Dynamically-Wiresized Elmore-Based Routing Constructions*

Todd D. Hodes, Bernard A. McCoy, and Gabriel Robins

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

Abstract

We analyze the impact of wiresizing on the performance of Elmore-based routing constructions. Whereas previous wiresizing schemes are static *(i.e., they wiresize an existing topology), we introduce a new* dynamic *wiresizing technique, which uses wiresizing considerations to* drive *the routing construction itself. Simulations show that dynamic wiresizing affords superior performance over static wiresizing, and also avoids topological degeneracies. Moreover, dynamically-wiresized Elmore-based routing constructions significantly outperform all previous methods (including A-Trees) in term of maximum source-sink signal delay, affording up to 73% average SPICE delay improvement over traditional Steiner routing.*

1 Introduction

Due to the scaling of VLSI technology, interconnect delay has recently become a dominant concern in the design of complex, high-performance circuits [8] [19]. The typical goal of performance-driven routing is to minimize average or maximum source-sink delay. Much early work implicitly equated optimal routing with minimum-cost Steiner routing [9] [11] [12] [13] [15], but recently it became increasingly apparent that for leading-edge technologies, delay minimization and wirelength minimization are not synonymous [5].

A general tradeoff formulation was given in [6], where both the cost and radius of the routing construction are guaranteed to simultaneously be within *constant* factors of optimal. The cost-radius tradeoff may also be viewed as one between competing minimum spanning tree (MST) (or minimum-cost Steiner tree) and shortest-path tree (SPT) constructions [1]. Along similar lines, [7] have recently proposed the use of rectilinear Steiner arborescences [17] (or A-Trees), and use *wiresizing* to minimize signal delay.

There are two common shortcomings to previous high-performance routing methods: (1) their optimization criteria are primarily "geometric" in nature (as opposed to minimizing physical delay), and (2) they are "oblivious" to particular technology parameters (i.e., they produce the *same* routing construction for *different* values of wire resistance, capacitance, etc.). To overcome these flaws, Boese, Kahng and Robins [4] have recently developed a construction which greedily optimizes the Elmore delay formula *directly* to produce low-delay routing trees. Not only are these constructions adaptable to the prevailing technology parameters, but they were found to be near-optimal with respect to Elmore delay for a wide range of technology parameters [2]. Moreover, it was shown that Elmore delay has high fidelity to physical (SPICE-computed) delay over a range of IC technologies, i.e. near-optimal Elmore delay implies near-optimal SPICE delay [3].

In this paper, we analyze the impact of wiresizing on the performance of Elmore-based routing constructions. Whereas previous wiresizing schemes are *static* (i.e., they take as input a complete fixed routing topology and then try to find a good wiresizing for it), we introduce a new practical Elmore-based wiresizing technique that is *dynamic*, i.e. we use wiresizing considerations to *drive* the routing construction itself. Our simulations indicate that dynamic wiresizing affords superior performance over static wiresizing, and also avoids degenerate star-like topologies. Moreover, we show that dynamically-wiresized Elmore-based routing constructions outperform all previous methods, yielding up to 73% reduction in SPICE delay over traditional Steiner routing for MCM routing regimes.

2 Problem Formulation

Our overall goal is as follows: given an arbitrary set of pins with a designated source, we wish to electrically connect all the pins so that the maximum source-sink signal propagation delay is minimized. Ideally, a routing algorithm will compute and optimize signal delays according to a detailed circuit simulation, such as that provided by SPICE [14]. However, the computation times required by SPICE are prohibitive for routing tree construction, and therefore more efficient delay estimators are needed. As recently shown

*The corresponding author is Professor Gabriel Robins, Department of Computer Science, Thornton Hall, University of Virginia, Charlottesville, VA 22903-2442, U.S.A., Email: robins@cs.virginia.edu, phone: (804) 982-2207.

by Boese, Kahng, McCoy, and Robins [2], both the *fidelity* and *accuracy* of Elmore's distributed RC delay approximation are surprisingly high with respect to more complex delay estimators, such as the "Two-Pole" distributed RCL simulator of [20], as well as the SPICE circuit simulator [14]. We therefore use the Elmore formula to guide our routing constructions.

We begin with some definitions and notation. 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 tree* $T = (N, E)$, where $E \subseteq N \times N$. 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* of T 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 which spans the net and which minimizes the maximum source-sink delay.

While it is known that delay in a routing tree is a non-linear phenomenon [10], many previous methods for routing tree construction have either implicitly or explicitly assumed that delay is proportional to source-sink pathlength. Thus, such methods only attempt to *heuristically* capture the goal of "high performance," and it is therefore not surprising that when trees produced by these methods are tested using SPICE, their performance often proves disappointing. Thus, we strive to directly optimize a more realistic delay measure, such as the Elmore delay [10] [18].

Given a routing tree T rooted at n_0, let e_i denote the edge from 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 $r_{e_i}(c_{e_i}/2 + C_i)$. Let r_d denote the output driver resistance at the net's source. The Elmore delay $t_{ED}(n_i)$ at sink n_i is:

$$t_{ED}(n_i) = r_d C_{n_0} \quad + \sum_{e_j \in path(n_0, n_i)} r_{e_j}(c_{e_j}/2 + C_j)$$

Elmore delay has a compact definition and can be quickly evaluated at *all* sinks in $O(k)$ time [18], which enables an efficient implementation.

Wiresizing (i.e., increasing the widths of certain wires) can improve signal propagation delay by trading-off capacitance for resistance: when a wire width is increased, additional capacitance is induced, but some overall source-sink resistances may decrease. The goal of wiresizing is to find wire segments in the routing where an increase in capacitance is more than compensated for by the corresponding drop in resistances, thus improving overall signal delay. Given a fixed tree T, let $w(e_i)$ denote the width assignment of edge e_i and for simplicity we let $w(e_i)$ range over a discreet set of values $\{w_1, w_2, ..., w_k\}$. We can now define our problem as:

Optimal Wiresized Routing Problem: Given a signal net $N = \{n_1, n_2, ..., n_k\}$ with source n_0 and a set of widths $W = \{w_0, w_1, ..., w_j\}$, $w_0 < w_1 < \cdots < w_j$, find a set of points S and construct a routing tree $T = (N \cup S, E)$, $E \subseteq (N \cup S) \times (N \cup S)$, with each $e \in E$ having weight $w(e) \in W$, such that $t(T) = \max_{i=1}^{k} t(n_i)$ is minimized.

3 Dynamic Wiresizing

Given a fixed topology, the greedy wiresizing scheme of [7] recursively wiresizes each subtree; as long as overall maximum tree delay improvement is possible, each edge connecting the root to a subtree is widened. This *static greedy wiresizing* (SGW) scheme is formalized in Figure 1; it generalizes the greedy wiresizing scheme of [7], in that it allows for an arbitrary delay calculation to be used. Note that this method is *static*, meaning that the interconnect topology is determined *before* wiresizing commences and is not allowed to change during the wiresizing process.

Static Greedy Wiresizing (SGW) Algorithm
Input: Tree $T = (V, E)$ with source $n_0 \in N$ and a set W of edge widths **Output:** Wiresized tree spanning N

For each node $n_i \in V$ such that $e = (n_0, n_i) \in E$ **Do**
Call SGW on the subtree routed at n_i
Repeat
delay$_{old} = t(T)$
Increase w_r to w_{r+1} of edge e
Until delay$_{old} < t(T)$
Decrease w_r to w_{r-1} of edge e

Figure 1: A static greedy wiresizing algorithm.

While static greedy wiresizing provides a near-optimal wiresizing for a *given* topology [7], such a wiresizing process is largely constrained by that fixed input topology. Ideally we would like to compute the *optimal* combination of routing topology *and* wiresizing; unfortunately, this is not computationally feasible. On the other hand, we do not want to completely dissociate the topology construction from the wiresizing issues (as was done in [7]), since such a strategy will not benefit from a possible synergy between these.

With this in mind, we give a *dynamic* wiresizing algorithm that hybridizes the routing topology construction with the wiresizing process. Our new construction combines the Elmore routing tree method of [4] with the greedy wiresizing method above. Following [4], our method is analogous to Prim's minimum spanning tree construction [16]: starting with a degenerate tree initially consisting of only the source pin, we grow the tree at each step by finding a new pin to connect to the tree, as to minimize the Elmore delay in the *wiresized* current topology. In other words, in each step we invoke the SGW routine once for each candidate edge and add the edge that yields the best wiresized tree. The algorithm terminates when the construction spans the entire net.

Note that during the execution, a partial topology is not *actually* wiresized, but instead its edges are left having the minimum width; rather, wiresizing considerations are used as a *guide* to drive the edge-selection process. When the topology spans all the net pins, we invoke the static wiresizing algorithm one final time and return the resulting *wiresized* tree. The algorithm, which we call DWSERT, is formalized in Figure 2.

Dynamically Wiresized Steiner Elmore Routing Tree (DWSERT) Algorithm

Input: Signal net N with source $n_0 \in N$
Output: Wiresized low-delay Steiner tree spanning N

$T = (V, E) = (\{n_0\}, \emptyset)$
$M = N - \{n_0\}$
While $M \neq \emptyset$ do
    Find $u \in M$, and a point w on some edge of E which minimizes the maximum Elmore delay from n_0 to any leaf in the *wiresized* tree SGW$(V \cup \{u, w\}, E \cup \{(u, w)\})$
    $V = V \cup \{u, w\}$
    $E = E \cup \{(u, w)\}$
    $M = M - \{u\}$
Output SGW$(T = (V, E))$

Figure 2: Algorithm DWSERT: constructing a *dynamically* wiresized low-delay routing.

4 Experimental Results

We have implemented the dynamically wiresized DWSERT method using C in the UNIX Sun environment. We have compared it to: (i) the best-performing Iterated 1-Steiner (I1S) construction of Kahng and Robins [12]; (ii) the SERT construction of [4] (which is equivalent to DWSERT without the wiresizing); and (iii) the arborescence-based A-Tree method of [7], as well as the statically wiresized version of these (WI1S, WSERT, and WA-Tree, respectively). We tested these algorithms on 50 random nets of up to 25 pins, uniformly distributed in the $100000\mu \times 100000\mu$ grid, with the source being one of the pins chosen at random. Our technology parameters correspond to a typical multichip module interconnect parameters, and were provided by the AT&T Microelectronics Division.

Table 1 gives the average percent SPICE delay improvement in maximum source-sink delay relative to the corresponding I1S values. In other words, each entry in the table represents the average percent improvement in maximum delay as compared to the maximum delay for the I1S routing over the same net. We see that SERT substantially beats A-Tree for all net sizes. Moreover, static wiresizing dramatically improves delay when applied to either an I1S tree or an A-Tree.

Only little improvement occurs when SERT is statically wiresized. This is because near-optimal SERT topologies tend to be star-like (i.e., most sinks being directly connected to the source; thus the lower resistance of a wider edge does not compensate for the added capacitance). On the other hand, DWSERT does not yield degenerate topologies. DWSERT improves over WA-Tree for most net sizes, and is thus the winner among the various methods. Figure 3 depicts the wiresized I1S, A-Tree, and SERT constructions for the same random net.

Average max source-sink SPICE Delay					
net size =	5	10	15	20	25
I1S	0.0	0.0	0.0	0.0	0.0
WI1S	28.6	34.4	38.2	36.8	38.4
A-Tree	7.4	23.4	27.7	43.8	43.7
WA-Tree	38.2	53.6	56.0	67.5	68.3
SERT	27.2	52.2	61.9	63.9	67.9
WSERT	31.6	54.4	63.7	65.4	69.8
DWSERT	32.2	56.2	65.9	68.4	72.7

Table 1: SPICE simulation results comparing the I1S, SERT, and A-Tree constructions, as well as their wiresized versions. Each entry corresponds to an average percent improvement over I1S.

5 Conclusions

We have analyzed the impact of wiresizing on the performance of Elmore-based routing constructions. Whereas previous wiresizing schemes are *static* (i.e., they wiresize a fixed existing topology), we introduced a new *dynamic* Elmore-based wiresizing technique, using wiresizing considerations to *drive* the routing construction itself. SPICE simulations indicate that dynamic wiresizing affords improved performance over static wiresizing, and yield non-degenerate topologies. Moreover, dynamically-wiresized Elmore-based constructions seem to significantly outperform all previous methods (including A-Trees) in term of maximum source-sink SPICE delay, affording up to 73% average delay improvement over traditional Steiner routing.

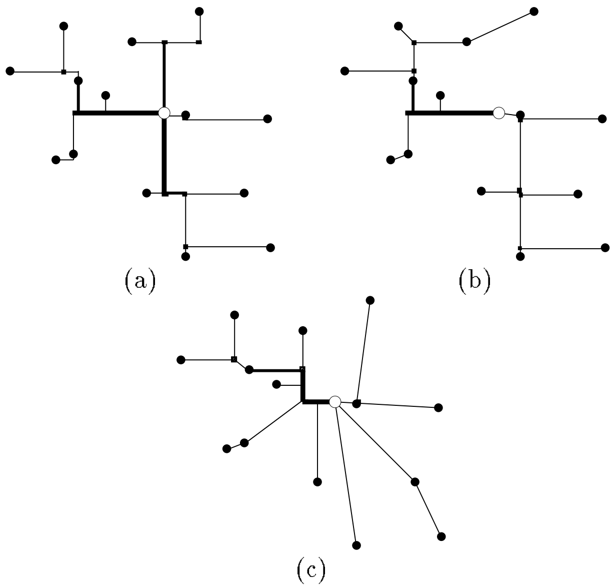

Figure 3: A comparison of the different constructions for a random 15-pin net (hollow dot is the source pin): (a) the statically Wiresized A-Tree has maximum source-sink delay of 3.00ns (the non-wiresized A-Tree has a delay of 4.05ns; (b) the (statically) wiresized I1S tree has delay of 4.05 ns (the non-wiresized I1S tree has delay of 6.05ns); (c) the dynamically Wiresized SERT has a delay of 2.55ns, a 15% improvement over statically wiresized A-Tree.

References

[1] C. J. Alpert, T. C. Hu, J. H. Huang, and A. B. Kahng, *A Direct Combination of the Prim and Dijkstra Constructions for Improved Performance-Driven Global Routing*, Tech. Rep. CSD-TR-920051, Computer Science Department, UCLA, 1992.

[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, *Towards Optimal Routing Trees*, in Proc. ACM/SIGDA Physical Design Workshop, Lake Arrowhead, CA, April 1993, pp. 44–51.

[4] 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.

[5] J. Cohoon and J. Randall, *Critical Net Routing*, in Proc. IEEE Intl. Conf. Computer Design, Cambridge, MA, October 1991, pp. 174–177.

[6] 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.

[7] J. Cong, K. S. Leung, and D. Zhou, *Performance-Driven Interconnect Design Based on Distributed RC Delay Model*, in Proc. ACM/IEEE Design Automation Conf., Dallas, June 1993, pp. 606–611.

[8] W. E. Donath, R. J. Norman, B. K. Agrawal, S. E. Bello, S. Y. Han, J. M. Kurtzberg, P. Lowy, and R. I. McMillan, *Timing Driven Placement Using Complete Path Delays*, in Proc. ACM/IEEE Design Automation Conf., 1990, pp. 84–89.

[9] A. E. Dunlop, V. D. Agrawal, D. Deutsch, M. F. Jukl, P. Kozak, and M. Wiesel, *Chip Layout Optimization Using Critical Path Weighting*, in Proc. ACM/IEEE Design Automation Conf., 1984, pp. 133–136.

[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] M. A. B. Jackson, E. S. Kuh, and M. Marek-Sadowska, *Timing-Driven Routing for Building Block Layout*, in Proc. IEEE Intl. Symp. Circuits and Systems, 1987, pp. 518–519.

[12] 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.

[13] A. B. Kahng and G. Robins, *On Performance Bounds for a Class of Rectilinear Steiner Tree Heuristics in Arbitrary Dimension*, IEEE Trans. Computer-Aided Design, 11 (1992), pp. 1462–1465.

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

[15] S. Prasitjutrakul and W. J. Kubitz, *A Timing-Driven Global Router for Custom Chip Design*, in Proc. IEEE Intl. Conf. Computer-Aided Design, Santa Clara, CA, November 1990, pp. 48–51.

[16] A. Prim, *Shortest Connecting Networks and Some Generalizations*, Bell Syst. Tech J., 36 (1957), pp. 1389–1401.

[17] S. K. Rao, P. Sadayappan, F. K. Hwang, and P. W. Shor, *The Rectilinear Steiner Arborescence Problem*, Algorithmica, (1992), pp. 277–288.

[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.

[20] D. Zhou, S. Su, F. Tsui, D. S. Gao, and J. Cong, *Analysis of Trees of Transmission Lines*, Tech. Rep. CSD-TR-920010, Computer Science Department, UCLA, 1992.