ON OPTIMAL INTERCONNECTIONS FOR VLSI

ON OPTIMAL INTERCONNECTIONS FOR VLSI

Andrew B. Kahng
University of California
Los Angeles, California, USA

■

Gabriel Robins
University of Virginia
Charlottesville, Virginia, USA

KLUWER ACADEMIC PUBLISHERS
Boston/London/Dordrecht

To the field of VLSI CAD

CONTENTS

LIST OF FIGURES

Chapter 5

LIST OF TABLES

1 PRELIMINARIES

1.1 PREFACE

This book discusses problems of "optimal interconnection" and describes efficient algorithms for several basic formulations. Our domain of application is the computer-aided design (CAD) of very large-scale integrated (VLSI) circuits, wherein interconnection design is now one of the most actively studied areas. However, much of what we develop can be applied to other domains ranging from urban planning to the design of communication networks. Because most formulations that we study are intractable, the term "optimal" in some sense is a misnomer: rather, our focus is on the reasoned and principled development of good heuristics.

This book is an outgrowth of the 1992 Ph.D. dissertation of Gabriel Robins [203] at the UCLA Computer Science Department. As such, it retains a highly personal perspective: it gives a retrospective of our own research, and it is colored by our research interests and our background in discrete algorithms and optimization. Our treatment also attempts to convey a sense of history – how our field has co-evolved with an emerging "science of VLSI design". With recent years having seen VLSI designs become increasingly performance-dominated, and thus interconnect-dominated, VLSI interconnections are indeed a rich domain for this historical view. In particular, our research on interconnection design has spanned the field's rapid transition from purely geometric formulations to more "physically-motivated" formulations.

Although we do not attempt an encyclopedic treatment, we do describe key relevant works, and the discussion is largely self-contained. We envision that this book will be useful as a reference for researchers and CAD algorithm developers, or as reading for a seminar on VLSI CAD, heuristic algorithms, or geometric optimization. Our own codes, which are cited throughout the book, are freely available to interested parties; see our contact information below.

1.2 THE DOMAIN OF DISCOURSE: ROUTING IN VLSI PHYSICAL DESIGN

Let us first outline the context for our particular subfield of VLSI CAD, namely, the global routing phase of physical design. For more complete reviews of VLSI design, and physical design in particular, the reader is referred to [168, 182, 194, 216].

The goal of VLSI CAD is to transform a high-level system description into a set of mask geometries for fabrication. This is typically accomplished by the following sequence of stages (see Figure 1.1).

- **Design Specification:** Starting from a real-world requirement (e.g. "secure communication"), a high-level system description (e.g., the "DES" data encryption standard) is developed which includes such parameters as architecture, performance, area, power, cost and technology.
- **Functional Design:** The design is transformed into a behavioral specification which captures the system I/O behavior using mathematical equations, timing diagrams, instruction sets and other devices.
- **Logic Design:** The functional design is represented in logical form, typically via Boolean expressions which may be subsequently optimized to reduce the complexity of the system description.
- **Structural Design:** The logic design is represented as a circuit using components from an available library of modules (e.g., NAND and NOR gates, standard cells, or building-block macros); this may also involve technology mapping steps.
- **Physical Design:** The structural design is transformed into the mask geometry for fabrication while adhering to underlying design rules for the chosen technology.

The last stage in this process, **physical design**, contains our area of interest. Physical design consists of two major steps. First, the *placement* step maps

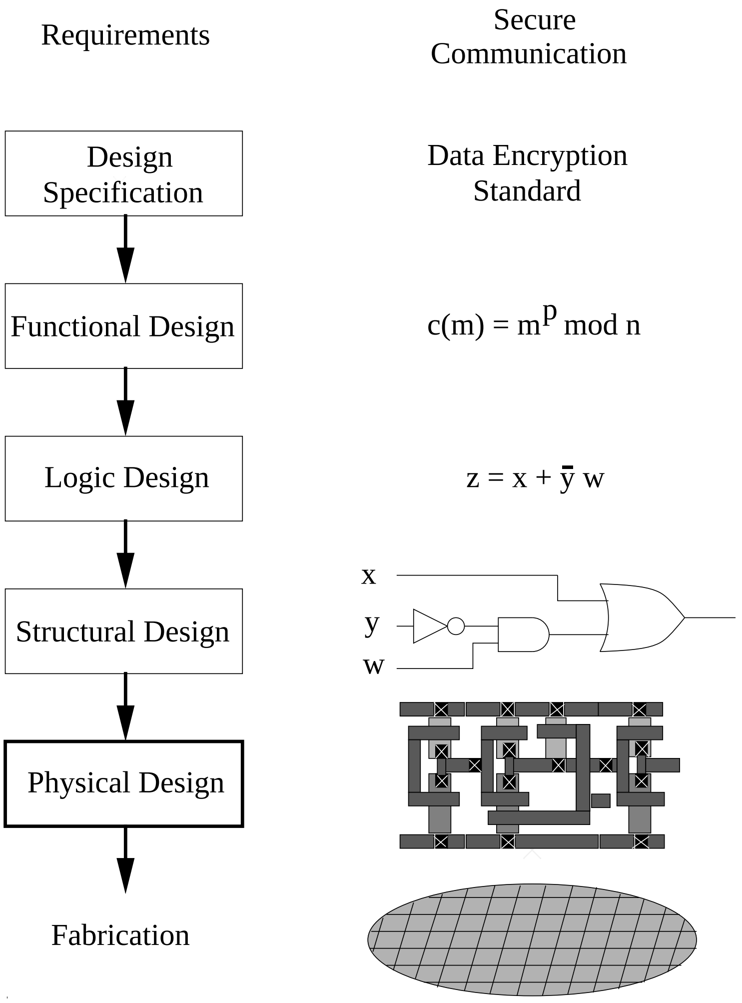

Figure 1.1 The VLSI design process.

functional units (modules) onto portions of a *layout region*, e.g., the surface of a chip. Second, the *routing* step interconnects specified sets of terminals, i.e., the *signal nets* of the design, by wiring within *routing regions* that lie between or over the functional units. (A signal net consists of a module output terminal together with the various module input terminals to which the output signal must be delivered.)

Within the field of physical design, prevailing objectives have evolved over the years in response to advances in VLSI technology. When system operating frequencies were dominated by device switching speeds, placement and routing optimizations centered on reduction of total routing area. Subsequent advances in fabrication technology have increased packing densities, allowing more and faster devices to be placed on larger ICs. Leading-edge fabrication technology now goes well into submicron feature sizes, and circuit speeds are approaching gigahertz frequencies. The reduced feature size implies more resistive interconnects, and increased system complexity implies larger layout regions. Thus, minimization of interconnection delay has become the major concern in physical design.

In light of this trend, *performance-driven* physical design has seen much research activity within the past five years. Early works focused on performance-driven placement, with the standard objective being the close placement of modules belonging to timing-critical paths. However, performance-driven placement algorithms will achieve their intended effect only when the associated routing algorithms can realize the full potential of a high-quality placement. Thus, the emphasis in routing objectives has shifted from area minimization to delay minimization, and more recently to the *control* of interconnect delay (e.g., by limiting skews or delays at particular terminals). This range of routing objectives – area, delay, skew and beyond – defines the scope of this book.

Once an objective has been established, the actual routing of a given signal net can be decomposed into *global* and *detailed* routing. The *global routing* phase is a higher-level process during which the routing topologies of signal nets are defined over the available routing regions. Then, the *detailed routing* phase produces the actual geometries which realize the required connectivity on the fabricated chip. Our work applies to the global routing phase of physical design.[1]

We assume that during the global routing phase, all module and terminal locations have already been fixed in the plane, so that we need only ensure

[1] This traditional taxonomy may seem ambiguous. We do not address standard "detailed routing" topics such as switchbox routing or river routing. However, optimizing routing area and performance requires a concern with the specific geometry of the routing. In our discussion, we will define a routing topology by specifying for each edge its length and width, and the location of its endpoints; our work addresses "global routing" in that the particular detailed embedding of an edge between its endpoints does not matter.

electrical connectivity of the signal nets. With *standard-cell* or *gate-array* design methodologies, which have many small functional modules, global routing may be viewed as taking place in Manhattan geometry, i.e., distances between terminals are given by rectilinear distance. In other words, these design methodologies possess sufficiently high *porosity* that the routing problem can be formulated in the geometric plane. On the other hand, *building-block* design methodologies involve larger functional blocks or macro cells. Since these are often treated as obstacles, the routing problem is formulated with respect to a weighted *routing graph* that represents the available routing area. A standard model is the *channel intersection graph* (CIG), where each edge represents a *channel*(i.e., the empty rectangular space between adjacent modules) and each vertex corresponds to the intersection of two orthogonal channels [193] (see Figure 1.2). The edge weights of the CIG can be used to model channel width or congestion.

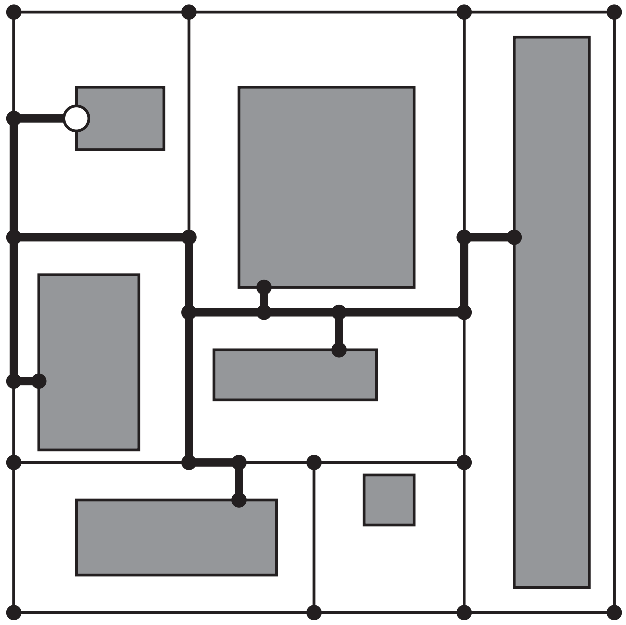

Figure 1.2 A channel intersection graph induced by a set of modules, and a routing tree that connects the highlighted terminals. The source is shown by a hollow dot.

A "true" global router processes multiple signal nets *simultaneously* using such techniques as simulated annealing, multicommodity flow or mathematical programming. However, many existing codes are *sequential*, or *"net-at-a-time"*, in that they establish a heuristic ordering of nets for routing and use ripup-and-retry techniques when the routing fails. (There are also even more fine-grain methods which route individual two-terminal subnets of signal nets.) With either type of global router, the key operation is to compute a good routing topology over a *single* signal net: hence, this book deals exclusively with methods that route a single net at a time.

As with previous routing constructions that have formed the basis of new global routers (e.g., "Steiner min-max trees"), each method that we develop can be transparently integrated into existing global routing approaches. In the mathematical programming approach, finding a routing solution for a given net generates a new entering basis column within a primal-dual iteration. In the sequential approach, routing solutions are found for the highest-priority nets first, leaving lower-priority nets to encounter more congestion and blockage. After each net is routed, the routing region costs (e.g., CIG edge weights) can be updated before the next net is processed.

We conclude this section with a review of basic conventions and terminology used throughout the book. We define a *terminal* to be a given location in the layout region. A signal *net* $S = \{s_0, s_1, s_2, \ldots, s_n\}$ is a set of $n+1$ terminals, with one terminal $s_0 \in S$ a designated *source* and the remaining terminals *sinks*. A *routing solution* is a set of wires that connects, i.e., spans, the terminals of a net so that a signal generated at the source will be propagated to all the sinks.

The rectilinear wiring technology implies an underlying "Manhattan" geometry, where the *distance* between points a and b is $d(a,b) = |a_x - b_x| + |a_y - b_y|$, i.e., the sum of the differences in their x- and y-coordinates. A *segment* is an uninterrupted horizontal or vertical wire, and any connection between two terminals will consist of one or more wire segments. VLSI and printed circuit board technologies admit multiple routing layers, where a preferred-direction routing methodology is used to facilitate design, manufacturability and reliability. In other words, the available wiring layers are partitioned, with horizontal wire segments preferentially routed on certain layers, and vertical wire segments routed on the other layers. A connection between two wire segments from different layers is called a *via*.

Sometimes it is convenient to embed S in an underlying *routing graph* $G = (V, E)$, consisting of a set of vertices V and a set of edges $E \subseteq V \times V$. Thus, the set of terminals is some $S \subseteq V$. A *subgraph* of G is a graph $G' = (V', E')$

with $V' \subseteq V$ and $E' \subseteq E$, and $E' \subseteq V' \times V'$. A routing solution is a subgraph of G that spans S. A *path* between two vertices $x, y \in V$ is a sequence of k edges of the form $(x, v_{i_1}), (v_{i_1}, v_{i_2}), \ldots, (v_{i_k}, y)$, where $(v_{i_m}, v_{i_{m+1}}) \in E$ for all $1 \leq m \leq k-1$. A graph is *connected* if there exists a path between each pair of vertices. A graph is a *tree* if it is connected but the removal of any edge will disconnect it. Since a tree topology uses the fewest edges of any spanning graph over the signal net, i.e., $|S| - 1 = n$ edges, routing formulations typically seek a tree topology.

A *weighted graph* has a non-negative real weight assigned to each of its edges. The *cost* of a weighted graph is the sum of its edge weights. A *shortest path* in G between two vertices $x, y \in V$, denoted by $minpath_G(x, y)$, is a minimum-cost path connecting x and y. In a tree T, $minpath_T(x, y)$ is simply the unique path between x and y. For a weighted graph G we use $dist_G(x, y)$ to denote the cost of $minpath_G(x, y)$. The distance from the source to a given sink s_i in a tree is denoted as $l_i = dist_T(s_0, s_i)$.

Because a signal net is inherently oriented from its source to its sinks, we use the special notation R_i to denote the cost of the shortest s_0-s_i path in G, i.e., $R_i = dist_G(s_0, s_i)$. We use R to denote the maximum R_i value over all sinks s_i, and say that R is the *radius* of the signal net. The radius of a routing tree T is $r(T) = \max_{1 \leq i \leq n} l_i$. Additional terminology will be developed throughout the following chapters, as needed. The reader is referred to, e.g., [67] or [92] for a more rigorous development of basic graph-theoretic concepts.

As noted at the outset, most problems encountered in VLSI CAD, including all of the interconnection formulations that we address, are intractable. While we resort to heuristic solutions, a basic precept in our work is to prove that our proposed heuristics perform well. For example, we often strive to show that the heuristic solution cost in the worst case (or average case) is no more than a constant factor from optimal. Since the practical relevance of a heuristic may hinge on issues beyond asymptotic time and space complexity, we also augment our performance bounds with empirical simulations using standard test cases from the literature, e.g., those maintained by ACM SIGDA (currently available by anonymous ftp to <mcnc.org>).

1.3 OVERVIEW OF THE BOOK

Beyond its sketch of our application domain of VLSI routing, the present chapter also surveys the main results contained in this book. Chapters 2, 3 and 4 are respectively entitled Area, Delay, and Skew. These form the core of the book, and address three fundamental routing objectives: (i) minimization of total wirelength, (ii) minimization of signal delay, and (iii) minimization of skew among signal arrival times. Chapter 5 provides new frameworks for the simultaneous optimization of multiple competing objectives; one such framework allows various unifications of the techniques developed in the preceding three chapters. The following subsections summarize the key developments of each chapter.

1.3.1 Minimum Area: The Steiner Minimal Tree Problem

VLSI design rules dictate a minimum separation between wires, and therefore the area occupied by the routing on a chip is roughly proportional to the total wirelength of the routing. Added wirelength generally increases signal delay and power consumption due to increased resistance and capacitance. Other system cost measures, e.g., those based on fabrication cost, yield and reliability, also increase with chip area. Thus, a fundamental objective is to minimize the total wirelength required to connect a prescribed set of points in the plane, i.e., the terminals of a given signal net. The subject of Chapter 2 is the *Steiner minimal tree* (SMT) problem, which for a given net S asks for a set S' of *Steiner points* such that the total edgelength of the minimum spanning tree (MST) over $S \cup S'$ is minimized. The main insight is that the points of S' will serve as internal nodes of the tree – "intermediate junction points" – which reduce the interconnection cost. Without introducing such points, the minimum-cost solution would simply be a minimum spanning tree over S.

The SMT problem is well-studied in combinatorial optimization and network design; see the monographs [138] and [139]. The geometry of VLSI, which usually allows only vertical and horizontal wiring directions, has motivated studies of the *rectilinear* version of the problem, typically for the wirelength estimation and global routing phases of layout design. With only a few highly constrained exceptions, existing variants of the SMT problem are NP-complete. Most SMT heuristics in the literature have analogies to classic minimum spanning tree constructions; this is in part due to the MST being a constant-factor approximation to the SMT, with performance ratio $\frac{3}{2}$ in the rectilinear metric. However, the first result of Chapter 2 defines a general class of "MST-based"

SMT heuristics, and shows that such methods cannot have performance ratio better than that of the simple MST approximation.

The focus of Chapter 2 lies in developing the Iterated 1-Steiner (I1S) heuristic, which iteratively finds optimal Steiner points that are added directly into the set S. The I1S construction thus avoids traditional analogies to minimum spanning tree solutions, and in practice achieves good performance even on inputs that are pathological for previous heuristics. For random 8-point planar instances, I1S solution costs are optimal for 90% of all instances, and average within 0.25% of optimal overall. (The I1S approach also applies to graph instances and higher-dimensional geometric instances.) The chapter describes a straightforward, efficient implementation of I1S, along with such enhancements as a parallel implementation that achieves near-linear speedup. Similarities between I1S and the recent method of Zelikovsky are also discussed.

Finally, Chapter 2 develops the result that any pointset in the Manhattan plane has an MST with maximum degree 4, and that in three-dimensional Manhattan space the maximum MST degree is 14 (the best previous bounds were 6 and 26, respectively); this improves I1S runtimes and is also of independent theoretical interest. The chapter concludes with a discussion of the Steiner problem in graphs.

1.3.2 Minimum Delay: Toward Optimal-Delay Routing Trees

Chapter 3 considers minimization of signal delay, which is synonymous with "performance-driven" system design. As VLSI technology scales to smaller feature sizes and larger layout areas, signal delays become interconnect-dominated, i.e., signal delay through interconnects increasingly dominates delay through devices. In leading-edge technologies, minimum-delay wiring topologies can differ substantially from minimum-area (SMT) wiring topologies.

The signal delay objective takes us from the *unoriented* pointset of the Steiner minimal tree problem to an *oriented* collection of terminals in the layout plane. Such a collection of terminals, which we call a *signal net*, has one identified *source* terminal; the remaining terminals are *sinks*. Typically, the source terminal is the output of a gate, and the sinks are the fanins for that output signal at inputs of other gates.

The discussion of Chapter 3 centers on four issues which have guided recent progress in minimum-delay routing heuristics. First, there is the issue of *technology-dependence* in the routing construction, e.g., a simple analysis of Elmore delay in distributed RC trees shows that routing objectives should be dependent on parameters of the prevailing interconnect technology. We thus give a taxonomy of methods based on their tunability to specific technology parameters and signal net criticalities, and demonstrate the advantages of such tunable methods as the "Elmore routing tree" approach and the Prim-Dijkstra tradeoff.

Second, the chapter compares *"actual delay"*, versus *geometric*, routing objectives. To a first-order approximation, signal delay from the source to a given sink is proportional to the source-sink pathlength in the routing tree. This *linear delay* approximation suggests minimizing the maximum source-sink pathlength in the routing tree (i.e., a geometric "minimum-radius" criterion). On the other hand, reducing the total cost of the routing tree will reduce its lumped capacitance (i.e., a geometric "minimum-cost" criterion). We review how early works employed geometric criteria to achieve tractability in both the design and the analysis of routing heuristics. Of particular interest is a "bounded-radius, bounded-cost" (BRBC) approach which seeks a minimum-cost routing tree subject to a given bound on tree radius; we describe an algorithm which simultaneously minimizes both tree cost and tree radius to within constant factors of optimal. The BRBC approach and its analysis generalize to Steiner routing and to routing in arbitrary weighted graphs that capture the variation of routing costs over the layout region. The chapter gives details of recent methods, notably the "Elmore routing tree" variants which obtain reduced signal delays by optimizing higher-order delay estimates directly.

Third, we discuss minimization of *sink-dependent* delay, as opposed to *net-dependent* delay. Here, the key observation is that timing-driven placement and routing are typically iterated with static timing estimation, so that critical-path information is available during the routing tree construction. With this in mind, the traditional objective of minimizing maximum sink delay is "net-dependent" in that it ignores available path-dependent information. An approach which optimizes delay to identified critical sinks, such as that given in 1993 by Boese, Kahng and Robins [34], seems better matched to modern design methodologies. More recent work of Boese et al. provides an interesting addendum to the earlier SMT discussion: it generalizes Hanan's theorem to Elmore delay-optimal Steiner trees and gives a new "peeling" decomposition for optimal Steiner trees.

Finally, Chapter 3 addresses the issue of *demonstrable quality* for minimum-delay routing heuristics. Analogous to the empirical studies of the I1S SMT heuristic in Chapter 2, we present empirical studies showing near-optimality of a construction for minimum Elmore delay at prescribed critical sinks. The chapter concludes with a review of two other recent advances in performance-driven interconnect design; these involve wiresizing and non-tree routing techniques. An Appendix provides the basic theory behind several efficient delay estimates, and also discusses measures of accuracy and fidelity for the linear, Elmore, and two-pole delay approximations.

1.3.3 Minimum Skew: The Zero-Skew Clock Routing Problem

In a high-performance VLSI design, circuit speed is limited not only by the signal propagation within and between circuit elements, but also by the *skew* between signal arrival times. The form of skew most often studied is *clock skew*, i.e., the difference between longest and shortest arrival times of a clock signal at synchronizing elements of the circuit. Clock skew minimization, and in particular the "zero-skew clock routing" problem, has become a central issue in the design of leading-edge systems. However, it should be noted that skew control for arbitrary signal nets is also of increasing importance, as are related problems of *prescribed-skew* or *bounded-skew* routing.

Chapter 4 discusses clock tree construction to minimize skew and wirelength as a combination of two processes: *topology generation*, and *geometric embedding* of the topology. We present methods which accomplish each of these processes using either the linear or Elmore delay model to guide the construction. Our discussion focuses on so-called "exact zero skew" clock routing constructions.

The first part of Chapter 4 uses the linear delay model to motivate a *pathlength-balanced tree* problem formulation, which seeks a minimum-cost tree with all source-sink pathlengths of equal length. We describe a simple approach, based on iterative geometric matching, for generating a clock tree topology while simultaneously embedding it in the layout region.

The second part of the chapter describes the Deferred-Merge Embedding (DME) algorithm, which embeds any prescribed connection topology (i.e., a binary tree with the clock sinks at the leaves), so as to create a clock tree with zero skew while minimizing total wirelength. The algorithm runs in linear time, and always yields *exact* zero skew trees with respect to a given monotone delay model such as linear or Elmore delay. The DME method achieves substantial cost reductions over earlier constructions, and can be combined with previous methods that concentrate on generation of the clock tree topology.

Finally, the third part of the chapter unifies the topology generation and geometric embedding of exact zero-skew clock trees. Under the linear delay model, the two phases of the DME algorithm (bottom-up identification of loci for "zero-skew balance points", followed by top-down selection of these balance points within a minimum-delay zero-skew embedding) can be replaced by a single top-down phase. Where DME would nominally require a prescribed topology as input, this top-down construction allows the clock tree topology to be determined dynamically and flexibly while being optimally embedded at the same time. A natural outgrowth is a DME-like algorithm for *single-layer*, exact zero-skew clock routing; such a construction is increasingly sought to minimize signal attenuation through vias, simplify buffering optimizations, and maximize process-variation independence.

Chapter 4 also describes extensions of these clock routing methods to "min-max" delay constraints and *bounded-skew* routing for general signal nets. The chapter concludes by noting additional issues and problem formulations, including optimal buffering hierarchies for minimum phase delay, and multiple-level clock trees for multi-chip module packaging.

1.3.4 Multiple Objectives

The last chapter of the book, Chapter 5, discusses frameworks and techniques which enable the simultaneous optimization of multiple competing objectives. Section 5.1 notes that beyond the nominal total wirelength, the grid-based structure of VLSI routing resources provides additional information for determining the impact of a given routing solution on layout area. The discussion explores a new *minimum density* objective for spanning and Steiner tree constructions, which seeks to balance the use of horizontal and vertical routing resources. We describe two heuristic constructions for low-density spanning trees whose outputs are within small constants of optimal with respect to both tree cost and density. (The proof techniques suggest a constructive lower bound scheme which affords tighter estimates of solution quality for a given problem instance.) Of particular interest is that the minimum density objective can be transparently combined with, e.g., minimum radius or minimum skew – without affecting asymptotic solution quality with respect to these competing objectives.

While previous chapters each focus on a fundamental routing criterion (i.e., area, delay or skew), many secondary objectives may exist, including congestion avoidance, jog minimization, reliability, etc. Section 5.2 develops a general framework of *multi-weighted graphs*, in which multiple competing objectives can be simultaneously optimized. This is accomplished by assigning to each edge a *vector* of weights, corresponding to the various optimization criteria; graph searches are then guided by the weighted average of the edge weights according to designer-specified tradeoff parameters. This framework is applicable to graph-based routing regimes, such as building-block design and field-programmable gate array layout.

Finally, we describe optimization within the framework of a *continuously-weighted layout region*, which can be induced by the simultaneous consideration of multiple criteria (e.g., reliability, thermal density, and routing congestion). Within this framework, we consider a problem which has applications ranging from circuit board routing to vehicle navigation, namely, finding a minimum-cost prescribed-width path connecting a given source and destination [131]. Previous path routing approaches such as Dijkstra's algorithm implicitly assume that the path is of zero width, but this assumption is usually not realistic (e.g., consider routing a wide bus, or traces on a circuit board). Section 5.3 develops a network-flow based approach to prescribed-width routing in a continuously weighted region. Interestingly, the extension to higher dimensions can solve a discrete version of Plateau's problem, which seeks a minimum-area surface that spans a given closed curve [130].

1.4 ACKNOWLEDGMENTS

This book is the product of the research, suggestions, and technical assistance of many individuals. We first thank the students who have been so dedicated to the research that forms the basis of this book. In alphabetical order[2], they are: Mike Alexander, Charles J. Alpert, Kenneth D. Boese, Dennis Jen-Hsin Huang, Berni A. McCoy, Chung-Wen Albert Tsao and Tongtong Zhang. Any list of specific debts must begin with Ken Boese, who developed much of the core material in Chapters 3 and 4, including the characterization of delay-optimal routing trees and the results concerning the DME clock routing algorithm. The precise exposition in these sections is a product of Ken's efforts. Berni McCoy dedicated well over a year to investigations of accuracy and fidelity of delay estimates, near-optimality of the ERT construction, dynamic wiresizing and non-tree routing – these results appear throughout Chapter 3. Mike Alexander developed the graph generalization of I1S in Chapter 2, as well as the multi-

[2] Since early 1991, listing names in alphabetical order has been the "official" policy on all our publications as well.

weighted graphs framework of Chapter 5. Chuck Alpert and Dennis Huang pursued the Prim-Dijkstra tradeoff of Chapter 3 through its many incarnations; Chuck also contributed to the study of minimum-density routing trees in Section 5.1. Albert Tsao developed the Planar-DME algorithm which forms the capstone of Chapter 4. Ken and Chuck, along with Lars Hagen, provided many critical comments as this book took shape. Brett Coryell and Brian Robinson provided invaluable help throughout the final stages of writing. Certainly, it is our students who have always been our best critics, motivators, and colleagues.

The various research collaborations that form the basis of this book list a number of other coauthors: Tim Barrera, Ting-Hai Chao, Jim Cohoon, Jason Cong, Todd Hodes, Joseph Ganley, Jeff Griffith, Jan-Ming Ho, Yu-Chin Hsu, T. C. Hu, David Karger, Kwok-Shing Hardy Leung, Sally McKee, Sudhakar Muddu, Jeff Salowe, Majid Sarrafzadeh, C. K. Wong and Dian Zhou. David Karger suggested the second Prim-Dijkstra tradeoff of Chapter 3. Dian Zhou provided us with the original "Two-Pole" simulator code, while Sudhakar Muddu provided invaluable amendments to this code and the totality of our knowledge concerning delay analysis of interconnects. Jason Cong provided the geometric analysis of H-flipping cited in Chapter 4, as well as the bounded-radius minimum routing tree problem formulation in Chapter 3. Since 1990, Jason and his students have brought much energy to VLSI CAD at UCLA.

Others who have over the years provided advice, feedback, and/or use of their codes include: Jim Aylor, Marshall Bern, Stephen Brown, John Canny, Pak Chan, Kamal Chaudhary, Brett Coryell, Erik Cota-Robles, Wayne Wei-Ming Dai, Miloš Ercegovac, Eli Gafni, Basil Gordon, Sheila Greibach, Lars Hagen, Rajeev Jain, Kevin Karplus, Samir Khuller, Ernest S. Kuh, C. L. Liu, John Pfaltz, Sinai Robins, Brian Robinson, Jonathan Rose, Andy Schwab, Michael Shur, Ashok Vittal and Neal Young. Eli Gafni provided the key pointer to the shallow-light construction of Awerbuch, Baratz and Peleg that led to the BRBC algorithm in Chapter 3.

The dedication of this book, *"To the field of VLSI CAD"*, requires some elaboration. As newcomers to the field, we are grateful for the inspiration provided by the leading researchers who preceded us. Above all, Professor T. C. Hu of U.C. San Diego has been the one constant source of guidance, wisdom and research interaction in our academic careers. His influence predates our studies in VLSI CAD, and goes much deeper; he has truly shaped us both. Professor Ernest S. Kuh of U.C. Berkeley has profoundly influenced how the field of VLSI CAD is defined today. There is an ethic of quiet excellence in "kuhsgroup" alumni: Professors Chung-Kuan Cheng, Kwang-Ting (Tim) Cheng, Wayne Wei-Ming Dai and Malgorzata Marek-Sadowska, as well as Professor Kuh himself, will

long remain our models of collegiality, activity and impact. Professor C. L. Liu of the University of Illinois has for several years given wholeheartedly of his experience, advice, and support. He is a gifted teacher, scholar and raconteur, and it is always a rare pleasure to be in his company. His former students – Jason Cong and Martin Wong in particular – are of course models for all young faculty in the field. Further inspirations have derived from Majid Sarrafzadeh at Northwestern University; Daniel Gajski and his group at U.C. Irvine; Robert Brayton at U.C. Berkeley; Thomas Lengauer at GMD Bonn; Ralph Otten at Delft; and many others. The field that brings such individuals together truly deserves to flourish.

On a more personal level, Gabriel Robins would like to thank Bill Wulf and Anita Jones for all their support and sage advice, and for inspiring and nurturing so much of the shared vision that is unique to computer science at the University of Virginia. Randy Pausch has been a continuing source of inspiration, and a firm advocate of "the right culture". Together, the UVa Department of Computer Science and its Chair Jim Ortega deserve much credit for their support of young faculty development.

Our work was supported by National Science Foundation Young Investigator Awards MIP-9257982 and MIP-9457412, by National Science Foundation Research Initiation Award MIP-9110696, by Army Research Office grants DAAK-70-92-K-0001 and DAAL-03-92-G-0050, by setup funds provided by the UCLA School of Engineering and Applied Science during 1989-1991, by research initiation funds provided by the University of Virginia School of Engineering during 1992-1993, and by an IBM Graduate Fellowship. Part of this book was written during a Spring 1993 sabbatical that was hosted by Professor Ernest S. Kuh and his research group. Finally, this book would not exist without the incredible patience of Carl Harris at Kluwer Academic Publishers – a debt that goes beyond any possible statement of thanks.

Andrew B. Kahng
Department of Computer Science
University of California, Los Angeles
Los Angeles, CA 90024-1596
<abk@cs.ucla.edu>

Gabriel Robins
Department of Computer Science
University of Virginia
Charlottesville, VA 22903-2442
<robins@cs.virginia.edu>