University of California

Los Angeles

Modeling and Design Optimization Considering Nanometer Process Variation Effects

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Electrical Engineering

by

Jinjun Xiong

2006

The dissertation of Jinjun Xiong is approved.

Milos D. Ercegovac

Lieven Vandenberghe

Ingrid Verbauwhede

Lei He, Committee Chair

University of California, Los Angeles

2006

To My Family.

Table of Contents

List of Figures

List of Tables

Acknowledgments

This dissertation has been developed over the past few years. Many people have helped in many ways, and I happily acknowledge them.

It is clear, however, who should go first. Dr. Lei He, my adviser, has continuously given me guidance and support over the course of my Ph.D. studies. He not only helped me to develop skills to solve a problem, but also encouraged me to explore new research directions and identify interesting research problems as an independent researcher. This dissertation would not exist without his enormous encouragement, support, and inspiration.

I am indebted to Dr. Ingrid Verbauwhede, Dr. Milos D. Ercegovac, and Dr. Lieven Vandenberghe for serving as members of my dissertation committee. I am particularly grateful to Dr. Lieven Vandenberghe for his help and advice about my research and career plan. I enjoyed every courses he offered at UCLA and being a Teaching Assistant with him.

Dr. Chandu Visweswariah and Dr. Vladimir Zolotov have earned my special thanks as my research mentors when I was an intern at the IBM Thomas J. Watson research center. Their intelligence, dedication, and patience have set a model for myself as a researcher in the future. I enjoyed every minute working with them. Every discussion with them was a source of inspiration. Chapters 3, 4, 7 and 8 of this dissertation have been benefited from interactions with them. I thank Dr. Chandu Visweswariah in particular for bringing me to IBM as an intern. The half year experience at Yorktown was just not long enough to enjoy the research, the people, and the abundant learning opportunities.

My sincere thanks go to Dr. Andrew B. Kahng for his help with my research and my career plan. I enjoyed collaborating with him on the study of CMP-induced systematic variations, which is reported in Chapter 5. His meticulous attention to every detail of research has also earned my hearty respect.

I have met a group of fabulous people at Rio Design Automation, Inc. My half-year internship with them was a blessing for me. That experience helped sharpen my skills in developing product-quality softwares. I gratefully acknowledge them for their helping hands and minds.

I am indebted to all the professors who have imparted to me the knowledge that expanded my view of the world and deepened my appreciation of nature. My particular thanks go to Prof. Miodrag Potkonjak, Prof. Majid Sarrafzadeh, Prof. Jason Cong, Prof. Mani B. Srivastava, Prof. Asad A. Abidi, Prof. Chih-Kong Ken Yang, Prof. Steven Jacobsen, Prof. Kewal K. Saluja, Prof. Parameswaran Ramanathan, Prof. Charles Kime, Prof. Charlie Chung-Ping Chen, Prof. Shiv G. Kapoor, and Prof. Mark W. Spong.

I am also indebted to my former advisers, Prof. Jing Pei at Tsinghua University, and Prof. Xudong Zhang at the University of Illinois, Urbana-Champaign. They helped me in many ways on the journey toward my Ph.D. studies.

It has been a great pleasure to collaborate with researchers at the Computer Science Department at Tsinghua University. Collaboration with them has generated some of the work reported in Section 9.1. In particular, I would like to acknowledge Prof. Xianlong Hong, Prof. Tong Jing, Ms. Lin Zhang, and Mr. Jinghong Liang for their contributions.

I treasure the opportunity I have had to work with a group of wonderful people at different schools, inside and outside classes, and would like to thank them for being around with helpful hands, especially those friends at the UCLA Design Automation Lab. In particular, I thank Changbo Long, King Ho Tam, Lerong Cheng, and Yiyu Shi for the amazing work we have done together. The work on designing sleep transistor for low power as reported in Section 9.3 is a joint project with Changbo Long. King Ho Tam has helped on the experiments and data-collection for the initial work on buffer insertion considering process variation as presented in Section 6.3; and later we jointly worked on the study of CMP-induced process variations as discussed in Chapter 5. In particular, the experiment results as reported in Section 5.4 are due to his implementation. Lerong Cheng has helped me with the proof of Theorem 5, and is collaborating with me on the work of FPGA design optimization considering process variations. Yiyu Shi and I are currently working together on the project of power network optimization considering process variation effects.

Last, and most of all, I am grateful to my family. I thank my wife Qin in particular. This dissertation becomes so humble compared to her constant inspiration, support, and love.

Vita

1976	Born, People's Republic of China.
1994–1998	B.E. (honor) in Precision Instrument, Measurement and Control, Tsinghua University, Beijing, China. Distinguished Undergraduate Fellowship, 1994–1998. Research Assistant, Optical Memory National Engineering Research Center, Tsinghua University, 1997–1998.
1996–1998	B.E. in Industrial Engineering, Tsinghua University, Beijing, China.
1998–2000	M.E. (honor) in Precision Instrument, Measurement and Control, Tsinghua University, Beijing, China. Distinguished Graduate Fellowship, 1998–2000. Research Assistant, Optical Memory National Engineering Research Center, Tsinghua University, 1998–2000.
2000–2001	First year Ph.D. program, in Mechanical and Industrial Engineering, University of Illinois, Urbana-Champaign, Illinois, USA. Research Assistant, UIUC Bio-Mechanical and Ergonomic Lab, 2000–2001.
2001–2002	M.S. in Electrical and Computer Engineering, University of Wisconsin, Madison, Wisconsin, USA. Distinguished Graduate Fellowship, 2001–2002.

2002–present	Ph.D. program, in Electrical Engineering, University of California, Los Angeles, California, USA. Distinguished Graduate Fellowship, 2002–2003. Teaching Assistant, Applied Numerical Computing, Fall 2004, Spring 2005. Research Assistant, UCLA Design Automation Lab, 2002–present.
2002	Beijing Science and Technology Award (Bronze), the Government of Beijing, China, 2002. For the work done at Optical Memory National Engineering Research Center, Tsinghua University, 1998–2000.
2004	Research Intern, Rio Design Automation, Inc., Santa Clara, California, USA. Worked on the first commercially available package-aware chip design tool, RioMagic®.
2005	Research Intern, IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA. Worked on the first product-quality statistical static timing analysis tool, EinsStat®.

Publications

Patents:

Jing Pei, Yu Xiao, **Jinjun Xiong**, "Seamless Integration of Multiple Storage Systems under One Target", *China Patent*, CN 1094217C, issued on Nov. 13, 2002.

Jing Pei, **Jinjun Xiong**, Yuan Li, Longfa Pan, "Architecture and Implementation of a Plug-and-play High-capacity Virtual Compact Disc Storage System", *China Patent*, CN 1118758C, issued on Aug. 20, 2003.

Jinjun Xiong, Jing Pei, "An Extendable High-capacity Virtual Compact Disc File System: Construction Algorithms and Hardware Implementation", *China Patent*, CN 1137441C, issued on Feb. 4, 2004.

Jing Pei, **Jinjun Xiong**, Yu Xiao, "A Multiple-driver-connected Compact Disc Library System under Single SCSI Logic Unit", *filed with the China Patent Office*, CN 1263310A, Aug. 2000.

Natesan Venkateswaran, Chandu Visweswariah, **Jinjun Xiong**, Vladimir Zolotov, "System and Method of Criticality and Sensitivity Prediction in Statistical Timing Analysis", *filed with the U.S. Patent Office*, Docket YOR-8-2005-0358US1, Nov. 2005.

Chandu Visweswariah, **Jinjun Xiong**, Vladimir Zolotov, "System and Method for Optimization of a Digital Integrated Circuit with Yield Considerations", *filed with the U.S. Patent Office*, Docket YOR-9-2005-0500US1, February 2006.

Book:

Jinjun Xiong, "Mastering Flash 4, Illustrated", (in Chinese), *China Electricity Power Press*, Beijing, China, ISBN: 7-5083-0303-2, 2000.

Journal Papers:

Xudong Zhang, **Jinjun Xiong**, and Angela M. Bishop, “The effects of load and speed on lumbar vertebral kinematics during lifting motions”, *Human Factors*, Vol. 45, Issue 2, pages 2969–306, 2003.

Xudong Zhang and **Jinjun Xiong**, “Model-guided derivation of lumbar vertebral kinematics in vivo reveals the difference between external marker-defined and internal segmental rotations”, *Journal of Biomechanics*, Vol. 36, Issue 1, pages 9–17, Jan. 2003.

Jinjun Xiong and Lei He, “Full-chip routing optimization with RLC crosstalk budgeting”, *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 23:366–377, March 2004.

Jinjun Xiong and Lei He, “Extended global routing with RLC crosstalk constraints”, *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, 13:319–329, March 2005.

Jinjun Xiong and Lei He, “Full-chip multilevel routing for power and signal integrity”, *Integration, the VLSI Journal*, to appear, 2006.

Lei He, Andrew B. Kahng, King Ho Tam, and **Jinjun Xiong**, “Simultaneous buffer insertion and wire sizing considering systematic CMP variation and random Leff variation”, *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, to appear, 2006.

Conference Papers:

Jinjun Xiong, Longfa Pan, Jing Pei, and Desheng Meng, "Eight terabyte storage system by a new file system", in *Fifth Int'l Symposium on Optical Storage*, SPIE Vol. 4085, pages 327-31, 2000.

Xudong Zhang, **Jinjun Xiong**, and Angela M. Bishop, "The effects of load and speed on vertebral kinematics during lifting motions", in *Forty-fifth Annual Meeting of Human Factors and Ergonomic Society*, 2001.

Xudong Zhang, **Jinjun Xiong**, and Angela M. Bishop, "Vertebral kinamtic descriptions based on in vivo measurement of surface marker motions", in *Twenty-fifth Annual Meeting of America Society of Biomechanics*, 2001.

Jinjun Xiong, Jun Chen, James Ma, and Lei He, "Post global routing optimization with RLC crosstalk constraints", in *Proc. Int. Conf. on Computer Aided Design*, pages 504–509, Nov. 2002.

Ling Zhang, Tong Jing, Xianlong Hong, Jingyu Xu, **Jinjun Xiong**, and Lei He, "Performance optimization global routing with RLC crosstalk constraints", in *Proc. of Intl. Conference on Application Specific Integrated Circuits*, pages 191–194, Oct. 2003.

Jinjun Xiong and Lei He, "Full-chip multilevel routing for power and signal integrity", in *Proc. Design Automation and Test in Europe*, pages 1116 – 1121, Feb. 2004.

Jinjun Xiong and Lei He, "Integrity-driven power and signal network co-design", in *ACM/IEEE International Workshop on Timing Issues, Austin, Texas*, Feb. 2004.

Changbo Long, **Jinjun Xiong**, and Lei He, "On optimal physical synthesis of sleep transistors", in *Proc. Int. Symp. on Physical Design*, pages 156–161, March 2004.

Ling Zhang, Tong Jing, Xianlong Hong, Jingyu Xu, **Jinjun Xiong**, and Lei He. "Performance and RLC crosstalk driven global routing", in *International Symposium on Circuits and Systems*, pages 65–68, May 2004. **Best Student Paper**.

Xin Zhao, Yici Cai, Qiang Zhou, Xianlong Hong, Lei He, and **Jinjun Xiong**, "Shielding area optimization under the solution of interconnect crosstalk", in *International Symposium on Circuits and Systems*, pages 23–26, May 2004.

Lei He, Andrew B. Kahng, Kingho Tam, and **Jinjun Xiong**, "Variability-driven considerations in the design of integrated-circuit global interconnects", in *IEEE VLSI Multilevel Interconnection Conference*, Oct. 2004. **Invited Paper**.

Jinjun Xiong and Lei He, "Probabilistic congestion model considering shielding for crosstalk reduction", in *Proc. Asia South Pacific Design Automation Conf.*, pages 739–742, Jan. 2005.

Tong Jing, Ling Zhang, Jinghong Liang, Jingyu Xu, Xianlong Hong, **Jinjun**

Xiong, and Lei He, "A min-area solution to performance and RLC crosstalk driven global routing problem", in *Proc. Asia South Pacific Design Automation Conf.*, pages 115–120, Jan. 2005.

Lei He, Andrew B. Kahng, Kingho Tam, and **Jinjun Xiong**, "Design of IC interconnects with accurate modeling of chemical-mechanical planarization", in *Intl. Society for Optical Engineering (SPIE) Symposium on Microlithography*, Feb. 2005.

Jinjun Xiong, Kingho Tam, and Lei He, "Buffer insertion considering process variation", in *Proc. Design Automation and Test in Europe*, pages 970–975, 2005.

Lei He, Andrew B. Kahng, Kingho Tam, and **Jinjun Xiong**, "Simultaneous buffer insertion and wire sizing considering systematic CMP variation and random Leff variation", in *Proc. Int. Symp. on Physical Design*, pages 78–85, April 2005.

Jinghong Liang, Tong Jing, Xianlong Hong, **Jinjun Xiong**, and Lei He, "Power Ground network aware and row-based solutions to the crosstalk driven routing problem", in *Proc. of Intl. Conference on Application Specific Integrated Circuits*, Oct. 2005.

Jinjun Xiong, Yiu-Chung Wong, Egino Sarto, and Lei He, "Constraint driven I/O planning and placement for chip-package co-design", in *Proc. Asia South Pacific Design Automation Conf.*, Jan. 2006.

Jinjun Xiong, Vladimir Zolotov, Natesan Venkateswaran, and Chandu Visweswariah, "Criticality computation in parameterized statistical timing", in *ACM/IEEE International Workshop on Timing Issues, San Jose, California*, Feb. 2006.

Vladimir Zolotov, **Jinjun Xiong**, and Chandu Visweswariah, "Computation of yield gradients from statistical timing analysis", in *ACM/IEEE International Workshop on Timing Issues, San Jose, California*, Feb. 2006.

Jinjun Xiong and Lei He, "Fast buffer insertion considering process variation", in *Proc. Int. Symp. on Physical Design*, April 2006.

Jinjun Xiong, Vladimir Zolotov, and Lei He, "Robust extraction of spatial correlation", in *Proc. Int. Symp. on Physical Design*, April 2006. **Best Paper Award**.

Jinjun Xiong, Vladimir Zolotov, Natesan Venkateswaran, and Chandu Visweswariah, "Criticality computation in parameterized statistical timing", in *IEEE/AMC Design Automation Conference, San Francisco, California*, July 2006.

Lerong Cheng, **Jinjun Xiong**, and Lei He, "FPGA performance optimization via chipwise placement considering process variations", in *Sixteenth International Conference on Field Programmable Logic and Applications, Madrid, Spain*, August 2006.

Abstract of the Dissertation

Modeling and Design Optimization Considering Nanometer Process Variation Effects

by

Jinjun Xiong

Doctor of Philosophy in Electrical Engineering

University of California, Los Angeles, 2006

Professor Lei He, Chair

As the CMOS technology marches into the nanometer manufacturing regime, it comes upon many hurdles that would stop its journey forward if not appropriately handled. Process variation is among them. Process variation prevents the printing of exact geometries onto silicon as designers intend for, so that even the same copy of a transistor at different locations on the same die will be different. Process variation also prevents an exact prediction of the characteristics of transistors and interconnects, let alone a whole design consisting of billions of them.

This dissertation addresses this increasingly important issue of handling process variation to sustain design success into the future. It consists of four key aspects, i.e., characterization, modeling, analysis, and optimization.

This dissertation first develops two robust techniques to characterize different process variation components for a manufacturing process. Within-die spatial variation is characterized either as a homogeneous and isotropic random field or a positive semidefinite correlation matrix. From noisy measurement data, two robust extraction techniques are developed to extract either a valid spatial correlation function that describes the spatial variation behavior of a homogeneous and isotropic random field, or a guaranteed positive semidefinite correlation matrix that captures the spatial correlation effects of any manufacturing process.

This dissertation then proposes an efficient method to model spatial correlations in the context of parameterized statistical static timing analysis (SSTA). It proves that this method is a generalization of an existing method based on principle component analysis (PCA) for spatial correlation modeling. This method can simultaneously and efficiently capture multiple spatially correlated process parameters with different spatial properties. Compared to the existing PCA-based approach, this method is significantly more efficient but does not compromise accuracy. Using this method, the overhead of modeling spatial correlation in the parameterized SSTA is almost negligible.

This dissertation also presents a comprehensive empirical study on the impact of systematic variation caused by chemical mechanical polishing (CMP) on interconnect parasitics and design optimization. Two typical variation effects — dummy fill insertion for planarization, and dishing and erosion caused by CMP — are studied with the development of a CMP-aware RC parasitic extraction model for global interconnects.

This dissertation proposes a novel methodology for delay minimization through buffer insertion that takes different process variation effects into account. Two different techniques to handle the correlated process variations under nonlinear operations are developed. A provable transitive closure pruning rule is proposed such that pruning can be done in linear time in the presence of process variation. The proposed techniques enable an efficient implementation of variation-aware buffer insertion, and it shows that buffer insertion considering correlated process variations can significantly improve the parametric timing yield compared to the conventional variation-oblivious buffering solutions.

This dissertation also proposes one variation-aware metric, called *conditional criticality probability*, to guide the optimization engine to explore different design alternatives in a multi-dimensional process space while considering the effects of process variation. This metric is variation-aware in the sense that it captures the criticality of each portion of the circuits for the whole process space, thus pinpointing the exact weak parts that need to be worked on in order to improve the design. A novel algorithm is developed to compute the conditional criticality probability accurately for all timing edges of a design. The algorithm is efficient and proven to have linear complexity in time and space.

This dissertation also contributes to the ideal goal of any design, i.e., design for yield or design for profit directly. It first shows that the parametric timing yield of a design can be analytically written as a function of any portion of the design even in the presence of process variations. It then develops a novel and efficient method to compute the gradient of parametric yield with respect to the delay of each gate or wire in the design. The resulting gradients can be rank-ordered for discrete optimization in a physical synthesis setting, or fed to a nonlinear optimizer for continuous optimization of design parameters such as transistor sizes, thus enabling formal design for yield (or design for profit) optimization.

To the best of our knowledge, this dissertation is the first of the kind that provides novel solutions to all aspects of design automation, ranging from variation characterization, variation modeling, and variation-aware analysis, to variation-aware optimization, in order to consider the nanometer process variation effects.

CHAPTER 1

Introduction

CMOS manufacturing in the nanometer region has resulted in miraculous achievements in the world of electronics, where high performance, low power, multi-functionality, reliability, and small sizes are often targeted at the same time. Because of the aggressive scaling down of transistors and interconnects, ultra-large-scale integration (ULSI), system-on-a-chip (SOC), and system-in-package (SIP) are common-places for today's chip designers, package designers, and system integration designers.

This achievement is made available through advanced manufacturing technologies, innovative design techniques, and sophisticated CAD (computer-aided design) support, and the three aspects are connected to each other with many interactions and mutual influences. These connections become even a necessity as the CMOS technology scales down to the nanometer region, in which inevitable manufacturing fluctuations and imperfections make it a Herculean task to predict the characteristics of transistors and interconnects printed on silicon. We are not able to make the same transistors on different copies of the same chip, or even at different locations on the same chip. The impact of this variation on design is tremendous, and it has generated a lot of research interest in finding ways to handle the variation in a positive manner, broadly called "design for manufacturing" (DFM).

Although innovation in all three fields is required to address the variability problem, this dissertation mainly focuses on the CAD aspect innovation to continue the journey toward successful electronic designs. Section 1.1 gives a brief introduction on process variation, its causes, and its impact on CAD techniques. Section 1.2 then reviews some recent work on process variation. Section 1.3 discusses the major contributions of this dissertation. Section 1.4 concludes this chapter with outlines of this dissertation.

1.1 Introduction to Process Variation

Three closely related but distinct concepts describe variability phenomena for a design [30, 70, 72], i.e., *design uncertainty*, *operating uncertainty*, and *process variation*. Because these terms are sometimes (confusingly) used interchangeably in the literature, we would like to distinguish them explicitly so that this dissertation can be set in a clear context.

- *Design uncertainty*: describes the hardware-model discrepancy. For example, circuit models used in any simulation environment (such as SPICE) are an approximated description of physical-world phenomena, and they almost always cause some error in predicating the real circuit response. Another example is engineering approximation, which happens almost all the time during design, either to reduce simulation runtime, or to simplify a complicated problem and make it tractable.

- *Operating uncertainty* describes the inability to predict the operating environments under which a circuit functions. The operating condition may include supply voltage, ambient temperature, and on-chip temperature gradient. For example, the demand for low power causes aggressive scaling of supply voltages, which in turn make voltage variations a significant part of the overall variation. Similarly, the quest for higher operating frequency causes large within-die temperature variations. Because the performance of both devices and interconnects is temperature-dependent, the temperature variation causes performance variation.

- *Process variation*: describes those random phenomena that happen in the course of manufacturing chips. These random phenomena make the physical characteristics of chips differ from what designers intend, and the characteristics also tend to vary among all manufactured chips of the same design. Process variation is inherent to any manufacturing process.

This dissertation considers the impact of process variation on design.

Process variation is mainly meant for physical process parameters, such as channel length, doping density, oxide thickness, and metal thickness and width. Process variation has also, however, meant for design parameters, including front-end-of-line (FEOL) device characteristics (such as drive current, sub-threshold leakage current, threshold voltage and gate delays), and back-end-of-line (BEOL) interconnect characteristics (such as interconnect resistance and capacitance). All these design parameters are functions of those varying physical process parameters.

Process variation can be classified as either *static* or *dynamic* [71]. Static process variation does not change over time, or changes slowly in a small scale. Static process variation is usually caused by manufacturing imperfections. In contrast, dynamic process variation changes over time. For example, negative bias temperature instability (NBTI) increases PMOS device threshold voltage and reduces its current drive differently at different product ages. And it has also been observed that the critical path in a design changes over its life span.

According to the sources of variation, process variation can also be classified into two types [93]:

- *Catastrophic defects* are caused by isolated random events (such as particles or other contaminations) during manufacturing, which render chips non-functional.

- *Parametric variations* are caused by random fluctuations in process conditions so that the physical properties of some parameters on a chip differ from the original design. The fluctuations may include aberrations in stepper lens, the dopant density, and the manufacturing temperature.

This dissertation focuses on parametric process variations that are static.

One of the most important impacts of process variations on design is *manufacturing yield loss*. Manufacturing yield is defined as the ratio between the number of chips that meet design specifications and the total number of manufactured chips [63]. Chips can fail to meet design specifications because of either functional failures caused by catastrophic defects, or parametric variations. As shown in Figure 1.1, parametric variations cause functioning chips exhibiting a wide range of performance variation, and a fraction of them are outside the desired performance specification window. It is those "functioning" chips failing to meet design specification that cause parametric *yield loss*.

Yield is closely related to *profit* and it is a direct measure of the success of a product. Therefore, it is desirable to bring yield into design consideration so that designers can optimize a *design for yield or profit* directly. This dissertation contributes toward this end.

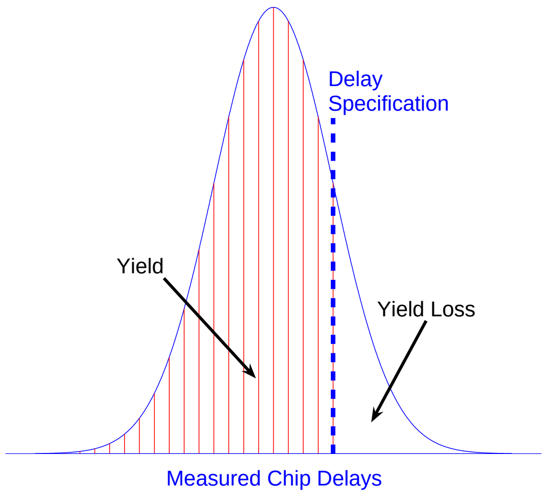


Figure 1.1: Yield and yield loss caused by parametric variations.

According to the scope of their occurrence, parametric process variations can be classified into the following two categories [30, 70, 10].

- *Die-to-die variation*, which is also called inter-die variation or between-die variation, describes the variation that affect parameters in different dies differently, but affect parameters within a die equally.

- *Within-die variation*, which is also called intra-die variation, across-chip variation, on-chip variation, spatial variation, or spatial correlation[1], describes the variation that affects process parameters at different locations of the same die differently.

Or, according to the scale of their causes, parametric process variations can also be classified into the following two categories [72].

[1]Spatial variation or spatial correlation is used more frequently in recent literatures than others.

- *Systematic variation* describes the deterministic portion of the variation. The physical cause of this variation is usually those identifiable deterministic features or patterns. For example, process conditions may vary randomly from wafer to wafer, but they may have a portion that is deterministically shared for all dies within the wafter. Similarly, process conditions may vary randomly from die to die, but a portion of them may be deterministically shared within a die. For example, inter-layer dielectric thickness variation is systematic and depends on layout density.

- *Random variation* describes the variation that is independent of any other conditions. The physical cause of this variation is usually not well understood, and thus it behaves more like a stochastic process. For example, discrete doping placement randomly changes MOSFET threshold voltage.

In the past, die-to-die systematic variation has been the major source of process variations [31, 30, 70]. Thus case analysis (e.g., worst and best cases) under different process corners is usually enough to cover the whole process space. For example, for timing analysis, chips are timed by sorting them into two cases, fast chips and slow chips, in order to handle the impact of die-to-die systematic variation. If timing signoff can be achieved in both cases, all chips are guaranteed to meet the timing specification in the whole process space with some safe margin to "guardband" designs.

This is no longer true, however, when other types of variation become comparable to the die-to-die systematic variation for nanometer CMOS manufacturing technologies. For example, one of the most prominent impacts of nanometer CMOS technology is that transistor channel lengths are on the order of sub-wavelengths of light. Sub-wavelength lithography makes it almost impossible to transfer the exact channel lengths from drawings to silicon. So both systematic

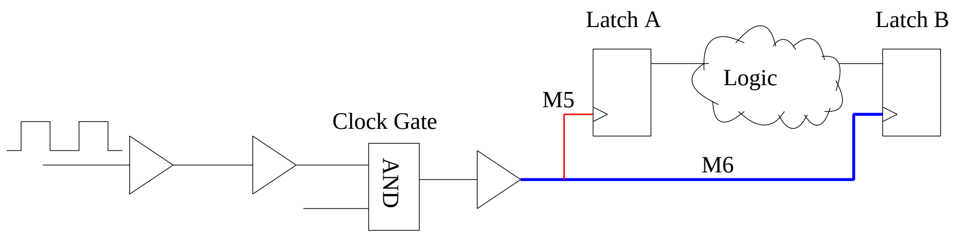


Figure 1.2: Example showing the problem of conventional case-based timing analysis. Courtesy of IBM.

and random within-die variations of channel length exceed the die-to-die systematic variation. Therefore, within-die variation is becoming a growing threat to the performance and functionality of future gigascale integration (GSI) [10].

One lesson learned from real hardware, courtesy of IBM [71, 95], is shown in Figure 1.2. According to the conventional case-based timing analysis, the circuit in Figure 1.2 should function well. Hardware tests, however, showed that the chip failed because of a hold-time violation between latch A and B, even though all other parameters of the hardware were within the manufacturing specifications. A close examination showed that the two clock routes in the final stage were dominated by two different types of metal layers (M5 and M6). Because of the large within-die random variation, the two metal layers exhibited different timing characteristics. They were not fast or slow at the same time! It is the mistrack between M5 and M6 that caused a clock skew between latch A and B, thus failed the chip with hold-time violation.

The lessons learned from the above example are multi-fold. First, it shows that the conventional case-based analysis is no longer adequate to cover the whole process space. Second, BEOL interconnect variations are no longer negligible compared to FEOL devices variation, and they contribute to a significant portion of the overall delay variation (more than 50% according to [73]). Third, BEOL variations are likely uncorrelated (mistrack), implying that the number of independent sources of variation grows quickly as more metal layers are added. Therefore, it is infeasible for the conventional case analysis to cover all possible corners, as the number of corners is exponentially proportion to the number of independent sources of variation.

To make things worse, the quest for high performance, large density, and low power in VLSI chips further decreases the already narrowing "design window", thus reducing chips' tolerance for variations. It becomes more and more difficult for chip designers to find a viable solution in the presence of increasingly large process variation. Designers are forced to look for an alternative approach capable of modeling process variation *accurately* and handling process variations *correctly* and *efficiently*. Solutions from this dissertation provide such alternatives for design exploration.

1.2 Literature Review

Statistical design optimization that considers process variation effects involves four key aspects: characterization, modeling, analysis, and optimization (see Figure 1.3). These four aspects are the basic building blocks of an integrated design framework that can combat variability. For example, in 1969, the author of [7] has already studied the effects of nonuniform diffusion process on the electrical characteristics of bipolar transistors. A model was developed to predict the sensitivity of current gain to different diffusion parameters. Guided by this model, the author was able to reduce the sensitivity of current gain to the junction depth by modifying the structure of transistor. This work embodied the importance of the above four key aspects, which are usually integrated together to improve a design. In view of the way that present-day methodologies handle variability, the

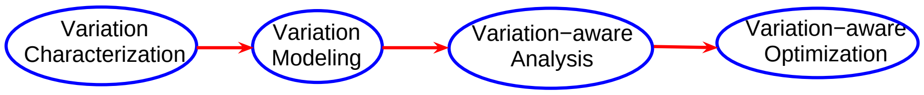


Figure 1.3: Four key aspects for design for variability: characterization, modeling, analysis, and optimization.

author of [93] cautioned that design sign-off based on corner analysis are "onerous, pessimistic, and risky, all at the same time." The author asserted that to truly deal with variability for design requires a change of our design methodology, ranging from modeling, analysis, and synthesis.

A wealth of publications can be found in the literature. Hence it would be impossible to attempt to provide a comprehensive literature review in this section. Instead, this section tries to present some representative work within each category, with emphasis on statistical timing analysis and optimization. Other work closely related to this dissertation will be reviewed at the beginning of each chapter whenever it becomes relevant.

The characterization of process variation is inherently more difficult than understanding the nominal or extreme behavior of the process [72]. Accurate characterization requires large amounts of data from foundry to estimate distributions and correlations among process parameters. Some phenomena are also tightly coupled with different design styles (such as layout patterns), which makes characterization even harder, as design and fabrication groups are usually belonging to different organizations in reality. A lot of work, however, have attempted to address this issue. For example, by analyzing a $0.18\mu m$ technology, [76] showed that a significant systematic within-die variation of channel lengths (called CD for "critical dimension") has strong dependence on the local layout patterns. A method was developed that can predict circuit performance based on the information of this systematic CD variation. [65] studied the impact of spatial pattern dependent variation on circuit performance caused by chemical-mechanical polishing (CMP). They showed that interconnect delay and clock skew were strongly affected by CMP and CD variations for both aluminum and copper interconnect technologies. More information on this regard can be found in [31, 76, 91, 8, 65, 33, 51, 49, 67, 77, 53, 73, 46].

Work on variation modeling tries to understand the impact of process variation on design, and hence provide vehicles to help designers achieve better designs. For example, [10] derived a model that describes the maximum clock frequency (FMAX) distribution of a microprocessor for a $0.25\mu m$ technology. It showed that within-die variations mainly impact the mean of FMAX, while die-to-die variation account for the majority of the variance of FMAX. Based on their model, the authors predicated that systematic within-die variation would impose the largest performance degradation for future technologies. [70] proposed a modeling and simulation methodologies to analyze the impact of various sources of variation on performance. The framework is based on a first-order sensitivity analysis. An excellent tutorial paper on statistical circuit modeling and optimization is given by [30]. More work on process variation modeling can be found in [30, 10, 14, 86, 70, 75, 32, 54, 50, 87].

Techniques for statistical static timing analysis (SSTA) can be mainly classified into two camps: path-based approaches such as [59, 60, 3, 44, 68] and block-based approaches such as [27, 1, 16, 94, 74, 114]. From the first camp, [44] proposed three numerical methods, i.e., parallelepiped method, ellipsoid method, and binding probability method, for timing analysis and yield predication. Because the number of paths can be exponential in terms of the size of a timing graph, paths-based SSTA techniques usually have high computation cost. From the second camp, [27] proposed a block-based SSTA based on a piece-wise uniform distribution (PDF) or a piece-wise linear distribution (CDF) modeling of gate delays. To handle the path re-convergence problem, a *dependent list* is maintained at each node that lists all previous stage nodes on which current node's arrival time depends. [1] considered both inter-die variation and intra-die variation (spatial variation) for SSTA. To handle the nonlinear variational maximum operation, the authors of [1] proposed to compute an upper bound CDF for the maximum operation [3], i.e., taking the component wise maximum operations instead of the maximum of the whole sum. Because this approximation may result in a very loose upper bound, the authors further refined the solution by a heuristic. [74] also proposed to find the upper and lower bounds for an arbitrary distribution. Under the normal distribution assumption of delays, [52] proposed a table-look-up based approach to convert the maximum between two random variables into a one-dimensional table look-up problem in order to compute the mean, variance, and covariance. Two seminal work on block-based SSTA are [16] and [94]. Both proposed to model the statistical timing quantities in the timing graph as a linear combination of a set of independent normal random variables, called the first-order canonical form in [94]. Efficient operations were defined for this linear representation, thus achieving linear complexity for SSTA computation. The efficiency of block-based SSTA techniques hence attracted more research attention on the block-based SSTA than the path-based SSTA recently. The SSTA techniques have also been extended to consider other effects, such as nonlinear timing models [15, 113], systematic process variation [36], statistical leakage power analysis [88, 79], and statistical yield analysis [68].

Work on statistical optimization is relatively new, but it has already generated a variety of publications in literature, in particular, gate sizing [19, 78, 2, 64, 84, 4]. For example, [19] employed the Lagrangian relaxation algorithm to find optimal transistor sizes. A SSTA engine was embedded inside the optimization loop to guide optimization. [78] proposed to identify a set of critical paths based on a "disutility" function that evaluates the mean and variance of gate and path delays. A statistical sizing algorithm based on a constrained nonlinear optimization formulation was developed to improve the delays of these selected critical paths, each node of which was heuristically weighted by its "criticality index", a concept originally defined in operations research community [28] and similar to the concept of "criticality" proposed by [94] to the CAD community. [37] performed a TILOS-like gate sizing algorithm to achieve a targeted yield. An incremental, parameterized SSTA tool is used to evaluate the qualify of solutions at each optimization step. A first-order linear delay model with fitted process sensitivities was used to capture both correlated and uncorrelated process parameters. Criticality probabilities [94] were used to actively guide optimization. The authors showed that this approach can improve performance and reduce run-time and area, when compared with the deterministic optimization.

Recently, statistical optimization has also been applied for power minimization [9, 64], and yield maximization [69, 20, 55]. Optimization techniques considering the CMP-induced systematic variation have also been studied [89, 35, 18].

1.3 Dissertation Contributions

The main theme of this dissertation echos the four key aspects as shown in Figure 1.3: characterization, modeling, analysis, and optimization. Chapter 3 deals with process variation characterization. Chapter 4 studies how to model spatial correlation for statistical timing analysis; while chapter 5 tries to understand how to model CMP-induced systematic variations and its impact on design optimization. The rest of the dissertation considers how to optimize designs for performance and for yield while taking into account process variations.

The major contributions of this dissertation are as follows:

- Two robust techniques to characterize manufacturing spatial variation. When the spatially correlated process variation is modeled as a *homogeneous and isotropic random field*, a valid spatial correlation function is extracted by solving a constrained nonlinear optimization problem; otherwise, a valid spatial correlation matrix is extracted through a modified alternative projection algorithm based on the theory of convex analysis. Both techniques guarantee that the extraction results are closest to the real measurement data even in the presence of significant random noises.

- An efficient method to model spatial correlations in the context of parameterized statistical static timing analysis (SSTA). Multiple spatially-correlated process parameters with different spatial properties are incorporated simultaneously and efficiently. The method guarantees to capture the underlying spatial correlation with minimal error.

- A comprehensive empirical study on the impact of systematic chemical mechanical polishing (CMP) variation on interconnect parasitics and design optimization. It shows that (1) fill insertion for CMP planarization significantly increases interconnect capacitance, and different fill patterns introduces additional variations; and (2) CMP-induced dishing and erosion effects can significantly increase interconnect resistance, but have limited impact on capacitance. Based on this study, a table-based best fill pattern insertion algorithm is developed that minimizes the impact of CMP-induced systematic variations on interconnect design optimization.

- A novel methodology to solve the buffer insertion problem considering process variations. Two different techniques to handle correlated process variations under nonlinear operations are developed. A provable transitive closure pruning rule is proposed that makes linear complexity variation-aware pruning possible. The proposed techniques enable an efficient implementation of variation-aware buffer insertion, and it shows that buffer insertion considering correlated process variations can significantly improve the parametric timing yield compared to the conventional deterministic buffering algorithms.

- A new variation-aware diagnostic metric, conditional criticality probability, for guiding robust circuit optimization. A novel algorithm is developed to compute the conditional criticality probability accurately for all edges in the timing graph of a design. The algorithm is efficient and proved to have linear complexity in time and space.

- A novel and efficient method to compute the gradient of parametric yield with respect to the delay of each gate or wire. The resulting gradients can be rank-ordered for discrete optimization in a physical synthesis setting, or fed to a nonlinear optimizer for continuous optimization of design parameters such as transistor sizes, thus enabling formal mathematical yield optimization directly.

1.4 Dissertation Outline

The rest of the dissertation is organized as follows.

Chapter 2 first gives a unified process variation model based on first-order approximation. It then reviews some important concepts that will be used extensively throughout the rest of the dissertation, including the first-order canonical form to represent the timing quantities in a timing graph, normal assumptions on random process variation, atomic operations on first-order canonical forms, and finally parameterized statistical static timing analysis (SSTA).

Chapter 3 presents two robust techniques to characterize spatial correlation based on noisy measurement data. Motivated by the necessity for a valid spatial correlation modeling, it presents two problem formulations that are suited for different manufacturing processes. If a manufacturing process results in a spatial correlation that can be modeled by a *homogeneous and isotropic random field*, based on the theory of random fields, it is sufficient to extract a valid spatial correlation function to characterize spatial correlation. This is achieved by solving a constrained nonlinear optimization problem in section 3.4. If a manufacturing process's spatial correlation can only be described through a correlation matrix, a guaranteed positive semidefinite correlation matrix must be extracted from the noisy measurement data. This is achieved in section 3.5 by employing a modified alternative projection algorithm based on the theory of convex analysis. Experiment results based on a Monte-Carlo model is reported in section 3.6, and it shows that the proposed techniques can recover the correlation function and matrix with very high accuracy even in the presence of significant random noises.

Chapter 4 presents an efficient method to model spatial correlations in the context of parameterized SSTA. It starts with the introduction of existing approaches to model the spatial correlation for parameterized SSTA. In particular, the existing approach based on principle component analysis (PCA) is discussed, and its major disadvantages are revealed. It then presents an efficient method to model spatial correlations with cases studies. Experiment results from real industrial designs show that the proposed approach is significantly more efficient than the existing PCA-based technique. Results also show that by using the proposed method, the overhead of modeling spatial correlation in parameterized SSTA is small.

Chapter 5 studies the impact of systematic variation caused by Chemical-Mechanical Polishing (CMP) on interconnect parasitics and design optimization. CMP-related variation caused by dummy fill insertion, and dishing and erosion is considered. Section 5.2.1 presents an algorithm that allows systematic exploration of different fill patterns that are "equivalent" with respect to foundry rules. Section 5.2.2 examines the impact of fills and fill patterns on interconnect capacitance. Section 5.2.3 studies interconnect parasitic variations caused by dishing and erosion based on a multi-step CMP process model. From these studies, section 5.2.4 develops a CMP-aware table-based model for interconnect parasitic extraction. The rest of the chapter solves a simultaneous buffer insertion, wire sizing, and fill insertion problem by using the developed CMP-aware model. It shows that the proposed algorithm can reduce delay, power, and area simultaneously compared to an existing algorithm.

Chapter 6 presents a novel method of solving the buffer insertion problem considering process variations (BIPV). Section 6.2 gives the problem formulation, and reviews the general dynamic-programming-based algorithm for solving the deterministic version of buffer insertion. Section 6.3 and 6.4 present two different techniques to solve the BIPV problems. Both techniques overload the key operations used in solving the deterministic buffer insertion. The first technique employs a numerical integration method to compute the joint probability density function (JPDF). The second employs the first-order canonical form to implicitly represent the JPDF. To speed-up dynamic-programming-based buffering algorithms, two variation-aware pruning rules, the two-sided threshold-based pruning rule and the transitive closure-based pruning rule, are discussed. It is proved that the transitive closure-based pruning rule makes linear complexity pruning possible even when process variation is taken into account. Thus the proposed techniques allow an efficient implementation of variation-aware buffer insertion. Experiment results confirm the efficiency of the proposed techniques.

Chapter 7 presents a new variation-aware diagnostic metric, conditional criticality probability, to guide robust circuit optimization in the presence of correlated process variation. Section 7.2 motivates the need for a variation-aware metric by showing that the conventional slack is no longer a good metric for design optimization. Section 7.3 and 7.4.3 develop a novel algorithm to compute the criticality probability accurately for all edges in the timing graph of a design. It also proves that the algorithm has linear complexity in time and space. Section 7.5 then introduces the concept of conditional criticality probability and the need for such concept for circuit optimization. It shows that the computation of conditional criticality probability is as easy as computing the unconditional counterpart. Experiment results presented in section 7.6 prove the correctness of the proposed algorithm in computing criticality compared to the Monte Carlo simulation. It also shows that for large industrial designs, the proposed algorithm computes all edge criticalities in short time. The high accuracy and fast speed of the algorithm warrant future applications of criticality probability to robust circuit optimization.

Chapter 8 proposes a novel and efficient method to compute yield gradients. Section 8.2 derives analytical formulas for the differentiation of the statistical maximum of two canonical forms with respect to all parameters of one canonical form. Section 8.3 first shows that how a circuit's performance can be represented as a function of one timing edge's canonical form. It then uses the formulas developed in section 8.2 to compute the gradient of a circuit's performance with respect to the timing edge of interest. Two possible ways of relating parametric yield gradients with respect to a circuit's performance gradient are discussed. By simple chain-ruling, it shows that the parametric yield gradient with respect to any timing edge of interest can be computed analytically, thus opening the possibility of direct design for yield. Some possible applications of this finding are elaborated in section 8.4.

Finally, the appendix in chapter 9 summarizes some other research projects that have been done over the course of this dissertation.

CHAPTER 2

Preliminary

2.1 First-order Process Variation Decomposition

We denote F as a measurable physical process parameter, such as channel length, channel width, silicon oxide thickness, and wire thickness. Because of manufacturing process variations, these physical parameters are no longer fixed values. We model the parameter as a random variable, and write it as

$$F = f_0 + F_r = f_0 + X_g + X_s + X_r. \tag{2.1}$$

To capture the spatial correlation between quantities at different locations, we further represent the spatial variation part at location j as follows:

$$X_{s,j} \quad = \quad \sum_k d_{j,k} X_{j,k}, \tag{2.2}$$

where $X_{j,k}$ are independent random variables, and $d_{j,k}$ are coefficients of $X_{j,k}$. In Chapter 4, we will present techniques on how to represent the intra-chip spatial variation in the form as shown in (2.2).

Therefore, for the process parameter F at location j, we have

$$F_j \quad = \quad f_0 + X_g + \sum_k d_{j,k} X_{j,k} + X_r, \tag{2.3}$$

where X_g, $X_{j,k}$, and X_r are all independent random variables. We call equation (2.3) as the *first-order process variation form.*

The rational behind this first-order process variation model is that if the underlying parametric variations are small, any nonlinear relationship can be reasonably captured by a first-order approximation. Such an approximation has been well accepted for statistical timing analysis [16, 94].

2.2 First-order Canonical Form

To model the impact of process variations on both device and interconnect characteristics (such as delay and power), we represent these characteristics of interest as random variables, which are complicated functions of the underlying physical process parameters, such as channel length, doping density, gate oxide thickness.

We employ a first order approximation technique to capture the impact of process parameters on the characteristics. Mathematically, it can be described as

$$D = d_0 + \sum_i d_i \cdot F^i, \tag{2.4}$$

where D is the characteristic of interest, and F^i are different underlying physical process parameters, which are not independent to each other in general. The nominal value of D is d_0, while sensitivities of D with respect to F^i are given by d_i. If the function that describes the characteristics of interest with respect to the process parameters is known, we can obtain (2.4) analytically via the first order Taylor expansion of the function. Otherwise, SPICE simulation can be used to extract the nominal value of d_0 and the sensitivities of d_i.

To capture the impact of process variations on D at location j, we plug (2.3) into (2.4) and obtain

$$D_j = d_0 + \sum_i d_i f_0^i + \sum_i d_i X_g^i + \sum_i \sum_k d_i d_{j,k} X_{j,k}^i + \sum_i d_i X_r^i. \tag{2.5}$$

The above equation can be also written compactly as

$$D_j = D_{j,0} + \alpha_{\mathbf{j}}{}^{\mathbf{T}}\mathbf{X}, \tag{2.6}$$

where $D_{j,0}$ is the mean value of D_j; $\mathbf{X}$ is a random variable vector with its each component independent to one another, including inter-chip global variation X_g^i, spatial correlation $X_{j,k}^i$, and purely random variation X_r^i; and $\alpha_{\mathbf{j}}$ is the coefficient vector of $\mathbf{X}$. We call (2.6) as the *first-order canonical form*.

For simplicity of presentation, in the following, we also write D in its expanded form as follows

$$D = d_0 + \sum_{i=1}^{n} d_i X_i + d_r X_r = d_0 + \alpha^{\mathbf{T}}\mathbf{X} \tag{2.7}$$

where d_0 is the mean or nominal delay of D; X_i are the correlated random variables that may be shared among different characteristics; X_r is the uncorrelated random variable that is not shared by anyone else; and d_i and d_r are the sensitivities to the variations X_i and X_r, respectively.

2.3 Modeling Process Variation as Normal Distribution

Recall that a random variable X follows a normal distribution[1] if and only if its probability density function (PDF) can be represented as

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/(2\sigma^2)}, \tag{2.8}$$

where μ and σ^2 are the mean and variance of X, respectively. The cumulative density function (CDF) of X is given by

$$F(x) = \int_{-\infty}^{x} f(t)dt. \tag{2.9}$$

[1]Normal distribution is also called Gaussian distribution.

It is obvious that the mean and variance are the only two parameters that determine a normal distribution. When $\mu = 0$ and $\sigma^2 = 1$, we also say that X follows a *standard normal distribution*, and we denote its PDF and CDF as $\phi(x)$ and $\Phi(x)$, respectively.

The definition of normal distribution can also be extended to the multivariable case. If $\mathbf{X}$ is a $n \times 1$ random vector that follows a joint normal distribution, its PDF is defined as follows

$$f(\mathbf{x}) \quad = \quad \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} exp(-\frac{1}{2}(\mathbf{x} - \mu)' \mathbf{\Sigma^{-1}} (\mathbf{x} - \mu)), \tag{2.10}$$

where μ is the mean vector of $\mathbf{X}$, Σ is the covariance matrix of $\mathbf{X}$, and $|\Sigma|$ is the determinant of Σ. The covariance matrix Σ is given by

$$\Sigma = \begin{bmatrix} \sigma_1^2 & cov(x_1, x_2) & cov(x_1, x_3) & ... & cov(x_1, x_n) \\ cov(x_1, x_2) & \sigma_2^2 & cov(x_2, x_3) & ... & cov(x_2, x_n) \\ cov(x_1, x_3) & cov(x_2, x_3) & \sigma_3^2 & ... & cov(x_3, x_n) \\ . & & & & . \\ cov(x_1, x_n) & cov(x_2, x_n) & cov(x_3, x_n) & ... & \sigma_n^2 \end{bmatrix}, \tag{2.11}$$

where $cov(x_i, x_j)$ is the covariance between x_i and x_j. The *correlation coefficient* $\rho_{i,j}$ between x_i and x_j is defined as

$$\rho_{i,j} \quad = \quad \frac{cov(x_i, x_j)}{\sigma_i \sigma_j}, \tag{2.12}$$

Knowing the mean vector μ and the covariance matrix Σ in (2.10) is enough to determine the multivariate normal distribution of $\mathbf{X}$. When each component of $\mathbf{X}$ has zero mean and unit variance, i.e., each component x_i is a standard normal distribution, then the covariance matrix Σ becomes the correlation matrix Ω, i.e.,

$$\Omega = \begin{bmatrix} 1 & \rho_{1,2} & \rho_{1,3} & \cdots & \rho_{1,n} \\ \rho_{1,2} & 1 & \rho_{2,3} & \cdots & \rho_{2,n} \\ \rho_{1,3} & \rho_{2,3} & 1 & \cdots & \rho_{3,n} \\ \cdot & & & & \cdot \\ \rho_{1,n} & \rho_{2,n} & \rho_{3,n} & \cdots & 1 \end{bmatrix}. \tag{2.13}$$

In the rest of the dissertation, we will assume that all X_i and X_r in (2.7) follows a standard normal distributions unless otherwise specified differently. Because all X_i and X_r follow a normal distribution, so is D in (2.7).

The mean value of D is given by d_0, and the variance of D is given by

$$\sigma_D^2 = \sum_{i=1}^{n} d_i^2 + d_r^2 = \alpha^{\mathbf{T}}\alpha. \tag{2.14}$$

For two different canonical forms that are given by

$$D_p = d_{p,0} + \sum_{i=1}^{n} d_{p,i}X_i + d_{p,r}X_{p,r} = d_{p,0} + \alpha_{\mathbf{p}}{}^{\mathbf{T}}\mathbf{X}, \tag{2.15}$$

$$D_q = d_{q,0} + \sum_{i=1}^{n} d_{q,i}X_i + d_{q,r}X_{p,r} = d_{q,0} + \alpha_{\mathbf{q}}{}^{\mathbf{T}}\mathbf{X}, \tag{2.16}$$

they form a joint bivariate normal distributions. Their covariance is computed as

$$cov(D_p, D_q) = \sum_{i=1}^{n} d_{p,i}d_{q,i} = \alpha_{\mathbf{p}}{}^{\mathbf{T}}\alpha_{\mathbf{q}}. \tag{2.17}$$

2.4 Atomic Operations on First-order Canonical Forms

2.4.1 Summation and Subtraction

For two canonical forms D_p and D_q, the sum of them is also a normal distribution, and it can be represented as another canonical form by summing their corresponding coefficients, i.e.,

$$\begin{aligned} D &= D_p + D_q & (2.18) \\ &= (d_{p,0} + d_{q,0}) + \sum_{i=1}^{n}(d_{p,i} + d_{q,i})X_i + d_{p,r}X_{p,r} + d_{q,r}X_{q,r} & (2.19) \\ &= d_0 + \sum_{i=1}^{n} d_i X_i + d_r X_r, & (2.20) \end{aligned}$$

where $d_0 = d_{p,0} + d_{q,0}$, $d_i = d_{p,i} + d_{q,i}$, and $d_r = \sqrt{d_{p,r}^2 + d_{q,r}^2}$.

Similarly, the difference between D_p and D_q is also a normal distribution, and it can be represented as another canonical form as follows

$$\begin{aligned} D &= D_p - D_q & (2.21) \\ &= (d_{p,0} - d_{q,0}) + \sum_{i=1}^{n}(d_{p,i} - d_{q,i})X_i + d_{p,r}X_{p,r} - d_{q,r}X_{q,r} & (2.22) \\ &= d_0 + \sum_{i=1}^{n} d_i X_i + d_r X_r, & (2.23) \end{aligned}$$

where $d_0 = d_{p,0} - d_{q,0}$, $d_i = d_{p,i} - d_{q,i}$, and $d_r = \sqrt{d_{p,r}^2 + d_{q,r}^2}$.

Because the results obtained after summation and subtraction are still a canonical form, we can repeatedly apply the above operations for more than two canonical forms.

2.4.2 Tightness Probability

The *tightness probability* of D_p over D_q is defined as the probability of D_p being greater than D_q, i.e.,

$$T_p = P(D_p > D_q). \tag{2.24}$$

As both D_p and D_q are normal distributions, the tightness probability T_p can be computed as [21, 13]

$$T_p = \Phi\left(\frac{d_{p,0} - d_{q,0}}{\theta}\right) \tag{2.25}$$

where Φ is the CDF of a standard normal distribution; and θ is given by

$$\theta = \sqrt{\sigma_p^2 + \sigma_q^2 - 2cov(D_p, D_q)}, \tag{2.26}$$

where σ_p^2 and σ_q^2 are variance of D_p and D_q, respectively.

Similarly, we can define the *tightness probability* of D_q over D_p, which can be numerically computed as $T_q = 1 - T_p$.

2.4.3 Maximum and Minimum Operations

The maximum (or minimum) of two normal distributions results in a distribution that is no longer normal in general. Thus it can not be represented as a canonical form exactly. For efficient computation, however, it is often desirable to keep all computation results in a uniform representation (such as the canonical forms), so that we can apply the same operations repeatedly for more than two variables.

This is achieved by approximating the maximum (or minimum) operation as a linear operation. The result of this approximation is a weighted sum of its two input canonical forms. The weight is closely related to individual canonical form's tightness probability. To see how this works, we first note that the exact mean and variance of max(D_p, D_q) can be computed as follows [21, 13]

$$\begin{aligned} \mu &= T_p \cdot d_{p,0} + T_q \cdot d_{q,0} + \theta \cdot \phi\left(\frac{d_{p,0} - d_{q,0}}{\theta}\right), & (2.27) \\ \sigma^2 &= T_p(\sigma_p^2 + d_{p,0}^2) + T_q(\sigma_q^2 + d_{q,0}^2) \\ &\quad +(d_{p,0} + d_{q,0}) \cdot \theta \cdot \phi\left(\frac{d_{p,0} - d_{q,0}}{\theta}\right) - \mu^2, & (2.28) \end{aligned}$$

where ϕ is the PDF of a standard normal distribution, and T_p and T_q are the tightness probability of D_p and D_q, respectively. Then the statistical maximum of D_p and D_q is approximated as follows

$$\max(D_p, D_q) = \mu + \sum_{i=1}^{n}(T_p \cdot d_{p,i} + T_q \cdot d_{q,i})X_i + d_r X_r, \tag{2.29}$$

where μ is the exact mean of $\max(D_p, D_q)$ as given in (2.27).

To match the variance of (2.29) to exact variance of $\max(D_p, D_q)$ as given in (2.28), [94] introduces an uncorrelated random variable X_r with coefficient d_r. That is, the value of d_r is determined by matching the variance of (2.29) to (2.28). It has been proved in [85] that the above matching process is guaranteed to find a positive number for d_r. Another way to match the exact variance of $\max(D_p, D_q)$ is to scale the coefficients of X_i in (2.29), thus without introducing the uncorrelated random variable in (2.29) [16].

The minimum operation is approximated in a similar way. We first compute the exact mean and variance of $\min(D_p, D_q)$ as follows

$$\mu = T_q \cdot d_{p,0} + T_p \cdot d_{q,0} - \theta \cdot \phi\left(\frac{d_{q,0} - d_{p,0}}{\theta}\right) \tag{2.30}$$

$$\begin{aligned}\sigma^2 &= (\sigma_p^2 + T_{p,0}^2)T_q + (\sigma_q^2 + T_{q,0}^2)(T_p) \\ &\quad +(d_{p,0} + T_{q,0}) \cdot \theta \cdot \phi\left(\frac{d_{q,0} - d_{p,0}}{\theta}\right) - \mu^2.\end{aligned} \tag{2.31}$$

Then the statistical minimum of D_p and D_q is approximated as follows:

$$\min(D_p, D_q) = \mu + \sum_{i=1}^{n}(T_q \cdot d_{p,i} + T_p \cdot d_{q,i})X_i + d_r X_r, \tag{2.32}$$

where μ is the mean of $\min(D_p, D_q)$ given in (2.30), and d_r is determined by matching the variance of (2.32) to the exact variance of $\min(D_p, D_q)$ as given in (2.31).

2.5 Parameterized Statistical Static Timing Analysis

Consider a design represented as a netlist, we abstract the design into an acyclic directed timing graph $G(V, E)$, where each node $v \in V$ represents either the input or output of a gate, each edge $e \in E$ represents either the interconnection between gates or the internal signal propagation paths within the gates. The direction of the edge represents the signal propagation direction along the edge. Latches and flip-flops are modeled as signal propagation with timing constraints. These timing constraints are transformed into timing checks as additional circuit outputs. We associate each edge in G with a timing quantity D_e that models the signal propagation delay from the source of the edge to the sink of the edge. In general, the edge delay D_e is a function of gate types, input signal slew, and output loads, and may also change the output signal slew. One such an example is shown in Figure 2.1.

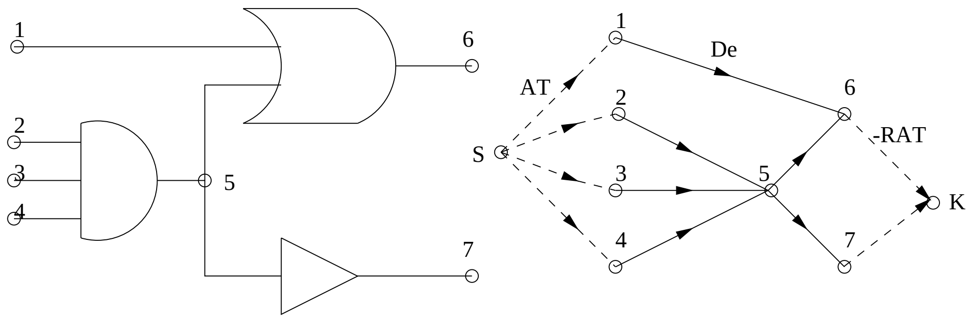


Figure 2.1: A circuit and its timing graph.

In the following, for simplicity, only the late mode statistical timing analysis is considered. We associate each node v in G with two quantities: (1) the arrival time AT, which is defined as the latest time that signals propagated from all primary inputs can switch at v; and (2) the required arrival time RAT, which is defined as the latest time that signals are allowed to switch at v without causing timing violations for all downstream end points, where timing tests will be conducted. The difference between RAT and AT gives the slack of of that node. Usually the AT at the primary inputs and the RAT at the end points are known *a priori* from design specification. Without loss of generality, we add a virtual source S and a virtual sink K into the timing graph G as shown in Figure

2.1, where the virtual source connects to all primary input with the delay on the edge equal to the AT at the primary input, and all end points connect to the virtual sink with the delay on the edge equal to the negative RAT at the end point. Therefore, we have zero for AT at the virtual source, and zero RAT at the virtual sink. In this way, we can abstract all circuits, both sequential and combinational, with all timing tests into a uniform timing graph representation. Moreover, the AT at the virtual sink gives the slack of the entire chip with its sign flipped. At any internal node v, the AT gives the longest path delay from the virtual source to the node v, the RAT gives the longest path delay from the virtual sink to the node with its sign flipped.

In block based timing analysis, we compute the AT at all timing points (except the virtual source) in G by forward propagating the AT in G in a breast first manner, and at each timing edge the AT is increased by D_e, and at each timing point the one with the maximum AT determines the AT at that timing point. Similarly, we compute the RAT at all timing points (except the virtual sink) by backward propagating the RAT in G in a breast first manner, and at each timing edge the RAT is decreased by D_e, and at each timing point the one with the minimum RAT determines the RAT at that timing point. Therefore, it is obvious that during the block based timing analysis, four atomic operations are required: summation, subtraction maximum, and minimum.

This applies to both deterministic timing analysis and parameterized statistical static timing analysis (SSTA). In deterministic timing analysis, all edge delays D_e are deterministic values. In contrast, in parameterized SSTA, all timing quantities are random variables represented by first-order canonical forms. In deterministic timing analysis, the four atomic operations are straightforward; in parameterized SSTA, the four atomic operations have to be defined statistically as shown in section 2.4.

CHAPTER 3

Modeling and Extraction of Spatial Correlation

The increased variability of process parameters makes it important yet challenging to extract the statistical characteristics and spatial correlation of process variation. Recent progress in statistical static timing analysis also makes the extraction important for modern chip designs. Existing approaches extract either only a deterministic component of spatial variation. Or these approaches do not consider the actual difficulties in computing a valid spatial correlation function, ignoring the fact that not every function and matrix can be used to describe the spatial correlation. Applying mathematical theories from random fields and convex analysis, we develop (1) a robust technique to extract a valid spatial correlation function by solving a constrained nonlinear optimization problem; and (2) a robust technique to extract a valid spatial correlation matrix by employing a modified alternative projection algorithm. Our novel techniques guarantee to extract a valid spatial correlation function and matrix from measurement data, even if those measurements are affected by unavoidable random noises. Experiment results, obtained from data generated by a Monte-Carlo model, confirm the accuracy and robustness of our techniques; and show that we are able to recover the correlation function and matrix with very high accuracy even in the presence of significant random noises.

3.1 Introduction

It is of importance to characterize process variation, because that information is essential for any attempts to analyze or optimize designs statistically. For example, it is necessary to know the variations of device parameters in order to build the statistical delay models for both devices and interconnects, which are the inputs for both SSTA and robust circuit tuning. Recent SSTA techniques considering spatially correlated parameters [16, 1, 114], however, assume that the required spatial correlation information given as a correlation matrix is known *a priori* and is always valid, i.e., the spatial correlation matrix is always positive semidefinite. In fact, the only way to obtain these variation characteristics is to extract them from silicon measurements. Because of unavoidable measurement errors, there is no guarantee that the so-obtained correlation coefficients can form a valid correlation matrix.

To the best of our knowledge, no existing work has provided a detailed technique to extract that information properly from measurements except some preliminary exploration in [49, 67, 86]. The extraction of the deterministic component of L_{eff} variation was considered in detail in [75] for the 0.18μm CMOS technology. But that publication ignored the random component of spatial variations, justifying its approach by the fact that for the 0.18μm CMOS technology random variations were not significant. Another two recent publications [29, 32] limited their consideration by simple computation of the spatial correlation coefficient that is a function of distance, which is either a linear [29] or piece-wise linear function [32]. There is no verification, however, that the extracted correlation function was a valid correlation function, i.e., any correlation matrix generated from this function must be positive semidefinite. In fact, theoretically, as we will shown in this chapter, neither linear nor piece-wise linear functions are valid spatial correlation functions.

The major contribution of this work is as follows. We provide the theoretical foundations for extracting the valid spatial correlation information from silicon measurements. We develop a robust technique to extract a valid spatial correlation function by solving a constrained nonlinear optimization problem. We also develop a robust technique to extract a valid spatial correlation matrix by employing a modified alternative projection algorithm. Our techniques are based upon the mathematical theories of random fields and convex analysis, and it is guaranteed that the resulting correlation function and correlation matrix are not only valid, but also the closest ones to the underlying model even if the data are distorted by significant measurement noises. Experiment results based upon a Monte-Carlo model confirm the accuracy and robustness of our techniques. We achieve less than 10% errors for the extracted process variations even if the measurement noise is more than 100% of the total process variations. Because of the promising results, we plan to apply our techniques to real wafer data to extract the spatial correlation information in the future. A preliminary version of this chapter was presented at ISPD 2006 [107].

3.2 Preliminary

3.2.1 Process Variation Classification

There are two orthogonal ways to classify process variations. The first one is to classify the variations according to the scope of their occurrence [30, 70, 10] as follows

- *Die-to-die variation*, which is also called inter-die variation or between-die variation, describes the variation that affect parameters in different dies differently, but affect parameters within a die equally.

- *Within-die variation*, which is also called intra-die variation, across-chip variation, on-chip variation, *spatial variation*, or *spatial correlation*[1], describes the variation that affects process parameters at different locations of the same die differently.

Or, according to the scale of their causes, process variations can also be classified into the following two categories [70, 72].

- *Systematic variation* describes the deterministic portion of the variation. The physical cause of this variation is usually those identifiable deterministic features or patterns.

- *Random variation* describes the variation that is independent of any other conditions. The physical cause of this variation is usually not well understood, and thus it behaves more like a stochastic process.

Both systematic and random variations need to be considered to accurately model the impact of process variations on designs. For example, the systematic transistor channel length variation can be more than 50% of its overall variations, and its root of causes includes through-pitch variation due to proximity (pitch) effects, through-process variation due to defocus condition, topography variation, mask variation, and etching [36]. Because systematic variations usually can be modeled accurately once a circuit's physical layout is known [17, 75]; it can be corrected via techniques such as optical proximity correction (OPC), which post-processes mask data so that the distortion of printed image caused by proximity environment of the designed shapes can be reduced [46]. In contrast, random

[1]Spatial variation or spatial correlation is used more frequently in recent literatures than others.

variations are more like a stochastic process, and there is no clear trend or pattern to be predicted. It is the random process variation that permits us to treat designs statistically, including statistical timing analysis and optimization [16, 94].

We denote F as the measurable process parameter of interest, which can be either a physical parameter, like channel length, channel width, silicon oxide thickness, and wire thickness, or a parametric quantity, such as gate delay and threshold voltage[2]. Because of manufacturing process variations, these process parameters are no longer fixed values. We model the parameter as a random variable, which is a complicated function of die-to-die (D2D) systematic and random variations, and within-die (WID) systematic and random variations. Conceptually, we can represent it as

$$F = h(Z_{D2D,sys}, Z_{D2D,rnd}, Z_{WID,sys}, Z_{WID,rnd}), \tag{3.1}$$

where $Z_{D2D,sys}$ models the die-to-die systematic variation; $Z_{D2D,rnd}$ models the die-to-die random variation; $Z_{WID,sys}$ models the within-die systematic variation; and $Z_{WID,rnd}$ models the within-die random variation. All variation components are further complicated functions of the manufacturing process, feature's relative location in the wafer, feature's relative location in the die, and feature's local geometry patterns, to name just a few.

[2] Without loss of generality, we use one generic process parameter F in the following discussion. But it is understood that the same techniques to be presented can be easily extended to multiple process parameters.

3.2.2 Process Variation Decomposition

Assuming the impact of each variation component is linear, we write (3.1) as follows

$$\begin{aligned} F &= h_0 + h_1(Z_{D2D,sys}) + h_2(Z_{D2D,rnd}) \\ &\quad + h_3(Z_{WID,sys}) + h_4(Z_{WID,rnd}) + X_r, \end{aligned} \tag{3.2}$$

where h_0 is a function that models the nominal value of F under nominal manufacturing conditions without any variation; h_1, h_2, h_3, and h_4 are functions that model the impact of respective variation component (i.e., $Z_{D2D,sys}$, $Z_{D2D,rnd}$, $Z_{WID,sys}$, and $Z_{WID,rnd}$ on F; and X_r is a residual part that models the purely independent random variation that is not explainable by other variation components. The sum of h_1 and h_2 reflects the fluctuation of F caused by *die-to-die variation*; and the sum of h_3 and h_4 reflects the fluctuation of F caused by *within-die variation*. The sum of h_1 and h_3 reflects the fluctuation of F caused by *systematic variation*; and the sum of h_2 and h_4 reflects the fluctuation of F caused by *random variation*.

$$\begin{aligned} F_s &= h_1(Z_{D2D,sys}) + h_3(Z_{WID,sys}), \\ F_r &= h_2(Z_{D2D,rnd}) + h_4(Z_{WID,rnd}) + X_r, \end{aligned}$$

where F_s models the systematic variation of F; while F_r is a zero-mean random variable that models the random variation of F. Hence, we have

$$F = h_0 + F_s + F_r. \tag{3.3}$$

The variance of F, σ_F^2, is also called the *overall chip variance*.

3.3 Spatial Correlation Modeling and Problem Formulations

It has been observed that devices that are physically close to each other are more likely to have similar characteristics than devices that are far apart. This phenomenon is captured by the modeling of *spatial correlation*. In the following, we introduce two ways to model the spatial correlation, each of which has its own value and applies to different process variation scenarios.

3.3.1 Grid-based Model of Spatial Correlation

3.3.1.1 Modeling

In this model, a set of grid cells is super-imposed on top of the chip area as shown in Fig. 3.1. It is assumed that the difference between process parameters only occurs for process parameters at different grid cells; and all process parameters within the same grid cell will have the same characteristics. In another word, the spatial correlation for process parameters within one grid cell is always one, and it is only interesting to know the spatial correlation between process parameters at different grid cells.

The grid-based spatial correlation model can be adapted to handle more complicated variation scenarios by varying the number, size, and shape of grid cells. For example, it is believed that process control at chip center area is better than at chip boundaries, hence spatial correlation at the chip center area is more uniform than at the boundaries. In this case, we can apply the griding scheme as shown in Fig. 3.1(b), where the center grid cells are coarser and less, while the boundary grid cells are finer and more. Hence the grid-based model can easily capture the non-uniform spatial correlation phenomena across the whole chip. Moreover, the

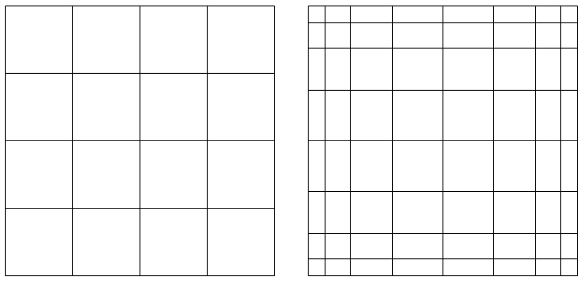


Figure 3.1: Grid-based spatial correlation models. (a) Uniform grids. (b) Non-uniform grids.

shape of grids can be also non-rectangular, and for different process parameters, the griding schemes may also be different[3].

If we associate every grid cell i in the chip area with a random variable F_i and denote its variance as $\sigma_{F_i}^2$, then for the parameter of interests at two different grid cells i and j, the *overall covariance* between them is given by

$$cov(F_i, F_j) \equiv \rho_{i,j} \cdot \sigma_{F_i} \cdot \sigma_{F_j}, \tag{3.4}$$

where $\rho_{i,j}$ is the *overall process correlation* between process parameters at grid cell i and j.

For M number of chosen grid cells on the chip, we assume the joint spatial variation $\mathbf{F} = (F_1, F_2, ..., F_M)^T$ follows a multivariate Gaussian process with respect to their respective physical locations on the chip. To fully characterize the M-dimensional Gaussian distribution, we need to know the variance $\sigma_{F_i}^2$ for all

[3] The optimal way of griding, including grid numbers, grid shapes, and grid sizes, can be decided under the guidance of good knowledge of manufacturing process, which is not addressed in this chapter.

grid cells and their corresponding correlation matrix $\mathbf{\Omega}$ as shown in (3.5)

$$\mathbf{\Omega} = \begin{bmatrix} 1 & \rho_{1,2} & \rho_{1,3} & \cdots & \rho_{1,M} \\ \rho_{1,2} & 1 & \rho_{2,3} & \cdots & \rho_{2,M} \\ \rho_{1,3} & \rho_{2,3} & 1 & \dots & \rho_{3,M} \\ \cdot & & & & \cdot \\ \rho_{1,M} & \rho_{2,M} & \rho_{3,M} & \cdots & 1 \end{bmatrix}. \tag{3.5}$$

A *valid* correlation matrix must be *positive semidefinite* by its definition [43].

3.3.1.2 Problem Formulation

Based on the grid-spatial correlation model, we propose the following problem formulation:

Formulation 1 Extraction of Spatial Correlation Matrix*: Given M number of grid cells on a chip to model the spatial correlation, and noisy measurement data for the parameter of interest at these grid cells, extract the overall process variation at very grid cell $\sigma_{F_i}^2$, and their corresponding spatial correlation matrix Ω as shown in (3.5), so that it not only accurately captures the underlying process variation model, but the extracted correlation matrix Ω is always positive semidefinite.*

Extracting a valid correlation matrix is of practical significance. For example, SSTA tools such as [16], which are based upon principle component analysis, require that the spatial correlation matrix must be valid and known *a priori*.

Even though the grid-based spatial correlation model is intuitively simple and easy to use, it has its own limitations. The foremost one is the inherent accuracy-versus-efficiency issue because of its fundamental assumption, which states that all parameters of interests within one grid cell have the same characteristics. To justify such an assumption, the size of each grid cell can not be too large, which in turn increases the total number of grid cells required for modeling. From extraction point of view, on the one hand, the more number of grid cells, the more number of measurement sites within each chip, hence the more expensive to extract such a model. On the other hand, the physical limitation of measurement devices also prevents the grid cell size being too small, otherwise, the measurement probe would not fit into one grid cell. Because of these inherent limitations of the grid-based modeling approach, in the next section, we propose a more flexible approach to model spatial correlation.

3.3.2 Gridless-based Model of Spatial Correlation

3.3.2.1 Modeling

Because the systematic variation is more like a deterministic variation [70], we lump it with the nominal value h_0, i.e.,

$$f_0 = h_0 + F_s, \tag{3.6}$$

where f_0 is the mean value of F with the systematic variation considered. The extraction of mean value f_0 is relatively easy, and is essentially done through averaging. For example, [75] has presented a methodology to extract the intra-die systematic variation of critical dimension (CD or effective channel length) by averaging out measurement of CD at different locations of the same die.

In the following, we mainly concern ourselves in extracting the random variation parts F_r. This is a more challenging task, because simply taking averaging of measurements would not give us any useful information on the zero-mean random process variations' characteristics. Towards this end, we rewrite the random variation F_r as follows:

$$F_r \quad = \quad X_g + X_s + X_r$$

where X_g models the *inter-chip global variation* that affects all features within the same chip equally, but is different among different chips obtained from different lots, wafers, or even the same wafer. In another word, $X_g = h_2(Z_{D2D,rnd})$. *Intra-chip spatial correlation* is modeled by X_s, which is different for features at different locations within the same chip. In another word, $X_s = h_4(Z_{WID,rnd})$. Therefore, we have

$$F = f_0 + F_r = f_0 + X_g + X_s + X_r. \tag{3.7}$$

The three types of random variations, X_g, X_s and X_r, are *independent* by definition. Hence the variance of F_r is given by

$$\sigma_{F_r}^2 \quad = \quad \sigma_G^2 + \sigma_S^2 + \sigma_R^2, \tag{3.8}$$

where σ_G^2, σ_S^2, and σ_R^2 are the variances of X_g, X_s, and X_r, respectively. When the systematic variation is excluded, the *overall chip variance* is equivalent to the random variance, i.e., $\sigma_F^2 = \sigma_{F_r}^2$.

We model the random part of process variation F_r as a *homogeneous and isotropic* rand filed, whose formal definition is introduced as follows:

Definition 1 Random Field *is a real random function $F(x, y)$ of position (x, y) in the 2-dimensional space $\mathcal{R}^2$.*

Definition 2 Homogeneous and Isotropic Random Field *is a random field $F(x, y)$ whose mean and variance are constants, and whose correlation function $\rho(x_i, x_j, y_i, y_j)$ between any two points depends only on the distance v between them, i.e.,*

$$\rho(x_i, x_j, y_i, y_j) = \rho(v_{i,j}), \tag{3.9}$$

where $v_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$.

If the spatial variation follows a homogeneous and isotropic random field, then the same distance $v_{i,j}$ always corresponds to the same $\rho(v_{i,j})$, regardless of their locations. Therefore, for simplicity, we denote $\rho(v_{i,j})$ as $\rho(v)$ in the following whenever there is no ambiguity.

Note that the overall process variation, F as shown in (3.3) that includes both systematic variation and random variation, does not necessarily follow a homogeneous and isotropic random field. But if we only look at the random variation part, then the physical properties of the random variation part F_r would be very likely to follow a homogeneous and isotropic random field, particularly when the manufacturing process becomes mature and stable.

3.3.2.2 Valid Spatial Correlation Function

Formally, a *valid spatial correlation function* $\rho(v)$ is a function such that the correlation matrix generated from $\rho(v)$ for arbitrary number of points on the two-dimensional space is always *positive semidefinite*.

In its simplest way, a valid spatial correlation function should satisfy the following necessary but not sufficient conditions[4]:

$$\rho(0) = 1, \tag{3.10}$$

$$0 \le \rho(v) \le 1, \tag{3.11}$$

$$\rho'(v) \le 0. \tag{3.12}$$

Equations (3.10) and (3.11) are required by the definition of correlation coefficient [43]. The interpretation of (3.12) is that the spatial correlation is a monotonically

[4] In the context of process variation, we are only interested in the spatial correlation that is non-negative.

decreasing function of distance, i.e., as devices become further apart, the correlation between them becomes smaller. The *correlation distance* $\overline{v}$ is the distance beyond which the spatial correlation $\rho(v)$ becomes sufficient small and can be approximated as zero, i.e., $\rho(v) \approx 0$ for all $v \geq \overline{v}$. For simplicity, in the following when we describe the correlation function, we only give the function form for any $v \in [0, \overline{v}]$ whenever there is no ambiguity. For example, Fig. 3.2 shows a piece-wise monotonically decreasing function, and the function form is $\rho(v) = -v^2 + 1$ for any $v \in [0, \overline{v}]$ with $\overline{v} = 1$.

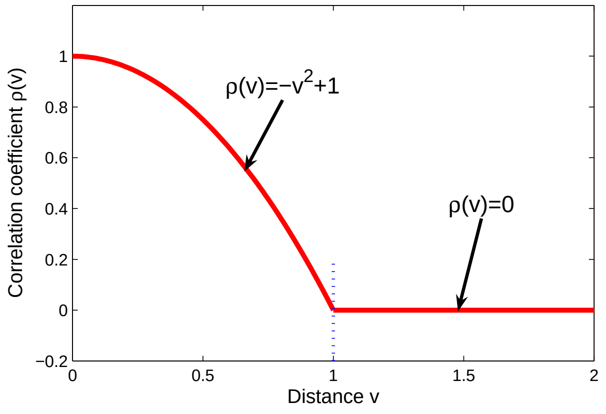


Figure 3.2: A monotonically decreasing function that is not a valid spatial correlation function.

Contrary to the common wisdom, we show that not all monotonically decreasing functions qualify for the spatial correlation function. For example, for the function as shown in Fig. 3.2, if we assume $\rho(v)$ is a valid spatial correlation function, then the correlation matrix $\mathbf{\Omega}$ between any three points on the die as shown in Fig. 3.3 can be built as follows

$$\Omega = \begin{bmatrix} 1 & \rho(d_1) & \rho(d_3) \\ \rho(d_1) & 1 & \rho(d_2) \\ \rho(d_3) & \rho(d_2) & 1 \end{bmatrix}, \tag{3.13}$$

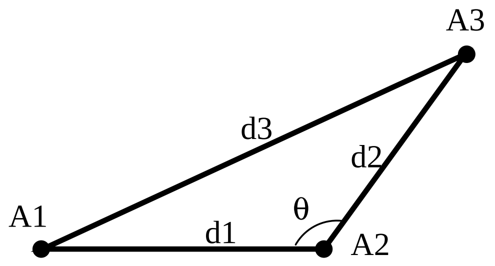


Figure 3.3: Any three points on the die.

If $d_1 = 31/32$, $d_2 = 1/2$, and $d_3 = 1/2$, under $\rho(v) = -v^2 + 1$, we obtain a correlation matrix[5]

$$\Omega = \begin{bmatrix} 1 & 0.0615 & 0.75 \\ 0.0615 & 1 & 0.75 \\ 0.75 & 0.75 & 1 \end{bmatrix}. \tag{3.14}$$

However, it is easy to show that the resulting matrix is not a *valid* correlation matrix, as the smallest eigenvalue of this matrix is -0.0303, implying that this matrix is not positive semidefinite!

Recall that a matrix is a positive semidefinite matrix if and only if its every *principal sub-matrix* has a non-negative determinant, whereas the *principal sub-matrices* are formed by removing row-column pairs from the original symmetric matrix [66]. Therefore, in order for Ω in (3.13) to be a positive semidefinite

[5]The three points form a triangle, which imposes constraints on the possible choices of d_1, d_2 and d_3, i.e., $d_1 + d_2 > d_3$, $d_2 + d_3 > d_1$, and $d_1 + d_3 > d_2$.

matrix, we require the following two equations to hold

$$1 - \rho(d_1)^2 \geq 0, \tag{3.15}$$

$$1 + 2\rho(d_1)\rho(d_2)\rho(d_3) \geq \rho(d_1)^2 + \rho(d_2)^2 + \rho(d_3)^2. \tag{3.16}$$

Equation (3.15) is automatically satisfied following (3.11). Therefore, we only need to check (3.16). Plugging the values of $d_1 = 31/32$, $d_2 = 1/2$, and $d_3 = 1/2$ with $\rho(v) = -v^2 + 1$ into (3.16), we can see that it violates the constraints of (3.16). This explains why function $\rho(v) = -v^2 + 1$ is not a valid spatial correlation function.

This leads us to the question of "what type of monotonic decreasing functions qualify to be a valid spatial correlation function?" To answer this question, we introduce the following theorem:

Theorem 1 *A necessary and sufficient condition for the function $\rho(v)$ to be a valid spatial correlation function of a homogeneous and isotropic random field is that it can be represented in the form of*

$$\rho(v) = \int_0^\infty J_0(\omega v) d(\Phi(\omega)), \tag{3.17}$$

where $J_0(t)$ is the Bessel function of order zero and $\Phi(\omega)$ is a real nondecreasing function on $[0, \infty)$ such that for some non-negative p,

$$\int_0^\infty \frac{d\Phi(\omega)}{(1+\omega^2)^p} < \infty. \tag{3.18}$$

Proof: See [110] for the proof. □

Based on the above theorem, we derive the following two corollaries:

Corollary 1 *The monotonically decreasing* exponential *function (3.19) and double* exponential *function (3.20), i.e.,*

$$\begin{aligned} \rho(v) &= exp(-bv), & (3.19) \\ \rho(v) &= exp(-b^2v^2), & (3.20) \end{aligned}$$

are valid spatial correlation functions. The constant b is a parameter that regulates the decaying rate of the correlation function with respect to distance v. The correlation distance for the two functions are infinity, i.e., $\overline{v} = \infty$.

Proof: By way of construction, we find that $\Phi(\omega)$ =1-$\frac{1}{\sqrt{1+\omega^2/b^2}}$ and $\Phi(\omega)$=1-$exp(-\frac{\omega^2}{4b^2})$ satisfy the conditions as specified in Theorem 1. Plugging them into (3.17), we obtain the corresponding correlation functions as (3.19) and (3.20), respectively. In another word, the exponential function (3.19) and double exponential function (3.20) are valid spatial correlation functions [12]. □

Corollary 2 *The monotonically decreasing linear function in the form of*

$$\rho(v) = -av + b, \forall v \in [0, \overline{v}], \tag{3.21}$$

with $\overline{v} \leq b/a$ is not a valid spatial correlation function, where a and b are two positive numbers.

Proof: To prove that (3.21) is not a valid spatial correlation function, all we need to do is to find a counter example by which a correlation matrix generated from it is not positive semidefinite. One such counter example was given in [5], which convincingly shows that the monotonically decreasing linear function as shown in (3.21) is not a valid spatial correlation function. □

The implication of corollary 2 is interesting to note, because intuitively people may think that the monotonically decreasing linear function is valid for spatial correlation modeling, and the work as shown in [29] did apply it to real wafer data. But the corollary 2 tells us that such a practice is not correct.

In general, it is difficult to check whether an arbitrary function form is a valid correlation function [5]. For example, for an arbitrary piece-wise linear function, i.e, with arbitrary number of linear segments and each with arbitrary slopes, we cannot provide (and fail to find) any theoretical proof whether or not it is valid. But we at least can say for sure that not all piece-wise linear function is valid, because as shown in [5] for some particular piece-wise linear function, the spatial correlation matrix generated from it is not valid. In the work of [32], the authors proposed to use a piece-wise linear function to model the spatial correlation function. However, there is no guarantee that the so-obtained piece-wise linear function is valid. The authors of [32] also did not provide any theoretical justification for their approach.

3.3.2.3 Problem Formulation

When the spatial variation follows a homogeneous and isotropic random field, we propose the following first problem formulation:

Formulation 2 Extraction of Spatial Correlation Function: *Given noisy measurement data for the parameter of interest with possible inconsistency, extract the inter-chip global variation component σ_G^2, the intra-chip spatial variation component σ_S^2, the random variation component σ_R^2, and the spatial correlation function $\rho(v)$, so that the extracted variation components accurately capture the underlying variation model, and the spatial correlation function is always a valid correlation function satisfying condition (3.17).*

If the spatial variation is modeled as a *homogeneous and isotropic random field* in a two dimensional space $\mathcal{R}^2$, then for the parameter of interest at *arbitrary* two different points, their covariance is

$$\begin{aligned} cov(F_i, F_j) &= cov(X_g, X_g) + cov(X_{s,i}, X_{s,j}) && (3.22)\\ &= \sigma_G^2 + \rho(v)\sigma_S^2, && (3.23)\end{aligned}$$

where $\rho(v)$ is the spatial correlation coefficient between two locations that are v distance apart. In another word, we can characterize the process variation by extracting the inter-chip global variation σ_G^2, intra-chip spatial variation σ_S^2, and the correlation function $\rho(v)$.

For the parameter of interest at two different locations with distance of v, the *overall process correlation* between them is thus given by

$$\rho_v \equiv \frac{cov(F_i, F_j)}{\sigma_{F_i}\sigma_{F_j}} \tag{3.24}$$

$$= \frac{\sigma_G^2 + \rho(v)\sigma_S^2}{\sigma_G^2 + \sigma_S^2 + \sigma_R^2}. \tag{3.25}$$

Because the spatial correlation $\rho(v)$ is a function of the distance v, so is the overall process correlation ρ_v. As $\rho(v)$ is homogeneous and isotropic, so is ρ_v. Because of the one-to-one correspondence between spatial correlation $\rho(v)$ and the overall process correlation ρ_v, extracting the spatial correlation function $\rho(v)$ is equivalent to extracting the overall process correlation function ρ_v.

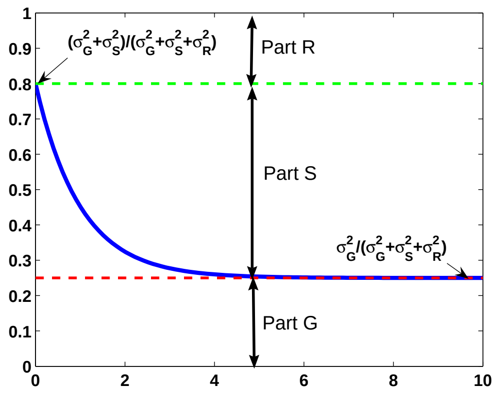


Figure 3.4: A possible curve for the overall process correlation according to (3.25) in the absence of measurement noise.

In Fig. 3.4, we show a possible curve for the overall correlation ρ_v as a function of the distance v as given by (3.25). According to Fig. 3.4, the total correlation can be divided into three parts: part G is the correlation caused by the inter-chip global variation; part S is the correlation caused by the intra-chip spatial correlation; and part R is caused by the purely uncorrelated random variation. We can see that the overall process correlation ρ_v starts to settle at a constant value when the distance becomes large enough (greater than the *correlation distance* $\overline{v}$), which means that even for devices from the same chip that are far apart, there is still some correlation between them due to their shared global variations. We can also see that there is a sudden drop from one for ρ_v at distance zero. The cause for that drop is the purely uncorrelated random variation, such that even for devices that are very close to each other, they are still not perfectly correlated. Perfect correlation ($\rho_v = 1$) only occurs when the two devices are in fact the same device[6].

3.3.3 Extraction Setup

To experimentally characterize the process variation, we obtain N samples of a chip, and choose M number of sites on each chip where measurement is conducted. The sites are denoted as (x_i, y_i), and the distance between any two sites is denoted as $v_{i,j}$. We denote each measurement of the parameter of interest F as $f_{k,i}$ for the k^{th} chip on the i^{th} site.

Note that in order to obtain these measurement data, it usually requires careful design of test structures, placement of test structures spanning a range of

[6] Note that a similar plot showing the trend of the overall process correlation with respect to distances has also been empirically observed in [32] based on wafer-scale measurements. But the authors of [32] did not provide a theoretical explanation of this phenomena as we do in this chapter.

areas on the die, and measurement procedures to collect data. Those details are beyond the scope of this chapter, and interested readers are referred to [72] for more information.

In the following, we present techniques to solve the above two problem formulations as discussed in section 3.3.1 and 3.3.2, respectively. We first solve the extraction of spatial correlation function in section 3.4, then solve the extraction of spatial correlation matrix in section 3.5.

3.4 Extraction of Valid Spatial Correlation Function

3.4.1 Global Variation Extraction

We treat each measurement of the parameter of interest F as a sampling of the quantity in (3.7). Given N samples of a chip and M number of measurement sites on each chip, we group the measured data $f_{k,i}$ by their chip locations as follows: $f_{k,.}$=$[f_{k,1}, ..., f_{k,M}]$ for k=1 to N, or by their site locations as follows: $f_{.,i}$=$[f_{1,i}, ..., f_{N,i}]$ for i=1 to M. For better presentation, we denote the actual variance as σ^2 with an upper case letter in subscript, like σ_G^2 for the global variation component; and denote the extracted variance as σ^2 with a lower case letter in subscript, like σ_g^2 for the extracted global variation component.

We approximate the overall chip variance σ_F^2 by computing the *unbiased sam-*

ple variance [43] of $f_{k,i}$ as follows[7]

$$\sigma_F^2 \approx \sigma_f^2 = \frac{1}{MN-1}\left(\sum_i \sum_k f_{k,i}^2 - \frac{(\sum_i \sum_k f_{k,i})^2}{MN}\right). \tag{3.26}$$

For all samples of the parameter of interest F within a particular chip c, because the inter-chip global variation X_g changes the value of parameter for all samples with the same chip by the same amount, the *overall within-chip variance* is thus given by

$$\sigma_{F_c}^2 = \sigma_S^2 + \sigma_R^2. \tag{3.27}$$

We estimate the overall within-chip variation by computing the *unbiased sample variance* [43] of $f_{k,\cdot}$ as follows

$$\sigma_{F_c}^2 \approx \sigma_{f_k}^2 = \frac{1}{M-1}\left(\sum_i f_{k,i}^2 - \frac{(\sum_i f_{k,i})^2}{M}\right). \tag{3.28}$$

For different $f_{k,\cdot}$, we may get different estimation of $\sigma_{F_c}^2$ caused by inconsistent measurement. To improve the accuracy, we estimate the overall within chip variance by taking the average value of $\sigma_{f_k}^2$. We denote the resulting average value as $\sigma_{f_c}^2 \approx \sigma_{F_c}^2$.

Knowing the estimation of the overall chip variance σ_f^2 and the overall within-chip variance $\sigma_{f_c}^2$, we extract the inter-chip global variation by

$$\sigma_G^2 = \sigma_F^2 - \sigma_{F_c}^2 \approx \sigma_g^2 = \sigma_f^2 - \sigma_{f_c}^2. \tag{3.29}$$

[7] The advantage of using the unbiased sample variance over sample variance is that it will not over- or under-estimate the true quantity. In practice, the true or exact variance of a population is not known a priori and has to be computed based on samples. Unbiased sample variance is good at estimating the true variance in this case. In contrast, the sample variance merely measures the variance for the given *finite* number of samples, hence it is a biased estimator of the true variance. For more information about unbiased sample variance and variance, please refer to [43].

3.4.2 Spatial Correlation Extraction

For any two different sets of $f_{.,i}$ and $f_{.,j}$ at two different sites that are v distance apart, we estimate the covariance of F_i and F_j by computing the *unbiased sample covariance* [43] of $f_{.,i}$ and $f_{.,j}$ as follows

$$\begin{aligned} cov(F_i, F_j) &\approx cov(f_{.,i}, f_{.,j}) && (3.30)\\ &= \frac{\sum_k f_{k,i} f_{k,j}}{N-1} - \frac{\sum_k f_{k,i} \sum_k f_{k,j}}{N(N-1)}. && (3.31) \end{aligned}$$

For simplicity, we also denote $cov(f_{.,i}, f_{.,j})$ as $cov(v)$ to show that it is a function of two points that are v distance apart.

According to (3.23) and (3.29), we estimate the product of spatial variation σ_S^2 and spatial correlation $\rho(v)$ as follows:

$$\sigma_S^2 \cdot \rho(v) = cov(F_i, F_j) - \sigma_G^2 \approx cov(v) - \sigma_g^2. \tag{3.32}$$

Because $\rho(v)$ is a function of v, we need to compute $\rho(v)$ for different pairs of sites with different distances in order to obtain the full description of $\rho(v)$. But there are two challenges in doing that: (1) we do not know the exact value of spatial variation σ_S^2; (2) because of unavoidable measurement errors, the data set computed as above may not be consistent. Therefore, in the following, we propose a robust technique to find the spatial correlation function $\rho(v)$ and σ_S^2 accurately. Moreover, the resulting $\rho(v)$ is guaranteed to be a valid spatial correlation function.

Given the data set (v, $cov(v)$) as computed from (3.31), we formulate the robust spatial variation extraction problem as the following optimization problem:

$$\begin{aligned} \min_{\Phi, \sigma_s^2} : \quad & \| \, \sigma_s^2 \int_0^\infty J_0(\omega v) d(\Phi(\omega)) - cov(v) + \sigma_g^2 \, \|, && (3.33)\\ s.t. \quad & \sigma_s^2 \le \sigma_{f_c}^2, \\ & \int_0^\infty \frac{d\Phi(\omega)}{(1+\omega^2)^p} < \infty. \end{aligned}$$

In other words, we find a valid spatial correlation function by solving a constrained nonlinear optimization problem, so that the resulting spatial correlation function minimizes the total error with respect to measurement data. After obtaining $\Phi(\omega)$, we plug it into (3.17) to obtain the valid spatial correlation function $\rho(v)$.

The above problem formulation is very general and applies to any real non-decreasing function $\Phi(\omega)$. For practical use, however, there is no need to enumerate all possible choices of $\Phi(\omega)$ in order to find the optimal $\rho(v)$. Moreover, as we have discussed in section 3.3.2.2, it is also difficult to check the validity of an arbitrary spatial correlation function.

Therefore, to make the above problem tractable, we can approximate the experimentally measured correlation function with a function selected from a family of functions that are proved to be valid spatial correlation functions. To serve such a purpose, it is sufficient to chose a family of functions $\Phi(\omega)$ so that the $\rho(v)$ obtained from (3.17) contains a rich set of functions for the purpose of modeling spatial correlation.

It has been shown in [12] that by choosing a proper family function of $\Phi(\omega)$, we obtain a very general family of spatial correlation functions

$$\rho(v) = 2\left(\frac{bv}{2}\right)^{s-1} K_{s-1}(bv)\Gamma(s-1)^{-1}, \tag{3.34}$$

where K is the modified Bessel function of the second kind, Γ is the gamma function, and b and s are two real parameter numbers that regulate the shape of the function. By varying b and s, we obtain different spatial correlation functions. For example, the exponential function as shown in (3.17) can be generated from (3.34) by choosing $s = 3/2$.

To show that the function of (3.34) indeed provides us a rich set of correlation functions that suffice for our spatial correlation modeling, we plot the function of (3.34) under different parameters of b and s. Fig. 3.5 shows a few samples of correlation functions generated from (3.34) by setting b to be 0.1, 1 and 10, and varying s from 2 to 10 with a step size of 2. From the figure, we see that the correlation function (3.34) indeed can generate a rich gamut of correlation functions for the purpose of spatial correlation modeling.

Without loss of generality, in the following, equation (3.34) will be used as the candidate[8] correlation function in (3.33). Moreover, 2-norm is used as a measure of the objective function in (3.33). Therefore, we rewrite the optimization problem as given in (3.33) as follows

$$\begin{aligned} \min_{b,s,\sigma_s^2} : \quad & \sum [2\sigma_s^2 (\tfrac{bv}{2})^{s-1} K_{s-1}(bv)\Gamma(s-1)^{-1} - cov(v) + \sigma_g^2]^2, \\ s.t. \quad & \sigma_s^2 \le \sigma_{f_c}^2. \end{aligned} \tag{3.35}$$

This is a constrained nonlinear least square problem, and we can solve it efficiently via any nonlinear least square technique [22]. Note that problem (3.35) is not a convex problem in general, hence we cannot guarantee to find a global optimal solution. But as this kind of least-square minimization problem is well-studied in the literature, good solvers are available to find a solution with high quality. Our experimental results to be presented also confirms this argument. Moreover, as all nonlinear minimization engines are sensitive to the initial guess, obtaining a high quality solution sometimes may require to try different initial guesses.

After solving the above problem, we obtain the estimated spatial variation component $\sigma_S^2 \approx \sigma_s^2$, and the parameter b and s. By plugging b and s into (3.34),

[8] Function (3.34) is chosen over the exponential (3.19) or double exponential (3.20) function as a candidate spatial correlation function in this work. The reason is that it has more parameters (b and s) and contains the exponential function as a special case (with $s = 3/2$). This gives us considerably more flexibility to fit the data but still with reasonable complexity. Apparently, other choices of candidate functions are possible. But we have to be careful in assuring that the candidate functions are valid spatial correlation function. As pointed out by [5], it is always simpler and safer to use those "approved" valid spatial correlation functions, as testing the validity of a arbitrary function form (such as linear and piece-wise linear) is almost always timing consuming and difficult.

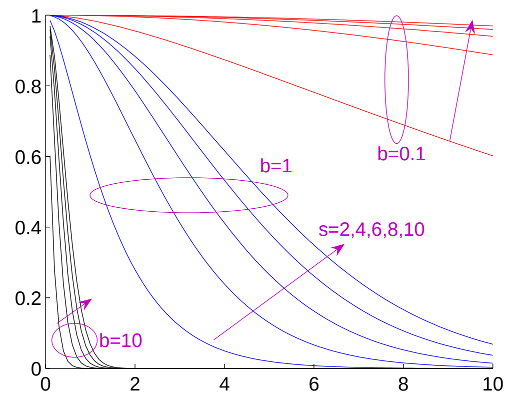


Figure 3.5: Correlation functions generated from (3.34).

we obtain the estimated spatial correlation function $\overline{\rho(v)} \approx \rho(v)$. Therefore, we have obtained all information about the spatial variation component: both the variance of spatial variation and the spatial correlation function.

3.4.3 Overall Algorithm

The overall algorithm for characterizing the process variation is summarized as shown in Fig. 3.6:

```
1  Extract global variation $\sigma_g^2$ by (3.29);
2  Solve (3.35) to obtain $\sigma_s^2$ and $b$ and $s$;
3  Extract $\overline{\rho(v)}$ by plugging $b$ and $s$ into (3.34);
4  Extract random variation $\sigma_r^2$ by (3.36);
5  Extract overall process correlation by (3.25);
```

Figure 3.6: Algorithm for characterization of process variation.

We first extract the global variation component σ_g^2 by using formula (3.29). We then solve the nonlinear least square optimization problem as defined in (3.35) to obtain the spatial variation component σ_s^2, and the parameter of b and s that define the spatial correlation function for a homogeneous and isotropic random field as shown in (3.34). According to (3.8), we extract the random variation component by using the following formula:

$$\sigma_R^2 = \sigma_F^2 - \sigma_G^2 - \sigma_S^2 \approx \sigma_r^2 = \sigma_f^2 - \sigma_g^2 - \sigma_s^2. \tag{3.36}$$

By plugging all variation components into (3.25), we obtain the overall process correlation at any distance.

3.5 Extraction of Spatial Correlation Matrix

3.5.1 Overall Algorithm

We are given measurement data for M points of interest on a chip and we have N samples of the same chip. We extract the overall process spatial correlation as follows.

We first estimate the covariance between any two points of interest by (3.31). We then estimate the variance of each point of interest by computing its *unbiased sample variance* [43] as follows:

$$\sigma_{F_i}^2 \approx \sigma_{f_i}^2 = \frac{1}{N-1}\left(\sum_k f_{k,i}^2 - \frac{(\sum_k f_{k,i})^2}{N}\right). \tag{3.37}$$

By plugging the estimated $\sigma_{f_i}^2$ and $\sigma_{f_j}^2$ and $cov(f_{.,i}, f_{.,j})$ from (3.31) into (3.24), we obtain the estimated overall process correlation coefficient

$$\rho_{i,j} = \frac{cov(F_i, F_j)}{\sigma_{F_i}\sigma_{F_j}} \approx \frac{cov(f_{.,i}, f_{.,j})}{\sigma_{f_i}\sigma_{f_j}}. \tag{3.38}$$

For the given M points of interest, we have $M(M-1)/2$ number of pairs of points F_i and F_j and the corresponding $M(M-1)/2$ number of estimated correlation coefficients $\rho_{i,j}$. Putting all $\rho_{i,j}$ into (3.5), we obtain the estimated overall process spatial correlation matrix $A \approx \Omega$.

Note that in order for the above estimated A to qualify for a correlation matrix, it has to be a *positive semidefinite* matrix. But we can not guarantee that such a property would hold automatically for the resulting A due to the unreliable (sometimes even inconsistent) measurement data. We solve this problem by employing the *modified alternative projection algorithm* to be presented in the next section to robustly extract a valid correlation matrix Ω from the unreliable measurement data.

The overall algorithm for extracting a valid spatial correlation matrix is summarized as follows in Fig. 3.7:

```
1   Compute $cov(f_i, f_j)$ by (3.31);
2   Compute $\sigma^2_{f_i}$ by (3.37);
3   Compute $\rho_{i,j}$ by (3.38);
4   Compute $A$ by assembling $\rho_{i,j}$ into (3.5);
5   Compute $\Omega$ via the modified alternative projection algorithm;
```

Figure 3.7: Algorithm for extracting a valid spatial correlation matrix.

3.5.2 Modified Alternative Projection Algorithm

The robust extraction of a consistent correlation matrix problem can be formulated as the following optimization problem. For a given symmetrical matrix A with elements $a_{i,j}$ between 0 and 1, find a correlation matrix Ω that is mostly close to A. Mathematically, the closeness can be measured via the distance between two matrices, i.e.,

$$\min_{\Omega} : \quad \| A - \Omega \|, \tag{3.39}$$

$$s.t. : \quad \Omega \in correlation\ matrix. \tag{3.40}$$

We use the weighted Frobenius norm to measure the distance between two matrix. Recall that the Frobenius norm is defined as $\| A \|_F^2 = \sum a_{i,j}^2$. One of the weighted Frobenius norms is the W-norm as defined by

$$\| A \|_W = \| W^{1/2} A W^{1/2} \|_F, \tag{3.41}$$

where W is a symmetric positive definite matrix.

This problem is also called the nearest correlation matrix problem [41], or the least-squares covariance adjustment problem [11]. As a proof of concept, we solve this problem by employing the modified alternative projection algorithm proposed in [41]. The idea is to iteratively project the symmetric matrix A onto two convex sets alternatively, and at the end of iteration, the final projected matrix is the solution to the optimization problem as defined in (3.39).

We first define the sets

$$\begin{aligned} U &= \{Y = Y^T \in \mathbf{R}^{\mathbf{n}\times\mathbf{n}} : \mathbf{y_{ii}} = \mathbf{1}\}, && (3.42)\\ S &= \{Y = Y^T \in \mathbf{R}^{\mathbf{n}\times\mathbf{n}} : \mathbf{Y} \geq \mathbf{0}\}, && (3.43) \end{aligned}$$

where the notation $Y \geq 0$ means that Y is positive semidefinite. Our desired correlation matrix Ω as shown in (3.39) is a matrix that is in the intersection of U and S and has the shortest distance to A in a weighted Frobenius norm. Since S and U are both closed convex sets, so is their intersection. It thus follows from standard results in approximation theory that the minimum Ω in (3.39) is obtainable and is unique.

Moreover, for a symmetric matrix $A \in \mathbf{R}^{\mathbf{n}\times\mathbf{n}}$ with spectral decomposition (or eigen-value decomposition) $A = QDQ^T$, where $D = diag(\lambda_i)$ and Q is orthogonal, we introduce the following notations

$$A_+ = Q diag(max(\lambda_i, 0)) Q^T. \tag{3.44}$$

We denote $P_U(A)$ and $P_S(A)$ as the projections of A onto U and S, respectively. Then for a given W-norm, $P_U(A)$ can be computed analytically via the following formula.

$$P_U(A) = A - W^{-1} diag(\theta_i) W^{-1}, \tag{3.45}$$

where $\theta = [\theta_1, ..., \theta_n]^T$ is the solution of the linear system

$$(W^{-1} \circ W^{-1})\theta = diag(A - I), \tag{3.46}$$

where $\circ$ denotes the Hadamard product: $A \circ B$=$(a_{i,j} b_{i,j})$, i.e., element-wise matrix multiplication.

For a given W-norm, $P_S(A)$ can also be computed analytically via the following formula.

$$P_S(A) = W^{-1/2}((W^{1/2} A W^{1/2})_+) W^{-1}. \tag{3.47}$$

When the W-norm is taken as the identity I, i.e., the unweighted Frobenius norm, $P_U(A)$ is simply as

$$P_U(A) = (p_{ij}) \tag{3.48}$$

with $p_{ij} = a_{ij}$ for all $i \neq j$ and $p_{ij} = 1$ for all $i = j$. For $P_S(A)$, it is simply as

$$P_S(A) = A_+ = Q diag(max(\lambda_i, 0)) Q^T. \tag{3.49}$$

The following modified alternative projection algorithm as shown in Fig. 3.8 can be used to solve the nearest correlation matrix problem as defined in (3.39).

It has been proved that when $k \to \infty$, both X_k and Y_k converge to the desired correlation matrix Ω. Moreover, it has been theoretically shown that the convergence of the alternative projection algorithm is linear [41]. This conclusion has also been experimentally verified in section 3.6.2.

```
$\Delta S_0 = 0$, $Y_0$=$A$
for $k$=1,2,...
  $R_k = Y_{k-1}$ - $\Delta S_{k-1}$
  $X_k = P_S(R_k)$
  $\Delta S_k = X_k$ - $R_k$
  $Y_k = P_U(X_k)$
end
$\Omega = Y_k$
```

Figure 3.8: The modified alternative projection algorithm.

Among many possible choices, the following convergence condition can be used in Fig. 3.8 to stop the loop:

$$max \left\{ \frac{\| X_k - X_{k-1} \|}{\| X_k \|}, \frac{\| Y_k - Y_{k-1} \|}{\| Y_k \|}, \frac{\| Y_k - X_k \|}{\| Y_k \|} \right\} \leq \epsilon$$

where ϵ is a small tolerance number (say $\epsilon = 10^{-8}$).

3.6 Experiment Results

We employ a Monte Carlo model of measurement to verify the robustness and accuracy of our extraction algorithms in this chapter. One of the advantages of using Monte Carlo simulation is that it allows us to simulate different variation scenarios and measurement settings that are difficult to control in reality. By comparing the extracted variation components with the known variation components used in the Monte Carlo model, we can quantitatively examine how robust and how accurate our extraction algorithms are in the presence of different amount of measurement errors. Such a study is useful because it provides us the confidence in applying the algorithms to real wafter measurement.

3.6.1 Extraction of Valid Spatial Correlation Function

In this experiment, the Monte Carlo model is based on a valid correlation function $\rho(v)$ that follows a homogeneous and isotropic random field, but with different variation amounts for the three variation components (σ_G^2, σ_S^2, and σ_R^2). We simulate the measurement process by generating a set of measurement data from N number of sample chips and M number of measurement sites on each chip. To model the reality due to measurement error, we add a Gaussian noise with different variation amounts during the Monte Carlo sampling. By applying the algorithm as shown in Fig. 3.6, we extract the global variation component σ_g^2, random variation component σ_r^2, spatial variation component σ_s^2, and parameter of b and s that define the spatial correlation function $\overline{\rho(v)}$ for a homogeneous and isotropic random field as shown in (3.34). By plugging all variation components into (3.25), we obtain the overall process correlation at any distance. We measure the accuracy of our extraction algorithm for the global variation and spatial variation, but not the random variation as it is indistinguishable from the added measurement noise. For the global variation component, the relative error is given by

$$err(\sigma_G^2) = \frac{\sigma_g^2 - \sigma_G^2}{\sigma_G^2}. \tag{3.50}$$

For the spatial variation component, the relative error is given by

$$err(\sigma_S^2) = \frac{\sigma_s^2 - \sigma_S^2}{\sigma_S^2}. \tag{3.51}$$

And for the spatial correlation function, the relative error is given by

$$err(\rho(v)) = \frac{\parallel \overline{\rho(v)} - \rho(v) \parallel}{\parallel \rho(v) \parallel}. \tag{3.52}$$

From statistical theories, we know that if we have more measurement data, we have more confidence in the accuracy of statistics obtained from measurements.

In reality, however, measurement of chips is usually very time-consuming and expensive. Therefore, it is desirable to attain similar accuracy yet with as few number of measurement data as possible. A robust extraction algorithm helps to achieve that goal.

We report experiment results in Table 3.1, where N is the number of sample chips, M is the number of measurement sites, $Noise$ is the amount of random noise added into the Monte Carlo model in terms of the total variation (σ_G^2 + σ_S^2 + σ_R^2). The product of N and M gives the total number of measurements.

According to Table 3.1, we see that our algorithm is very accurate in extracting different variation components, yet very robust to different amount of random noise. For example, with N=2000, M=60 and $Noise$=10%, our extracted results have about 0.4% error for the global variation, 1.9% error for the spatial variation, and 2.0% error for the spatial correlation function. When the noise amount changes from 10% to 100%, the accuracy of our results almost does not change at all. This convincingly shows that our extraction algorithm is very resilient to the measurement noise.

We further test the robustness of our algorithm by reducing the number of chip samples N from 2000 to 1500, 1000, and 500. We see that when there are reasonable number of chip samples (1500 and 1000), our algorithm still gives quite accurate results, and the maximum error for the global variation is no more than 10%, and the maximum error in either the spatial variation or spatial correlation function is less than 5%. When the chip samples drop to 500, we start to see a larger error (but no more than 20%) in the extracted global variation. These observations are expected, because according to the statistical sampling theories, there is a lower bound on the number of samples in order to obtain reasonably accurate statistics.

Table 3.1: Process variation extraction.

N	M	$Noise$	$err(\sigma_G^2)$	$err(\sigma_S^2)$	$err(\rho(v))$
2000	60	10%	0.4%	-1.9%	2.0%
		50%	0.3%	-2.8%	2.7%
		100%	0.3%	-2.6%	3.7%
1500	60	10%	4.1%	2.5%	0.9%
		50%	3.9%	2.1%	1.0%
		100%	3.8%	2.0%	1.2%
1000	60	10%	7.5%	1.2%	1.0%
		50%	7.2%	1.0%	1.0%
		100%	6.9%	1.4%	1.0%
500	60	10%	17.8%	10.9%	6.6%
		50%	18.3%	6.1%	4.8%
		100%	18.6%	4.7%	3.1%
1000	50	10%	6.5%	0.8%	2.8%
		50%	5.7%	-0.4%	3.0%
		100%	5.1%	-3.0%	3.5%
	40	10%	8.6%	-4.1%	6.5%
		50%	8.7%	-3.9%	7.0%
		100%	8.9%	-2.3%	8.4%

Moreover, we observe that because of the optimization procedure used to extract the spatial variation and spatial correlation function as shown in (3.35), the extraction of those two parts is not as sensitive to the number of sample chips as the global variation extraction does.

We further fix the number of sample chips N to be 1000 and vary the number of measurement sites M on the chip from 60 to 50 and 40 to study how the accuracy of our algorithm changes. From Table 3.1, we see that our algorithm still gives quite accurate results. When M changes from 60 to 40, we only see slight increase of errors for all extracted variation components, and none of them has more than 10% error.

We further plot one of the extracted overall process correlation functions in Fig. 3.9, where the (red) triangle points are the model data from the Monte

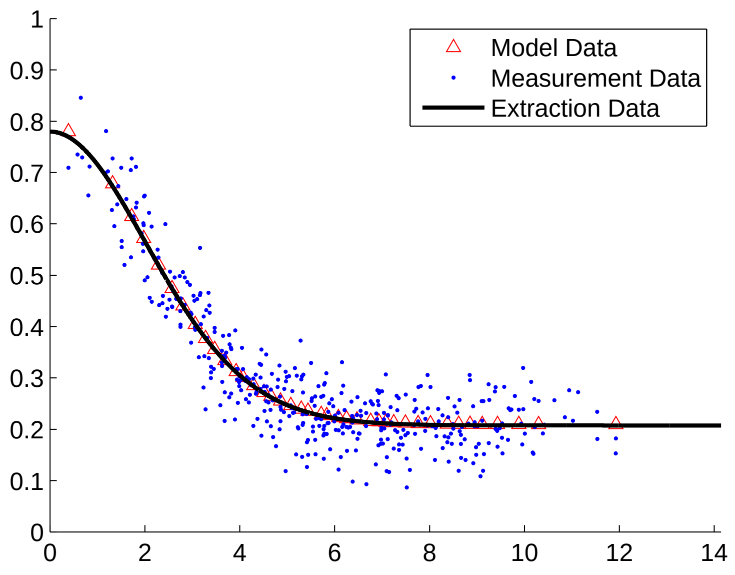


Figure 3.9: Experiment on extracting the overall process correlation function.

Carlo model, the (blue) dotted points are data from our measurements with noise added. Obviously, the measurement data are noisy, not consistent, and are quite difficult to use directly. But after applying our algorithm, we obtain a very robust yet consistent results as shown in the (black) continuous curve, which not only captures the underlying process model, but also provide consistent extrapolation results for those data points that are not even available from measurement.

3.6.2 Extraction of Valid Spatial Correlation Matrix

In the second experiment, we obtain the measurement data for M number of grids of interest on the chip based on a Monte Carlo model with different Gaussian noise added. We want to obtain the overall process correlation matrix for the M number of grids. We apply the algorithm as shown in Fig. 3.7 to achieve this goal.

Table 3.2: Overall process correlation matrix extraction.

Points	50	100	150	200
$\lambda(A)_{least}$	-0.83	-1.43	-1.84	-2.38
$\lambda(\Omega)_{least}$	0	0	0	0
$\parallel A - \Omega \parallel_2$	2.09	4.35	6.85	9.39
$\frac{\|A-\Omega\|_2}{\|A\|_2}$	5.2%	5.9%	6.6%	7.3%
Iterations	23	34	38	41

We show experimental results in Table 3.2. According to the algorithm as shown in Fig. 3.7, we compute individual pair-wise correlations and then put them together to obtain an estimated correlation matrix A. Because of measurement noise, the resulting correlation matrix may not be positive semidefinite as illustrated by the second row in Table 3.2, where the smallest eigenvalue λ_{least} of A is shown. For example, when we have 200 points, the measured correlation matrix has the smallest eigenvalue -2.38. The negative eigenvalue indicates that the measured correlation matrix is not positive semidefinite. On the contrary, after applying the modified alternative project algorithm as shown in Fig. 3.8, we can always find a "closest" yet valid correlation matrix Ω. And the resulting matrix Ω has all non-negative eigenvalues as shown in the third row in Table 3.2. Moreover, the difference between Ω and A is very small (no more than 10%).

We also report the number of iterations needed for the algorithm as shown in Fig. 3.7 to converge. We find that the algorithm converges reasonable fast, and it takes only 41 iterations for the largest test case with M=200. We further plot the change of the least eigen-value of Y_k (which is negative) in each iteration in Fig. 3.10, where the y-axis is the log-plot of the negative of the least eigen-value of Y_k, and x-axis is the iteration numbers. According to Fig. 3.10, we see that

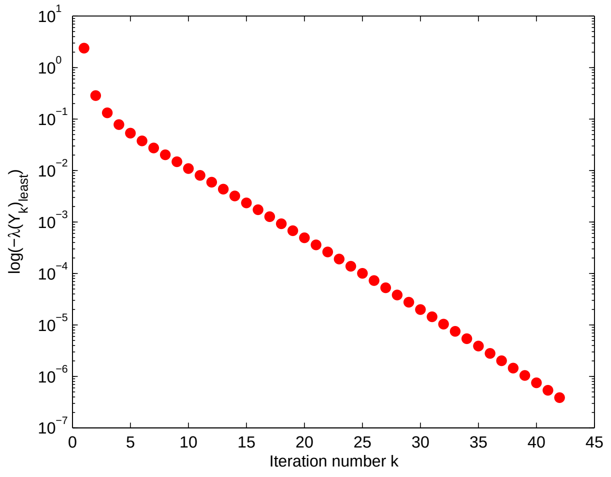


Figure 3.10: The change of the least eigen-value of Y_k in the alternative projection algorithm as shown in Fig. 3.7.

the the least eigen-value of Y_k gets improved quickly in the first two iterations, and then its improvement becomes relative stable in the following iterations.

According to the definition of rate of convergence [81], we have

$$\lim_{k} \frac{|\lambda(Y_{k+1})_{least}|}{|\lambda(Y_k)_{least}|} = \mu, \tag{3.53}$$

where the number μ is called the rate of convergence, and it should be between 0 and 1. If μ=0, then the sequence of $\lambda(Y_k)_{least}$ converges superlinearly. Otherwise, the sequence converges linearly with the rate of convergence of μ. We plot the estimated rate of convergence in each iteration in Fig. 3.11. We observe that the alternative projection algorithm indeed has a linear convergence, which is in agree with the theoretical results given by [43]. The the rate of convergence in this particular example as shown in Fig. 3.11 is approximately 0.72.

In summary, our experiment results convincingly show that our proposed extraction algorithms can accurately extract different variation components and are robust to the unavoidable measurement noise. Moreover, it is guaranteed

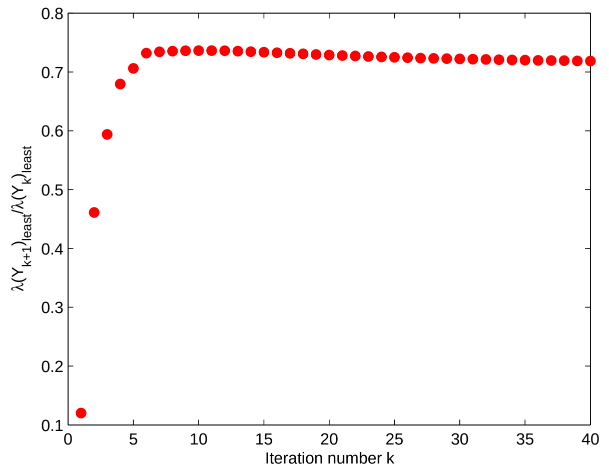


Figure 3.11: Estimated rate of convergence of the alternative projection algorithm as shown in Fig. 3.7.

that our algorithms always produce a *valid* spatial correlation function or spatial correlation matrix, which warrants the validity of further operations on these extracted variation data.

3.7 Conclusion and Discussion

Robust extraction of statistical characteristics of process parameters is essential to achieve the benefits provided by statistical timing analysis and robust circuit optimization. In this chapter, we have developed a novel technique to robustly extract the statistical characteristics of process variation from experimental measurements. Our technique guarantees that the resulting spatial correlation function and spatial correlation matrix are always valid and are the closest to the measurement data even if the data are distorted by some measurement noise.

In this chapter, we have assumed that the spatial correlation follows a Gaussian random process, hence only second-order moments are enough to characterize the variation. In the future, we will remove such an assumption and develop techniques that apply to more general random processes. We also plan to apply this technique to real wafer data and use the extracted process characteristics for robust mixed signal circuits tuning with consideration of correlated process variations in the future.

CHAPTER 4

Efficient Modeling of Spatial Correlations for Parameterized Statistical Static Timing Analysis

This chapter presents an efficient method to model spatial correlations in the context of parameterized statistical static timing analysis (SSTA). Multiple spatially-correlated process parameters with different spatial properties can be handled by this method simultaneously and efficiently. The method constructs a first-order canonical form to model the spatial correlation, so that the resulting forms capture the underlying spatial correlation with minimal error. The proposed technique is implemented in an industrial SSTA tool. Experiment results with industrial circuits show that the proposed approach is significantly more efficient than the existing approach without loosing accuracy. Results also show that by using the proposed approach, the overhead of modeling spatial correlation in parameterized SSTA is small in terms of both memory and runtime even for large industrial designs.

4.1 Introduction

In parametrized statistical static timing analysis (SSTA), we need to represent delay quantities in the timing graph as a first-order canonical form as discussed in section 2.2, in which delays are a linear function of a set of mutually independent random variables. This representation greatly simplifies the computation of SSTA, as operations (such as summation, subtraction, maximization and minimization) of timing quantities can be carried out easily as shown in section 2.4. Several authors have addressed the problem of modeling spatial correlation in parameterized SSTA [16, 1, 48, 114]. The problem, however, is far from being solved for mainstream practical use.

The quad-tree approach proposed in [1] represented gate delays as a linear combination of a set of independent variables that correspond to a set of hierarchical rectangular grids. The sharing of hierarchical grids is used to model the spatial correlation between gate delays. The authors assumed that those coefficients used in the linear combination are given by users. It is not clear, however, how users can compute these coefficients to model a given spatial correlation behavior. Another drawback is that devices near the center of the chip will belong to different hierarchical grids, thus under-estimating their spatial correlations.

The approach of [16] also used a set of rectangular grids to model spatial correlations. It is assumed that a correlation matrix for those grid random variables was known. Principal component analysis (PCA) was used to represent gate delays as linear functions of a set of uncorrelated random variables. As we will show in section 4.3, the PCA-based approach, however, has several drawbacks, making it inefficient for large designs.

In another work [48], the authors proposed to model spatial correlations by expressing a gate delay as a linear function of four independent random variables that corresponds to four corners of the chip. The coefficients of this function depended on the distance between the gate and the chip corners. This approach, however, is very crude in capturing various spatial correlation behaviors because of the limited freedom in choosing both correlation coefficients and the number of random variables.

To avoid the difficulty of modeling spatial correlation, [114] proposed to process those spatially correlated process parameters directly in parameterized SSTA via an *extended pseudo-canonical form*. But the covariance between different timing quantities has to be computed by multiplying the correlation matrix with the coefficients of the extended pseudo-canonical forms; thus loosing the efficiency of a parameterized SSTA. Direct handling extended pseudo-canonical forms with spatially correlated terms also complicates other necessary operations, such as computation of standard deviation and result reporting, for a parameterized SSTA.

This chapter develops a novel method to model correlations in the context of parameterized SSTA. The method is accurate and efficient, and it can be used for large industrial designs with different spatial correlation behaviors. The technique is fully compatible with the existing parameterized SSTA framework and requires no modification of its core engine in order to consider spatial correlations. Experiment results with industrial circuits show that the proposed approach is significantly more efficient than the existing PCA-based technique without loosing accuracy. Results also show that by using the proposed method, the overhead of modeling spatial correlation in parameterized SSTA can be as small as 6% in memory and 25% in runtime even for large industrial designs.

4.2 Parametrized Spatial Correlation Modeling

We divide the chip area into a set of grids, such as rectangular grids as shown in Figure 4.1. The distance between two grids is defined as the center to center distance. We model that the process parameter of interest at each grid as a random variable F_j. Using the first-order approximation, we represent device or interconnect characteristics (such as delay) of interest, D_k, as follows

$$D_k = D_{k,0} + \sum_j \beta_j F_j, \tag{4.1}$$

where $D_{k,0}$ is the mean value of D_k; F_j models the physical process parameters (such as channel length, and oxide thickness) that affect D_k; and β_j is the sensitivity of D_k to F_j.

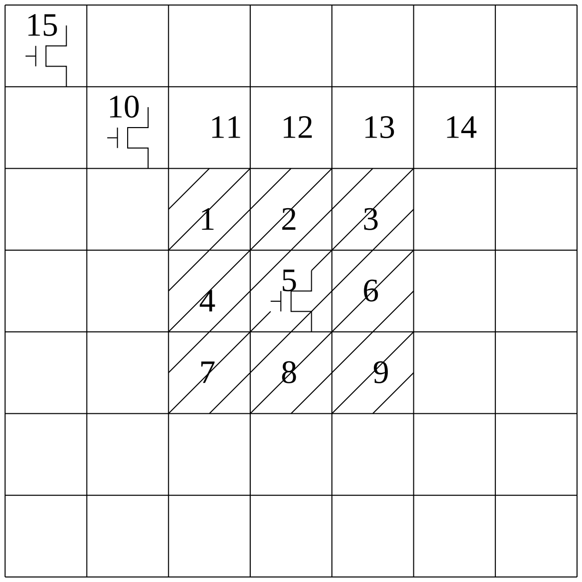


Figure 4.1: Modeling of the spatial correlations.

To compactly represent all delay quantities in the timing graph in a matrix form, we have

$$\mathbf{D} = \mathbf{D_0} + \mathbf{B_{t,n}} \cdot \mathbf{F}, \tag{4.2}$$

where $\mathbf{D}$ is a $t \times 1$ random vector that represents all delay quantities of interests; $\mathbf{D_0}$ is a $t \times 1$ mean vector for $\mathbf{D}$; $\mathbf{F}$ is a $n \times 1$ random vector that includes all physical process parameters at all grids; and $\mathbf{B_{t,n}}$ is a $t \times n$ coefficient matrix. The value of n equals to the product of the number of random process parameters and the number of grids in the chip. For example, if we have three random process parameters, and we divide the chip by a 4×4 rectangular grids, then $n = 3 \times 4 \times 4 = 48$.

In the presence of spatial correlation, process parameters at different locations are not independent. This spatial correlation can be captured by the spatial covariance matrix of $\mathbf{F}$. Without loss of generality, we assume that the random physical process parameters $\mathbf{F}$ follow a multivariate normal distribution with zero mean and unit variance. Therefore, the covariance matrix of $\mathbf{F}$ becomes the correlation matrix, which we denote as $\mathbf{\Omega_{n,n}}$. The *spatial correlation distance* is defined as the minimum distance at which the spatially correlation between two grids is sufficiently small (or smaller than a given threshold value). For a fix chip area, if the spatial correlation distance is large, meaning parameters at grids far apart are still spatially correlated, the correlation matrix $\mathbf{\Omega_{n,n}}$ will be very dense; on the contrary, if the spatial correlation distance is relatively small, then the correlation matrix $\mathbf{\Omega_{n,n}}$ will be sparse because a lot of entries in $\mathbf{\Omega_{n,n}}$ will be zero.

To represent (4.2) as a first-order canonical form, we need to transform the correlated random vector $\mathbf{F}$ into a linear combination of a set of uncorrelated random variables.

4.3 Review of PCA-based Spatial Correlation Modeling

The authors of [16] employed the principle component analysis (PCA) to represent (4.2) as a first-order canonical form. The PCA-based approach has been used extensively for parametrized SSTA in the literature. Hence we review this approach with discussion of its drawbacks in the following.

[16] first performs the eigen-value decomposition (EVD) on the correlation matrix $\mathbf{\Omega_{n,n}}$ to obtain

$$\mathbf{\Omega_{n,n}} = \mathbf{V_{n,n}} \cdot \mathbf{\Lambda_{n,n}} \cdot \mathbf{V^T_{n,n}}, \tag{4.3}$$

where $\mathbf{\Lambda_{n,n}}$ is a diagonal matrix with each entry being an eigen-value of $\mathbf{\Omega_{n,n}}$; $\mathbf{V_{n,n}}$ is an orthonormal matrix with each column being an eigen-vector of $\mathbf{\Omega_{n,n}}$. Because the correlation matrix $\mathbf{\Omega_{n,n}}$ is positive semidefinite by definition, eigen-value decomposition on $\mathbf{\Omega_{n,n}}$ is always feasible. By changing of variables, it obtains

$$\mathbf{F} = \mathbf{V_{n,n}} \cdot \mathbf{\Lambda^{1/2}_{n,n}} \cdot \mathbf{X}, \tag{4.4}$$

where $\mathbf{X}$ is an $n \times 1$ random vector with its each component being an independent random variable with a standard normal distribution. By plugging (4.4) into (4.2), we obtain

$$\begin{aligned} \mathbf{D} &= \mathbf{D_0} + \mathbf{B_{t,n}} \cdot \mathbf{V_{n,n}} \cdot \mathbf{\Lambda^{1/2}_{n,n}} \cdot \mathbf{X} & (4.5) \\ &= \mathbf{D_0} + \mathbf{A_{t,n}} \cdot \mathbf{X}, & (4.6) \end{aligned}$$

where $\mathbf{A_{t,n}} = \mathbf{B_{t,n}} \cdot \mathbf{V_{n,n}} \cdot \mathbf{\Lambda^{1/2}_{n,n}}$, and it is the new coefficient matrix of $\mathbf{X}$. As $\mathbf{X}$ contains mutually independent random variables, (4.6) is in a first-order canonical form. The approach, however, has several drawbacks.

Firstly, in order to perform eigen-value decomposition, the PCA-based approach requires the matrix $\mathbf{\Omega_{n,n}}$ to be a valid correlation matrix, i.e., it must be positive semidefinite. In reality, however, the matrix $\mathbf{\Omega_{n,n}}$ is obtained through measurement. Because of unavoidable measurement errors, the correlation matrix obtained from measurement may not always be positive semidefinite. This problem has been ignored in the literature. In chapter 3, we have developed a novel technique to address this problem.

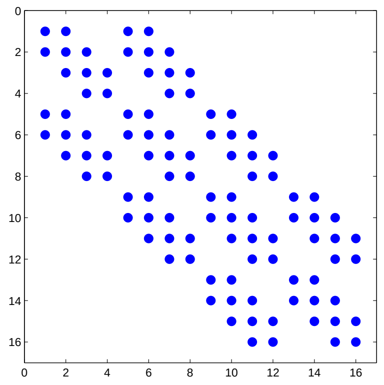


Figure 4.2: Structure of the spatial correlation matrix for 4×4 grids.

Secondly, the computation cost of eigen-value decomposition for $\mathbf{\Omega_{n,n}}$ is high. As we have discussed above, the value of n depends on both the number of random process parameters and the number of grids in the chip. To reasonably capture the spatial correlation and timing characteristics, the number of grids required to cover the whole chip area can not be too small. For example, for a chip that is $10 \times 10mm^2$ with a correlation distance as $100\mu m$, dividing the chip area into 100×100 grids is barely enough to model the spatial correlation, as each grid itself is already $100\mu m$ in length. In this case, even we only consider one source of random process parameter, the matrix size $\mathbf{\Omega_{n,n}}$ would be 10000×10000. Eigen-value decomposition on such a large matrix is quite expensive. To make it even worse, for modern CMOS manufacturing technologies, we can easily have more than tens of sources of random process parameters. This apparently renders the PCA-based approach not practical.

Lastly, the PCA-based approach also complicates the SSTA computation. In the original representation as shown in (4.1), we note that delay D_k is usually affected by a small number of local grid random variables, thus the matrix structure of $\mathbf{B_{t,n}}$ is sparse. In addition, because the spatial correlation distance is

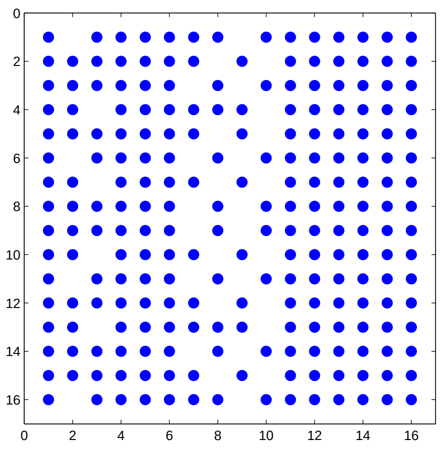


Figure 4.3: Structure of matrix $V_{16,16}$ for 4×4 grids.

usually relatively small compared to the chip size, process parameters at each grid are only spatially correlated to process parameters not far away. This property makes the structure of $\mathbf{\Omega_{n,n}}$ also sparse. After eigen-value decomposition on $\mathbf{\Omega_{n,n}}$, however, the obtained $\mathbf{V_{n,n}}$ may be dense. Because of the dense structure of $\mathbf{V_{n,n}}$. the new matrix $\mathbf{A_{t,n}}$ obtained from (4.6) does not have the sparse structure any more.

As an example, we generate a sparse correlation matrix $\mathbf{\Omega_{n,n}}$ with $n = 16$, whose structure is shown in Figure 4.2. After EVD, we obtain $\mathbf{V_{n,n}}$, whose structure is also shown in Figure 4.3. We can clearly see that the matrix $\mathbf{V_{n,n}}$ does not preserve the sparse structure of the original correlation matrix $\mathbf{\Omega_{16,16}}$.

The impact of this observation is that under the new set of random variables, each delay quantity is a function of almost all modeled random variables in $\mathbf{X}$! In other words, instead of having a short representation as shown in (4.1), we now have a very long representation for each delay quantity. If we were to use these new first-order canonical forms for SSTA computation, it would obviously require more computations for each atomic operation, such as addition, subtraction, maximization and minimization.

In the following, we present a novel technique that overcomes the above problems in modeling spatial correlation.

4.4 Efficient Fitting-based Spatial Correlation Modeling

4.4.1 Principles

We notice that our ultimate goal is to represent delay quantities as a first-order canonical form in the presence of spatial correlation. We also notice that for those random process parameters exhibiting spatial correlation, they are more likely to be correlated with parameters close by than parameters far away.

Hence, for the purpose of modeling spatial correlation, we associate each grid with one (or a number of) dedicated *spatial random variable* $X_{s,i}$ that is independent to one another. We introduce the concept of *correlation index* $\mathcal{I}$, which defines a set of spatial random variables to model the spatial correlation behavior of F_j^i, the ith process parameter at grid j, i.e.,

$$F_j^i = \sum_{k \in \mathcal{I}} \gamma_k \cdot X_{s,k}, \tag{4.7}$$

where γ_k is coefficient to be determined. The concept of correlation index helps to model the spatial correlation. For example, for two parameters that are physically closer to each other, we can define their respective correlation indices in a way so that they share more common spatial random variables. On the other hand, for two that are physically apart from each other, their correlation indices can be defined so that they do not share any common correlated random variables.

To capture this behavior, the correlation index for each F_j^i can be defined according to its physical location in the grids. For example, by letting each grid's correlation index contain spatial random variables associated with the grid's closest nine neighboring grids as shown in Figure 4.1 for parameters at grid 5, then F_j^i can be represented as a linear combination of spatial random variables associated with those nine grids.

To compactly represent the random process parameters for all grids in a matrix form, we obtain

$$\mathbf{F} = \mathbf{G_{n,q}} \cdot \mathbf{X_s}, \tag{4.8}$$

where $\mathbf{X_s}$ is a $q \times 1$ spatial random vector, $\mathbf{G_{n,q}}$ is a coefficient matrix, whose structure is determined by the correlation index $\mathcal{I}$ following (4.7). Because the correlation index $\mathcal{I}$ contains only neighboring grids' spatial random variables, which are a small fraction of the total grids, the structure of $\mathbf{G_{n,q}}$ is sparse.

We can obtain the first-order canonical forms by plugging (4.8) into (4.2) as follows

$$\mathbf{D} = \mathbf{D_0} + \mathbf{B_{t,n}} \cdot \mathbf{G_{n,q}} \cdot \mathbf{X_s}. \tag{4.9}$$

Because both $\mathbf{B_{t,n}}$ and $\mathbf{G_{n,q}}$ are sparse, the first-order canonical form as defined in (4.9) will be much shorter than that from (4.6). Thus SSTA based on our method using (4.9) is expected to run faster than that based on PCA approach using (4.6).

To determine $\mathbf{G_{n,q}}$, we compute the correlation matrix as defined by (4.8) as follows

$$\mathbf{\Omega_{n,n}} = \mathbf{E}(\mathbf{F} \cdot \mathbf{F^T}) = \mathbf{G_{n,q}} \cdot \mathbf{E}(\mathbf{X_s} \cdot \mathbf{X_s^T}) \cdot \mathbf{G_{n,q}^T} = \mathbf{G_{n,q}} \cdot \mathbf{G_{n,q}^T}. \tag{4.10}$$

As we know all $n \times n$ entries in $\mathbf{\Omega_{n,n}}$, by equating each entry of $\mathbf{\Omega_{n,n}}$ with each entry of $\mathbf{G_{n,q}} \cdot \mathbf{G_{n,q}^T}$, we obtain n^2 equations. The number of unknowns in $\mathbf{G_{n,q}}$

depends on the correlation index. In general, we will define the correlation index in a way to make $\mathbf{G_{n,q}}$ sparse, so the exact number of unknowns in $\mathbf{G_{n,q}}$ will be far less than nq. Hence, we are likely to end up with an over-determined system of equations, i.e., the number of equations is greater than the number of unknowns.

The problem of solving $G_{n,q}$ according to (4.10) can be formalized as an unconstrained optimization problem that minimizes the difference between $\mathbf{G_{n,q}} \cdot \mathbf{G^T_{n,q}}$ and $\mathbf{\Omega_{n,n}}$, i.e.,

$$\min: \ \| \mathbf{G_{n,q}} \cdot \mathbf{G^T_{n,q}} - \mathbf{\Omega_{n,n}} \| \, . \tag{4.11}$$

Once the minimization problem is formulated, it can be easily solved via any nonlinear unconstrained minimization technique. Because the above problem mimics the least-square fitting problem, we call our method as *fitting-based method* for spatial correlation modeling.

It should be noted that the above minimization problem (4.11) is in fact a more general formulation of the eigen-value decomposition problem for PCA. For example, if we let the number of spatial random variables q equal to the number of grids n, which is the setting for a PCA-based approach, then $\mathbf{G_{n,q}}$ becomes a square matrix $\mathbf{G_{n,n}}$. Without further assuming any special structure on $\mathbf{G_{n,n}}$, we obtain the optimal solution to (4.11) as $\mathbf{G_{n,n}} = \mathbf{V_{n,n}} \cdot \mathbf{\Lambda^{1/2}}$, with a global minimum value of zero for (4.11). By plugging the solution into (4.9), we obtain exactly the same results as obtained from the PCA approach as shown in (4.4)!

In another word, the PCA-based approach achieves the global minimum to problem (4.11) but at the expense of longer canonical forms. In contrast, by choosing a sparse structure for $\mathbf{G_{n,q}}$, we can not only avoid the expensive EVD operation, but also obtain shorter canonical forms. The only price is that our solution to (4.11) may not be optimal. We believe the formulation of (4.11) is better than the PCA-based approach in that it provides a smooth trade-off between model accuracy and runtime efficiency. Experiment results confirm that, in practice, our approach is indeed much more efficient than the PCA-based approach without loosing much accuracy.

The exact formulation of (4.11) depends on the specific structure of $\mathbf{G_{n,q}}$, or equivalently, the choice of correlation index for each grid. It is this flexibility that allows us to trade model accuracy with runtime efficiency smoothly. In the following, we would like to illustrate the method through some concrete example of $\mathbf{G_{n,q}}$ and $\mathbf{\Omega_{n,n}}$.

4.4.2 Case Studies

In this section, we discuss how to determine the value of γ_k in (4.7), or equivalently $\mathbf{G_{n,q}}$ through cases studies.

Without loss of generality, in the following, we only consider one process parameter F. Among F's three random variation components, i.e., inter-chip global variation, spatial correlation, and uncorrelated random variation, we only consider its spatial correlation part. Following the definition from chapter 3, we model the spatial correlation as a *homogeneous and isotropic random field* with the spatial correlation function given as $f(d)$. All distances are normalized with respect to one grid length. We generate the correlation matrix $\mathbf{\Omega_{n,n}}$ by using the spatial correlation function $f(d)$. Because spatial correlation is homogeneous and isotropic, the number of unique correlation coefficients in $\mathbf{\Omega_{n,n}}$ equals to the number of unique distances between two correlated grids. For example, assuming the spatial correlation distance as $2\sqrt{2}$ for grids as shown in Figure 4.4, we obtain five unique correlation distances that correspond that five unique correlation coefficients, which do not include coefficient zero for distance greater than the spatial correlation distance, nor coefficient one for distance within one grid. Therefore, there are total five unique entries in the correlation matrix $\mathbf{\Omega_{n,n}}$ generated by the spatial correlation function $f(d)$.

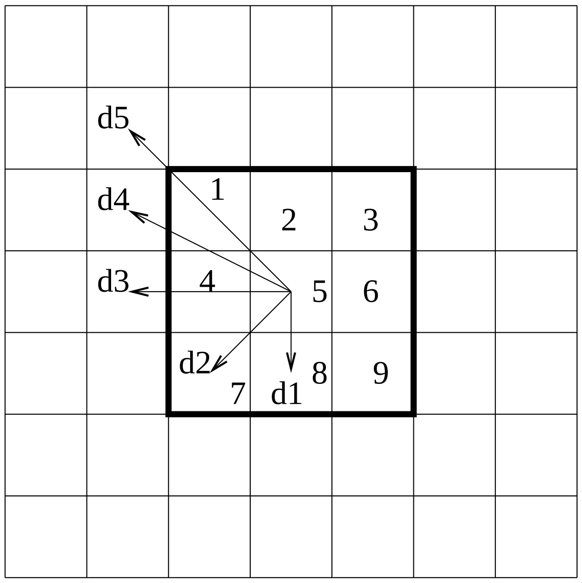


Figure 4.4: Five different distances that define five non-trivial spatial correlation coefficients between two correlated grids, assuming the spatial correlation distance is $2\sqrt{2}$.

For the same example as shown in Figure 4.4, we let the correlation index for each grid include those spatial random variables that are associated with the grid's closest nine neighboring grids (including the grid itself). If we associate each grid with one spatial random variable, then parameters at each grid can be represented as a linear combination of nine spatial random variables. For example, for the parameter at grid 5 in Figure 4.4, we have

$$F_5 = \sum_{k=1..9} \gamma_k \cdot X_{s,k}. \tag{4.12}$$

Under this model, parameters at different grids will share different spatial random variables, and this sharing is closely related to their respective physical locations.

For example, as shown in Figure 4.1, the parameter located at grid 12 (F_{12}) shares $X_{s,1}$, $X_{s,2}$ and $X_{s,3}$ with F_5; the parameter at grid 10 (F_{10}) shares only $X_{s,1}$ with F_5; while parameter at grid 15 (F_{15}) shares nothing with F_5. In another word, F_{12} will more spatially correlated with F_5 than F_{10} does, while there is no spatial correlation between F_{15} and F_5 at all.

Because the spatial correlation is only a function of distance, it makes sense to set the same value for all γ_k if their corresponding grids are the same distance away from grid 5. Then among the nine coefficients in (4.12), γ_1 to γ_9, we only have three unique values, i.e., γ_5, γ_1=γ_3=γ_7=γ_9, and γ_2=γ_4=γ_6=γ_8, because grid 1, 3, 7 and 9 have the same distance from grid 5, so do grid 2, 4, 6 and 8 from grid 5. In another word, for the parameter of interests at grid 5, we have $F_5 = \gamma_5 X_{s,5} + \gamma_1\ (X_{s,1} + X_{s,3} + X_{s,7} + X_{s,9})\ +\gamma_2\ (X_{s,2} + X_{s,4} + X_{s,6} + X_{s,8})$.

By equating the spatial correlations from our model with those in $\mathbf{\Omega_{n,n}}$ according to (4.10), we obtain the following system of equations

$$4\gamma_2^2 + 4\gamma_1^2 + \gamma_5^2 = 1 \tag{4.13}$$

$$2\gamma_5\gamma_2 + 4\gamma_2\gamma_1 = f(d_1) \tag{4.14}$$

$$2\gamma_2^2 + 2\gamma_5\gamma_1 = f(d_2) \tag{4.15}$$

$$2\gamma_1^2 + \gamma_2^2 = f(d_3) \tag{4.16}$$

$$2\gamma_2^2\gamma_1 = f(d_4) \tag{4.17}$$

$$\gamma_1^2 = f(d_5) \tag{4.18}$$

According to the above system of equations (4.13) to (4.18), we have total of six equations with only three unknown coefficients to be determined. Thus it is an over-determined system. We can solve it by formulating it as a non-linear least square problem.

If we want obtain a determined system of equations, we can associate three independent spatial random variables into each grid, and each of them has its own coefficient when presented in (4.7). Then following the same calculation as shown above, we would have six equations similar to (4.13) to (4.18) with some modification. Because we now have six unknown coefficients to be determined. the so-obtained system of equations become a determined system. This problem can be solved by a Newton-Raphelson procedure.

Under the same spirit, we can extend this technique to other grids. For example, if we divide the chip area into a set of hexagon grids as shown in Figure 4.5, we would have two unknown coefficients to be determined, assuming that the correlation index only contains the first level neighborhood grids. A close examination would reveal that under this case, we have three different distances between two grids that share at least one common neighboring grid. Therefore, such a model is also an over-determined system. To obtain a determined system, we can again introduce one additional independent spatial random variable into each grid.

4.5 Experiment Results

The proposed technique for handling spatial correlations has been implemented in the industrial statistical timing analysis tool, EinsStat [94]. For each spatially correlated process parameter, EinsStat allows to specify different spatial correlation functions with different correlation distances. In chapter 3, we have presented a technique to extract a valid spatial correlation function based on measurement data. Table 4.1 shows the characteristics of three real industry ASIC designs that are used to test our implementation.

Among the modeled six sources of random process parameters in our exper-

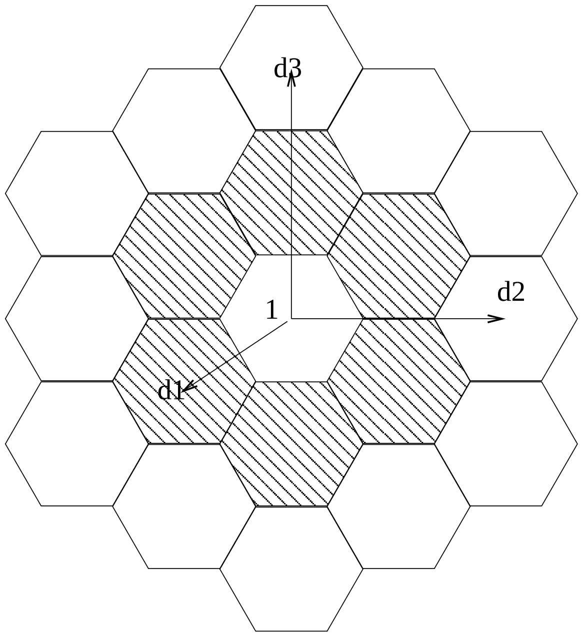


Figure 4.5: Three different distances that define three unique spatial correlation coefficients for hexagon grids.

Table 4.1: Characteristics of test cases.

Chip	Gates	Width (μm)	Length (μm)
D1	87249	1200	1080
D2	57390	1342	1344
D3	8969	9996	9996

iments, only one of them has spatial correlation. The 3-sigma variation of this spatially correlated process parameter is set as 10% to 20% of its nominal value. The 3-sigma variation for the rest of process variations is set as 5% in total. We distribute the total variation into three parts: inter-chip variation (20%), intra-chip spatial correlation (60%), and uncorrelated random variation (20%). The purpose for this setting is to accentuate the effect of spatial correlation.

For comparison purpose, we have also implemented the PCA-based approach. Theoretically, the PCA-based approach is exact in capturing the given spatial

Table 4.2: Experiment results on accuracy comparison.

Chip	Corr.	Grid	PCA-based		Fitting-based	
	Dist. (μm)	Size (μm)	Mean (ns)	Std. Dev. (ns)	Mean (ns)	Std. Dev. (ns)
D1	400	80	3.715	0.178	3.715	0.176
D2	400	80	4.210	0.624	4.210	0.652
D3	4000	800	11.45	0.918	11.45	0.987

correlation. Therefore, we use it as the baseline case for accuracy comparison. Note that, for the sparse correlation matrix, we used the sparse version eigenvalue decomposition for the PCA-based approach.

Table 4.2 compares the accuracy between our fitting-based approach and the existing PCA-based approach. The correlation distance is shown in column 2 for all test cases, with the grid size of each design in column 3. Compared to the grid size, the correlation distance in this set of experiments is relative large. The reason is that we want to guarantee that the PCA-based approach can achieve high accuracy. We report the mean and standard deviation of the circuit delay in column 4 and 5, obtained from the PCA-based approach; and in column 6 and 7, obtained from our fitting-based approach. We observe that our proposed approach can achieve the same mean delay value as the PCA-based approach, and the standard deviation from our approach is almost the same as the one from the PCA-based approach. This convincingly shows that our approach is accurate in capturing the spatial correlation for parameterized SSTA.

We further compare runtime performance between our approach and the PCA-based approach in Table 4.3. The runtime include both the spatial correlation modeling time and the parameterized SSTA runtime. Results are report for the first design D1. Different correlation distance with different grid size are tested. According to Table 4.2, we can see that when the grid size becomes increasingly smaller, the PCA-based approach becomes very slow. This observation is

Table 4.3: Experiment results on run time comparison for D1.

Correlation Distance (μm)	Grid Size	Grid Number	PCA-based (second)	Fitting-based (second)
400	160	63	127	80
200	80	143	124	80
100	40	378	128	83
50	20	1167	390	87
25	10	4599	17703	98

expected, as the smaller the grid size, the larger grid number, hence the larger the correlation matrix fed into eigen-value decomposition. For large correlation matrix, eigen-value decomposition becomes the bottle-neck of the whole computation. Moreover, in our experiments, we also observe that for large matrices, eigen-value decomposition has convergence and accuracy issues. In contrast, our proposed method is almost invariant to the number of grids used to model the spatial correlation. Compared to the PCA-based approach, our method achieves more than 150× speedup.

We also compare the timing results (mean and standard deviation), runtime, and memory usage of our approach with the basic SSTA without modeling spatial correlation (denoted as SSTA-SP) as shown in Table 4.4. Compared to the timing results obtained from SSTA without considering spatial correlation, considering spatial correlation seems to have no impact on the mean circuit delay, but does change the standard deviation. When the correlation distance is increasingly smaller, the standard deviation becomes smaller, implying that the spatial correlation becomes more localized. Compared to SSTA-SP, our approach incurs very small amount of runtime overhead. For example, the SSTA-SP consumes 63 seconds CPU time, while our approach considering spatial correlation consumes no more than 98 seconds CPU time. The relative runtime overhead is about 50% at most. In terms of memory usage, the SSTA-SP method need 930 megabytes,

Table 4.4: Experiment results compared with basic SSTA for D1.

	Corr.	Grid	Delay(ns)		Runtime	Memory(MB)	
	Dist.(μm)	Number	Mean	Std.Dev.	(second)	Total	Overhead
SSTA-SP			3.715	0.205	63	930	0
SSTA+SP	200	143	3.715	0.185	80	1110	180
	50	1167	3.715	0.166	87	1270	340
	25	4599	3.715	0.150	98	1550	580

while considering spatial correlation using our method need no more than 1550 megabytes. The relative memory overhead is no more than 50%.

These experiment results convincingly demonstrate that the proposed technique provides an efficient way to handle spatial correlation while retaining similar accuracy compared to the PCA-based approach. Moreover, the proposed approach can handle large designs with small runtime and memory overhead, call compared to the base statistical timing without considering spatial correlation.

4.6 Conclusion and Discussion

In this chapter, we have presented a novel technique to model spatial correlation in parameterized statistical static timing analysis. The new technique has been implemented in the industrial statistical static timing analysis tool, EinsStat. The implemented software has been used for timing analysis of real industrial ASIC designs. The results proved that our method can handle real designs accurately with small memory overhead and run time overhead.

CHAPTER 5

Simultaneous Buffer Insertion and Wire Sizing Considering Systematic CMP Variation

This chapter studies the impact of Chemical Mechanical Polishing (CMP)-induced systematic variation on interconnect parasitics and design optimization. It shows that (1) fill insertion for CMP planarization significantly increases interconnect capacitance, and different fill patterns introduces additional variations; and (2) CMP-induced dishing and erosion effects can significantly increase interconnect resistance, but have limited impact on capacitance. Considering a table-based best fill insertion to minimize CMP effects and its associated RC parasitics with dishing and erosion, we solve a simultaneous buffer insertion, wire sizing, and fill insertion problem (SBWF). Experiment results show that by solving the SBWF problem, we can, on average, improve timing by 1.6%, reduce power by 3% with 4.9% less buffer area, all compared to the conventional two-step approach, i.e., solving simultaneous buffer insertion and wire sizing problem followed by fill pattern insertion.

5.1 Introduction

Design uncertainty in nanometer technology nodes threatens cost-effectiveness of high-performance circuit manufacturing processes. The main cause for design uncertainty is two-fold: systematic manufacturing process variation and random process variations due to small geometric dimensions [93]. For example, chemical-mechanical planarization (CMP) is an enabling manufacturing process to achieve uniformity of dielectric and conductor height in back-end-of-line (BEOL) process step. However, CMP also introduces systematic design variations due to *dummy fill* insertion [18] and *dishing* and *erosion* [91].

It can be intuitively understood that dummy fill insertion for CMP planarization would change interconnect parasitics. Such parasitic variation should be accurately accounted in order to achieve interconnect optimization, especially when technology continues to scale down to nano-meter region. However, existing research in this regard is very limited and there is no systematic study in the literature that have quantitatively studied the interconnect parasitic variations due to CMP process. For example, as we will show later in this chapter, interconnect capacitance is affected not only by dummy fill insertion, but also by different dummy fill patterns. However, such combined impacts have been largely ignored by existing researchers. For example, [87] assumed one regular fill pattern array and showed that the increase of interconnect capacitance due to such a fill pattern cannot be ignored for interconnect optimization. In [53], the variation of total capacitance due to the Boolean-based placement of dummy fills is considered and it has shown that up to 25% variation is possible. However, it is explained that such a variation is mainly due to intra-die variation but not fill pattern per se. [34] did propose to examine the impact due to different fill patterns, however, no quantitative experiment results have been reported.

The first contribution of this chapter is a study of interconnect parasitic variations due to CMP effects. Specifically, different fill patterns that are "equivalent" with respect to foundry rules, and dishing and erosion of conductors and dielectric similar to those predicted by ITRS [82]. The second contribution of this chapter is to develop an efficient algorithm for simultaneous buffer insertion, wire sizing, and fill insertion considering CMP effects. Results from this chapter have been reported in [38, 39, 40].

5.2 Modeling of CMP Variation

The following two types of CMP effects are considered, i.e., dummy fill insertion, and dishing and erosion. Dummy fill insertion improves the uniformity of metal feature density and enhances the planarization that can be obtained by CMP, but may also change the coupling and total capacitance of interconnects. Dishing and erosion phenomena change interconnect cross-sections [91], and hence may affect interconnect capacitance and resistance.

5.2.1 Fill Patterns

We assume rectangular, isothetic fill features aligned horizontally and vertically between two adjacent interconnects as shown in Figure 5.1. In the figure, conductors A and B are *active* interconnects and the metal shapes between them are dummy fills. We assume all dummy fills are implemented as floating metals in the final layout, as floating dummy-fills are preferred for most ASIC designs due to the short design time and considerable area to be filled [77, 87]. Each distinct *fill pattern* is specified by: (1) the number of fill rows (M) and columns (N); (2) the series of widths $\{W_i\}_{i=1,\ldots,N}$ and lengths $\{L_j\}_{j=1,\ldots,M}$ of fills; (3) the series of horizontal and vertical spacings, $\{S_{x,i}\}_{i=1,\ldots,N}$ and $\{S_{y,j}\}_{j=1,\ldots,M}$, between fills. We denote a fill pattern by $P(M, N, W_i, L_j, S_{x,i}, S_{y,j})$ for simplicity.

To specify the amount of fill metal needed in the space and the resulting metal

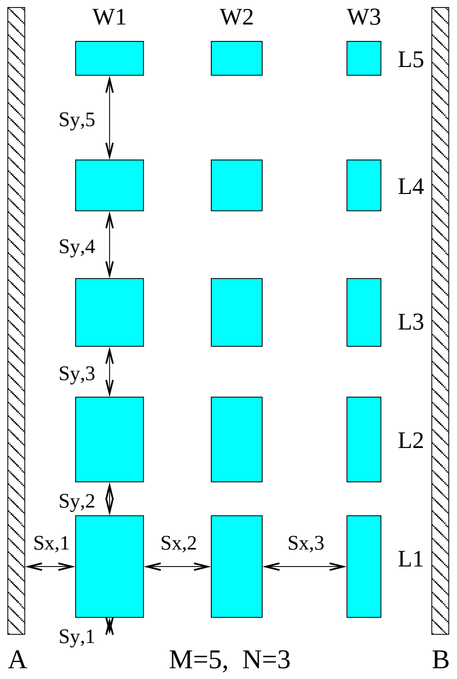


Figure 5.1: Fill pattern definition.

density between two adjacent interconnects, we need the following two definitions.

Definition 3 *Effective metal density ρ_{Cu} – the proportion of the area in a planarization window [91] that all metal features (interconnect + dummy fill metal) occupies, which is usually a hard requirement from the foundry.*

Definition 4 *Local metal density ρ_f – the proportion of the oxide area between two neighboring interconnects that dummy fill metal occupies, which is found by either rule-based method in the industry or by the recently proposed model-based method [89] to achieve ρ_{Cu}.*

To achieve CMP planarity and yield optimization, the foundry usually requires an effective metal density ρ_{Cu} to be satisfied in a "fixed-dissection" regime [18, 35]. Fixed-dissection fill synthesis typically results in a number of tiles (i.e., square regions of layout, usually several tens of microns on a side) wherein prescribed amounts of fill features are to be inserted to meet individual tile's metal density requirement. This translates to assigning the amount dummy fill feature to the space between interconnects, and such amount is expressed in terms of local metal density ρ_f as defined in Definition 4. The inserted fill features subject to at least two foundry-dependent constraints: (1) each fill feature dimension is within the bounds [$\overline{W_l}$,$\overline{W_u}$], and (2) the spacing between any two neighboring fill shapes is at least $\overline{S_l}$. A *valid* fill pattern $P(M, N, W_i, L_j, S_{x,i}, S_{y,j})$ between two adjacent interconnects achieves the required fill feature area and satisfies all design rules.

The required fill area A is computed by $\sum_i W_i \cdot \sum_j L_j = W_b \cdot L_b$, with W_b and L_b as the total fill width budget and length budget, respectively. Hence the total horizontal (or vertical) spacing budget is computed by $S_{x,b} = \sum_j S_{x,i} = W_t - W_b$ (or $S_{y,b} = \sum_j S_{y,j} = L_t - L_b$), where W_t is the spacing between active interconnects and L_t is the length of the active interconnects. Finding a valid fill pattern is equivalent to distributing the budgets of W_b, L_b, $S_{x,b}$, and $S_{y,b}$ among their respective series $\{W_i\}$, $\{L_j\}$, $\{S_{x,i}\}$, and $\{S_{y,j}\}$, which also determines M and N. To solve this problem, we define a positive *distribution characteristic function* (DCF) $f(z)$, where z is an integer variable that takes the index of the element in the series. The i^{th} element of the series is obtained by $f(i)$ plus the lower bound value as specified by filling rules. For example, the value of the i^{th} width $W_i = f(i)+\overline{W_l}$. If the so-obtained W_i exceeds the upper bound $\overline{W_u}$, we take the upper bound value. Therefore, we can obtain a DRC-clean series under the given budget for a chosen DCF; and different DCFs allow us to systematically explore different fill patterns.

To illustrate this point, we take the width series $\{W_i\}$ as an example. If we define $f(z)$ as a constant number, all W_i will have the same value, i.e., all fills

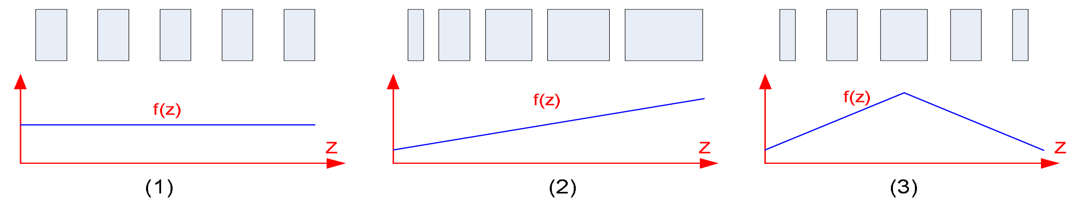


Figure 5.2: Geometrical interpretation of DCF.

have uniform width. If we define $f(z)$ as a linear increasing function, the fills will have a progressively increasing width along the x-axis. If we define $f(z)$ as a triangular function with a convex shape, the center fills will have the largest width, and fills further away from the center will have a progressively decreasing width along the x-axis. Figure 5.2 shows three DCFs and their corresponding geometrical interpretation. In addition to defining different DCFs, we can also try different DCF combinations for $\{W_i\}$, $\{L_j\}$, $\{S_{x,i}\}$, and $\{S_{y,j}\}$ to obtain more fill patterns.

The overall algorithm for searching different valid fill patterns for a given interconnect pair is shown in Figure 5.3.

5.2.2 Fill Pattern Induced Variation

In the following, we examine the impact of fills and fill patterns on interconnect capacitance. We consider the coupling capacitance (C_c) between active interconnects and total capacitance (C_s) of an individual interconnect that is the sum of C_c, area capacitance and fringe capacitance. Intuitively, we can think of C_c as the capacitance between two parallel plates (active interconnect) in the simplest scenario. On the one hand, as the capacitance of a capacitor is inversely proportional to the distance between the two plates, inserting floating fills reduces the distance, which therefore results in larger capacitance. On the other hand,

```
for (all ($W_b$,$L_b$), such that $W_b \cdot L_b = T.A$)
  $S_{x,b} = T.W_t$ - $W_b$;
  $S_{y,b} = T.L_t$ - $L_b$;
  for (all valid $N$,$M$)
   for (all valid length DCF)
   $\{L_j\}$ = lengthDCF($T$,$L_b$,$N$);
    for (all valid width DCF)
    $\{W_i\}$ = widthDCF($T$,$W_b$,$N$);
    for (all valid y spacing DCF)
     $\{S_{y,j}\}$ = spaceYDCF($T$,$S_{y,b}$,$N$);
     for (all valid x spacing DCF)
      $\{S_{x,j}\}$ = spaceXDCF($T$,$S_{x,b}$,$M$);
      $P_v$ = genFillPattern($M, N, W_i, L_j, S_{x,i}, S_{y,j}$);
      $T$.fillList.push($P_v$);
```

Figure 5.3: Algorithm for fill pattern exploration.

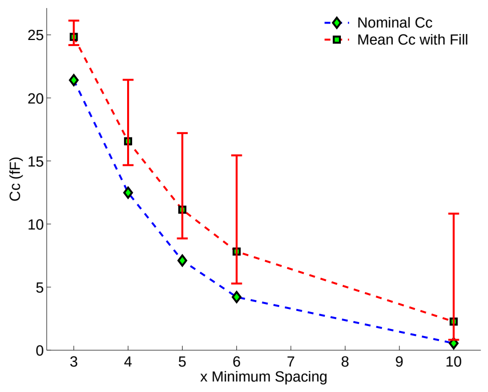
inserting floating dummy fill between the two parallel plates is equivalently to have two "bigger" capacitors connected in serial, which may decrease the capacitance. Therefore, the final C_c is a combined result of the above two effects. In the general case, such a first-order relationship is not that straightforward to derive, hence we resort to the more accurate 3D field solver to examine the impact empirically. We use QuickCap [62], a commercial signoff-quality tool, to extract C_c and C_s. The on-chip interconnect is modeled as a stripline where the interconnect layer is sandwiched between two ground planes. We study global interconnects in the 65nm technology node, with conductor dimensions and spacing derived from the ITRS [82]. For each layout, the interconnect width is set to the minimum width while the spacing between two active interconnects varies from 3× to 10×

minimum spacing[1]. Interconnect length is $1000\mu m$ for all layouts. For a given layout structure, we first extract the nominal C_c and C_s under the nominal geometries, without considering effects of either fill insertion or dishing and erosion. We then extract C_c and C_s under the same nominal geometric values but with fill insertion.

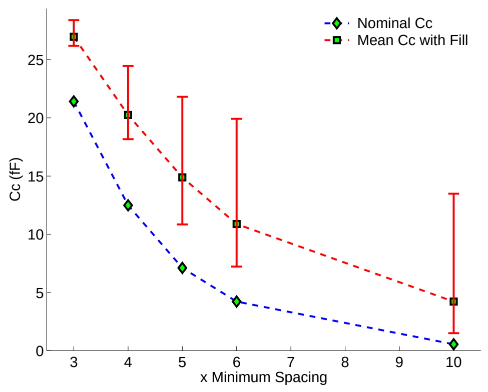
Figures 5.4 and 5.5 pot the variation of coupling capacitance C_c and total capacitance C_s, respectively, when fills are inserted to satisfy the required local metal density ρ_f. We examine the cases where $\rho_f = 0.3, 0.5, 0.7$. We vary the spacing between interconnects from $3\times$ to $10\times$ minimum spacing. The curves with diamond symbols are the nominal C_c or C_s without fill insertion. For each interconnect configuration (given the interconnect spacing and local metal density requirement), there are many valid fill patterns and each results in different C_c and C_s. In both Figure 5.4 and Figure 5.5, the curves with square symbols represent the mean values of C_c and C_s, respectively. The ranges of C_c and C_s are represented by their respective maximum and minimum values among all the fill patterns that we have explored; these are shown in Figure 5.4 and 5.5 as well.

From Figure 5.4, we observe that different fill patterns indeed result in different coupling capacitances, and that fill insertion always increases the coupling capacitance when compared to the nominal case without considering fill insertion. This observation shows that the reduced distance effect due to dummy fill insertion dominates the capacitor serial connection effect, hence the combined effect is increased C_c. Furthermore, the gap between the nominal C_c curve and the mean value C_c curve shows the average increase of C_c due to fill insertion. When the local metal density requirement increases, C_c increase since fill insertion also grows. Moreover, for the same local metal density, the relative change

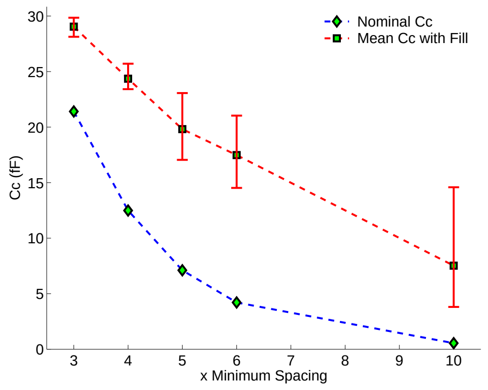
[1]To have fill insertion between active interconnect without violating design rules, the minimum spacing between active interconnect is $3\times$ minimum spacing rule.



(a) ρ_f=0.3

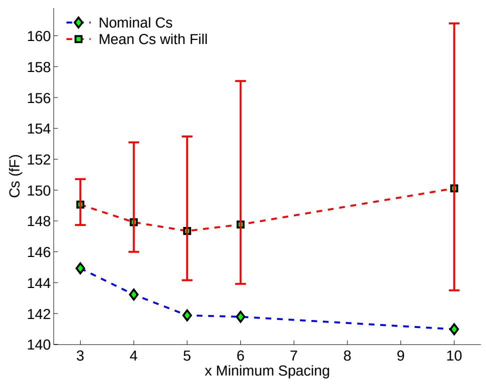


(b) ρ_f=0.5

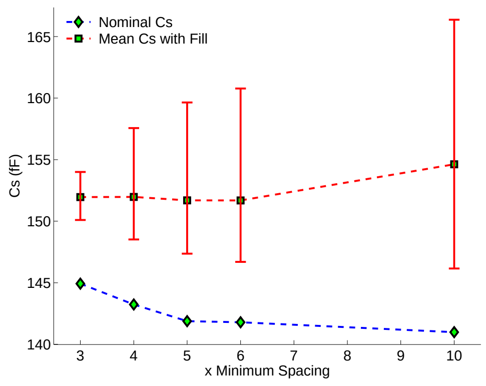


(c) ρ_f=0.7

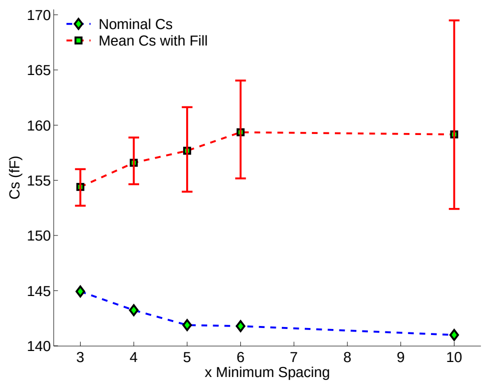
Figure 5.4: Distribution of coupling capacitance C_c.



(a) ρ_f=0.3



(b) ρ_f=0.5

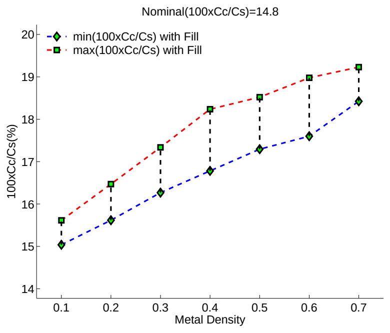


(c) ρ_f=0.7

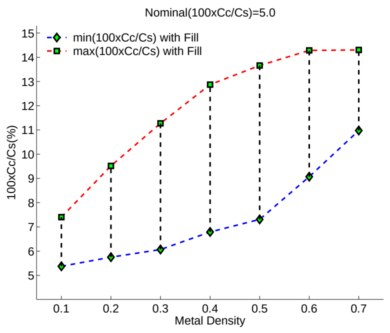
Figure 5.5: Distribution of total capacitance C_s.

of C_c increases as metal spacing increases. For example, when local metal density $\rho_f = 0.5$, the relative C_c change is about 25% on average when the spacing between interconnect is 3× minimum spacing, and is more than tripled when the spacing becomes 6× minimum spacing. Similar observations hold for the total capacitance C_s data in Figure 5.5, except that the relative change of C_s due to fill insertion is less dramatic than that of C_c. Nevertheless, we observe more than 10% relative change of C_s. We conclude that (1) fill insertion significantly increases both C_c and C_s when compared to the nominal case without considering fill insertion; (2) the relative change is more prominent for C_c than for C_s; and (3) different fill patterns yield different C_c and C_s values.

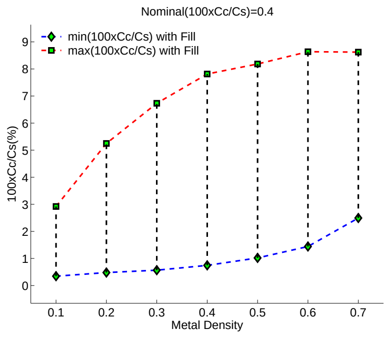
To study the relative importance of the coupling capacitance variation versus the total capacitance variation due to fill insertion, in Figure 5.6 we plot the percentage of C_c over C_s with respect to different local metal densities ρ_f (0.1 to 0.7) between active interconnects, whose spacing is chosen as 3×, 5× and 10× minimum spacing, respectively. Because different fill patterns have different C_c and C_s, we only report results for the fill pattern that results in either minimum or maximum C_c over C_s among all fill patterns studied. The gap between the maximum and minimum percentage curves shows the potential variation due to fill insertion. According to Figure 5.6, we see that fill insertion increases the relative percentage of C_c over C_s compared to the nominal percentage of C_c over C_s without fill insertion as shown in the title of each plot, and that the relative percentage increase becomes larger as the local metal density increases. Moreover, when the metal spacing becomes larger, the relative percentage of C_c over C_s is also increasingly larger compared to the nominal case. On the other hand, because the coupling capacitance decreases as the metal spacing increases, the combined C_c increase is not very significant. In our study, we find that the coupling capacitance is no more than 20% of the total capacitance among all test



(a) 3× minimum spacing



(b) 5× minimum spacing



(c) 10× minimum spacing

Figure 5.6: The percentage of C_c over C_s for different local metal density requirement ρ_f.

cases we have studied.

In summary, fill insertion has significant impact on C_c and different fill pattern densities can result in widely varying C_c. Even though variation of C_s is less dramatic, we still see a spread of more than 10% in relation to the nominal C_s. Therefore, to obtain robust designs that will meet requirements (e.g., delay and parametric yield) after insertion of dummy fill, the variation (increase) of both C_c and C_s must be considered by the design flow.

5.2.3 Dishing and Erosion Induced Variation

Dishing and erosion caused by CMP [33] is illustrated in Figure 5.7. Dishing is the difference between the height of the copper in the trench of the metal interconnect and that of the dielectric in the space surrounding the trenches. Erosion is the difference between the dielectric thickness before CMP and that after CMP. The sum of dishing and erosion is the total loss of metal thickness.

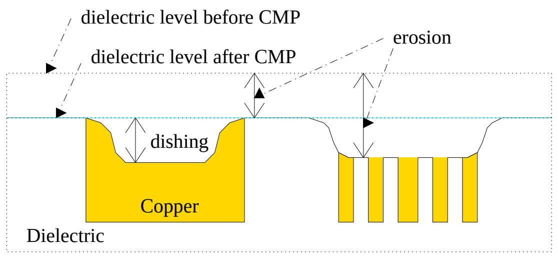


Figure 5.7: Dishing and Erosion in Copper CMP.

We employ the dishing and erosion model for a multi-step CMP process to calculate post-CMP interconnect geometries [33], which is the only public available copper CMP model with detailed parameters published. Our methodology to be presented, however, does not depend on the specific CMP model used.

Table 5.1 shows the resistance for a $1000\mu m$ long global interconnect bus

Table 5.1: Resistance for 65nm global interconnects.

Width (μm)	Space (μm)	R_0 (Ω)	R_f (Ω)
0.24	0.95	186	239 (28.7%)
2.61	0.95	16.9	22.1 (30.6%)
4.75	0.95	9.29	12.3 (31.4%)
0.24	1.43	186	239 (28.8%)
2.61	1.43	16.9	22.1 (30.9%)
4.75	1.43	9.29	12.2 (31.7%)

structure under the 65nm technology node. R_0 is the resistance computed from the geometry values obtained from ITRS specifications, i.e., dishing and erosion effects are not taken into account. R_f is the resistance after fill insertion which fulfills 50% metal density requirement (i.e. $\rho_{Cu} = 0.5$). Based on this, we include the metal loss due to dishing and erosion when computing R_f. From Table 5.1, we can see that resistance variation due to dishing and erosion is significant, and that resistance is always increasing, potentially by more than 30%. As width increases, the resistance variation becomes increasingly severe. For example, when conductor width increases from 0.24μm to 4.75μm, the resistance variation increases from 29% to 32%. Because we assume the same metal density for all interconnects, the resistance is only a function of width; this is due to inherent limitations of the dishing and erosion models [91] we employ.

Table 5.2: Capacitance for 65nm global interconnects.

Width	Space	Nominal		CMP-Fill		CMP+FILL	
μm	μm	$C_{c,0}$	$C_{s,0}$	$C_{c,1}$ (Δ%)	$C_{s,1}$ (Δ%)	$C_{c,f}$ (Δ%)	$C_{s,f}$ (Δ%)
0.24	0.95	25.2	286	24.5 (-2.63%)	285 (-0.33%)	33.5 (33.06%)	286 (-0.11%)
2.61	0.95	26.1	967	25.1 (-3.78%)	965 (-0.19%)	32.9 (26.33%)	954 (-1.35%)
4.75	0.95	25.2	1560	26.0 (2.97%)	1571 (0.68%)	31.9 (26.51%)	1556 (-0.23%)
0.24	1.43	8.4	284	8.6 (2.54%)	283 (-0.13%)	20.3 (142.71%)	289 (1.88%)
2.61	1.43	8.7	957	8.3 (-4.35%)	954 (-0.29%)	21.0 (141.81%)	960 (0.36%)
4.75	1.43	7.8	1574	8.4 (8.11%)	1553 (-1.36%)	19.4 (148.81%)	1564 (-0.69%)

All capacitance values in Table 5.2 are extracted using QuickCap [62]. $C_{c,0}$ and $C_{s,0}$ are the coupling capacitance and total capacitance without considering fill insertion or dishing and erosion effects. $C_{c,1}$ and $C_{s,1}$ are the coupling capacitance and total capacitance for the same assumed structure as in Section 5.2.2, taking geometry variations due to dishing and erosion effects (but no fill insertion) into account. Finally, $C_{c,f}$ and $C_{s,f}$ are the coupling capacitance and total capacitance when effects due to dummy fill, dishing and erosion are all taken into consideration. The percentages in the brackets show the relative changes from values which do not consider any CMP effect (columns 3, 5 and 6). From Table 5.2, we observe that dishing and erosion alone have marginal impact on capacitance for most design contexts. In light of these results, we do not consider dishing and erosion effects on capacitance.

5.2.4 Table-based fill pattern look-up and RC Model

Based upon our study of CMP-induced RC parasitic variations, we tabulate the extracted capacitance in a table indexed by active interconnect width, spacing and local metal density under an optimized fill pattern. Note that varying metal spacing affects the local metal density requirement in the space. During interconnect optimization, each enumerated spacing option requires an appropriate adjustment to the amount of required local metal density. Therefore the fill pattern and RC of all combinations of spacing and local metal density have to be recorded in the table to accommodate any arbitrary spacing and adjusted local metal density. Moreover, as different fill patterns under the same local metal density result in different capacitance values as shown in Section 5.2.2, each table entry only saves the fill pattern and the resulting capacitance under the *best* fill pattern, which gives the minimum C_c among all patterns. The resistance is computed considering dishing and erosion effects. In the following, we denote the resulting RC models as *CMP-aware* RC parasitic models. In contrast, interconnect parasitics without consideration of fill pattern insertion, dishing or erosion effects is called *CMP-oblivious* RC model.

5.3 Simultaneous Buffer Insertion, Wire Sizing, and Fill Insertion

In this section, we formulate the problem of simultaneous buffer insertion, wire sizing, and fill insertion to examine the impact of CMP on interconnect design as follows

Formulation 3 Simultaneous Buffer Insertion, Wire Sizing, and Fill Insertion (SBWF): *Given the topology of a routing tree, its neighboring routing structure, required arrival times and loading capacitances specified at all sinks, determine the placement of buffers, the width of each wire segment, and the fill patterns between the routing tree and its neighboring wires, such that the required arrival time at the root is maximized subject to (1) the slew rate constraint at all sinks and inputs to all buffers; and (2) the effective metal density for CMP planarization.*

We solve the SBWF problem by following the similar dynamic programming framework used to solve either the pure buffer insertion [58], or the pure asymmetric wire sizing problem [23]. Section 6.2.1 gives a quick review of the dynamic programming-based buffering algorithm. Therefore, we only mention the changes that we need to make to solve the SBWF problem in the following.

We consider asymmetric wire sizing for the given routing tree, and assume the center line of the tree is fixed for wire sizing. Hence we can associate each routing segment with two wire width variables, w_1 and w_2, that measure the distance from the center of the segment to its two edges, respectively. Similarly, for each segment, we have two spacing, s_1 and s_2, that measure the distance from the center of the segment to its two neighboring nets. To satisfy the minimum spacing rule s_{min} between any two wires, we enforce a maximum possible wire widths for each wire segment's wire width variables, i.e., $w_1 \leq s_1 - s_{min}$ and $w_2 \leq s_2 - s_{min}$. Without loosing generality, we assume discrete wire sizing for w_1 and w_2 with the minimum wire sizing step is half of the minimum wire width w_{min} decided by designing rules. We illustrate these definitions for an edge $e_{i,j}$ in Figure 5.8.

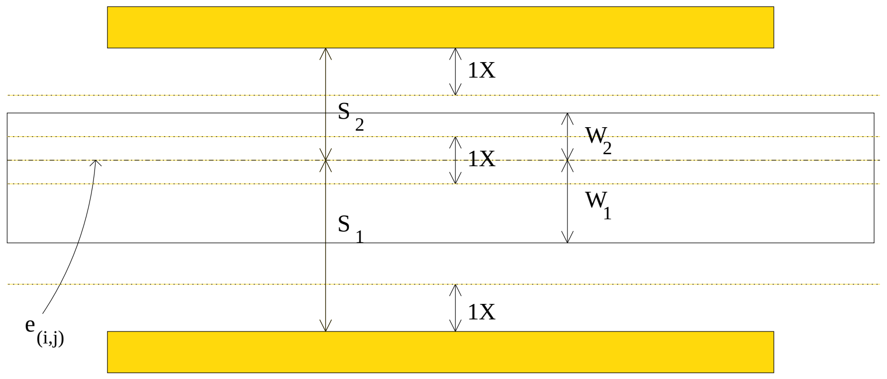


Figure 5.8: Illustration of asymmetric wire sizing.

For every edge $e_{i,j}$, two local dummy fill densities, ρ_f^1 and ρ_f^2, need to be decided in order to achieve the required effective metal density for CMP planarization. When either one of the two wire widths (w_1 and w_2) changes, the two local dummy fill densities also need to be changed, and such an updating procedure is required after each wire sizing step. Algorithm from [89] can be used to serve this purpose.

We employ the Elmore delay model to compute the delay. Each buffer type is modeled by its input capacitance, intrinsic delay, and output resistance. Each wire segment is modeled as a π model. The resistance and capacitance for each wire segment should be determined as follows. (1) We first decide wire segment's wire widths (w_1 and w_2), which allow us to compute the wire's nominal resistance and capacitance without fills. (2) We then compute the two local metal densities (ρ_f^1 and ρ_f^2) to satisfy the effective metal density requirement. (3) Knowing the two local metal densities, we can look-up the CMP-aware RC table to find the "best" fill pattern to insert into the opening space between the wire segment and its neighboring wires. The best fill pattern is determined so that the final adjusted resistance and capacitance result a minimum delay.

We employ the slew model from [6] because of its high fidelity over real design metrics. The slew is computed as $\ln 9 \cdot D_i$, with D_i being the maximum delay from the buffer output at node n to all its downstream buffer input nodes or sinks. Any solution that results in a slew greater than the required slew constraint is discarded.

Because we use the CMP-aware RC table to compute partial solutions during the course of dynamic programming, we take into account the impact of CMP on design while solving the SBWF problem.

In contrast, a best possible solution based on existing practices would solve the same problem in two steps. The simultaneous buffer insertion and wire sizing problem can be solved first to find a best possible RAT at the root, but without considering CMP-effects. After fixing the buffer locations and wire sizes, it then proceeds to insert the best possible fill patterns into the open space of each wire segments. The final RAT at the root need to be adjusted to account for the increase of resistance and capacitance caused by fill insertion. Because of this adjustment to RC parasitics, delays and slews of the original solution will also change. These changes may cause some input buffer nodes' slew constraints to be violated. For fair comparison, we need to adjust the original solution to make it valid, i.e., satisfying all slew constraints. To achieve this, we can over constrain the slew requirement at the first step, in which we solve the simultaneous buffer insertion and wire sizing problem without considering CMP effects. After that, we then insert fill patterns to achieve CMP planarization. Because we already leave some margins for slew adjustment, it is possible that the final solution would result in slews that satisfy the required constraint. This two-step approach is denoted as SBW/F in the following.

We further define the *over-constrain ratio*, κ, as the ratio between the over-constrained slew to the required slew. For slew constraints, the smaller it is, the harder it is to satisfy. Thus over-constraining slew means a reduced value for slew constraints. Therefore, κ is a value smaller than one. To avoid over-design for SBW/F, we need to find the largest possible κ that still guarantees a valid solution after SBW/F is solved. To find this value, we can use a binary search on κ. That is, at each iteration, we solve the SBW/F problem with a given κ value until we find the the largest possible κ with a guaranteed valid solution.

The overall time complexity of our SBWF algorithm is similar to the conventional buffer insertion algorithm, which is known to be polynomial.

5.4 Experiment Results

Both SBWF and SBW/F algorithms have been implemented and tested on a set of benchmarks ($r1$ to $r5$) obtained from the public domain [90], as well as some randomly generated benchmarks ($s1$ to $s10$). The experiments are performed on an Intel Xeon 1.9Ghz Linux workstation with two giga-byte memory.

The characteristics for global interconnects and devices are based on the 65nm

technology as defined in [82]. We use the typical buffer sizes (20× to 120×) and wire sizes ($0.5w_{min}$ to $4.5w_min$) as are used in real designs. All buffers are two-stage cascaded buffers. Because there is no physical layout information in the original benchmarks, we randomly generate the neighboring wire spacing data and the local metal density requirements for each wire segment for all test cases. In our experiments, we find that $\kappa = 0.84$ gives the maximum possible value so that the final results can satisfy the required slew constraints for all benchmarks.

We report experiment results in Table 5.3. All results are based on the accurate CMP-aware RC models for interconnect. The objective for both SBW/F and SBWF is to maximize the required arrival time at the root. A solution with a larger RAT implies a smaller delay, and is more preferable. We find that among all benchmarks, SBWF always results in a larger RAT, and the average timing improvement from SBWF over SBW/F is 1.6%.

As a by-product for RAT optimization, we observe that SBWF also uses less buffer area and power compared to SBW/F. The average saving on buffer area is about 5%, while the average saving on power measured as energy per switch is about 3%. As a tradeoff for these benefits, the SBWF generally uses more wiring area than SBW/F. We have seen about 5% increase on average.

We also compare the runtime between SBWF and SBW/F. Note that the real SBW/F runtime should also include the the binary search iterations in order to find the correct over-constrain rate κ. We, however, only report runtime for one such iteration for SBW/F, and compare that with our SBWF algorithm. From Table 5.3, we see that SBWF has slightly larger runtime compared to one iteration run of SBW/F. This small increase is mainly due to the extra time needed to evaluate the CMP models for dishing and erosion geometries.

In summary, by solving buffer insertion, wire sizing, and fill insertion simultaneously for CMP-planarization, we can achieve better design in terms of delay, buffer area and power with less runtime (iteration-free).

5.5 Conclusion and Discussion

In this chapter, we have studied the impacts of Chemical Mechanical Polishing (CMP)-induced systematic variation on interconnect design. We have shown that fill insertion has a substantial impact on capacitance. Different fill pattern density can result in widely varying capacitance distribution. Dishing and erosion similar to those predicted by the ITRS roadmap can cause interconnect resistance varying up to 30%, but has limited impact on interconnect capacitance. Our study on RC parasitics provides us with an accurate, table look-up based RC model considering systematic CMP variation effects with pre-calculated best fill patterns. Equipped with such a model, we have studied a simultaneous buffer insertion, wire sizing, and fill insertion problem (SBWF). Experiment results have shown that the proposed SBWF approach can achieve 1.6% delay reduction, 3% power reduction and 4.9% buffer area reduction on average when compared to a conventional design flow which performs fill insertion after buffer insertion and wire sizing (SBW/F).

In this work, we assume a fixed routing topology with buffer insertion and wire sizing as a post layout synthesis process. In the future, it remains to study simultaneous routing topology generation with buffer insertion and wire sizing considering systematic and random variations due to both CMP and device effects.

Table 5.3: Experiment results for SBW/F and SBWF.

			SBW/F ($\kappa = 0.84$)					SBWF				
test-case	wire length (m)	# sink	wire area (mm^2)	buffer area (x min)	RAT (ps)	power (pJ)	run-time (s)	wire area (mm^2) (Δ%)	buffer area (x min) (Δ%)	RAT (ps) (Δ%)	power (pJ) (Δ%)	run-time (s)
s1	0.03	19	0.10	2920	-1007	22	0	0.10 (0.9%)	2680 (-8.2%)	-1001 (0.6%)	21 (-6.0%)	0
s2	0.04	29	0.11	3420	-1175	26	0	0.12 (2.0%)	3140 (-8.2%)	-1133 (3.6%)	25 (-5.7%)	1
s3	0.05	49	0.14	4380	-1589	33	1	0.15 (9.5%)	4360 (-0.5%)	-1567 (1.3%)	34 (0.9%)	1
s4	0.07	99	0.18	6180	-1386	47	2	0.19 (8.0%)	6060 (-1.9%)	-1380 (0.4%)	46 (-0.5%)	2
s5	0.10	199	0.26	8820	-2436	67	4	0.27 (5.3%)	8500 (-3.6%)	-2409 (1.1%)	66 (-2.1%)	5
s6	0.13	299	0.31	11720	-2294	88	7	0.33 (5.9%)	11020 (-6.0%)	-2235 (2.6%)	84 (-3.9%)	8
s7	0.16	499	0.38	15220	-3794	113	16	0.40 (5.1%)	14520 (-4.6%)	-3787 (0.2%)	110 (-3.0%)	22
s8	0.19	699	0.43	18320	-3170	136	37	0.45 (4.7%)	17260 (-5.8%)	-3141 (0.9%)	131 (-4.0%)	47
s9	0.21	799	0.47	19700	-2967	147	34	0.49 (3.0%)	18580 (-5.7%)	-2867 (3.4%)	141 (-4.0%)	38
s10	0.22	899	0.51	21000	-2830	157	57	0.53 (3.7%)	20580 (-2.0%)	-2782 (1.7%)	155 (-1.1%)	69
r1	1.32	267	3.79	110000	-4955	838	69	3.97 (4.8%)	104180 (-5.3%)	-4844 (2.3%)	811 (-3.2%)	27
r2	2.60	598	7.32	212760	-6148	1625	0	7.74 (5.7%)	202840 (-4.7%)	-6031 (1.9%)	1582 (-2.6%)	71
r3	3.37	862	9.33	275760	-7358	2103	102	9.89 (6.1%)	261180 (-5.3%)	-7297 (0.8%)	2038 (-3.1%)	91
r4	6.81	1903	18.90	554260	-10748	4233	170	19.83 (4.9%)	522980 (-5.6%)	-10592 (1.4%)	4086 (-3.5%)	175
r5	10.20	3101	28.16	823100	-11984	6297	256	29.48 (4.7%)	777920 (-5.5%)	-11804 (1.5%)	6084 (-3.4%)	271
								(5.0%)	(-4.9%)	(1.6%)	(-3.0%)	

CHAPTER 6

BIPV: Buffer Insertion Considering Process Variations

Advanced process technologies call for a proactive consideration of process variations in design to ensure high parametric timing yield. While essential in almost any of today's high performance IC designs, buffer insertion has not gained enough attention to address this issue. In this chapter, we propose a novel algorithm for buffer insertion that considers process variations. Specifically, we provide some theoretical foundations that help to solve the variation-aware buffer insertion problem. The major contribution of this work is two-fold: (1) two different techniques to handle correlated process variations under nonlinear operations; (2) a provable transitive closure pruning rule that makes linear complexity variation-aware pruning possible. The proposed techniques enable an efficient implementation of variation-aware buffer insertion. We show that, compared to the conventional deterministic approach, the proposed buffer insertion algorithm considering correlated process variations improves the parametric timing yield by more than 15%. We believe that the theoretical contribution of this work is not limited to buffer insertion problem per se, and may be leveraged to solve other CAD problems in the presence of process variation effects.

6.1 Introduction

Many difficulties arise when solving the buffer insertion problem while taking into consideration of process variations. The first challenge is to model buffer solutions as random variables that are correlated. Such correlations are not only caused by correlated process variation (such as global variation and spatial correlation), but also caused by the way buffer solutions are computed. Because almost all existing buffering algorithms follow to some degree the same dynamic programming paradigm [92], in which solutions are computed recursively from downstream nodes, solutions from the same subtree are inherently correlated. The second difficulty is that operations involved in computing different buffer solutions, such as the minimum and multiplication operations, are nonlinear, which further complicates the problem of handling the correlated variations. The third is how to define the *pruning rule* in the presence of process variations, i.e., how to compare different solutions to determine which one is better. Without an efficient pruning rule, a straight-forward implementation of the dynamic programming based buffering algorithm would increase the complexity exponentially.

To the best of our knowledge, there are two recent publications that have attempted to solve the buffer insertion problem considering process variation [47, 25]. However, none of them has addressed the above difficulties with definite answers. For example, [47] considered only the effect of wire-length variation, even though wire-length variation is not a typical process variation. The authors assumed that there was no correlation between different solutions, which is clearly not true. They also proposed three heuristic pruning rules, but none of them can bound the complexity of the algorithm. Under the assumption of an ideal setting (such as infinitely long two-pin nets) with simplification, [25] showed that buffer insertion is not sensitive to process variation. It is not clear, however, that this insensitivity to process variation holds for designs in real setting (e.g., with finite wire length and multiple pins).

This chapter proposes a novel algorithm for buffer insertion that considers correlated process variations. It develops an efficient technique to handle the correlated process variations under nonlinear operations. It also introduces a provable transitive closure pruning rule, which makes linear complexity variation-aware pruning possible. Equipped with the above techniques, we show an efficient implementation of the variation-aware buffer insertion algorithm. We also show that compared to the conventional deterministic approach, our buffer insertion algorithm considering correlated process variations improves the parametric timing yield by more than 15%. Results from this chapter have been reported in [105, 103].

6.2 BIPV Problem Formulation

We formulate the following buffer insertion considering process variation problem formulation:

Formulation 4 Buffer Insertion considering Process Variation (BIPV) Problem: *Given the topology of a routing tree with parasitic capacitance and resistance, required arrival times and loading capacitances specified at all sinks, determine the placement of buffers in the routing tree such that the probability of the required arrival time at the root meeting the design specification is maximized with the consideration of process variations for both interconnect and devices, and as a secondary objective, the number of buffers used are minimized.*

For a given routing tree, two figures-of-merit are associated with every legal buffer position t in the tree: i.e., the *input loading capacitance* (or *downstream* *loading capacitance*) C_t and the *required arrival time* T_t. We characterize a device (or buffer) in terms of its gate capacitance (C_b), intrinsic delay (T_b) and output resistance (R_b). For a given interconnect segment in the layout, we characterize it by its lumped resistance R_w and capacitance C_w.

6.2.1 Buffer Insertion Algorithm Overview

For simplicity of presentation, we follow the same argument as [92] by assuming that the routing tree is given as a binary routing tree and the *legal* buffer positions (nodes) are directly after the branching points of the tree. Note that the methodology to be presented in this work does not depend on these assumptions.

We follow the same dynamic programming paradigm as [92] to solve the buffer insertion problem, including both the deterministic version and the variation-aware version. The overall algorithm (BIPV-ALG) is shown in the top block of Fig. 6.1. We first traverse the routing tree bottom-up once and for all sub-trees we build a set of *dominant solutions*, a concept which we will define in section 6.5. After we reach the root, we pick an optimal solution from the set of kept dominant solutions, optimizing for the required arrival time and the number of buffers inserted. We then back-track the chosen optimal solution to determine the solution for each sub-tree recursively. The key part to this algorithm is the bottom-up traversal of the routing tree. Therefore, we also present the detailed bottom-up algorithm in the bottom block of Fig. 6.1.

According to the algorithm as shown in Figure 6.1, there are three key operations required to compute solutions at each node, i.e., solution after adding a wire, solution after adding a buffer, and solution after merging two solutions. To speedup the dynamic programming based algorithm, a pruning procedure is also required.

```
BIPV-ALG($t$)
  Input: root of the routing tree $t$.
  Output: Solution to BIPV problem.
Z = BIPV-BOTTOM-UP-ALG($t$);
$Z^*$ = PICK-BEST-SOL(Z);
BACK-TRACK-SOL($Z^*$);
```

```
BIPV-BOTTOM-UP-ALG($t$)
  Output: a set of dominant buffered routing trees rooted at $t$.
/* bottom-up traversal of the routing tree */
If node $t$ is sink
  $Z$=INITIALIZATION($C_s$,$T_s$);
  return ($Z$);
Else
  /* Compute solutions from sub-trees */
  $Z_m$ = BIPV-BOTTOM-UP-ALG(t.left);
  $Z_n$ = BIPV-BOTTOM-UP-ALG(t.right);
  /* Merge two solutions */
  $Z = \emptyset$;
  For each solution $(C_m, T_m)$ from $Z_m$
    For each solution $(C_n, T_n)$ from $Z_n$
      $(C_t, T_t)$=MERGE-SUBNODE-JPDF($(C_m, T_m)$,$(C_n, T_n)$);
      $Z = Z \cup (C_t, T_t)$;
  $Z$ = PRUNING($Z$);
  if node $t$ is root
    return($Z$);
  $Z = \emptyset$;
  For each solution $(C_n, T_n)$ in $Z_t$
    $(C_t, T_t)$ = ADD-WIRE($(C_n, T_n)$, $(C_w, R_w)$);
    $Z = Z \cup (C_t, T_t)$;
    $(C_t, T_t)$ = ADD-BUFFER($(C_t, T_t)$,$(C_b, T_b, R_b)$);
    $Z = Z \cup (C_t, T_t)$;
  return $Z$;
```

Figure 6.1: Algorithm for the buffer insertion problem.

Therefore, we only need to provide details on the three key operations with appropriately defined pruning strategies to solve the BIVP problem. This arguments hold for both the BIPV problem and the conventional deterministic buffer insertion problem.

6.2.2 Review of Deterministic Buffer Insertion Algorithm

When each interconnect segment in the routing tree is modeled by a π model, under the Elmore delay model, C_t and T_t can be computed recursively as follows. By adding a wire at node n, all solutions at n are propagated to the other end of the wire t as follows

$$C_t = C_n + C_w \tag{6.1}$$

$$T_t = T_n - R_w \cdot C_n - \frac{1}{2} \cdot R_w \cdot C_w. \tag{6.2}$$

By adding a buffer at node n, all solutions at n are propagated to the input of the buffer as follows

$$C_t = C_b \tag{6.3}$$

$$T_t = T_n - T_b - R_b \cdot C_n. \tag{6.4}$$

If the solution at node t is obtained by merging two solutions from its two subtrees rooted at nodes m and n, respectively, then solutions at the merging point are computed by

$$C_t = C_n + C_m \tag{6.5}$$

$$T_t = min(T_n, T_m). \tag{6.6}$$

Knowing the above three key operations, the deterministic dynamic programming based buffer insertion can be solved by recursively applying the above three operations to obtain new solutions as we traverse the routing tree bottom-up.

Similarly, to solve the variation-aware buffer insertion, we only need to know how to handle the above three operations considering process variation. In the following, we discuss two variation-aware buffering algorithms under different variation models.

6.3 BIPV Algorithm by JPDF Computation

In this section, we only consider local random variations for both devices and interconnect. i.e., ignoring both global variation and spatial correlation. Therefore, devices or interconnects at different locations are independent. We model buffer characteristics, i.e., C_b, R_b, and T_b, as random variables whose joint probability density function (JPDF) is $g(C_b, R_b, T_b)$ with its domain given by Ω_{C_b,R_b,T_b}. We model interconnect characteristics, C_w and R_w are random variables whose JPDF is $h(C_w, R_w)$ with its domain given by Ω_{C_w,R_w}. We assume that device and interconnect variations are independent.

According to (6.1) and (6.2), or (6.3) and (6.4), we know that the upstream variations do not change the distribution of C_t and T_t, as C_t and T_t are only functions of node t's downstream random variations. Therefore, C_t and T_t's distributions are independent of their upstream random variations. Moreover, if (C_{t1}, T_{t1}) and (C_{t2}, T_{t2}) share some (or do not share any) common down-stream paths, then C_{t1}, T_{t1}, C_{t2}, and T_{t2} are mutually correlated (or independent).

Even thought (C_t, T_t) depend on their downstream random variations in a complicated way, the way we compute (C_t, T_t) is via the recursive equations according to either (6.1) and (6.2), or (6.3) and (6.4), or (6.5) and (6.6). Therefore, we develop an efficient algorithm to compute the JPDF of (C_t, T_t) in a recursive fashion while we traverse the routing tree. This is particularly useful in the context of dynamic programming, because only incremental computation is necessary at each node. To develop the recursive computation formula, we employ the multivariate transformation technique [42]. We denote the JPDF of node t's direct downstream nodes as f_{C_n,T_n} with its domain given by Ω_{C_n,T_n}.

6.3.1 JPDF after Adding a Wire

To compute the JPDF of (C_t, T_t) obtained from (6.1) and (6.2) via the multivariate transformation technique, we introduce two new random variables X and Y as follows:

$$X = C_w; \tag{6.7}$$

$$Y = R_w; \tag{6.8}$$

$$C_t = C_n + C_w; \tag{6.9}$$

$$T_t = T_n - R_w \cdot C_n - \frac{1}{2} \cdot R_w \cdot C_w. \tag{6.10}$$

By transforming variables, we obtain

$$C_w = X; \tag{6.11}$$

$$R_w = Y; \tag{6.12}$$

$$C_n = C_t - X; \tag{6.13}$$

$$T_n = T_t + Y \cdot C_t - \frac{1}{2} \cdot Y \cdot X. \tag{6.14}$$

It is easy to verify that the mapping between (C_w, R_w, C_n, T_n) and (X, Y, C_t, T_t) is a one-to-one mapping. Therefore, the Jacobian is given by

$$\begin{aligned} J &= \begin{vmatrix} \frac{\partial C_w}{\partial X} & \frac{\partial C_w}{\partial Y} & \frac{\partial C_w}{\partial C_t} & \frac{\partial C_w}{\partial T_t} \\ \frac{\partial R_w}{\partial X} & \frac{\partial R_w}{\partial Y} & \frac{\partial R_w}{\partial C_t} & \frac{\partial R_w}{\partial T_t} \\ \frac{\partial C_n}{\partial X} & \frac{\partial C_n}{\partial Y} & \frac{\partial C_n}{\partial C_t} & \frac{\partial C_n}{\partial T_t} \\ \frac{\partial T_n}{\partial X} & \frac{\partial T_n}{\partial Y} & \frac{\partial T_n}{\partial C_t} & \frac{\partial T_n}{\partial T_t} \end{vmatrix} \\ &= \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ \frac{1}{2} \cdot Y & C_t - \frac{1}{2} \cdot X & Y & 1 \end{vmatrix} \\ &= 1. \end{aligned} \tag{6.15}$$

As C_w and R_w are independent of C_n and T_n, the JPDF of (C_w, R_w, C_n, T_n) is given by: $f_{C_w,R_w,C_n,T_n} = h(C_w, R_w) \cdot f_{C_n,T_n}$. Therefore, we obtain the JPDF of (X, Y, C_t, T_t) as follows:

$$\begin{aligned} f_{X,Y,C_t,T_t} &= |J| \cdot f_{C_w,R_w,C_n,T_n}(C_w, R_w, C_n, T_n) = h(C_w, R_w) \cdot f_{C_n,T_n}(C_n, T_n) \\ &= h(C_w, R_w) \cdot f_{C_n,T_n}(C_t - X, T_t + Y \cdot C_t - \frac{X \cdot Y}{2}) \end{aligned} \quad (6.16)$$

Then the JPDF of (C_t, T_t) is obtained by

$$f_{C_t,T_t} = \int_{\Omega_{X,Y}(C_t,T_t)} f_{X,Y,C_t,T_t} dX dY, \quad (6.17)$$

where $\Omega_{X,Y}(C_t, T_t)$ is the domain for X, Y in terms of C_t and T_t. Knowing C_w and R_w's domain Ω_{C_w,R_w} and (C_n, T_n)'s domain $\Omega_{C_n,Tn}$, we can deduce $\Omega_{X,Y}(C_t, T_t)$ according to (6.7), (6.8) (6.9) and (6.10).

6.3.2 JPDF after Adding a Buffer

To compute the JPDF of (C_t, T_t) obtained from (6.3) and (6.4) via the multivariate transformation technique, we introduce three new random variables X, Y and Z as follows:

$$X = C_n; \quad (6.18)$$

$$Y = R_b; \quad (6.19)$$

$$Z = T_b; \quad (6.20)$$

$$C_t = C_b; \quad (6.21)$$

$$T_t = T_n - T_b - R_b \cdot C_n. \quad (6.22)$$

By transforming variables, we obtain

$$C_b = C_t; \tag{6.23}$$

$$R_b = Y; \tag{6.24}$$

$$T_b = Z; \tag{6.25}$$

$$C_n = X; \tag{6.26}$$

$$T_n = T_t + Z + Y \cdot X. \tag{6.27}$$

It is easy to show that the mapping between $(C_b, R_b, T_b, C_n, T_n)$ and (X, Y, Z, C_t, T_t) is a one-to-one mapping. Therefore, the Jacobian is given by

$$\begin{aligned} J &= \begin{vmatrix} \frac{\partial C_b}{\partial X} & \frac{\partial C_b}{\partial Y} & \frac{\partial C_b}{\partial Z} & \frac{\partial C_b}{\partial C_t} & \frac{\partial C_b}{\partial T_t} \\ \frac{\partial R_b}{\partial X} & \frac{\partial R_b}{\partial Y} & \frac{\partial R_b}{\partial Z} & \frac{\partial R_b}{\partial C_t} & \frac{\partial R_b}{\partial T_t} \\ \frac{\partial T_b}{\partial X} & \frac{\partial T_b}{\partial Y} & \frac{\partial T_b}{\partial Z} & \frac{\partial T_b}{\partial C_t} & \frac{\partial T_b}{\partial T_t} \\ \frac{\partial C_n}{\partial X} & \frac{\partial C_n}{\partial Y} & \frac{\partial C_n}{\partial Z} & \frac{\partial C_n}{\partial C_t} & \frac{\partial C_n}{\partial T_t} \\ \frac{\partial T_n}{\partial X} & \frac{\partial T_n}{\partial Y} & \frac{\partial T_n}{\partial Z} & \frac{\partial T_n}{\partial C_t} & \frac{\partial T_n}{\partial T_t} \end{vmatrix} \\ &= \begin{vmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ Y & X & 1 & 0 & 1 \end{vmatrix} = -1. \end{aligned} \tag{6.28}$$

As C_b, R_b, and T_b are independent of C_n and T_n, the JPDF of $(C_b, R_b, T_b, C_n, T_n)$ is given by: f_{C_b,R_b,T_b,C_n,T_n} $=g(C_b, R_b, T_b) \cdot f_{C_n,T_n}$. Therefore, we obtain the JPDF of (X, Y, Z, C_t, T_t) as follows:

$$\begin{aligned} f_{X,Y,Z,C_t,T_t} &= |J| \cdot f_{C_b,R_b,T_b,C_n,T_n} = g(C_b, R_b, T_b) \cdot f_{C_n,T_n} \\ &= g(C_t, Y, Z) \cdot f_{C_n,T_n}(X, T_t + Z + Y \cdot X). \end{aligned} \tag{6.29}$$

Then JPDF of (C_t, T_t) is obtained by

$$f_{C_t,T_t} = \int_{\Omega_{X,Y,Z(C_t,T_t)}} f_{X,Y,Z,C_t,T_t} dX dY dZ, \tag{6.30}$$

where $\Omega_{X,Y,Z}(X,Y,Z)$ is the new domain for X, Y and Z in terms of C_t and T_t. Knowing C_b, R_b and T_b's domain Ω_{C_b,R_b,T_b} and (C_n, T_n)'s domain $\Omega_{C_n,Tn}$, we can deduce $\Omega_{X,Y,Z}(X,Y,Z)$ according to (6.18), (6.19), (6.20), (6.21) and (6.22).

6.3.3 JPDF after Merging Two Solutions

To compute the JPDF for (C_t, T_t) after merging two solutions according to (6.5) and (6.6), we introduce two new random variables as follows:

$$X = C_m \tag{6.31}$$

$$Y = T_n + T_m \tag{6.32}$$

$$C_t = C_n + C_m \tag{6.33}$$

$$T_t = min(T_n, T_m) \tag{6.34}$$

Because of the min-function, there is no one-to-one mapping relation between (C_m, T_m, C_n, T_n) and (X, Y, C_t, T_t), thus we cannot use the multivariate transformation technique to compute the JPDF of (C_t, T_t) directly. But we note that the original domain Ω_{C_m,T_m,C_n,T_n} for (C_m, T_m, C_n, T_n) can be divided into two disjoint sub-domains Ω_1 and Ω_2, where Ω_1 is the sub-domain with $T_m \leq T_n$ and Ω_2 is the other sub-domain with $T_m > T_n$. For each disjointed sub-domain, we can show that there exists a one-to-one mapping between (C_m, T_m, C_n, T_n) and (X, Y, C_t, T_t), thus we can compute the JPDF for each sub-domain by using the multivariate transformation technique. Then by combining the two sub-domains' JPDF, we can compute the JPDF for the whole domain [42].

We first find the JPDF for sub-domain Ω_1 with $T_m \leq T_n$. We have:

$$T_t = min(T_m, T_n) = T_m \quad (6.35)$$

After transformation of variables, we have

$$\begin{aligned} C_m &= X; & (6.36)\\ C_n &= C_t - X; & (6.37)\\ T_m &= T_t; & (6.38)\\ T_n &= Y - T_t. & (6.39)\end{aligned}$$

The Jacobian is given by

$$\begin{aligned} J &= \begin{vmatrix} \frac{\partial C_m}{\partial X} & \frac{\partial C_m}{\partial Y} & \frac{\partial C_m}{\partial C_t} & \frac{\partial C_m}{\partial T_t} \\ \frac{\partial C_n}{\partial X} & \frac{\partial C_n}{\partial Y} & \frac{\partial C_n}{\partial C_t} & \frac{\partial C_n}{\partial T_t} \\ \frac{\partial T_m}{\partial X} & \frac{\partial T_m}{\partial Y} & \frac{\partial T_m}{\partial C_t} & \frac{\partial T_m}{\partial T_t} \\ \frac{\partial T_n}{\partial X} & \frac{\partial T_n}{\partial Y} & \frac{\partial T_n}{\partial C_t} & \frac{\partial T_n}{\partial T_t} \end{vmatrix} \\ &= \begin{vmatrix} 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & -1 \end{vmatrix} = 1. \quad (6.40)\end{aligned}$$

As (C_m, T_m) is independent of (C_n, T_n), the JPDF of (C_m, T_m, C_n, T_n) is given by: $f_{C_m,T_m,C_n,T_n} = f_{C_m,T_m} \cdot f_{C_n,T_n}$, Therefore, we can obtain the JPDF of (X, Y, C_t, T_t) in sub-domain Ω_1 as follows:

$$\begin{aligned} f_{\Omega_1} &= |J| \cdot f_{C_m,T_m,C_n,T_n} = f_{C_m,T_m} \cdot f_{C_n,T_n} \\ &= f_{C_m,T_m}(X, T_t) \cdot f_{C_n,T_n}(C_t - X, Y - T_t). \quad (6.41)\end{aligned}$$

Similarly, for the sub-domain Ω_2 with $T_m > T_n$, We have:

$$\begin{aligned} f_{\Omega_2} &= |J| \cdot f_{C_m,T_m,C_n,T_n} \\ &= f_{C_m,T_m}(X, Y - T_t) \cdot f_{C_n,T_n}(C_t - X,\ T_t). \quad (6.42)\end{aligned}$$

Combining (6.41) and (6.42) together, we obtain the The JPDF for (X, Y, C_t, T_t) in the whole domain Ω_{X,Y,C_t,T_t} as follows [42]:

$$\begin{aligned} f_{X,Y,C_t,T_t} &= f_{\Omega_1}(X, Y, C_t, T_t) + f_{\Omega_2}(X, Y, C_t, T_t) \\ &= f_{C_m,T_m}(X, T_t) \cdot f_{C_n,T_n}(C_t - X, Y - T_t) + \\ &\quad f_{C_m,T_m}(X, Y - T_t) \cdot f_{C_n,T_n}(C_t - X, T_t) \end{aligned} \tag{6.43}$$

Then the JPDF of (C_t, T_t) is obtained by

$$f_{C_t,T_t}(C_t, T_t) = \int_{\Omega_{X,Y}(C_t,T_t)} f_{X,Y,C_t,T_t} dX dY, \tag{6.44}$$

where $\Omega_{X,Y}(C_t, T_t)$ is the new domain for (X, Y) in terms of C_t and T_t. Knowing (C_m, T_m)'s domain Ω_{C_m,T_m} and (C_n, T_n)'s domain $\Omega_{C_n,Tn}$, we can deduce $\Omega_{X,Y}(C_t, T_t)$ from (6.31),(6.32), (6.33) and (6.34).

6.3.4 JPDF Initialization

To carry on the above recursive computation of JPDF, we have to set up the initial conditions starting from sinks. As sink's C_s and T_s are constants and buffers are always inserted at sink's upstream node with a wire between them, (6.1) and (6.2) should be used for JPDF initialization. After transformation of variables, we have:

$$C_w = C_t - C_s; \tag{6.45}$$

$$R_w = \frac{2(T_s - T_t)}{(C_t + C_s)}. \tag{6.46}$$

It is easy to verify that the mapping between (C_w, R_w) and (C_t, T_t) is a one-to-one mapping. Therefore, the Jacobian is given by

$$\begin{aligned} J &= \begin{vmatrix} \frac{\partial C_w}{\partial C_t} & \frac{\partial C_w}{\partial T_t} \\ \frac{\partial R_w}{\partial C_t} & \frac{\partial R_w}{\partial T_t} \end{vmatrix} \\ &= \begin{vmatrix} 1 & 0 \\ \frac{-2(T_s - T_t)}{(C_t + C_s)^2} & \frac{-2}{(C_t + C_s)} \end{vmatrix} = \frac{-2}{(C_t + C_s)}. \end{aligned} \tag{6.47}$$

As the JPDF for (C_w, R_w) is already known as $h(C_w, R_w)$, we can obtain the JPDF of (C_t, T_t) as follows:

$$f_{C_t, T_t} = |J| \cdot h(C_w, R_w) = \frac{2}{(C_t + C_s)} \cdot h(C_t - C_s, \frac{2(T_s - T_t)}{(C_t + C_s)})$$

6.4 BIPV Algorithm by First-order Canonical Form

In this section, we consider all sources of random variations (including global variation and spatial correlation) for devices and interconnect. We employ the first-order canonical form to model all characteristics of interests, such as T_b, C_b and R_b for devices, and R_w and C_w for interconnect, as a random variable, i.e., $T_b = T_{b0} + \gamma_b^T X$, $C_b = C_{b0} + \eta_b^T X$, $R_b = R_{b0} + \zeta_b^T X$, $C_w = C_{w0} + \eta_w^T X$, $R_w = R_{w0} + \zeta_w^T X$.

Applying these random variables to (6.1) to (6.6), we obtain solutions C_t and T_t that are functions of T_b, C_b, R_b, R_w and C_w, hence they are also random variables. However, because of the non-linear operators (multiplication and minimum operations) involved in computing the new solution, the distributions are no longer in the first order canonical form.

Therefore, to make the computation efficient, we propose a novel approximation technique in the following that keeps C_t and T_t after nonlinear operations still in the first order canonical form without loss of much accuracy.

To proceed, we represent all downstream node solutions by the following first order canonical forms:

$$
\begin{aligned}
C_n &= C_{n0} + \alpha_n^T \cdot X, && (6.48)\\
T_n &= T_{n0} + \beta_n^T \cdot X. && (6.49)
\end{aligned}
$$

At the sink, C_n is the loading capacitance and T_n is the required arrival time, which are known from design specification.

6.4.1 Canonical Form after Adding a Wire

If the solution at node t is obtained by adding a wire at its direct downstream node n, we compute the new solution at node t as follows:

$$
\begin{aligned}
C_t &= (C_{n0} + C_{w0}) + (\alpha_n^T + \eta_w^T) \cdot X, && (6.50)\\
T_t &= (T_{n0} - R_{w0} \cdot C_{n0} - \frac{1}{2} \cdot R_{w0} \cdot C_{w0}) + (\beta_n^T - C_{n0} \cdot \zeta_w^T - R_{w0} \cdot \alpha_n^T) \cdot X \\
&\quad - \frac{1}{2}(R_{w0} \cdot \eta_w^T + C_{w0} \cdot \zeta_w^T) \cdot X - X^T(\zeta_w \cdot \alpha_n^T + \frac{1}{2}\zeta_w \cdot \eta_w^T)X \\
&= T_{t0} + \delta_t^T \cdot X - X^T \Gamma X, && (6.51)
\end{aligned}
$$

where T_{t0}=$T_{n0} - R_{w0} \cdot C_{n0} - \frac{1}{2} \cdot R_{w0} \cdot C_{w0}$, δ_t=$\beta_n - C_{n0} \cdot \zeta_w - R_{w0} \cdot \alpha_n$- $\frac{1}{2}(\cdot R_{w0} \cdot \eta_w + C_{w0} \cdot \zeta_w)$, and Γ=$\zeta_w \cdot \alpha_n^T + \frac{1}{2}\zeta_w \cdot \eta_w^T$. It is obvious that C_t in (6.50) is already a canonical form, but T_t in (6.51) is not due to the quadratic term $X^T \Gamma X$. To represent T_t as a canonical form, we have the following Theorems:

Theorem 2 *Given random variables in vector form X that follow a standard multivariate Gaussian distribution as N(0,I), i.e., $E(X) = 0$ and $E(X^2) = I$, for any vector δ and matrix Γ, we have*

$$
\begin{aligned}
E(X^T \Gamma X) &= tr(\Gamma), && (6.52)\\
E(X^T \Gamma X \delta^T X) &= 0, && (6.53)\\
E((X^T \Gamma X)^2) &= 2tr(\Gamma^2) + tr(\Gamma)^2. && (6.54)
\end{aligned}
$$

where $E(\cdot)$ is the expectation operation of a random variable, and $tr(\cdot)$ is the trace operation of a matrix which takes the sum of diagonal elements of the matrix.

Proof: This theorem is a special case of the more general results as proved in [113]. We recap the key steps pertaining to our proof as follows.

We first note the following identities:

$$\begin{aligned} cov(X_1, X_2) &= E(X_1X_2) - E(X_1)E(X_2), & (6.55)\\ E(X_1X_2) &= cov(X_1, X_2) + E(X_1)E(X_2), & (6.56)\\ cov(X^T\Gamma_1X, X^T\Gamma_2X) &= 2tr(\Gamma_1\Gamma_2), & (6.57)\\ cov(X^T\Gamma X, \delta^T X) &= 0. & (6.58)\end{aligned}$$

Therefore, we have

$$\begin{aligned} E(X^T\Gamma X) &= E(tr(\Gamma XX^T)) = tr(\Gamma E(XX^T)) = tr(\Gamma), & (6.59)\\ E(X^T\Gamma X\delta^T X) &= cov(X^T\Gamma X, \delta^T X) + E(X^T\Gamma X)E(\delta^T X) & (6.60)\\ &= 0 + tr(\Gamma)\delta^T E(X) = 0 + 0 = 0, & (6.61)\\ E((X^T\Gamma X)^2) &= cov(X^T\Gamma X, X^T\Gamma X) + E(X^T\Gamma X)^2 & (6.62)\\ &= 2tr(\Gamma^2) + tr(\Gamma)^2. & (6.63)\end{aligned}$$

□

Theorem 3 *Given $\Gamma = \alpha\beta^T + \zeta\eta^T$, we have*

$$\begin{aligned} tr(\Gamma) &= \beta^T\alpha + \eta^T\zeta, & (6.64)\\ tr(\Gamma^2) &= (\beta^T\alpha)^2 + (\eta^T\zeta)^2 + 2(\beta^T\alpha)(\eta^T\zeta), & (6.65)\end{aligned}$$

where α, β, ζ, and η are all vectors.

Proof:

$$tr(\Gamma) = tr(\alpha\beta^T) + tr(\zeta\eta^T) = tr(\beta^T\alpha) + tr(\eta^T\zeta) = \beta^T\alpha + \eta^T\zeta. \tag{6.66}$$

Because

$$\begin{aligned}\Gamma^2 &= \alpha\beta^T\alpha\beta^T + \alpha\beta^T\zeta\eta^T + \zeta\eta^T\alpha\beta^T + \zeta\eta^T\zeta\eta^T \\ &= (\beta^T\alpha)(\alpha\beta^T) + (\beta^T\zeta)(\alpha\eta^T) + (\eta^T\alpha)(\zeta\beta^T) + (\eta^T\zeta)(\zeta\eta^T),\end{aligned}$$

so we have

$$\begin{aligned}tr(\Gamma^2) &= (\beta^T\alpha)tr(\alpha\beta^T) + (\beta^T\zeta)tr(\alpha\eta^T) + (\eta^T\alpha)tr(\zeta\beta^T) + (\eta^T\zeta)tr(\zeta\eta^T) \\ &= (\beta^T\alpha)tr(\beta^T\alpha) + (\beta^T\zeta)tr(\eta^T\alpha) + (\eta^T\alpha)tr(\beta^T\zeta) + (\eta^T\zeta)tr(\eta^T\zeta) \\ &= (\beta^T\alpha)^2 + (\eta^T\zeta)^2 + 2(\beta^T\alpha)(\eta^T\zeta).\end{aligned}$$

□

We then compute the first two moments of T_t in (6.51) as follows:

$$\begin{aligned}E(T_t) &= T_{t0} + \delta_t^T \cdot E(X) - E(X^T\Gamma X) = T_{t0} - tr(\Gamma), \\ E(T_t^2) &= E(T_{t0}^2 + 2T_{t0}\delta_t^T X - 2T_{t0}X^T\Gamma X \\ &\quad + X^T\delta_t\delta_t^T X - 2X^T\Gamma X\delta_t^T X + X^T\Gamma X X^T\Gamma X) \\ &= T_{t0}^2 + 2T_{t0}\delta_t^T E(X) - 2T_{t0}E(X^T\Gamma X) \\ &\quad + E(X^T\delta_t\delta_t^T X) - 2E(X^T\Gamma X\delta_t^T X) + E(X^T\Gamma X X^T\Gamma X) \\ &= T_{t0}^2 + 0 - 2T_{t0}tr(\Gamma) + tr(\delta_t\delta_t^T) - 0 + tr(\Gamma^2) + tr(\Gamma)^2 \\ &= T_{t0}^2 - 2T_{t0}tr(\Gamma) + tr(\delta_t^T\delta_t) + tr(\Gamma^2) + tr(\Gamma)^2 \\ &= (T_{t0} - tr(\Gamma))^2 + \delta_t^T\delta + 2tr(\Gamma^2).\end{aligned}$$

Knowing the first two moments, we compute the mean and variance of T_t in (6.51) as follows:

$$\mu(T_t) = E(T_t) = T_{t0} - tr(\Gamma), \tag{6.67}$$

$$\sigma^2(T_t) = E(T_t^2) - E(T_t)^2 = \delta_t^T\delta + 2tr(\Gamma^2). \tag{6.68}$$

We then approximate (6.51) by the following canonical form that matches its mean and variance with (6.67) and (6.68), i.e.,

$$T_t \;=\; (T_{t0} - tr(\Gamma)) + \sqrt{1 + \frac{2tr(\Gamma^2)}{\delta_t^T \delta}} \cdot \delta_t^T \cdot X. \qquad (6.69)$$

The above approximation is justified because the amount of variation is relatively small compared to the nominal value. Therefore, by matching the mean and variance, we only lose accuracy for the higher order (third moment and above) terms. Experiment results confirm the effectiveness of this approximation.

6.4.2 Canonical Form after Adding a Buffer

If the solution at node t is obtained by adding a buffer at its direct downstream node n, we compute new solutions as follows:

$$\begin{aligned} C_t &= C_{b0} + \eta_b^T \cdot X, & (6.70) \\ T_t &= (T_{n0} - T_{b0} - R_{b0} \cdot C_{n,0}) & \\ &\quad +(\beta_n^T - \beta_b^T - Ln0 \cdot \zeta_b^T - R_{b0} \cdot \alpha_n^T)X - X^T\zeta_b \cdot \alpha_n^T X, & \\ &= T_{t0} + \delta_t^T \cdot X - X^T \Gamma X, & (6.71) \end{aligned}$$

where T_{t0}=$T_{n0} - T_{b0} - R_{b0} \cdot C_{n,0}$, δ_t=$\beta_n - \beta_b - C_{n0} \cdot \zeta_b - R_{b0} \cdot \alpha_n$, and Γ=$\zeta_b \cdot \alpha_n^T$. It is obvious that C_t in (6.70) is already a canonical form, but T_t in (6.71) is not due to the quadratic term $X^T\Gamma X$. By using the similar technique as discussed above, we approximate (6.71) via the same canonical form as shown in (6.69), but with its own T_{t0}, δ_t, Γ and X.

6.4.3 Canonical Form after Merging Two Solutions

If the solution at node t is obtained by merging two solutions from its two subtrees rooted at nodes m and n, respectively, we compute the new solutions C_t

by

$$C_t \;=\; (C_{n0} + C_{m0}) + (\alpha_n^T + \alpha_m^T) \cdot X, \tag{6.72}$$

which is already a first order canonical form. To express T_t after the minimum operation still to be a first order canonical form, we resort to the approximation technique as discussed in section 2.4.3, and we rewrite the equation of (2.32) as

$$T_t = T_{t0} + \beta_t^T \cdot X. \tag{6.73}$$

By replacing (6.50) and (6.69) with (6.1) and (6.2), (6.70) and (6.69) with (6.3) and (6.4), and (6.72) and (6.73) with (6.5) and (6.6), respectively, we have replaced all key operations needed in a deterministic buffer insertion algorithm with its respective variation counterpart. Therefore, we obtain a variation-aware buffer insertion algorithm. Moreover, because we always keep solutions in first order canonical form after each operation, we can apply the same technique recursively to compute all new solutions while traversing the routing tree bottom up.

Note that the same approximation method via moment matching techniques is not restricted to the first-order canonical model, and it can be extended to handle other nonlinear (like quadratic) forms as well.

6.5 Variation Aware Pruning

6.5.1 Review of Deterministic Pruning

The major complexity of dynamic programming based buffer insertion lies in the merging of two sets of solutions obtained from two different sub-trees. In general, the total number of possible combinations for merging is $n \cdot m$, where n and m

are the number of solutions from two sub-trees, respectively. If all combinations are kept at each merging node, the number of solutions will grow exponentially towards the root.

To avoid this problem, [92] proposed to define the *dominance relationship* (or *pruning rule*) between two solutions such that solution (C_1, T_1) dominates solution (C_2, T_2) if condition $C_1 < C_2$ and $T_1 > T_2$ are satisfied. In other words, solution (C_2, T_2) is redundant and can be removed. [92] proved that instead of $n \cdot m$, there would be no more than $n + m$ number of solutions after pruning.

Even though pruning helps reduce the total number of solutions, in general, we still have to pay the price of $O(n{\cdot}m)$ at each node in order to obtain all possible combinations for merging. In deterministic buffer insertion, such a procedure is further reduced to $O(n + m)$ by using a merge sort like operation on the two sets of already sorted solutions, i.e., the merging and pruning operations can be done simultaneously instead of using two separate operations as shown in Figure 6.1. Based upon the above two linear operations on pruning and merging, [92, 58] proved that by keeping only dominating solutions at every node, the dynamic programming based algorithm can solve the buffer insertion problem in $O(N^2)$ time without losing optimality, where N is the number of possible buffer locations. When there are B types of buffers in the library, [56] proved that the deterministic buffer insertion problem can be solved optimally in $O(B \cdot N^2)$.

6.5.2 Two-sided Threshold Based Pruning

A straight-forward way to extend the conventional dominance relationship between two solutions in the presence of process variation is as follows: solution (C_1, T_1) is said to dominate solution (C_2, T_2) if condition $P(C_1 \leq C_2) = 1$ and $P(T_1 \geq T_2) = 1$ are satisfied. In other words, solution (C_1, T_1) always results in a larger required arrival time but with a less loading capacitance when compared to solution (C_2, T_2). The physical interpretation of this criterion is well understood. However, there are two problems when it comes to practical implementation. First, for a continuous JPDF for two random variables (C_t and T_t), the domain is usually defined over the whole feasible region: i.e., $0 \leq C_t \leq \infty$ and $0 \leq T_t \leq \infty$. Therefore, it is almost impossible to satisfy the conditions of $P(T_1 \geq T_2) = 1$ and $P(C_1 \leq C_2) = 1$ for any two given solutions. Second, assuming the first problem can be solved, there is no guarantee that the number of solutions after pruning will increase polynomially. Chances are that it will, most likely, grow exponentially, and this has been experimentally confirmed by [47]. In the following, we propose a new set of pruning rules that are in line with designers' intuition.

We recognize that for designers, there is always a design goal in their minds when they compare different design alternatives. Based upon the above observations, we give the following definition of dominance relationship between two solutions, which is closely related to designers' willingness to accept uncertainty for a given design.

Recall that the $(100\alpha)^{th}$ percentile of a p.d.f. $f(x)$ is a number π_α such that the area under $f(x)$ to the left of of π_α is α [43]. That is,

$$\alpha = \int_{-\infty}^{\pi_\alpha} f(x)dx. \tag{6.74}$$

In other words, π_α gives a measure of designers preference for certainty in choosing the design parameter x in the presence of variations, such that the final design would have x less than π_α with $(100\alpha)\%$ certainty.

Suppose designers choose π_{α_l} and π_{α_u} as C_t's two percentiles with $0 \leq \alpha_l < \alpha_u \leq 1$, and π_{β_l} and π_{β_u} as T_t's two percentiles with $0 \leq \beta_l < \beta_u \leq 1$, which reflect designers preference for certainty in choosing different solutions (C_t, T_t) to

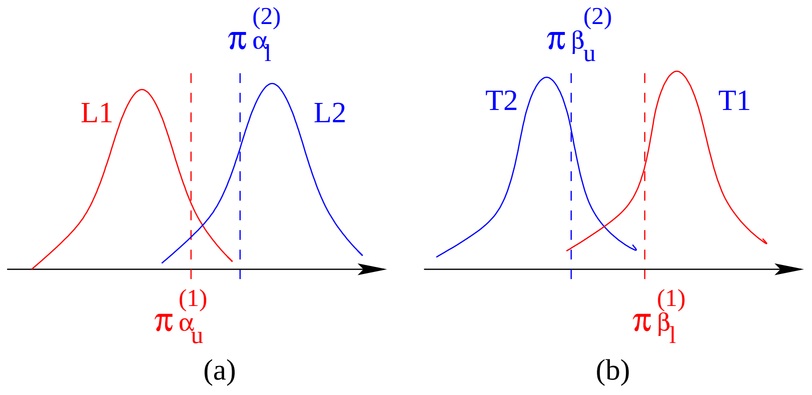


Figure 6.2: Graphic interpretation of dominance relationship between (L_1, T_1) and (L_2, T_2), where (a) refers to (6.75) and (b) refers to (6.76).

the BIPV problem. Then solution (C_1, T_1) is said to dominate solution (C_2, T_2) if the following conditions are satisfied:

$$\pi_{\alpha_u}^{(1)} < \pi_{\alpha_l}^{(2)} \tag{6.75}$$

$$\pi_{\beta_l}^{(1)} > \pi_{\beta_u}^{(2)} \tag{6.76}$$

Another way to look at this dominance relationship is that solution (C_1, T_1) has a high probability of producing solutions with a larger required arrival time and a smaller loading capacitance. A graphical interpretation of this dominance relationship is shown in Fig. 6.2. Knowing α_l, α_u, β_l, and β_u, we can compute π_{α_l}, π_{α_u}, π_{β_l}, and π_{β_u} according to (6.74), which requires us to know the PDF of C_t and T_t, respectively. As we already know the JPDF of (L_t, T_t), the PDF of L_t and T_t can be computed as two marginal PDF's of the JPDF, respectively [43].

The choice of α_l, α_u, β_l and β_u reflects the trade-off between tolerance of variation risks and run-time efficiency.

6.5.3 Transitive Closure Based Pruning

We observe that for deterministic buffer insertion, the linear time operations for pruning and merging are made possible because of the following two properties: (1) for any two given solutions, there exists an *ordering property* between them so that comparing them is always possible, i.e., T_1 is either greater than T_2 or less than T_2; (2) there exists a *transitive ordering property* between solutions, i.e., if $T_1 > T_2$ and $T_2 > T_3$, then $T_1 > T_3$ (similarly, if $C_1 < C_2$ and $C_2 < C_3$, then $C_1 < C_3$). If we ensure that the above two properties hold for solutions considering process variations, we can achieve similar linear time complexity for both pruning and merging. In the following, we propose a new variation aware pruning rule that enables us to keep both merging and pruning operations in linear complexity even in the presence of process variations.

We first extend the deterministic dominance relation between (C_1, T_1) and (C_2, T_2) by enforcing:

$$P(C_1 \leq C_2) = 1, \tag{6.77}$$

$$P(T_1 \geq T_2) = 1. \tag{6.78}$$

The physical interpretation of this extension is that solution (C_1, T_1) has 100% propability (almost always) to result in a larger required arrival time but with a less loading capacitance when compared to solution (C_2, T_2). We have the following Lemma[1]:

Lemma 1 *Given T_1, T_2 and T_3 as three* **dependent** *random variables with arbitrary distributions, if $P(T_1 > T_2) = 1$, $P(T_2 > T_3) = 1$, then $P(T_1 > T_3) = 1$.*

[1]For simplicity, we only use T to illustrate the idea. It is understood that same results can be applied to L as well.

Proof: Let $X = T_1 - T_2$ and $Y = T2 - T_3$ and the JPDF of X and Y be $f(x, y)$. As $P(T_1 > T_2) = P(X > 0) = \int_0^{+\infty} x \int_{-\infty}^{+\infty} f(x, y)dy = 1$, we have $\int_{-\infty}^{0} dx \int_{-\infty}^{+\infty} f(x, y)dy = 1$ - $\int_0^{+\infty} dx \int_{-\infty}^{+\infty} f(x, y)dy = 1$ - $1 = 0$. Because $f(x, y) \geq 0$ for all x and y, we have $f(x, y) = 0$ for $x < 0$. Similarly, from $P(T_2 > T_3) = P(Y > 0) = 1$, we have $f(x, y) = 0$ for $y < 0$. Therefore, we have $\int_0^{+\infty} dy \int_0^{+\infty} f(x, y)dx = 1$. Then $P(T_1 > T_3) = P(X + Y > 0) = \int_{-\infty}^{+\infty} dy \int_{-y}^{+\infty} f(x, y)dx = \int_0^{+\infty} dy \int_{-y}^{+\infty} f(x, y)dx \geq \int_0^{+\infty} dy \int_0^{+\infty} f(x, y)dx = 1$. As we know $P(T_1 > T_3) \leq 1$, we must have $P(T_1 > T_3) = 1$. □

Lemma 1 shows that comparison between T_1 and T_2 based upon $P(T_1 > T_2)$=1 enforces the transitive ordering property between solutions. However, for any two given solutions, it is not always possible to compare them. Moreover, in practice, such a 100% probability requirement is too restrictive. Therefore, we relax such a requirement by adding two parameters such that solution (C_1, T_1) is said to dominate solution (C_2, T_2) if the following two conditions hold:

$$P(C_1 < C_2) \geq \overline{p_C} = 0.5, \tag{6.79}$$

$$P(T_1 > T_2) \geq \overline{p_T} = 0.5, \tag{6.80}$$

In other words, it is likely that C_1 is less than C_2 while T_1 is greater than T_2 in the probabilistic sense. We call the pruning rule as defined by (6.79) and (6.80) as *transitive-closure* based pruning rule.

We have the following lemma:

Lemma 2 *Given T_1 and T_2 as two* **dependent** *random variables with arbitrary distributions, we have either $P(T_1 > T_2) \geq 0.5$ or $P(T_1 < T_2) \geq 0.5$.*

Proof: The proof follows directly from the fact that $P(T_1 > T_2)$+$P(T_1 < T_2)$=1. □

Lemma 2 shows that comparison based upon $P(T_1 > T_2) > 0.5$ results in a proper ordering between two random solutions T_1 and T_2. Therefore, the remaining problem is whether or not such a pruning rule preserves the transitive ordering property between solutions. Unfortunately, for arbitrary distributions, we can show that it does not preserve the transitive ordering property in general. In other words, for arbitrary random distributions, when $\overline{p_C} = 1$ and $\overline{p_T} = 1$, the transitive ordering property holds but not necessary the ordering property; when $\overline{p_C} = 0.5$ and $\overline{p_T} = 0.5$, the ordering property holds but not necessary the transitive ordering property. Therefore, to have the two proprieties to hold simultaneously, we may have to impose some restrictions on the type of distributions for those random solutions. In the following, we prove that when the random solutions follow a joint normal distribution, both properties indeed hold simultaneously, which is stated in the following Lemma:

Lemma 3 *Given T_1, T_2 and T_3 as three* **dependent** *random variables with joint normal distributions, if $P(T_1 > T_2) > 0.5$, $P(T_2 > T_3) > 0.5$, then $P(T_1 > T_3) >$* 0.5

Proof: To see this, we assume both T_1 and T_2 are normal, and we have the following closed form to evaluate the probability of $T_1 > T_2$ according to [13],

$$P(T_1 > T_2) = \Phi(\frac{\mu_{T_1} - \mu_{T_2}}{\sigma_{T_1,T_2}}), \tag{6.81}$$

where Φ is the cumulative density function (CDF) of a standard normal distribution; μ_{T_1}, and μ_{T_2} are the mean values of T_1 and T_2, respectively; and σ_{T_1,T_2} can be computed by

$$\sigma_{T_1,T_2} = (\sigma_{T_1}^2 - 2 \cdot \rho_{T_1,T_2}\sigma_{T_1}\sigma_{T_2} + \sigma_{T_2}^2)^{1/2}, \tag{6.82}$$

where $\sigma_{T_1}^2$ and $\sigma_{T_2}^2$ are variance of T_1 and T_2, respectively; and ρ_{T_1,T_2} is the correlation coefficient of T_1 and T_2.

Because any CDF function is a non-decreasing function, and for the standard normal distribution $\Phi(0) = 0.5$, then we have $\Phi(x) > 0.5$ for any $x > 0$. Therefore, to have $P(T_1 > T_2) > 0.5$ is equivalent to have $\frac{\mu_{T_1} - \mu_{T_2}}{\sigma_{T_1,T_2}} > 0$. Because σ_{T_1,T_2} is positive according to (6.82), hence we have $\mu_{T_1} > \mu_{T_2}$. Knowing $P(T_1 > T_2) > 0.5$ and $P(T_2 > T_3) > 0.5$, we have $\mu_{T_1} > \mu_{T_2}$ and $\mu_{T_2} > \mu_{T_3}$. Therefore, we have $\mu_{T_1} > \mu_{T_3}$, which is equivalent to $P(T_1 > T_3) > 0.5$. $\square$

According to the proof of the above lemma, it is easy to see that the following equivalent statement also holds:

Lemma 4 *Given T_1 and T_2 as two* **dependent** *random variables with joint normal distribution, for $P(T_1 > T_2) > 0.5$ to hold, it is necessary and sufficient to have $\mu_{T_1} > \mu_{T_2}$ with μ_{T_1} and μ_{T_2} being the mean for T_1 and T_2, respectively.*

Based upon the above discussion, we have the following theorem regarding the complexity of the dynamic programming based variation aware buffer insertion algorithm:

Theorem 4 *By utilizing the transitive closure based pruning rule as defined in (6.79) and (6.80) with $\overline{p_C}$=0.5 and $\overline{p_T}$=0.5, the variation aware buffer insertion problem can be solved in $O(B \cdot N^2)$, where B is the number of buffer types in the library, and N is the number of legal buffer positions.*

Proof: According to Lemma (2) and (3), we can compare two random solutions and order them much the same way as in the deterministic approach. Therefore, following similar arguments as in [92, 58] and [56], we conclude that our variation aware buffer insertion algorithm under the transitive closure based pruning rule has the same complexity as the deterministic algorithm, which is $O(B \cdot N^2)$. $\square$

Next we discuss the extension of the above transitive closure pruning rule for other choices of $\overline{p_C}$ and $\overline{p_T}$ and see how the two desired properties are affected.

In fact, we have the following theorem which proves that for $\overline{p_T}$ (or $\overline{p_L}$) between 0.5 and 1, the transitive ordering property always hold.

Theorem 5 *Given T_1, T_2 and T_3 as three* **dependent** *random variables with joint normal distributions, if $P(T_1 > T_2) > \overline{P_T}$, $P(T_2 > T_3) > \overline{P_T}$, then $P(T_1 > T_3) > \overline{P_T}$ for any constant $\overline{P_T}$ between 0.5 and 1, i.e., $0.5 \leq \overline{P_T} \leq 1$.*

Proof: Define $X = T_1 - T_2$ and $Y = T_2 - T_3$, then we have $X + Y = T_1 - T_3$. Therefore, $P(T_1 > T_2)$=$P(X > 0)$, $P(T_2 > T_3)$=$P(Y > 0)$, and $P(T_1 > T_3)$=$P(X + Y > 0)$.

Because T_1, T_2 and T_3 are joint normal, then X and Y are also normal. Denote the PDF of X as $N(\mu_x, \sigma_x)$ and the PDF of Y as $N(\mu_y, \sigma_y)$, where μ_x and σ_x (similarly μ_y and σ_y) are the mean and standard deviation for X (similarly Y), respectively. Hence we can obtain the PDF of $X + Y$, which is also a normal distribution, as $N(\mu_x + \mu_y, \sqrt{\sigma_x^2 + \sigma_y 2 + 2\rho\sigma_x\sigma_y})$ with ρ being the correlation coefficient between X and Y. We have

$$P(X > 0) = P(\frac{X - \mu_x}{\sigma_x} > -\frac{\mu_x}{\sigma_x}) = 1 - \Phi(-\frac{\mu_x}{\sigma_x}), \tag{6.83}$$

where Φ is the CDF of a standard normal distribution. According to the property of the standard normal distribution, we have $\Phi(-t) = 1 - \Phi(t)$, therefore we have

$$P(T_1 > T_2) = P(X > 0) = \Phi(\frac{\mu_x}{\sigma_x}). \tag{6.84}$$

As we already know $P(T_1 > T_2) > \overline{P_T}$, we hence have $\Phi(\frac{\mu_x}{\sigma_x}) > \overline{P_T}$. Since any CDF function is also a non-decreasing function, we have

$$\frac{\mu_x}{\sigma_x} > \bar{t}, \tag{6.85}$$

where $\Phi(\bar{t}) = \overline{P_T}$. Moreover, for $0.5 \leq \overline{P_T} \leq 1$, we have $\bar{t} > 0$. Similarly, we have

$$\frac{\mu_y}{\sigma_y} > \bar{t}. \tag{6.86}$$

From (6.85) and (6.86), we have

$$\begin{aligned}
\mu_x + \mu_y &> (\sigma_x + \sigma_y)\bar{t}, & (6.87)\\
\frac{\mu_x + \mu_y}{\sqrt{\sigma_x^2 + \sigma_y^2 + 2\rho\sigma_x\sigma_y}} &> \frac{(\sigma_x + \sigma_y)\bar{t}}{\sqrt{\sigma_x^2 + \sigma_y^2 + 2\rho\sigma_x\sigma_y}}. & (6.88)
\end{aligned}$$

Because $-1 \leq \rho \leq 1$, it is easy to show that

$$\frac{(\sigma_x + \sigma_y)}{\sqrt{\sigma_x^2 + \sigma_y^2 + 2\rho\sigma_x\sigma_y}} \geq 1. \quad (6.89)$$

As $\bar{t} > 0$, by multiplying both sides of (6.89) by $\bar{t} > 0$ and then combining it with (6.88), we have

$$\frac{\mu_x + \mu_y}{\sqrt{\sigma_x^2 + \sigma_y 2 + 2\rho\sigma_x\sigma_y}} > \frac{(\sigma_x + \sigma_y)\bar{t}}{\sqrt{\sigma_x^2 + \sigma_y 2 + 2\rho\sigma_x\sigma_y}} \geq \bar{t}. \quad (6.90)$$

Therefore, by the fact that Φ is a non-decreasing function and (6.90), we finally have

$$\begin{aligned}
P(T_1 > T_3) &= P(X + Y > 0)\\
&= \Phi(\frac{\mu_x + \mu_y}{\sqrt{\sigma_x^2 + \sigma_y^2 + 2\rho\sigma_x\sigma_y}})\\
&> \Phi(\bar{t}) = \overline{P_T}. & (6.91)
\end{aligned}$$

□

Having proved the transitive ordering property, it is attempting to see whether the ordering property as in Lemma (2) also holds for different choice of $\overline{p_C}$ and $\overline{p_T}$. Unfortunately, such an extension is not true in general. Therefore, results of this extension are of mainly theoretical interests at this point. One of our future work plans is to leverage these theoretical results for other CAD applications while taking into account the process variation effects.

6.6 Experiment Results

6.6.1 Experiment Setting

Two sets of benchmarks are obtained from the public domain for our experiments [83]. The characteristics of the benchmarks are shown in Table 6.1.

Table 6.1: Characteristics of benchmarks.

Bench	Sinks	Buffer Positions
p1	269	537
p2	603	1205
r1	267	533
r2	598	1195
r3	862	1723
r4	1903	3805
r5	3101	6201

Because of the lack of access to the real wafer data, we derive the process variation data based upon the literature that addresses similar process variation issues but in the context of statistical timing analysis [1]. In our experiment, the 65nm BSIM technology is assumed. We budget the random device variation, inter-die global variation, intra-die spatial variation, and interconnect variation all to be 5% of its nominal value, respectively. Moreover, to model the spatial variation, we divide the chip layout into different grids with the length of each grid as 500μm. For devices located at a particular grid, their characteristics are affected by a set of nearby grids. We distribute the budgeted 5% spatial variation into different regions with the sensitivity of each region forming an isotropic stationary Gaussian process with a value that tapers off at a distance of about 2mm.

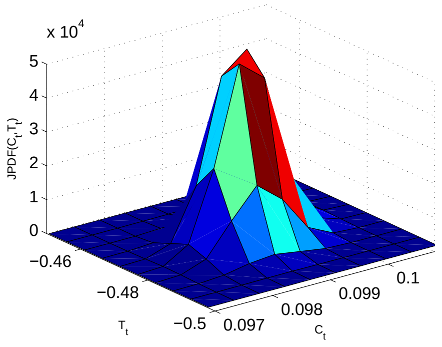
6.6.2 BIPV Algorithms Comparison

Because of the complication resulting from nonlinear operations in computing C_t and T_t, in section 6.3, we have employed the numerical integration method to compute the distribution of C_t and T_t explicitly. To make the numerical method attractable, we further assume that the variations in devices and interconnects are independent, hence ignoring their shared inter-chip global variation and intra-chip spatial correlations. The PDFs of C_t and T_t are computed numerically via hyper-dimensional integration.

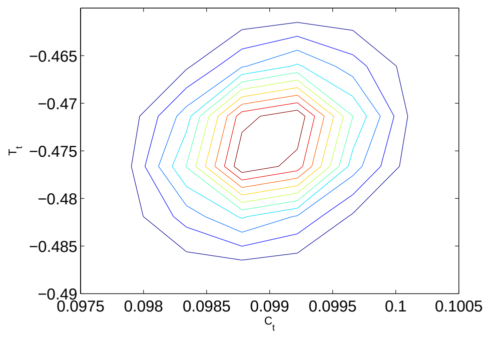
In theory, the results obtained through numerical integration are exact, and are more accurate even than the method of Monte Carlo simulation, provided that there is no numerical error with the integration. But numerical integration is notoriously difficult to implement with reasonable accuracy, and the computation complexity is high. For example, our early implementation [105] of the JPDF-based BIPV algorithm is done in C language, and we employed the simplest trapezoidal method for numerical integration. However, even this least sophisticated numerical integration method requires hours and hours of CPU time to solve the BIPV problem for a small routing tree with 9 pins (8 sinks). Moreover, we later found that the results from this simple numerical integration has very rough edges due to accumulated numerical errors, which renders results after integration unstable. Therefore, for practice concern, the algorithm as discussed in section 6.3 is not feasible. But the derivation as shown in section 6.3 is purely general, and we do not make any assumptions on the specific distributions (such as normal distributions) for all random variables. Therefore, its existence is still of theoretic interest, and we will show one of its uses in the following.

To avoid the difficulties associated with numerical integration, we employ the first-order canonical form to represent both device and interconnect variations, thus implicitly assuming that all random variables follow a normal distribution. Based upon the canonical form, we have presented another BIPV algorithm in section 6.4. One advantage of using canonical forms to represent all quantities is that the correlation between them is implicitly considered, hence we can also consider both global variation and spatial correlations. To simplify the computation while still handle the nonlinear operations on two correlated distributions, we approximate the new canonical form for C_t and T_t as another normal distributions as discussed in section 6.4.

Among the two nonlinear operations (minimum and multiplication), the minimum operation on two normal distributions have been shown to be reasonably accurate if we approximate it as another normal distribution. This has been well studied in the statistical timing analysis community, for example, [16, 94]. In contrast, it is not well studied regarding the accuracy of the approximation of results from multiplication of two normal distributions as another normal distribution. In the following, we use the theoretic results developed in section 6.4 to study the accuracy issue numerically.



(a) JPDF of (C_t, T_t)



(b) Contour plot of (C_t, T_t)

Figure 6.3: Numerical example for the JPDF of (C_t, T_t).

One numerical integration example by following the derivation results as shown in equation (6.17) is plotted in Figure 6.3.

Under the canonical model, the JPDFs of (C_w, R_w) and (C_n, T_n), i.e., $h(C_w, R_w)$ and f_{C_n,T_n}, all follow a bivariate normal distribution. To use equation (6.17), we further assume that there is no correlation between $h(C_w, R_w)$ and f_{C_n,T_n}. To achieve more stable numerical results, we implemented the two-dimensional numerical integration in Matlab by using its "dlbquad" function, which in term calls "quad" function for one-dimensional numerical integration. The function "quad" implements the recursive adaptive Simpson quadrature algorithm [81], which is far more sophisticated than what we have implemented (trapezoidal method) in C [105]. From Figure 6.3, we observe that the JPDF of (C_t, T_t) after multiplication operations can be reasonably approximated as another bivariate normal distributions. From Figure 6.3, we also observe that the roughness of JPDF computation due to numerical approximation also shows up even in our Matlab implementation. A more subtle observation from Figure 6.3 is that there seems to have a singular point at the mean. This observation is in line with what has been reported in [96]. We shall note that in order to obtain the rough result as shown in Figure 6.3 in Matlab, we need more than four hours CPU time on a Pentium 1.7GHz machine with 516 mega-bytes memory.

6.6.3 Runtime Comparison

We compare the efficiency of the two different pruning rules in the following. And we use the same first-order canonical form to represent all quantities while solving the BIPV problem.

The two-sided threshold based pruning rule, despite of its intuitive definition, is computationally expensive to use for large designs, because it is not guaranteed that the number of solutions after pruning is linearly. We denote the algorithm under the two-sided threshold based pruning rule as T2P, and the one under the transitive-closure pruning rule as vawBuf.

Table 6.2: Runtime comparison in seconds.

Bench	detBuf	T2P [105]	vawBuf	Speedup
p1	0.00	25.4	1.0	25.4×
p2	0.01	-	4.3	-
r1	0.00	-	3.6	-
r2	0.00	-	15.0	-
r3	0.02	-	27.5	-
r4	0.04	-	88.9	-
r5	0.08	-	195.8	-

In Table 6.2, we report the runtime for both algorithms based upon the benchmarks we have tested. According to Table 6.2, we can see that the implemented T2P algorithm can handle much larger benchmarks than what was originally reported in [105], and the largest tested benchmark is $p1$ with 269 sinks. This improvement is mainly due to the avoidance of computing JPDF explicitly. However, we still fail to use the improved algorithm of [105] to run larger benchmarks. In fact, for the rest of tested benchmarks, it fails due to exceeding either memory capacity ($2G$) or tolerable time limit (4 hours in our setting). This observation is expected, because the two-side threshold based pruning rule only imposes partially ordering between solutions, rendering the complexity of merging and pruning very high.

In contrast, by using the transitive closure pruning rule, our vawBuf algorithm can easily run through all benchmarks and for the largest benchmark $r5$, the runtime is about 3 minutes. This significant runtime speedup is achieved because the transitive closure pruning rule as discussed in section 6.5 enforces a relatively strict ordering between solutions, thus enables an efficient implementation for both merging and pruning.

We also report the runtime for the conventional deterministic buffering algorithm (detBuf) in Table 6.2. We find the our vawBuf algorithm runs slower than detBuf, but this is expected because of the additional computation needed to handle correlated process variations. If we plot the runtime of vawBuf versus the number of legal buffer positions, however, we can see that the runtime of our vawBuf algorithm scales almost linearly with respect to the benchmark size.

6.6.4 Timing Optimization

Enabled with the efficient implementation of buffer insertion considering process variations, we run our buffer insertion algorithm on the benchmarks for RAT optimization and study the effect of process variation on buffered interconnect design.

We first verify the accuracy of our approach in predicting the timing distribution via Monte Carlo simulation. For one buffered routing tree with inserted buffers, we run our vawBuf algorithm under the first-order process variation model to compute the delay distribution at the root, which is approximated as a normal distribution. For the same buffered routing tree, we use Monte Carlo simulation to obtain the delay distribution at the root. Fig. 6.4 shows both the PDF computed from our algorithm and the Monte Carlo simulated PDF for one of the benchmarks ($r5$). We see that the distribution from Monte Carlo simulation almost follows a normal distribution as predicted by our vawBuf algorithm. Moreover, our algorithm is reasonably accurate in predicating the distribution compared to that from Monte Carlo simulation, and tends to be conservative.

Having said that, we shall point out that in general, if we only compute the product of two normal distributions once, the results can be very different from

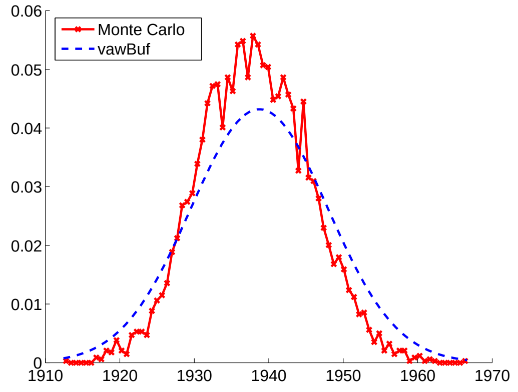


Figure 6.4: Delay distributions at the root as predicted by our vawBuf algorithm and the Monte Carlo simulation.

a normal distribution, and only under some conditions this approximation is acceptable. A thorough treatment of this subject can be found in [96], in which the theoretic derivation of those conditions are given. However, when more and more such approximation operations are applied, by central limit theorem, the final results would still look like a normal distribution. This conclusion has also been empirically observed in both our experiments as shown in Figure 6.4 and the work by [96].

We further compare the solution quality between the conventional deterministic buffering algorithm and our variation aware buffer algorithm. For fair comparison, we run Monte Carlo simulation as a golden test to obtain the delay distribution at the root for both cases. One such example based on benchmark $r5$ is shown in Fig. 6.5. We employ the 3-sigma delay in the distribution as a figure-of-merit to compare the results. The 3-sigma delay defines the worst case delay in the distribution such that almost all manufactured designs ($> 99.8\%$) will have a smaller delay than the 3-sigma delay even in the presence of process variation. By comparing the 3-sigma delay in Fig. 6.5, we observe that the

Table 6.3: Comparison between deterministic buffering (detBuf) and variation-aware buffering (vawBuf) based on Monte Carlo simulations. The yield of vawBuf is 100%.

Bench	detBuf				vawBuf		
	Buffer	Mean (%)	3-sigma Delay(%)	Yield Loss	Buffer (%)	Mean	3-sigma Delay
p1	58	2374	2403	0%	60 (3.3%)	2375	2403
p2	149	3161	3203	0%	156 (4.5%)	3161	3204
r1	59	772	790	0%	65 (9.2%)	771	790
r2	112	1109 (1.7%)	1128 (1.5%)	35.3%	135 (17%)	1090	1111
r3	173	1127 (0.7%)	1147 (0.5%)	1.6%	188 (8%)	1119	1142
r4	320	1700 (1.5%)	1723 (1.4%)	54.9%	374 (14.4%)	1674	1699
r5	544	1958(1%)	1986 (1.0%)	17.9%	608 (10.5%)	1938	1966
Avg		0.7%	0.6%	15.7%	9.6%		

3-sigma delay obtained from vawBuf is much better than that from detBuf.

If we further define the 3-sigma delay of vawBuf as 100% timing yield point, we can then use it to find the timing yield for the deterministic design. The difference between these two indicates the potential timing yield loss as shown in Fig. 6.5. We report the comparison between the deterministic buffering algorithm and our variation aware buffering algorithm in Table 6.3.

According to Table 6.3, we can see that compared to the deterministic buffering, our variation aware buffer insertion improves the 3-sigma timing by 0.6% on average, and the parametric timing yield by more than 15%, respectively. This highlights the importance of developing efficient algorithms for IC designs to actively attack process variation effects.

Interestingly, we observe that, for some relatively small benchmarks, the improvement for 3-sigma delay and yield is almost negligible, while for some large benchmarks, the improvement is quite significant. These observations to some degree agree with what has been reported in [25] for infinity long two-pin nets.

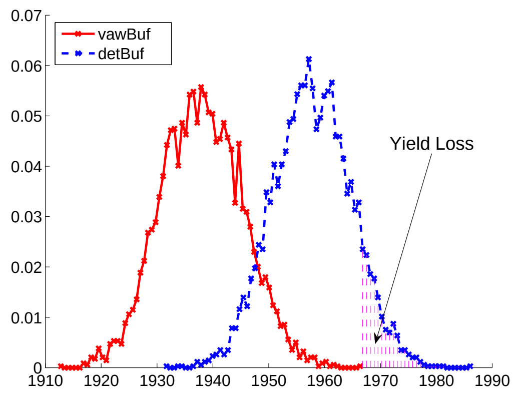


Figure 6.5: Delay distribution comparison between our vawBuf algorithm and the deterministic buffering algorithm (detBuf).

There is a need, however, to look into the theoretical explanation to the above observations.

We also report the number of buffers inserted for both algorithms in Table 6.3. We see that our variation aware buffering algorithm tends to put more buffers into the design to combat the correlated process variations than the deterministic worst case design.

6.7 Conclusion and Discussion

A novel algorithm for buffer insertion considering process variation has been proposed. We have developed an efficient approximation technique to handle the correlated process variations under nonlinear operations. We have also proposed a provable transitive closure pruning rule that enables efficient implementation of the buffer insertion algorithm considering correlated process variations. We have applied the algorithm for timing optimization and concluded that process variation must be considered to achieve optimal designs for parametric timing yield, and buffer insertion considering correlated variation improves the timing yield by more than 15% on average.

CHAPTER 7

Criticality: A Variation-aware Metric for Optimization

Chips manufactured in 90 nm technology have shown large parametric variations, and a worsening trend is predicted. These parametric variations make circuit optimization difficult since different paths are frequency-limiting in different parts of the multi-dimensional process space. Therefore, it is desirable to have a new diagnostic metric for robust circuit optimization. *Criticality probabilities*, both the conditional and unconditional versions, are such a metric. This chapter presents a novel algorithm to compute the criticality probability of every edge in the timing graph of a design with linear complexity in the circuit size. Using industrial benchmarks, we verify the correctness of our criticality computation via Monte Carlo simulation. We also show that for large industrial designs with 442,000 gates, our algorithm computes all edge criticalities in less than 160 seconds. The high accuracy and fast speed of the algorithm warrant future application of criticality probability for robust circuit optimization.

7.1 Introduction

As technology nodes shrink to 90 nm and below, it becomes more difficult to manufacture chips with guaranteed parametric timing yield due to substantial increase of process variations [93]. If these effects are not considered properly, the potential for silicon failure is high, and the associated cost for a design re-spin is prohibitive. Therefore, statistical methods have recently attracted attention as a promising approach to improve parametric timing yield.

In the deterministic approach, a circuit is optimized for a single combination of process parameters. As a result of manufacturing, however, we receive chips corresponding to various combinations of process parameters. Deterministic optimization cannot guarantee that the chip satisfies design requirements for all or most of these combinations. Statistical optimization [37, 4, 20] is targeted to solve this problem. The goal of statistical optimization is to maximize yield while satisfying timing, area, power and other design constraints. This goal can be achieved only by considering the whole space of process variations. Parameterized statistical static timing analysis (SSTA) [16, 94] provides that kind of exploration, computing the circuit delay as a function of process parameters. Unfortunately, knowing circuit delay is not enough. The optimization needs more detailed guidance to select circuit fragments requiring improvement. The timing analyzer drives deterministic optimization by identifying a critical path. In the presence of process variations, the critical path is not unique because different paths can be critical in different regions of the process space.

There is a useful logical extension of the concept of a critical path called criticality probability [94]. Similar concepts were used in the context of PERT networks where it was called criticality index [28]. The criticality of a path is the probability of manufacturing a chip in which the path is critical. The higher the criticality, the more important it is to improve the timing characteristics of the path. In [55] it is shown that the criticality of a path is equal to the sensitivity of the mean of the circuit delay with respect to the mean of the path delay. But we believe the concept of criticality is more convenient than sensitivity. First, the definition of criticality is simpler, clearer and more intuitive, as it naturally extends the conventional concept of critical path. Second, criticality computation, as we will show in this chapter, does not require complicated chain ruling as was used for sensitivity computation [55]. Third, criticality allows us to define a new concept, called conditional criticality, which gives us information on how critical a gate is among those manufactured chips that fail timing. Conditional criticality is more useful for optimization since the optimization should focus only on chips that violates timing constraints. Moreover, conditional criticality can be used for other applications such as circuit synthesis and test generation.

There was an attempt to compute criticalities by multiplying tightness probabilities [94]. This approach assumes that the tightness probabilities represent independent events and can be multiplied, which is not correct due to globally correlated parameters and path reconvergence. As was shown in [37], however, even such inaccurate criticalities can be useful for guiding optimization.

In this chapter, we develop an accurate and efficient technique to compute criticalities for all timing edges in the context of parameterized SSTA. We use the same graph cutset concept [26] as [20] does, although both came to the same idea independently. The difference is that [20] used cutset for timing yield gradient computation, and this is achieved by perturbing PDFs of timing edges. In contrast, we focus on efficient computation of criticality without perturbation. Our computation uses only efficient operations of statistical minimum and maximum, and tightness probability computation. We do not make any assumptions about independence approximations on tightness probabilities. The proposed algorithm has linear complexity in the number of timing edges. We propose a new concept of conditional criticality and develop an efficient technique for its computation. We implement our algorithms in an industrial statistical timing analyzer and verify the correctness of our implementation by Monte Carlo simulation. We show that for an industrial ASIC with 442,000 gates, our algorithm computes all edge criticalities in less than 160 seconds. Results of this chapter have been reported in [108, 109].

7.2 Motivation: the Need for a Variation-aware Metric

Timing closure in the presence of process variations is a nightmare. Traditional incremental timing is run at a chosen process corner to guide the optimization, but timing sign-off requires either multi-corner timing or statistical timing. If sign-off timing is not achieved, the optimization continues with some annotations from the sign-off timing as to which cones of logic need improvement. The trouble is that as soon as timing is closed at process corner A, violations are seen at process corner B, and vice versa, and the target remains elusive.

Traditional timing provides two important diagnostics: (1) the identity of the critical path, which is the location of the most "bang for the buck" during optimization; (2) timing slack or margin. In the presence of process variations, unfortunately, neither of these diagnostics is useful. Each point in the process space can have a unique critical path, so in reality there is a set of critical paths, each of which has a non-zero probability of being critical. For illustration purposes, Fig. 7.1 shows a 2-dimensional space with process parameters P_1 and P_2, along with contours of equal probability. Realistic situations will have a much higher dimensionality. The X in the figure shows the process corner at which timing is conducted in order to guide optimization, at which a particular path is found to be the most critical. As show in Fig. 7.1a, this path may be critical in only a small portion of the process space around X, or it may be critical in much

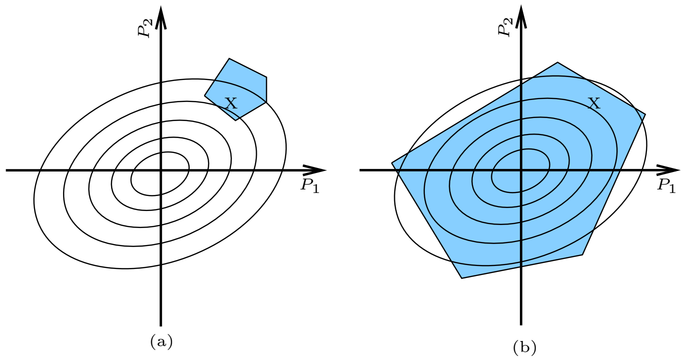


Figure 7.1: Criticality of a critical path in (a) a small region of the process space; (b) a large region of the process space.

of the high-probability process space as shown in Fig. 7.1b. Thus it is unclear from deterministic timing whether or not it is important to improve the timing characteristics of this path.

One quickly comes to the conclusion that one requires either multi-corner slack or statistical slack to guide optimization. Multi-corner slacks cannot be obtained incrementally and an exponential number of corners in the process space makes this procedure prohibitive. Instead, statistical slack is more conducive to incremental operation of the timer, as described in [94]. Statistical slack, however, also has problems in serving as a good metric for optimization, because statistical slack is a distribution, and present-day optimization tools typically do not know how to deal with distributions. Of course, statistical slack is richer in information content than a distribution, since it is typically parameterized by the sources of variation (i.e., a first-order canonical form). this information opens up possibilities of using this parameterized model to *choose* the type of optimization that would best improve timing characteristics. But this also makes it even more complicated for the optimizer to use statistical slack as a metric.

One possibility is to use a σ-sampled value of the slack as a metric, e.g., the

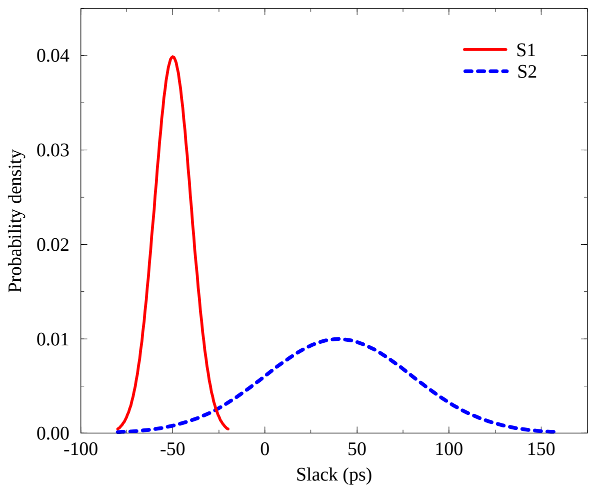


Figure 7.2: Statistical slack of two paths with the same -3σ value.

$\mu - 3\sigma$ value of the slack. This has the advantage of ensuring adequate parametric yield when timing is closed, and being a single number that represents the entire process space. Unfortunately, this metric has problems, too. Fig. 7.2 shows two slack distributions $S1$ and $S2$ which have the same -3σ value. In the case of $S1$, the distribution is pretty tight and improvement (i.e., movement to the right) is best accomplished by moving the entire distribution. Moving an entire distribution typically costs area and power since it is often achieved by inserting buffers or up-sizing gates. On the other hand, $S2$ can be improved by reducing its sensitivity to process, i.e., tightening its distribution. Unfortunately, σ-sampled slack does not give us this type of insight.

Now we will consider one last example to demonstrate the problems with statistical slack. Consider a situation in which two σ-sampled path slacks are -70 and -50 ps, respectively. Traditional wisdom says to improve the first path till its slack reaches -50 ps, and then try to improve both. Depending on the spread and correlation of the two distributions, this may not be so wise. If the spread of the slacks is large and the two slacks are uncorrelated, then the second path may be almost as much of a yield limiter as the first. If they are tightly correlated, however, then traditional wisdom is sound.

7.3 Criticality Probability

Some of the desired features of a good variation-aware diagnostic metric are: (1) preferably, a single number; (2) preferably, a number with a fixed range such as 0 to 1; (3) a number that implicitly represents criticality in the entire process space; (4) a number that is implicitly correlation-aware; and (5) a number that can be computed efficiently. Criticality probability is a metric that satisfies all these requirements.

7.3.1 Definitions and Properties

Definition 5 Criticality of a path *is the probability of manufacturing a chip in which this path is critical.*

Definition 6 Criticality of a set of paths *is the probability of manufacturing a chip in which at least one path from this set is critical.*

Definition 7 Criticality of an edge (node) *is the probability of manufacturing a chip in which this edge (node) is on the critical path.*

According to the definitions, the criticality of a timing edge (node) is also the criticality of the set of all paths going through that edge (node). The concept of criticality can be extended to a circuit gate, and even to any fragment of a timing graph or circuit. This flexibility is useful for circuit optimization. The concept of criticality is illustrated in Figure 7.3 on the example of a simple circuit with its two-dimensional process space. A path $P1$ is critical if the process parameters fall in the area $P1$ in the process space during manufacturing. The criticality of the path $P1$ is therefore the probability of manufacturing a chip with the combination of parameters falling in this area $P1$.

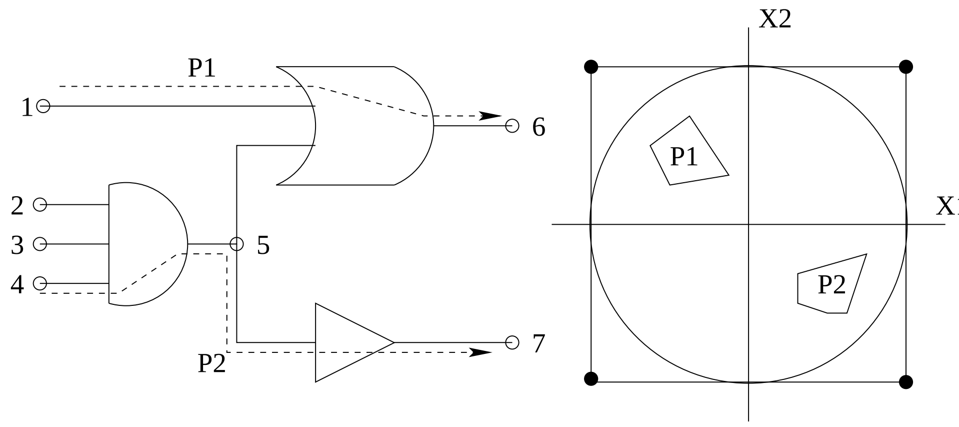


Figure 7.3: Critical paths and process subspaces.

The criticality of a set of paths S can be expressed by the following integral:

$$\int\cdots\int_{\max\limits_{P_i\in S}(D(P_i))>\max\limits_{P_j\notin S}(D(P_j))} p(X_1, X_2, \cdots)dX_1dX_2\cdots \tag{7.1}$$

where $p(X_1, X_2, \cdots)$ is the joint PDF of process parameters $X_1, X_2, \cdots$; P_i, P_j are circuit paths belonging and not belonging to the set S, respectively, and $D(P_i)$, $D(P_j)$ are path delays as functions of process parameters. Obviously, this formula is not practical for computing criticalities, as it requires multidimensional integration across complex polyhedrons. So, we need a better way of computing criticalities. It is useful to prove the following simple lemma.

Lemma 5 *If two canonical forms A and B differ in at least one coefficient, then the probability that they are equal is 0.*

Proof: The equality of two canonical forms is expressed as

$$a_0 + \sum_{i=1}^{n} a_i X_i + a_r X_{ra} = b_0 + \sum_{i=1}^{n} b_i X_i + b_r X_{rb}. \tag{7.2}$$

This equality can be rewritten as

$$(a_0 - b_0) + \sum_{i=1}^{n} (a_i - b_i) X_i + a_r X_{ra} - b_r X_{rb} = 0. \tag{7.3}$$

This equation defines a hyperplane in the process space because at least one of its coefficients is not zero. The probability that the canonical forms are equal is a volume integral of the joint PDF of the process parameters over this hyperplane. Obviously this probability is zero for any practical probability distribution because the thickness of any hyperplane is zero. □

Using this lemma we prove the following theorem:

Theorem 6 *The criticality of a set S of paths is the sum of the criticalities of these paths.*

Proof: If no paths have the same canonical form of their delays, the theorem is obvious because the probability that several paths are critical simultaneously is 0. Assume that n paths have the same canonical form D of delay and that the probability that this delay is larger than any other path delay in the circuit is p. Then, according to convention, each of these paths is assigned a criticality probability of p/n. Thus, the theorem holds in this case, too. □

From this theorem we derive the following corollaries.

Corollary 3 *Edge (node) criticality is equal to the sum of criticalities of the paths going though this edge (node).*

Corollary 4 *If all the circuit's paths are divided into two non-intersecting subsets A and B, then the criticality of subset A is the probability that the maximum* *delay of the paths belonging to the set A is larger than the maximum delay of the paths belonging to the set B.*

In order to compute edge criticality, we compute the canonical form A of the maximum delay of the paths going through this edge and the canonical form B of the maximum delay of the paths not going through this edge. Then the edge criticality is simply the probability that $A > B$, or, in other words, the tightness probability of A with respect to B.

7.3.2 Principles for Criticality Computation

In order to explain edge criticality computation, we give several definitions and recall some facts from network theory.

Definition 8 Edge slack *of an edge is the maximum delay of all paths going through the edge.*

Definition 9 Complement edge slack *of an edge is the maximum delay of all paths not going through the edge.*

The maximum of any edge's slack and complement slack is the longest path of the circuit, which is the negative of the slack of the circuit in late mode (edge slacks are defined in this manner to avoid minus signs in the following derivations).

From these definitions, it follows that the edge criticality is the probability that the edge slack is greater than the complement edge slack, i.e.,

$$p(e_{critical}) = p(edgeslack > complementedgeslack). \tag{7.4}$$

In other words, the tightness probability of the edge slack over the complement edge slack. Tightness probability can be computed by formula (2.25).

7.4 Efficient Edge Criticality Computation

7.4.1 Edge Slack Computation

The set of paths going through edge e forms a so-called edge flow graph G_e consisting of three parts: the edge input cone, the edge e itself, and the edge output cone. The slack s_e of edge e is simply delay $D(G_e)$ of its flow graph G_e. It can be expressed as:

$$s_e \equiv d(G_e) = D_{incone} + d(e) + D_{outcone}, \tag{7.5}$$

where D_{incone} is the delay of the edge input cone, $d(e)$ is the delay of the edge e, $D_{outcone}$ is the delay of the edge output cone. Recalling the definition of arrival and required arrival times, we express the slack of edge $e = (i, t)$ as follows:

$$s_e = AT(i) + d(e) - RAT(t) \tag{7.6}$$

where $AT(i)$ is the arrival time at the initial node i of the edge e and $RAT(t)$ is the required arrival time at terminal node t of the edge e. Figure 7.4 illustrates edge slack computation.

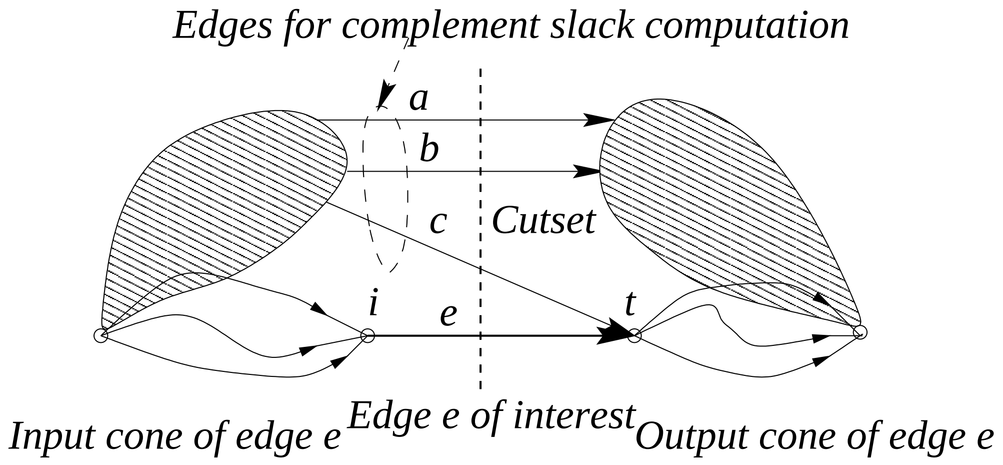


Figure 7.4: Edge slack computation.

7.4.2 Cutset Construction

Computation of complement edge slack is more complicated and requires additional considerations. In network theory [26], a cut between source and sink nodes is defined as a set of edges whose removal from the network disconnects the source and sink nodes. Here, we consider only those cuts satisfying the condition that each path from the source to the sink has only one common edge with each cuts. We call these cuts as *a minimal separating cutset*. The algorithm as shown in Figure 7.5 computes a minimal separating cutset Ω_i covering a given timing graph. For any edge of the timing graph, this algorithm computes at least one cut containing that edge.

```
Levelize the timing graph by topological sort
$\Omega_0 = \{Edges\, outgoing\, from\, source\, node\}$
For each level $i$
    $\Gamma = \{Edges\, incoming\, to\, level\, i\}$
    $\Lambda = \{Edges\, outgoing\, from\, level\, i\}$
    $\Omega_i = \Omega_{i-1} - \Gamma + \Lambda$
```

Figure 7.5: Algorithm for cutset computation.

Figure 7.6 illustrates cutset computation. We move a scan line along the levelized timing graph from the source to the sink. Each time we step over a level of the graph, we transform the current cut into the next one by excluding the edges coming into the nodes of the current level and including the edges going out from the nodes of the current level. Any cuts computed by the algorithm separates the nodes of the timing graph into two sets: N_i containing the source node and N_f containing the sink node. Any node from the set N_i belongs to the lower level of the timing graph than any node from the set N_f. From that, we can

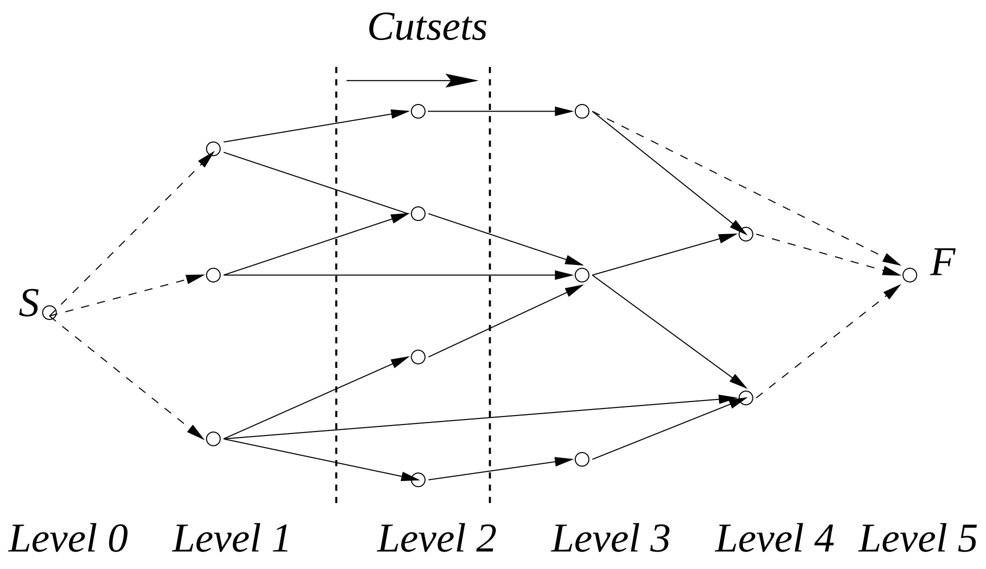


Figure 7.6: Computation of a set of cuts.

conclude that any path from the source to the sink intersects any cuts computed by this algorithm exactly at one edge.

In order to compute the complement slack for edge e, we consider a minimal separating cut C_e containing this edge. Let $C_{\overline{e}} = C_e - \{e\}$ be a set of edges in the cut except e. Then the set of paths going through the edges of the set $C_{\overline{e}}$ includes all the paths of the timing graph except the paths going through the edge e. The maximum delay of the paths going through the set of edges $C_{\overline{e}}$ is exactly the complement slack of the edge e. The complement slack can be computed as the statistical maximum of all the edge slacks of members of the set $C_{\overline{e}}$. The complement slack of edge e, shown in Figure 7.4, is computed as

$$s_{\overline{e}} = max(s_a, s_b, s_c) = max(s_a, max(s_b, s_c)). \tag{7.7}$$

Such a naive implementation of this approach, unfortunately, has a quadratic complexity, because each edge criticality requires calculation of the maximum of all other edge slacks.

7.4.3 Efficient Complement Edge Slack Computation

7.4.3.1 Cutset Based Complement Edge Slack Computation

The efficiency of complement slack computation can be improved if we use a tree data structure to re-use intermediate complement slack values. We construct a hierarchical partition of the cutset and compute all the complement slacks simultaneously, remembering and reusing slacks and complement slacks of the partition subsets. For simplicity, we assume that the cutset has $n = 2^t$ edges and construct a balanced binary partition tree shown in Figure 7.7. However, our approach can be applied to cuts with an arbitrary number of edges. The construction of the partition tree can be done either top down by sequentially splitting the sets of edges into equal parts or bottom up by merging pairs of edges and then pairs of the subsets of edges. Each leaf node in the partition tree

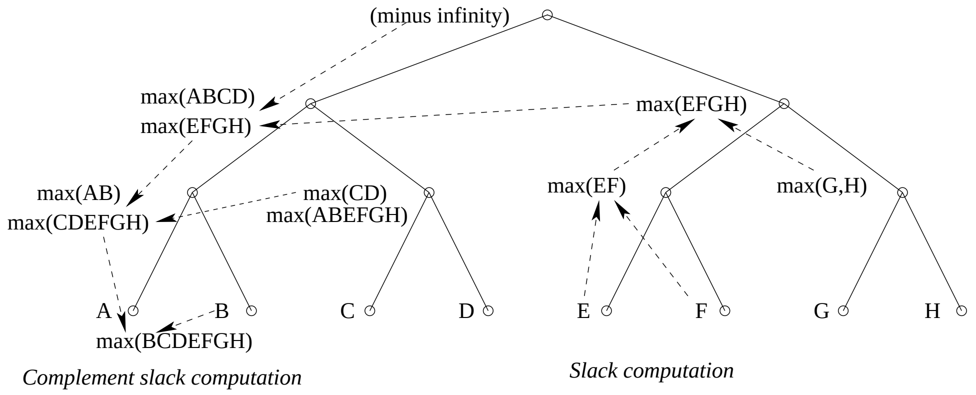


Figure 7.7: Binary partition tree.

represents one edge of the cuts. Each non-leaf node defines two sets of edges: the set of the node's children and the set of the edges that are not the node's children. With each node of the tree we associate a node slack and a node complement slack. The node slack is the maximum of slacks of its child edges. The node complement slack is the maximum of the slacks of non-child edges. For a leaf node, these two slacks are exactly the edge slack and the complement edge slack. The computation of node slacks and complement node slacks is illustrated in Figure 7.7. The algorithm as shown in Figure 7.8 computes the slacks associated with tree nodes and complement edge slacks.

```
Construct a partition tree of the cut edges;
Assign edge slacks to leaf nodes;
Traverse the tree bottom-up, and for each non-leaf node
  Slack = max(children's slacks);
Assign minus infinity slack to the root node
Traverse the tree top-down, and for each node
  Complement slack = max(parent complement slack, sibling node slack);
For each leaf node
  Edge criticality = P(edge slack > complement edge slack);
```

Figure 7.8: Algorithm for complement edge slack computation.

This algorithm computes criticalities of all edges in the cut simultaneously and has linear complexity as the number of max operations is proportional to the number of the tree nodes. A detailed analysis shows that it requires $4n - 6$ max operations where tightness probability computation is considered as a max operation, too.

7.4.4 Edge Criticality Computation

The overall algorithm to compute the edge criticality for all edges in a given timing graph G is shown in Figure 7.9.

Regarding the complexity of the algorithm for edge criticality computation, we have the following theorem

```
1 Levelize the timing graph $G$;
2 $\Omega = \emptyset$;
3 For $l$ =0 to $l_{max}$
4   Add to $\Omega$ edges whose source node level = $l$;
5   Remove from $\Omega$ edges whose sink node level = $l$;
6   $\Omega_{es}$ = compute-edge-slack($\Omega$);
7   $\Omega_{ecc}$ = compute-edge-criticality($\Omega_{es}$);
8 End
```

Figure 7.9: Algorithm for edge criticality computation.

Theorem 7 *The algorithm for computing edge criticality for all edges in the timing graph G as shown in Figure 7.9 has a complexity of $O(l_{max}N)$, where l_{max} is the maximum depth in G, N is the number of edges in G.*

Proof: Levelizing the timing graph G as shown in line 1 can be done in $O(N)$ by performing a breast first traverse of G. Forming the cut set at level l as shown in line 4 and 5 can be done in $O(m_l)$ with m_l equal to the number of edges being added and removed at level l. Considering the loop in line 3, the cut set construction for all levels equals to $O(m_0)$+$O(m_1)$+...+$O(m_{l_{max}})$, which is again $O(N)$ because each edge is added and removed only once, i.e., $m_0 + m_1 + ... + m_{l_{max}}$=$2N$.

According to section 7.4.1 and 7.4.3, both the computation of edge slacks and the computation of complement edge slacks for edges at level l can be done in $O(n_l)$, where n_l is the number edges in the cut set Ω of level l. Hence the computation of edge criticality for edges at level l can also be done in $O(n_l)$. Considering the loop in line 3, all edges' edge criticality can be computed in $O(n_0)$+$O(n_1)$+...+$O(n_{l_{max}})$, which is equivalent to $O(l_{max}N)$ because the sum of

$n_0 + n_1 + ... + n_{l_{max}}$ is upper bounded by $l_{max}N$. Therefore, the total complexity of the algorithm in Figure 7.9 is $O(l_{max}N)$. $\square$

7.4.5 Speedup Technique

In practical circuits, an edge may intersect many cuts as is shown in Figure 7.10. This often happens in sequential circuits with many flip-flops. In real circuits more than 50% of edges go through multiple cutsets. Occurrence of edges in multiple cuts significantly slows down criticality computation because the same edge is processed multiple times in different cuts. In order to improve efficiency,

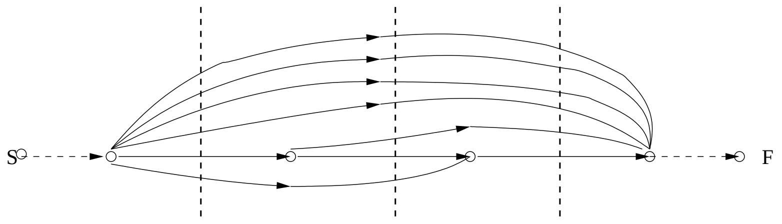


Figure 7.10: Edges going through many cuts.

we developed a technique to eliminate repeated processing of the same edge. This technique is based on the observation that after the criticality of an edge is computed once, the slack of this edge is used only for computing the complement slacks of the other edges by means of a max operation. The computation of the complement slacks does not require knowledge of the individual edge slacks and any group of edges can be represented by the maximum of their edge slacks. The improved algorithm for edge criticality computation is shown in Figure 7.11.

The idea behind the algorithm as shown in Figure 7.11 is that we only compute each edge's edge criticality once, and for those edges that fly across many levels as originally shown in Figure 7.10, we keep an array of flying edge slacks $\Psi[t]$ with t=$0, ..., l_{max}$. The purpose of $\Psi[t]$ is to keep the statistical maximum of the edge slacks for those edges that go through multiple cuts and whose sink level equals

```
1   Levelize the timing graph $G$;
2   $\Psi[0, ..., l_{max}]$ = -$\infty$;
3   For $l$ =0 to $l_{max}$
4     $\Omega = \emptyset$;
5     Add to $\Omega$ edges whose source level = $l$;
6     $\Omega_{es}$ = compute-edge-slack($\Omega$);
7     $\Omega_{ecc}$ = compute-edge-criticality($\Omega_{es} \bigcup \Psi[l+1, ..., l_{max}]$);
8     Denote edges in $\Omega$ whose sink level $> l$ as $\Theta$;
9     For each edge in $\Theta$
10      $t$ = edge's sink level;
11      $\Psi[t]$ = max($\Psi[t]$, edge's edge slack);
12    End
13  End
```

Figure 7.11: Improved algorithm to compute edge criticality.

to t (line 11). Therefore, at each level, we only add those newly appeared edges (line 4 and 5) and compute their edge slacks (line 6) once. Then we combine them with all other flying edges' edge slacks whose sink level is greater than the current level l, and feed it to the edge criticality computation algorithm (line 7). Among the returned edge criticality, only those criticality for the newly added edges are saved. Line 9, 10 and 11 are used to update flying edges' edge slacks for next iteration.

The complexity of this algorithm is shown in the following theorem

Theorem 8 *The algorithm for computing edge criticality for all edges in the timing graph G as shown in Figure 7.11 has a complexity of $O(N)+O(l_{max}^2)$, where l_{max} is the maximum depth in G, N is the number of the edges in G.*

Proof: The proof follows similar arguments as shown in the proof of Theorem 7 except that: (1) each edge appears in Ω only once, so that the edge slack and edge criticality for each edge is also computed once; (2) the flying edges' edge slacks are embedded within two loops (line 3 and 9), therefore, updating the flying edges' edge slack takes $O(l_{max}^2)$. Considering everything together, the total complexity of the algorithm as shown in Fig. 7.11 is $O(N)$+$O(l_{max}^2)$. $\square$

For a practical timing graph G, the maximum level number l_{max} is much smaller than N, therefore, the complexity of the above algorithm is in fact $O(N)$, which implies *constant time* for computing the criticality of each edge.

7.5 Conditional Criticality

The criticality introduced above is the probability of a path, edge or node being critical among all the manufactured chips. However, optimization for yield is interested in improving only those chips that violate timing constraints without wasting resources speeding up chips that are already sufficiently fast. In other words, if an edge or path is only critical in a subset of the process space and there are no failing chips in that region, then there is no point improving that edge or path. The concept of conditional criticality helps to solve this problem by providing information on how critical an edge is among failing chips only.

Definition 10 Conditional criticality of a path (edge, node) *is the conditional probability of manufacturing a chip in which this path (edge, node) is critical, conditional upon the chip violating its timing constraints.*

Conditional criticality of an edge e is expressed as follows:

$$p(e_{critical}|chip_{fails}) = \frac{p(e_{critical}, chip_{fails})}{p(chip_{fails})} \tag{7.8}$$

where $p(e_{critical}|chip_{fails})$ is the conditional probability that the edge e is critical conditional upon the timing constraints being violated; $p(e_{critical}, chip_{fails})$ is the probability that the edge e is critical and the timing constraints are violated; $p(chip_{fails})$ is the probability that the timing constraints are violated. In other words, conditional criticality is the probability of being critical computed for the sample space consisting of only those chips violating timing constraints.

In our case, the required time at the sink node is always 0. So the condition that the timing constraints are violated is $D_{ckt} > 0$ where D_{ckt} is the circuit delay. The probability $p(chip_{fails})$ that the timing constraints are violated is the tightness probability of D_{ckt} with respect to 0. The condition that the edge e is critical is $s_e > s_{\overline{e}}$ where s_e is the slack of the edge e and $s_{\overline{e}}$ is the complement slack of the edge e. This condition can be rewritten as $s_e - s_{\overline{e}} > 0$. Then the condition that both the edge e is critical and the timing constraints are violated is expressed as

$$min(s_e - s_{\overline{e}}, D_{ckt}) > 0. \tag{7.9}$$

So, the probability $p(e_{critical}, chip_{fails})$ that the edge e is critical and the timing constraints are violated is just tightness probability of $min(s_e - s_{\overline{e}}, D_{ckt})$ with respect to 0. Using edge slacks and complement edge slacks, this probability can be efficiently calculated by linear approximation of the statistical min operation. If the distribution of statistical min is highly skewed to the left, the accuracy of the Gaussian approximation can be low. In this case, it is better to represent the result of the min operation with a skewed normal distribution and compute the tightness probability numerically. Substituting the computed probabilities into (7.8), we compute the conditional criticality of the edge e conditional upon the timing constraints being violated. Thus conditional edge criticalities can be computed with the same efficiency as unconditional ones. Similarly, it is possible to compute conditional criticality only considering chips that meet their timing constraints. This kind of conditional criticality is useful for statistical optimization by down-sizing gates to reduce power consumption.

7.6 Experiment Results

We have implemented the proposed technique for computing both unconditional and conditional criticalities in the industrial statistical static timing analyzer EinsStat [94]. The criticalities are calculated for all timing edges for all clock phases, rising and falling transitions, early and late modes. We implemented three algorithms for criticality computation: the basic algorithm with straightforward calculation of the complement edge slacks, the partition tree based algorithm, and the algorithm with the speed up technique eliminating repeated processing of edges intersecting multiple cuts.

Table 7.1 shows the run times of the different algorithms and compares them with the run time of basic SSTA. Columns 1 and 2 show chip name and the number of gates. Columns 3, 4, 5, 6 report CPU time for SSTA and three different versions of the criticality computation algorithm. Additionally, columns 5 and 6 provide information on the absolute and relative overhead of criticality computation (of all edges in the graph) over and above statistical timing. From Table 7.1 we see that both the partition tree approach and the elimination of repeated processing of edges intersecting multiple cuts are important enhancements of the criticality computation technique. Only with these modifications is the criticality computation always faster than the base statistical timing, making it sufficiently fast for such applications as optimization, synthesis and test generation. From Table 7.1 we see that for the large design with 442,000 gates the proposed technique can compute criticalities in only 2.66 minutes of CPU time, while it takes

Table 7.1: Run time comparison

Chip	Size	SSTA	Basic techn.	Partition tree	Speedup techn.
D1	0.15k	1.97s	0.36s	0.20s/10.2%	0.19s/9.6%
D2	0.66k	7.28s	0.23s	0.04s/0.5%	0.04s/0.5%
D3	3.04k	3.37s	13.3m	4.44s/1.3x	0.39s/11.6%
D4	57.4k	21.5s	-	1.92m/5.4x	14.1s/65.6%
D5	87.2k	1.40m	-	52.5s/62.7%	27.5s/32.9%
D6	442k	24.9m	-	14.6m/58.6%	2.66m/10.7%

24.9 minutes to perform the basic statistical timing analysis.

For verifying the accuracy of our algorithms, we implemented a Monte Carlo technique for criticality computation. We generate 10,000 random samples of process parameters, compute gate delays corresponding to them and perform deterministic timing analysis to find a critical path corresponding to each sample. Counting the number of times that each edge is on the critical path for all Monte-Carlo samples, we compute the criticality probability of that edge.

Table 7.2 shows the accuracy of the criticality computation for different amounts of process variation. The amount of process variation is given as the average standard deviation of gate delay expressed as a percentage of nominal gate delay. The first line of the table reports the maximum difference between criticalities computed by the proposed algorithm and the Monte-Carlo technique. The second line reports the sum of the absolute values of the differences between the criticalities computed by the proposed algorithm and Monte-Carlo for all edges of the timing graph. We see that the proposed technique has high accuracy.

Additional investigation of the sources of computational error shows that the main part of the error is due to the linear approximation of the statistical min and max operations used in the parameterized SSTA. This error grows when the delay variation is larger, which we can see from this table.

Table 7.2: Accuracy of criticality computation

σ (%)	0.3%	0.6%	1.5%	3%	4.5%	5%
maxDiff	0.0004	0.0003	0.0012	0.0006	0.0296	0.0754
sumDiff	0.0012	0.0359	0.144	0.0585	0.711	1.104

We performed a number of experiments to compute conditional criticalities. The experiments show that conditional criticalities can be computed with approximately the same accuracy and CPU time as the unconditional ones. This conclusion fully agrees with our expectation since the computation of conditional criticalities is only a little extra work after the computation of unconditional criticalities. On the other hand, our experiments show that the values of the conditional criticality can be significantly different from the values of the unconditional ones. In fact, conditional criticality significantly depends on the length of the circuit clock cycle while the value of unconditional criticality is always the same. Fig. 7.12 demonstrates the dependence of the conditional criticalities on clock cycle length for four different timing edges. We see that when the clock cycle is longer, the conditional criticality of different edges may either increase or decrease depending on the circuit topology and process variation.

7.7 Conclusions and Discussion

In this chapter, we described an accurate and efficient method for computing criticalities of all timing edges in the context of parameterized block-based statistical timing analysis. Our algorithm computes criticalities of all edges of the timing graph with linear complexity with respect to the number of timing edges and can compute both unconditional and conditional criticalities. We implemented the proposed algorithm in an industrial SSTA tool targeted for both sign-off timing

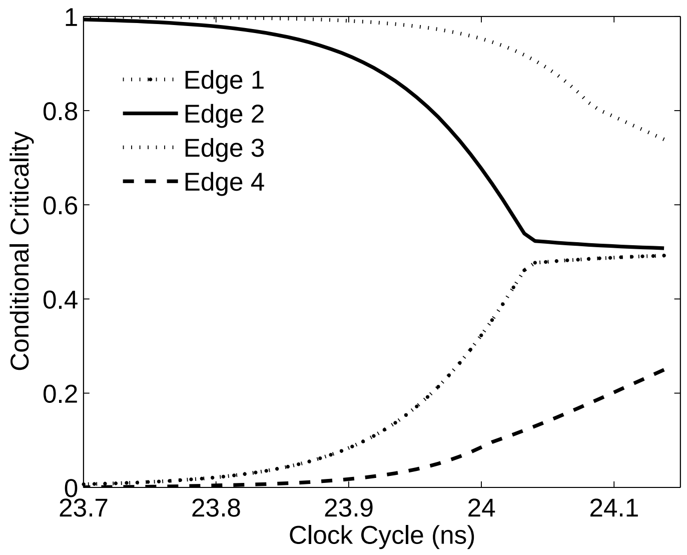


Figure 7.12: Conditional criticality as a function of clock cycle

analysis and for guiding circuit synthesis and optimization. Computational experiments with industrial circuit designs having up to 442K gates demonstrated high accuracy and low run time of the proposed technique. The maximum error of criticality computation is about 7.5% compared to Monte-Carlo simulations. The CPU time of criticality computation varies from 10% to 65% of the CPU time of statistical timing analysis even for large designs.

CHAPTER 8

Yield Gradients Computation

Statistical timing is an efficient way of taking into account process variations during performance analysis. However, optimizing a circuit across the entire process space is an extremely difficult challenge. Traditionally, static timing has been useful both for timing sign-off and to provide diagnostics for optimization. Traditional diagnostics such as the notion of a unique critical path or timing slack can no longer be used as metrics to guide optimization in the presence of variability. This chapter presents a novel and efficient method to compute the gradient of parametric yield with respect to the delay of each gate or wire. The resulting gradients can be rank-ordered for discrete optimization in a physical synthesis setting, or fed to a nonlinear optimizer for continuous optimization of design parameters such as transistor sizes, thus enabling formal mathematical yield optimization.

8.1 Introduction

Timing optimization is traditionally conducted by physical synthesis in a library-based flow or transistor sizing in a transistor-level custom design methodology. The optimization in either case is guided by deterministic timing slack or margins. In the presence of process variations, the goal is to close timing in the entire process space. Traditional metrics like deterministic slack fail to provide correct guidance to optimizers. Criticality or conditional criticality as we discussed in chapter 7 is a useful variation-aware metric for optimization. However, this probability still does not tell us how the parametric yield of the chip is impacted by changing the delay characteristics of an edge of the timing graph (by sizing or buffering, for example).

In this chapter, we propose a novel and efficient method of computing yield gradients as a post-processing step after statistical timing has been completed. We compute the gradient of *parametric timing yield* (including consideration of variance) with respect to each timing edge. The yield gradients (in conjunction with statistical slacks or timing margins) directly enable various types of optimization such as parametric yield maximization or power minimization subject to yield and timing requirements.

In our approach we use the same concept of graph cutsets as [20], but we do not use any perturbation of timing edge PDFs or convolution. Instead, we derive explicit analytical expressions for all components of yield gradients, which makes our approach efficient. Unlike [55] we do not use any chain-ruling or propagation of sensitivities through the timing graph, which in turn greatly simplifies our computations. Results of this chapter have been also reported in [116].

8.2 Differentiation of Statistical Maximum Operation

In this section, we derive formulas for differentiation of the statistical max operator with respect to its arguments, which will be used in the next section for yield gradient computation.

8.2.1 Preliminaries

Let A and B be two first-order canonical forms

$$\begin{aligned} A &= a_0 + \sum a_i X_i + a_r X_A, && (8.1) \\ B &= b_0 + \sum b_i X_i + b_r X_B, && (8.2) \end{aligned}$$

where X_i are the correlated unit-Gaussian random variations, X_A and X_B are uncorrelated unit-Gaussian random variations, a_i and b_i are sensitivities to correlated random variations, and a_r and b_r are the sensitivities to uncorrelated random variations, respectively. The variance of A and B and their covariance are

$$\begin{aligned} \sigma_A^2 &= \sum a_i^2 + a_r^2, && (8.3) \\ \sigma_B^2 &= \sum b_i^2 + b_r^2, && (8.4) \\ cov(A,B) &= \sum a_i b_i. && (8.5) \end{aligned}$$

Some frequently-used equations are listed below.

$$\begin{aligned} \phi(r) &\equiv \frac{1}{\sqrt{2\pi}} exp(-\frac{r^2}{2}) && (8.6) \\ \Phi(r) &\equiv \int_{-\infty}^{r} \phi(q)dq && (8.7) \\ \theta &\equiv (\sigma_A^2 + \sigma_B^2 - 2cov(A,B))^{1/2} && (8.8) \\ \frac{\partial\phi(r)}{\partial r} &= -r\frac{1}{\sqrt{2\pi}} exp(-\frac{r^2}{2}) = -r\phi(r) && (8.9) \\ \frac{\partial\Phi(r)}{\partial r} &= \phi(r). && (8.10) \end{aligned}$$

Let $Z = \max(A, B)$. The mean of Z is given by

$$\begin{aligned} z_0 &= \Phi(\frac{a_0 - b_0}{\theta})a_0 + \left[1 - \Phi(\frac{a_0 - b_0}{\theta})\right] b_0 + \theta\phi(\frac{a_0 - b_0}{\theta}) \\ &= \Phi a_0 + (1 - \Phi)b_0 + \theta\phi. && (8.11) \end{aligned}$$

For simplicity, we have used Φ and ϕ to represent $\Phi(\frac{a_0-b_0}{\theta})$ and $\phi(\frac{a_0-b_0}{\theta})$. This notation will be used wherever there is no ambiguity.

The variance of Z is

$$\sigma_Z^2 = (\sigma_A^2 + a_0^2)\Phi + (\sigma_B^2 + b_0^2)(1-\Phi) + (a_0+b_0)\theta\phi - z_0^2. \tag{8.12}$$

We are interested in the sensitivity of z_0 and σ_Z to the a_i, b_i, a_r and b_r parameters. Because of the symmetry between A and B, we will only focus on the derivation of sensitivity with respect to a_0, a_i and a_r.

8.2.2 Sensitivity of Maximum's Mean

8.2.2.1 With Respect to Mean

We first derive the sensitivity of z_0 with respect to a_0.

$$\frac{\partial z_0}{\partial a_0} = \frac{\partial \Phi}{\partial a_0}a_0 + \Phi - \frac{\partial \Phi}{\partial a_0}b_0 + \frac{\partial \theta}{\partial a_0}\phi + \theta\frac{\partial \phi}{\partial a_0}. \tag{8.13}$$

It is easy to show that the following equations hold.

$$\begin{aligned}\frac{\partial \theta}{\partial a_0} &= 0, &(8.14)\\ \frac{\partial \Phi}{\partial a_0} &= \phi(\frac{a_0-b_0}{\theta})\frac{1}{\theta} = \frac{\phi}{\theta}, &(8.15)\\ \frac{\partial \phi}{\partial a_0} &= -\frac{a_0-b_0}{\theta}\phi(\frac{a_0-b_0}{\theta})\frac{1}{\theta} = -\frac{a_0-b_0}{\theta^2}\phi. &(8.16)\end{aligned}$$

Therefore, we have

$$\frac{\partial z_0}{\partial a_0} = (a_0-b_0)\frac{\phi}{\theta} + \Phi - \frac{a_0-b_0}{\theta^2}\phi\theta = \Phi. \tag{8.17}$$

8.2.2.2 With respect to correlated sensitivity

Next, we derive the sensitivity of z_0 with respect to the correlated sensitivity term a_i.

$$\frac{\partial z_0}{\partial a_i} = \frac{\partial \Phi}{\partial a_i} a_0 - \frac{\partial \Phi}{\partial a_i} b_0 + \frac{\partial \theta}{\partial a_i}\phi + \theta \frac{\partial \phi}{\partial a_i}. \tag{8.18}$$

It is easy to show that the following equations hold.

$$\begin{aligned} \frac{\partial \Phi}{\partial a_i} &= \phi \frac{\partial(\frac{a_0-b_0}{\theta})}{\partial a_i} = -\phi \frac{a_0-b_0}{\theta^2}\frac{\partial \theta}{\partial a_i}, \textit{ and} && (8.19) \\ \frac{\partial \phi}{\partial a_i} &= -\frac{a_0-b_0}{\theta}\phi \frac{\partial(\frac{a_0-b_0}{\theta})}{\partial a_i} = \frac{(a_0-b_0)^2}{\theta^3}\phi \frac{\partial \theta}{\partial a_i}. && (8.20) \end{aligned}$$

Therefore, we have

$$\begin{aligned} \frac{\partial z_0}{\partial a_i} &= -\phi \frac{a_0-b_0}{\theta^2}\frac{\partial \theta}{\partial a_i}(a_0-b_0) + \frac{\partial \theta}{\partial a_i}\phi + \theta \frac{(a_0-b_0)^2}{\theta^3}\phi \frac{\partial \theta}{\partial a_i} \\ &= \left\{ -\frac{(a_0-b_0)^2}{\theta^2} + 1 + \theta \frac{(a_0-b_0)^2}{\theta^3} \right\} \phi \frac{\partial \theta}{\partial a_i} \\ &= \phi \frac{\partial \theta}{\partial a_i}. && (8.21) \end{aligned}$$

We need to compute $\frac{\partial \theta}{\partial a_i}$, which is given by

$$\begin{aligned} \frac{\partial \theta}{\partial a_i} &= \frac{\partial(\sigma_A^2 + \sigma_B^2 - 2cov(A,B))^{1/2}}{\partial a_i} \\ &= \frac{1}{2\theta}\frac{\partial(\sigma_A^2 + \sigma_B^2 - 2cov(A,B))}{\partial a_i} \\ &= \frac{1}{2\theta}\left\{ \frac{\partial \sigma_A^2}{\partial a_i} + \frac{\partial \sigma_B^2}{\partial a_i} - 2\frac{\partial cov(A,B)}{\partial a_i} \right\} \\ &= \frac{1}{2\theta}(2a_i + 0 - 2b_i) \\ &= (a_i - b_i)\frac{1}{\theta}. && (8.22) \end{aligned}$$

Therefore, in summary, we have

$$\frac{\partial z_0}{\partial a_i} = (a_i - b_i)\frac{\phi}{\theta} \tag{8.23}$$

8.2.2.3 With respect to uncorrelated sensitivity

We derive the sensitivity of z_0 with respect to the uncorrelated sensitivity term a_r.

$$\frac{\partial z_0}{\partial a_r} = \frac{\partial \Phi}{\partial a_r} a_0 - \frac{\partial \Phi}{\partial a_r} b_0 + \frac{\partial \theta}{\partial a_r} \phi + \theta \frac{\partial \phi}{\partial a_r}. \tag{8.24}$$

It is easy to show that the following equations hold.

$$\frac{\partial \Phi}{\partial a_r} = \phi \frac{\partial(\frac{a_0 - b_0}{\theta})}{\partial a_r} = -\phi \frac{a_0 - b_0}{\theta^2} \frac{\partial \theta}{\partial a_r}, \ and \tag{8.25}$$

$$\frac{\partial \phi}{\partial a_r} = -\frac{a_0 - b_0}{\theta} \phi \frac{\partial(\frac{a_0 - b_0}{\theta})}{\partial a_r} = \frac{(a_0 - b_0)^2}{\theta^3} \phi \frac{\partial \theta}{\partial a_r}. \tag{8.26}$$

Therefore, we have

$$\begin{aligned}
\frac{\partial z_0}{\partial a_r} &= -\phi \frac{a_0 - b_0}{\theta^2} \frac{\partial \theta}{\partial a_r}(a_0 - b_0) + \frac{\partial \theta}{\partial a_r} \phi + \theta \frac{(a_0 - b_0)^2}{\theta^3} \phi \frac{\partial \theta}{\partial a_r} \\
&= \left\{ -\frac{(a_0 - b_0)^2}{\theta^2} + 1 + \theta \frac{(a_0 - b_0)^2}{\theta^3} \right\} \phi \frac{\partial \theta}{\partial a_r} \\
&= \phi \frac{\partial \theta}{\partial a_r}.
\end{aligned} \tag{8.27}$$

We need to compute $\frac{\partial \theta}{\partial a_r}$, which is given by

$$\begin{aligned}
\frac{\partial \theta}{\partial a_r} &= \frac{\partial(\sigma_A^2 + \sigma_B^2 - 2cov(A, B))^{1/2}}{\partial a_r} \\
&= \frac{1}{2\theta} \frac{\partial(\sigma_A^2 + \sigma_B^2 - 2cov(A, B))}{\partial a_r} \\
&= \frac{1}{2\theta} \left\{ \frac{\partial \sigma_A^2}{\partial a_r} + \frac{\partial \sigma_B^2}{\partial a_r} - 2 \frac{\partial cov(A, B)}{\partial a_r} \right\} \\
&= \frac{1}{2\theta}(2a_r + 0 - 0) \\
&= a_r \frac{1}{\theta}.
\end{aligned} \tag{8.28}$$

Therefore, in summary, we have

$$\frac{\partial z_0}{\partial a_r} = a_r \frac{\phi}{\theta}. \tag{8.29}$$

8.2.3 Sensitivity of Maximum's Sigma

8.2.3.1 With Respect to Mean

We derive the sensitivity of σ_Z with respect to a_0.

$$\begin{aligned}
\frac{\partial \sigma_Z^2}{\partial a_0} &= 2\sigma_Z \frac{\partial \sigma_Z}{\partial a_0} \\
&= \left(\frac{\partial \sigma_A^2}{\partial a_0} + \frac{\partial a_0^2}{\partial a_0}\right)\Phi + (\sigma_A^2 + a_0^2)\frac{\partial \Phi}{\partial a_0} - (\sigma_B^2 + b_0^2)\frac{\partial \Phi}{\partial a_0} \\
&\quad + \theta\phi + (a_0 + b_0)\frac{\partial \theta}{\partial a_0}\phi + (a_0 + b_0)\theta\frac{\partial \phi}{\partial a_0} - 2z_0\frac{\partial z_0}{\partial a_0} \\
&= 2a_0\Phi + (\sigma_A^2 + a_0^2)\frac{\phi}{\theta} - (\sigma_B^2 + b_0^2)\frac{\phi}{\theta} + \theta\phi \\
&\quad - (a_0 + b_0)\theta(a_0 - b_0)\frac{\phi}{\theta^2} - 2z_0\Phi \\
&= 2(a_0 - z_0)\Phi + (\sigma_A^2 + a_0^2 - \sigma_B^2 - b_0^2)\frac{\phi}{\theta} - (a_0^2 - b_0^2)\frac{\phi}{\theta} + \theta\phi \\
&= 2(a_0 - z_0)\Phi + (\sigma_A^2 - \sigma_B^2)\frac{\phi}{\theta} + \theta\phi. \qquad (8.30)
\end{aligned}$$

Therefore, in summary, we have

$$\frac{\partial \sigma_Z}{\partial a_0} = \frac{1}{2\sigma_Z}\left\{2(a_0 - z_0)\Phi + (\sigma_A^2 - \sigma_B^2)\frac{\phi}{\theta} + \theta\phi\right\}. \qquad (8.31)$$

8.2.3.2 With Respect to Correlated Sensitivity

We derive the sensitivity of σ_Z with respect to the correlated sensitivity term a_i. We have

$$\begin{aligned}
\frac{\partial \sigma_Z^2}{\partial a_i} &= 2\sigma_Z \frac{\partial \sigma_Z}{\partial a_i} \\
&= \left(\frac{\partial \sigma_A^2}{\partial a_i} + \frac{\partial a_0^2}{\partial a_i}\right)\Phi + (\sigma_A^2 + a_0^2)\frac{\partial \Phi}{\partial a_i} + \left(\frac{\partial \sigma_B^2}{\partial a_i} + \frac{\partial b_0^2}{\partial a_i}\right)(1-\Phi) \\
&\quad -(\sigma_B^2 + b_0^2)\frac{\partial \Phi}{\partial a_i} + \left(\frac{\partial a_0}{\partial a_i} + \frac{\partial b_0}{\partial a_i}\right)\theta\phi + (a_0+b_0)\frac{\partial \theta}{\partial a_i}\phi \\
&\quad +(a_0+b_0)\theta\frac{\partial \phi}{\partial a_i} - 2z_0\frac{\partial z_0}{\partial a_i} \\
&= 2a_i\Phi - (\sigma_A^2 + a_0^2 - \sigma_B^2 - b_0^2)\phi\frac{a_0-b_0}{\theta^2}\frac{\partial \theta}{\partial a_i} \\
&\quad +(a_0+b_0)\frac{\partial \theta}{\partial a_i}\phi + (a_0+b_0)\theta\frac{(a_0-b_0)^2}{\theta^3}\phi\frac{\partial \theta}{\partial a_i} - 2z_0\frac{\partial z_0}{\partial a_i} \\
&= 2a_i\Phi - 2z_0\frac{\partial z_0}{\partial a_i} + \left\{(-\sigma_A^2 - a_0^2 + \sigma_B^2 + b_0^2)\frac{a_0-b_0}{\theta^2}\right. \\
&\quad \left. +a_0 + b_0 + \frac{(a_0^2-b_0^2)(a_0-b_0)}{\theta^2}\right\}\phi\frac{\partial \theta}{\partial a_i} \\
&= 2a_i\Phi - 2z_0\frac{\partial z_0}{\partial a_i} + \left\{(-\sigma_A^2 - a_0^2 + \sigma_B^2 + b_0^2 + a_0^2 - b_0^2)\frac{a_0-b_0}{\theta^2}\right. \\
&\quad \left. +a_0 + b_0\right\}\phi\frac{\partial \theta}{\partial a_i} \\
&= 2a_i\Phi - 2z_0\frac{\partial z_0}{\partial a_i} + \left\{(\sigma_B^2 - \sigma_A^2)\frac{a_0-b_0}{\theta^2} + a_0 + b_0\right\}\phi\frac{\partial \theta}{\partial a_i}. \qquad (8.32)
\end{aligned}$$

Therefore, the sensitivity can be computed as

$$\frac{\partial \sigma_Z}{\partial a_i} = \frac{1}{\sigma_Z}\left[a_i\Phi - z_0\frac{\partial z_0}{\partial a_i} + \left\{(\sigma_B^2 - \sigma_A^2)\frac{a_0-b_0}{\theta^2} + a_0 + b_0\right\}\frac{\phi}{2}\frac{\partial \theta}{\partial a_i}\right]$$

$$\begin{aligned}
\frac{\partial \sigma_Z}{\partial a_i} &= \frac{1}{\sigma_Z}\left[a_i\Phi - z_0(a_i-b_i)\frac{\phi}{\theta} + (a_i-b_i)\right. \qquad (8.33) \\
&\quad \left.\left\{a_0 + b_0 + (a_0-b_0)\frac{\sigma_B^2 - \sigma_A^2}{\theta^2}\right\}\frac{\phi}{2\theta}\right].
\end{aligned}$$

8.2.3.3 With Respect to Uncorrelated Sensitivity

We derive the sensitivity of σ_Z with respect to the uncorrelated term a_r. We have

$$\begin{aligned}
\frac{\partial \sigma_Z^2}{\partial a_r} &= 2\sigma_Z \frac{\partial \sigma_Z}{\partial a_r} \\
&= \left(\frac{\partial \sigma_A^2}{\partial a_r} + \frac{\partial a_0^2}{\partial a_r}\right)\Phi + (\sigma_A^2 + a_0^2)\frac{\partial \Phi}{\partial a_r} + \left(\frac{\partial \sigma_B^2}{\partial a_r} + \frac{\partial b_0^2}{\partial a_r}\right)(1-\Phi) \\
&\quad -(\sigma_B^2 + b_0^2)\frac{\partial \Phi}{\partial a_r} + \left(\frac{\partial a_0}{\partial a_r} + \frac{\partial b_0}{\partial a_r}\right)\theta\phi + (a_0+b_0)\frac{\partial \theta}{\partial a_r}\phi \\
&\quad +(a_0+b_0)\theta\frac{\partial \phi}{\partial a_r} - 2z_0\frac{\partial z_0}{\partial a_r} \\
&= 2a_r\Phi - (\sigma_A^2 + a_0^2 - \sigma_B^2 - b_0^2)\phi\frac{a_0-b_0}{\theta^2}\frac{\partial \theta}{\partial a_r} \\
&\quad +(a_0+b_0)\frac{\partial \theta}{\partial a_r}\phi + (a_0+b_0)\theta\frac{(a_0-b_0)^2}{\theta^3}\phi\frac{\partial \theta}{\partial a_r} - 2z_0\frac{\partial z_0}{\partial a_r} \\
&= 2a_r\Phi - 2z_0\frac{\partial z_0}{\partial a_r} + \left\{(-\sigma_A^2 - a_0^2 + \sigma_B^2 + b_0^2)\frac{a_0-b_0}{\theta^2}\right. \\
&\quad \left. +a_0 + b_0 + \frac{(a_0^2-b_0^2)(a_0-b_0)}{\theta^2}\right\}\phi\frac{\partial \theta}{\partial a_r} \\
&= 2a_r\Phi - 2z_0\frac{\partial z_0}{\partial a_r} + \left\{(-\sigma_A^2 - a_0^2 + \sigma_B^2 + b_0^2 + a_0^2 - b_0^2)\frac{a_0-b_0}{\theta^2}\right. \\
&\quad \left. +a_0 + b_0\right\}\phi\frac{\partial \theta}{\partial a_r} \\
&= 2a_r\Phi - 2z_0\frac{\partial z_0}{\partial a_r} + \left\{(\sigma_B^2 - \sigma_A^2)\frac{a_0-b_0}{\theta^2} + a_0 + b_0\right\}\phi\frac{\partial \theta}{\partial a_r}. \qquad (8.34)
\end{aligned}$$

Finally,

$$\frac{\partial \sigma_Z}{\partial a_r} = \frac{1}{\sigma_Z}\left[a_r\Phi - z_0 a_r\frac{\phi}{\theta} + a_r\left\{a_0 + b_0 + (a_0-b_0)\frac{\sigma_B^2-\sigma_A^2}{\theta^2}\right\}\frac{\phi}{2\theta}\right]. \qquad (8.35)$$

8.3 Yield Gradient Computation

Using the formulas derived above, in this section, we show how yield gradients can be computed in two different scenarios.

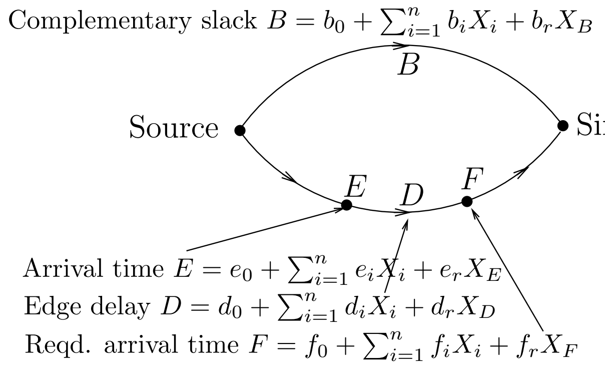

Complementary slack $B = b_0 + \sum_{i=1}^{n} b_i X_i + b_r X_B$
B
Source
Sink
E
D
F
Arrival time $E = e_0 + \sum_{i=1}^{n} e_i X_i + e_r X_E$
Edge delay $D = d_0 + \sum_{i=1}^{n} d_i X_i + d_r X_D$
Reqd. arrival time $F = f_0 + \sum_{i=1}^{n} f_i X_i + f_r X_F$

Figure 8.1: Derivation of yield gradients.

8.3.1 Principle of Yield Gradient Computation

Our goal is to find the gradient of the parametric yield of the chip to all components of the delay of a single edge of the timing graph. This problem can be divided into two parts. Once we know the statistical slack of the chip as a distribution, the parametric yield is easily expressed in terms of the CDF of the statistical slack. So if we could isolate the dependence of the statistical slack on the delay of a single edge, we could differentiate that relation to obtain the necessary yield gradients.

Let us focus on a single edge D. Fig. 8.1 gives us a convenient abstraction for the derivation. In the figure, B represents the complement slack of edge D and comprises the ensemble of all paths of the graph that do not pass through D. The arrival time E at the from-node of edge D represents the delay of the ensemble of paths from the source node to edge D. Similarly, the required arrival time F of the to-node of edge D represents the negative of the delay of the ensemble of paths from edge D to the sink node. The path Source-E-D-F-Sink is the ensemble of all paths of the timing graph that pass through D. Thus all paths of the graph have conceptually been split into two buckets – those that pass through D and those that do not. We have isolated the impact of the edge D on the statistical slack of the chip.

Next, we can write the statistical slack of the chip as the length of the longest path in Fig. 8.1

$$Chip\ slack \equiv Z = \max(B, E + D - F) \tag{8.36}$$

where each of B, E, D and F is expressed in first-order canonical form as shown in the figure. The globally correlated process variables are represented by random variables $X_i, i = 1, 2, \cdots, n$ and independently random variation by X_B, X_E, X_D and X_F. It is very important to note that B does not depend on the delay of the edge of interest since it represents the ensemble of paths that *do not pass* through D. Similarly, E and F do not depend on the delay of edge D since they represent the delay of the longest path in the fanin cone and fanout cone of D, respectively. Therefore, we can differentiate (8.36) to obtain the gradient of the chip slack Z to the components of D, i.e., $\frac{\partial z_0}{\partial d_0}$, $\frac{\partial z_0}{\partial d_i}$, $\frac{\partial z_0}{\partial d_r}$, $\frac{\partial \sigma_z}{\partial d_0}$, $\frac{\partial \sigma_z}{\partial d_i}$, and $\frac{\partial \sigma_z}{\partial d_r}$. All these computations can be obtained by using the formulas we have derived in previous section.

8.3.2 Performance Maximization for Yield

This situation is ASIC-like in that we assume a required yield has been specified, and we seek to compute the maximum performance (clock frequency, or, equivalently, slack) at which the chip can safely be operated. Assume that the required parametric yield is p, a constant. The longest delay of the timing graph in Fig. 8.1 is a statistical quantity Z which we assume has a mean value z_0 and a standard deviation σ_Z. Then the corresponding parametric longest path delay that meets the yield requirement p is denoted by $z(p)$. The smaller the value of Z, the better. The parametric delay which meets the requirement can be expressed as

$$z(p) = \Phi^{-1}(p)\sigma_Z + z_0, \tag{8.37}$$

where $\Phi^{-1}()$ is the inverse CDF of a standard normal. This equation simply says that the required yield can be translated into a required number of sigmas on the longest path delay distribution. Our goal now is to obtain the gradient of $z(p)$ with respect to d_0, $d_i, i = 1, 2, \cdots, n$ and d_r. This is easily accomplished by simply differentiating (8.37) with respect to each of these quantities.

$$\frac{\partial z(p)}{\partial d_i} = \Phi^{-1}(p)\frac{\partial \sigma_Z}{\partial d_i} + \frac{\partial z_0}{\partial d_i}, i = 0, 1, \cdots, n, r. \tag{8.38}$$

The two partial derivatives on the right hand side are readily obtained from the boxed equations in the Appendix (8.17), (8.23), (8.29), (8.31), (8.33) and (8.35).

8.3.3 Yield Maximization for Performance

This situation is microprocessor-like in that we assume a target frequency has been specified and we are trying to maximize the yield of chips that will achieve this frequency. Assume that the target frequency translates into a required longest path delay of T or less. In this case, our parametric yield p is simply the probability that the longest path Z has a value that is T or less. The yield is therefore $p(T) = \Phi\left\{(T - z_0)/\sigma_Z\right\}$. This simple equation can be differentiated with the help of the formulas in the appendix to obtain the necessary yield gradients

$$\begin{aligned}\frac{\partial p(T)}{\partial d_i} &= \frac{1}{\sigma_Z^2}\phi\left(\frac{T - z_0}{\sigma_Z}\right)\left\{-\sigma_Z\frac{\partial z_0}{\partial d_i} + (T - z_0)\frac{\partial \sigma_Z}{\partial d_i}\right\}, \\ & \quad i = 0, 1, \cdots, n, r.\end{aligned} \tag{8.39}$$

Although the formulas are lengthy, they are easily coded for efficient evaluation on a computer. In terms of implementation, when criticality probability is computed for an edge, a statistical maximum operation is performed between its edge slack and complementary edge slack. At that point, the above formulas allow us to compute yield gradients with respect to individual edge delays.

8.4 Applications of Yield Gradients

One immediate application of yield gradients is in a formal transistor-sizing optimization context such as EinsTuner [24]. A possible formulation of the optimization problem over the space of transistor widths W is as follows.

$$\begin{array}{ll} \max_W & \textit{parametric yield } p(T) \\ s.t. & \textit{performance requirement } T \\ s.t. & \textit{area, slew and other constraints.} \end{array} \tag{8.40}$$

In this case, we need to provide to the nonlinear optimizer at each iteration the yield at the required performance T, and the gradient of that yield with respect to all transistor sizes. This formulation fits well within a transistor-level tuning context since the gradient of the yield with respect to an edge delay can be chain-ruled with the gradient of the edge delay with respect to the individual transistor widths. The latter computation, by the adjoint method, is already a part of the transistor sizing formulation used presently. Of course, changing a transistor size will typically change the delay of multiple edges of the timing graph as well as fanin edges due to loading effects and fanout edges due to slew effects. All of these sensitivities can be chain-ruled and included in a straightforward manner.

The objective function could easily be changed from parametric yield to *profit* (or any other integral measure of the distribution of the chip slack) in the case of a sorted and speed-binned chip. Sorting implies that different ranges of speeds bring in different amounts of profit. The yield of each bin can be expressed in terms of z_0 and σ_Z as above and then the weighted sum of the profits over all bins differentiated to obtain the necessary gradients.

A second possible formulation is shown below.

$$\begin{array}{ll} \min_W & \textit{delay } z(p) \\ s.t. & \textit{a required yield } p \\ s.t. & \textit{area, slew and other constraints.} \end{array} \tag{8.41}$$

In this formulation, a required yield is specified, and we need to provide to the nonlinear optimizer at each iteration the fastest performance that can be achieved and the gradient of that performance with respect to all transistor widths.

Even in the context of discrete optimization, yield gradients can be used to rank order portions of the circuit that most need attention in order to increase the yield of the circuit to the necessary value, or to increase the performance at a required yield. Having quantitative gradients will decrease the guess-work and try-evaluate-undo loop that is common in physical synthesis during late timing correction and early-mode padding.

8.5 Conclusion and Discussion

Timing closure in the presence of variability implies closure across the entire relevant process space, which is an extremely challenging task. Traditional diagnostics and metrics like critical paths and deterministic timing slacks are inadequate for the optimization task. This chapter describes a novel method for computing parametric yield gradients accurately and efficiently, and proposes various applications that can take advantage of yield gradients. Although the actual optimization applications described in this section have not been implemented, we feel that yield gradients will enable powerful new ways of optimizing circuits in the presence of variability.

CHAPTER 9

Appendix

There are a few other research projects that have been done over the course of this dissertation. They fall into the following three categories: (1) routing optimization for signal and power integrity; (2) modeling and physical design for system-in-package; and (3) sleep transistor design for low power. As these researches are different from the main theme of this dissertation, they are not included in the main body of this dissertation, For completeness, however, this chapter summarizes the main points of these work with our contributions. Details of these work can be found in publications.

9.1 Routing Optimization for Signal and Power Integrity

As we are progressing into 90nm and beyond technologies, the performance of system-on-chip (SoC) is found to be mainly constrained by on-chip interconnects. Signal integrity and power integrity related to signal, clock and power/ground (P/G) network designs have emerged as two of the primary barriers for the the further advancement of gigascale SoC integration.

We study the full chip interconnect synthesis problems with RLC crosstalk constraints for gigascale system-on-chip design. We consider simultaneous shield insertion and net ordering (SINO), and solve the following two problems. (1) Given a global routing solution, decide an RLC crosstalk budgeting solution such that the final routing solution satisfies RLC crosstalk constraints and the routing area is minimized. (2) Given a netlist, synthesize a global routing solution with consideration of simultaneous shield insertion and net ordering such that the RLC crosstalk constraints are satisfied and the overhead due to shield insertion is minimized. We formulate and solve the noise budgeting problem as a linear programming problem, and propose an iterative-deletion based global routing synthesis technique with shield reservation and minimization. After global routing and noise budgeting, we perform simultaneous shielding insertion and net ordering to meet the partitioned RLC crosstalk bounds in each region, and carry out a local refinement procedure to further reduce the routing congestion and number of shields. Experiments using large industrial benchmark circuits show that compared to the best alternative routing algorithms we have studied, the above multi-phase algorithm framework uses less runtime and reduces the number of overflow and total routing area by up to 18.4% and 6.4%, respectively. Detailed research contribution and results have been reported in [99, 101, 97, 111, 112, 115, 45].

We also study the co-design problem for both signal network and P/G network under signal integrity and power integrity constraints. We present a novel design methodology that simultaneously considers global signal routing and power network design under integrity constraints. The main contributions are (1) a simple yet accurate power net estimation formula that decides the minimum number of power nets needed to satisfy both power and signal integrity constraints prior to detailed layout; (2) a probabilistic congestion model considering shielding for both signal integrity and power integrity; and (3) a novel multilevel routing framework to support the co-design of signal and power networks with explicit consideration of both signal and power integrity constraints. The proposed design methodology is a one-pass solution to the co-design of power and signal networks in the sense that no iteration between them is required in order to meet design closure. Experiment results using large industrial benchmarks show that compared to the state-of-the-art alternative design approach in literature, the proposed method can reduce the power network area by 19.4% on average under the same signal and power integrity constraints with better routing quality, but use less runtime. Detailed research results can be found in the following publications [104, 98, 100, 57, 102].

9.2 Modeling and Physical Design for System-in-Package

To address the severe design closure problem that faces today's high performance system-in-package or system-in-board designs, we formulate, for the first time, a constraint-driven I/O planning and placement problem, and formally introduced a set of design constraints into the academia community [106]. Our work in this area includes the modeling of signal and power integrity constraints for chip-package-board co-design, extraction of interconnect parasitics for system level connectivity, and efficient algorithms for I/O planning and placement. Experiment results using real industry designs show that by using our methodology, design productivity can be greatly improved. For example, for an industrial ASIC design, our method can effectively find a feasible solution and satisfy all given design constraints in less than 10 minutes, while the existing approach simply cannot close the design in one day, yet still leaving more than 30% design constraint violations. Research results of this work has been integrated into an industry tool, RioMagic [80], which is the first commercially available package-aware chip design and optimization tool for today's chip designers.

9.3 Sleep Transistor Design for Low Power

Considering P/G network's integrity as given by voltage drop constraints, we study the following two problems for low power designs [61]: (1) sleep transistor insertion problem and (2) sleep transistor insertion and co-sizing of P/G network and sleep transistor problem. Our contributions in studying these two problems include: (1) we prove that there exist multiple sleep transistor insertion solutions that will all lead to the same minimum area solution for both sleep transistor and P/G network; (2) we develop an optimal algorithm for one-port power network; and (3) the above optimal algorithm is further extended to the design of a multi-port power network, whose optimality has been experimentally validated. Compared with the best known approach, our algorithm reduces sleep transistor area by up to 44.1% for the first problem, and reduces the weighted power network and sleep transistor area by up to 61.3% for the second problem.

References

[1] A. Agarwal, D. Blaauw, and V. Zolotov. Statistical timing analysis for intra-die process variations with spatial correlations. In *Proc. Int. Conf. on Computer Aided Design*, pages 900 – 907, Nov. 2003.

[2] A. Agarwal, K. Chopra, and D. Blaauw. Statistical timing based optimization using gate sizing. In *DATE*, Mar 2005.

[3] A. Agarwal, V. Zolotov, and D.T. Blaauw. Statistical timing analysis using bounds and selective enumeration. In *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, volume 22, pages 1243 – 1260, Sept. 2003.

[4] Aseem Agarwal, Kaviraj S. Chopra, David Blaauw, and Vladimir Zolotov. Circuit optimization using statistical static timing analysis. In *Proc. Design Automation Conf*, pages 321–324, June 2005.

[5] Margaret Armstrong1 and Romain Jabin. Variogram models must be positive-definite. *Mathematical Geology*, 13:455–459, Oct. 1981.

[6] H. Bakoglu. *Circuits, Interconnects and Packaging for VLSI*. Addison-Wesley, 1990.

[7] Robert Berry. Correlation of diffusion process variations with variations in electrical parameters of bipolar transistors. *Proceedings of the IEEE*, pages 1513–1517, Sept. 1969.

[8] D. Boning, B. Lee, C. Oji, D. Ouma, T. Park, T. Smith, and T. Tugbawa. Pattern dependent modeling for cmp optimization and control. In *Chemical Mechanical Polishing Symposium, MRS Spring Meeting*, Apr 1999.

[9] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De. Parameter variations and impacts on circuits and microarchitecture. In *DAC 03*, Jun 2003.

[10] K. A. Bowman, S. G. Duvall, and J. D. Meindl. Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration. *Solid-State Circuits, IEEE Journal of*, pages 183–190, Feb. 2002.

[11] Stephen Boyd and Lin Xiao. Least-squares covariance matrix adjustment. *SIAM Journal on Matrix Analysis and Applications*, 27:532–546, 2005.

[12] Rafael L. Bras and Ignacio Rodriguez-Iturbe. *Random Funtions and Hydrology*. Dover Publishers, 1985.

[13] Michael Cain. The moment-generating function of the minimum of bivariate normal random variables. In *The American Statistician*, volume 48, May 1994.

[14] Y. Cao, P. Gupta, A. Kahng, D. Sylvester, and J. Yang. Design sensitivities to variability: Extrapolations and assessments in nanometer VLSI. In *ASIC/SOC Conference*, Sept 2002.

[15] H. Chang, V. Zolotov, C. Visweswariah, and S. Narayan. Parameterized block-based statistical timing analysis with non-Gaussian and nonlinear parameters. In *Proc. Design Automation Conf*, pages 71–76, June 2005. Anaheim, CA.

[16] Hongliang Chang and Sachin S. Sapatnekar. Statistical timing analysis considering spatial correlations using a single PERT-like traversal. In *Proc. Int. Conf. on Computer Aided Design*, pages 621 – 625, Nov. 2003.

[17] L. Chen, L. Milor, C. Ouyang, W. Maly, and Y. Peng. Analysis of the impact of proximity correction algorithms on circuit performance. *IEEE Transactions on Semiconductor Manufacturing*, 12:313–322, 1999.

[18] Y. Chen, P. Gupta, and A. B. Kahng. Performance-impact limited area fill synthesis. In *DAC*, Jun 2003.

[19] S. Choi, B. Paul, and K. Roy. Novel sizing algorithm for yield improvement under process variation in nanometer technology. In *Proc. Design Automation Conf*, Jun 2004.

[20] K. Chopra, S. Shah, A. Srivastava, D. Blaauw, and D. Sylvester. Parametric yield maximization using gate sizing based on efficient statistical power and delay gradient computation. *Proc. Int. Conf. on Computer Aided Design*, pages 1023–1028, November 2005. San Jose, CA.

[21] C.E. Clark. The greatest of a finite set of random variables. In *Operations Research*, pages 85 – 91, 1961.

[22] T.F. Coleman and Y. Li. An interior, trust region approach for nonlinear minimization subject to bounds. In *SIAM Journal on Optimization*, volume 6, pages 418–445, 1996.

[23] J. Cong, L. He, C. Koh, and Z. Pan. Interconnect sizing and spacing with considering of coupling capacitance. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 20(9):1164–1169, 2001.

[24] A. R. Conn, I. M. Elfadel, W. W. Molzen, Jr., P. R. O'Brien, P. N. Strenski, C. Visweswariah, and C. B. Whan. Gradient-based optimization of custom circuits using a static-timing formulation. *Proc. 1999 Design Automation Conference*, pages 452–459, June 1999. New Orleans, LA.

[25] Liang Deng and Martin D.F. Wong. Buffer insertion under process variations for delay minimization. In *Proc. Int. Conf. on Computer Aided Design*, pages 317–321, 2005.

[26] N. Deo. *Graph theory with applications to engineering and computer science*. Prentice-Hall, 1974.

[27] A. Devgan and C. Kashyap. Block-based static timing analysis with uncertainty. In *ICCAD 03*, Nov 2003.

[28] Bajis Dodin and S.E. Elmaghraby. Approximating the criticality indices of the activities in PERT networks. *Management Science*, 13:207–223, 1985.

[29] Ji-Seong Doh, Dae-Wook Kim, Sang-Hoon Lee, Jong-Bae Lee, Young kwan Park, Moon-Hyun Yoo, and Jeong-Taek Kong. A unified statistical model for inter-die and intra-die process variation. In *Simulation of Semiconductor Processes and Devices, SISPAD, International Conference on*, pages 131–134, Sept. 2005.

[30] S. G. Duvall. Statistical circuit modeling and optimization. In *5th Intl. Workshop Statistical Metrology*, pages 56–63, June 2000.

[31] M. Eisele, J. Berthold, D. Schmitt-Landsiedel, and R. Mahnkopf. The impact of intra-die device parameter variations on path delays and on the design for yield of low voltage digital circuits. In *Proc. Int. Symp. on Low Power Electronics and Design*, pages 237–242, Aug. 1996.

[32] Paul Friedberg, Yu Cao, Jason Cain, Ruth Wang, Jan Rabaey, and Costas Spanos. Modeling within-die spatial correlation effects for process-design co-optimization. In *Sixth International Symposium on Quality of Electronic Design*, pages 516 – 521, March 2005.

[33] T. E. Gbondo-Tugbawa. *Chip-Scale Modeling of Pattern Dependencies in Copper Chemical Mechanical Polishing Process*. PhD thesis, Massachusetts Institute of Technology, 2002.

[34] W. Grobman, M. Thompson, R. Wang, C. Yuan, R. Tian, and E. Demircan. Reticle enhancement technology: implications and challenges for physical design. In *Proc. Design Automation Conf*, pages 73 – 78, 2001.

[35] P. Gupta and A. B. Kahng. Manufacturing-aware physical design. In *ICCAD*, Oct 2003.

[36] Puneet Gupta and Fook-Luen Heng. Toward a systematic-variation aware timing methodology. In *DAC '04: Proceedings of the 41st annual conference on Design automation*, pages 321–326, New York, NY, USA, 2004. ACM Press.

[37] M. R. Guthaus, N. Venkateswaran, C. Visweswariah, and V. Zolotov. Gate sizing using incremental parameterized statistical timing analysis. In *Proc. Int. Conf. on Computer Aided Design*, Nov 2005.

[38] Lei He, Andrew B. Kahng, Kingho Tam, and Jinjun Xiong. Variability-driven considerations in the design of integrated-circuit global interconnects. In *IEEE VLSI Multilevel Interconnection Conference*, Oct. 2004.

[39] Lei He, Andrew B. Kahng, Kingho Tam, and Jinjun Xiong. Design of IC interconnects with accurate modeling of chemical-mechanical planarization. In *Intl. Society for Optical Engineering (SPIE) Symposium on Microlithography*, Feb. 2005.

[40] Lei He, Andrew B. Kahng, Kingho Tam, and Jinjun Xiong. Simultaneous buffer insertion and wire sizing considering systematic CMP variation and random Leff variation. In *Proc. Int. Symp. on Physical Design*, pages 78–85, April 2005.

[41] N. Higham. Computing the nearest correlation matrix - a problem from finance. In *IMA Journal of Numerical Analysis*, volume 22, pages 329–343, 2002.

[42] R.V. Hogg and A.T. Craig. *Introduction to Mathematical Statistics*. Macmillan, 1995.

[43] R.V. Hogg and E.A. Tanis. *Probability and Statistical Inference*. Prentice Hall, 2001.

[44] J. Jess, K. Kalafala, S. Naidu, R. Otten, and C. Visweswariah. Statistical timing for parametric yield prediction of digital integrated circuits. In *Proc. Design Automation Conf*, June 2003.

[45] Tong Jing, Ling Zhang, Jinghong Liang, Jingyu Xu, Xianlong Hong, Jinjun Xiong, and Lei He. A min-area solution to performance and RLC crosstalk driven global routing problem. In *Proc. Asia South Pacific Design Automation Conf.*, pages 115–120, Jan. 2005.

[46] Andrew B. Kahng and Y. C. Pati. Subwavelength lithography and its potential impact on design and eda. In *DAC '99: Proceedings of the 36th ACM/IEEE conference on Design automation*, pages 799–804, New York, NY, USA, 1999. ACM Press.

[47] V. Khandelwal, A. Davoodi, A. Nanavati, and A. Srivastava. A probabilistic approach to buffer insertion. In *Proc. Int. Conf. on Computer Aided Design*, 2003.

[48] Vishal Khandelwal and Ankur Srivastava. A general framework for accurate statistical timing analysis considering correlations. In *Proc. Design Automation Conf*, pages 89 – 94, June 2005.

[49] J.K. Kibarian and A. Strojwas. Using spatial information to analyze correlations between test structure data. In *Semiconductor Manufacturing, IEEE Transactions on*, pages 219 – 225, Aug. 1991.

[50] A. Kurokawa, T. Kanamoto, A. Kasebe, Y. Inoue, and H. Masuda. Efficient capacitance extraction method for interconnects with dummy fills. In *Custom Integrated Circuits Conference, 2004. Proceedings of the IEEE*, pages 485 – 488, Oct. 2004.

[51] S. Lakshminarayanan, P. Wright, and J. Pallinti. Electrical characterization of the copper cmp process and derivation of metal layout rules. *IEEE Trans. on Semiconductor Manufacturing*, 16(4):668–676, 2003.

[52] J. Le, X. Li, and L. Pileggi. Stac: Statisical timing analysis with correlation. In *Proc. Design Automation Conf*, Jun 2004.

[53] Keun-Ho Lee, Jin-Kyu Park, Young-Nam Yoon, Dai-Hyun Jung, Jai-Pil Shin, Young-Kwan Park, and Jeong-Taek Kong. Analyzing the effects of floating dummy-fills: from feature scale analysis to full-chip rc extraction. In *Electron Devices Meeting, 2001. IEDM Technical Digest. International*, pages 31.3.1 – 31.3.4, Dec. 2001.

[54] Won-Seok Lee, Keun-Ho Lee, Jin-Kyu Park, Tae-Kyung Kim, Young-Kwan Park, and Jeong-Taek Kong. Investigation of the capacitance deviation due to metal-fills and the effective interconnect geometry modeling. In *Quality Electronic Design, 2003. Proceedings. Fourth International Symposium on*, pages 373 – 376, March 2003.

[55] X. Li, J. Le, M. Celik, and L. T. Pileggi. Defining statistical sensitivity for timing optimization of logic circuits with large-scale process and enviromental variations. *Proc. Int. Conf. on Computer Aided Design*, pages 844–851, November 2005. San Jose, CA.

[56] Zhuo Li and Weiping Shi. An $o(bn^2)$ time algorithm for optimal buffer insertion with b buffer types. In *Design, Automation and Test in Europe*, pages 1324–1329, March 2005.

[57] Jinghong Liang, Tong Jing, Xianlong Hong, Jinjun Xiong, and Lei He. Power/Ground network aware and row-based solutions to the crosstalk driven routing problem. In *Proc. of Intl. Conference on Application Specific Integrated Circuits*, Oct. 2005.

[58] J. Lillis, Chung-Kuan Cheng, and T.-T.Y. Lin. Optimal wire sizing and buffer insertion for low power and a generalized delay model. In *Solid-State Circuits, IEEE Journal of*, volume 31, pages 437 – 447, March 1996.

[59] Jing-Jia Liou, A. Krstic, Kwang-Ting Cheng, D.A. Mukherjee, and S. Kundu. Performance sensitivity analysis using statistical methods and its applications to delay testing. In *Design Automation Conference, Proceedings of the Asia and South Pacific*, pages 587 – 592, Jan. 2000.

[60] Jing-Jia Liou, A. Krstic, L.-C. Wang, and Kwang-Ting Cheng. False-path-aware statistical timing analysis and efficient path selection for delay testing and timing validation. In *Design Automation Conference, Proceedings*, pages 566 – 569, June 2002.

[61] Changbo Long, Jinjun Xiong, and Lei He. On optimal physical synthesis of sleep transistors. In *Proc. Int. Symp. on Physical Design*, pages 156–161, March 2004.

[62] Inc. Magma Design Automation. Quickcap user manual. In *http://www.magma-da.com/*, 2004.

[63] W. Maly, A.J. Strojwas, and S.W. Director. VLSI yield prediction and estimation: A unified framework. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 5:114–130, Jan. 1986.

[64] Murari Mani, Anirudh Devgan, and Michael Orshansky. An efficient algorithm for statistical minimization of total power under timing yield constraints. In *Proc. Design Automation Conf*, pages 309–314, June 2005.

[65] V. Mehrotra, Shiou Lin Sam, D. Boning, A. Chandrakasan, R. Vallishayee, and S. Nassif. A methodology for modeling the effects of systematic within-die interconnect and device variation on circuit performance. In *Proc. Design Automation Conf*, pages 172–175, June 2000.

[66] Carl D. Meyer. *Matrix Analysis and Applied Linear Algebra.* SIAM, 2001.

[67] P.K. Mozumder and L.M. Loewenstein. Method for semiconductor process optimization using functional representations of spatial variations and selectivity. In *Components, Hybrids, and Manufacturing Technology, IEEE Transactions on*, pages 311 – 316, June 1992.

[68] Farid N. Najm and Noel Menezes. Statistical timing analysis based on a timing yield model. In *Proc. Design Automation Conf*, pages 460 – 465, June 2004.

[69] Farid N. Najm and Noel Menezes. Statistical timing analysis based on a timing yield model. In *DAC '04: Proceedings of the 41st annual conference on Design automation*, pages 460–465, New York, NY, USA, 2004. ACM Press.

[70] S.R. Nassif. Modeling and analysis of manufacturing variations. In *Proc. IEEE Int. Conf. on Custom Integrated Circuits*, pages 223–228, May 2001.

[71] S.R. Nassif. The titanic: what went wrong! In *Proc. Design Automation Conf*, pages 349–350, June 2005.

[72] S.R. Nassif, D. Boning, and N. Hakim. The care and feeding of your statistical static timer. In *Proc. Int. Conf. on Computer Aided Design*, pages 138–139, Nov. 2004.

[73] Nagaraj NS, Tom Bonifield, Abha Singh, Clive Bittlestone, Usha Narasimha, Viet Le, and Anthony Hill. Beol variability and impact on rc extraction. In *DAC '05: Proceedings of the 42nd annual conference on Design automation*, pages 758–759, New York, NY, USA, 2005. ACM Press.

[74] M. Orshansky and A. Bandyopadhyay. Fast statistical timing analysis handling arbitrary delay correlations. In *Proc. Design Automation Conf*, June 2004.

[75] M. Orshansky, L. Milor, P. Chen, K. Keutzer, and C. Hu. Impact of spatial intrachip gate length variability on the performance of high-speed digital circuits. In *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pages 544–553, May 2002.

[76] Michael Orshansky, Linda Milor, Ly Nguyen, Gene Hill, Yeng Peng, and Chenming Hu. Intra-field gate cd variability and its impact on circuit performance. In *IEEE IEDM Tech. Dig.*, pages 479–482, Dec. 1999.

[77] Jin-Kyu Park, Keun-Ho Lee, Joo-Hee Lee, Young-Kwan Park, and Jeong-Taek Kong. An exhaustive method for characterizing the interconnect capacitance considering the floating dummy-fills by employing an efficient field solving algorithm. In *Simulation of Semiconductor Processes and Devices, 2000. SISPAD 2000. 2000 International Conference on*, pages 98 – 101, Sept. 2000.

[78] S. Raj, S. Vrudhula, and J. Wang. A methodology to improve timing yield in the presence of process variation. In *Proc. Design Automation Conf*, Jun 2004.

[79] R. Rao, A. Devgan, D. Blaauw, and D. Sylvester. Parametric yield estimation considering leakage variability. In *Proc. Design Automation Conf*, June 2004.

[80] Inc. Rio Design Automation. Riomagic user manual. In *http://www.rio-da.com/*, 2005.

[81] Michelle Schatzman. *Numerical analysis: a mathematical introduction*. Clarendon Press, Oxford, 2002.

[82] Semiconductor Industry Association. *International Technology Roadmap for Semiconductors*, 2003.

[83] W. Shi and Z. Li. An o(nlogn) time algorithm for optimal buffer insertion. In *Proc. Design Automation Conf*, Jun 2003.

[84] Jaskirat Singh, Vidyasagar Nookala, Zhi-Quan Luo, and Sachin S. Sapatnekar. Robust gate sizing by geometric programming. In *Proc. Design Automation Conf*, pages 315–320, June 2005.

[85] Debjit Sinha, Narendra V. Shenoy, and Hai Zhou. Statistical gate sizing for timing yield optimization. In *Proc. Int. Conf. on Computer Aided Design*, pages 1037–1042, 2005.

[86] B.E. Stine, D.S. Boning, and J.E. Chung. Analysis and decomposition of spatial variation in integrated circuit processes and devices. In *Semiconductor Manufacturing, IEEE Transactions on*, pages 24 – 41, Feb. 1997.

[87] B.E. Stine, D.S. Boning, J.E. Chung, L. Camilletti, F. Kruppa, E.R. Equi, W. Loh, S. Prasad, M. Muthukrishnan, D. Towery, M. Berman, and Kapoor A. The physical and electrical effects of metal-fill patterning practices for oxide chemical-mechanical polishing processes. *Electron Devices, IEEE Transactions on*, 45:665 – 679, March 1998.

[88] Haihua Su, F. Liu, A. Devgan, E. Acar, and S. Nassif. Full chip leakage-estimation considering power supply and temperature variations. In *Proc. Int. Symp. on Low Power Electronics and Design*, pages 78–83, Aug. 2003.

[89] R. Tian, X. Tang, and D. Wong. Dummy-feature placement for chemical-mechanical polishing uniformity in a shallow trench isolation process. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 21:63 – 71, Jan 2002.

[90] R.-S. Tsay. An exact zero-skew clock routing algorithm. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, CAD-12(2):242–249, February 1993.

[91] T. Tugbawa, T. Park, D. Boning, T. Pan, P. Li, S. Hymes, T. Brown, and L. Camilletti. A mathematical model of pattern dependencies in cu cmp processes. In *CMP Symposium, Electrochemical Society Meeting*, Oct 1999.

[92] L. P. P. P. van Ginneken. Buffer placement in distributed RC-tree networks for minimal Elmore delay. In *Proc. IEEE Int. Symp. on Circuits and Systems*, pages 865–868, 1990.

[93] C. Visweswariah. Death, taxes and failing chips. In *DAC 03*, June 2003.

[94] C. Visweswariah, K. Ravindran, K. Kalafala, S. Walker, and S. Narayan. First-order incremental block-based statistical timing analysis. In *Proc. Design Automation Conf*, June 2004.

[95] Chandu Visweswariah. Statistical analysis and design of digital integrated circuits. In *EDA Forum 2005, Hanover, Germany*, Nov. 2005.

[96] Robert Ware and Frank Lad. Approximating the distribution for sums of products of normal variables. In *Technical Report, UCDMS 2003/15, The Mathematics and Statistics department at Canterbury University*, 2003.

[97] Jinjun Xiong, Jun Chen, James Ma, and Lei He. Post global routing optimization with RLC crosstalk constraints. In *Proc. Int. Conf. on Computer Aided Design*, pages 504–509, Nov. 2002.

[98] Jinjun Xiong and Lei He. Full-chip multilevel routing for power and signal integrity. In *Proc. Design Automation and Test in Europe*, pages 1116 – 1121, Feb. 2004.

[99] Jinjun Xiong and Lei He. Full-chip routing optimization with RLC crosstalk budgeting. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 23:366–377, March 2004.

[100] Jinjun Xiong and Lei He. Integrity-driven power and signal network co-design. In *ACM/IEEE International Workshop on Timing Issues, Austin, Texas*, Feb. 2004.

[101] Jinjun Xiong and Lei He. Extended global routing with RLC crosstalk constraints. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, 13:319–329, March 2005.

[102] Jinjun Xiong and Lei He. Probabilistic congestion model considering shielding for crosstalk reduction. In *Proc. Asia South Pacific Design Automation Conf.*, pages 739–742, Jan. 2005.

[103] Jinjun Xiong and Lei He. Fast buffer insertion considering process variation. In *Proc. Int. Symp. on Physical Design*, April 2006.

[104] Jinjun Xiong and Lei He. Full-chip multilevel routing for power and signal integrity. *Integration, the VLSI Journal*, 2006.

[105] Jinjun Xiong, Kingho Tam, and Lei He. Buffer insertion considering process variation. In *Proc. Design Automation and Test in Europe*, pages 970–975, 2005.

[106] Jinjun Xiong, Yiu-Chung Wong, Egino Sarto, and Lei He. Constraint driven I/O planning and placement for chip-package co-design. In *Proc. Asia South Pacific Design Automation Conf.*, Jan. 2006.

[107] Jinjun Xiong, Vladimir Zolotov, and Lei He. Robust extraction of spatial correlation. In *Proc. Int. Symp. on Physical Design*, April 2006.

[108] Jinjun Xiong, Vladimir Zolotov, Natesan Venkateswaran, and Chandu Visweswariah. Criticality computation in parameterized statistical timing. In *ACM/IEEE International Workshop on Timing Issues, San Jose, California*, Feb. 2006.

[109] Jinjun Xiong, Vladimir Zolotov, Natesan Venkateswaran, and Chandu Visweswariah. Criticality computation in parameterized statistical timing. In *Proc. Design Automation Conf*, July 2006.

[110] A.M. Yaglom. Some classes of random fields in n-dimensional space, related to stationary random processes. In *Theory Probability Applications*, volume 2, pages 273–320, 1957.

[111] Ling Zhang, Tong Jing, Xianlong Hong, Jingyu Xu, Jinjun Xiong, and Lei He. Performance optimization global routing with RLC crosstalk constraints. In *Proc. of Intl. Conference on Application Specific Integrated Circuits*, pages 191–194, Oct. 2003.

[112] Ling Zhang, Tong Jing, Xianlong Hong, Jingyu Xu, Jinjun Xiong, and Lei He. Performance and RLC crosstalk driven global routing. In *International Symposium on Circuits and Systems*, pages 65–68, May 2004.

[113] Lizheng Zhang, Weijen Chen, Yuhen Hu, John A. Gubner, and Charlie Chung-Ping Chen. Correlation-preserved non-gaussian statistical timing analysis with quadratic timing model. In *Proc. Design Automation Conf*, pages 83 – 88, June 2005.

[114] Lizheng Zhang, Weijen Chen, Yuhen Hu, John A. Gubner, and Charlie Chung-Ping Chen. Statistical timing analysis with extended pseudo-canonical timing model. In *Proc. Design Automation and Test in Europe*, pages 952 – 957, March 2005.

[115] Xin Zhao, Yici Cai, Qiang Zhou, Xianlong Hong, Lei He, and Jinjun Xiong. Shielding area optimization under the solution of interconnect crosstalk. In *International Symposium on Circuits and Systems*, pages 23–26, May 2004.

[116] Vladimir Zolotov, Jinjun Xiong, and Chandu Visweswariah. Computation of yield gradients from statistical timing analysis. In *ACM/IEEE International Workshop on Timing Issues, San Jose, California*, Feb. 2006.