Collocation points					
Model	Normal	Shift	Time (s)	Speedup to MC	Mean err. (%)	Std err. (%)
Linear	13	21	121	716	0.01	−1.89
Quadratic	85	221	866	100	−0.07	−0.83

The third case is still the parallel-line structure, but with the line spacing reduced to 1 μm, and the Std of random variable is reduced to 0.1 μm accordingly. To highlight the second-order effect of statistical capacitance, we set the spacing to be the only variation source in this case. The computational results are listed in Table 8.5. Since there is only one variable for each window, the Gaussian-Hermite quadrature is used in the HPC method, as suggested by Remark 8.2 in Sect. 8.4.1. The results show the linear model has relatively large error on Std of statistical capacitances. From this case we can see that the quadratic model is needed for densely routed interconnects.

8.5.2 A Large Case with Multilayered Structure

Finally, a practical multilayered interconnect structure shown in Fig. 8.10 is tested as the fourth case. The critical net shifts from the middle layer to the upper layer with an orthogonal turn. This structure is partitioned with eight normal windows where the critical net lies in one layer and one shift window where the net lies in two layers. The whole size of the structure is about 50 μm × 50 μm. The variation sources are chosen like that shown in Fig. 8.2b, and the normal and shift windows include six and ten variables, respectively. The Stds of all kinds of variations are set to 0.1 μm, and the correlation lengths are all 8 μm. The computational results are listed in Table 8.6. From it we can see the high accuracy of the proposed method, and its speedup ratios to MC simulation are 716 and 100, for the linear and quadratic model, respectively.

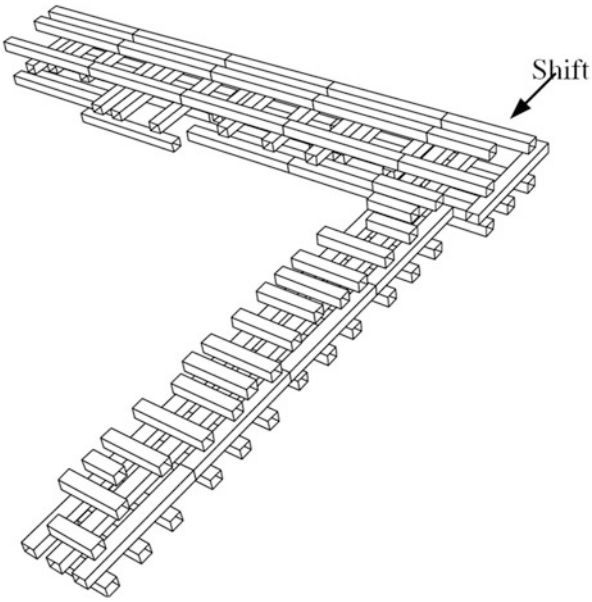

Fig. 8.10 A practical multilayered interconnect structure, including a critical net and its neighbor nets

8.6 Summary

For the corner-based extraction methodology considering the process variation, an incremental BEM algorithm is presented to accelerate the library-building procedure for full-chip capacitance extraction. By reusing the coefficient matrix and the solution of nearby pattern, this method achieves more than 5X speedup with negligible error, for the capacitance extraction of a series of varied interconnect patterns.

A practical framework for chip-level capacitance extraction considering spatially correlated random variations is proposed. It considers the covariance of capacitances from different windows and the statistical assembly of window capacitances. Both linear and quadratic stochastic models are built for intra-window and inter-window capacitances, and the efficient technique for generating the quadratic model for full-path capacitance is proposed. Numerical results demonstrate the proposed method is hundred times faster than the MC simulation with 10,000 samples and is suitable for the chip-level extraction tasks.

Appendix 8.A. Complete Proof of Theorem 8.3

Before proving Theorem 8.3, we firstly give three Lemmas.

Lemma 8.A.1 *For a Gaussian random variable ξ with $N(0,1)$ distribution, the mean values of powers of ξ are*

$$E(\xi) = 0, \quad E\left(\xi^2\right) = 1, \quad E\left(\xi^3\right) = 0, \quad E\left(\xi^4\right) = 3. \tag{8.A.1}$$

Lemma 8.A.1 states the basic property of the $N(0,1)$ random distribution.

Lemma 8.A.2 *For a set of independent variable $\boldsymbol{\xi} = [\xi_1, \xi_2, \ldots, \xi_d]^{\mathrm{T}}$ with $N(0,1)$ distribution, we have*

$$E[\xi_1 \varphi(\xi_2, \ldots, \xi_d)] = E\left[\xi_1^3 \varphi(\xi_2, \ldots, \xi_d)\right] = 0, \tag{8.A.2}$$

$$E\left[\xi_1^2 \varphi(\xi_2, \ldots, \xi_d)\right] = E[\varphi(\xi_2, \ldots, \xi_d)], \tag{8.A.3}$$

$$E\left[\xi_1^4 \varphi(\xi_2, \ldots, \xi_d)\right] = 3E[\varphi(\xi_2, \ldots, \xi_d)], \tag{8.A.4}$$

where $\varphi(\cdot)$ is an arbitrary function.

Proof We firstly prove (8.A.3). Since $\boldsymbol{\xi}$ is a set of independent variables,

$$\begin{aligned}
E\left[\xi_1^2 \varphi(\xi_2, \ldots, \xi_d)\right] &= \int_{\xi_d} \cdots \int_{\xi_1} \xi_1^2 \varphi(\xi_2, \ldots, \xi_d) \mathcal{P}(\boldsymbol{\xi}) \, \mathrm{d}\xi_1 \ldots \mathrm{d}\xi_d \\
&= \int_{\xi_d} \ldots \int_{\xi_2} \varphi(\xi_2, \ldots, \xi_d) \left[\int_{\xi_1} \xi_1^2 \mathcal{P}(\xi_1) \, \mathrm{d}\xi_1 \right] \times \\
&\quad \times \mathcal{P}(\xi_2, \ldots, \xi_d) \, \mathrm{d}\xi_2 \ldots \mathrm{d}\xi_d \\
&= \int_{\xi_d} \ldots \int_{\xi_2} \varphi(\xi_2, \ldots, \xi_d) E\left(\xi_1^2\right) \mathcal{P}(\xi_2, \ldots, \xi_d) \, \mathrm{d}\xi_2 \ldots \mathrm{d}\xi_d \\
&= E[\varphi(\xi_2, \ldots, \xi_d)].
\end{aligned}$$

The last equality is due to (8.A.1).

For (8.A.2) and (8.A.4), they can be proved in a similar way. ■

Lemma 8.A.3 *For a set of zero-mean Gaussian variables $\boldsymbol{\zeta} = [\zeta_1, \zeta_2, \ldots, \zeta_d]^{\mathrm{T}}$, if $k = \sum_{i=1}^{d} k_i$ equals 1 or 3,*

$$E\left(\zeta_1^{k_1} \zeta_2^{k_2} \ldots \zeta_d^{k_d}\right) = 0. \tag{8.A.5}$$

Proof From Theorem 8.1, we know $\boldsymbol{\zeta}$ can be converted to a set of independent variables $\boldsymbol{\xi}$ with $N(0,1)$ distribution through $\boldsymbol{\zeta} = \boldsymbol{L}\boldsymbol{\xi}$, where $\boldsymbol{L}$ is the Cholesky factor of matrix $\Delta n(\boldsymbol{\zeta})$. Then, each ζ_i can be expressed as a linear combination of $\{\xi_i\}$, and $\zeta_1^{k_1}\zeta_2^{k_2}\ldots\zeta_d^{k_d}$ can be substituted with $\{\xi_i\}$ to yield

$$\zeta_1^{k_1}\zeta_2^{k_2}\ldots\zeta_d^{k_d} = \sum_j a_j \xi_1^{l_{j,1}}\xi_2^{l_{j,2}}\ldots\xi_d^{l_{j,d}} \quad \text{and}$$

$$E\left(\zeta_1^{k_1}\zeta_2^{k_2}\ldots\zeta_d^{k_d}\right) = \sum_j a_j E\left(\xi_1^{l_{j,1}}\xi_2^{l_{j,2}}\ldots\xi_d^{l_{j,d}}\right). \tag{8.A.6}$$

We notice that $\sum_{i=1}^{d} l_{j,i} = \sum_{i=1}^{d} k_i$ for all j and equals to 1 or 3. Obviously, there is at least one odd-number power $l_{j,i}$, equal 1 or 3. Without loss of generality, we assume $l_{j,1} = 1$, then

$$E\left(\xi_1^{l_{j,1}}\xi_2^{l_{j,2}}\ldots\xi_d^{l_{j,d}}\right) = E\left(\xi_1\xi_2^{l_{j,2}}\ldots\xi_d^{l_{j,d}}\right) = 0,$$

according to Lemma 8.A.2. This means each summation item on the right-hand side of (8.A.6) is zero, so that (8.A.5) holds. ■

By the way, Eq. (8.A.5) actually holds for any k of odd number.

Proof of Theorem 8.3 With Theorem 8.2, below we prove (8.28, 8.29, 8.30, 8.31, and 8.32) one by one. Firstly, look at (8.30) and (8.31):

$$\operatorname{cov}\left(\xi_{(i)a}, \xi_{(j)b}^2 - 1\right) = E\left[\xi_{(i)a}\left(\xi_{(j)b}^2 - 1\right)\right] = E\left(\xi_{(i)a}\xi_{(j)b}^2\right) - E\left(\xi_{(i)a}\right) = 0,$$

$$\operatorname{cov}\left(\xi_{(i)a}, \xi_{(j)b}\xi_{(j)d}\right) = E\left(\xi_{(i)a}\xi_{(j)b}\xi_{(j)d}\right) = 0.$$

In either equation, the last equal mark is due to Lemma 8.A.3. Note that the involved variables in each equation are not a set of independent variable.

For (8.28),

$$\begin{aligned}\operatorname{cov}\left(\xi_{(i)a}^2 - 1, \xi_{(j)b}^2 - 1\right) &= E\left[\left(\xi_{(i)a}^2 - 1\right)\left(\xi_{(j)b}^2 - 1\right)\right] \\ &= E\left(\xi_{(i)a}^2\xi_{(j)b}^2\right) - E\left(\xi_{(i)a}^2\right) - E\left(\xi_{(j)b}^2\right) + 1.\end{aligned}$$

After applying Lemma 8.A.1, we have

$$\operatorname{cov}\left(\xi_{(i)a}^2 - 1, \xi_{(j)b}^2 - 1\right) = E\left(\xi_{(i)a}^2\xi_{(j)b}^2\right) - 1.$$

If $\xi_{(i)a}$ and $\xi_{(j)b}$ are identical, $E(\xi_{(i)a}^2\xi_{(j)b}^2) = E(\xi_{(i)a}^4) = 3$, and

$$\operatorname{cov}\left(\xi_{(i)a}^2 - 1, \xi_{(j)b}^2 - 1\right) = 2 = 2\operatorname{cov}\left(\xi_{(i)a}, \xi_{(j)b}\right)^2.$$

Otherwise, we convert $\xi_{(i)a}$ and $\xi_{(j)b}$ to two independent variables ξ_1 and ξ_2:

$$\Delta n\left(\xi_{(i)a}, \xi_{(j)b}\right) = \begin{bmatrix} 1 & x \\ x & 1 \end{bmatrix}, \text{and } \boldsymbol{L} = \begin{bmatrix} 1 & 0 \\ x & \sqrt{1-x^2} \end{bmatrix},$$

where $x = \text{cov}(\xi_{(i)a}, \xi_{(j)b})$. Then,

$$\begin{bmatrix} \xi_{(i)a} \\ \xi_{(j)b} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ x & \sqrt{1-x^2} \end{bmatrix} \begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix}, \begin{cases} \xi_{(i)a} = \xi_1 \\ \xi_{(j)b} = x\xi_1 + \sqrt{1-x^2}\xi_2 \end{cases}, \text{and}$$

$$\text{cov}\left(\xi_{(i)a}^2 - 1, \xi_{(j)b}^2 - 1\right) = E\left[\xi_1^2\left(x\xi_1 + \sqrt{1-x^2}\xi_2\right)^2\right] - 1.$$

Utilizing Lemma 8.A.1 and Lemma 8.A.2, we have

$$\text{cov}\left(\xi_{(i)a}^2 - 1, \xi_{(j)b}^2 - 1\right) = 3x^2 + \left(1 - x^2\right) - 1 = 2\,\text{cov}\left(\xi_{(i)a}, \xi_{(j)b}\right)^2.$$

This proves (8.28).
For (8.32),

$$\text{cov}\left(\xi_{(i)a}^2 - 1, \xi_{(j)b}\xi_{(j)d}\right) = E\left(\xi_{(i)a}^2\xi_{(j)b}\xi_{(j)d}\right) - E\left(\xi_{(j)b}\xi_{(j)d}\right) = E\left(\xi_{(i)a}^2\xi_{(j)b}\xi_{(j)d}\right),$$

where the last equal mark holds because $\{\xi_{(j)b}, \xi_{(j)d}\}$ is a set of independent variables with $N(0,1)$ distribution.

If $\xi_{(i)a}$ is identical to $\xi_{(j)b}$ or $\xi_{(j)d}$, (8.A.2) in Lemma 8.A.2 can be applied, and then $E(\xi_{(i)a}^2\xi_{(j)b}\xi_{(j)d}) = 0$. Otherwise, we can convert the variables to three independent variables $\{\xi_1, \xi_2, \xi_2\}$ and have

$$\begin{cases} \xi_{(j)b} = \xi_1 \\ \xi_{(j)d} = \xi_2 \\ \xi_{(i)a} = x_1\xi_1 + x_2\xi_2 + \sqrt{1 - x_1^2 - x_2^2}\xi_3 \end{cases},$$

where $x_1 = \text{cov}(\xi_{(i)a}, \xi_{(j)b}), x_2 = \text{cov}(\xi_{(i)a}, \xi_{(j)d})$. Then,

$$\begin{aligned} E\left(\xi_{(i)a}^2\xi_{(j)b}\xi_{(j)d}\right) &= E\left[\xi_1\xi_2\left(x_1\xi_1 + x_2\xi_2 + \sqrt{1 - x_1^2 - x_2^2}\xi_3\right)^2\right] \\ &= 2x_1x_2E\left(\xi_1^2\xi_2^2\right) \\ &= 2x_1x_2 \\ &= 2\,\text{cov}\left(\xi_{(i)a}, \xi_{(j)b}\right)\text{cov}\left(\xi_{(i)a}, \xi_{(j)d}\right). \end{aligned}$$

Note that if $\xi_{(i)a}$ is identical to $\xi_{(j)b}$ or $\xi_{(j)d}$, $\text{cov}(\xi_{(i)a}, \xi_{(j)b})\,\text{cov}(\xi_{(i)a}, \xi_{(j)d}) = 0$. So, for the both cases, $\text{cov}(\xi_{(i)a}^2 - 1, \xi_{(j)b}\xi_{(j)d}) = 2\,\text{cov}(\xi_{(i)a}, \xi_{(j)b})\,\text{cov}(\xi_{(i)a}, \xi_{(j)d})$, and (8.32) is proved.

Finally, we look at $\mathrm{cov}(\xi_{(i)a}\xi_{(i)c}, \xi_{(j)b}\xi_{(j)d})$ in (8.30). There are three kinds of situation:

1. The set $\{\xi_{(i)a}, \xi_{(i)c}, \xi_{(j)b}, \xi_{(j)d}\}$ only involves two unique variables. That is, $\xi_{(i)a} = \xi_{(j)b}$ and $\xi_{(i)c} = \xi_{(j)d}$ or $\xi_{(i)a} = \xi_{(j)d}$ and $\xi_{(i)c} = \xi_{(j)b}$. Without loss of generality, we assume $\xi_{(i)a} = \xi_{(j)b}$ and $\xi_{(i)c} = \xi_{(j)d}$. So,

$$\mathrm{cov}\left(\xi_{(i)a}\xi_{(i)c}, \xi_{(j)b}\xi_{(j)d}\right) = E\left(\xi^2_{(i)a}\xi^2_{(i)c}\right) = 1. \tag{8.A.7}$$

2. The set $\{\xi_{(i)a}, \xi_{(i)c}, \xi_{(j)b}, \xi_{(j)d}\}$ only involves three unique variables. Without loss of generality, we assume $\xi_{(i)a} = \xi_{(j)d}$ and $\xi_{(i)c} \neq \xi_{(j)b}$. We can convert them to three independent variables $\{\xi_1, \xi_2, \xi_3\}$ and have

$$\begin{cases} \xi_{(i)a} = \xi_{(j)d} = \xi_1 \\ \xi_{(i)c} = \xi_2 \\ \xi_{(j)b} = x_1\xi_1 + x_2\xi_2 + \sqrt{1 - x_1^2 - x_2^2}\xi_3 \end{cases},$$

where $x_1 = \mathrm{cov}(\xi_{(i)a}, \xi_{(j)b}), x_2 = \mathrm{cov}(\xi_{(i)c}, \xi_{(j)b})$. So,

$$\begin{aligned} \mathrm{cov}\left(\xi_{(i)a}\xi_{(i)c}, \xi_{(j)b}\xi_{(j)d}\right) &= E\left[\xi_1^2\xi_2\left(x_1\xi_1 + x_2\xi_2 + \sqrt{1 - x_1^2 - x_2^2}\xi_3\right)\right] \\ &= x_2 E\left(\xi_1^2\xi_2^2\right) = \mathrm{cov}\left(\xi_{(i)c}, \xi_{(j)b}\right). \end{aligned} \tag{8.A.8}$$

3. The set $\{\xi_{(i)a}, \xi_{(i)c}, \xi_{(j)b}, \xi_{(j)d}\}$ involves four unique variables. We can transform them to four independent variables $\{\xi_1, \xi_2, \xi_3, \xi_4\}$ and have

$$\begin{cases} \xi_{(i)a} = \xi_1 \\ \xi_{(i)c} = \xi_2 \\ \xi_{(j)b} = l_{3,1}\xi_1 + l_{3,2}\xi_2 + l_{3,3}\xi_3 \\ \xi_{(j)d} = l_{4,1}\xi_1 + l_{4,2}\xi_2 + l_{4,3}\xi_3 + l_{4,4}\xi_4 \end{cases},$$

where $l_{3,1} = \mathrm{cov}(\xi_{(i)a}, \xi_{(j)b})$, $l_{3,2} = \mathrm{cov}(\xi_{(i)c}, \xi_{(j)b})$, $l_{4,1} = \mathrm{cov}(\xi_{(i)a}, \xi_{(j)d})$, and $l_{4,2} = \mathrm{cov}(\xi_{(i)c}, \xi_{(j)d})$. So,

$$\begin{aligned} &\mathrm{cov}\left(\xi_{(i)a}\xi_{(i)c}, \xi_{(j)b}\xi_{(j)d}\right) \\ &\quad = E\left[\xi_1\xi_2\left(l_{3,1}\xi_1 + l_{3,2}\xi_2 + l_{3,3}\xi_3\right)\left(l_{4,1}\xi_1 + l_{4,2}\xi_2 + l_{4,3}\xi_3 + l_{4,4}\xi_4\right)\right] \\ &\quad = \left(l_{3,1}l_{4,2} + l_{3,2}l_{4,1}\right) E\left(\xi_1^2\xi_2^2\right) \\ &\quad = \mathrm{cov}\left(\xi_{(i)a}, \xi_{(j)b}\right)\mathrm{cov}\left(\xi_{(i)c}, \xi_{(j)d}\right) + \mathrm{cov}\left(\xi_{(i)a}, \xi_{(j)d}\right)\mathrm{cov}\left(\xi_{(i)c}, \xi_{(j)b}\right). \end{aligned} \tag{8.A.9}$$

Notice that the results in the first two situations, i.e., (8.A.7) and (8.A.8), are compatible to (8.A.9). So, (8.29) is proved. ■

Chapter 9
Statistical Capacitance Extraction Based on Continuous-Surface Geometric Model

Statistical capacitance extraction is required to capture the uncertainties in the nanometer manufacturing process and to provide the basis of the effective signal integrity and timing analysis for IC design. The geometric variation of interconnect wire plays a major role in the statistical capacitance extraction.

In this chapter, we present the techniques regarding the geometric variation-aware capacitance modeling. Firstly, a continuous-surface variation (CSV) model is proposed to accurately imitate the random geometric variations of on-chip interconnects. Its advantages over other existing geometric variation models are demonstrated with a typical interconnect structure under the 65-nm technology. Then, three techniques are proposed for the efficient statistical capacitance extraction based on the CSV model. Among them, the weighted principle factor analysis and the multi-core parallel computing techniques accelerate the Hermite polynomial collocation (HPC) method for statistical extraction. Then, for the chip-level extraction, the formulas for calculating the inter-window capacitance covariance are proposed, which utilize the pseudo-inverse of matrix and consider the CSV model. Finally, the comprehensive modeling and capacitance analysis techniques are proposed for the realistic line-edge roughness (LER), which is increasingly important for sub-45-nm technologies. The boundary element method (BEM)-based approach for sensitivity calculation and parallel linear-model HPC are presented. With the short-range interconnect structures under the 45-nm down to 19-nm technologies, we demonstrate that the CSV-based sensitivity modeling is accurate and most efficient for the 45-nm technology, and the linear-model HPC technique is accurate for all technology nodes and several tens to hundreds times faster than the MC simulation with 5,000 samples.

W. Yu and X. Wang, *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*, DOI 10.1007/978-3-642-54298-5_9,

9.1 The Continuous-Surface Model for Geometric Variation

In this section, we firstly review the models for the geometric variation of interconnects and classify those in existing works into three kinds. Among the three kinds, the CSV model proposed in Yu et al. [174] is most reasonable for capturing the two-direction spatially correlated geometric variations. However, it has a drawback. An improved model is then presented to revamp it, by carefully setting the random variables. Considering the 65-nm process technology and the major variation quantities, a typical interconnect structure is employed to generate test cases with different wire spacing and surface roughness. Through the comparison experiments with other existing models, the advantages of the improved CSV model are demonstrated. Several criteria are drawn for the choices in the statistical capacitance modeling in different scenarios.

9.1.1 Three Geometric Variation Models

Random variations are depicted with a set of random variables $\boldsymbol{\xi}$. The geometric variation behavior of on-chip interconnects is often assumed to follow the spatially correlated multivariate Gaussian distribution:

$$\rho\left(\xi_i\right) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\xi_i^2}{2\sigma^2}\right), \tag{9.1}$$

$$\operatorname{cov}\left(\xi_i, \xi_j\right) = \sigma^2 \exp\left(\frac{-\left|\overrightarrow{r_i} - \overrightarrow{r_j}\right|^2}{\eta^2}\right), \tag{9.2}$$

where $\rho(\xi_i)$ is the probability density function (PDF) of the ith variable ξ_i and σ is the standard derivation (Std). The spatial correlation between two variables is reflected by the correlation function in (9.2), where η is called correlation length. $\overrightarrow{r_i}$ and $\overrightarrow{r_j}$ are the spatial locations associated with ξ_i and ξ_j, respectively. There are other forms of correlation function, which may be extracted through sophisticated techniques using statistical data from actual chips [166]. The larger the correlation length, the stronger the correlation between variables spatially distributed.

For variation-aware capacitance extraction, an interconnect structure can be modeled with the following three geometric models:

- Discontinuous-surface variation (DSV) model: It is assumed that each panel on nominal conductor surface fluctuates along the normal direction of surface while keeping its shape unchanged (see Fig. 9.1). Thus, discontinuous surfaces are generated. The DSV model was employed for statistical capacitance extraction in Jiang et al. [71], Shen et al. [130], and Zhu et al. [202].

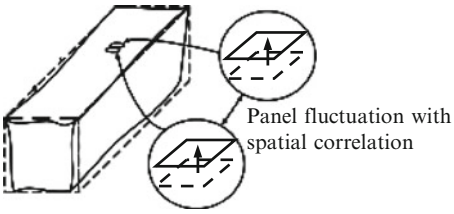

Fig. 9.1 The DSV model for statistical capacitance extraction

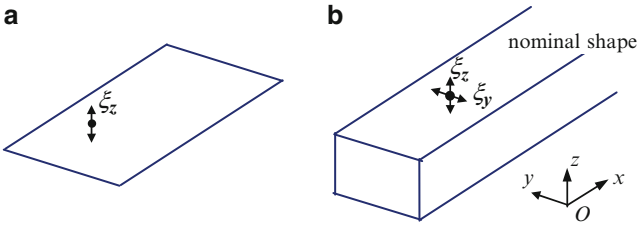

Fig. 9.2 A setting of random variables for the CSV model. (**a**) A 2-D plane, (**b**) A 3-D interconnect

- Continuous-surface variation (CSV) model: This model characterizes the fluctuations of vertices of nominal surface panels. The variational vertices are then connected with triangular panels to form continuous surfaces. The CSV model was used for 2-D plane structure in El-Moselhy and Daniel [41, 42] and, for the first time, for 3-D interconnects in Yu et al. [174].
- Variation as a whole (VAW) model: It is assumed that the surface of nominal conductor fluctuates as a whole, such that the variational geometry keeps the same shape as the nominal geometry. The width and thickness of interconnect wire are often set to be random variables. This model was employed for sensitivity calculation [20, 81] and for statistical extraction of 2-D cross-sectional structure in El-Moselhy and Daniel [42].

Among the three models, the VAW model is the simplest one, which does not consider the detailed fluctuation of surface panels. The DSV model considers the detailed geometric variations, but generates an incomplete surface, that obviously deviates from the actual situation. For a 2-D plane, it is straightforward to improve it with a CSV model. However, for actual cases with two-direction variations of height and width, building the CSV model is not trivial. In Yu et al. [174], a CSV model was proposed, where two random variables are defined for each discretization vertex on nominal surface (as shown in Fig. 9.2). This model generates continuous surface while not increasing the number of variables.

To build the variation model for statistical capacitance extraction, we also need to divide the random variables into groups according to the assumption of variation source and their correlation. In each group, variables are correlated with the relationship defined by (9.2), while different variable groups are independent

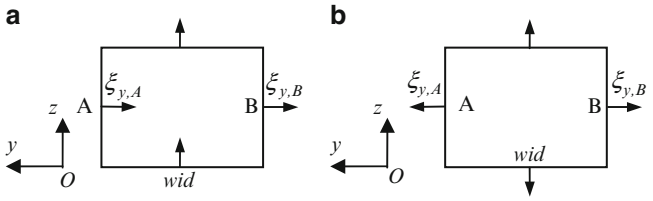

Fig. 9.3 The cross section of an interconnect wire, and the positive variation directions affecting the calculation of wire width and thickness. (**a**) The model in Yu et al. [174], (**b**) the proposed model

from each other. For the on-chip variation, the surfaces of interconnect wire are fabricated in different manufacture steps that should be considered while setting the variable groups.

Besides, the width and thickness of interconnect wire and the dielectric thickness are three major variable quantities [65, 154]. In existing DSV and CSV models, the relationship between these quantities and surface random variables is not well investigated. As we will show in the next subsection, the CSV model in Yu et al. [174] can cause very large surface fluctuation when describing a moderate width variation. Also, the variation of dielectric thickness is seldom considered in existing works.

9.1.2 The Reasonable CSV Model for On-Chip Interconnect

The CSV model in Yu et al. [174] sets two random variables for each vertex. The variables along the tangential direction of surface include some redundant information. Moreover, to make natural transition of surfaces along an arris, the variables on the opposite surfaces of the same wire are set to be correlated and have the same positive direction. This setting is illustrated by Fig. 9.3a.

For illustration, we consider the distance between two surface vertices at the same horizontal level, i.e., points A and B in Fig. 9.3a. Suppose the corresponding variables $\xi_{y,A}$ and $\xi_{y,B}$ follow the Gaussian distribution with Std of σ_y, and the correlation length for the y-direction variables is η_y. So, the wire width at this position is

$$\xi_W = \xi_{y,B} + \text{wid} - \xi_{y,A}, \tag{9.3}$$

where wid is the nominal wire width. Equation (9.3) suggests that ξ_W also follows the Gaussian distribution, whose Std is

$$\text{std}\left(\xi_W\right) = \sqrt{E\left(\xi_W^2\right) - E^2\left(\xi_W\right)} = \sqrt{E\left(\xi_{y,B}^2\right) + E\left(\xi_{y,A}^2\right) - 2\,\text{cov}\left(\xi_{y,B}, \xi_{y,A}\right)}. \tag{9.4}$$

With the property of Gaussian random variable and (9.2), we have

$$\text{std}\,(\xi_W) = \sqrt{2\sigma_y^2 - 2\sigma_y^2 \exp\left(\frac{-\text{wid}^2}{\eta_y^2}\right)} \approx \sigma_y \cdot \frac{\sqrt{2}\cdot \text{wid}}{\eta_y}. \tag{9.5}$$

The last approximation is due to the fact that $\text{wid}^2/\eta_y^2 << 1$ since the correlation length is usually much larger than wire width. Suppose $\eta_y/\text{wid} = 8$, with (9.5) we can derive that $\sigma_y \approx 5.66 \cdot \text{std}(\xi_W)$. This means, in order to make the Std of width being 10 % of its nominal value, σ_y should be as large as 56.6 % of the width. The large variance of surface fluctuation is obviously unrealistic. The analysis with this example shows that the model in Yu et al. [174] can hardly depict large variation of wire width or thickness.

To overcome the drawback of this CSV model, we firstly change the variation direction to the outer normal direction of nominal surface, as shown in Fig. 9.3b. With this modification, the width of wire in the example becomes

$$\text{std}\,(\xi_W) = \sqrt{2\sigma_y^2 - 2\sigma_y^2 \exp\left(\frac{-\text{wid}^2}{\eta_y^2}\right)} \approx \sigma_y \cdot \frac{\sqrt{2}\cdot \text{wid}}{\eta_y}, \tag{9.6}$$

whose Std is

$$\text{std}\,(\xi_W) = \sqrt{2\sigma_y^2 - 2\sigma_y^2 \exp\left(\frac{-\text{wid}^2}{\eta_y^2}\right)} \approx \sigma_y \cdot \frac{\sqrt{2}\cdot \text{wid}}{\eta_y}. \tag{9.7}$$

Equation (9.7) suggests that the Std of width approximates to 2 times of σ_y. This is reasonable and enables generation of realistic interconnects with large width variance.

Under the assumption of variation directions, the two sides of top surface tend to move toward opposite directions. Therefore, how to make the natural surface transition at the arris becomes a problem. We propose a new scheme for the variable setting, illustrated by Fig. 9.4a. In this scheme, only one random variable is set for each vertex, representing the normal-direction variation. Then, to avoid the distortion and singularity of surface near the arris, a tangential displacement is generated for each nominal vertex, which is regarded as a derived variable [denoted by $\boldsymbol{\xi}^*$ in Fig. 9.4a]. The value of derived variable is interpolated with two random variables. For example, at point E in Fig. 9.4a,

$$\xi^*_{y,E} = \xi_{y,D} + \left(\xi_{y,C} - \xi_{y,D}\right) \frac{\left|\overrightarrow{r}_E - \overrightarrow{r}_D\right|}{\left|\overrightarrow{r}_C - \overrightarrow{r}_D\right|}. \tag{9.8}$$

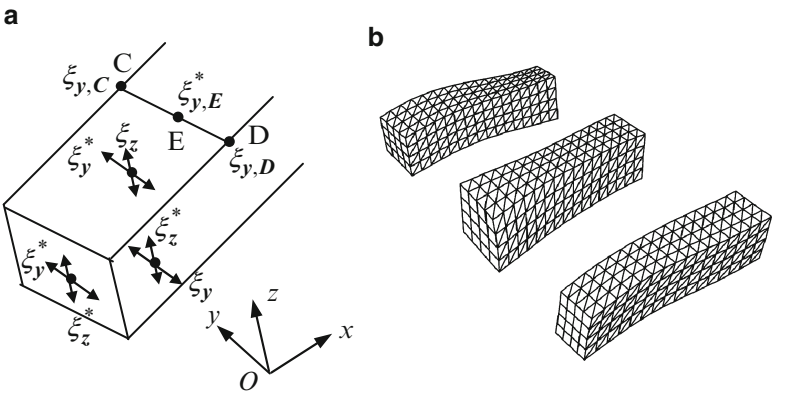

Fig. 9.4 The variable setting of the ICSV model (**a**) and the variational geometry of 3-D interconnects generated with it (**b**) (Reprinted with permission from Yu et al. [186] © 2012 Elsevier)

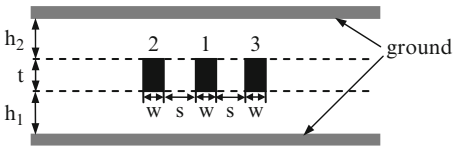

Fig. 9.5 The cross-sectional view of an interconnect structure with three parallel wires sandwiched by two ground planes

So, the value of derived variable reflects the propagation of the variations of points C and D. This scheme preserves the relative spatial relationship of two vertices after geometry variation and thus generates a normal shape with continuous surfaces.

The variational surfaces of an interconnect wire can be classified as top, bottom, left-side, and right-side surfaces. So, we assume that the random variables form four groups, each of which obeys a Gaussian distribution with spatial correlation. With this assumption, (9.7) becomes

$$\text{std}\,(\xi_W) = \sqrt{2}\sigma_y. \tag{9.9}$$

Now, we can set a reasonable σ_y or σ_z to model the variations of wire width or thickness. Figure 9.4b shows an example structure generated with the proposed model, which includes three parallel wires with surface variations. Besides, to consider the variation of dielectric thickness, we can simply include additional variables, one for an interlayer dielectric (ILD) thickness.

9.1.3 The Comparison of Three Geometric Variation Models

Figure 9.5 shows a typical on-chip interconnect structure for building the capacitance library. Investigating the statistical capacitance of this structure is very

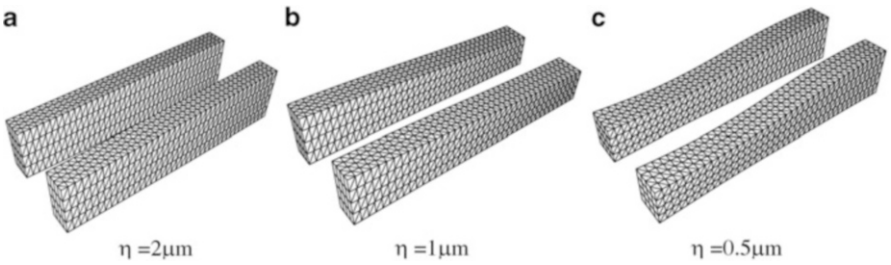

Fig. 9.6 The variational geometry of 1-μm length wires with the maximum top-surface fluctuation

important for on-chip interconnect modeling. We consider the 65-nm technology and set the nominal values of the wire width (w), wire thickness (t), and dielectric thickness (h_1, h_2) for this structure: $w = 0.1$ μm, $t = h_1 = h_2 = 0.2$ μm. These values are obtained from Cao et al. [26] and are similar with those in International Roadmap for Semiconductors prediction [68]. We assume the length of the parallel wires to be $L = 1$ μm.

Because the top and bottom planes are the approximation of the densely routed environment and the coupling capacitance is becoming prominent, we ignore the variations on the surfaces of both planes but set the dielectric thickness (h_1 and h_2) to be random variable. Both wire width and thickness variations are considered, with the same correlation length for simplicity. In this work, we only consider the random variation and set the variation Std (σ) of width and thickness to be 10 % of the nominal value (this makes the 3σ range to be 30 % of the nominal dimension).

With different value of the correlation length (η), the test structure will exhibit different magnitude of surface roughness. In Fig. 9.6, we show the variational geometry of the test structure with different values of η. These pictures are generated with the CSV model and are selected from 10,000 samples for MC simulation to reveal the maximum top-surface fluctuation. The variations of their top-surface z-coordinates are 0.035 μm, 0.065 μm, and 0.082 μm, respectively. In the following experiments, we assume the value of η varies from 0.5 to 2 μm, to reflect the actual surface roughness of interconnects at sub-65-nm-technology nodes. Changing the value of η also generates test cases with different ratios of structure dimension to correlation length (the L/η ratio). With them, we investigate the validity of the geometric models in this section and compare the efficiency of different statistical extraction methods in Sect. 9.2.2. In the experiments, the MC simulations with 10,000 samples are performed. For each deterministic sample structure, the capacitance is extracted with FastCap 2.0 [99].

For the test cases, we use the VAW, DSV, and CSV models to describe the variational geometry and then compare the MC simulation results of statistical capacitance. For the VAW model, there are only five random variables, while the Std of surface variables in DSV and CSV is set to 7.07 % to generate 10 % Std of width and thickness [see (9.9)].

Table 9.1 The comparison of the three models ($s = 0.2$ μm, $\eta = 1$ μm)

	C_{11}/Error of C_{11}		C_{12}/Error of C_{12}		$C_{1,gnd}$/Error of $C_{1,gnd}$	
	Mean	Std	Mean	Std	Mean	Std
CSV[a]	51.54	2.798	−11.19	1.184	−13.39	1.163
DSV	−0.1 %	**−25.8 %**	−0.1 %	**−20.7 %**	−0.0 %	−0.1 %
VAW	−0.6 %	**9.0 %**	−0.6 %	**8.6 %**	−0.5 %	−1.6 %

[a]The capacitance values are listed, in unit of 10^{-18} F

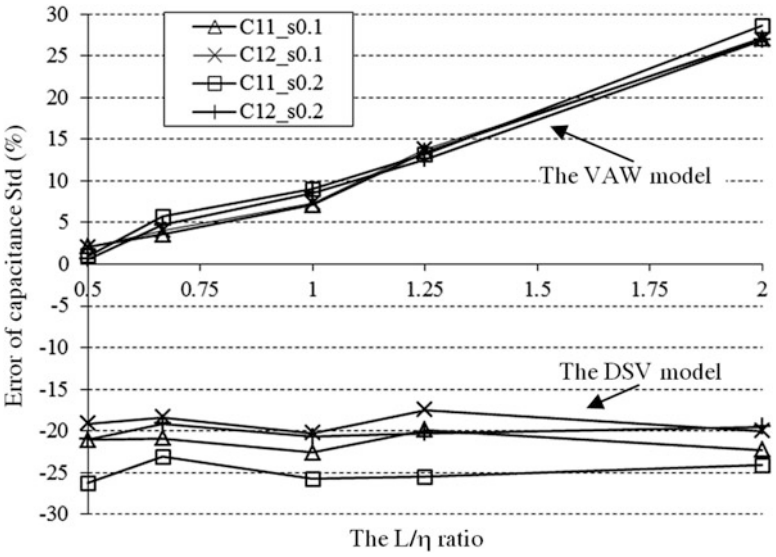

Fig. 9.7 The relative errors of the VAW model and DSV model in the capacitance Std

Firstly, we look at the case with $\eta = 1$ μm, which corresponds to the L/η ratio of 1. The simulation errors of the DSV and VAW are listed in Table 9.1, where the results of CSV are regarded as golden value. Then, other cases with different η and s (wire spacing) are simulated, and the errors of the two models are shown in Fig. 9.7.

From Table 9.1, we can see that the three models have little discrepancy in the mean value of extracted capacitances. However, the Std errors of total capacitance C_{11} and coupling capacitance C_{12} from both models are prominent. The DSV largely underestimates the capacitance variation, with more than 20 % error. The VAW overestimates the capacitance variation, since it is equivalent to an infinite correlation length. Figure 9.7 shows the trends of error varying with the ratio of L/η. Experiments with different value of L (wire length) are also performed, which show similar trend as that in Fig. 9.7, and suggest the L/η ratio is a key factor. The results reveal that the error of VAW is nearly proportional to the value of L/η. If the ratio equals to 2, the error can be as large as 30 %. If the ratio is less than 2/3, the error becomes less than 3 %.

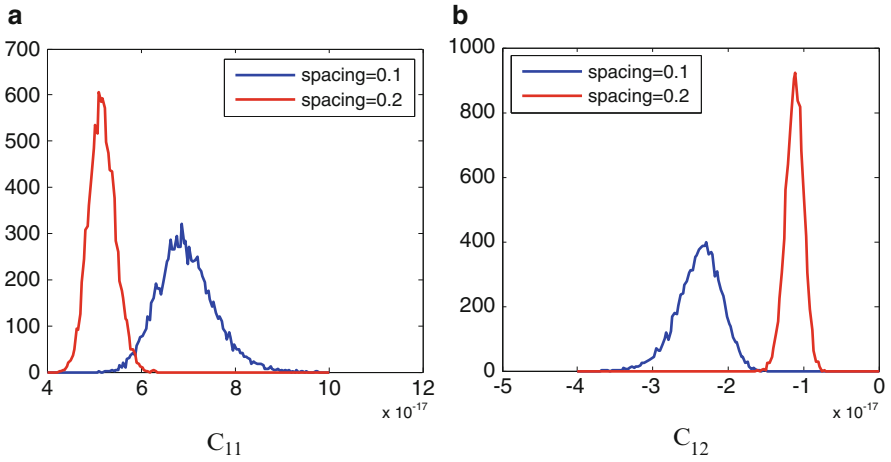

Fig. 9.8 The PDF of statistical capacitances for the case with $\eta = 1$ μm

The simulation results also show that the Std of capacitance is larger than 5 % of the mean value for C_{11} and larger than 10 % for C_{12}. Figure 9.8 gives the PDF of statistical capacitances for the case with $\eta = 1$ μm, which shows skewed probability distributions. It suggests that the quadratic model is needed for statistical capacitance, and the case with smaller wire spacing has more profound quadratic effect.

Based on the experimental results, we give the following criteria regarding statistical capacitance extraction.

CR1. The DSV model always underestimates the Std of both total and coupling capacitances, with at least 20 % error.

CR2. The VAW model overestimates the Std of capacitances, with the error proportional to the L/η ratio of extracted structure. For small-size structures with $L/\eta < 2/3$, the variational geometry can be simplified with VAW model.

CR3. If the Std of the geometric variation is 10 % of nominal value, the variational capacitances have skewed distribution requiring quadratic stochastic modeling. But for structure with large wire spacing, the skewness will become minor, and the linear model may be sufficient.

9.2 Efficient Statistical Extraction Techniques

In this section, we present three techniques for the statistical capacitance extraction based on the CSV geometric model. Firstly, a weighted principle factor analysis (wPFA) approach is presented to reduce the random variables used to construct the variational geometric model. Then, a parallel statistical solver called ***statCap*** is

presented, which combines the CSV model, wPFA, and HPC techniques. Finally, the chip-level statistical extraction in Sect. 8.4 is extended to consider the CSV model, which includes the novel formulas for calculating the inter-window capacitance covariance.

9.2.1 The Weighted PFA for Variable Reduction

Incorporating the CSV geometric model, the HPC technique in Sect. 8.3.2 can be used to generate the quadratic statistical capacitance. The computational time is proportional to d^2, where d is the number of independent variables. Random variable reduction must be performed for the proposed geometric variation model.

The principle factor analysis (PFA) is employed in Jiang et al. [71], Shen et al. [130], and Zhu et al. [202] to reduce the random variables, which is done by eigen-decomposition on the covariance matrix $\boldsymbol{\Delta n}(\boldsymbol{\xi})$ of geometric random variables $\boldsymbol{\xi}$. If the first K largest eigenvalues are $\lambda_1, \lambda_2, \ldots, \lambda_K$, and $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_K$ are the corresponding eigenvectors, the variables $\boldsymbol{\xi}$ can be approximated by the linear combination of K dominant variables $\boldsymbol{\zeta}$:

$$\boldsymbol{\xi} \approx \sum_{i=1}^{K} \sqrt{\lambda_i}\, \boldsymbol{e}_i \zeta_i. \tag{9.10}$$

Here $\{\zeta_i\}$ is a set of independent variables with $N(0,1)$ distribution that can be used to form the Hermite polynomial chaos. Thus, the statistical capacitance extraction for generating quadratic model includes the following steps:

1. Perform the eigen-decomposition on the covariance matrix of physical variations generated with (9.2) to obtain the first K largest eigenvalues and eigenvectors.
2. With the technique of sparse-grid quadrature, obtain the integral points $\boldsymbol{\zeta}^i$ and corresponding weights.
3. For each $\boldsymbol{\zeta}^i$, convert it to physical parameters $\boldsymbol{\xi}^i$ with (9.10) and then generate and extract the deterministic conductor structure with continuous surfaces. This step obtains the deterministic capacitances $C(\boldsymbol{\zeta}^i)$.
4. Calculate the coefficients of the quadratic capacitance expression with (8.21).

It should be addressed that the number of dominant variables required in PFA mainly depends on the correlation function (9.2) and is independent of the BEM discretization. Another comment is that the eigen-decomposition is not very time-consuming, since it is performed only once and the dimension of covariance matrix for intra-window structure is just moderate. If there are several groups of variables and no correlation between any two groups, the PFA shall be performed for each group separately.

The PFA method compares variable parameters one with another without considering their different impact on the output capacitance. A new variable reduction

technique was proposed in Mitev et al. [95] to take the impact on output performance into account. Inspired by Mitev et al. [95], we propose a weighted PFA (wPFA) technique for better variable reduction.

If a weight is defined for each physical variable ξ_i to reflect its impact on output, then a set of new variables $\boldsymbol{\xi}^*$ are formed:

$$\boldsymbol{\xi}^* = \boldsymbol{W}\boldsymbol{\xi}, \tag{9.11}$$

where $\boldsymbol{W}$ is a diagonal matrix of weights. The covariance matrix $\boldsymbol{\Delta n}(\boldsymbol{\xi}^*)$ of $\boldsymbol{\xi}^*$ includes the weight information, and performing PFA based on it makes weighted variable reduction. We have

$$\Delta \boldsymbol{n}\left(\boldsymbol{\xi}^*\right) = E\left(\boldsymbol{W}\boldsymbol{\xi}\left(\boldsymbol{W}\boldsymbol{\xi}\right)^{\mathrm{T}}\right) = \boldsymbol{W}\,\Delta \boldsymbol{n}\left(\boldsymbol{\xi}\right)\boldsymbol{W}^{\mathrm{T}}, \tag{9.12}$$

and denote its eigenvalues and eigenvectors by $\{\lambda_i^*\}$ and $\{\boldsymbol{e}_i^*\}$. Then, the variables $\boldsymbol{\xi}$ can be approximated by the linear combination of a set of independent dominant variables $\boldsymbol{\zeta}^*$:

$$\boldsymbol{\xi} = \boldsymbol{W}^{-1}\boldsymbol{\xi}^* \approx \boldsymbol{W}^{-1}\sum_{i=1}^{K}\sqrt{\lambda_i^*}\boldsymbol{e}_i^*\zeta_i^*. \tag{9.13}$$

For capacitance extraction, the capacitance value is the sum of panel charges, and the charge distribution is not uniform. At different positions, there is charge contribution with different significance. Based on this observation, we define the weight for variable ξ_i according to the charge around its position. With the capacitance extraction for the nominal structure performed in advance, the charges of panels are assigned to corresponding vertices. So, the diagonal entry of $\boldsymbol{W}$ is

$$w_i = \sum_{\overrightarrow{r}_i \in \text{panel}_j} \frac{q_j}{3}, \tag{9.14}$$

where q_j is the charge of panel j and $\overrightarrow{r}_i$ stands for the vertex corresponding to variable ξ_i.

Except one additional extraction of nominal structure, the weighted PFA hardly increases computational expense of PFA as $\boldsymbol{W}$ is diagonal.

9.2.2 Parallel Statistical Capacitance Extraction

We have implemented a statistical capacitance solver (called ***statCap***) based on the proposed CSV geometric model. The solver employs the HPC technique as an efficient non-MC method and the wPFA technique for variable reduction. The algorithm in ***statCap*** is described as Algorithm 9.1.

Algorithm 9.1 The HPC-Based Statistical Capacitance Solver

1. Analyze the nominal geometry to generate covariance matrices of random variables.
2. Factorize the covariance matrices with the weighted PFA technique.
3. Generate the integral points $\{\boldsymbol{\zeta}^{(i)}\}$ of sparse-grid integration, and then covert it to random geometric variables $\{\boldsymbol{\xi}^{(i)}\}$ for the sample structures.
4. Generate the values of derived variables $\{\boldsymbol{\xi}^{*(i)}\}$, and write down the description files of sample structures.
5. For each sample structure, extract capacitances with deterministic 3-D capacitance solvers.
6. Process the results of the deterministic extractions, and calculate key metrics of statistical capacitance.

Under the CSV model, the whole geometry cannot be constructed with only the sample of random variables. So, in step 4, the values of derived variables are calculated with interpolation as that in (9.8). In step 6, the statistical expression of capacitance is firstly calculated [174], and then the values of mean and Std or the PDF can be obtained.

It should be pointed out that the main flow in Algorithm 9.1 is also suitable for the statistical extraction using the MC method. In that scenario, the standard Cholesky factorization, i.e., $\boldsymbol{\Delta n} = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}$, is employed in step 2. Then, in step 3, $\boldsymbol{\zeta}^{(i)}$ is the ith sample of random variables with $N(0,1)$ distribution, and $\boldsymbol{\xi}^{(i)} = \boldsymbol{L}\boldsymbol{\zeta}^{(i)}$.

In Algorithm 9.1, the fifth step includes solving capacitances with 3-D field solver, where the number of samples $N \approx 2d^2$ for a quadratic statistical model, where d is the number of independent variables [174, 192]. Therefore, step 5 consumes most of the computing time of the statistical capacitance extraction. Because the sample structures are independent from each other, the work can be distributed to multiple processors easily. We have written ***statCap*** in MATLAB [93] and parallelized steps 4 and 5 in Algorithm 9.1, with the help of the Parallel Computing Toolbox of MATLAB.

With the test structure in Sect. 9.1.3, several experiments are carried out to compare the accuracy and efficiency of the MC method and HPC-based methods under different settings. All experiments are performed on a Linux server with 8 Xeon CPU cores with 2.13GHz. According to the **CR2** and **CR3** in Sect. 9.1.3, we consider the test cases with correlation length $\eta = 0.5$, 0.8, 1, and 1.5 μm and fix the wire spacing $s = 0.1$ μm. For the wPFA or the normal PFA, the criterion for truncation error of eigenvalues is set to 0.05.

Table 9.2 shows the capacitance errors of the HPC-based method with the wPFA technique, to the results of MC simulation. It validates the good accuracy of wPFA.

In Table 9.3, the numbers of variables after reduction by the wPFA and normal PFA are compared. The numbers of sample structures they correspond to are also listed, which is the times of invoking the deterministic capacitance solver. This table shows that wPFA can reduce the computing time for 22 % or more for the first three cases. For the last case, wPFA cannot make further reduction, since there are only ten variables.

Table 9.2 The errors of the "HPC + wPFA" method for the four test cases

η (μm)		0.5	0.8	1	1.5
Error for C_{11}	Mean	0.2 %	0.3 %	0.0 %	0.2 %
	Std	−0.6 %	1.1 %	−0.9 %	0.4 %
Error for C_{12}	Mean	0.1 %	0.4 %	0.0 %	0.4 %
	Std	−1.6 %	0.5 %	−1.6 %	0.0 %

Table 9.3 The computational results of "HPC + wPFA" and MC simulation

η (μm)		0.5	0.8	1	1.5
The L/η ratio		2	1.25	1	0.67
PFA	# variable	26	16	14	10
	# sample	1,431	561	435	231
wPFA	# variable	22	14	10	10
	# sample	1,036	436	232	232
Reduction by wPFA		**28 %**	**22 %**	**47 %**	**0 %**
Time of MC simulation (s)	serial	14,501	14,500	14,477	14,518
	parallel	1,966	1,966	1,958	1,962
Time of HPC + wPFA(s)	serial	1,473.7	610.9	333.2	331
	parallel	200.4	88.4	47.6	46.8
Sp. of "HPC + wPFA" to MC		**9.8**	**22**	**41**	**42**

The serial and parallel computational times of both methods are also listed in Table 9.3. From it we can see that both methods achieve about 7X speedup with the parallelization on an 8-core machine. And the speedup of non-MC method to MC simulation decreases with the increase of the L/η ratio. If $L/\eta = 2$, the non-MC method is 9.8 times faster than MC method, while the speedup ratio is 42 for the case with $L/\eta = 0.67$.

From the experiments, we can get the following criteria:

CR4. The weighted PFA for statistical capacitance extraction can achieve up to 47 % efficiency improvement over the normal PFA, without loss of accuracy.

CR5. The HPC-based method can be easily parallelized and achieves 7X speedup on an 8-core machine.

CR6. The HPC-based method is tens of times faster than the MC method for the test structures. For larger structures whose dimension is larger than 2X of correlation length η, its speedup to MC method will be lost, and the MC method or other better methods (such as [42]) may be the best choice.

9.2.3 Calculating the Inter-window Covariance of Capacitance

Considering the CSV geometric model, the chip-level statistical extraction scheme in Sect. 8.4 should be modified. For the intra-window extraction, the wPFA should be used to make the HPC technique feasible. Below a novel technique is proposed for calculating the inter-window capacitance covariance.

Suppose the quadratic capacitance expressions for the kth capacitance in windows i and j are

$$C_{i,k} = \sum_{p=1}^{M_i} c_{i,k,p} \Psi_p \left(\boldsymbol{\zeta}^*\right) \text{ and } C_{j,k} = \sum_{p=1}^{M_j} c_{j,k,p} \Psi_p \left(\boldsymbol{\zeta'}^*\right), \tag{9.15}$$

respectively. The covariance of capacitance can be evaluated with:

$$\operatorname{cov}\left(C_{i,k}, C_{.j,k}\right) = \sum_{p=1}^{M_i} \sum_{q=1}^{M_j} c_{i,k,p} c_{j,k,q} \operatorname{cov}\left(\Psi_p \left(\boldsymbol{\zeta}^*\right), \Psi_q \left(\boldsymbol{\zeta'}^*\right)\right). \tag{9.16}$$

Here $\boldsymbol{\zeta}^*$ and $\boldsymbol{\zeta}'^*$ are dominant variable sets in windows i and j, respectively. Thus, the problem becomes calculating the covariance between Hermite polynomials.

We first determine the covariance between a variable ζ_a^* in windows i and a variable ζ'^*_b in windows j. The relationship between $\boldsymbol{\zeta}^*$ and physical variables $\boldsymbol{\xi}$ in the weighted PFA (9.13) can be rewritten as

$$\boldsymbol{\xi} = \boldsymbol{L}_i \boldsymbol{\zeta}^*, \tag{9.17}$$

where $\boldsymbol{L}_i$ is a $n \times K$ matrix, n is the number of correlated physical variables, and K is the number of dominant factors. With the concept of pseudo-inverse, we derive from (9.17)

$$\boldsymbol{\zeta}^* \approx \boldsymbol{G}_i \boldsymbol{\xi}, \tag{9.18}$$

where $\boldsymbol{G}_i$ is the pseudo-inverse of $\boldsymbol{L}_i$, with dimensions of $K \times n$:

$$\boldsymbol{G}_i = \left(\boldsymbol{L}_i^{\mathrm{T}} \boldsymbol{L}_i\right)^{-1} \boldsymbol{L}_i^{\mathrm{T}}. \tag{9.19}$$

Now,

$$\operatorname{cov}\left(\zeta_a^*, \zeta_b'^*\right) = \operatorname{cov}\left(\sum_s g_{i,a,s} \xi_s, \sum_t g_{j,b,t} \xi_t'\right) = \sum_s \sum_t g_{i,a,s} g_{j,b,t} \operatorname{cov}\left(\xi_s, \xi_t'\right), \tag{9.20}$$

where $g_{i,a,s}$ and $g_{j,b,t}$ represent matrix entries of $\boldsymbol{G}_i$ and $\boldsymbol{G}_j$, respectively. $\operatorname{cov}(\xi_s, \xi_t')$ can be checked out from the geometric variation model.

With the covariance between two dominant variables from different windows, we can calculate the covariance between Hermite polynomials. For problem with large n, directly calculating (9.20) is time-consuming. Considering the attenuation of correlation function (9.2), we can ignore the covariance item of two variables physically far away from each other. This will reduce the computational expense.

The inter-window covariance of capacitance can be used to calculate the full-path capacitance. The mean values of relevant intra-window capacitances are assembled to obtain the mean of full-path capacitance, with the same manner as nonstatistical extraction [134]. The capacitance variance of a whole net should be calculated with the technique given in Sect. 8.4.4.

An experiment has been carried out on a test case with two parallel lines dissected into 10 windows. For this case, each window includes 1,024 triangular panels, and the wPFA reduces the variables to 8 dominant factors. The HPC solves 154 deterministic structures for each window to generate the quadratic model. This takes 322 s for all 10 windows. Additional 40.5 s are spent on calculating the statistical full-path capacitance, mainly devoted to calculating the capacitance variance with (9.20) and (8.38). The total simulation time of the MC with 10,000 samples are 20,918 s for the 10-window structure. Therefore, the approach for calculating the capacitance covariance is efficient and produces a full-path statistical extraction 58X faster than the MC simulation [174].

9.3 Fast Approaches to Model the Line-Edge Roughness

In this section, efficient techniques are presented to extract the statistical interconnect capacitance due to realistic random geometric variations, especially the line-edge roughness (LER). We firstly introduce the background of LER effect and related capacitance modeling. Then, an efficient approach based on boundary element method for calculating the capacitance sensitivity is reviewed. After that, the sensitivity calculation is extended to consider the CSV model for LER and to produce the statistical capacitance variance. An efficient capacitance modeling scheme which combines the sensitivity approach and fast linear-model HPC technique is proposed for the LER effect under the 45-nm down to 19-nm process technologies. Numerical results validate the efficiency of the proposed scheme and demonstrate how it is useful for analyzing the variational capacitances.

9.3.1 Background

The geometric variation of interconnect wire is caused by different mechanisms. The thickness variations in metals and interlevel dielectrics (ILD), for instance, are mainly caused by chemical-mechanical polishing (CMP), while the lithography steps induce the contour variations. Due to technology scaling, the random variations are becoming prominent. A typical random geometric variation is the line-edge roughness (LER). The LER variation has been extensively studied for the impact on the performance of transistors [2, 112] and demands to be considered in the variational capacitance modeling of interconnects [19, 140, 148].

Over the past several years, a lot of research has been dedicated to the variational capacitance extraction considering the random variations [42, 71, 130, 174, 202]. The emphasis of these algorithms is to derive the quadratic variational capacitance model considering the spatially correlated random variables. However, less attention was paid, in these works, to the actual scenario of on-chip process variation, and nonrealistic variation parameters were often assumed. In Sect. 9.1, we have proposed a continuous-surface variation (CSV) model to describe realistic geometric variations of interconnects. Comprehensive analysis and comparison were also carried out to reveal its rationality and necessity. Notice that most of the above works should be referred to be more theoretical rather than practical, because the demonstrated examples do not match the actual scenario of on-chip variation.

Actually, the magnitude of random variation of on-chip interconnect is not so profound. Therefore, the sensitivities of capacitance are utilized for variational capacitance modeling, which demands much less computational cost. Efficient methods of sensitivity calculation were recently proposed based on the floating random walk method [43] and the boundary element method (BEM) [19, 20]. However, in most of these works, the nominal conductor surface is assumed to vary as a whole. That is, they employ the simplified VAW geometric model. This also deviates from actual process technology. In Bi et al. [19], the LER variation was considered in calculating the capacitance sensitivity, but the underlying geometric variation model still has flaw. In Stucchi et al. [141] and Twaddle et al. [148], the variational capacitance caused by the LER was investigated for different technology nodes. However, the model only considers the width variation of interconnect wire, and the expensive MC simulations were employed.

Below, we firstly present a comprehensive model to depict the LER and other on-chip random variations. Then, in the following subsections, we consider the structures of short-range interconnects under the 45-nm down to 19-nm technologies and propose efficient approaches to model their variational capacitances caused by the LER effect.

The variational surfaces of an interconnect wire can be classified as top, bottom, left-side, and right-side surfaces. They correspond to different groups of variables, each of which can be assumed to obey a Gaussian distribution with spatial correlation. The random variables in the CSV model are the *surface variables*. Because the variations of opposite surfaces are almost independent [19, 112, 140, 148], the Std of wire width is $\sqrt{2}$ times of the Std of surface variable. So is the Std of wire thickness.

The LER is a typical random variation, arising primarily from polymer aggregation in the photoresist. The LER causes the line-width roughness (LWR). A key metric of LER is the *absolute roughness amplitude*, equal to $3\sigma_{\mathrm{LER}}$ [140, 148], where σ_{LER} is the Std of width-direction surface variables in the CSV model. The other key parameter of LER is the correlation length along the line-edge η_{LER} [19, 140, 148]. Notice that this parameter only reflects the surface fluctuation along length direction, and in the exiting works, it is assumed that the LER keeps unchanged along the thickness direction (i.e., the correlation length along

z-direction is infinite). To overcome this limitation, we define a z-direction correlation length η_z. Then, the correlation function for the sidewall variables becomes

$$\operatorname{cov}\left(\xi_i, \xi_j\right) = \sigma^2 \exp\left(-\frac{\left|r_{i,x} - r_{j,x}\right|^2}{\eta_{\mathrm{LER}}^2} - \frac{\left|r_{i,z} - r_{j,z}\right|^2}{\eta_z^2}\right), \tag{9.21}$$

where $r_{i,x}$ and $r_{i,z}$ stand for the x-coordinate and z-coordinate of the position associated with ξ_i, respectively.

In Bi et al. [19], Stucchi et al. [140], and Twaddle et al. [148], only the sidewall variation (LER) was considered with a DSV-like geometric model. For comprehensive modeling of variations, the wire thickness variation is also included in this work using the CSV model.

There is a significant and worsening gap between current LER in even the highest-resolution electron-beam lithography and the LER predicted by the ITRS [2, 140, 148]. This means the LER does not scale accordingly and becomes an increasingly larger fraction of wire dimension. Thus, the LER-induced variability is increasingly important for sub-45-nm technologies [8, 19, 68, 112, 148].

9.3.2 The Adjoint Field Technique for Sensitivity Calculation

The *adjoint field technique* (AFT) was proposed in Bi et al. [20] to calculate the capacitance sensitivity. Using the AFT, the sensitivities can be calculated as a by-product of capacitance extraction of nominal structure, with little extra expense.

For a system with N conductors, suppose $\boldsymbol{V}$ and $\boldsymbol{Q}$ denote the potentials and charges of conductors, and the matrix $\boldsymbol{C}$ represents the (short-circuit) capacitance matrix. Then,

$$\boldsymbol{C}\boldsymbol{V} = \boldsymbol{Q}. \tag{9.22}$$

Applying *Tellegen's theorem* to the electrostatic field, we have

$$\left(\widehat{\boldsymbol{V}}, (\Delta \boldsymbol{C})\, \boldsymbol{V}\right) = < (\Delta\varepsilon)\, E, \widehat{E} >, \tag{9.23}$$

where E is the electric field intensity and notation "$\widehat{\ }$" indicates the adjoint field quantities. $\Delta\boldsymbol{C}$ and $\Delta\varepsilon$ are the effective changes of capacitance matrix and medium permittivity induced by a perturbation of geometric parameter p. Notation "(,)" means the vector inner product, while "<, >" means the inner product of two functions defined by 3-D integral. From (9.23), it is derived that

$$\widehat{\boldsymbol{V}}^T (\Delta \boldsymbol{C})\, \boldsymbol{V} = \int_{\Omega} (\Delta\varepsilon)\, E \widehat{E} \mathrm{d}r. \tag{9.24}$$

In order to calculate ΔC_{ij}, the original field is set with $V_j = 1$, $V_k = 0$ $(k \neq j)$, while the adjoint field is set with $\widehat{V}_i = 1$, $\widehat{V}_k = 0$, $(k \neq i)$. Thus,

$$\Delta C_{ij} = \int_{\Omega} (\Delta\varepsilon)\, E\, \widehat{E} \mathrm{d}r. \tag{9.25}$$

Under this setting, the sensitivity of C_{ij} with respect to p is

$$\frac{\partial C_{ij}}{\partial p} = \lim_{\Delta p \to 0} \frac{1}{\Delta p} \int_{\Omega'} (\Delta\varepsilon)\, E\, \widehat{E} \mathrm{d}r, \tag{9.26}$$

while Ω' is a local region where the permittivity and electric field change due to the geometry change induced by Δp. In Bi et al. [19, 20], where the VAW or DSV geometric model was assumed, Δp is the perturbation of surface panel normal to the surface. Assume Δp causes a set of panels (denoted by S_p) to move from their nominal positions. Thus,

$$\frac{\partial C_{ij}}{\partial p} = -\sum_{k \in S_p} \varepsilon_k A_k E_k \widehat{E}_k, \tag{9.27}$$

where A_k and E_k are the area and normal electrical field intensity of panel k, respectively. And ε_k is the dielectric permittivity near panel k. If Δp is very small, the electric field can be regarded unchanged, except for the region where dielectric is replaced by conductor or vice versa. Thus, Eq. (9.27) holds.

A special case is that p is only associated with one surface panel (denoted as panel p). It produces the so-called *panel sensitivity* in Bi et al. [19]:

$$\frac{\partial C_{ij}}{\partial p} = -\frac{q_p \widehat{q}_p}{\varepsilon A_p}. \tag{9.28}$$

Here q_p and $\widehat{q}_p$ are the panel charges in the original and adjoint field settings, respectively.

9.3.3 Two Efficient Approaches

In this subsection, we firstly combine the AFT and the CSV model to calculate the statistical capacitances of the nanometer interconnects. Then, the techniques for linear-model HPC approach are presented.

In the CSV model, the varying parameter p is associated with vertex, rather than panel. In this case, the formula for capacitance sensitivity (9.26) is still valid. With the AFT, only the small region where dielectric is replaced by conductor, or vice versa, needs to be considered. In Fig. 9.9, the discretization of two conductors under

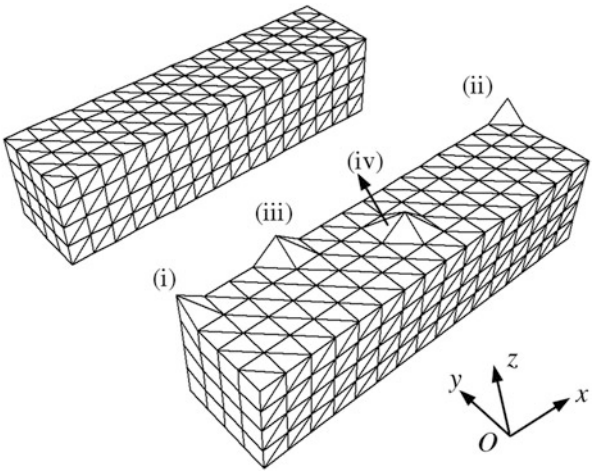

Fig. 9.9 The perturbation of geometric variables in the CSV model for the sensitivity calculation (Reprinted with permission from Yu et al. [186] © 2012 Elsevier)

the CSV model and the perturbations of geometric variables are shown. We can see that the changed region of a variable p is a kind of pyramid, whose volume is 1/3 of that of prism in Bi et al. [19, 20]. So,

$$\frac{\partial C_{ij}}{\partial p} = -\frac{1}{3}\sum_{k\in S_p} \varepsilon_k A_k E_k \widehat{E}_k, \tag{9.29}$$

where S_p denotes the indices of panels surrounding the disturbed vertex. We further derive

$$\frac{\partial C_{ij}}{\partial p} = -\frac{1}{3\varepsilon}\sum_{k\in S_p} \frac{q_k \widehat{q}_k}{A_k}. \tag{9.30}$$

Here we assume that the panels have the same surrounding dielectric with permittivity of ε, which is an usual situation.

Due to the triangular discretization employed by CSV model, the set S_p differs for different location of vertex. As shown in Fig. 9.9, there are four kinds of situation: (i) – (iv). For them, the sizes of S_p are 2, 1, 3, and 6, respectively.

Once the sensitivity for each variable is obtained, we have the linear-model approximation for a capacitance perturbation ΔC induced by these variables:

$$\Delta C = \sum_{i=1}^{G}\sum_{j=1}^{n_i} \mathcal{S}_{ij}\,\Delta p_{ij}, \tag{9.31}$$

where G is the number of variable groups and Δp_{ij} is the perturbation of the jth variable in the ith group. $\mathcal{S}_{ij}$ denotes the sensitivity, i.e., $\partial C/\partial p_{ij}$. Due to the random

nature of variables, the variance and Std of capacitance are needed. The capacitance variance can be calculated based on (9.31), i.e.,

$$\mathrm{var}(C) = \mathrm{var}\left(\sum_{i=1}^{G}\sum_{j=1}^{n_i} \mathcal{S}_{ij}\, p_{ij}\right) = \sum_{i=1}^{G}\left[\mathcal{S}_i{}^{\mathrm{T}}\, \mathrm{cov}\,(\boldsymbol{\xi}_i)\, \mathcal{S}_i\right], \tag{9.32}$$

where $\boldsymbol{\xi}_i$ is the random variables in the ith group and $\mathrm{cov}(\boldsymbol{\xi}_i)$ is the corresponding covariance matrix. $\mathcal{S}_i$ denotes the column vector of sensitivities: $[\partial C/\partial p_{i1}, \ldots, \partial C/\partial p_{i,n_i}]^{\mathrm{T}}$. And the Std of capacitance is $\mathrm{Std}(C) = \sqrt{\mathrm{var}(C)}$.

With (9.30), (9.31), and (9.32), the capacitance sensitivities and variances can be calculated for the CSV modeled variational interconnect structure. The sensitivity only depends on panel charges of the nominal structure and thus can be obtained with once extraction of the nominal structure. The sensitivity-based approach is highly efficient for generating the variational capacitance model for small-magnitude geometric variations. Notice that the sensitivity-based approach does not produce the mean value of statistical capacitance, which has little discrepancy to the nominal value if the variation is small enough.

On the other hand, the existing works on HPC or SSCM were mainly focused on obtaining the quadratic model of the variational capacitance. However, the application of linear capacitance model was almost ignored. Suppose there are d independent random variables, the linear capacitance model is

$$C\,(\boldsymbol{\zeta}) = C_0 + \sum_{i=1}^{d} c_i\, \zeta_i\,. \tag{9.33}$$

Here, C_0 is the mean value of the statistical capacitance, and c_i is the coefficient of Gaussian variable ζ_i. The coefficients C_0 and c_i are calculated with (8.21). Note that (9.31) and (9.33) are both linear expressions characterizing the relationship between the capacitance and random variables, but their underlying theory is different. The sensitivity is something about Taylor's expansion, while the linear-model HPC represents an overall fitting of the stochastic function. Therefore, with (9.33), we may produce capacitance variance with better accuracy than the sensitivity-based approach. Besides, another merit of the HPC technique is that it is able to reveal the difference between the mean value of variational capacitance and the capacitance of nominal structure.

The wPFA technique in Sect. 9.2.1 is used to reduce the large amount of surface variables to d principal independent variables. Then, with the sparse-grid technique, $m = 2d + 1$ integral points are needed to calculate the coefficients in (9.33). Each integral point corresponds to a set of values of independent variables and further a variational geometry which can be extracted by a deterministic capacitance solver. Compared with the quadratic-model HPC, which involves $2d^2 + 3d + 1$ integral points, linear-model HPC has much less computational cost but should have enough accuracy for small-magnitude variations. Finally, the parallel computing technique

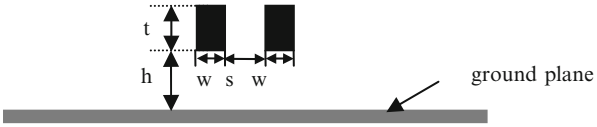

Fig. 9.10 The cross section of a two-parallel-wire structure (Reprinted with permission from Yu et al. [186] © 2012 Elsevier)

Table 9.4 The nominal values of wire width, spacing, thickness, and dielectric height in the test cases

1/2 pitch	w (nm)	s (nm)	t (nm)	h (nm)
45 nm	51	51	92	100
38 nm	43	43	77	85
19 nm	21.5	21.5	43	50

can be utilized to accelerate the linear-model HPC approach, which has achieved more than 85 % parallelization efficiency on a multi-core/multi-CPU platform in our previous experiments.

9.3.4 Numerical Results

In this subsection, we consider the typical short-range interconnect structures within standard cell, in the 45-nm and below technologies. The sensitivity-based approach and linear-model HPC approach are validated through the comparison with the results of MC simulations. All numerical results are obtained on a Linux server with 8 Xeon CPU cores with 2.13 GHz. The algorithms for statistical capacitance extraction have been implemented in the MATLAB program ***statCap***, which invokes FastCap [99] to extract the capacitances for each sample structure.

The structure of parallel wires on Metal1 layer is tested, where the LER-induced RC variability is prominent [148]. Figure 9.10 shows an example of the test structure. We consider three technology nodes, with interconnect parameters obtained from [68]. The values of wire width, spacing, thickness, and dielectric height are listed in Table 9.4. Because the geometric variation is the major concern, we just assume the single-dielectric situation that the conductors are surrounded by a homogeneous dielectric with relative permittivity of 1.

On both sidewalls of the wires, there is the LER variation. According to Asenov et al. [2], Stucchi et al. [140], and Twaddle et al. [148], we assume $3\sigma_{\mathrm{LER}} = 6$ nm, and $\eta_{\mathrm{LER}} = 20$ nm for all technology nodes. To consider the fluctuation along z-direction, η_z is set to be 1,000 nm. For the top and bottom surface variation caused by CMP, we set the Std of top and bottom surface variables to be 1 nm, and the correlation length is 1,000 nm. Notice that the actual thickness variation of on-chip interconnects is the superposition of pattern-dependent systematic variation

Table 9.5 The computational results of the CSV-based and DSV-based sensitivity approaches and MC simulation for the two-wire structures (capacitance in unit of 10^{-18} F)

Tech. node	Method	Mean (C_{11})	Std (C_{11})	Error (%)	Mean (C_{12})	Std (C_{12})	Error (%)	Time (s)
45 nm	MC-5000	8.34	0.125	N/A	−2.97	0.0911	N/A	8,184
	Sens-CSV	8.31[a]	0.117	**−6.2**	−2.95[a]	0.0827	**−9.2**	2.3
	Sens-DSV	8.29[a]	0.111	**−11.3**	−2.94[a]	0.0792	**−13.1**	1.8
	MC-5000	7.71	0.138	N/A	−2.85	0.101	N/A	12,014
38 nm	Sens-CSV	7.69[a]	0.127	**−8.2**	−2.83[a]	0.0923	**−8.5**	2.6
	Sens-DSV	7.67[a]	0.121	**−12.7**	−2.82[a]	0.0891	**−11.7**	1.9
19 nm	MC-5000	6.35	0.247	N/A	−2.90	0.215	N/A	42,707
	Sens-CSV	6.29[a]	0.207	**−17**	−2.84[a]	0.175	**−19**	7.9
	Sens-DSV	6.27[a]	0.199	**−20**	−2.83[a]	0.170	**−21**	3.0

[a]The capacitance extracted with the nominal structure

and random variation, and the former is the major factor. Since only the part of random variation is considered in this work, the above setting should be reasonable.

The two-wire structure shown in Fig. 9.10 is extracted with the CSV-based sensitivity approach, the DSV-based sensitivity approach [19], and the MC simulation with 5,000 samples. The computational results are listed in Table 9.5. Std(C_{11}) is the Std of total capacitance of wire 1, while Std(C_{12}) means the Std of coupling capacitance between wire 1 and 2. Since the sensitivity approach does not produce the statistical mean of capacitance, we list the capacitances of the nominal structure in the table. In this experiment, the length of wires is set to be $L = 100$ nm.

From Table 9.5, we can see that mean value is very close to the nominal capacitance, whose error is less than 3 %. The CSV-based sensitivity method has better accuracy than the DSV-based sensitivity method. This is because the former approximates the actual situation better. For the 45-nm and 38-nm technology, the CSV-based sensitivity method is able to calculate the Std of total capacitance and coupling capacitance with less than 10 % error. However, the error of capacitance Std increases to about 20 % for the 19-nm technology. We have varied the length of wires from 50 to 500 nm and repeated the experiment. Similar results are obtained, which show the CSV-based sensitivity method produces better accuracy. For the technology with narrow wires, the sensitivity-based approach has larger error due to the prominent LER variation. Notice that the line-width variation ($3\sigma_{\text{LER}}$*1.414) under the 19-nm technology has increased to be 39 % of the nominal width.

For the two-wire structures, the sensitivity-based method needs to solve the nominal geometry with two settings of bias voltages. While for the MC simulation, 5,000 sample geometry are solved. The results in Table 9.5 show that the speedup ratio of the sensitivity-based method to the MC simulation is about or larger than 3,000. The DSV-based sensitivity method is a little bit faster than the CSV-based method, because the former involves fewer surface panels.

As suggested in the last subsection that the linear-model HPC would have better accuracy than the sensitivity-based approach, we use it to extract the structures with stronger LER variation. The linear-model HPC employing wPFA (for briefness, we denoted it by wHPC-1) is used to extract the statistical capacitances of the two-

Table 9.6 The computational results for 19-nm-technology two-wire structures (capacitance in unit of 10^{-18} F)

Two wires	Method	Std (C_{11})	Error (%)	Std (C_{12})	Error (%)
$L = 50$ nm	MC-5000	0.180	N/A	0.147	N/A
	Sens-CSV	0.151	**−16**	0.122	**−17**
	wHPC-1	0.176	**−2.3**	0.140	**−4.4**
$L = 100$ nm	MC-5000	0.247	N/A	0.215	N/A
	Sens-CSV	0.207	**−17**	0.175	**−19**
	wHPC-1	0.234	**−5.5**	0.200	**−7.3**
$L = 200$ nm	MC-5000	0.342	N/A	0.305	N/A
	Sens-CSV	0.300	**−12**	0.257	**−16**
	wHPC-1	0.331	**−3.4**	0.293	**−3.9**

Table 9.7 The computational results for 19-nm-technology three-wire structures (capacitance in unit of 10^{-18} F)

Three wires	Method	Std (C_{11})	Error (%)	Std (C_{12})	Error (%)
$L = 50$ nm	MC-5000	0.240	N/A	0.147	N/A
	wHPC-1	0.234	**−2.3**	0.139	**−5.4**
$L = 100$ nm	MC-5000	0.329	N/A	0.211	N/A
	wHPC-1	0.325	**−1.3**	0.200	**−5.3**
$L = 200$ nm	MC-5000	0.495	N/A	0.302	N/A
	wHPC-1	0.473	**−4.3**	0.291	**−3.6**

wire structure and a similar structure with three parallel wires under the 19-nm technology. The results for cases with different wire lengths are listed in Tables 9.6 and 9.7. Since the error of mean value is less than 3 % (also demonstrated by other existing works, e.g., [192]), here only the results about the Std of capacitance are listed.

From Table 9.6, we see that the wHPC-1 is able to reduce at least half error of the CSV-based sensitivity method. The experiments on the three-wire structure produce the similar results. As a whole, the error of wHPC-1 is always less than 8 %.

The computational time of wHPC-1 and MC simulation is shown in Table 9.8. We also list the numbers of independent variables (# variable) and deterministic sample structures for solving (# sample) in the method of wHPC-1. The last row in Table 9.8 is the time of parallel computing on the 8-core machine, while other data of time are obtained with serial computing. For the purpose of comparison, the # sample for the quadratic-model HPC (wHPC-2) is also given. The quadratic-model HPC has good accuracy, but its computational time is proportional to # sample. So in the experiments, wHPC-2 should be several tens of times slower than wHPC-1 and even slower than the MC simulation with 5,000 samples. On the contrary, the wHPC-1 is several tens to several hundred times faster than the MC method. While comparing with the sensitivity-based method in Table 9.5, the paralleled wHPC-1 (time is 70.9 s for $L = 100$ nm) is about 10X slower. We think this is the affordable expense for improving accuracy.

Table 9.8 The comparison of wHPC-1 and MC method on the computing time

		Two wires			Three wires		
L (nm)		50	100	200	50	100	200
wHPC-1	# variable (d)	14	26	42	20	38	62
	# sample (m)	30	54	86	42	78	126
	Time (s)	118	465.7	212.5	217.1	746.4	421.4
wHPC-2	# sample (m)	436	1,432	3,656	862	3,004	7,876
MC	Time (s)	20,014	42,707	12,723	26,062	47,956	16,906
Sp. of wHPC-1 to MC		**170**	**92**	**60**	**120**	**64**	**40**
Time of parallel wHPC-1(s)		17.2	70.9	32.7	32.8	112	62.0

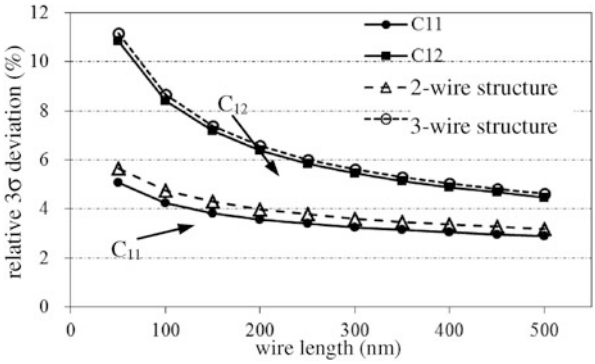

Fig. 9.11 Plot of capacitance variation as a function of wire length for the 45-nm technology (Reprinted with permission from Yu et al. [186] © 2012 Elsevier)

9.3.5 More Analysis Results and Discussion

Using both approaches, we can easily analyze the variational capacitance caused by the on-chip random variations. The 45-nm-technology cases in last subsection are the representative of cases with week variation effect, and we tackle them with the CSV-based sensitivity approach. The 19-nm-technology cases are the representative of cases with strong variation effect, which should be extracted with the linear-model HPC technique. The relative standard deviation Std(C)/C is always referred to as "mismatch" by designers, to model the effect of LER and other variation on capacitances. With the 45-nm and 19-nm cases, we can study the relationship between Std(C)/C and the wire length. The simulation results are shown in Figs. 9.11 and 9.12; the relative deviation 3Std(C)/C is plotted. On the 8-core machine, the simulation for Fig. 9.11 only costs 2 min, while the simulation time for Fig. 9.12 is about 36 min.

From the plots we can see that, as wire length is scaled from 500 to 50 nm, the related 3σ deviation increases from about 4 to 11 % for C_{12}, under the 45-nm technology. And there is larger increase of related 3σ deviation for the 19-nm technology (from 12 to 28 %). This is because the variation of capacitance is averaged out as the wire length increases. The mismatch of coupling capacitance

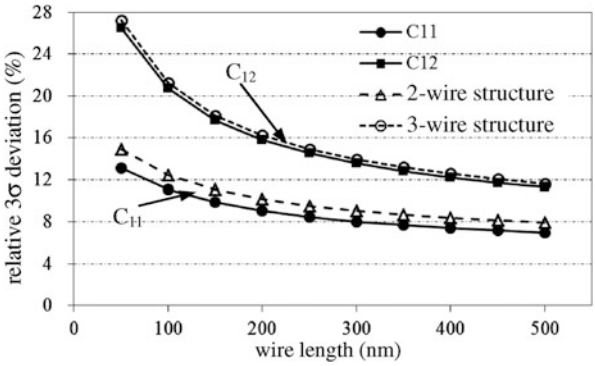

Fig. 9.12 Plot of capacitance variation as a function of wire length for the 19-nm technology (Reprinted with permission from Yu et al. [186] © 2012 Elsevier)

C_{12} is about double of that of the total capacitance C_{11}. While comparing Fig. 9.11 and Fig. 9.12, we see that the capacitance variation is doubled if the feature size scales from 45 to 19 nm. The capacitance variation is almost inverse proportional to the feature size. Further analysis could be performed, such as to study the impact of parameters σ and η on Std(C)/C. These analyses are useful for circuit designers to estimate mismatches and optimize the critical structures accordingly.

For large structure with longer wire length, the advantage of wPFA-accelerated HPC technique will be lost. In that case, the window-based extraction technique [174, 192] can be used to reduce the size of structure that the HPC technique handles.

Both the CSV-based sensitivity approach and the linear-model HPC technique are based on utilizing the BEM to solve deterministic interconnect structures. The former uses the surface panel charges extracted from the nominal geometry to generate the capacitance sensitivities [see (9.30)], while the latter uses the BEM solver to calculate the capacitances [i.e., $C(\boldsymbol{\zeta}^i)$ in (8.21)] of the sample structures with irregular geometry. So, there is no problem to append the numerical results in this work with the cases with multiple dielectric materials as in reality, since BEM is applicable to multi-dielectric structures. The only difference is that, for multi-dielectric cases, each BEM extraction needs more computational time due to the additional discretization of dielectric interfaces. However, this would not degrade the advantages of the presented techniques for statistical extraction.

9.4 Summary

To model the random geometric variations of on-chip interconnects, a reasonable continuous-surface variation (CSV) model is proposed. A series of experiments on a typical 65-nm-technology structure are carried out to compare the CSV model and other exiting models describing the geometric variations. The results demonstrate that the discontinuous-surface variation (DSV) model always underestimates the Std of both total and coupling capacitances, with 20 % or larger error. And the

variation as a whole (VAW) model overestimates the Std of capacitances, with the error proportional to the L/η ratio of the extracted structure, where L is the length of wire and η is the correlation length of variation.

A weighted principle factor analysis (PFA) technique for the capacitance extraction based on the Hermite polynomial collocation (HPC) is proposed to reduce the random variables, which achieves up to 47 % efficiency improvement over the normal PFA. Combined with the parallel computing technique, the statistical capacitance extraction becomes very efficient and exhibits 7X parallel speedup on an 8-core machine. Besides, the formulas for calculating the inter-window capacitance covariance are proposed to consider the CSV model. The pseudo-inverse of matrix is utilized to accelerate the computation.

Finally, the geometric model and capacitance extraction techniques are investigated to consider the realistic LER of interconnects under the 45-nm and below technologies. The fast BEM-based sensitivity method is extended to consider the accurate CSV model, and the linear-model HPC technique is presented for the purpose of fast statistical capacitance extraction. The numerical experiments on short-range interconnects with LER effect demonstrate that the CSV-based sensitivity approach and the linear-model HPC technique have sufficient accuracy exhibiting several orders of magnitude speedup to the MC simulation. Moreover, the linear-model HPC technique has better accuracy and is applicable for structures with strong LER under the 19-nm technology. The presented approaches have also been applied to the analysis of manufacturing variabilities of the design optimization.

Chapter 10
Fast Floating Random Walk Method for Capacitance Extraction

The field-solver algorithm for capacitance extraction can be classified into two categories: (1) the conventional deterministic algorithms based on boundary element method (BEM) [28, 99, 133, 170, 179, 185], finite element method (FEM) [29, 149], etc., and (2) the floating random walk (FRW) algorithm with stochastic nature [11, 12, 22, 38, 43, 44, 64, 69, 75, 82, 83, 203]. The deterministic algorithms are fast and accurate, but not suitable for large-scale structures due to the large demand of computational time and the bottleneck of memory usage.

In this chapter, we present the algorithms employed in an FRW-based 3-D capacitance solver, called RWCap [125]. The solver is able to efficiently simulate the structures with Manhattan geometry and multilayered dielectrics, which is the major scenario for the capacitance extraction of VLSI interconnects. The main contributions in this solver are as follows:

1. An efficient approach is proposed for the capacitance extraction with multilayered dielectrics, which utilizes the numerically characterized surface Green's function and weight value for the cubic transition domain with two dielectric layers. With the precharacterization procedure for a given process technology, the proposed approach accelerates the FRW-based extraction for up to 160X with very small memory overhead.
2. A comprehensive variance reduction approach is proposed to accelerate the convergence rate of the FRW algorithm. The approach includes an importance sampling-based technique to minimize the variance of weight value distribution and a scheme combining it and the stratified sampling technique. This approach brings 3X or more speedup to the proposed FRW algorithm for multi-dielectric capacitance extraction, without memory overhead and the loss of accuracy.
3. A parallel FRW algorithm on the multi-core/multi-CPU platform is proposed. A space management technique is presented to make the FRW-based solver efficient while handling large structures. Numerical results show the parallel RWCap achieves more than 6X speedup on a machine with 8 CPU cores.

W. Yu and X. Wang, *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*, DOI 10.1007/978-3-642-54298-5_10,

It should be pointed out that there is little literature which reveals the algorithm details of the 3-D FRW for multi-dielectric capacitance extraction. Because the transition probability for a cubic domain with inhomogeneous dielectric cannot be derived analytically, extending the single-dielectric algorithm [69, 83] to the multi-dielectric structure is not straightforward. The presented work started the research on the fast FRW capacitance solver for VLSI interconnects, from the academic perspective.

10.1 The Basic Floating Random Walk Algorithms

The fundamental formula of the FRW algorithm is [83]

$$\phi(\boldsymbol{r}) = \oint_S P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right)\phi\left(\boldsymbol{r}^{(1)}\right) \mathrm{d}\boldsymbol{r}^{(1)}, \tag{10.1}$$

where $\phi(\boldsymbol{r})$ is the electric potential at point $\boldsymbol{r}$ and S is a closed surface surrounding $\boldsymbol{r}$. $P(\boldsymbol{r}, \boldsymbol{r}^{(1)})$ is called the surface Green's function. For a fixed $\boldsymbol{r}$, $P(\boldsymbol{r}, \boldsymbol{r}^{(1)})$ can be regarded as the probability density function (PDF) for selecting a random point $\boldsymbol{r}^{(1)}$ on S. In this sense, $\phi(\boldsymbol{r})$ can be estimated by the mean value of $\phi(\boldsymbol{r}^{(1)})$, providing a sufficient large number of sample points $\boldsymbol{r}^{(1)}$ on S are evaluated. If S is the surface of a homogeneous cube centered at $\boldsymbol{r}$, $P(\boldsymbol{r}, \boldsymbol{r}^{(1)})$ only depends on the relative position of $\boldsymbol{r}^{(1)}$ and is not related to the size of cube [69, 83]. More importantly, this Green's function can be derived analytically [69] and pre-calculated and stored as the discrete probabilities for jumping to the discretized cells of the cube surface.

In the situation that $\phi(\boldsymbol{r}^{(1)})$ is unknown, we apply (10.1) recursively to obtain the following nested integral formula:

$$\begin{aligned}\phi(\boldsymbol{r}) = &\oint_{S^{(1)}} P^{(1)}\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right)\oint_{S^{(2)}} P^{(2)}\left(\boldsymbol{r}^{(1)}, \boldsymbol{r}^{(2)}\right)\cdots \\ &\oint_{S^{(k+1)}} P^{(k+1)}\left(\boldsymbol{r}^{(k)}, \boldsymbol{r}^{(k+1)}\right)\phi\left(\boldsymbol{r}^{(k+1)}\right) \mathrm{d}\boldsymbol{r}^{(k+1)}\cdots \mathrm{d}\boldsymbol{r}^{(2)}\mathrm{d}\boldsymbol{r}^{(1)},\end{aligned} \tag{10.2}$$

where $S^{(i)}(i = 1, \ldots, k+1)$ is the surface of the ith cube centered at $\boldsymbol{r}^{(i-1)}$. $P^{(i)}(\boldsymbol{r}^{(i-1)}, \boldsymbol{r}^{(i)})$ and $(i = 1, \ldots, k+1)$ are the surface Green's functions relating the potentials at $\boldsymbol{r}^{(i-1)}$ to $\boldsymbol{r}^{(i)}$. This can be interpreted as a floating random walk procedure: for the ith hop of a walk, the maximum conductor-free cube centered at $\boldsymbol{r}^{(i-1)}$ is constructed, and then a point $\boldsymbol{r}^{(i)}$ is randomly selected on the cube surface according to the discrete probabilities obtained with $P^{(i)}(\boldsymbol{r}^{(i-1)}, \boldsymbol{r}^{(i)})$. Note that to obtain the probabilities, we only need to consider unit-size cube for the ith cube and the corresponding positions of $\boldsymbol{r}^{(i-1)}$ and $\boldsymbol{r}^{(i)}$ in the unit-size cube. The walk terminates after k hops if the potential at point $\boldsymbol{r}^{(k)}$ is known, e.g., it is on a conductor surface

in the problem of capacitance extraction. With the surface Green's function and derived sampling probabilities for a unit-size cube calculated in advance [69], the major cost of random walk is for geometric operations. After performing many walks, the mean value of these estimates approximates $\phi(\boldsymbol{r})$ very well.

For extracting capacitances among conductors, the relationship between conductor charge and potential is needed. So, a *Gaussian surface* G_j is constructed to enclose conductor j (the master conductor), and according to the Gauss theorem,

$$Q_j = \oint_{G_j} \overrightarrow{D}(\boldsymbol{r}) \cdot \widehat{n}(\boldsymbol{r}) \, \mathrm{d}\boldsymbol{r} = \oint_{G_j} F(\boldsymbol{r}) \left(-\nabla\phi(\boldsymbol{r})\right) \cdot \widehat{n}(\boldsymbol{r}) \, \mathrm{d}\boldsymbol{r}, \tag{10.3}$$

where Q_j is the charge on conductor j, $F(\boldsymbol{r})$ is the dielectric permittivity at point $\boldsymbol{r}$, and $\widehat{n}(\boldsymbol{r})$ is the normal direction of G_j at $\boldsymbol{r}$. With (10.1) substituted to (10.3), the following formula can be derived:

$$Q_j = \oint_{G_j} F(\boldsymbol{r}) \, g \oint_{S^{(1)}} \omega\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) P^{(1)}\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \phi\left(\boldsymbol{r}^{(1)}\right) \mathrm{d}\boldsymbol{r}^{(1)} \mathrm{d}\boldsymbol{r}, \tag{10.4}$$

where the weight value

$$\omega\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) = -\frac{\nabla_r P^{(1)}\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})}{g P^{(1)}\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right)}. \tag{10.5}$$

∇_r is the gradient operator with respect to $\boldsymbol{r}$, and the constant g satisfies $\oint_{G_j} F(\boldsymbol{r}) \, g \mathrm{d}\boldsymbol{r} = 1$. Now the first integral in (10.4) can be interpreted as a stochastic sampling procedure on G_j, and the second integral can be calculated with the above FRW procedure based on (10.2). The only difference is the weight value (10.5), which contributes to the estimated value of Q_j and relates Q_j to the conductor potentials (voltages) through the FRW procedure. Thus, averaging the weight values from many walks produces the self- and coupling capacitances of conductor j. Note that the calculation of weight value can also be accelerated with a precharacterization process as that for the transition probability. The above deduction derives the FRW algorithm for capacitance extraction (see Algorithm 10.1).

The total computing time of FRW algorithm is roughly

$$T_{\text{total}} = N_{\text{walk}} \cdot N_{\text{hop}} \cdot T_{\text{hop}}, \tag{10.6}$$

where N_{walk} is the number of random walks/paths, N_{hop} is the average number of hops in a walk, and T_{hop} is the average computing time for a hop. The techniques proposed in this chapter reduce N_{walk}, N_{hop}, and T_{hop}, so that they produce an efficient FRW algorithm for capacitance extraction.

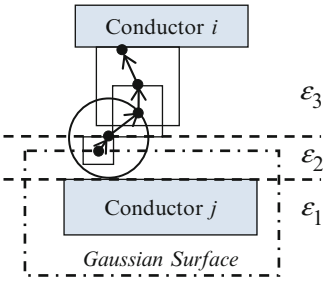

Fig. 10.1 A "shortest" random walk in the multi-dielectric structure with the usage of spherical transition domain

The analytical surface Green's function for the spherical transition domain with different hemispheres can help the FRW to handle the situation with multiple dielectrics [64, 123]. Since the single-dielectric FRW relies on that the transition cube is within a single dielectric, the sphere transition domain is used to continue the walk stopping at dielectric interface. The resulting FRW algorithm for multi-dielectric extraction is described as Algorithm 10.1.

Algorithm 10.1 The Basic FRW for Multi-dielectric Problem

1: Load the precomputed probabilities and weight values for single-dielectric cubic transition domain;
2: Construct the Gaussian surface enclosing master conductor j;
3: $C_{ji} := 0, \forall\, i;\ npath := 0;$
4: **Repeat**
5: $npath := npath + 1;$
6: Pick a point $\boldsymbol{r}^{(0)}$ on Gaussian surface, and then generate a single-dielectric cubic transition domain T centered at it; pick a point $\boldsymbol{r}^{(1)}$ on the surface of T according to the precomputed probabilities, and then calculate the weight value ω with the help of the precomputed weight values;
7: **While** the current point is not on a conductor **do**
8: **If** the current point is on dielectric interface, construct a sphere transition domain, **otherwise** construct a single-dielectric cubic domain;
9: Pick a point on the domain surface, according to the probabilities of cubic domain or sphere domain;
10: **End**
11: $C_{ji} := C_{ji} + \omega;$ //the current point is on conductor i
12: **Until** the convergence criterion is met
13: $C_{ji} := C_{ji}/npath, \forall\, i;$

However, while extracting the actual interconnect structure with multiple dielectrics, Algorithm 10.1 would induce many hops in a walk and thus is not efficient. Figure 10.1 illustrates an example of walk in the basic FRW algorithm.

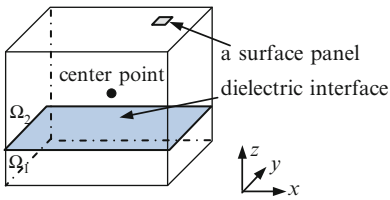

Fig. 10.2 A transition cube with multiple dielectrics

10.1.1 Numerical Technique to Calculate Multi-dielectric Surface Green's Function

With the finite difference method (FDM), the transition probabilities can be calculated for a transition cube with multiple dielectrics [43]. Suppose there is a cubic domain with multi-dielectric layers, as shown in Fig. 10.2. After discretization, the cube surface is dissected into small panels. The transition probability what we want is the relationship of the potentials at the center point and a surface panel.

In each homogeneous subdomain, the Laplace equation holds

$$\nabla^2\phi = \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} + \frac{\partial^2\phi}{\partial z^2} = 0. \tag{10.7}$$

At the dielectric interface, there is the continuity condition:

$$\varepsilon^+ \frac{\partial\phi}{\partial z^+} = \varepsilon^- \frac{\partial\phi}{\partial z^-}, \tag{10.8}$$

where ε^+ and ε^- are the permittivities of up and down dielectrics, respectively. Using the FDM, the following matrix equation is derived from (10.7) and (10.8):

$$\begin{bmatrix} \boldsymbol{E}_{11} & \boldsymbol{E}_{12} & \boldsymbol{E}_{13} \\ \boldsymbol{O} & \boldsymbol{I}_2 & \boldsymbol{O} \\ \boldsymbol{E}_{31} & \boldsymbol{O} & \boldsymbol{D}_{33} \end{bmatrix} \begin{bmatrix} \boldsymbol{\phi}_{\mathrm{I}} \\ \boldsymbol{\phi}_{\mathrm{B}} \\ \boldsymbol{\phi}_{\mathrm{F}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{f}_{\mathrm{B}} \\ \boldsymbol{0} \end{bmatrix}, \tag{10.9}$$

where $\boldsymbol{\phi}_{\mathrm{I}}$, $\boldsymbol{\phi}_{\mathrm{B}}$, and $\boldsymbol{\phi}_{\mathrm{F}}$ are the potential unknowns on inner grid points, the surface panels, and dielectric interfaces, respectively. $\boldsymbol{f}_{\mathrm{B}}$ is the boundary potentials, which may be regarded as the Dirichlet boundary condition. The $\boldsymbol{E}$ matrices are sparse finite difference matrices discretizing the Laplace operator. The third block row of (10.9) is derived from (10.8), and therefore, $\boldsymbol{D}_{33}$ is a diagonal matrix. Also note that $\boldsymbol{I}_2$ is an identity matrix. We then have

$$\boldsymbol{\phi}_{\mathrm{I}} = -\left(\boldsymbol{E}_{11} - \boldsymbol{E}_{13}\boldsymbol{D}_{33}^{-1}\boldsymbol{E}_{31}\right)^{-1}\boldsymbol{E}_{12}\boldsymbol{f}_{\mathrm{B}}. \tag{10.10}$$

It expresses the relationship between the inner points and the boundary points. Suppose the center of the cube is the kth inner grid points. Then, the potential of center point is

$$\phi_k = \boldsymbol{e}_k^{\mathrm{T}} \boldsymbol{\phi}_I = -\left(\left(\boldsymbol{E}_{11} - \boldsymbol{E}_{13} \boldsymbol{D}_{33}^{-1} \boldsymbol{E}_{31} \right)^{-\mathrm{T}} \boldsymbol{e}_k \right)^{\mathrm{T}} \boldsymbol{E}_{12} \boldsymbol{f}_{\mathrm{B}}, \tag{10.11}$$

where $\boldsymbol{e}_k$ denotes a column vector where the kth element is 1 and otherwise 0. From (10.11), we can see that the row vector

$$\boldsymbol{P}_k = -\left(\left(\boldsymbol{E}_{11} - \boldsymbol{E}_{13} \boldsymbol{D}_{33}^{-1} \boldsymbol{E}_{31} \right)^{-\mathrm{T}} \boldsymbol{e}_k \right)^{\mathrm{T}} \boldsymbol{E}_{12} \tag{10.12}$$

represents the discrete probabilities for transition from the center point to the boundary panels and is what we want.

The numerical technique can be generalized to consider different boundary conditions, such as the Neumann boundary, floating potential boundary, etc. [43]. However, it is impractical to precompute and tabulate the transition probabilities for the maximum conductor-free transition cubes with all possible dielectric configurations. The question is how many transition domains should be precharacterized to well balance the memory usage and computational efficiency of the multi-dielectric capacitance extraction.

10.2 A Multi-dielectric FRW Algorithm with the Precharacterized Probabilities and Weight Values

In this section, we first introduce the proposed approach to precharacterize the transition cubes with two dielectric layers. Then, the details of the FDM techniques for the precharacterization are presented. The multi-dielectric FRW algorithm is finally given.

10.2.1 The Basic Idea

Modern VLSI process technology involves multiple layers of metal wires and dielectrics. The coupling capacitance mainly exists between the wires at the same layer or at two adjacent layers, due to the densely routed wires. If the hop in FRW walk is able to cross one dielectric interface, the number of hops in a walk may not increase much as compared with that for single-dielectric circumstance. Therefore, numerically characterizing the surface Green's function for the cubic transition domain with two dielectric layers may bring sufficient benefit to the extraction of multi-dielectric VLSI structures. Figure 10.3 shows an example of walk under the

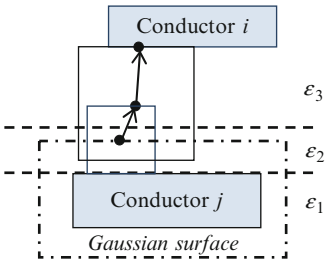

Fig. 10.3 A "shortest" random walk in the multi-dielectric structure with the usage of multi-dielectric transition probability and weight value

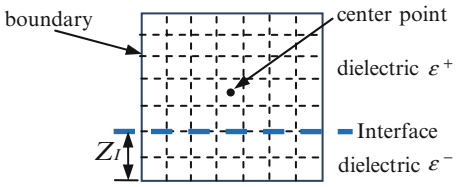

Fig. 10.4 The cubic transition domain with FDM grids for precharacterizing the multi-dielectric process technology (2-D cross-sectional view)

assumption that the hop is able to cross one dielectric interface. Comparing Fig. 10.3 and Fig. 10.1, we can see that the number of hops is largely reduced. On the other hand, for a specified technology with n_d dielectric layers (adjacent layers with same permittivity are regarded as one layer), there are no more than n_d adjacent dielectric pairs with distinct pair of permittivities. For each pair of permittivities, the surface Green's function is precharacterized as Green's function tables (GFTs), and the weight value is precharacterized as weight value tables (WVTs). The computing time for the precharacterization procedure and the memory usage for recalling the GFTs and WVTs should be moderate for a given process technology.

In the precharacterization procedure, we only consider the unit-size cubic transition domain, which is discretized with FDM grids. The cube includes two dielectric layers, and the position of dielectric interface is set to conform to the FDM grid (see Fig. 10.4 for a cross-sectional view). For each dielectric configuration (permittivity pair and interface position), the FDM technique based on Sect. 10.2.3 is employed to derive the corresponding GFT and WVT.

10.2.2 The Details of the Precharacterization Procedure

For precharacterizing the surface Green's function accurately, some details of FDM should be considered. Firstly, to avoid the potential unknown setting on the edge or corner of the cube, each surface unknown is set at the center of boundary panel. Also note that there is an unknown at the center of cube. To connect these unknowns with an FDM grid, we firstly divide the cube into $N \times N \times N$ equal-sized cells and then attach one unknown to each cell's center. Here N should be an odd number (see Fig. 10.4). These N^3 unknowns form the $\boldsymbol{\phi}_I$ in (10.9) for inner grid points. The boundary unknowns attached to panel's center point form $\boldsymbol{\phi}_B$. Because the height

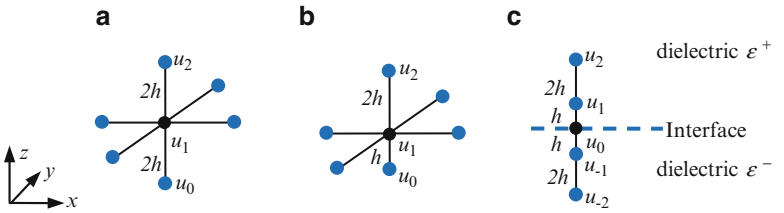

Fig. 10.5 Illustration of finite difference schemes for (**a**) inner grid point, (**b**) grid point near boundary, and (**c**) point on interface

of interface Z_I has the value of $1/N$, $2/N$, $\ldots$, $(N-1)/N$, extra unknowns $\boldsymbol{\phi}_F$ are defined on the interface surface to connect inner points upward and downward.

Different finite difference (FD) schemes are used to generate the coefficient matrix blocks in (10.9). For the uniform inner grid point (see Fig. 10.5a), the standard seven-point differential scheme with second-order accuracy is used. Figure 10.5b shows the grid points near the boundary, where the grid step varies (we assume the inner grid size is $2h$). For this nonuniform grid, the Lagrange interpolation is used to derive the differential scheme with second-order accuracy. For example, the z-direction derivative for the point in Fig. 10.5b is approximated with

$$\frac{\partial^2 \phi}{\partial z^2} \approx \frac{2u_0}{3h^2} - \frac{u_1}{h^2} + \frac{u_2}{3h^2}. \tag{10.13}$$

Here u denotes the potential on grid point. For the point on interface, in order to be consistent with the standard seven-point differential scheme for inner point, the formula with second-order accuracy is derived to approximate (10.8). As shown in Fig. 10.5c, on each side of interface, the potentials of two points are used to approximate $\partial\phi/\partial z^+$ or $\partial\phi/\partial z^-$. For this example, the derived difference equation for (10.8) is

$$\varepsilon^+ \frac{-8u_0 + 9u_1 - u_2}{6h} = \varepsilon^- \frac{-8u_0 - 9u_{-1} + u_{-2}}{6h}. \tag{10.14}$$

It should be pointed out that if using the first-order difference equation for (10.8), we find out that the obtained surface Green's function would include some obvious errors.

In the FDM solution, the height of interface Z_I cannot be 1/2, which omits an important situation when the walk stops at a dielectric interface. A special treatment should be taken. As shown in Fig. 10.6a, which is a side view, the cube edge should be divided into even segments, so that there is no boundary panel across the interface. Therefore, the center point is not an inner grid point. As shown in Fig. 10.6b, the potential at the center point can be approximated by its eight neighbors, i.e.,

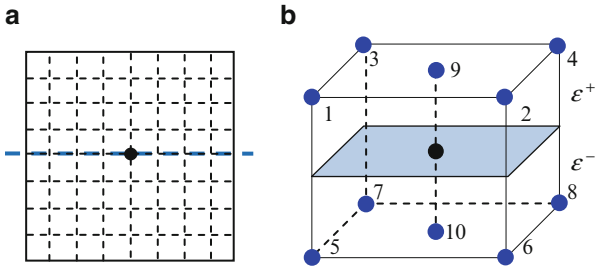

Fig. 10.6 The situation where the height of interface $Z_I = 1/2$. (**a**) The side view of the unit-size cube. (**b**) FDM grids at the center of cube

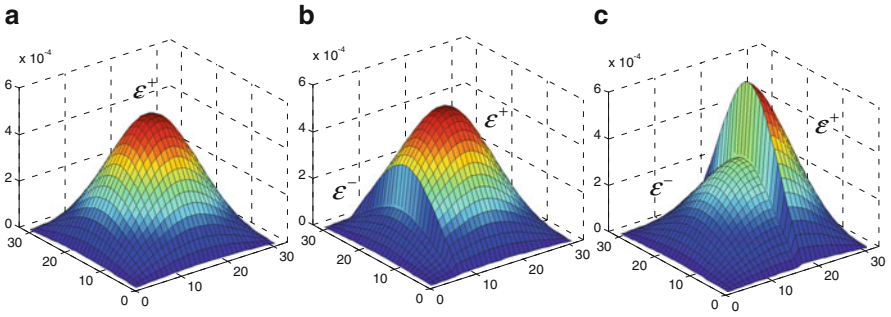

Fig. 10.7 The GFT distributions on cube boundaries when $\varepsilon^- = 2.6$ and $\varepsilon^+ = 5$. (**a**) On the top face. (**b**) On the side wall, for $Z_I = 1/5$. (**c**) On the side wall, for $Z_I = 1/2$

$$u_c = \frac{\varepsilon^+ u_9 + \varepsilon^- u_{10}}{\varepsilon^+ + \varepsilon^-} = \frac{\varepsilon^+ (u_1 + u_2 + u_3 + u_4) + \varepsilon^- (u_5 + u_6 + u_7 + u_8)}{4(\varepsilon^+ + \varepsilon^-)}, \tag{10.15}$$

where u_c denotes the potential of the center point. In practice, a vector $\boldsymbol{e}_k'$ is constructed to record the positions of nodes from 1 to 8 in the vector $\boldsymbol{\phi}_I$ and the their contributions to u_c, according to (10.15). A similar formula to (10.12) can be used to efficiently calculate the discrete probabilities for the transition cube whose dielectric interface is at height 1/2:

$$\boldsymbol{P}_k' = -\left(\left(\boldsymbol{E}_{11} - \boldsymbol{E}_{13}\boldsymbol{D}_{33}^{-1}\boldsymbol{E}_{31}\right)^{-\mathrm{T}}\boldsymbol{e}_k'\right)^{\mathrm{T}}\boldsymbol{E}_{12}. \tag{10.16}$$

Figure 10.7 shows the distributions of the precharacterized GFT for the configuration: $\varepsilon^- = 2.6$ and $\varepsilon^+ = 5$. Note that the plot in Fig. 10.7a is similar to that of single-dielectric surface Green's function in Iverson and Le Coz [69], and the plots in Fig. 10.7b, c demonstrate that there is much larger probability to hop toward the dielectric region with larger permittivity. By setting $\varepsilon^- = \varepsilon^+$, we have used the single-dielectric Green's function to validate the accuracy of the numerical GFTs obtained with FDM.

To calculate the weight value in (10.5), we firstly derive

$$\omega\left(\boldsymbol{r},\boldsymbol{r}^{(1)}\right)=-\frac{1}{g\cdot L}\cdot\frac{\frac{\partial P\left(\boldsymbol{r},\boldsymbol{r}^{(1)}\right)}{\partial x}n_x+\frac{\partial P\left(\boldsymbol{r},\boldsymbol{r}^{(1)}\right)}{\partial y}n_y+\frac{\partial P\left(\boldsymbol{r},\boldsymbol{r}^{(1)}\right)}{\partial z}n_z}{P\left(\boldsymbol{r},\boldsymbol{r}^{(1)}\right)}, \tag{10.17}$$

where L is the edge length of the first cube, and we assume $\widehat{n}\left(\boldsymbol{r}\right)=[n_x,\,n_y,\,n_z]^{\mathrm{T}}$. Here $P(\boldsymbol{r},\,\boldsymbol{r}^{(1)})$ denotes the surface Green's function for the unit-size cube, and $\boldsymbol{r}$ and $\boldsymbol{r}^{(1)}$ are the corresponding points in the unit-size cube. Since we have obtained the GFT representing $P(\boldsymbol{r},\,\boldsymbol{r}^{(1)})$, the finite difference formula for partial derivative should be employed to calculate the weight value. For example, $\partial P/\partial x$ stands for the sensitivity of the probabilities with respect to the x-axis coordinate of the center point of cube. Since $\boldsymbol{P}_k$ in (10.11) is the GFT associated with the center point, we denote the GFTs associated with its six neighbor points to be $\boldsymbol{P}_{k+1}$, $\boldsymbol{P}_{k-1}$, $\boldsymbol{P}_{k+N}$, $\boldsymbol{P}_{k-N}$, $\boldsymbol{P}_{k+N^2}$, and $\boldsymbol{P}_{k-N^2}$. Then, with the centered difference formula, we have

$$\frac{\partial\boldsymbol{P}_k}{\partial x}\approx\frac{\boldsymbol{P}_{k+1}-\boldsymbol{P}_{k-1}}{4h}=-\left(\left(\boldsymbol{E}_{11}-\boldsymbol{E}_{13}\boldsymbol{D}_{33}^{-1}\boldsymbol{E}_{31}\right)^{-\mathrm{T}}\tilde{\boldsymbol{e}}_1\right)^{\mathrm{T}}\boldsymbol{E}_{12}, \tag{10.18}$$

$$\frac{\partial\boldsymbol{P}_k}{\partial y}\approx\frac{\boldsymbol{P}_{k+N}-\boldsymbol{P}_{k-N}}{4h}=-\left(\left(\boldsymbol{E}_{11}-\boldsymbol{E}_{13}\boldsymbol{D}_{33}^{-1}\boldsymbol{E}_{31}\right)^{-\mathrm{T}}\tilde{\boldsymbol{e}}_N\right)^{\mathrm{T}}\boldsymbol{E}_{12}, \tag{10.19}$$

$$\frac{\partial\boldsymbol{P}_k}{\partial z}\approx\frac{\boldsymbol{P}_{k+N^2}-\boldsymbol{P}_{k-N^2}}{4h}=-\left(\left(\boldsymbol{E}_{11}-\boldsymbol{E}_{13}\boldsymbol{D}_{33}^{-1}\boldsymbol{E}_{31}\right)^{-\mathrm{T}}\tilde{\boldsymbol{e}}_{N^2}\right)^{\mathrm{T}}\boldsymbol{E}_{12}. \tag{10.20}$$

Here vector $\tilde{\boldsymbol{e}}_i$ $(i=1,N,N^2)$ is defined as

$$\tilde{\boldsymbol{e}}_i(j)=\begin{cases}1/4h, & j=k+i\\ -1/4h, & j=k-i\\ 0, & \text{otherwise}\end{cases}. \tag{10.21}$$

Equations (10.18), (10.19), and (10.20) are the formulas for calculating the partial derivatives in (10.17). With a small value of h, they have sufficient accuracy. In practice, the values of $\frac{\partial P/\partial x}{P}$, $\frac{\partial P/\partial y}{P}$, and $\frac{\partial P/\partial z}{P}$ for each boundary panel (the position of $\boldsymbol{r}^{(1)}$) are precomputed and stored as the weight value tables (WVTs). With the WVTs, the weight value (10.17) in actual walk can be calculated quickly.

It should be pointed out that precharacterizing the WVT for the multi-dielectric cube enlarges the size of the first transition domain and is very important for improving the computational efficiency. Since ω is inversely proportional to the size of the first cube L, and without the multi-dielectric WVT, the first cube is bounded by the nearest dielectric interface, the value of ω would be very large if there is a thin dielectric layer near the Gaussian surface. This large sample value would further slow down the convergence of MC procedure. Therefore, the multi-dielectric WVT can enlarge the size of first transition cube and thus prominently reduces the number of walks to attain a certain accuracy goal [203].

For a predefined value of N (segment number along cube edge), N positions of dielectric interface are considered. Totally, $4N$ times of equation solution with the coefficient matrix of $(\boldsymbol{E}_{11} - \boldsymbol{E}_{13}\boldsymbol{D}_{33}^{-1}\boldsymbol{E}_{31})^{\mathrm{T}}$ (N for GFTs and $3N$ for WVTs) are involved for a pair of permittivities. As for the memory usage, $6N^2$ real numbers are needed to present a single GFT. For the $4N$ GFTs and WVTs which characterize a dielectric pair, the memory space is needed to store $24N^3$ about numbers. If $N = 31$ and the double-precision number is used, the storage is about 5.5 MB. If three dielectric layers are allowed in the precharacterized transition cube, the memory storing the three-layer GFTs and WVTs would be about $N/2$ times of that for two-layer GFTs and WVTs, due to the two dielectric interfaces. This will make the extra memory for the two-layer, three-layer GFTs and WVTs approaching to 1 GB, if a process technology includes 10 three-consecutive-dielectric configurations. Besides, allowing three dielectric layers in the precharacterized transition domain will increase the complexity of implementation, such as matching the interface heights in actual transition cube and those in the precharacterized. Therefore, we only precharacterize the domain with two dielectric layers to well balance the memory usage and computational efficiency.

10.2.3 The FRW Algorithm with Multi-dielectric GFTs and WVTs

The following algorithm describes the FRW algorithm utilizing the multi-dielectric GFTs and WVTs.

Algorithm 10.2 The FRW Using Multi-dielectric GFTs and WVTs

(1) Load the precomputed transition probabilities, weight values for single-dielectric cubic transition domain;
(2) Load the precomputed multi-dielectric GFTs and WVTs;
Construct the Gaussian surface enclosing master conductor j;
$C_{ji} := 0, \forall\, i;\ npath := 0;$
Repeat
  $npath: = npath + 1;$
  Pick a point $\boldsymbol{r}^{(0)}$ on Gaussian surface, and then generate a cubic transition domain (may contain two dielectrics) T centered at it; pick a point $\boldsymbol{r}^{(1)}$ on the surface of T according to the precomputed probabilities, and then calculate the weight value ω with the help of the WVTs;
  While the current point is not on a conductor **do**
    Construct the largest conductor-free cubic domain containing at most two dielectrics;
    Pick a point on the domain surface, according to the transition probabilities of cubic domain;
  End
  $C_{ji} := C_{ji} + \omega\ ;$ //the current point is on conductor i
Until the stopping criterion is met
$C_{ji} := C_{ji}/npath, \forall\, i;$

In Algorithm 10.2, a shrinking operation for the cube may be performed after generating the transition cube (in steps 6 and 8). Its aim is to adjust the size of transition cube to make the height of dielectric interface to be i/N ($i = 1, 2, \ldots, N-1$) or 1/2 of the cube size. Therefore, the precharacterized GFTs and WVTs can be used without loss of accuracy.

10.3 The Techniques for Variance Reduction

In this section, we firstly review the basic ideas of importance sampling and stratified sampling, followed by the variance reduction techniques proposed in Batterywala and Desai [12]. Then, we propose the technique to minimize the variance of weight value distribution in the FRW and a comprehensive variance reduction Scheme.

10.3.1 Background

The MC method can be used to calculate a multidimensional integral:

$$I = \int_{\Omega} f(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}. \tag{10.22}$$

If we pick n random samples $\boldsymbol{x}_i$ uniformly distributed in Ω,

$$I_n = A \cdot \frac{\sum_{i=1}^{n} f(\boldsymbol{x}_i)}{n} \tag{10.23}$$

becomes an estimate for I, where $A = \int_{\Omega} d\boldsymbol{x}$. Note that I_n is the mean value of $A{\cdot}f(\boldsymbol{x}_i)$ and is also a random quantity due to the randomness of $\boldsymbol{x}_i$. The variance of function $A{\cdot}f(\boldsymbol{x})$ can be approximated by the variance of the discrete estimates $A{\cdot}f(\boldsymbol{x}_i)$:

$$\mathrm{Var}_n = \frac{A^2 \sum_{i=1}^{n} \left(f(\boldsymbol{x}_i) \right)^2 - n \cdot I_n^2}{n}. \tag{10.24}$$

Due to the central limit theorem [113], I_n obeys the normal probability distribution, and its standard deviation (Std) can be estimated with

$$\mathrm{err}_n \approx \frac{\sqrt{\mathrm{Var}\left(A \cdot f(\boldsymbol{x})\right)}}{\sqrt{n}} \approx \frac{\sqrt{\mathrm{Var}_n}}{\sqrt{n}}, \tag{10.25}$$

where $\mathrm{Var}(A{\cdot}f(\boldsymbol{x}))$ denotes the variance of function $A{\cdot}f(\boldsymbol{x})$. err_n in (10.25) is also called the $1-\sigma$ error of I_n. With (10.25), it can be observed that the error of I_n is inversely proportional to the square root of the number of samples. Since I_n conforms to the normal distribution, the $1-\sigma$ error err_n means the error bound at the confidence level of 68 %. And it has the confidence level of 99.7 % that the result is within the error of $3{\cdot}\mathrm{err}_n$.

According to the above discussion, reducing the variance of the integrand $f(\boldsymbol{x})$ brings smaller err_n, and therefore I_n with higher accuracy, for a given number of sample points. The idea of *importance sampling* (IS) is to find a positive-valued function $q(\boldsymbol{x})$ such that $\int_\Omega q(\boldsymbol{x})d\boldsymbol{x}=1$ and $q(\boldsymbol{x})$ is nearly proportional to the function $f(\boldsymbol{x})$. Then, (10.22) is converted to

$$I=\int_\Omega \frac{f(\boldsymbol{x})}{q(\boldsymbol{x})}q(\boldsymbol{x})\mathrm{d}\boldsymbol{x}. \tag{10.26}$$

We can use $q(\boldsymbol{x})$ as a probability density function for the sampling of Ω and evaluate $f(\boldsymbol{x}_i)/q(\boldsymbol{x}_i)$ as the estimate in the MC procedure. This means that $f(\boldsymbol{x})/q(\boldsymbol{x})$ is regarded as the integrand function in the MC integration. Since $q(\boldsymbol{x})$ is nearly proportional to $f(\boldsymbol{x})$, the variance of the new integrand is much less than that of $f(\boldsymbol{x})$. Therefore, the MC procedure corresponding to (10.26) will converge much faster for a given error threshold.

Another variance reduction technique is the stratified sampling (SS), whose idea is to divide the original integral to several integrals on its subdomains (also called "strata") and then calculate them separately. For example, dividing the Ω in (10.22) into two equal-sized subdomains Ω_a and Ω_b, we have

$$I=I_a+I_b=\int_{\Omega a} f(\boldsymbol{x})\mathrm{d}\boldsymbol{x}+\int_{\Omega b} f(\boldsymbol{x})\mathrm{d}\boldsymbol{x}. \tag{10.27}$$

If the MC method is used to calculate I_a and I_b separately, the formula (10.23) can be substituted with

$$I_n'=I_{n_a}^{(a)}+I_{n_b}^{(b)}=\frac{A}{2}\cdot\frac{\sum_{i=1}^{n_a} f(\boldsymbol{x}_{a,i})}{n_a}+\frac{A}{2}\cdot\frac{\sum_{i=1}^{n_b} f(\boldsymbol{x}_{b,i})}{n_b}, \tag{10.28}$$

where n_a and n_b are the sampling numbers in Ω_a and Ω_b, respectively ($n_a+n_b=n$). Due to the stochastic independence of $I_{n_a}^{(a)}$ and $I_{n_b}^{(b)}$, the variance of $I_n^{'}$ is the sum of the variances of $I_{n_a}^{(a)}$ and $I_{n_b}^{(b)}$. It is proved that [113]

$$\mathrm{Var}\left(I_n'\right)<\mathrm{Var}\left(I_n\right), \tag{10.29}$$

if $f(\boldsymbol{x})$ has different mean value in subdomains Ω_a and Ω_b. Furthermore, $\mathrm{Var}(I_n^{'})$ achieves its minimum value if $(n_a/n_b)^2$ equals to the ratio of variances of function $f(\boldsymbol{x})$ in Ω_a and Ω_b. Because the $\mathrm{Var}(I_n)$ is the square of the Std in (10.25), calculating (10.28) with the idea of SS produces the estimation to the original integral with less $1-\sigma$ standard error. Thus, we need fewer MC samples to achieve a specified accuracy criterion.

The above theory of statistics is applicable to the FRW algorithm, because Eq. (10.4) is an integral. With the formula for the variance of capacitance estimates, similar to (10.24), the $1-\sigma$ error of capacitance result is predictable [69, 82, 83]. So, usually the accuracy criterion is set as the termination condition of FRW algorithm. In practice, the algorithm checks the error after every certain number of walks. Once the $1-\sigma$ error is below the specified accuracy goal, the algorithm will terminate.

In Batterywala and Desai [12], the variance reduction techniques were proposed for the FRW-based capacitance extraction. Firstly, the capacitance was regarded as the integral of normal electric field intensity on the Gaussian surface. Based on that the electric field diminishes with the inverse of distance, the function $q(\boldsymbol{x})$ for IS was simply defined to be proportional to $1/D(\boldsymbol{r})$, where $D(\boldsymbol{r})$ was the distance of point $\boldsymbol{r}$ to the conductor inside the Gaussian surface. The stratified sampling was also applied, which decomposed the first transition cube's surface into 16 pieces (strata), and the FRW calculations were performed for each stratum separately. It should be pointed out that a variant FRW formula, instead of (10.4) (referred to as the standard FRW formula), was used in Batterywala and Desai [12]:

$$Q_j = \oint_{G_j} F(\boldsymbol{r})\, g \oint_{S^{(1)}} \omega_c\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \phi\left(\boldsymbol{r}^{(1)}\right) \mathrm{d}\boldsymbol{r}^{(1)} \mathrm{d}\boldsymbol{r}. \tag{10.30}$$

The variant weight value is

$$\omega_c\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) = -\frac{\nabla_r P^{(1)}\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})}{g}. \tag{10.31}$$

Accordingly, the sampling scheme in $S^{(1)}$ obeys the uniform probabilities, rather than the probabilities derived by the surface Green's function. In the next two subsections, we will show that the variant FRW formula (10.30) causes worse convergence behavior than the standard FRW (10.4), and the variance reduction techniques were not fully exploited in Batterywala and Desai [12].

10.3.2 The Importance Sampling with the Weight Values Averaged

In the FRW algorithm, the weight value (10.5), or (10.17), is the estimate of capacitance in the MC procedure. From another viewpoint, it is the estimation to

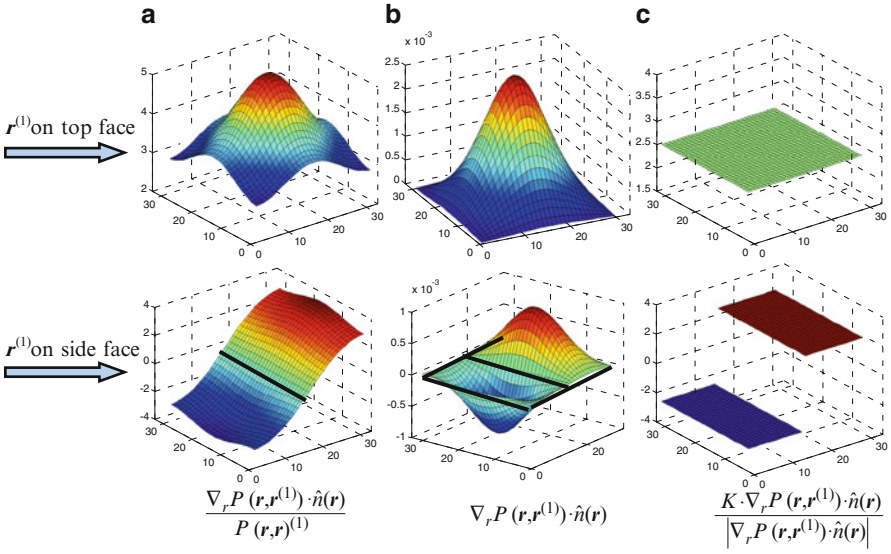

Fig. 10.8 The values in WVT for $\boldsymbol{r}^{(1)}$ on top and side faces of the cube. (**a**) For the ω of the standard FRW, (**b**) for the ω_c of the variant FRW, and (**c**) for the $\widetilde{\omega}$ using the importance sampling

the integrand function (the normal electric field intensity) of the integral (10.4). Although the integrand is unknown, we can instead reduce the variance of weight values by transforming the FRW formula with the IS technique. It is noticed that with the help of WVT, we actually know the possible values of the weight value. Suppose $\boldsymbol{r}$ is on the top surface of the Gaussian surface, which means $\widehat{n}(\boldsymbol{r})$ is along the z-direction. In Fig. 10.8, we draw the distributions of the opposite of weight value for the $\boldsymbol{r}^{(1)}$ on top and side faces of the cube. Figure 10.8a corresponds to the ω in the standard FRW, while Fig. 10.8b corresponds to the ω_c in the variant FRW formula (10.31).

With Fig. 10.8a, b and (10.17), we know that ω and ω_c are nonpositive for $\boldsymbol{r}^{(1)}$ on the top face, while they may approach to zero for $\boldsymbol{r}^{(1)}$ on the side face (outlined with bold lines). This kind of value distribution demonstrates that the weight value has a certain degree of variance. And the variance of ω_c seems larger than that of ω, which is verified by the numerical experiments showing slower convergence rate of the variant FRW.

We now seek a function $q(\boldsymbol{x})$ to modify the integral formula (10.4) so as to reduce the variance of resulting weight value, based on the precomputed WVTs and GFTs. For the transition cube with a specific dielectric configuration, we first calculate

$$K = \oint_S \left|\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}\left(\boldsymbol{r}\right)\right| \mathrm{d}\boldsymbol{r}^{(1)}, \tag{10.32}$$

where S and $P(\boldsymbol{r}, \boldsymbol{r}^{(1)})$ are the surface and the surface Green's function of the unit-size cube, respectively. Since $\boldsymbol{r}$ is the center of cube, K is a positive constant. We transform (10.4) to

$$\begin{aligned} Q_j &= \oint_{G_j} F(\boldsymbol{r}) \, g \oint_S - \frac{\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})}{L \cdot g} \phi\left(\boldsymbol{r}^{(1)}\right) \mathrm{d}\boldsymbol{r}^{(1)} \mathrm{d}\boldsymbol{r} \\ &= \oint_{G_j} F(\boldsymbol{r}) \, g \oint_S \frac{-K \cdot \nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})}{L \cdot g \left|\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})\right|} \cdot \frac{\left|\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})\right|}{K} \times \\ &\quad \times \phi\left(\boldsymbol{r}^{(1)}\right) \mathrm{d}\boldsymbol{r}^{(1)} \mathrm{d}\boldsymbol{r}, \end{aligned} \tag{10.33}$$

where L is the size of the first cube, with which the integral domain is converted from $S^{(1)}$ to the unit-size cube S.

We define function $q(\boldsymbol{x})$ to be

$$q\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) = \frac{\left|\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})\right|}{K} = \frac{\left|\omega\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right)\right| P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right)}{K}. \tag{10.34}$$

Here $\omega(\boldsymbol{r}, \boldsymbol{r}^{(1)})$ denotes the weight value for the unit-size cube, which is a little bit different from (10.5). According to (10.32), $q(\boldsymbol{r}, \boldsymbol{r}^{(1)})$ can serve as the PDF for sampling S. Then,

$$Q_j = \oint_{G_j} F(\boldsymbol{r}) \, g \oint_S \frac{-K \cdot \nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})}{L \cdot g \left|\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})\right|} \cdot q\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \phi\left(\boldsymbol{r}^{(1)}\right) \mathrm{d}\boldsymbol{r}^{(1)} \mathrm{d}\boldsymbol{r}. \tag{10.35}$$

This suggests a new sampling scheme, where the weight value is

$$\tilde{\omega}\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) = \begin{cases} -K/(L \cdot g), & \text{if } \nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r}) > 0 \\ K/(L \cdot g), & \text{if } \nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r}) < 0. \end{cases} \tag{10.36}$$

Note it is impossible that $\boldsymbol{r}^{(1)}$ has a value such that $\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r}) = 0$, according to the sampling probability function (10.34). In Fig. 10.8c, we draw the distribution of the opposite of the new weight value (10.36). It is almost constant for cubes with same size and dielectric configuration. Thus, the new FRW scheme may exhibit much less variance.

Now, the point sampling on the first cube obeys the new probabilities obtained from $q(\boldsymbol{r}, \boldsymbol{r}^{(1)})$, which can be generated by multiplying the original GFT and WVT, element by element. Similar to the derivation of (10.17), the new probabilities

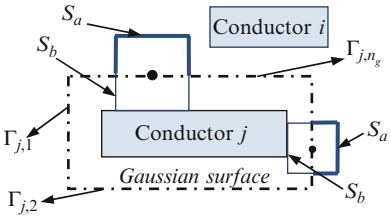

Fig. 10.9 A 2-D view illustrating the n_g faces of a Gaussian surface and the two half surfaces of the first transition cube

$q(\boldsymbol{r}, \boldsymbol{r}^{(1)})$ differ with different directions of $\widehat{n}(\boldsymbol{r})$. We tabulate it with tables called GFT_ISs, which have the same size of storage as the original WVTs. The new weight values (10.36) are also precomputed and tabulated, but requiring much less storage.

10.3.3 The Comprehensive Variance Reduction Scheme

Although the weight values are averaged with formula (10.35), they still fluctuate when the first cube varies with different positions of $\boldsymbol{r}$. This kind of variance can be further handled with the SS technique. Since for a given cube the weight value has only two nonzero values of (10.36), decomposing the cube's surface into 16 strata as in Batterywala and Desai [12] is unnecessary. We just divide the cube's surface into two strata, one with positive weight value and the other with negative weight value. Instead, the Gaussian surface is decomposed to capture the different variances of weight values on its different faces. Suppose the Gaussian surface has n_g faces (see Fig. 10.9). The original Eq. (10.4) is converted to

$$\begin{aligned} Q_j &= \sum_{k=1}^{n_g} \int_{\Gamma_{j,k}} F(\boldsymbol{r})\, g \oint_S -\frac{\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})}{L \cdot g} \phi\left(\boldsymbol{r}^{(1)}\right) \mathrm{d}\boldsymbol{r}^{(1)} \mathrm{d}\boldsymbol{r} \\ &= \sum_{k=1}^{n_g} \int_{\Gamma_{j,k}} F(\boldsymbol{r})\, g \int_{S_a} -\frac{\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})}{L \cdot g} \phi\left(\boldsymbol{r}^{(1)}\right) \mathrm{d}\boldsymbol{r}^{(1)} \mathrm{d}\boldsymbol{r} + \\ &\quad \sum_{k=1}^{n_g} \int_{\Gamma_{j,k}} F(\boldsymbol{r})\, g \int_{S_b} -\frac{\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})}{L \cdot g} \phi\left(\boldsymbol{r}^{(1)}\right) \mathrm{d}\boldsymbol{r}^{(1)} \mathrm{d}\boldsymbol{r}, \end{aligned} \tag{10.37}$$

where $\Gamma_{j,k}$ stands for the kth face of the Gaussian surface G_j. S_a stands for the surface of the scaled half cube outside G_j, and S_b stands for the surface of the scaled half cube inside G_j. Note that $\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})$ always has a positive value for $\boldsymbol{r}^{(1)}$

on S_a and a negative value for $\boldsymbol{r}^{(1)}$ on S_b.[1] Now, Q_j becomes the summation of $2n_{\mathrm{g}}$ integrals. Each integral can be calculated separately. For example,

$$I_k = \int_{\Gamma_{j,k}} F(\boldsymbol{r})\, g \int_{S_a} -\frac{\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})}{L \cdot g} \phi\left(\boldsymbol{r}^{(1)}\right) \mathrm{d}\boldsymbol{r}^{(1)} \mathrm{d}\boldsymbol{r}, \quad \left(1 \le k \le n_{\mathrm{g}}\right). \tag{10.38}$$

Because $\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})$ is positive for $\boldsymbol{r}^{(1)}$ on S_a, we define

$$K_a = \int_{S_a} \nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})\, \mathrm{d}\boldsymbol{r}^{(1)}. \tag{10.39}$$

Then,

$$q_a\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) = \frac{\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})}{K_a} \tag{10.40}$$

becomes a PDF for the second integral in (10.38). We derive

$$I_k = A_k \int_{\Gamma_{j,k}} \frac{F(\boldsymbol{r})\, g}{A_k} \int_{S_a} -\frac{K_a}{L \cdot g} q_a\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \phi\left(\boldsymbol{r}^{(1)}\right) \mathrm{d}\boldsymbol{r}^{(1)} \mathrm{d}\boldsymbol{r}, \quad \left(1 \le k \le n_{\mathrm{g}}\right), \tag{10.41}$$

where $A_k = \int_{\Gamma_{j,k}} F(\boldsymbol{r})\, g d\boldsymbol{r}$. This can be interpreted by an FRW procedure, where the sampling on S_a obeys the PDF of $q_a(\boldsymbol{r}, \boldsymbol{r}^{(1)})$ and the corresponding weight value is

$$\tilde{\omega}_k\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) = \frac{-K_a}{L \cdot g}, \quad \left(1 \le k \le n_{\mathrm{g}}\right). \tag{10.42}$$

Similar derivation can be applied to

$$I_k = \int_{\Gamma_{j,k-n_{\mathrm{g}}}} F(\boldsymbol{r})\, g \int_{S_b} -\frac{\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})}{L \cdot g} \phi\left(\boldsymbol{r}^{(1)}\right) \mathrm{d}\boldsymbol{r}^{(1)} \mathrm{d}\boldsymbol{r}, \quad \left(k > n_{\mathrm{g}}\right), \tag{10.43}$$

[1] Supposing $\widehat{n}(\boldsymbol{r})$ is along the positive z-axis direction, $\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r}) = \partial P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) / \partial z = \lim_{\Delta z \to 0} \left[P\left(\boldsymbol{r} + \Delta z, \boldsymbol{r}^{(1)}\right) - P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right)\right] / \Delta z$. Because $P(\boldsymbol{r}, \boldsymbol{r}^{(1)})$ means the correlation (or contribution) coefficient of the potential at point $\boldsymbol{r}^{(1)}$ to the potential at point $\boldsymbol{r}$, a movement of $\boldsymbol{r}$ toward $\boldsymbol{r}^{(1)}$ makes this correlation stronger. So, if $\boldsymbol{r}^{(1)}$ is on S_a (see Fig. 10.9), $P(\boldsymbol{r} + \Delta z, \boldsymbol{r}^{(1)}) - P(\boldsymbol{r}, \boldsymbol{r}^{(1)}) > 0$, and thus $\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r})$ has a positive value. For $\boldsymbol{r}^{(1)}$ on S_b, we can similarly show that $\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}(\boldsymbol{r}) < 0$.

provided that K_b and $q_b(\boldsymbol{r}, \boldsymbol{r}^{(1)})$ are defined similarly to K_a and $q_a(\boldsymbol{r}, \boldsymbol{r}^{(1)})$, respectively. And for $k > n_g$, we let A_k equal to A_{k-n_g}. Suppose n_k walks are used to calculate I_k, $(k = 1, \ldots, 2n_g)$, and the weight values returned are $\tilde{\omega}_{k,1}, \ldots, \tilde{\omega}_{k,n_k}$. The estimation for (10.37) with the SS technique is computed as

$$\overline{Q}_j = \sum_{k=1}^{2n_g} \theta_k = \sum_{k=1}^{2n_g} \left(\frac{A_k}{n_k} \sum_{i=1}^{n_k} \tilde{\omega}_{k,i} \right). \tag{10.44}$$

And the corresponding variance is computed piecewise as

$$\mathrm{Var}\left(\overline{Q}_j\right) = \sum_{k=1}^{2n_g} \left(\frac{A_k^2}{n_k^2} \sum_{i=1}^{n_k} \tilde{\omega}_{k,i}^2 - \frac{\theta_k^2}{n_k} \right). \tag{10.45}$$

It should be pointed out that (10.44) and (10.45) need to be explained for calculating the capacitances and variance of capacitance estimates. Comparing (10.39) and (10.40) with (10.32) and (10.34), we conclude that the new sampling scheme is compatible with the scheme in the last subsection. We do not need to change the precharacterization scheme, which tabulates the surface Green's function $P(\boldsymbol{r}, \boldsymbol{r}^{(1)})$, the new PDF $q(\boldsymbol{r}, \boldsymbol{r}^{(1)})$ in (10.34), and the K in (10.32). Actually, K is double of K_a in (10.39),[2] and the complete formula of (10.42) is

$$\tilde{\omega}_k\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) = \begin{cases} \frac{-K}{2L \cdot g}, & 1 \le k \le n_g \\ \frac{K}{2L \cdot g}, & n_g < k \le 2n_g. \end{cases} \tag{10.46}$$

So, the weight value tabulated as in the last subsection can still be used.

In our implementation, the values of n_k, $(k = 1, \ldots, 2n_g)$ are not predefined. Instead, the FRW procedure corresponding to the IS technique in the last subsection is performed. The actual walks are registered to the $2n_g$ strata, followed by calculating the capacitance value and error estimation with formulas similar to (10.44) and (10.45). Finally, the IS technique in Batterywala and Desai [12] can also be applied to handle the first integral. All these consist of a comprehensive variance reduction scheme, and the resulting FRW algorithm is described as Algorithm 10.3.

[2] Because $\oint_S P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \mathrm{d}\boldsymbol{r}^{(1)} = 1, \oint_S \nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}\left(\boldsymbol{r}\right) \mathrm{d}\boldsymbol{r}^{(1)} = 0$. According to the definition of K_a and K_b, $\oint_S \nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}\left(\boldsymbol{r}\right) \mathrm{d}\boldsymbol{r}^{(1)} = K_a + K_b = 0$. We have proved that $K_a > 0$, which means K_b is its opposite number. So, $K = \oint_S \left|\nabla_r P\left(\boldsymbol{r}, \boldsymbol{r}^{(1)}\right) \cdot \widehat{n}\left(\boldsymbol{r}\right)\right| \mathrm{d}\boldsymbol{r}^{(1)} = K_a - K_b = 2K_a$.

Algorithm 10.3 The FRW Using the Variance Reduction Techniques

1: For each dielectric configuration occurring in the test case, load the available GFT, GFT_ISs, and WVTs.
2: Construct the Gaussian surface enclosing master conductor j;
3: $C_{ji} := 0, \forall\, i;\ n_k := 0, \forall\, k;$
4: **Repeat**
5: Pick a point $\boldsymbol{r}^{(0)}$ on Gaussian surface, and then generate a cubic transition domain (may contain two dielectrics) T centered at it; pick a point $\boldsymbol{r}^{(1)}$ on the surface of T according to the precomputed GFT_IS, and then calculate the weight value $\tilde{\omega}_k$ with (10.46) considering the position of $\boldsymbol{r}^{(0)}$ and $\boldsymbol{r}^{(1)}$;
6: **While** the current point is not on a conductor **do**
7: Construct the largest conductor-free cubic domain containing at most two dielectrics;
8: Pick a point on the domain surface, according to the precomputed GFT of the cubic domain;
9: **End**
10: Register $\tilde{\omega}_k$ to strata k for C_{ji};//the current point is on conductor i
11: $n_k := n_k + 1$;
12 Calculate the standard error with formula similar to (10.45);
13: **Until** the stopping criterion is met
14: Calculate $C_{ji}, \forall\, i$ with formula similar to (10.44).

10.4 The Space Management Technique and Parallel Implementation

In this section, the space management technique is firstly presented to reduce the computing time for structures with a large amount of conductors. Then, the parallel technique is presented to boost the computational speed of FRW-based capacitance extraction on the multi-core/multi-CPU platform.

10.4.1 The Space Management Technique

For each FRW hop, the distance from the current point to conductors should be measured to construct the maximum conductor-free transition cube. Its computing time should be reduced as short as possible because millions of hops are involved in the FRW algorithm. In Bansal [7], several data structures were discussed to speed up the distance queries in the FRW algorithm. Here we present a space management technique based on the Octree spatial data structure [127] to organize the conductors in the problem such that the distance is calculated for only a few of conductors for each hop. This technique is required for the case involving many conductors. Each Octree node represents a cubic spatial domain, which may be decomposed into eight

subdomains, and each one corresponds to a child node in the tree. For each node of Octree, we want to have a number of possible nearest conductors (candidate list) for the spatial points it represents. To build such an Octree, we firstly define the *dominant* relationship.

Definition 10.1 T is a cubic domain, and B_1 and B_2 are conductors. B_1 *dominates* B_2 regarding T, iff $\forall P \in T$, $d(P, B_1) \leq d(P, B_2)$, where $d(\ ,\ )$ is the function of the ∞-norm distance in 3-D space.

The Octree is constructed through a procedure of inserting conductors. Firstly, the Octree only has one node (root node), representing the whole problem space. Then, all conductors in the problem are inserted to this node, one by one, with the following algorithm.

Algorithm 10.4 InsertToOctree (Conductor B, Node T)

```
1:  If T is a leaf node then
2:    For each b in the candidate list of T do
3:      If b dominate B then return;
4:      Elseif B dominate b then
5:        Remove b from the candidate list of T;
6:      Endif
7:    Endfor
8:    Add B to the candidate list of T;
9:    If the size of T's candidate list is larger than threshold then
10:     Divide T into 8 child nodes: T1, ..., T8;
11:     For each b in the candidate list of T do
12:       For i = 1 to 8, InsertToOctree(b, Ti); EndFor
13:     EndFor
14:   Endif
15: Else
16:   For each child c of T do
17:     InsertToOctree(B, c);
18:   Endfor
19: Endif
```

Note that a *threshold* is predefined as the maximum size of the candidate list. In steps 12 and 17 of Algorithm 10.4, there are recursive calls. The *dominant* relationship of conductor blocks can be judged with their distances to the cubic domain of current node.

Once the Octree is established for the extracted structure, in each hop only the conductors in the candidate list of the leaf node that the current point belongs to are inquired to calculate the distance. So, the time complexity of each hop is about $O(h_o + t_o)$, where h_o is the height of Octree and t_o is the threshold. We have tested two structures with 201 and 402 wires, respectively. Numerical results show the running time of FRW algorithm is reduced by 72 or 88 %, with negligible memory overhead.

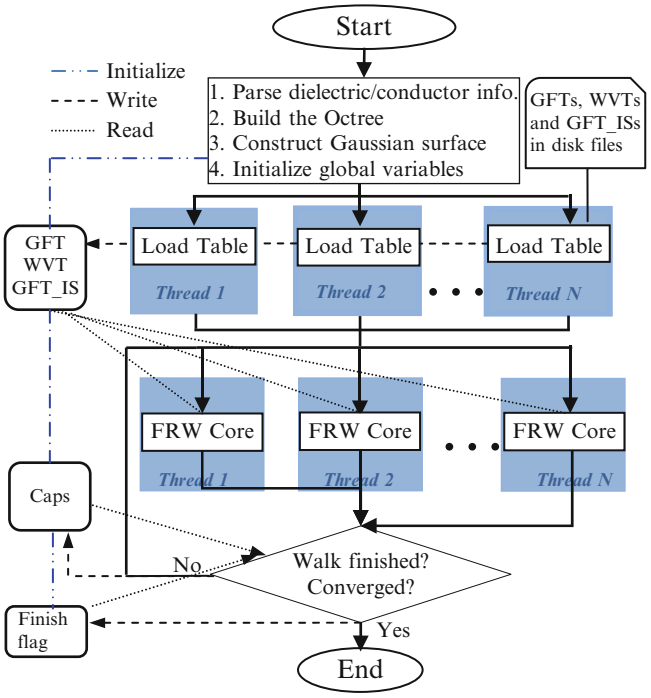

Fig. 10.10 The flowchart of the parallel FRW on multi-core platform

10.4.2 The Parallel Implementation

The FRW algorithm is very suitable for parallelization, since the walks are independent to each other. Figure 10.10 shows the flowchart of the parallel FRW algorithm on a multi-core/multi-CPU platform. Multiple threads are allocated to execute the random walk procedure (steps 5–9 in Algorithm 10.3; denoted by "FRW core" in Fig. 10.10). In order to reduce the expense of communication, a unique random number generator [97] is kept for each thread. The "lock" operation only happens when updating the value of capacitance and checking the program termination criteria with total number of walks or estimated error. This check is performed for every m walks ($m = 1{,}000$). The work of loading GFTs, WVTs, and GFT_ISs is also parallelized without difficulty. The *pthread* APIs are used in our implementation.

For large structure including thousands of conductor blocks, serially building the Octree (step 2 in Fig. 10.10) may cost a lot of time. To reduce this expense, the construction of Octree can also be parallelized. The basic idea is that in the insertion operation, every conductor is independent to each other and the operation on every tree node is also independent to each other.

10.5 Numerical Results

We have developed a C++ program RWCap with the proposed techniques. To evaluate the efficiency of different techniques, four subversions of RWCap are defined:

- RWCap(O): The basic multi-dielectric FRW (Algorithm 10.1)
- RWCap(M): The FRW using multi-dielectric GFT and WVT (Algorithm 10.2)
- RWCap(IS): The FRW using the IS technique proposed in Sect. 10.4.2
- RWCap(R): The FRW using all variance reduction techniques (Algorithm 10.3)

The termination criterion of RWCap is that the $1-\sigma$ error of self-capacitance becomes smaller than 1 % of the mean value. This means that the relative error is within 3 % (in a 99.7 % confidence level). The FDM-based method to generate the multi-dielectric GFTs and WVTs is implemented in MATLAB, using the functions for sparse matrix. Unless otherwise stated, the value of N (segment number along cube edge) is set to 31.

As revealed in Batterywala and Desai [12], the placement of Gaussian surface has an impact on the performance of FRW algorithm. In that work, the Gaussian surface was constructed to enclose the master conductor in such a way that points on it are equidistant from the conductor and from other adjacent conductors. However, this strategy is not valid for the case with rare conductors, where along some directions the equidistant position becomes very far away from the master conductor. This worsens the convergence behavior of the FRW procedure. So, we adopt a different strategy which firstly finds the six axis-direction equidistant positions from the master and then use the minimum of the distances to construct the Gaussian surface surrounding the master conductor.

The experiments are carried out on a Linux server with Intel Xeon E5620 8-core CPU of 2.40 GHz. The computational time listed does not include that for loading the multi-dielectric GFTs, WVTs, and GFT_ISs, because this can be performed only once for a process technology. We store these data with binary format files, and the time for loading them is less than 0.5 s for each tested process technology.

10.5.1 Test Cases

The typical 180-nm technology and 45-nm technology described in Deschacht et al. [39] are considered. The cross sections of the test structures with three metal layers are shown in Fig. 10.11, where the relative permittivity of each dielectric layer is labeled. For the 180-nm technology, the width and height of each wire are 300 nm and 530 nm, respectively, while in the 45-nm technology, the width and height are 70 nm and 140 nm, respectively. The first test structure includes three parallel wires in M2 layer. For the second structure, two sets of 19 wires are added to M1 and M3 layers, respectively, with random width and spacing (see Fig. 10.12). The third

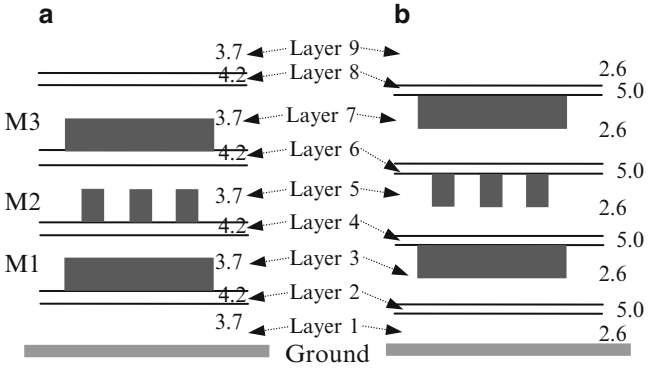

Fig. 10.11 The cross section of process technology used in the test cases. (**a**) The 180-nm technology, where layers 2, 4, 6, and 8 are thin dielectrics with thickness of 200 nm, and the thickness of layers 3, 5, and 7 is 1,180 nm. (**b**) The 45-nm technology, where layers 2, 4, 6, and 8 are thin dielectrics with thickness of 40 nm, and the thickness of layers 3, 5, and 7 is 220 nm

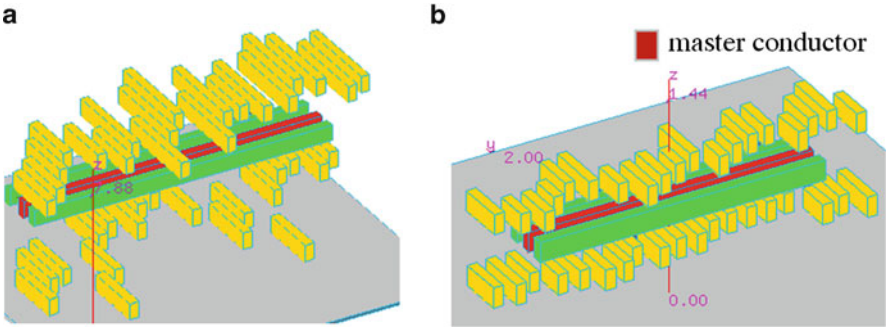

Fig. 10.12 The 3-D views of the test cases: (**a**) case 2 and (**b**) case 5

structure is a 5×5 bus structure distributed in M2 and M3 layers. Combining with the two process technologies, we obtain six test cases as listed in Table 10.1. For each case, the conductor in the middle is set as the master conductor, and the self- and coupling capacitances of the master conductor are extracted.

We firstly validate the proposed techniques with experiments of serial computing. Then, the efficiency of the parallel RWCap is demonstrated.

10.5.2 Validating the Multi-dielectric FRW Algorithm

The six test cases are extracted with RWCap(O) and RWCap(M), whose results are compared in Table 10.2. Besides the self-capacitance C_{self}, we also list a major

Table 10.1 The description of six test cases

	Technology	Description
Case 1	180-nm technology	3 parallel wires. Wire length is 10,000 nm, and the spacing is 300 nm to the left and 500 nm to the right
Case 2		41 wires in 3 metal layers. Wire width is 300 nm
Case 3		5×5 bus structure. Wire length is 3,000 nm, and the spacing is 300 nm
Case 4	45-nm technology	3 parallel wires. Wire length is 2,000 nm, and the wire spacing is 70 nm
Case 5		41 wires in 3 metal layers. Wire width is 70 nm
Case 6		5×5 bus structure. Wire length is 700 nm, and the spacing is 70 nm

Table 10.2 The computational results of RWCap(O) and RWCap(M) (capacitance in unit of 10^{-16} F, memory in unit of MB)

Case	RWCap	C_{self}	C_{c1}	Std(C_{c1})	# walks	Memory	Time (s)	Speedup
1	O	18.90	6.42	1.6 %	6,098 K	1.6	1468.55	N/A
	M	18.61	6.32	1.5 %	206 K	13.4	17.46	84
2	O	19.50	5.22	1.9 %	5,512 K	1.7	368.88	N/A
	M	19.40	5.33	1.7 %	179 K	13.6	5.13	72
3	O	7.25	2.61	2.0 %	4,180 K	1.6	533.88	N/A
	M	7.34	2.64	1.7 %	132 K	13.5	6.13	87
4	O	3.60	1.62	1.5 %	8,075 K	1.5	2554.09	N/A
	M	3.64	1.61	1.5 %	160 K	13.4	15.93	160
5	O	3.95	1.36	1.9 %	6,195 K	1.6	516.48	N/A
	M	3.94	1.35	1.8 %	122 K	13.5	4.12	125
6	O	1.43	0.525	1.7 %	7,417 K	1.6	1261.99	N/A
	M	1.42	0.504	1.7 %	150 K	13.5	9.59	132

coupling capacitance as C_{c1} which is between the master and its right-side neighbor wire. Std(C_{c1}) in Table 10.2 is the ratio of this coupling capacitance's Std to its mean value. Note that the Std of C_{self} equals 1 % of its mean value, which is the termination criterion.

The capacitance values in Table 10.2 validate the accuracy of RWCap(M), i.e., Algorithm 10.2. For several cases, the Std of the coupling capacitance is even reduced with the RWCap(M). The values of Std(C_{c1}) less than 2 % also demonstrate that the FRW algorithm is able to calculate the major coupling capacitance accurately. The data of CPU time suggest that the technique of utilizing the multi-dielectric GFTs and WVTs speeds up the original FRW algorithm by several tens to more than one hundred times. And the memory overhead is only 12 MB, which is negligible. The benefit brought by the multi-dielectric WVTs is actually larger than that of GFTs, because the former avoids the first transition domain with very small size and therefore largely reduces the number of walks for convergence. This is more prominent for the 45-nm technology, because there is a 40-nm-thickness thin dielectric layer just above the master conductor. We find out with the experiments that the speedup brought by the multi-dielectric GFTs solely is from 2X to 3X.

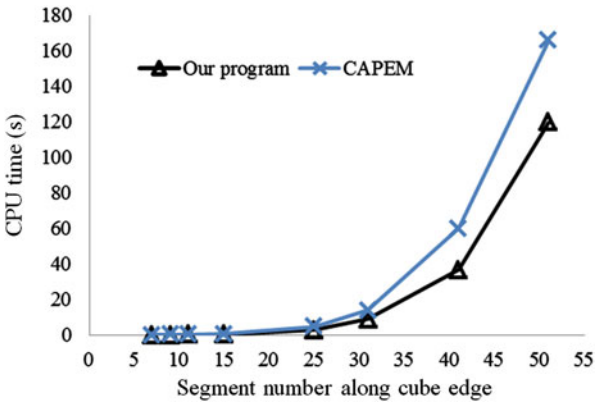

Fig. 10.13 The computing time for generating the multi-dielectric GFTs and WVTs with various segment numbers along the cube's edge. The results are obtained on a Linux server with Intel Xeon 3.0-GHz CPU

We have also solved "case 4" with FastCap [99] and CAPEM [38]. CAPEM is a binary-code program, whose technical details are not published yet. FastCap's result is 3.55×10^{-16} F, with the time of 34.7 s and 1.8-GB memory. It validates the accuracy of RWCap (with discrepancy of 2.5 %) and shows that FastCap is inferior even for this smaller case due to a lot of panels caused by the discretization of dielectric interfaces. With 100-K walks, CAPEM's result is 3.62 ± 0.18 (5 %) while consuming 773 s. To compare fairly, we also run RWCap(M) for 100-K walks, which produces C_{self} of 3.49 ± 0.045 (1.3 %) after 9.5 s. This means RWCap(M) is 81X faster than CAPEM and has better convergence rate.

CAPEM also has the function of precharacterizing the transition cube with multilayered dielectrics. While setting different values of N (segment number along cube edge), we have compared the computational times of generating the GFTs and WVTs for a dielectric configuration of cube with CAPEM and our program in Fig. 10.13. The figure demonstrates that the precharacterization procedure with the presented techniques is faster than that of CAPEM. For one tested process technology, there are two permittivity pairs, and the precharacterization time for RWCap(M) is about 524 s. In the precharacterization, generating GFTs and generating WVTs cost about 1/4 and 3/4 of the total time, respectively. Also note that while performing the precharacterization for RWCap(IS) and RWCap(R), the computational time will not increase, as we have discussed at the end of Sect. 10.4.2.

Comparing the CPU times in Table 10.2, it is revealed that the time is not scaled with the complexity of structure. This is the distinct property of the FRW algorithm, as compared with the conventional algorithms. For example, case 4 has the most complex structure including 41 wires (see Fig. 10.12b), but it costs the least CPU time. The reason is that for a case with dense environment conductors, the walk will terminate quickly.

10.5.3 Validating the Variance Reduction Techniques

To demonstrate the efficiency of the proposed variance reduction techniques, RWCap(M) is compared with RWCap(IS) and RWCap(R). The results are listed in Table 10.3. From this table, we can see that the importance sampling and stratified sampling techniques largely reduce the number of walks for the same accuracy criterion. For the six cases, they bring the speedup ratios from 2.8X to 3.6X (**3.2X** on average). We have also implemented the variant FRW formula (30) and the corresponding variance reduction technique in Batterywala and Desai [12] with our program, which is referred to as "variant FRW" and "variant FRW(R)" in Table 10.3. Compared with them, our technique also shows more than 2X speedup, and RWCap(IS) using the technique of importance sampling is comparable or superior to the "variant FRW(R)." Note that the stratified sampling in Batterywala and Desai [12] only considers the distribution of single-dielectric weight value, which makes its speedup not prominent for the multi-dielectric cases.

The capacitance values in Table 10.3 partially validate the accuracy of RWCap(R). For the complete validation, we run RWCap(R) for 3,000 times and then plot the distribution of the extracted C_{self}. Figure 10.14 shows this distribution for case 1, accompanied by the distribution obtained with RWCap(M). Both plots approximate to the normal distribution, and the calculated Std ($<$1 % of mean value) suggests that the accuracy of extracted C_{self} is not degraded with the variance reduction. Besides, we compare the Std(C_{c1}) in Tables 10.3 and Table 10.2 and find out that RWCap(R) also preserves the accuracy of coupling capacitance.

10.5.4 Comparing with the Fast Boundary Element Method

Experiments are carried out to compare RWCap(R) with a fast boundary method utilizing the quasi-multiple medium (QMM) technique [114, 179]. The QMM-accelerated BEM (called QBEM) [114] is suitable for multi-dielectric cases and has exhibited about 10X speedup over the FastCap. The 45-nm-technology cases are tested, and the computational results are listed in Table 10.4. To compare their performance for large-scale structure, the 7th test case is constructed, which includes 403 wires in the three metal layers (with similar structure to case 5 but has 200 crossing wires in each neighbor layer).

For the extraction with QBEM, the Neumann boundary is needed to surround the extracted structure, while the FRW algorithm assumes the Dirichlet boundary far from the structure. So, the self-capacitance in the problem for FRW is always larger than that in the problem for QBEM. Based on this analysis, we find out that the discrepancy of QBEM and RWCap(R) on C_{self} (see Table 10.4) is fairly small. As for the coupling capacitance, the discrepancy of both solvers on C_{c1} is no more

Table 10.3 The computational results of RWCap(R) and its comparison with other FRW-based algorithms (capacitance in unit of 10^{-16} F, time in unit of second)

	RWCap(M)		RWCap(IS)		RWCap(R)							Variant FRW		Variant FRW(R)	
Case	# walk	Time	# walk	Time	# walk	C_{self}	C_{c1}	Std(C_{c1})	Time	Speedup[a]	Speedup[b]	# walk	Time	# walk	Time
1	206 K	17.46	151 K	11.80	61 K	18.40	6.41	1.9 %	4.82	3.6	2.7	348 K	29.07	145 K	12.83
2	179 K	5.13	131 K	3.72	56 K	19.51	5.47	2.1 %	1.51	3.4	2.4	307 K	8.46	128 K	3.65
3	132 K	6.13	96 K	4.22	46 K	7.28	2.69	1.8 %	2.10	2.9	2.4	246 K	11.54	110 K	5.04
4	160 K	15.93	114 K	10.75	51 K	3.65	1.63	1.6 %	4.93	3.3	2.8	292 K	30.01	130 K	13.95
5	122 K	4.12	90 K	2.89	44 K	3.91	1.36	1.9 %	1.47	2.8	2.4	232 K	7.77	100 K	3.52
6	150 K	9.59	104 K	6.13	51 K	1.44	0.517	1.9 %	3.13	3.1	2.6	267 K	16.65	131 K	8.21

[a]The speedup ratio to RWCap(M)

[b]The speedup ratio to the variance reduction accelerated variant FRW algorithm in Batterywala and Desai [12], i.e., variant FRW(R)

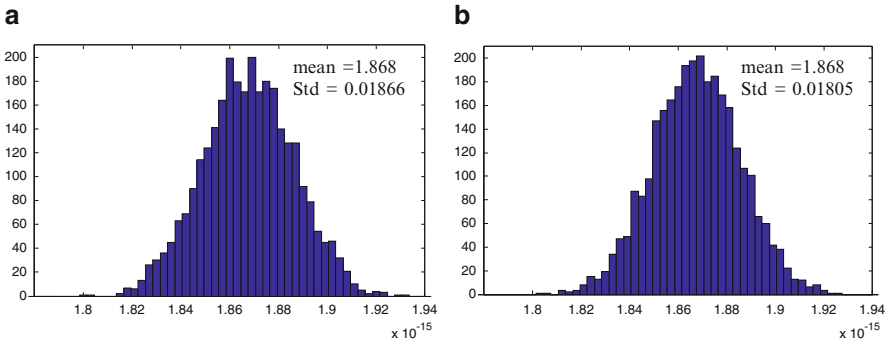

Fig. 10.14 The distribution of the C_{self} extracted with (**a**) RWCap(M) and (**b**) RWCap(R) for 3,000 runs

Table 10.4 The comparison of RWCap(R) and QBEM for the 45-nm-technology cases (capacitance in unit of 10^{-16} F, time in unit of second)

	QBEM				RWCap(R)					
Case	# panel	C_{self}	C_{c1}	Time	C_{self}	Dis.	C_{c1}	Dis.	Time	Speedup
4	9334	3.63	1.61	2.79	3.65	0.6 %	1.67	3.7 %	4.93	0.57
5	12211	3.81	1.32	2.44	3.95	3.7 %	1.36	3.0 %	1.47	1.7
6	13883	1.38	0.504	4.09	1.44	4.3 %	0.517	2.6 %	3.13	1.3
7[a]	54980	50.4	20.7	180	52.1	3.4 %	18.5	−10.6 %	4.96	36

[a]A 3-layer structure with totally 403 wires under the 45-nm technology

than 10 %. From the table, we also see that QBEM is faster than RWCap(R) for the smallest case, but this advantage is lost for more complex cases. For the largest 403-wire case, RWCap(R) becomes 36X faster than QBEM.

10.5.5 Validating the Efficiency of Parallel Computing

On the 8-core machine, experiments are carried out to validate the parallel implementation of RWCap(R). We firstly set 100,000 walks to be the termination criterion of RWCap and run it with different numbers of threads. The computing time for case 1 is listed in Table 10.5, which shows the high efficiency of the parallelization. For all cases, nearly 7X speedup can be achieved on the 8-core machine. While setting the standard error of 1 % as the termination criterion, the parallel speedup of RWCap decreases a little because more walks than required are assigned to balance the workload. However, it still achieves over 6X speedup on the 8-core machine, as shown in Table 10.5. The experiment is also performed for the 7th test case with 403 wires, whose results show the parallel RWCap costs about 0.81 s with the 1 % termination criterion. Comparing with the serial-computing time in Table 10.4, this means a 6.1X speedup.

Table 10.5 The computational time of the parallel RWCap(R) for case 1

	# thread	Time (s)	Speedup
Walk-number criterion (100 K)	1	8.05	N/A
	2	4.06	1.98
	4	2.03	3.97
	8	1.16	6.94
Accuracy criterion (1 %)	1	4.79	N/A
	8	0.70	6.82

10.6 Summary

By numerically precharacterizing the surface Green's function and the weight value for cubic transition domain with two dielectric layers, an efficient FRW algorithm is proposed for multi-dielectric capacitance extraction. The algorithm is further accelerated with a new variance reduction scheme based on the importance sampling and the stratified sampling. Along with the parallel computing technique, an efficient FRW solver called RWCap has been developed.

Numerical results on test structures with 9 dielectric layers under the 180- and 45-nm process technologies have validated the efficiency of proposed techniques. While comparing with other existing FRW algorithms, RWCap shows several tens of times speedup over CAPEM [38] and more than 2X speedup over the technique proposed in [12]. The comparisons with fast BEM-based solvers show that RWCap has comparable computing speed for small cases but for large case is several tens of times more efficient in CPU time and memory usage.

The RWCap program has been shared on the website of the first author [125].

Chapter 11
FRW-Based Solver for Chip-Scale Large Structures

Different from the conventional field-solver methods for capacitance extraction, such as BEM and FEM, the FRW algorithm is based on the Monte Carlo (MC) method for calculating integral, which converts the capacitance calculation to the procedure of random walk. Theoretically, the cost of the FRW algorithm for calculating the capacitances related to a specified master conductor is independent of the number of conductors in the problem. This makes the FRW algorithm with unique advantages for handling large-scale interconnect structures.

We have presented an FRW-based capacitance solver (called RWCap) in Chap. 10. However, only small- or medium-scale interconnect structures were demonstrated; the experiments with really large-scale interconnect structures (say, with over 10,000 wires) were not carried out to validate the efficiency of RWCap. In this chapter, efficient space management techniques are presented for the aim of extracting the large structure with up to one million wires as a whole. Both runtimes of constructing and inquiring the space management structures for interconnect wires have been substantially reduced. As a result, the improved FRW algorithm becomes faster than RWCap for thousands of times while extracting a single net and several to tens of times while extracting 100 nets.

11.1 Motivation

Today, the FRW algorithm has become the kernel of several commercial capacitance solvers (such as QuickCap™ and Rapid3D™). With the parallel computing techniques, these solvers have been applied to the block-level or chip-level extraction tasks in the sign-off verification of VLSI circuits. Note that the approximating approaches are used in them to handle the actual VLSI process technology. The theory of accurate and efficient FRW algorithm for large-scale multi-dielectric VLSI structures is not well established. Despite a space management approach, which divides and organizes the whole simulation domain into an Octree structure and

W. Yu and X. Wang, *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*, DOI 10.1007/978-3-642-54298-5_11,

generates the candidate lists representing cell's neighbor conductors, was employed in RWCap [187], it is not efficient enough if the real large-scale structures with thousands of conductors are handled. Another approach based on an array data structure has been used and achieved remarkable speedup over the K-D tree-based technique [121]. However, this approach is not well compared with other techniques, and its details are not published. In Bansal [7], several data structures were discussed to speed up the distance queries in the FRW algorithm. But they either have large space complexity or are preliminary ideas without implementation. Because the geometric computation in the FRW algorithm costs a large portion of the total computing time, it is important to have an efficient space management approach to accelerate it for the extraction of large-scale VLSI layouts.

The space management approach facilitates finding the nearest conductor during the procedure of random walk. It has similar purpose as the techniques for the nearest neighbor query (NNQ) problem in the area of database [108, 122]. However, the data set of conductors in the problem of capacitance extraction is not dynamically changed as that in the general database. Compared with the techniques for the NNQ problem (e.g., that based on the R-tree and the traversal with pruning strategies [122]), more efficient technique and data structure can be devised for the capacitance extraction problem.

A major step in each FRW hop is constructing a conductor-free transition cube (step 8 in Algorithm 10.2). Since millions of hops are involved in the FRW algorithm, the nearest distance from current point to conductor which is needed for constructing the transition cube should be calculated as fast as possible. A space management approach based on the Octree data structure [127] was presented in Sect. 10.4, to organize the conductors. With this approach, only the distances to a few of conductors, which are in the so-called candidate list, are calculated for each hop. Thus, the computing time of each hop is largely reduced while extracting the structure involving a lot of conductors. However, for larger interconnect structures, the approach proposed in Yu et al. [187] (also Sect. 10.4) consumes a lot of time for constructing the Octree. With the experiments on a test case with 37,062 conductors, we have found out that the Octree's construction time increases to 29 min. Note that extracting a net costs only 2.9 s on average, under a 0.5 % accuracy criterion.

11.2 Basic Operations of Space Management and Accelerating Techniques

In this section, we firstly present the fundamental concepts of the space management for the FRW algorithm. Then, several techniques are proposed to accelerate the generation and inquiry of the space management structure.

11.2.1 Basic Operations

The basic idea of space management is to divide the whole 3-D domain into organized small subdomains. Here we call the subdomain *spatial cell*. Each cell is attached with a *candidate list* of conductors such that for any point in the cell, its nearest conductor is in the list. With this approach, the inquiry of nearest conductor can be executed very quickly.

A primary operation for constructing the space management structure is generating the candidate list for a cell. We shall check the conductors one by one to see if they should be added to the list. To be rigorous, we give the following definitions, where $x_{\min}(\cdot)$ and $x_{\max}(\cdot)$ denote the minimum and maximum x-coordinates of a cuboid, respectively.

Definition 11.1 The X-distance between a cuboid A and a point P (x, y, z) is the larger one of $x_{\min}(\mathrm{A}) - x$ and $x - x_{\max}(\mathrm{A})$. The Y-distance and Z-distance between A and P are defined similarly.

Definition 11.2 The distance between a cuboid A and a point P is the maximum of their X-, Y-, and Z-distances.

Definition 11.3 The X-distance between the two cuboids (A and B) is the larger one of $x_{\min}(\mathrm{A}) - x_{\max}(\mathrm{B})$ and $x_{\min}(\mathrm{B}) - x_{\max}(\mathrm{A})$. The Y-distance and Z-distance between A and B are defined similarly.

Definition 11.4 The distance between the two cuboids is defined as the maximum of their X-, Y-, and Z-distances.

Note in our problem, there are only Manhattan geometries. It is assumed that each spatial cell or conductor is a cuboid and can be described by its two opposite vertices. We emphasize that the distances defined above can be a negative value, which is different from the ∞-norm. Below, $d(,)$ denotes the distance between the two cuboids or a cuboid and a point.

Definition 11.5 T is a spatial cell, and B_1 and B_2 are two conductors. If for any point $\mathrm{P} \in \mathrm{T}$, and $\mathrm{P} \notin \mathrm{B}_1 \cup \mathrm{B}_2$, $d(\mathrm{P}, \mathrm{B}_1) \leq d(\mathrm{P}, \mathrm{B}_2)$, we say B_1 *dominates* B_2 regarding T.

The domination relationship is the key to create the candidate list. If B_1 dominates B_2 regarding cell T, and B_1 is already in the candidate list of T, B_2 should not be inserted to the candidate list. This is because for any point in the cell, its distance to B_1 is not larger than that to B_2. So, B_2 can be ignored while finding the nearest conductor for any point in the cell. The domination relationship of two conductors can be judged by considering eight vertices of the cell. Algorithm 11.1 describes the procedure of candidate checking for a conductor [187, 203].

Algorithm 11.1 CandidateCheck (Conductor B, Cell T)

```
1.  For each b in the candidate list of T do
2.    If b dominate B then
3.      return;
4.    Elseif B dominate b then
5.      Remove b from the candidate list of T;
6.    Endif
7.  Endfor
8.  Add B to the candidate list of T.
```

11.2.2 Improving the Candidate Checking with Distance Limit

To get the candidate list for a cell, the procedure of candidate checking (Algorithm 11.1) should be performed for all conductors. For a large-scale problem where there are thousands of conductors and thousands of cells, this results in large computational expense. Especially, if only the capacitances of several critical nets are needed, constructing the space management structure would dominate the total computing time. After giving two definitions, we propose a technique to reduce the time for generating the candidate list.

Definition 11.6 The *size* of a cuboid cell is defined as the maximum of the cell's length, width, and height.

Definition 11.7 The *distance limit* of a cell is the minimum distance between the cell and a conductor in its candidate list plus the cell's size. L(T) denotes the distance limit of cell T.

In Fig. 11.1, we show four typical relationships between a cell and a conductor.[1] In all these cases, the X-distance between the cell and the conductor is $x_3 - x_2$, which is also the distance between the two cuboids. Note in three cases, the distance is a negative value. If a cell is intersected, but not fully occupied, by a cuboid conductor, the absolute value of their distance will be less than the size of the cell (see Fig. 11.1b, c). This means that the distance limit of the cell is positive. For the situation where the cell is fully occupied by a cuboid conductor (see Fig. 11.1d), the cell's candidate list is not useful because it is impossible that the random walk would stop in the cell. So, we shall only consider how to create the candidate list for the other kinds of cells, whose distance limit is definitely a positive value. We conclude with the following theorem.

[1] We regard the situation where conductor B is in cell T as a special case of the situation shown in Fig. 11.1c.

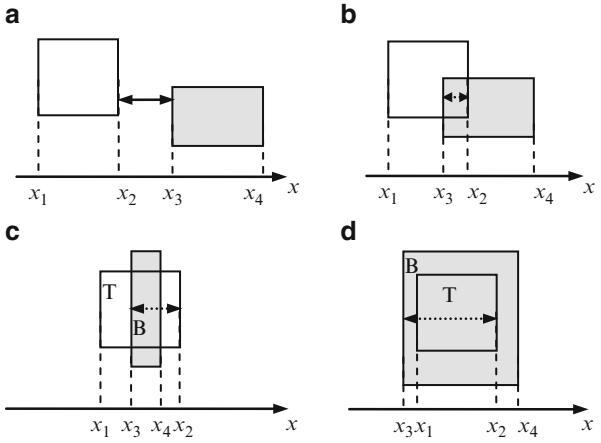

Fig. 11.1 Four topological relationships between a cell T and a conductor B: (**a**) separated, (**b**) intersected, (**c**) crossed, and (**d**) fully occupied. The X-distance of two cuboids is max $(x_3 - x_2, x_1 - x_4)$, and for the four cases, we have $d(\mathrm{T}, \mathrm{B}) = x_3 - x_2$

Theorem 11.1 *For any valid cell* T *in the space management structure, its distance limit* $L(\mathrm{T}) > 0$. *And if the cell is intersected by a conductor,*[2] $L(\mathrm{T}) \leq l$, *where* l *is the size of* T.

Actually, the distance limit $L(\mathrm{T})$ is an upper bound of the distance to nearest conductor from any point in cell T. While checking conductors one by one for generating the candidate list, the value of cell's distance limit dynamically changes. During this course, if the distance between a conductor and the cell is not less than the cell's distance limit, the conductor must have been dominated by a conductor in the candidate list and should not be inserted.

Theorem 11.2 *If the distance between a conductor and a cell is not less than the cell's distance limit, the conductor must have been dominated by a conductor in the cell's candidate list.*

Proof From Definition 11.7, there is a conductor B_0 in the cell T's candidate list, which fulfills

$$d\left(\mathrm{T}, \mathrm{B}_0\right) + l = L\left(\mathrm{T}\right), \tag{11.1}$$

where l is the size of cell T. For any point P in the conductor-free space within T, we have

$$d\left(\mathrm{P}, \mathrm{B}_0\right) \leq d\left(\mathrm{T}, \mathrm{B}_0\right) + l = L\left(\mathrm{T}\right). \tag{11.2}$$

[2]Here "intersect" includes touching the outer boundary of the cell.

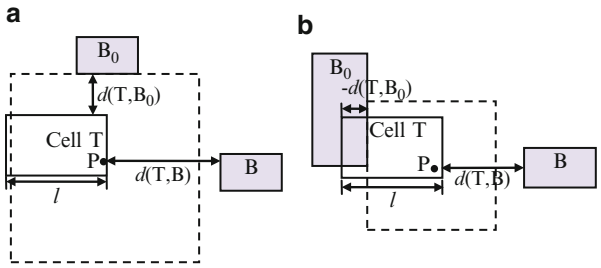

Fig. 11.2 The distance between a point P in cell T and a conductor B_0 is not larger than d(T, B_0) + l, where l is the size of the cell. (**a**) B_0 does not intersect T; (**b**) B_0 intersects T. The transition cube centered at P is also drawn with *dashed line*

The inequality can be explained with Fig. 11.2.

Now, if there is a conductor B, and $d(\mathrm{T}, \mathrm{B}) \geq L(\mathrm{T})$, then

$$d\,(\mathrm{P}, \mathrm{B}_0) \leq d\,(\mathrm{T}, \mathrm{B}) \leq d\,(\mathrm{P}, \mathrm{B})\,. \tag{11.3}$$

The last inequality in (11.3) is because the distance between the two cuboids is not larger than the distance between one of them and a point in the other. So, from (11.3), we see that B_0 dominates B.

With Theorem 11.2, the operations of candidate checking can be largely reduced for some conductors. We improve Algorithm 11.1 by utilizing the distance limit to reduce the judgments of domination relationship. This becomes Algorithm 11.2.

Algorithm 11.2 CandidateCheck2 (Conductor B, Cell T)

```
1.    d := d(B, T); l is the size of T;
2.    If d ≥ L(T) then return;
3.    For each b in the candidate list of T do
4.      If b dominate B then return;
5.      Elseif B dominate b then
6.        Remove b from the candidate list of T;
7.      Endif
8.    Endfor
9.    Add B to the candidate list of T;
10.   If (d+l)<L(T) then L(T) := d+l;
```

In Sect. 11.4, we will show that the distance limit brings large acceleration.

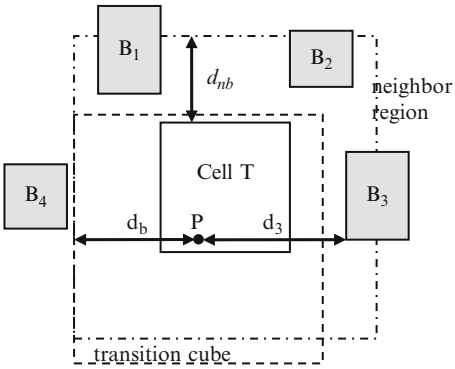

Fig. 11.3 Cell T's neighbor region and the conductors B_1, B_2, and B_3 in its incomplete candidate list. For point P, the distance to the boundary of neighbor region d_b should be used to construct the transition cube

11.2.3 Incomplete Candidate List

Up to now, it is assumed that we always construct the largest conductor-free transition cube. So, all conductors need to be considered for generating the candidate list. However, if we ignore the requirement for the largest size, it becomes not necessary to consider all conductors. We may just check the conductors in the cell's *neighbor region*. With this strategy, we create an *incomplete* candidate list, which is not for generating the largest transition cube. Thus, the computational time for the construction of the space management structure can be further reduced. On the contrary, the candidate list in the previous discussion is called the *complete* candidate list.

Using the complete candidate list, the candidate list for a cell cannot be empty, although the candidate conductor may be very far away. However, if there is no conductor in the neighbor region, the incomplete candidate list becomes empty. Here, we define the *neighbor region* of a cell by inflating it with the *extension size* d_{nb} (see Fig. 11.3). During the random walk procedure, by traversing the candidate list, we obtain the minimum distance from current point to the candidate conductors. If this distance is larger than the shortest distance between the point and the boundary of neighbor region, the latter should be used to guarantee that the transition cube does not intersect any conductor. Figure 11.3 illustrates this situation, where d_3 is the distance obtained with the incomplete candidate list but d_b should be used to construct the transition cube. This example also shows that the transition cube generated is not the largest one.

With larger neighbor region, the incomplete candidate list generated will be more close to the complete candidate list. The following theorem states in certain situations, the incomplete candidate list is actually a *complete* one.

Theorem 11.3 *Cell* T*'s candidate list generated with the conductors in* T*'s neighbor region is a complete candidate list, if the distance limit of* T *is not larger than the extension size* d_{nb} *of the neighbor region.*

Proof For any conductor B outside the neighbor region of T, we know $d(\mathrm{B}, \mathrm{T}) \geq d_{\mathrm{nb}}$. If the cell's distance limit $L(\mathrm{T}) \leq d_{\mathrm{nb}}$, we get $d(\mathrm{B}, \mathrm{T}) \geq L(\mathrm{T})$.

According to Theorem 11.2, B must have been dominated by a conductor in T's candidate list. So, no conductor outside the neighbor region should be added to the candidate list, and this candidate list is already a complete one.

The following corollary can be derived from Theorem 11.3 and Theorem 11.1:

Corollary 11.1 *Suppose* T *is a cell intersected with conductor. If the extension size* d_{nb} *of* T*'s neighbor region is not smaller than* T*'s size, the incomplete candidate list generated accordingly is a complete one.*

We can still use Algorithm 11.2 to generate the incomplete candidate list with the neighbor region. The difference is that we only check the conductors in the neighbor region, instead of all conductors. Alternatively, we may check all conductors but set the initial distance limit to be d_{nb}, instead of the infinity.

If we employ the idea of incomplete candidate list, the following algorithm is designed for generating the transition cube at a given point.

Algorithm 11.3 Find Nearest Distance (Point P)

1. Get the cell T which contains P;
2. d_{min}:= distance between P and boundary of T's neighbor region;
3. **For** each B in the candidate list of T **do**
4. $d_{\mathrm{p}} := d(\mathrm{P}, \mathrm{B})$;
5. **If** $d_{\mathrm{p}} < d_{\mathrm{min}}$ **then**
6. $d_{\mathrm{min}} := d_{\mathrm{p}}$;
7. **Endif**
8. **Endfor**
9. **return** d_{min}.

Notice that the distance obtained by inquiring the incomplete candidate list is restricted by the boundary of neighbor region. This would increase the number of hops in a walk and harm the computational efficiency of the FRW procedure. So, a suitable size of neighbor region should be chosen to achieve the best overall efficiency.

11.2.4 Reducing the Time for Inquiring the Candidate List

To find the nearest conductor from the current location of random walk, we need to locate the cell which contains the point and then traverse its candidate list. The first step depends on the space management structure. Here we consider the second step whose purpose is to calculate the minimum distance between the point and the conductors in the candidate list.

The traversal can be terminated once the distance between the point and a candidate is zero. This means that the point is on the conductor surface, and thus the random walk should terminate. Otherwise, the traversal is usually not terminated until the end of list is reached.

Here, we propose a strategy to terminate the traversal of candidate list earlier, which is based on sorting the candidate conductors in the ascending order of their distance to the cell. Denote the candidate list by $\{B_i\}$, where B_i is the ith item in the list. After traversing the first $j-1$ items, we get

$$d_{\min,\ j-1} = \min_{1\le i\le j-1} \{d\ (\mathrm{P}, \mathrm{B}_i)\}, \tag{11.4}$$

which is the current value of the minimum distance to conductor for the inquiry point P. While traversing the jth item B_j, if

$$d\left(\mathrm{B}_j, \mathrm{T}\right) \ge d_{\min, j-1}, \tag{11.5}$$

we derive

$$d\left(\mathrm{P}, \mathrm{B}_j\right) \ge d\left(\mathrm{B}_j, \mathrm{T}\right) \ge d_{\min, j-1}. \tag{11.6}$$

This means B_j will not affect the value of the minimum distance. Because the list is sorted, i.e.,

$$d\left(\mathrm{B}_{j+k}, \mathrm{T}\right) \ge d\left(\mathrm{B}_j, \mathrm{T}\right), \quad k = 1, 2, \ldots, \tag{11.7}$$

traversing the rest of the list would be unnecessary. Thus, the traversal can be terminated immediately. This proves the following theorem.

Theorem 11.4 *Suppose the candidate list* $\{\mathrm{B}_i\}$ *of cell* T *is sorted in the ascending order of the conductor's distance to* T. *While traversing the items in the list,* $\mathrm{B}_1, \mathrm{B}_2, \ldots$, *one by one, for calculating the minimum distance between* B_i *and a point* P *in* T, *if there is an item* B_j *such that*

$$d\left(\mathrm{B}_j, \mathrm{T}\right) \ge \min_{1\le i\le j-1} \{d\ (\mathrm{P}, \mathrm{B}_i)\}, \tag{11.8}$$

the traversal can be terminated at B_j.

Because the candidate list is usually not long, and we calculate and sort the distances between a cell and its candidates in the construction stage, the overhead of this fast inquiry technique is negligible. Note that for the candidate conductor which intersects the cell, its distance to the cell is negative. So, it cannot be B_j in (11.8). We should always check them in the traversal. However, for the candidates outside the cell, they may be skipped in the traversal with the proposed strategy. With the help of sorted candidate list, Algorithm 11.3 can be improved to become Algorithm 11.4.

Algorithm 11.4 Find Nearest Distance 2 (Point P)

```
1.  Get the cell T which contains P;
2.  d_min:= distance between P and boundary of T's neighbor region;
3.  For each B in the candidate list of T do
4.    d_p := d(P, B);
5.    If d_p==0 then
6.      return 0;
7.    Elseif d(B, T)≥d_min then
8.      return d_min;
9.    Elseif d_p<d_min then
10.     d_min:= d_p;
11.   Endif
12. Endfor
13. return d_min.
```

In Algorithm 11.4, two distances are needed in each iteration step. The d(B, T) is pre-calculated for every cell T and its candidate conductors in our implementation, but the distance d(P, B) has to be calculated online. This distance between a point and a conductor cuboid is calculated for millions of times in the FRW algorithm. So, we shall reduce its calculating time by all means. In our implementation, we follow Definitions 11.1 and 11.2 and calculate the distance with six subtractions and five maximums of two numbers. It is more efficient than the implementation with a couple of "if-else" clauses.

11.3 Space Management Structures and Approaches

There are several spatial data structures, such as K-D tree, uniform grid, and Octree [127]. The K-D tree is most commonly used for the multidimensional indexing. Like the binary tree, each node in K-D tree has two child nodes which correspond to two subdomains got by splitting the domain corresponding to the node. The Cartesian coordinate planes are alternately set to be the splitting plane, and the splitting position is chosen to make equal inquiry frequency in the two subdomains. However, in our problem of capacitance extraction, we cannot estimate the inquiry frequency of the stopping points in the FRW procedure. So, the K-D tree has no advantage for our application.

In this section, we firstly improve the Octree-based approach [187] with the proposed accelerating techniques. Then, we present two grid-based space management approaches, with and without the candidate list, respectively. Finally, a hybrid approach using both grid and Octree structures is proposed.

11.3.1 The Improved Octree-Based Approach

We firstly consider the construction of the Octree structure. Due to the property of a tree structure, for inserting each conductor to it, we must traverse it from the root node to a leaf node.[3] If the distance limit is defined for the nodes in traversal, unnecessary judgments of domination relationship can be avoided. If the distance between a conductor and a node is larger than the latter's distance limit, the conductor should not be inserted into the node and its descendants. When a conductor is inserted to a leaf node, we shall check if updating its distance limit is needed. We can also check and update the distance limits of all ancestor nodes. However, this is not done because its cost is usually larger than the benefit it brings. Besides, the candidate list of a node becomes useless once it is divided into eight child nodes. We shall delete the candidate list to release the memory space. Algorithm 11.5 describes the procedure of inserting a conductor to an Octree node. The whole structure is constructed by inserting all conductors one by one to the root node.

Algorithm 11.5 InsertToOctree (Conductor B, Node T)

```
1.  If d(B, T) ≥ L(T), then return;
2.  If T is a leaf node then
3.    CandidateCheck2(B, T);          //Algorithm 11.2
4.    If length of T's candidate list > thres1 and size of T > thres2 then
5.      Divide T into 8 child nodes: T1, ..., T8;
6.      For each b in the candidate list of T do
7.        For i = 1 to 8 do
8.          InsertToOctree(b, Ti);
9.        Endfor
10.     Endfor
11.   Endif
12. Else
13.   For each child node c of T do
14.     InsertToOctree(B, c);
15.   Endfor
16. Endif
```

Note in step 4 of Algorithm 11.5, a threshold for the size of leaf node (cell) is set. Because in some situations, dividing a node into eight child nodes cannot reduce the number of candidate conductors, this size threshold is necessary for avoiding the endless division of domain.

[3]Because each node in the Octree corresponds to a spatial cell, we regard the cell's attributes as the node's attributes to make the presentation concise.

When finding the nearest conductor from current point in the FRW procedure, the traversal of Octree ends at a leaf node which contains the point. The cost of this manipulation is proportional to the Octree's height. The left job is to get the minimum distance to conductor with the candidate list of the leaf node, whose cost is related with the length of candidate list. The technique of sorting the candidate list in Sect. 11.2.4 is helpful to reduce the time for inquiring the candidate list.

During the construction of the Octree, the conductors are nearly randomly inserted to the root node. So, some redundant candidate checking exists even the distance limit has been used. This redundancy can be further reduced by the approach of incomplete candidate list. In practice, we set the initial value of the distance limit to the extension size d_{nb} of neighbor region. A suitable value of d_{nb} can reduce the time of constructing the Octree with little overhead on the time of FRW procedure.

11.3.2 Two Grid-Based Approaches

The first step of inquiring the space management structure is to locate the cell containing the current point. To reduce its cost, a 3-D array representing the uniform partition of the whole domain is useful [121]. We call this the grid structure. Note the number of conductors in each cell is not uniform and so is the length of candidate list. Defining smaller cell size can reduce the maximum length of candidate list but causes a large number of cells. If a complete candidate list is generated for every grid cell, the construction time of grid will be very huge.

To reduce the construction time, a strategy different from the idea of using the candidate list can be adopted [121]. Only the intersected conductors are recorded for each cell, without candidate checking. During the random walk, the conductors in current cell and its adjacent cells are inquired to calculate the distance to conductor, similar to the approach with incomplete candidate list. Because redundant conductors are handled, the random walk will perform slower with this approach.

We can also generate the candidate list for each cell, if we only search in a neighbor region, which includes the cell's adjacent cells. For the problem with densely routed VLSI interconnects, this grid structure actually has the complete candidate list for many cells (according to Corollary 11.1). This largely reduces the construction cost, but does not degrade the efficiency of performing random walks. For the cell with too many candidates, it can be divided as a second-level grid (shown in Fig. 11.4). However, more levels of division should be avoided because it removes the advantage of grid structure for locating a point.

Algorithm 11.6 describes the procedure for constructing the two-level grid structure with candidate list.

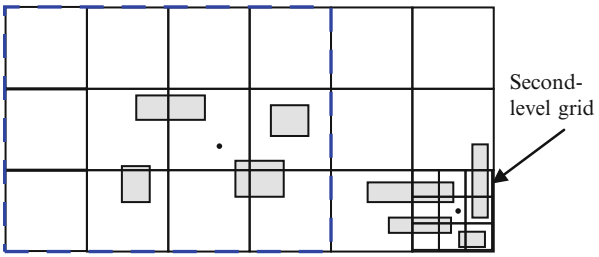

Fig. 11.4 A two-level grid structure. The neighbor cells of conductor B are outlined with the *blue dashed line* (Color figure online)

Algorithm 11.6 GenerateGrid (All Conductors, Grid G)

1. Suppose G is a 3-D array storing the cells, and the numbers of cells along x-, y-, and z-axis are n_x, n_y, and n_z, respectively;
2. Suppose the minimum x-, y-, and z-coordinates of the simulated domain are x_0, y_0, and z_0, respectively;
3. Set the distance limit of each cell in G to the size of the cell (s);
4. **For** each conductor B **do**
5. Get B's minimum and maximum coordinates: (x_{min}, y_{min}, z_{min}) and (x_{max}, y_{max}, z_{max});
6. ix1:= max($\lfloor (x_{\min} - x_0)/s \rfloor$, 1); ix2:= min($\lceil (x_{\max} - x_0)/s \rceil + 1$, n_x);
7. iy1:= max($\lfloor (y_{\min} - y_0)/s \rfloor$, 1); iy2:= min($\lceil (y_{\max} - y_0)/s \rceil + 1$, n_y);
8. iz1:= max($\lfloor (z_{\min} - z_0)/s \rfloor$, 1); iz2:= min($\lceil (z_{\max} - z_0)/s \rceil + 1$, n_z);
9. **For** i from ix1 to ix2, j from iy1 to iy2, k from iz1 to iz2 **do**
10. CandidateCheck2(B, G[i][j][k]); //Algorithm 11.2
11. **Endfor**
12. **Endfor**
13. **For** each cell T in G **do**
14. **If** T is in the top level **and** the length of T's candidate list > *threshold3* **then**
15. Divide T into a new 3-D grid G';
16. GenerateGrid(T's candidate list, G');
17. **Endif**
18. **Endfor**

11.3.3 The Hybrid Approach Using Grid and Octree

There is a drawback of the Octree-based approach, i.e., that the shape of the cell can be much different from a cube for the extraction of large-scale layout. This is because the lateral dimension of the simulated domain may be much larger than its vertical dimension. Because the maximum dimension of a cell (cell's size) is used to calculate the distance limit, the screening effect of the distance limit can be heavily weakened if the cell has large aspect ratio. In other words, the pruning effect of

the distance limit can be weakened if the cell has large aspect ratio, because with a same size of the cell, the cube-shaped cell has the largest volume. This means fewer leaf nodes for a given spatial domain. We actually have assumed cube-shaped cells in the grid-based approaches.

To overcome this drawback, we propose a hybrid approach using both grid and Octree structures. Firstly, the grid-based approach with candidate list is used to represent the whole domain, where each grid cell is a cube. Then, the Octree structure is constructed for each grid cell. This guarantees that every cell in the Octree is a cube.

The size of grid cell should be set as a large value (e.g., the height of the whole domain), so that the incomplete candidate list is almost a complete one and the expense of constructing the first-level grid is small. Algorithm 11.7 describes the construction procedure of the grid-Octree hybrid structure.

Algorithm 11.7 GenerateGridOctree

1. Suppose G is a 3-D array storing $n_x \times n_y \times n_z$ cells;
2. x_0, y_0, and z_0 denote the minimum coordinates of the whole domain;
3. s:= the size of the cell; set the distance limit of each cell in G to s;
4. **For** each conductor B **do**
5. Get B's extreme coordinates: $(x_{\min},y_{\min},z_{\min})$ and $(x_{\max},y_{\max},z_{\max})$;
6. ix1:= $\max(\lfloor(x_{\min}-x_0)/s\rfloor, 1)$; ix2:= $\min(\lceil(x_{\max}-x_0)/s\rceil+1, n_x)$;
7. iy1:= $\max(\lfloor(y_{\min}-y_0)/s\rfloor, 1)$; iy2:= $\min(\lceil(y_{\max}-y_0)/s\rceil+1, n_y)$;
8. iz1:= $\max(\lfloor(z_{\min}-z_0)/s\rfloor, 1)$; iz2:= $\min(\lceil(z_{\max}-z_0)/s\rceil+1, n_z)$;
9. **For** i from ix1 to ix2, j from iy1 to iy2, k from iz1 to iz2 **do**
10. CandidateCheck2(B, G[i][j][k]); //Algorithm 11.2
11. **Endfor**
12. **Endfor**
13. **For** each grid cell E_i **do**
14. Define an Octree root node T_i for the domain of E_i;
15. **For** each conductor $B_{i,j}$ in E_i's candidate list;
16. InsertToOctree($B_{i,j}$, T_i); //Algorithm 11.5
17. **Endfor**
18. **Endfor**

While inquiring the nearest conductor for a given point, we firstly locate a grid cell and then traverse the Octree for that cell. Compared with the improved Octree approach, this hybrid approach would consume less memory while keeping the same efficiency. As the grid with candidate list or the Octree approach, with this hybrid approach, the FRW procedure can perform very quickly. This is advantageous while comparing with the grid without candidate list.

11.4 Numerical Results

Based on the C++ program RWCap, we have implemented the proposed space management techniques. With several large VLSI layouts under the 180- and 45-nm process technologies, we firstly validate the efficiency of the accelerating techniques proposed in Sect. 11.2. Then, the space management approaches proposed in Sect. 11.3 are evaluated and compared with each other. Finally, the efficiency of the capacitance solver RWCap2, which employs the accelerating techniques and the grid-Octree hybrid structure, is demonstrated.

The experiments are carried out on a Linux server with Intel Xeon E5-2650 8-core CPU of 2.0 GHz. All results are obtained from the execution of serial computing. The accuracy criterion of FRW algorithm is set to 0.5 % $1-\sigma$ error.

11.4.1 Test Cases

We have four test cases. The first one is a $1{,}000 \times 1{,}000$ crossover structure, which includes totally 2,000 conductor wires. The width and height of each wire are both 14 nm, and the length is 28 μm. The distance between the two layers of wires is 86 nm.

The second test case is an actual design called "FreeCPU," based on the 180-nm technology with the minimum wire width of 200 nm [187]. It includes 37,062 conductor blocks in 5 metal layers, which forms 3,036 nets. The lateral and vertical dimensions of this case are about 700 μm and 9.4 μm, respectively. A portion of its layout is shown in Fig. 11.5.

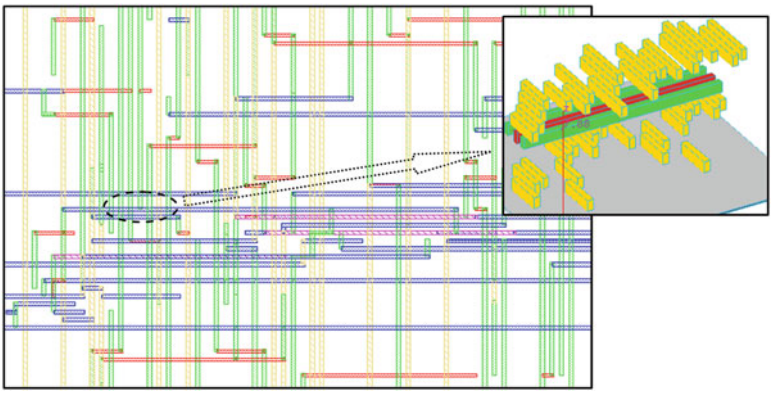

Fig. 11.5 A small part of the 2-D layout of case 2 ("FreeCPU") and its zoom-in. The wires on different layers are distinguished by different color patterns (Color figure online)

Table 11.1 Variable parameters in the space management approaches

Octree	The lower bound of the dimension of leaf node (l_t); the length threshold of the candidate list (n_t); the extension size of neighbor region (d_{nb})
Grid with candidate list	The size of the first-level grid cell (l_t); the length threshold of the candidate list for creating the second-level grid (n_t)
Grid without candidate list	The size of the first-level grid cell (l_t); the threshold of the number of conductors for creating the second-level grid (n_t)
Grid-Octree hybrid	The lower bound of the dimension of leaf node (l_t); the length threshold of the candidate list (n_t); the extension size of neighbor region (d_{nb}); the cell size of the top-level grid (l_{tt})

The third test case is an artificially created layout based on the 45-nm technology with the minimum wire width of 70 nm [187]. It includes 101,595 conductor blocks in three metal layers, with random widths (larger than the minimum width), spacings (larger than the minimum spacing), lengths, and positions. We only extract the capacitances of the metal wires in the middle layer. So, considering the three layers of wires is sufficient. The lateral size of this case is about 1,000 μm.

The fourth case is a larger case based on the 180-nm technology. Its parameters are similar to the "FreeCPU" case but with 484,441 conductor blocks.

For the evaluated spatial structures, there are several variable parameters. We list them in the following Table 11.1.

11.4.2 Validating the Three Accelerating Techniques

In this subsection, we use the Octree structure to demonstrate the efficiency of the proposed accelerating techniques. In the experiment, the length threshold of the candidate list for growing child nodes is set to 21. The lower bound of the node's dimension is set to 40 times of the minimum wire width. For the technique with incomplete candidate list, the extension size of neighbor region (d_{nb}) is set to 25 times of the minimum wire width. This makes the generated candidate list almost complete.

The total computing time of the FRW algorithm includes two parts: the time for constructing the space management structure and the time for the random walk procedure. We firstly validate the techniques with distance limit and incomplete candidate list. The original Octree-based approach in Yu et al. [187] is denoted as Octree (O), while Octree (DL) denotes the version with the distance limit. Octree (ICL) denotes the version using both distance limit and incomplete candidate list. The construction times for the three versions of Octree structure are listed in Table 11.2, where the number of Octree nodes and the minimum size of the cell (l_{min}) are also given. From the table, we see that the distance limit brings huge acceleration and the speedup is even larger for problem with more conductors.

Table 11.2 The comparison of different techniques on the construction time of the Octree structure (in unit of second)

Case	# node	l_{min} (μm)	# conductor	Octree (O)	Octree (DL)	Octree (ICL)	Sp.
1	27,145	0.41	2,000	81.3	0.36	0.29	**280**
2	83,441	7.11	37,062	1,757.9	3.10	0.74	**2,375**
3	253,761	2.34	101,595	16,595.6	8.21	2.43	**6,829**
4	1,061,361	6.44	484,441	N/A	83.12	17.12	N/A

Table 11.3 The efficiency of the fast inquiry technique on the random walk procedure with the Octree-based approach

			Time for extracting a net (s)		
Case	Average # walk	Average # hop/walk	FRW(O)	FRW(FastInq)	Speedup
1	2.82×10^5	9.1	3.04	1.61	**1.89**
2	3.03×10^5	12.5	2.97	1.41	**2.11**
3	1.89×10^5	10.5	1.73	0.81	**2.14**
4	2.77×10^5	12.7	3.63	1.49	**2.44**

The incomplete candidate list further reduces the construction time, for about 4X in this experiment. With the two techniques, the construction time for case 3 (with 101,595 conductors) is reduced by **6829X** and becomes only 2.4 s. For the largest case, the Octree (O) cannot finish the construction in a couple of days, while the improved approach costs only 17 s.

To validate the proposed techniques on the inquiry of space management structure, we randomly extract 100 nets for each case. The distance limit does not affect the inquiry time, and the incomplete distance list in this experiment affects it little. So, we only evaluate the efficiency of the fast inquiry technique in Sect. 11.2.4. The results are listed in Table 11.3, where FRW(O) denotes the original FRW algorithm and FRW(FastInq) denotes the FRW using the fast inquiry technique with sorted candidate list. The average number of walks, the average number of hops per walk, and the average time of performing the walks for extracting a net are listed. The results show that the proposed technique achieves over **2.1X** speedup for larger cases.

To investigate the effect of the extension size of neighbor region (d_{nb}) in the Octree-based approach with incomplete candidate list, we plot the curves of the construction time, the average time of million walks, the average time of million hops, and the average hop numbers of a walk in Fig. 11.6. The horizontal axis is d_{nb}/w_{min} in log scale (base 5), where w_{min} is the minimum wire width of the known process technology. The results show that if d_{nb} is larger than $25w_{min}$, there would be no difference in the walking procedure (see Fig. 11.6b, c, and d). As d_{nb} decreases, the number of hop per walk increases quickly. Although the time per hop may decrease due to shorter candidate list, the overall effect is that the time per walk increases (see Fig. 11.6b). So, we can choose $25w_{min}$ as an optimal d_{nb} to reduce the construction time without harm to the random walk procedure.

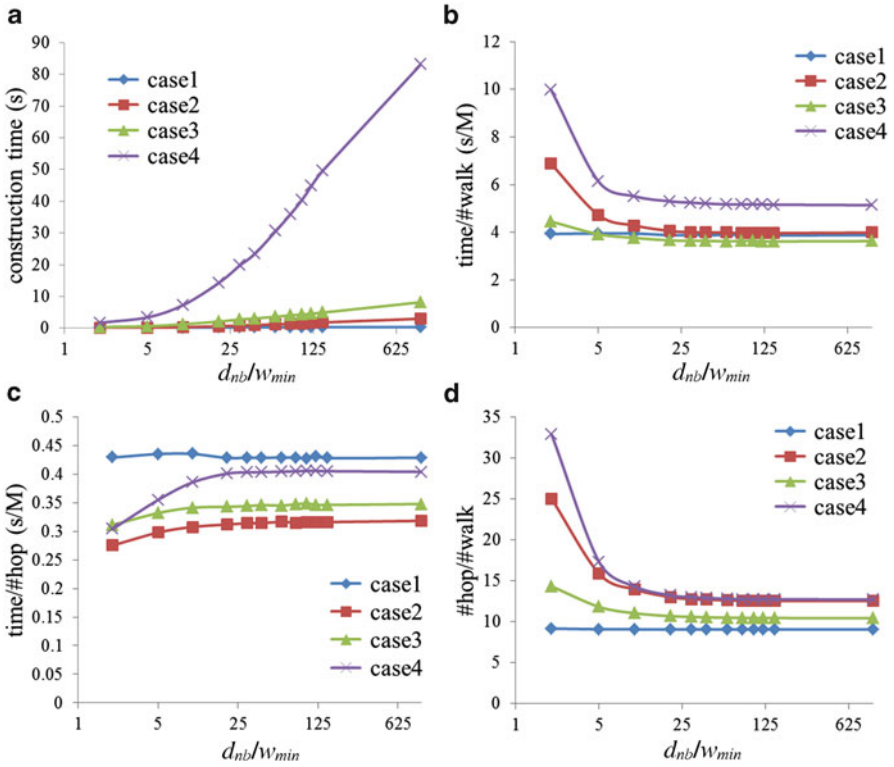

Fig. 11.6 The curves of the construction time (**a**), the average time of million walks (**b**), the average time of million hops (**c**), and the average number of hops in a walk (**d**), with varied values of d_{nb} and the Octree-based approach. w_{min} means the minimum wire width

11.4.3 Evaluating Different Space Management Approaches

In the variable parameters for the four space management approaches in Sect. 11.3 (see Table 11.1), we set the value of d_{nb} to $25w_{\text{min}}$ for the Octree and the grid-Octree hybrid structures. For the hybrid structure, we set the size of the top-level grid cell (l_{tt}) to the height of simulated domain. It makes the grid to be a 2-D grid. Now, each space management approach involves two variables. They are denoted by l_t and n_t, with slightly different meaning for different approaches. Note that d_{nb} and l_t should be measured in terms of the minimum wire width. This makes their effect changes little for cases from different process technologies.

We investigate the trends of the construction time, time of random walk procedure, and memory usage with varied values of l_t and n_t, for different space management structures. In Fig. 11.7, the construction time of the Octree and the grid with candidate list for case 2 is plotted. From the figure, we see that the time for constructing Octree decreases when l_t or n_t increases. This is because the Octree

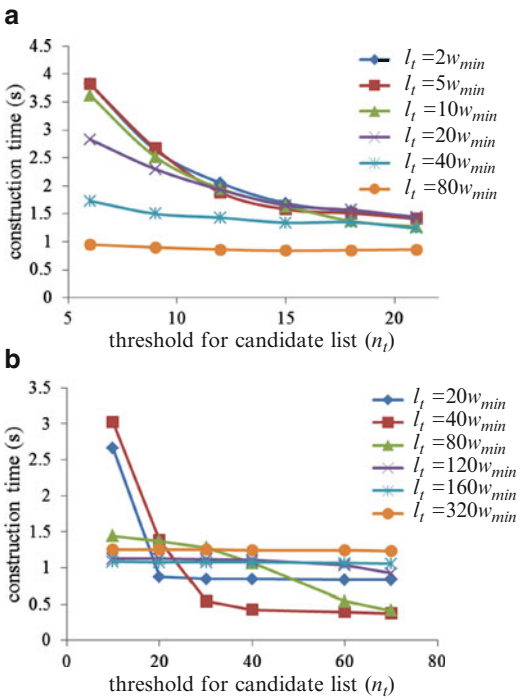

Fig. 11.7 The trends of the construction time of the Octree (**a**) and the grid with candidate list (**b**)

includes fewer cells with larger l_t or n_t. The trend for the grid structure is a little bit complex. A smaller n_t means a cell has larger probability to be divided into the next level and thus causes more construction time. If l_t is very large, each first-level cell includes many conductors and is divided into the second-level grid. So, the construction time would not change much with varied n_t, and larger l_t means larger neighbor region and thus more construction time. If l_t is small, the construction time increases quickly with decreased n_t, because more first-level cell is further divided. However, since smaller l_t corresponds to smaller neighbor region, the smallest l_t may not cause the largest construction time (see Fig. 11.7b). The trends for the grid without candidate list and the hybrid structure are similar to that in Fig. 11.7a, but their construction time is much shorter (mostly less than 1 s).

The time for performing random walk with the Octree and grid without candidate list is plotted in Fig. 11.8. From the figure, we see that with the Octree, the time of walk changes little or slightly increases when n_t increases. This is because, despite larger n_t increases the time for traversing the candidate list, it reduces the height of the tree. The sorted candidate list is also helpful for reducing the traversing time. As for the lower bound of size of leaf node (l_t), it affects only when it is very large and therefore causes a very long candidate list in leaf. While employing the grid without candidate list, the time of random walk increases with the increase of l_t or n_t. By comparing Fig. 11.8a and 11.8b, it is obvious that the grid without candidate list causes much slower random walks than the Octree structure with candidate list. The

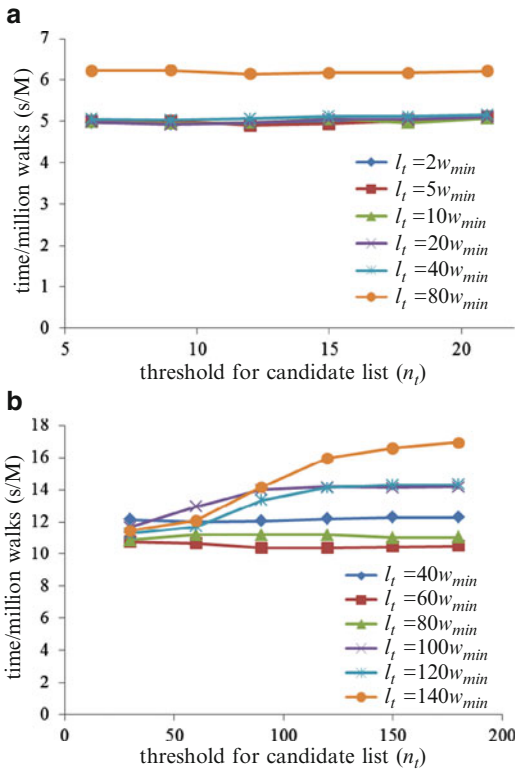

Fig. 11.8 The time for performing random walks with the Octree (**a**) and the grid without candidate list structure (**b**)

time trends of random walk with the grid with candidate list and the hybrid structure are similar to that of the Octree. And with the hybrid structure, the random walk is performed faster than that with Octree and grid with candidate list; the latter two have almost same performance.

For the four space management structures, their memory consumption varies in the same manner with changed l_t and n_t. It decreases when l_t or n_t increases, which is similar to the construction time in Fig. 11.7a. With same size of grid cells or Octree leaf nodes, the grid without candidate list consumes the least memory. For each structure, there is a trade-off between the memory usage or construction time and the efficiency brought to the random walk procedure. To reflect this trade-off, we plot the memory usage and time for performing random walk in Fig. 11.9, for the four structures. Some data are also listed in Table 11.4. From Fig. 11.9 and Table 11.4, we can see that the hybrid approach has advantages over the other three. With the same memory usage (∼20 MB), the hybrid approach makes **12 %** reduction on the time of random walk, compared with the Octree and the grid with candidate list. And it is **2.1**× faster than that with the grid without candidate list. With the same performance of random walk (∼5 s), the memory usage of the hybrid structure is *less than half* of that used by the improved Octree and grid with candidate list structures.

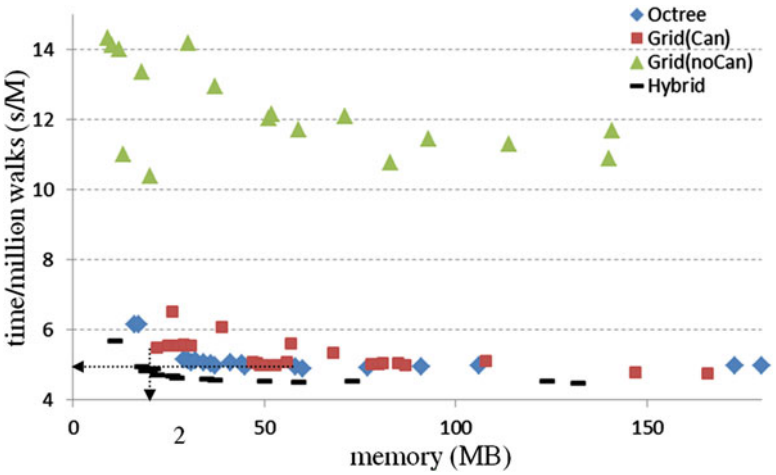

Fig. 11.9 The comparison of four space management approaches for the trade-off of memory usage and the time for performing random walks

Table 11.4 The memory usage and time of random walk for different values of l_t and n_t

Approach	l_t	n_t	Memory (MB)	Time/walk (10^{-6} s)
Improved Octree	$80w_{min}$	12	17	**6.14**
Grid with candidate list	$320w_{min}$	70	22	**5.48**
Grid without candidate list	$60w_{min}$	120	20	**10.38**
Grid-Octree hybrid	w_{min}	21	19	**4.84**
Improved Octree	$10w_{min}$	18	**37**	4.97
Grid with candidate list	$160w_{min}$	20	**50**	4.98
Grid-Octree hybrid	$16w_{min}$	21	**18**	4.96

Similar observations have been gotten with the experiments for other test cases. Note that the data of interconnect geometries cost about 6-MB memory in case 2 and the memory usage of 18 MB for space management is acceptable.

11.4.4 RWCap2 with the Hybrid Approach Using Grid and Octree

We have implemented the hybrid approach with grid and Octree in RWCap. The improved solver is called RWCap2 [125], where the values of l_t and n_t are set to $16w_{min}$ and 21, respectively. We firstly compare RWCap2 and RWCap for the test cases. The times for initialization (constructing space management structure) and tasks of extracting 1 net and 100 nets are listed in Table 11.5. The results of case 4 are not included, because RWCap costs too long time for initialization. RWCap2 costs **4.4 s** and 149-MB memory to construct the hybrid structure for case 4. From

Table 11.5 The comparison of RWCap and RWCap2

	RWCap				RWCap2					
		T_{walk} (s)				Mem[a]	T_{walk} (s)			Sp. for
Case	T_{cons} (s)	1 net	100 nets	1,000 nets	T_{cons} (s)	(MB)	1 net	100 nets	1,000 nets	1,000 nets[b]
1	81.3	3.00	304	3,185	0.13	6	1.17	119	1,178	**2.8**
2	1,758	1.80	297	2,935	0.34	18	0.80	132	1,265	**3.7**
3	16,596	1.78	173	1,729	1.37	87	0.75	71	718	**25.5**

[a]The memory usage for the space management structure
[b]With regard to the total computational time of capacitance extraction

Table 11.6 The comparison of RWCap2 and a commercial solver (capacitance in unit of 10^{-18} F, time in unit of 10^{-6} s)

	Commercial solver			RWCap2				
Structure	Cap.	Time/walk	Time	Cap.	Error (%)	Time/walk	Time	Sp.
100×100	120.3	11.05×10^{-6}	4.15	120.4	0.1	4.01×10^{-6}	1.13	**2.8**
500×500	598.9	13.44×10^{-6}	5.04	597.5	−0.2	3.89×10^{-6}	1.11	**3.5**
$1{,}000 \times 1{,}000$	1,197	15.41×10^{-6}	5.78	1,192	−0.4	4.06×10^{-6}	1.12	**3.8**

Table 11.5, we see that for the task of extracting a single net, RWCap2 is up to **7829**× faster than RWCap and for extracting 1,000 nets, the speedup is from **2.8 to 26**. Comparing the data in Table 11.5 and Tables 11.2 and 11.3, we can see the advantages of the grid-Octree hybrid structure over the Octree structure. Note the former includes fewer nodes, e.g., 35,653 for case 2.

We have also compared RWCap2 and an advance commercial FRW solver on a Linux server with AMD 2.4-GHz CPU with several crossover cases. For each one, the middle conductor in M2 layer is extracted. The capacitance results and the average time of a walk are listed in Table 11.6. From the data of time/walk, we see that the space management approach used in RWCap2 brings **3**× speedup over the other solver. For the total runtime, RWCap2 has larger speedup because its FRW procedure converges faster [187].

11.4.5 The Results for Multi-dielectric Cases

For cases 1 and 3, the multi-dielectric configuration of 45-nm technology is applied. For case 2, the 180-nm technology is applied. Thus, we get the multi-dielectric cases and perform the FRW-based capacitance extraction. The space management approaches are still valid for the multi-dielectric cases, except that when constructing the transition cube, the constraint of dielectric interfaces needs to be considered. The computational results obtained with RWCap and RWCap2 are listed in Table 11.7. Note that the multiple dielectrics do not affect the construction of space management structures, but increase the number of hops in a walk due to

Table 11.7 The comparison of RWCap and RWCap2 with test cases in multi-dielectric configurations

	RWCap			RWCap2				
		T_{walk} (s)			T_{walk} (s)		Speedup[a]	
Case	T_{cons} (s)	1 net	100 nets	T_{cons} (s)	1 net	100 nets	1 net	100 nets
1	83.1	10.1	968	0.13	4.36	444	**21**	**2.4**
2	1,758	19.6	4,030	0.34	9.71	2,080	**177**	**2.8**
3	16,596	14.7	1,699	1.37	6.41	793	**2,135**	**23**

[a]With regard to the total computational time of capacitance extraction

the constraint of dielectric interfaces. From Table 11.7, we see that the proposed techniques still bring 2× speedup to the random walk procedure, which is not degraded much as compared with the single-dielectric cases (Table 11.5). For the task of extracting a single net, RWCap2 is up to **2135**× faster than RWCap. The speedup ratio becomes **2.4** to **23** for extracting 100 nets.

11.5 Summary

Efficient techniques are proposed to largely accelerate the construction of space management structures and facilitate fast nearest conductor query in the FRW-based capacitance extraction. A new grid-Octree hybrid structure is proposed to achieve better trade-off between the costs of space management and the efficiency gain on the random walk procedure. Large single- and multi-dielectric VLSI interconnect structures have been used to validate the efficiency of proposed techniques.

In actual scenario, the distributed capacitances for a net (a number of connected interconnect wires) are needed. The environment of the nets, which constitutes chip-scale large structures, should be handled to preserve high accuracy. The detailed treatment for this scenario using the FRW-based extraction techniques can be found in Zhang and Yu [191].

References

1. Ardalan S, Sachdev M (2004) An overview of substrate noise reduction techniques. In: Proceedings of the 5th international symposium on quality electronic design (ISQED), 2004, pp 291–296
2. Asenov A, Kaya S, Brown AR (2003) Intrinsic parameter fluctuations in decananometer MOSFETs introduced by gate line edge roughness. IEEE Trans Electron Devices 50:1254–1260
3. Arora ND, Raol KV, Schumann R, Richardson LM (1996) Modeling and extraction of interconnect capacitances for multilayer VLSI circuits. IEEE Trans Comput Aided Des 15(1):58–67
4. Bachtold M, Emmenegger M, Korvink JG, Baltes H (1997) An error indicator and automatic adaptive meshing for electrostatic boundary element simulations. IEEE Trans Comput Aided Des 16:1439–1446
5. Bachtold M, Korvink JG, Baltes H (1996) Enhanced multipole acceleration technique for the solution of large Poisson computations. IEEE Trans Comput Aided Des 15(12):1541–1546
6. Bachtold M, Spasojevic M, Lage C, Ljung PB (2000) A system for full-chip and critical net parasitic extraction for ULSI interconnects using a fast 3-D field solver. IEEE Trans Comput Aided Des 19(3):325–338
7. Bansal N (1999) Randomized algorithms for capacitance estimation, Technical report. Indian Institute of Technology, Bombay, April 1999
8. Ban Y, Sundareswaran S, Pan DZ (2010) Electrical impact of line edge roughness on sub-45 nm node standard cell. J Micro/Nanolithogr MEMS MOEMS 9(1–10):041206
9. Barnes J, Hut P (1986) A hierarchical O (*N*log*N*) force-calculation algorithm. Nature 324:446–449
10. See Ref. [7]
11. Batterywala SH, Ananthakrishna R, Luo Y, Gyure A (2006) A statistical method for fast and accurate capacitance extraction in the presence of floating dummy fills. In: Proceedings of the 19th international conference on VLSI design, January 2006
12. Batterywala SH, Desai MP (2005) Variance reduction in Monte Carlo capacitance extraction. In: Proceedings of the 18th international conference on VLSI design, January 2005, pp 85–90
13. Beattie MW, Pileggi LT (1999) Error bounds for capacitance extraction via window techniques. IEEE Trans Comput Aided Des 18:311–321
14. Beattie MW, Pileggi LT (1999) Electromagnetic parasitic extraction via a multipole method with hierarchical refinement. In: Proceedings of the IEEE/ACM international conference on computer-aided design (ICCAD), November 1999, pp 437–444
15. Bebendorf M (2000) Approximation of boundary element matrices. Numerische Mathematik 86:565–589

W. Yu and X. Wang, *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*, DOI 10.1007/978-3-642-54298-5,

16. Bebendorf M, Kunis S (2009) Recompression techniques for adaptive cross approximation. J Integr Equat Appl 21(3):331–357
17. Bhoj AN, Joshi RV (2012) Transport analysis based 3D TCAD capacitance extraction for sub-32 nm SRAM structures. IEEE Electron Device Lett 33(2):158–160
18. Bhoj AN, Joshi RV, Jha NK (2013) 3-D-TCAD-based parasitic capacitance extraction for emerging multigate devices and circuits. IEEE Trans Very Large Scale Integr Syst 21(11):2094–2105
19. Bi Y, Harpe P, van der Meijs NP (2011) Efficient sensitivity-based capacitance modeling for systematic and random geometric variations. In: Proceedings of the Asia South Pacific design automation conference (ASP-DAC), January 2011, pp 61–66
20. Bi Y, van der Kolk K, Ioan D, van der Meijs NP (2008) Sensitivity computation of interconnect capacitances with respect to geometric parameters. In: Proceedings of the IEEE topical meeting on electrical performance of electronic packaging (EPEP), 2008, pp 209–212
21. Bontzios YI, Dimopoulos MG, Hatzopoulos AA (2011) An evolutionary method for efficient computation of mutual capacitance for VLSI circuits based on the method of images. Simul Model Pract Theory 19:638–648
22. Brambilla A, Maffezzoni P (2000) A statistical algorithm for 3-D capacitance extraction. IEEE Microw Guided Wave Lett 10(8):304–306
23. Brambilla A, Maffezzoni P, Bortesi L, Vendrame L (2003) Measurements and extractions of parasitic capacitances in ULSI layouts. IEEE Trans Electron Devices 50(11):2236–2247
24. Brebbia CA (1978) The boundary element method for engineers. Pentech Press, London
25. Bu S, Davies TG (1995) Effective evaluation of non-singular integrals in 3-D BEM. Adv Eng Softw 23:121–128
26. Cao Y et al. Predictive technology model. Available at: http://www.eas.asu.edu/~ptm/
27. Chai W, Jiao D (2013) A fast H-matrix based direct integral equation solver with reduced computational cost for large-scale interconnect extraction. IEEE Trans Compon Pack Manuf Technol 3(2):289–298
28. Chai W, Jiao D, Koh C-K (2009) A direct integral-equation solver of linear complexity for large-scale 3D capacitance and impedance extraction. In: Proceedings of the ACM/IEEE design automation conference (DAC), July 2009, pp 752–757
29. Chen G, Zhu H, Cui T, Chen Z, Zeng X, Cai W (2012) ParAFEMCap: a parallel adaptive finite-element method for 3-D VLSI interconnect capacitance extraction. IEEE Trans Microw Theory Tech 60(2):218–231
30. Chen Y, Kahng AB, Robins G et al (2002) Area fill synthesis for uniform layout density. IEEE Trans Comput Aided Des 21:1132–1147
31. Chou T, Cendes ZJ (1994) Capacitance calculation of IC packages using the finite element method and planes of symmetry. IEEE Trans Comput Aided Des 13(9):1159–1166
32. Choudhury U, Sangiovanni-Vicenelli A (1995) Automatic generation of analytical models for interconnect capacitances. IEEE Trans Comput Aided Des 14(4):470–480
33. Cong J, He L, Kahng AB, Noice D. Shirali N, Yen SC (1997) Analysis and justification of a simple, practical 2 1/2-D capacitance extraction methodology. In: Proceedings of the 34th design automation conference (DAC), July 1997, pp 627–632
34. Costa JP, Chou M, Silveira LM (1999) Efficient techniques for accurate modeling and simulation of substrate coupling in mixed-signal IC's. IEEE Trans Comput Aided Des 18(5):597–607
35. Cueto O, Charlet F, Farcy A (2002) An efficient algorithm for 3D interconnect capacitance extraction considering floating conductors. In: Proceedings of the international conference on simulation of semiconductor processes and devices (SISPAD), September 2002, pp 107–110
36. Dengi EA (1997) A parasitic capacitance extraction method for VLSI interconnect modeling. PhD thesis, Carnegie Mellon University, March 1997
37. Dengi EA, Rohrer RA (1998) Boundary element method macromodels for 2-D hierarchical capacitance extraction. In: Proceedings of the ACM/IEEE design automation conference (DAC), 1998, pp 218–223

38. Desai MP. The capacitance extraction tool. Available at: http://www.ee.iitb.ac.in/~microel/download
39. Deschacht D, de Rivaz S Farcy A, Lacrevaz T, Flechet B (2010) Keep on shrinking interconnect size: is it still the best solution? In: Proceedings of the international symposium on electronic manufacturing technology (IEMT), November 2010, pp 1–4
40. Donnay S, Gielen G (2003) Substrate noise coupling in mixed-signal ASICs. Kluwer, Boston
41. El-Moselhy T, Daniel L (2008) Stochastic integral equation solver for efficient variation-aware interconnect extraction. In: Proceedings of the ACM/IEEE design automation conference (DAC), June 2008, pp 415–420
42. El-Moselhy T, Daniel L (2010) Stochastic dominant singular vectors method for variation-aware extraction. In: Proceedings of the ACM/IEEE design automation conference (DAC), June 2010, pp 667–672
43. El-Moselhy TA, Elfadel IM, Daniel L (2008) A capacitance solver for incremental variation-aware extraction. In: Proceedings of the IEEE/ACM international conference on computer-aided design (ICCAD), November 2008, pp 662–669
44. El-Moselhy TA, Elfadel IM, Daniel L (2009) A hierarchical floating random walk algorithm for fabric-aware 3D capacitance extraction. In: Proceedings of the IEEE/ACM international conference on computer-aided design (ICCAD), November 2009, pp 752–758
45. El-Moselhy T, Elfadel IM, Daniel L (2010) A Markov chain based hierarchical algorithm for fabric-aware capacitance extraction. IEEE Trans Adv Pack 33(4):818–827
46. Fukuda S, Shigyo N, Kato K, Nakamura S (1990) A ULSI 2-D capacitance simulator for complex structures based on actual processes. IEEE Trans Comput Aided Des 9:39–47
47. Gao W, Yu Z (2006) Scalable compact circuit model and synthesis for RF CMOS spiral inductors. IEEE Trans Microw Theory Tech 54:1055–1064
48. Ghanem RG, Spanos PD (1991) Stochastic finite elements: a spectral approach. Springer, New York
49. Gharpurey R (1995) Modeling and analysis of substrate coupling in integrated circuits. PhD dissertation, University of California at Berkeley
50. Gharpurey R, Charbon E (2004) Substrate coupling: modeling, simulation and design perspectives. In: Proceedings of the 5th international symposium on quality electronic design (ISQED), 2004, pp 283–290
51. Ghapurey R, Meyer RG (1995) Analysis and simulation of substrate coupling in integrated circuits. Int J Circuit Theory Appl 23(4):381–394
52. Glegharbour M, Drake JM (1986) Calculation of multiterminal resistance in integrated circuits. IEEE Trans Circuits Syst 33(4):462–465
53. Golub GH, Van Loan CF (1996) Matrix computations, 3rd edn. Johns Hopkins University Press, Baltimore
54. Gong F, Yu H, He L (2009) Picap: a parallel and incremental capacitance extraction considering stochastic process variation. In: Proceedings of the 46th design automation conference (DAC), July 2009, pp 764–769
55. Gong W, Yu W, Lu Y, Tang Q, Zhou Q, Cai Y (2010) A parasitic extraction method of VLSI interconnects for pre-route timing analysis. In: Proceedings of the international conference on communications, circuits and systems (ICCCAS), Chengdu, China, July 2010, pp 871–875
56. Gope D, Jandhyala V (2004) PILOT: a fast algorithm for enhanced 3D parasitic capacitance extraction efficiency. Microw Opt Technol Lett 41(3):169–173
57. Greengard L, Rokhlin V (1987) A fast algorithm for particle simulations. J Comput Phys 73:325–348
58. Gu J, Wang Z, Hong X (2000) Hierarchical computation of 3-D interconnect capacitance using direct boundary element method. In: Proceedings of the Asia South Pacific design automation conference (ASP-DAC), 2000, pp 447–452
59. Gu J, Wang Z, Hong X (2000) A fast boundary element mesh generation approach for multi-hole surface. J Comput Aided Des Comput Graph 12:211–215 (in Chinese)
60. Harrington RF (1968) Field computation by moment methods. MacMillan, New York

61. Hong W, Sun WK, Zhu ZH et al (1998) A novel dimension-reduction technique for the capacitance extraction of 3-D VLSI interconnects. IEEE Trans Microw Theory Tech 46: 1037–1043
62. Horowitz M, Dutton RW (1983) Resistance extraction from mask layout data. IEEE Trans Comput Aided Des 2(3):145–150
63. Hou J, Wang Z, Hong X (1999) The hierarchical h-adaptive 3D boundary element computation of VLSI interconnect capacitance. In: Proceedings of the Asia South Pacific design automation conference (ASP-DAC), Hong Kong, January 1999, pp 93–96
64. Hu G, Yu W, Zhuang H, Zeng S (2011) Efficient floating random walk algorithm for interconnect capacitance extraction considering multiple dielectrics. In: Proceedings of the IEEE international conference on ASIC (ASICON), October 2011, pp 896–899
65. Huang J-F, Chang VCY, Liu S, Doong KYY, Chang K-J (2007) Modeling sub-90nm on-chip variation using Monte Carlo method for DFM. In: Proceedings of the Asia South Pacific design automation conference (ASP-DAC), January 2007, pp 221–225
66. Huang Q, Cruse TA (1993) Some notes on singular integral techniques in boundary element analysis. Inter J Numer Method Eng 36:2643–2659
67. Husain A (2001) Models for interconnect capacitance extraction. In: Proceedings of the international symposium on quality electronic design (ISQED), San Jose, CA, April 2001, pp 167–172
68. The International Technology Roadmap for Semiconductors (ITRS) (2010) Update edition. http://www.itrs.net
69. Iverson RB, Le Coz Y (2001) A floating random-walk algorithm for extracting electrical capacitance. Math Comput Simul 55:59–66
70. Jackson JD (1975) Classical electrodynamics. Wiley, New York
71. Jiang R, Fu W, Wang J, Lin V, Chen C (2005) Efficient statistical capacitance variability modeling with orthogonal principle factor analysis. In: Proceedings of the IEEE/ACM international conference on computer-aided design (ICCAD), 2005, pp 683–690
72. Jiang LJ, Rubin BJ, Morsey JD, Hu HT, Elfadel A (2006) Novel capacitance extraction method using direct boundary integral equation method and hierarchical approach. In: Proceedings of the IEEE topical meeting on electrical performance of electronic packaging (EPEP), October 2006, pp 331–334
73. Jin R, Cao Y, Li Z (1997) Fast parameter extraction for multiconductor interconnects in multilayered dielectric media using mixture method of equivalent source and measured equation of invariance. IEEE Trans Compon Pack Manuf Technol B Adv Pack 20(3): 235–240
74. Kahng AB, Robins G, Singh A et al (1999) Filling algorithm and analyses for layout density control. IEEE Trans Comput Aided Des 18:445–462
75. Kamon M, Iverson R (2006) High-accuracy parasitic extraction. In: Lavagno L, Scheffer L, Martin G (eds) EDA for IC implementation, circuit design, and process technology. CRC Press/Taylor & Francis Group, Boca Raton
76. Kane JH (1994) Boundary element analysis in engineering continuum mechanics. Prentice Hall, Englewood Cliffs
77. Kapur S, Long D (1997) IES3: a fast integral equation solver for efficient 3-dimensional extraction. In: Proceedings of the IEEE/ACM international conference on computer-aided design (ICCAD), November 1997, pp 448–455
78. Kapur S, Long D (1998) Efficient electromagnetic and electrostatic simulation using IES3. IEEE J Comput Sci Eng 5. Also available at: http://www.integrandsoftware.com/oldpapers/ieee.pdf
79. Kao WH, Lo C, Basel M, Singh R (2001) Parasitic extraction: current state of the art and future trends. Proc IEEE 89(5):729–739
80. Kurz S, Rain O, Rjasanow S (2002) The adaptive cross-approximation technique for the 3D boundary-element method. IEEE Trans Magn 38(2):421–424
81. Labun A (2004) Rapid method to account for process variation in full-chip capacitance extraction. IEEE Trans Comput Aided Design 23(6):941–951

82. Le Coz Y, Greub HJ, Iverson RB (1998) Performance of random walk capacitance extractors for IC interconnects: a numerical study. Solid State Electron 42(4):581–588
83. Le Coz Y, Iverson RB (1992) A stochastic algorithm for high speed capacitance extraction in integrated circuits. Solid State Electron 35(7):1005–1012
84. Lee W-S, Lee K-H, Park J-K et al (2003) Investigation of the capacitance deviation due to metal-fills and the effective interconnect geometry modeling. In: Proceedings of the IEEE international symposium on quality electronic design (ISQED), March 2003, pp 373–376
85. Leung CY, Walker SP (1997) Iterative solution of large three-dimensional BEM elastostatic analysis using the GMRES technique. Int J Numer Method Eng 40:2227–2236
86. Li K, Atsuki K, Hasegawa T (1997) General analytical solution of static green's functions for shielded and open arbitrarily multilayered media. IEEE Trans Microw Theory Tech 45(1):2–8
87. Liu Y, Nassif SR, Pileggi LT et al (2000) Impact of interconnect variations on the clock skew of a gigahertz microprocessor. In: Proceedings of the 37th design automation conference (DAC), Los Angeles, CA, 2000, pp 168–171
88. Liu YW, Lan K, Mei KK (1999) Computation of capacitance matrix for integrated circuit interconnects using on-surface MEI method. IEEE Microw Guided Wave Lett 9(8):303–304
89. Lu T, Wang Z, Yu W (2004) Hierarchical block boundary-element method (HBBEM): a fast field solver for 3-D capacitance extraction. IEEE Trans Microw Theory Tech 52(1):10–19
90. Malavasi E, Zanella S, Cao M (2002) Impact analysis of process variability on clock skew. In: Proceedings of the international symposium on quality electronic design (ISQED), San Jose, CA, April 2002, pp 129–132
91. Masoumi N, Elmasry MI, Safavi-Naeini S (2000) Fast and efficient parametric modeling of contact-to-substrate coupling. IEEE Trans Comput Aided Des 19(11):1282–1292
92. Masoumi N, Elmasry MI, Safavi-Naeini S, Hadi H (2002) A novel analytical model for evaluation of substrate crosstalk in VLSI circuits. In: Proceedings of the first IEEE international workshop on electronic design, test and application, 2002, pp 355–359
93. MATLAB. http://www.mathworks.com
94. Matsumoto M, Nishimura T (1998) Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Trans Model Comput Simul 8(1):3–30
95. Mitev A, Marefat M, Ma D, Wang JM (2007) Principle Hessian direction based parameter reduction with process variation. In: Proceedings of the IEEE/ACM international conference on computer-aided design (ICCAD), November 2007, pp 632–637
96. Mitsuhashi T, Yoshida K (1987) A resistance calculation algorithm and its application to circuit extraction. IEEE Trans Comput Aided Des 6(3):337–345
97. Nabors K, Kim S, White J, Senturia S (1992) FastCap user's guide. Massachusetts Institute of Technology, Cambridge. http://www.rle.mit.edu/cpg/
98. Nabors K, Kim S, White J (1992) Fast capacitance extraction of general three-dimensional structures. IEEE Trans Microw Theory Tech 40(7):1496–1506
99. Nabors K, White J (1991) FastCap: a multipole accelerated 3-D capacitance extraction program. IEEE Trans Comput Aided Des 10:1447–1459
100. Nabors K, White J (1992) Multipole-accelerated capacitance extraction algorithms for 3-D structures with multiple dielectrics. IEEE Trans Circuits Syst I 39:946–954
101. Niknejad, AM ASITIC. http://rfic.eecs.berkeley.edu/~niknejad/
102. Niknejad AM, Gharpurey R, Meyer RG (1998) Numerically stable Green function for modeling and analysis of substrate coupling in integrated circuits. IEEE Trans Comput Aided Des 17(4):305–315
103. Niknejad AM, Meyer RG (1998) Analysis, design, and optimization of spiral inductors and transformers for Si RF IC's. IEEE J Solid-State Circuits 33(10):1470–1481
104. Novak E, Ritter K (1999) Simple cubature formulas with high polynomial exactness. Constr Approx 15(4):449–522
105. Oh KS, Kuznetsov D, Schutt-Aine JE (1994) Capacitance computations in a multilayered dielectric medium using closed-form spatial green's functions. IEEE Trans Microw Theory Tech 42(8):1443–1453

106. Pan YC, Chew WC, Wan LX (2001) A fast multipole-method-based calculation of the capacitance matrix for multiple conductors above stratified dielectric media. IEEE Trans Microw Theory Tech 49(3):480–490
107. Pant S, Chiprout E (2006) Power grid physics and implications for CAD. In: Proceedings of the 43rd design automation conference (DAC), 2006, pp 199–204
108. Papadopoulos A, Manolopoulos Y (1997) Performance of nearest neighbor queries in R-trees. In: Proceedings of the international conference on database theory, 1997, pp 394–408
109. Park J-K, Lee K.H, Lee J-H et al (2000) An exhaustive method for characterizing the interconnect capacitance considering the floating dummy-fills by employing an efficient field solving algorithm. In: Proceedings of the international conference on simulation of semiconductor processes and devices (SISPAD), September 2000, pp 98–103
110. Pham HH, Nathan A (1999) An integral equation of the second kind for computation of capacitance. IEEE Trans Comput Aided Des 18(10):1435–1441
111. Phillips JR, White JK (1997) A precorrected-FFT method for electrostatic analysis of complicated 3-D structures. IEEE Trans Comput Aided Des 16(10):1059–1072
112. Poliakov P, Blomme P, Corbalan MM, Van Houdt J, Dehaene W (2011) Cross-cell interference variability aware model of fully planar NAND Flash memory including line edge roughness. Microelectron Reliab 51:919–924
113. Press WH, Teukolsky SA, Veterling WT, Flannery BP (1992) Numerical recipes in C: the art of scientific computing. Cambridge University Press, New York
114. QBEM for 3-D interconnect capacitance extraction. http://learn.tsinghua.edu.cn:8080/2003990088/qbem.htm
115. Qi X, Dutton RW (2003) Interconnect parasitic extraction of resistance, capacitance, and inductance. In: Davis J, Meindl JD (eds) Interconnect technology and design for gigascale integration. Kluwer, Boston
116. Qi X, Gyure A, Luo Y, Lo S, Shahram M, Singhal K (2006) Measurement and characterization of pattern dependent process variations of interconnect resistance, capacitance and inductance in nanometer technologies. In: Proceedings of the great lakes symposium on VLSI (GLSVLSI), 2006, pp 14–18
117. Qu H, Kong L, Xu Y, Xu X, Ren Z (2008) Finite-element computation of sensitivities of interconnect parasitic capacitances to the process variation in VLSI. IEEE Trans Magn 44(6):1386–1389
118. Ren Z, Lage C (2004) 3-D capacitance extraction of IC interconnects using field solvers and homogenization technique. IEEE Trans Magn 40(2):703–706
119. Ren Z, Qu H, Xu X (2012) Computation of second order capacitance sensitivity using adjoint method in finite element modeling. IEEE Trans Magn 48(2):231–234
120. Rjasanow S, Steinbach O (2007) The fast solution of boundary integral equations. Springer, New York
121. Rollins G (2010) Rapid3D 20X performance improvement. Online presentation, July 2010. http://www.synopsys.com/Community/UniversityProgram/Pages/Presentations.aspx
122. Roussopoulos N, Kelley S, Vincent F (1995) Nearest neighbor queries. In: Proceedings of the ACM international conference on management of data (SIGMOD), 1995, pp 71–79
123. Royer GM (1971) A Monte Carlo procedure for potential theory problems. IEEE Trans Microw Theory Tech 19(10):813–818
124. Ruehli AE, Brennan PA (1973) Efficient capacitance calculations for three-dimensional multiconductor systems. IEEE Trans Microw Theory Tech MTT-21(2):76–82
125. RWCap: a 3-D floating random walk based capacitance solver. http://learn.tsinghua.edu.cn:8080/2003990088/rwcap.htm
126. Saad Y, Schultz MH (1986) GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J Sci Stat Comput 7:856–869
127. Samet H (1990) Applications of spatial data structure. Addison-Wesley, Reading
128. Schrik E, van der Meijs NR (2002) Combined BEM/FEM substrate resistance modeling. In: Proceedings of the ACM/IEEE design automation conference (DAC), 2002, pp 771–776

129. Seidl A, Svoboda M et al (1988) CAPCAL-A 3-D capacitance solver for support of CAD systems. IEEE Trans Comput Aided Des 7(5):549–556
130. Shen R, Tan SX-D, Cui J, Yu W, Cai Y, Chen G (2010) Variational capacitance extraction and modeling based on orthogonal polynomial method. IEEE Trans Very Large Scale Integr Syst 18(11):1556–1566
131. Shen R, Tan SX-D, Yu H (2012) Statistical performance analysis and modeling techniques for nanometer VLSI designs. Springer, New York
132. Shi W, Liu J, Kakani N, Yu T (1998) A fast hierarchical algorithm for 3-D capacitance extraction. In: Proceedings of the 35th design automation conference (DAC), San Francisco, CA, June 1998, pp 211–217
133. Shi W, Liu J, Kakani N, Yu T (2002) A fast hierarchical algorithm for three-dimensional capacitance extraction. IEEE Trans Comput Aided Des 21(3):330–336
134. Shi W, Yu F (2004) A divide-and-conquer algorithm for 3-D capacitance extraction. IEEE Trans Comput Aided Des 23(8):1157–1163
135. Smedes T, van der Meijs NP, van Genderen AJ (1995) Extraction of circuit models for substrate cross-talks. In: Proceedings of the IEEE/ACM international conference on computer-aided design (ICCAD), 1995, pp 199–206
136. Soveiko N, Nakhla MS (2000) Efficient capacitance extraction computations in wavelet domain. IEEE Trans Circuits Syst I Fundam Theory Appl 47(5):684–701
137. Specogna R (2014) Extraction of VLSI multiconductor transmission line parameters by complementarity. IEEE Trans Very Large Scale Integr Syst 22(1):146–154
138. Stine BE, Boning DS, Chung JE et al (1998) The physical and electrical effects of metal-fill patterning practices for oxide chemical-mechanical polishing processes. IEEE Trans Electron Devices 45:665–679
139. Stine BS, Ouma DO, Divecha RR et al (1998) Rapid characterization and modeling of pattern dependent variation in chemical-mechanical polishing. IEEE Trans Semicond Manuf 11: 129–140
140. Stucchi M, Bamal M, Maex K (2007) Impact of line-edge roughness on resistance and capacitance of scaled interconnects. Microelectron Eng 84:2733–2737
141. Su DK, Loinaz MJ, Masui S, Wooley BA (1993) Experimental results and modeling techniques for substrate noise in mixed-signal integrated circuits. IEEE J Solid State Circuits 28(4):420–430
142. Sun W, Dai WWM, Hong W (1997) Fast parameter extraction of general interconnects using geometry independent measured equation of invariance. IEEE Trans Microw Theory Tech 45:827–836
143. Sun W, Hong W, Dai WW-M (1997) Resistance extraction using superconvergence accelerated boundary element method. Proc Asia Pac Microw Conf 3:1061–1064
144. Synopsys, Inc. (2003) Raphael reference manual, San Jose, CA, September 2003
145. Synopsys, Inc. Taurus Medici. http://www.synopsys.com/Tools/TCAD/DeviceSimulation/Pages/TaurusMedici.aspx
146. Tausch J, White J (1999) A multiscale method for fast capacitance extraction. In: Proceedings of the ACM/IEEE design automation conference (DAC), 1999, pp 537–542
147. Tsuboi K, Chow K, Nariki Y (2012) Analyzing the device parasitics sensitivity and accuracy of Calibre xACT 3D field solver extraction. Paper presented at the 49th design automation conference (DAC), June 2012
148. Twaddle FJ, Cumming DRS, Roy S, Asenov A, Drysdale TD (2009) RC variability of short-range interconnects. In: Proceedings of the international workshop on computational electronics (IWCE), 2009, pp 1–4
149. van der Meijs NP, van Genderen AJ (1989) An efficient finite element method for submicron IC capacitance extraction. In: Proceedings of the ACM/IEEE design automation conference (DAC), June 1989, pp 678–681
150. Vavasis SA (1992) Preconditioning for boundary integral equations. SIAM J Matrix Anal Appl 13(3):905–925

151. Veremey VV, Mittra R (1999) Efficient computation of interconnect capacitances using the domain decomposition approach. IEEE Trans Adv Pack 22(3):348–355
152. Veremey VV, Mittra R (2000) Domain decomposition approach for capacitance computation of nonorthogonal interconnect structures. IEEE Trans Microw Theory Tech 48(9):1428–1434
153. Verghese NK, Allstot DJ (1993) Rapid simulation of substrate coupling effects in mixed-mode ICs. In: Proceedings of the IEEE custom integrated circuits conference (CICC), 1993, pp 18.3.1–18.3.4
154. Verma KG, Kaushik BK, Singh R (2009) Effects of process variation in VLSI interconnects - a technical review. Microelectron Int 26(3):49–55
155. Walton AJ, Holwill RJ, Robertson JM (1985) Numerical simulation of resistive interconnects for integrated circuits. IEEE J Solid State Circuits 20:1252–1258
156. Wang HG, Chan CH, Tsang L, Jandhyala V (2006) On sampling algorithms in multilevel QR factorization method for magnetoquasistatic analysis of integrated circuits over multilayered lossy substrates. IEEE Trans Comput Aided Des 25(9):1777–1792
157. Wang X, Jandhyala V (2007) Parallel algorithms for fast integral equation based solvers. In: Proceedings of the IEEE topical meeting on electrical performance of electronic packaging (EPEP), 2007, pp 249–252
158. Wang X, Jandhyala V (2008) Enhanced hybrid MPI-OpenMP parallel electromagnetic simulations based on low-rank compressions. In: Proceedings of the IEEE international symposium on electromagnetic compatibility, 2008, pp 1–5
159. Wang X, Liu D, Yu W, Wang Z (2005) Improved boundary element method for fast 3-D interconnect resistance extraction. IEICE Trans Electron e88-c(2):232–240
160. Wang X, Yu W, Wang Z (2006) A new boundary element method for accurate modeling of lossy substrates with arbitrary doping profiles. In: Proceedings of the Asia South Pacific design automation conference (ASP-DAC), Yokohama, Japan, January 2006, pp 683–688
161. Wang X, Yu W, Wang Z (2006) Analytical-BEM coupling method for fast 3-D interconnect resistance extraction. Front Elec Electron Eng China 1(2):239–243
162. Wang X, Yu W, Wang Z (2006) Efficient direct boundary element method for resistance extraction of substrate with arbitrary doping profiles. IEEE Trans Comput Aided Des 25:3035–3042
163. Wang Z, Wu Q (1992) A two-dimensional resistance simulator using the boundary element method. IEEE Trans Comput Aided Des 11:497–504
164. Wang Z, Yuan Y, Wu Q (1996) A parallel multipole accelerated 3-D capacitance simulator based on an improved model. IEEE Trans Comput Aided Des 15:1441–1450
165. Wei C, Harington RF, Maultz JR et al (1984) Multiconductor transmission lines in multilayered dielectric media. IEEE Trans Microw Theory Tech 32:439–450
166. Xiong J, Zolotov V, He L (2007) Robust extraction of spatial correlation. IEEE Trans Comput Aided Des 26:619–631
167. Xiu D, Karniadakis GE (2002) The Wiener-Askey polynomial chaos for stochastic differential equations. SIAM JSci Comput 24(2):619–644
168. Xu C, Fiez T, Mayaram K (2005) On the numerical stability of greens function for substrate coupling in integrated circuits. IEEE Trans Comput Aided Des 24(4):653–658
169. Xu J, Li H, Yin W-Y, Mao J, Li L-W (2007) Capacitance extraction of three-dimensional interconnects using element-by-element finite element method (EBE-FEM) and preconditioned conjugate gradient (PCG) technique. IEICE Trans Electron E90-C(1):179–188
170. Yan S, Sarin V, Shi W (2005) Sparse transformations and preconditioners for 3-D capacitance extraction. IEEE Trans Comput Aided Des 24(9):1420–1426
171. Yan S, Sarin V, Shi W (2006) Fast 3-D capacitance extraction by inexact factorization and reduction. IEEE Trans Comput Aided Des 25(10):2282–2286
172. Ye Z, Yu W, Yu Z (2006) Efficient 3D capacitance extraction considering lossy substrate with multi-layered green's function. IEEE Trans Microw Theory Tech 54(5):2128–2137
173. Ye Z, Zhu Z, Phillips JR (2008) Generalized Krylov recycling methods for solution of multiple related linear equation systems in electromagnetic analysis. In: Proceedings of the 45th design automation conference (DAC), June 2008, pp 682–687

174. Yu W, Hu C, Zhang W (2009) Variational capacitance extraction of on-chip interconnects based on continuous surface model. In: Proceedings of the ACM/IEEE design automation conference (DAC), July 2009, pp 758–763
175. Yu W, Hu C, Zhang W (2011) Parallel statistical capacitance extraction of on-chip interconnects with an improved geometric variation model. In: Proceedings of the Asia South Pacific design automation conference (ASP-DAC), Yokohama, Japan, January 2011, pp 67–72
176. Yu W, Li L, Wang Z, Hong X (2004) Improved 3-D hierarchical interconnect capacitance extraction for the analog integrated circuit. In: Proceedings of the international conference on communications, circuits and systems (ICCCAS), June 2004, pp 1305–1309
177. Yu W, Wang X, Ye Z, Wang Z (2008) Efficient extraction of frequency-dependent substrate parasitics using direct boundary element method. IEEE Trans Comput Aided Des 27(8): 1508–1513
178. Yu W, Wang Z (2003) A fast quasi-multiple medium method for 3-D BEM calculation of parasitic capacitance. Comput Math Appl 45(12):1883–1894
179. Yu W, Wang Z (2004) Enhanced QMM-BEM solver for three-dimensional multiple-dielectric capacitance extraction within the finite domain. IEEE Trans Microw Theory Tech 52:560–566
180. Yu W, Wang Z (2005) Capacitance extraction. In: Chang K (ed) Encyclopedia of RF and microwave engineering. Wiley, Hoboken, pp 565–576
181. Yu W, Wang Z, Gu J (2003) Fast capacitance extraction of actual 3-D VLSI interconnects using quasi-multiple medium accelerated BEM. IEEE Trans Microw Theory Tech 51(1):109–119
182. Yu W, Wang Z, Hong X (2004) Preconditioned multi-zone boundary element analysis for fast 3D electric simulation. Eng Anal Bound Elem 28:1035–1044
183. Yu W, Wang Z, Wang Y, Lu T, Chen R (2004) An efficient boundary generation method for arbitrary complex structures in parasitic capacitance extraction. Chin J Semicond 25(3): 214–220
184. Yu W, Zhang L, Wang X, Wang Z (2006) An incremental boundary element method for the variation-aware library-building procedure of capacitance extraction. In: Proceedings of the 8th international conference on solid-state and integrated circuit technology (ICSICT), October 2006, pp 1435–1437
185. Yu W, Zhang M, Wang Z (2006) Efficient 3-D extraction of interconnect capacitance considering floating metal-fills with boundary element method. IEEE Trans Comput Aided Des 25(1):12–18
186. Yu W, Zhang Q, Ye Z, Luo Z (2012) Efficient statistical capacitance extraction of nanometer interconnects considering the on-chip line edge roughness. Microelectron Reliab 52(4): 704–710
187. Yu W, Zhuang H, Zhang C, Hu G, Liu Z (2013) RWCap: a floating random walk solver for 3-D capacitance extraction of VLSI interconnects. IEEE Trans Comput Aided Des 32(3):353–366
188. Zhai K, Yu W (2013) The 2-D boundary element techniques for capacitance extraction of nanometer VLSI interconnects. Int J Numer Model Electron Netw Devices Fields. doi: 10.1002/jnm.1934
189. Zhai K, Yu W, Zhuang H (2013) GPU-friendly floating random walk algorithm for capacitance extraction of VLSI interconnects. In: Proceedings of the design, automation & test in Europe conference (DATE), Grenoble, France, March 2013, pp 1661–1666
190. Zhang C, Yu W (2013) Efficient space management techniques for large-scale interconnect capacitance extraction with floating random walks. IEEE Trans Comput Aided Des 32(10):1633–1637
191. Zhang C, Yu W (2014) Efficient techniques for the capacitance extraction of chip-scale VLSI interconnects using floating random walk algorithm. In: Proceedings of the Asia South Pacific design automation conference (ASP-DAC), Singapore, January 2014
192. Zhang W, Yu W, Wang Z, Yu Z, Jiang R, Xiong J (2008) An efficient method for chip-level statistical capacitance extraction considering process variations with spatial correlation. In: Proceedings of the design, automation & test in Europe conference (DATE), March 2008, pp 580–585

193. Zhao J, Dai WWM et al. (1998) Efficient three-dimensional extraction based on static and full-wave layered green's functions. In: Proceedings of the ACM/IEEE design automation conference (DAC), 1998, pp 224–229
194. Zhao X, Feng Z, (2011) Fast multipole method on GPU: tackling 3-D capacitance extraction on massively parallel SIMD platforms. In: Proceedings of the ACM/IEEE design automation conference (DAC), June 2011, pp 558–563
195. Zheng J, Li Z (1997) Accelerating capacitance computation for interconnects in high speed MCM by Pade approximation. Electron Lett 33(3):217–218
196. Zhou Y, Li Z, Shi W (2007) Fast capacitance extraction in multilayer, conformal and embedded dielectric using hybrid boundary element method. In: Proceedings of the ACM/IEEE design automation conference (DAC), 2007, pp 835–840
197. Zhou Y, Li Z, Tian Y, Shi W, Liu F (2007) A new methodology for interconnect parasitics extraction considering photo-lithography effects. In: Proceedings of the Asia South Pacific design automation conference (ASP-DAC), Yokohama, Japan, January 2007, pp 450–455
198. Zhu Z, Demir A, White J (2004) A stochastic integral equation method for modeling the rough surface effect on interconnect capacitance. In: Proceedings of the IEEE/ACM international conference on computer-aided design (ICCAD), 2004, pp 887–891
199. Zhu Z, Hong W (1999) A generalized algorithm for the capacitance extraction of 3-D VLSI interconnects. IEEE Trans Microw Theory Tech 47:2027–2031
200. Zhu Z, Ji H, Hong W (1997) An efficient algorithm for the parameter extraction of 3-D interconnect structures in the VLSI circuits: domain decomposition method. IEEE Trans Microw Theory Tech 45:1179–1184
201. Zhu Z, White J (2005) FastSies: a fast stochastic integral equation solver for modeling the rough surface effect. In: Proceedings of the IEEE/ACM international conference on computer-aided design (ICCAD), 2005, pp 675–682
202. Zhu H, Zeng X, Cai W, Xue J, Zhou D (2007) A sparse grid based spectral stochastic collocation method for variations-aware capacitance extraction of interconnects under nanometer process technology. In: Proceedings of the design, automation & test in Europe conference (DATE), 2007, pp 1514–1519
203. Zhuang H, Yu W, Hu G, Liu Z, Ye Z (2012) Fast floating random walk algorithm for multi-dielectric capacitance extraction with numerical characterization of green's functions. In: Proceedings of the Asia South Pacific design automation conference (ASP-DAC), January 2012, pp 377–382

Index

A
Absolute roughness amplitude, 168
Adaptive cross approximation (ACA), 19, 34–37
Adjoint field technique (AFT), 169–170
Analytical integration, 42
Analytical QBEM, 83–85
Analytical resistance formula, 72
Area fill, 63
ASITIC, 116–118
Aspect ratio, 221, 222

B
Blocked Gauss method, 114, 115, 118
Boundary element method (BEM), 5, 6
Boundary element partition, 40, 46
Boundary integral equation method, 10, 11

C
Candidate checking, 211–215, 220
Candidate list, 199
Capacitance, 7–18
Capacitance covariance, 124, 137–139, 142, 145
Capacitance matrix, 8, 16
Capacitance sensitivity, 167–170, 172, 177
CAPEM, 204, 208
Central limit theorem, 190
Chemical-mechanical polishing (CMP), 61, 63
Chip-level capacitance extraction, 121, 123, 133, 144, 148
Compatibility equation, 40, 43
Conformal dielectric, 12
Continuous-surface variation model (CSV model), 153–162, 164, 167–172, 177, 178
Corner-based methodology, 121, 148
Correlation function, 129, 141, 142
Correlation length, 129, 147
Coupling capacitance, 9
Critical net, 212
CSV-based sensitivity approach, 153, 174–177
Cumulative distribution function (CDF), 144–146

D
Deep sub-micrometer (DSM), 1
Derived variable, 157, 158, 164
Design flow, 1, 2
Design for manufacturability (DFM), 63
Dielectric constant, 8
Direct boundary element method (direct BEM), 5
Direct boundary integral equation (direct BIE), 14
Dirichlet condition (i.e. Dirichlet boundary), 10
Discontinuous surface variation model (DSV model), 154, 155, 160, 161, 177
Discretized BIE, 40, 42, 43, 48, 50, 64
Distance limit, 212–216, 219–222, 224, 225
Distance query, 210
Domain discretization method, 10, 11
Dominant relationship (i.e. domination relationship), 199
Dominant variable, 162, 166
Doping profile, 92, 102–104, 106

W. Yu and X. Wang, *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*, DOI 10.1007/978-3-642-54298-5,

E
Embedded air gap, 70
Environment conductor, 9
Equivalent charge method, 11
Equivalent resistance network, 72, 84
Extended Jacobi (EJ) preconditioner, 51–54, 61, 66, 68

F
Far-field, 34
FastCap, 13, 15
Fast multipole method (FMM), 11, 14
Fictitious domain method, 63
Field solver, 2–6
Finite difference method (FDM), 10, 11
Finite-domain model, 10, 14
Finite element method (FEM), 10, 11
Floating dummy-fill, 39, 63, 65–67, 69, 70
Floating random walk (FRW), 5, 6
Free-space Green's function, 12, 15
Frequency-dependent substrate parasitics, 107–119
Full-path capacitance, 124, 133, 142–144
Fundamental solution, 14, 15

G
Gate-level simulation, 1
Gaussian distribution, 122, 130, 131
Gaussian-Hermite quadrature, 132–134
Gaussian surface, 17
Gauss-Legendre integration, 41, 42
Geometric variation, 121, 122
GMRES algorithm, 12
Graphic processing units (GPUs), 14, 18
Green's function, 11, 12, 14–17
Green's function table (GFT), 185, 187–190, 193, 194, 200, 201, 203, 204
Grid-based variation model, 128–130, 134–136
Grid-Octree hybrid structure, 222–224, 226, 229–231
Grid structure, 220, 221, 227

H
Hermite polynomial, 121, 123, 128, 130–133, 136–138
Hermite polynomial collocation (HPC), 121, 124, 128, 130–134, 136, 141, 145, 147
Hierarchical block boundary element method (HBBEM), 16
Hierarchical FRW (HFRW), 17
Hierarchical method, 11, 15
H-matrix, 13
Homogenous chaos expansion, 130

I
Importance sampling (IS), 190–195, 205, 208
Incomplete candidate list, 215–216, 220, 222, 224, 225
Incremental BEM, 124–128, 148
Independent variable, 162, 164, 172, 175
Indirect boundary element method (indirect BEM), 10–15
Infinite-domain model, 9, 10, 13
∞-norm, 211
Interconnect parasitics, 1
Inter-window covariance of capacitance, 165–167
Iterative solver, 19, 20, 26–34, 37

J
Jacobi preconditioner (i.e. diagonal preconditioner), 51–54, 61, 66, 68

K
KCL (Kirchhoff's current law), 85
K-D tree, 210, 218

L
Laplace equation, 9, 14, 15
Lateral resistivity variation, 91, 104–106
Library-building procedure, 121, 122, 125, 148
Linear boundary elements, 74–83, 86
Linear-model HPC, 153, 167, 170, 172–178
Line-edge roughness (LER), 153, 167–177
Line-width roughness (LWR), 168
Local expansion, 21, 23–24, 26
Localization character, 15
Lower-rank matrix compression, 11, 13, 14

M
Manhattan geometries, 211
Master conductor, 9, 17
Mesh neighbor preconditioner, 53
Method of moment (MoM), 11
Monte Carlo (MC) integration, 16
Multi-corner, 122

Multilayered Green's function, 12, 14
Multipole accelerated (MPA) method, 13, 14
Multipole expansion, 21–24, 26

N
Nearest neighbor queries (NNQ), 210
Near-field, 25
Nearly singular integral, 41
Neighbor region, 215, 216, 220, 224, 225, 227
Neumann condition (i.e. Neumann boundary), 9, 10, 14
Non-stratified substrate, 92
Non-uniform density partitioning, 46
Nonuniform meshing, 95–96
Numerical reduction, 96–99

O
Octree, 198–200
Overlapped domain decomposition method (ODDM), 39, 54–57

P
Parallel computing, 34, 36
Parasitic effect, 1, 2
Parasitic extraction, 1–4, 6
Parasitic parameter, 1
PASCAL, 39, 63, 69
Pattern library, 3
Pattern-matching approach, 3
Pattern mismatch, 3
Perturbed equation, 112–114
Polarization charge, 12
Precorrected fast Fourier transform (pFFT), 13
Principle factor analysis (PFA), 153, 161–166, 178
Probability density function (PDF), 16
Process technology, 179, 184, 185, 189, 201, 202
Process variation, 4, 121–152
Pruning effect, 221
pthread, 200

Q
QBEM, 54, 59–62, 66, 68, 70
Q3D, 10
QMM cutting number, 45, 46, 57, 61
Quadratic model, 161, 162, 167, 172, 175
Quasi-multiple medium (QMM) method, 15, 16
QuickCap, 209

R
Random number generator, 200
Random variation, 122, 129, 148
Rank-k approximation, 30
Raphael, 10
Rapid3D, 209
Ratio of structure dimension to correlation length (the L/η ratio), 159
Recompression, 35–37
Relative permittivity, 8
Rough-surface effect, 122
RWCap, 18

S
Self-capacitance (i.e. total capacitance), 9
Semi-analytical approach, 10
Sensitivity calculation, 123
Sherman-Morrison-Woodbury formula, 108–113
Shielding effect, 3
Short-range interconnects, 153, 168, 173, 178
Signal integrity, 1, 4
Sign-off verification, 209
Singular integral, 41, 49, 52
Singular-value decomposition (SVD), 13
Space management, 179, 195–208
Sparse grid quadrature, 132–134
Spatial cell, 211, 219
Spatial correlation, 122–124, 129, 130, 133–144
Spatially correlated random process variation, 121, 148
Spectral stochastic collocation method (SSCM), 122, 124, 131
SpiceLink, 56, 57
Square-counting, 3
statCap, 161, 163, 164, 173
Statistical capacitance extraction, 121, 122, 124, 133–148
Stratified sampling (SS), 179, 191, 192, 195, 197, 205, 208
Stratified structure, 47
subRCbem, 116–119
Substrate noise coupling, 2, 4
Surface Green's function, 16, 17, 179–188, 192, 194, 197, 208
Systematic variation, 122, 129

T
Technology CAD, 4
Technology precharacterization, 3, 4

Tellegen's theorem, 169
Timing analysis, 1
Total-charge Green's function approach, 12, 14
Transition domain, 179, 182, 184, 185, 188, 189, 198, 203, 208
Two-step approach, 108, 110–116, 119

V
Variable reduction, 162–163
Variance reduction, 179, 190–198, 201, 205, 208
Variation as a whole model (VAW model), 155, 159–161, 168, 170, 178

W
Weighted principle factor analysis (wPFA), 153, 161–165, 167, 172, 174, 177, 178
Weight value, 179, 181, 182, 184–190, 192–198, 205, 208
Weight value table (WVT), 185, 188–190, 193–195, 200, 201, 203, 204
Window structure, 123, 134