Non-Tree Routing*

Bernard A. McCoy and Gabriel Robins

Department of Computer Science, University of Virginia, Charlottesville, VA 22903-2442

Appeared in: IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 14, No. 6, June 1995, pp. 780-784.

Abstract

An implicit premise of existing routing methods is that the routing topology must correspond to a tree (i.e., it does not contain cycles). In this paper we investigate the consequences of abandoning this basic axiom, and instead we allow routing topologies that correspond to arbitrary graphs (i.e., where cycles are allowed). We show that non-tree routing can significantly improve signal propagation delay, reduce signal skew, and afford increased reliability with respect to open faults that may be caused by manufacturing defects and electro-migration. Simulations on uniformly-distributed nets indicate that depending on net size and technology parameters, our non-tree routing construction reduces maximum sourse-sink SPICE delay by an average of up to 62%, and reduces signal skew by an average of up to 63%, as compared with Steiner routing. Moreover, up to 77% of the total wirelength in non-trees can tolerate an open fault without disconnecting the circuit.

1 Introduction

Recent advances in VLSI technology have steadily improved chip packing densities. As feature sizes decrease, device switching speeds tend to increase; however, thinner wires have higher resistance, causing signal propagation delay through the interconnect to increase [1]. Thus, interconnection delay has a greater impact on circuit speed, being responsible for up to 70% of the clock cycle in the design of dense, high-performance circuits [19]. In light of this trend, performance-driven routing has become central to the design of leading-edge digital systems [14].

Minimum spanning trees with bounded source-sink pathlengths were proposed in [9]. Boese et al. [5] have developed a "critical sink" routing approach which significantly reduces delay to specified sinks, thereby exploiting the critical-path information that is implicitly available during iterative timing-driven layout. More recently, Boese et al. [4] [3] have identified and exploited a high-quality, algorithmically tractable model of interconnect delay, based on an upper bound [18] for Elmore delay.

*Professor Gabriel Robins is supported by NSF Young Investigator Award MIP-9457412. Correspondence should be directed to Professor Gabriel Robins, Department of Computer Science, Thornton Hall, University of Virginia, Charlottesville, VA 22903, USA, Email: robins@cs.virginia.edu, URL http://uvacs.cs.virginia.edu/~robins/, phone: (804) 982-2207. Bernard McCoy is with GE Fanuc, Charlottesville, VA 22901.

An implicit premise of previous methods is that a routing topology must correspond to a tree (i.e., an acyclic topology). In retrospect, this assumption seems natural, since a tree topology spans a net and thus achieves electrical connectivity using a minimum number of edges/wires. In this paper, we question this seemingly basic axiom, and investigate the consequences of allowing cycles in the routing. Thus, we recast the routing problem into a new formulation where the interconnection topology may correspond to an arbitrary graph.

At this point, the reader may ask: how can adding extra wires to an existing routing tree improve signal propagation delay? The answer lies in the tradeoff between the overall circuit capacitance and the source-sink resistances. Clearly, adding extra wires to a routing tree increases the overall capacitance; however, the added wires may significantly lower certain source-sink resistance values, which may in turn reduce source-sink signal propagation delays. The key observation is that often the decrease in resistance more than compensates for the associated increase in capacitance, especially in sub-micron technologies; a simple example of this phenomenon is illustrated in Figure 1. Based on these observations, we propose a new non-tree routing scheme. Extensive SPICE simulations indicate that depending on net size and technology parameters, our non-tree routings reduce signal propagation delay by an average of up to 62% as compared to traditional Steiner routing.

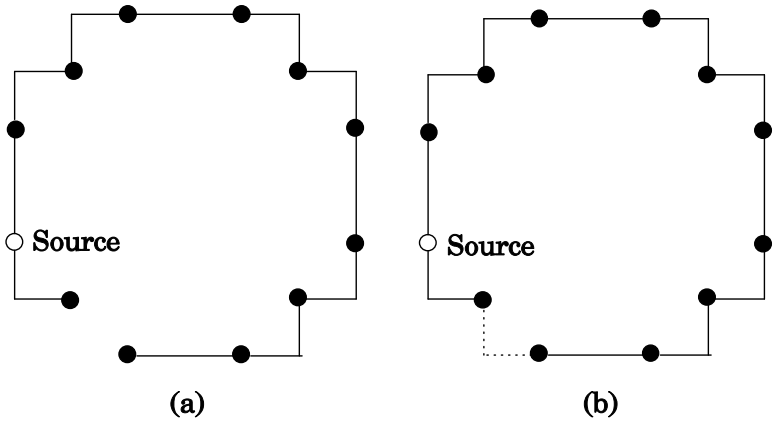

Figure 1: An example of how adding an extra edge to the minimum spanning tree on the left (a) can yield the routing topology with reduced interconnect delay on the right (b); in this example, routing topology (a) has maximum source-sink SPICE delay of 1.3ns, while the topology on the right has a SPICE delay of 1.0ns, a 23% improvement (at a total wirelength penalty of only 9%). The interconnect parameters used correspond to a MOSIS 0.8μ CMOS process.

With decreasing VLSI feature size, open faults in the interconnect (i.e., discontinuities in wires) tend to occur due to manufacturing defects and electro-migration, with both these problems becoming increasingly acute in the submicron regimes [11] [15]. Previous routing techniques do not address this issue of the

reliability of the routing, i.e., the ability of the interconnect to tolerate open faults. In contrast, our non-tree routing techniques can tolerate an open fault along the majority of the wires. Another benefit of non-tree routings is that they improve signal skew by an average of up to 63% over Steiner routing, as well as reduce signal reflection [6]. Finally, our basic approach is amenable to numerous extensions such as routing with critical sinks [5].

The rest of our paper is organized as follows. Section 2 gives basic definitions, formalizes the problem of constructing optimal-delay interconnection graph topologies, and discusses the delay models. In Section 3 we present a non-tree routing heuristic. Section 4 discusses the experimental results, and we conclude in Section 5. A preliminary version of this work has appeared in [16].

2 Problem Formulation

A *signal net* $N = \{n_0, n_1, ..., n_k\}$ is a fixed set of *pins* in the Manhattan plane to be connected by a *routing graph* $G = (N, E)$. Pin $n_0 \in N$ is a *source* (i.e., where the signal originates), and the remaining pins are *sinks* (i.e., where the signal propagates to). Each edge $e_{ij} \in E$ has an associated *edge cost*, d_{ij}, equal to the Manhattan distance between its two endpoints n_i and n_j; the *cost* (or total wirelength) of G is the sum of its edge costs. We use $t(n_i)$ to denote the signal propagation delay from the source to pin n_i. Our goal is to construct a routing topology which spans the net and which also minimizes the maximum source-sink delay:

Optimal Routing Graph (ORG) Problem: Given a signal net $N = \{n_0, n_1, ..., n_k\}$ with source n_0, find a set S of Steiner points and construct a routing graph $G = (N \cup S, E)$, $E \subseteq (N \cup S) \times (N \cup S)$, such that $t(G) = \max_{i=1}^{k} t(n_i)$ is minimized.

We note that the ORG problem is NP-complete by observing that if the R/C ratio is sufficiently small, the optimal solution to the ORG problem will be a *tree* with least wirelength – i.e., an optimal Steiner tree. The ORG formulation easily extends to address critical sinks, by associating a *criticality* $\alpha_i \geq 0$ with each sink n_i, reflecting timing information obtained during the performance-driven placement phase [4] [5]. The goal would then be to construct a routing which minimizes the weighted sum of the sink delays $\sum_{i=1}^{k} \alpha_i \cdot t(n_i)$.

The specific routing graph G that solves the ORG problem will depend on the model used to estimate the delay $t(G)$, as well as on the particular technology parameters. Ideally, we would like to compute and optimize delay according to the complete physical attributes of the circuit. To this end, we could use the circuit simulator SPICE [17], which is generally regarded as the best available tool for obtaining a precise, complete measure of interconnect delay.

Unfortunately, SPICE delay is too computationally prohibitive to evaluate during the routing phase of layout, and we are thus compelled to seek other alternatives. Another delay model is the Elmore delay formula [10], which was shown in [2] to have both high accuracy and fidelity in comparison with SPICE. The Elmore delay is defined as follows. Given routing tree T rooted at n_0, let e_i denote the edge from pin n_i to its parent. The resistance and capacitance of edge e_i are denoted by r_{e_i} and c_{e_i}, respectively. Let T_i denote the subtree of T rooted at n_i, and let c_i denote the sink capacitance of n_i. We use C_i to denote the *tree capacitance* of T_i, namely the sum of sink and edge capacitances in T_i. Using this notation, the Elmore delay along edge e_i is equal to $r_{e_i} \cdot (c_{e_i}/2 + C_i)$. Let r_d denote the output driver resistance at the net's source. Then the Elmore delay $t_{ED}(n_i)$ from source n_0 to sink n_i is given by: $t_{ED}(n_i) = r_d \cdot C_{n_0} + \sum_{e_j \in path(n_0, n_i)} r_{e_j} \cdot (c_{e_j}/2 + C_j)$.

We can extend the t_{ED} function to entire trees by defining $t_{ED}(T) = \max_{i=1}^{k} t_{ED}(n_i)$. Because of its relatively simple form, Elmore delay can be calculated in $O(k)$ time [18]. However, while the basic Elmore delay model outlined above applies only to tree topologies, Chan and Karplus have extended it to RC meshes [7]. Their method partitions the graph into a spanning tree and a set of m additional edges, then adds the extra edges back, updating the Elmore delay at each step. This increases the time complexity of the Elmore delay calculation to $O(k \cdot m)$. We use this method of delay calculation for general RC meshes in our approximation heuristic for the ORG problem (note that the work of [7] proposes a delay estimator but does not give a specific routing method).

Although the ORG problem formulation seeks to optimize delay, our solution below has additional benefits as well. For example, routings produced by our algorithm also have substantially reduced skew (i.e., maximum difference in signal propagation delay between any two terminals), which is important since a large signal skew diminishes the system performance [8]. Moreover, non-tree routings produced by our method afford circuit reliability in the sense that they are able to tolerate open faults due to manufacturing defects and electro-migration. This increase in reliability is quantified more precisely in Section 4.

3 Low Delay Routing Graph Heuristic

After testing a number of candidate heuristics for the ORG problem, the most effective method that we found is as follows: starting with a Steiner tree topology, we search for a new edge to add, so that the maximum source-sink delay in the resulting routing graph will be minimized. This edge is then added to the routing graph, and the process is iterated (i.e., we look for yet another good edge to add). We terminate when no further delay improvement is possible; thus, the maximum source-sink delay of the routing produced by our algorithm is guaranteed to be no worse than that of the initial routing (and

typically considerably better). Steiner points are allowed as junctures in the routing, in order to afford further opportunity for both delay and wirelength optimization, and we use the delay estimation method of [7] to guide our search inside the inner loop of this construction. An execution example of this method, called the Low Delay Routing Graph (LDRG) algorithm, is shown in Figure 2, while the formal statement of the algorithm is given in Figure 3.

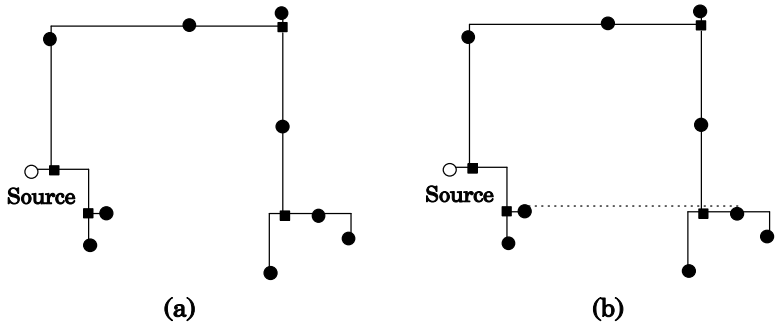

Figure 2: An execution of LDRG algorithm on a random 10-pin net. The Steiner tree shown on the left (a) has SPICE delay of 2.8ns (Steiner points are square), while the LDRG routing on the right (b) has SPICE delay of 1.9ns, a 32% improvement (the wirelength increase is 25%).

Low Delay Routing Graph (LDRG) Algorithm
Input: signal net N with source $n_0 \in N$ **Output:** low-delay routing graph $G = (\hat{N}, E)$
Compute a Steiner routing $G = (\hat{N}, E)$ over $\hat{N} = N \cup S$, where S are the possible Steiner points, and $E \subseteq \hat{N} \times \hat{N}$ is the set of Steiner tree edges **While** there is an edge $e_{ij} \in \hat{N} \times \hat{N}$ such that $t((\hat{N}, E \cup \{e_{ij}\})) < t(G)$ is minimized **Do** $G = (\hat{N}, E \cup \{e_{ij}\})$ **Output** resulting routing topology G

Figure 3: The Low Delay Routing Graph algorithm.

4 Experimental Results

We implemented the LDRG algorithm using C in the UNIX/Sun environment; code is available from the authors upon request. We ran trials on sets of 100 random nets for each of several net sizes, with pin locations uniformly distributed in a square layout region. Although internally the LDRG method uses the extension of the Elmore delay formula to graphs [7], for greater accuracy and realism, we used SPICE3e2 [17] to evaluate the performance of the actual routings produced by LDRG.

We used SPICE parameters that are representative of typical MOSIS 0.8μ, 1.2μ and 2.0μ CMOS IC processes, as well as a typical MCM technology (see Table 1). Our SPICE delay model assumes constant resistance and capacitance per unit length of interconnect (i.e., both resistance and capacitance are proportional to wirelength). In addition, sink loading capacitances were used at all the pins to model loads driven by the interconnect. In our LDRG implementation, the initial Steiner routing tree was computed using an efficient implementation of the Iterated 1-Steiner algorithm of Kahng and Robins [13], which is known to yield near-optimal Steiner trees [12].

Technology	IC1 2.0μ	IC2 1.2μ	IC3 0.8μ	MCM
driver resistance (Ω)	164	212	270	25
wire resistance ($\Omega/\mu m$)	0.033	0.073	0.112	0.008
wire capacitance ($fF/\mu m$)	0.019	0.022	0.039	0.06
sink loading capacitance (fF)	5.70	7.06	1.00	1000
layout area (mm^2)	10^2	10^2	10^2	100^2

Table 1: Technology parameters for three common CMOS IC processes and a typical MCM process.

Figure 4(a) shows the percent improvement in maximum source-sink delay over Steiner routing. Significant improvment is observed for example, in the IC3 (0.8μ CMOS) technology for 20-pin nets, where on average LDRG wins over Steiner routing by 27% while incurring a 14% wirelength penalty. Even larger improvment is seen in the MCM technology, with a performance improvement of 44% for 10-pin nets and of 62% for 20-pin nets. Note that the percent improvement in delay is consistently greater than the percent increase in wirelength across all IC technologies and net sizes. The detailed data is given in Table 2, and Table 3 shows how many extra edges are typically added to the initial topology by the LDRG algorithm.

We tallied the number of cases where LDRG was able to improve upon the initial Steiner routing (see Figure 4(b)). We observe that the number of improvable cases increases with the net size, and approaches 100% for 20-pin nets in all technologies; for MCM routing, LDRG seems superior to Steiner routing for all net sizes. Although the wirelength penalty may seem high at first glance, we note that only a small fraction of nets need to be routed as non-trees, i.e., only the nets that contain critical paths; other, non-critical nets may still be routed using traditional Steiner routing, and thus the *overall* wirelength penalty for the entire circuit when using non-tree routing is much smaller than that indicated by the data.

An additional benefit of non-tree routing is a significant reduction in signal skew (i.e., the maximum difference between signal arrival times at any two pins). Figure 4(d) shows the average percent improvment in signal skew over Steiner routing. For example, for 20-pin nets, LDRG yields 63% skew reduction for MCM and 43% skew reduction for IC3.

To quantify the increase in reliability afforded by non-tree routing, we measured the average percentage of the total wirelength that lies on cycles in the non-tree topology. This corresponds to the percentage of the total routing wirelength that can tolerate an open fault due to manufacturing defects or electro-migration, and is illustrated in Figure 4(e). For example, for 20-pin nets under the MCM technology, 77% of the wirelength can tolerate an open fault, and for IC3, 47% of the wirelength is fault-tolerant.

LDRG Algorithm Statistics										
	net	All Cases				Percent	Winners Only			
	size	Delay	Cost	Skew	Reliability	Winners	Delay	Cost	Skew	Reliability
IC1	5	0	0	0	0	0	0	0	0	0
	10	9	2	10	7	15	53	8	60	67
	20	15	5	25	20	45	32	9	54	49
IC2	5	0	0	0	0	0	0	0	0	0
	10	12	7	13	17	35	33	18	51	58
	20	40	17	40	45	100	40	17	40	45
IC3	5	8	7	9	36	35	22	38	8	44
	10	17	13	26	33	85	20	36	16	42
	20	27	14	43	47	100	27	43	14	47
MCM	5	38	92	40	87	100	38	92	40	87
	10	44	73	44	79	100	44	73	44	79
	20	62	50	63	77	100	62	50	63	77

Table 2: A detailed summary of the performance of LDRG. For each technology and net size, 100 random nets were generated using a uniform distribution; shown are the average percent improvements over Steiner routing for delay and skew. For cost, the average percent increase in wirelength is shown. For reliability, we give the percentage of total wirelength that lies on cycles in the routing topology, i.e., that can tolerate an open fault. "Percent winners" is the percent of cases where LDRG improved upon the initial Steiner routing, and the "winners only" statistics are averages over only the winners.

Number of Edges added by **LDRG**											
	net size	0	1	2	3	4	5	6	7	8	9
IC1	5	100									
	10	85	15								
	20	55	45								
IC2	5	100									
	10	65	35								
	20	0	75	25							
IC3	5	65	35								
	10	15	85								
	20	0	75	15	10						
MCM	5	0	80	20							
	10	0	5	15	25	55					
	20	0	30	10	5	5	10	5	10	10	15

Table 3: A summary of how many edges were added by the LDRG algorithm, shown as a percent of the total number of the 100 cases tested.

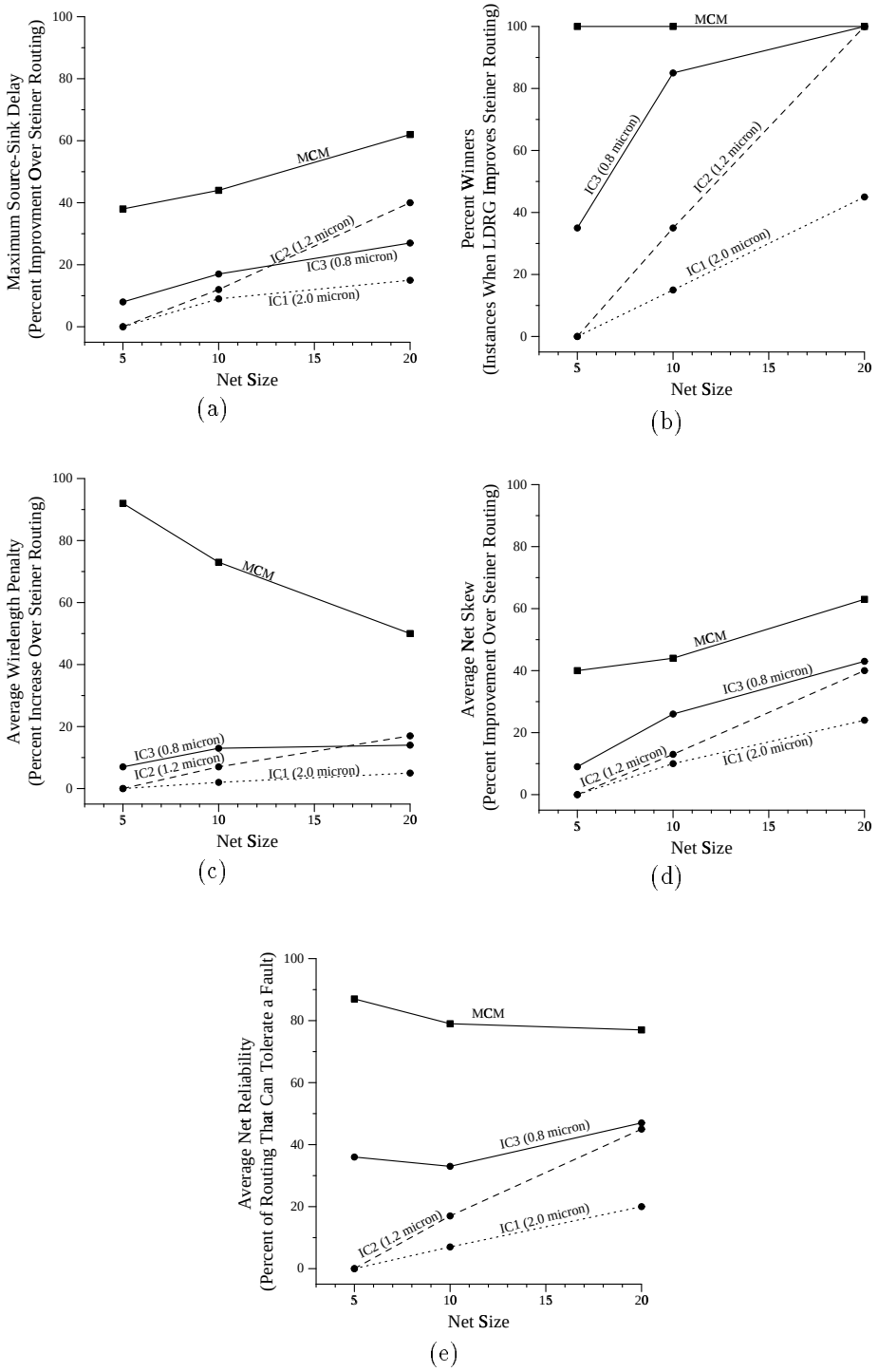

Figure 4: Averages for 100 uniformly distributed nets: (a) maximum source-sink delay improvement over Steiner routing; (b) percent of instances where LDRG improved upon the initial Steiner routing; (c) wirelength increase over Steiner routing; (d) signal skew improvement over Steiner routing; (e) routing reliability afforded by LDRG (i.e., average percent of the total wirelength per net that can tolerate an open fault).

5 Conclusions

We have explored the consequences of abandoning an implicit restriction common to previous routing formulations, namely the insistence on a strictly acyclic (tree) routing topology. Instead, we reformulated the routing problem as one of constructing a routing *graph* with low maximum source-sink delay. We have shown that adding a few extra wires to an initial routing tree can significantly improve signal propagation delay, by exploiting the tradeoff between circuit capacitance and path resistance. In particular, depending on net size and technology, non-tree routing can improve the average signal propagation delay by up to 62% over traditional Steiner routing. We have also shown that our non-tree routing technique also improves signal skew and the interconnect reliability, in the sense that non-tree routing can tolerate open faults due to manufacturing defects and electro-migration.

6 Acknowledgments

We thank Professors Pak Chan and Kevin Karplus for valuable discussions and for the use of their code in computing Elmore delay for general RC networks. Parasitics for the three IC technologies were provided by the MOSIS group, while the MCM interconnect parasitics are courtesy of Professor Wayne W.-M. Dai and the AT&T Microelectronics Division. Finally, we would like to thank the anonymous referees for their helpful comments. Additional information regarding our recent research and publications may be obtained on-line from URL http://uvacs.cs.virginia.edu/~robins/.

References

[1] H. Bakoglu, *Circuits, Interconnections and Packaging for VLSI*, Addison-Wesley, Reading, MA, 1990.

[2] K. D. Boese, A. B. Kahng, B. A. McCoy, and G. Robins, *Fidelity and Near-Optimality of Elmore-Based Routing Constructions*, in Proc. IEEE Intl. Conf. Computer Design, Cambridge, MA, October 1993, pp. 81–84.

[3] K. D. Boese, A. B. Kahng, B. A. McCoy, and G. Robins, *Rectilinear Steiner Trees with Minimum Elmore Delay*, in Proc. ACM/IEEE Design Automation Conf., San Diego, CA, June 1994, pp. 381–386.

[4] K. D. Boese, A. B. Kahng, B. A. McCoy, and G. Robins, *Near-Optimal Critical Sink Routing Tree Constructions*, IEEE Trans. Computer-Aided Design (to appear), (1995).

[5] K. D. Boese, A. B. Kahng, and G. Robins, *High-Performance Routing Trees With Identified Critical Sinks*, in Proc. ACM/IEEE Design Automation Conf., Dallas, June 1993, pp. 182–187.

[6] P. K. Chan, University of California, Santa Cruz, private communication, June 1993.

[7] P. K. Chan and K. Karplus, *Computing Signal Delay in General RC Networks by Tree/Link Partitioning*, IEEE Trans. Computer-Aided Design, 9 (1990), pp. 898–902.

[8] J. Cong, A. B. Kahng, and G. Robins, *Matching-Based Methods for High-Performance Clock Routing*, IEEE Trans. Computer-Aided Design, 12 (1993), pp. 1157–1169.

[9] J. Cong, A. B. Kahng, G. Robins, M. Sarrafzadeh, and C. K. Wong, *Provably Good Performance-Driven Global Routing*, IEEE Trans. Computer-Aided Design, 11 (1992), pp. 739–752.

[10] W. C. Elmore, *The Transient Response of Damped Linear Networks with Particular Regard to Wide-Band Amplifiers*, J. Appl. Phys., 19 (1948), pp. 55–63.

[11] R. L. Geiger, P. E. Allen, and N. R. Strader, *VLSI Design Techniques for Analog and Digital Circuits*, McGraw-Hill, New York, 1990.

[12] J. Griffith, G. Robins, J. S. Salowe, and T. Zhang, *Closing the Gap: Near-Optimal Steiner Trees in Polynomial Time*, IEEE Trans. Computer-Aided Design, 13 (1994), pp. 1351–1365.

[13] A. B. Kahng and G. Robins, *A New Class of Iterative Steiner Tree Heuristics With Good Performance*, IEEE Trans. Computer-Aided Design, 11 (1992), pp. 893–902.

[14] A. B. Kahng and G. Robins, *On Optimal Interconnections for VLSI Layout*, Kluwer Academic Publishers, Boston, MA, 1995.

[15] S. L. Long and S. E. Butner, *Gallium Arsenide Digital Integrated Circuits*, McGraw-Hill, New York, 1990.

[16] B. A. McCoy and G. Robins, *Non-Tree Routing*, in Proc. European Design and Test Conf., Paris, France, February 1994, pp. 430–434.

[17] L. Nagel, *SPICE2: A Computer Program to Simulate Semiconductor Circuits*, May 1975.

[18] J. Rubinstein, P. Penfield, and M. A. Horowitz, *Signal Delay in RC Tree Networks*, IEEE Trans. Computer-Aided Design, 2 (1983), pp. 202–211.

[19] S. Sutanthavibul and E. Shragowitz, *An Adaptive Timing-Driven Layout for High Speed VLSI*, in Proc. ACM/IEEE Design Automation Conf., 1990, pp. 90–95.