

- [18] B. Rohfleisch, B. Wurth, and K. Antreich, "Logic clause analysis for delay optimization," in *Proc. Design Automation Conf.*, June 1995, pp. 668–672.
- [19] M. Schulz and E. Auth, "Advanced automatic test pattern generation and redundancy identification techniques," in *Proc. Fault Tolerant Computing Symp.*, June 1988, pp. 30–34.
- [20] G. Stenz, B. M. Riess, B. Rohfleisch, and F. M. Johannes, "Timing driven placement in interaction with netlist transformations," in *Proc. Int. Symp. Physical Design*, Apr. 1997, pp. 36–41.
- [21] M. Yuguchi, Y. Nakamura, K. Wakabayashi, and T. Fujita, "Multi-Level Minimization based on Multi-Signal Implications," in *Proc. Design Automation Conf.*, June 1995, pp. 658–662.

## Interconnect Sizing and Spacing with Consideration of Coupling Capacitance

Jason Cong, Lei He, Cheng-Kok Koh, and Zhigang (David) Pan

**Abstract**—This paper studies interconnect sizing and spacing (ISS) problem with consideration of coupling capacitance for performance optimization of single or multiple critical nets. We introduce the formulation of symmetric and asymmetric wire sizing. We develop efficient bound computation algorithms for ISS optimization and prove their optimality under general interconnect resistance and capacitance models. Our experiments show that our algorithms are very effective and obtain significant performance improvement compared to previous wire-sizing/spacing algorithms.

**Index Terms**—Coupling capacitance, interconnect, wire sizing, wire spacing.

### I. INTRODUCTION

It has been widely recognized that for deep submicrometer (DSM) very large scale integration designs, interconnect plays a dominating role in determining the overall circuit performance [1]. Among various interconnect optimization techniques, wire sizing and spacing are effective techniques to determine proper width/spacing for one or multiple nets such that certain design objective is optimized.

It was first shown in [2] and [3] that when wire resistance becomes significant, as in DSM designs, proper wire sizing can effectively reduce the interconnect delay. Assuming that each wire segment has a set of discrete widths, their work presented the first optimal wire-sizing (OWS) algorithm for a single-source resistance–capacitance (RC) interconnect tree using a local refinement (LR) operation. It was later extended for a routing tree with multiple sources [4] and for the maximum delay objective using Lagrangian relaxation [5]. Another formulation of wire-sizing optimization is to determine the continuous wire

Manuscript received November 23, 1999; revised January 27, 2001. This paper was recommended by Associate Editor M. Pedram.

J. Cong is with the Computer Science Department, University of California, Los Angeles, CA 90095 USA.

L. He is with the Department of Electrical and Computer Engineering, University of Wisconsin, Madison, WI 53706 USA.

C.-K. Koh is with the Department of Electrical and Computer Engineering, Purdue University, West Lafayette, Indiana 47907 USA.

Z. Pan is with the IBM T.J. Watson Research Center, Yorktown Heights, NY 10598 USA.

This work is supported in part by the National Science Foundation Young Investigator Award MIP-9357582 and by the Intel Corporation.

Publisher Item Identifier S 0278-0070(01)06888-9.

shaping functions. The closed-form wire shaping functions were derived to minimize the Elmore delay, first without fringing capacitance [6] and later with fringing capacitance [7] and for a bidirectional wire [8]. Wire-sizing optimization is also studied using high-order delay model (e.g., [9] and [10]).

However, most of these works on wire sizing only sized a *single* net and did not explicitly consider the coupling capacitance, which has already become the dominating capacitance component. Moreover, the optimality of most algorithms assumed rather simple interconnect capacitance models (e.g., *constant* unit area and fringing capacitances), which no longer hold for DSM designs. In this paper, we study the interconnect sizing and spacing (ISS) problem with *explicit* consideration of coupling capacitance and under more general (e.g., table look-up) capacitance models. Our optimization target can be either a *single* critical net [single-net interconnect sizing and spacing (SISS)], or *global* optimization of multiple critical nets [global interconnect sizing and spacing (GISS)]. The major contribution of this paper includes the following.

- 1) We introduce the symmetric and asymmetric wire-sizing formulation. Since a net usually has asymmetric neighborhood structures, the asymmetric wire-sizing formulation provides more flexibility.
- 2) We extend the LR operation first introduced in [3] and show its optimality to bound an optimal SISS solution using the *dominance property for SISS*, under more general resistance and capacitance models than [3].
- 3) We introduce two bound-refinement (BR) operations and prove their *global optimality* to bound an optimal GISS solution using the *dominance property for GISS* (again, under general resistance and capacitance models).
- 4) We extend the dynamic programming (DP) algorithms [11]–[13] for SISS and GISS.

The rest of this paper is organized as follows. Section II formulates the problem. Sections III and IV present the properties and algorithms for SISS and GISS optimizations. Experimental results are shown in Section V, followed by the conclusion in Section VI. Proofs of theorems, together with validation of capacitance models that we assume for the optimality of our algorithms, are available in a technical report [14].

### II. PROBLEM FORMULATION

#### A. Symmetric and Asymmetric Wire Sizing

The purpose of this study is to perform postlayout (after global or detailed routing) ISS optimization to one or multiple timing-critical nets, based on some initial layout or topology of these nets. For each net  $\mathcal{N}_i$ , it consists of  $n_i + 1$  terminals  $\{s_0, \dots, s_{n_i}\}$  connected by a routing tree, denoted  $T_i$ . The terminal  $s_0^i$  denotes the source of  $\mathcal{N}_i$ . The rest of the terminals are sinks. The terminals (source and sinks) of  $T_i$  are at fixed locations, and  $T_i$  consists of  $m_i$  wire segments<sup>1</sup>  $\{E_1, \dots, E_{m_i}\}$ .

For a wire segment  $E$ , we define its *center line* to be a line that divides the initial layout evenly along the signal direction. For example, in Fig. 1(a), two horizontal wire segments  $E_1$  and  $E_2$  are shown with their center lines. We assume that the center line for each wire segment is fixed during wire sizing and spacing optimization. We call the

<sup>1</sup>A wire segment is the smallest unit for wire sizing. For delay minimization purpose, a long wire may be divided into a number of smaller wire segments. For ease of presentation, we assume that each wire segment has unit length [3]. Note that in this paper, a wire segment may also be referred to as an "edge"  $E$ , using graph terminology.

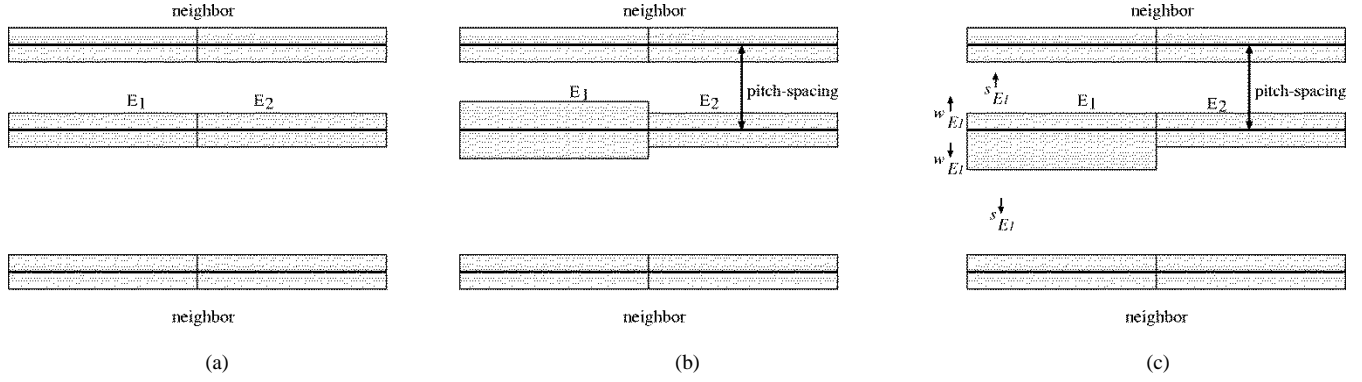


Fig. 1. (a) Wire segments with center lines. (b) Symmetric wire sizing. (c) Asymmetric wire sizing.

distance between adjacent center lines as the pitch spacing (PS). All previous works implicitly assumed *symmetric wire sizing* around the center lines, so each wire segment needs only one width to describe it. An example of symmetric wire sizing of the two wire segments  $E_1$  and  $E_2$  with a neighboring net is shown in Fig. 1(b).

However, symmetric wire sizing may be too restrictive when we take coupling capacitance into account because it is likely that the neighborhood structure of a net is *not* symmetric. We propose the *asymmetric wire sizing* that allows a wire to be sized asymmetrically with respect to its center line. Using the same example as in Fig. 1(b), we may want  $E_1$  to be farther away from its upper neighbor since the coupling capacitance would be less. As a result, we may only size up the bottom half piece and keep the top half intact, as shown in Fig. 1(c). In this case, each wire segment needs two widths to describe it. Let  $w_E$  denote the width of the wire segment  $E$ ,  $w_E^\downarrow$  ( $w_E^\uparrow$ ) be wire width below (above) the center line of  $E$ , and  $s_E^\downarrow$  ( $s_E^\uparrow$ ) be the edge-to-edge spacing from  $E$  to the neighbor wire segment below (above) it. Then,  $w_E = w_E^\downarrow + w_E^\uparrow$ . An asymmetric wire-sizing solution is valid if  $w_E^\downarrow \geq W_{\min}/2$  and  $w_E^\uparrow \geq W_{\min}/2$ . Note that for symmetric wire sizing,  $w_E^\downarrow = w_E^\uparrow = w_E/2$ . To avoid introducing additional notations, we also use  $w_{E^\downarrow}$  ( $w_{E^\uparrow}$ ) to denote the wire width for the left (right) part of a vertical wire segment and  $s_{E^\downarrow}$  ( $s_{E^\uparrow}$ ) to denote the spacing from  $E$  to its left (right) neighboring wire, respectively. In this paper, we assume that symmetric and asymmetric wire sizings have negligible difference in terms of wire resistance, if both have the same wire width for each corresponding wire segment [e.g., see Fig. 1(b) and (c)]. It is valid to the first order because for wire sizing to take effect, a wire needs to be sufficiently long (e.g., more than 5000 of the minimum wire width [1]). Then, the small resistance difference between symmetric and asymmetric wire sizing at the connections of different wire widths [e.g., see  $E_1$  and  $E_2$  in Fig. 1(b) and (c)] is negligible compared to the total wire resistance.

## B. SISS

Given an initial layout of  $n$  nets, the SISS problem is to find a symmetric or asymmetric wire-sizing solution  $\mathcal{W}$  for a single net  $\mathcal{N}$  (usually the most critical net) with the corresponding routing tree  $T$  to minimize the following delay objective:

$$t_T(\mathcal{W}) = \sum_{s_k \in \text{sink}(T)} \lambda_k \cdot t_T(s_k, \mathcal{W}) \quad (1)$$

where  $\lambda_k$  is the criticality of sink  $s_k$  and  $t_T(s_k, \mathcal{W})$  is the delay from source  $s_0$  to sink  $s_k$  with wire-sizing solution  $\mathcal{W}$ . Note that although only routing tree  $T$  appears in the above delay notation, other nets provide neighborhood structures and constraints thus will affect the value of  $t_T(\mathcal{W})$  as well.

We model the driver of the routing tree  $T$  by an effective resistance  $R_d$  and the routing tree itself by a distributed RC circuit and use the Elmore delay model [15] to guide performance optimization. For a wire segment  $E$ , let  $r_E$  be its resistance and  $c_E$  be its lumped capacitance, i.e.,  $c_E = c_a(E) \cdot w_E + c_f(E) + c_x(E)$ , where the  $c_a(E)$ ,  $c_f(E)$  and  $c_x(E)$  are the unit area, fringing, and coupling capacitances for  $E$ , respectively.<sup>2</sup> Let  $\text{Des}(E)$  be the set of descendant wire segments in the subtree rooted at  $E$  (excluding  $E$ ),  $\text{sink}(T)$  be the set of sinks in the routing tree  $T$ ,  $\text{sink}(E)$  be the set of sinks in the subtree rooted at  $E$ ,  $P_T(u, v)$  be the unique path from nodes  $u$  to  $v$  in  $T$ , and  $C_{\text{down}}(E)$  be the *total* downstream capacitance in the subtree rooted at  $E$  (including both wire and sink capacitances), i.e.,

$$C_{\text{down}}(E) = \sum_{s \in \text{sink}(E)} c_s + \sum_{E' \in \text{Des}(E)} c_{E'}.$$

Then, the Elmore delay from source  $s_0$  to sink  $s_k$  can be written as

$$t_T(s_k, \mathcal{W}) = R_d \cdot \left( \sum_{E \in T} c_E + \sum_{s \in \text{sink}(T)} c_s \right) + \sum_{E \in P_T(s_0, s_k)} r_E \cdot \left( \frac{c_E}{2} + C_{\text{down}}(E) \right). \quad (2)$$

Let  $\lambda(s_0) = \sum_{s_k \in \text{sink}(T)} \lambda_k$  and  $\lambda(E) = \sum_{s_k \in \text{sink}(E)} \lambda_k$ , (1) can be rewritten as

$$t_T(\mathcal{W}) = \lambda(s_0) \cdot R_d \cdot \left( \sum_{E \in T} c_E + \sum_{s \in \text{sink}(T)} c_s \right) + \sum_{E \in T} \lambda(E) \cdot r_E \cdot \left\{ \frac{c_E}{2} + C_{\text{down}}(E) \right\}. \quad (3)$$

If we treat  $\lambda(s_0)R_d$  as the new effective driver resistance and  $\lambda(E)r_E$  as the new effective wire resistance of  $E$ , (3) will be exactly of the same form of (2).

In this paper, the objective function is to minimize the weighted delay summation of all sinks, similar to [3]. It was shown by [5] and [16] that the weighted formulation can be used to solve other optimization problems such as minimizing the maximum delay or minimizing total area subject to delay constraint for each sink, using the Lagrangian relaxation technique. It relaxes all constraints using non-negative Lagrange multipliers with each Lagrange multiplier corresponding to a constraint. Finding the optimal set of Lagrange multipliers usually needs iterative adjustment, e.g., by the subgradient optimization method [16]. These Lagrange multipliers correspond to the

<sup>2</sup>For different wire segments, the unit area, fringing, and coupling capacitances may be different, so we write them as functions of  $E$  and they can be obtained by table look-up.

weights under our weighted formulation.<sup>3</sup> Then, under a given set of Lagrange multipliers (i.e., weights), our optimization algorithms (SISS and GISS) can be used to get the optimal or near-optimal solutions. Their role is the same as the greedy wire-sizing algorithm during the Lagrangian relaxation in [5], [16]. It shall be noted that the optimality of Lagrangian relaxation in [5] and [16] assumes *continuous* wire sizing, while our optimization algorithms work on *discrete wire sizing*, so the optimality may not be guaranteed.

### C. GISS

The GISS problem is to find a symmetric or asymmetric wire-sizing solution  $\mathcal{W}$  for all  $n$  critical nets under optimization so that the weighted delay of them

$$t(\mathcal{W}) = \sum_{i=1}^n \delta_i \cdot t_{T_i}(\mathcal{W}) \quad (4)$$

is minimized, where  $n$  is the number of critical nets to be simultaneously optimized and  $\delta_i$  is the criticality of net  $\mathcal{N}_i$ . Note that the net criticality  $\delta_i$  can be absorbed into each sink weight of net  $\mathcal{N}_i$ .

## III. ALGORITHM FOR SISS OPTIMIZATION

In this section, we study ISS optimization for a single critical net, i.e., the wire widths of all other nets are fixed. We first present the *dominance property* for an optimal SISS solution, then give an effective algorithm for it.

### A. Dominance Property for SISS

In [3], the LR operation was first introduced, which sizes *one* wire segment optimally at a time. Based on the LR, an elegant dominance property was obtained for OWS under simple constant interconnect resistance and capacitance models. We prove that the dominance property can be extended for SISS and under much more general resistance and capacitance models as follows.

*Definition 1—Monotone Resistance Model:* The wire resistance monotonically decreases as wire width increases, i.e.,  $r_E(w_E) < r_E(w'_E)$  if  $w_E > w'_E$ .

*Definition 2—Monotone Lumped Capacitance Model:* The lumped capacitance for a wire segment  $E$  monotonically increases as wire width increases, i.e.,  $c_E(w'_E) > c_E(w_E)$  if  $w'_E > w_E$ , given fixed neighboring structures.

Our study shows that the monotone resistance model and monotone lumped capacitance model always hold [14]. Then, we have the following theorem.

*Theorem 1—Dominance Property for SISS:* Let  $\mathcal{W}^*$  be an optimal SISS solution for a routing tree  $T$ . Under the monotone resistance model and the monotone lumped capacitance model, if a wire-sizing solution  $\mathcal{W}'$  dominates  $\mathcal{W}^*$ , then an LR of  $\mathcal{W}'$  will still dominate  $\mathcal{W}^*$ ; similarly, if  $\mathcal{W}'$  is dominated by  $\mathcal{W}^*$ , an LR of  $\mathcal{W}'$  will still be dominated by  $\mathcal{W}^*$ .

### B. Algorithm for SISS Optimization

1) *Bound Computation for SISS:* The dominance property suggest that one can start from some upper or lower bound of the optimal SISS solution and iteratively update these bounds toward the final optimal SISS solution using LR. The overall flow of bound computation for SISS is similar to that for OWS in [3]. Let  $\mathcal{W}^*$  be an optimal SISS solution and  $c_E(w_E^*, s_E^{\uparrow*}, s_E^{\downarrow*})$  be the lumped capacitance for an edge  $E$  based on the optimal wire sizing and spacing  $\mathcal{W}^*$ ,  $S^{\downarrow*}$ , and  $S^{\uparrow*}$ . Now, consider an upper bound  $\mathcal{W}^U$  of  $\mathcal{W}^*$ . From  $\mathcal{W}^U$ , we can obtain the

corresponding spacing lower bounds  $S^{\downarrow L}$  and  $S^{\uparrow L}$  for  $S^{\downarrow*}$  and  $S^{\uparrow*}$ , respectively. Under the monotone lumped capacitance model, we have  $c_E(w_E^U, s_E^{\downarrow L}, s_E^{\uparrow L}) \geq c_E(w_E^*, s_E^{\downarrow*}, s_E^{\uparrow*})$ . According to the dominance property, an LR operation to any wire segment of  $\mathcal{W}^U$  will still be an upper bound of  $\mathcal{W}^*$ . Similarly, the LR operation of an initial lower bound  $\mathcal{W}^L$  will still be a lower bound of  $\mathcal{W}^*$ .

Note that for the symmetric case, each wire segment just has one width, while for the asymmetric case, it has two widths: one for each side of its center line. The wire spacing is determined once the wire width is given because we assume the PS is fixed during the wire-sizing/spacing optimization. In this paper, we use an accurate capacitance lookup table, not a simple capacitance formula (constant  $c_a$  and  $c_f$ ) as in [3]. Thus, we need to enumerate the given discrete wire widths to get the local optimal symmetric or asymmetric wire-sizing solution.

2) *DP for SISS:* As we shall see in Section V, the bound-computation algorithm for SISS usually leads to exactly the the same lower and upper bound for each wire segment. When the lower and upper bounds do not meet, we use the bottom-up DP algorithm similar to [11] and [12] to compute the final SISS solution between the lower and upper bounds. The main idea for the DP is to compute and merge candidate wire-sizing solutions in a bottom-up manner starting from all sinks. During the merging phase, inferior (or called the *redundant* as in [11]), options are pruned out.<sup>4</sup> At the end of the bottom-up computation, the source may have a set of irredundant options. The optimal solution is picked according to the design objective. Tracing back from the source, we have the corresponding optimal wire-sizing and spacing solution.

Compared to previous DP algorithms in [11] and [12], our DP approach has the following features: 1) we consider the coupling capacitance between neighboring wires and 2) we keep two wire widths ( $w_E^{\downarrow}$  and  $w_E^{\uparrow}$ ) for each edge in the asymmetric scenario while performing bottom-up accumulation and top-down pruning.

## IV. ALGORITHM FOR GISS OPTIMIZATION

The SISS algorithm is for optimizing a single critical net. However, it largely depends on the previous layout of other neighboring nets. During layout optimization, there may have multiple critical nets that share limited routing resources. Thus, just optimizing one net may sacrifice the performance of other critical nets. In this section, we will study the GISS optimization for multiple nets. It is a much more difficult problem than SISS due to the interaction of multiple nets through coupling capacitance.

Before presenting the theory and algorithm on GISS, we shall point out the suboptimality of our previous approach in [17], where the GISS problem is decomposed into a set of single-net OWS problems to iteratively compute the lower and upper bound widths for each net. However, we found later that such bound computations did not consider the delay of neighboring nets during each single-net optimization and, thus, they may miss the optimal solution, especially when the sink weights are very different (see [14] for more details). In this section, we develop a new theory and algorithm that can guarantee the *global optimality* of bound computations. We also introduce two new operations for global bound computation of each wire segment, namely, *lower BR* and *upper BR*.

### A. Dominance Property for GISS

First of all, it shall be noted that during GISS optimization, the capacitance of a wire segment  $E$  is not only a function (explicit or implicit) of its own width, but also a function of the widths of its neighboring

<sup>3</sup>In the weighted formulation, we can easily incorporate other design objectives, such as area and power for optimization.

<sup>4</sup>In this paper, we use a more accurate table-based capacitance model than those assumed in [11] and [12], then the complexity of the DP may not be polynomial as in [11]. However, it is not a major concern in our overall flow since the bound computation by itself already computes the near-optimal solution.

---

**Bound Computation for GISS by BR**

---

```

i ← 0
WU(i) ← Initialize upper bound for each net
WL(i) ← Initialize lower bound for each net
do
  for each net Nj
    WU(i+1) ← Upper bound refinement for each wire in Nj
    WL(i+1) ← Lower bound refinement for each wire in Nj
  while (WU(i) ≠ WU(i-1) OR WL(i) ≠ WL(i-1))

```

---

Fig. 2. BR-based algorithm to compute upper and lower bounds for GISS.

wire segments. Then, the width change of a wire segment  $E$  will affect the delay terms of not just the net that  $E$  belongs to, but also all the neighboring nets of  $E$ . First, let us define the following two BR operations for a wire segment  $E$  for GISS optimization.

**Definition 3—Upper BR:** Let  $W^U$  and  $W^L$  be an upper and lower bound to an optimal GISS solution. We consider the following two initial settings for all wire segments: 1) all wire widths take their upper bound from  $W^U$  and 2) all wire widths take their upper bound from  $W^U$ , except for  $E$ 's neighboring wire segments, which take their lower bound from  $W^L$ . We compute the optimal width for  $E$  [i.e., to minimize the delay objective in (4) with other wire widths fixed] under these two scenarios, denoted as  $w_1^U(E)$  and  $w_2^U(E)$ , respectively. Then, the upper BR width for  $E$  is  $\text{MAX}(w_1^U(E), w_2^U(E))$ .

**Definition 4—Lower BR:** Let  $W^U$  and  $W^L$  be an upper and lower bound to an optimal GISS solution. We consider the following two initial settings for all wire segments: 1) all wire widths take their lower bound from  $W^L$  and 2) all wire widths take their lower bound from  $W^L$ , except for  $E$ 's neighboring wire segments, which take their upper bound from  $W^U$ . We compute the optimal width for  $E$  under these two scenarios, denoted as  $w_1^L(E)$  and  $w_2^L(E)$ , respectively. Then, the lower BR width for  $E$  is  $\text{MIN}(w_1^L(E), w_2^L(E))$ .

Note that these two BR operations are different from the LR operation in [3], as the former uses *both* upper and lower bounds to obtain *either* a new lower *or* a new upper bound, while the latter uses *only* upper bound to compute a new upper bound and *only* lower bound to get a new lower bound. Then, we have the following theorem.

**Theorem 2—Dominance Property for GISS:** Under the monotone resistance model and the monotone lumped capacitance model, the upper BR for any wire segment will dominate its optimal width and the lower BR for any wire segment will be dominated by its optimal width in an optimal GISS solution.

### B. Algorithm for GISS Optimization

The *dominance property for GISS*, together with the two BR operations associated with it, leads to a very effective *global* upper and lower bound computation algorithm for the GISS optimization problem. As in SISS, the algorithm for GISS also has two phases. The first phase is the upper and lower bound computation using BR. The second phase is the DP algorithm to compute the final GISS solution between the lower and upper bounds. In practice, our bound computation phase is very effective such that most wire segments have identical lower and upper bound.

1) **Bound Computation for GISS:** The bound-computation algorithm for GISS is shown in Fig. 2. The lower and upper BR operations are used to iteratively update the width upper and lower bounds for each wire segment of each net. The algorithm starts at the iteration  $i = 0$ . First, we initialize upper and lower bounds of all wire widths specified by the user inputs and the layout constraints. A sample initialization is shown in Fig. 3. Let  $E_l$  and  $E_u$  be two parallel horizontal edges, with  $E_l$  below  $E_u$ . Let  $W_{\min}$  be the minimum wire width and  $S_{\min}$  be the minimum spacing from the layout constraint.

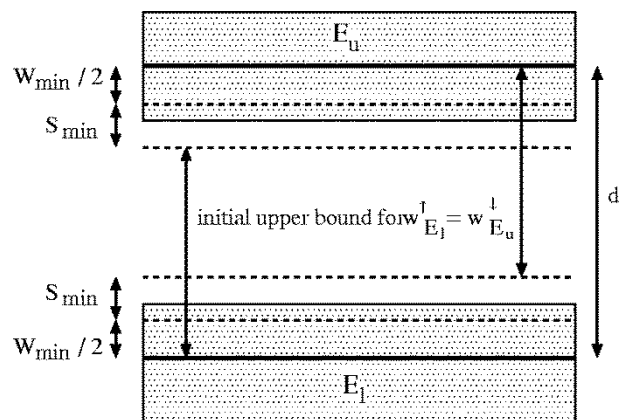


Fig. 3. Initialization of upper bound wire widths.

If the distance between the center lines of  $E_l$  and  $E_u$  is  $d$ , then the maximum width (i.e., the initial upper bound) for  $w^U_{E_l}$  (the side closer to  $E_u$ ) and  $w^L_{E_u}$  (the side closer to  $E_l$ ) is  $d - W_{\min}/2 - S_{\min}$ . The overall flow of iterative bound computation for GISS is exactly the same as that for SISS. The only difference is that GISS uses the BR operations and SISS uses the LR operation.

2) **DP for GISS:** In case that the upper and lower bounds do not meet, we will use the DP-based greedy algorithm to obtain the final wire-sizing solution for each net. We first take the lower bound width for each wire segment as the initial layout. Then, we perform the bottom-up DP similar to that for SISS to obtain the final GISS solution in a net-by-net manner following the decreasing order net priority. The major difference from SISS is that in GISS, our objective is the multiple-net weighted delay. So, the performance measure during the DP is now slightly modified to incorporate the delay terms from the neighboring nets (see [14] for more detail). All other DP processes for GISS is exactly the same as that for SISS, so we omit the details here. It shall be noted that the DP for SISS guarantees the optimality, but the DP for GISS can not guarantee it. Nevertheless, our optimal bound-computation process for GISS usually obtains convergent or tight lower and upper bounds, so the role of DP to a large extent is optional.

## V. EXPERIMENTAL RESULTS

We have implemented SISS and GISS algorithms using C++ on a SUN Ultra-SPARC 1 with 256-MB main memory. The parameters used in our experiments are based on a 0.18- $\mu\text{m}$  technology specified in NTRS roadmap [18]. We use the simple resistance model with the sheet resistance being  $0.0638 \Omega/\square$ . The minimum wire sizing  $W_{\min}$  is  $0.22 \mu\text{m}$  and minimum edge-to-edge spacing  $S_{\min}$  between neighboring wires is  $0.33 \mu\text{m}$ . In this case, the  $\text{min\_pitch}$ , defined as the sum of minimum spacing and minimum wire size, is equal to  $0.55 \mu\text{m}$ . The allowable wire widths for each side of the center line are from  $0.11$  to  $1.1 \mu\text{m}$ , with the incremental step to be  $0.11 \mu\text{m}$ . The area, fringing, and coupling capacitances are obtained by a table-lookup model simplified from the 2.5-D model in [19]. The driver resistance for each net is set to be  $119 \Omega$  (assuming each driver is 100 times of the minimum gate size) and the loading capacitance at each sink is  $12.0 \text{ fF}$ .

### A. SISS

We perform experiments for the optimal SISS algorithm on five nets extracted from an advanced industrial microprocessor. The number of sinks from Net1 to Net5 are 2, 3, 8, 3, and 15, respectively. We assign equal weight for each sink. Given the initial layout of these nets, we

TABLE I  
COMPARISON OF WEIGHTED DELAY AND AVERAGE WIRE WIDTH USING DIFFERENT ALGORITHMS

	length (mm)	Weighted Delay (ns)				Maximum Delay (ns)			
		MIN	OWS	SISS-S	SISS-A	MIN	OWS	SISS-S	SISS-A
Net1	3.6	0.08	0.08 (-0.0%)	0.08 (-0.0%)	0.08 (-0.0%)	0.14	0.14 (-0.0%)	0.14 (-0.0%)	0.14 (-0.0%)
Net2	6.6	0.31	0.19 (-39%)	0.18 (-42%)	0.15 (-52%)	0.34	0.22 (-35%)	0.22 (-35%)	0.19 (-44%)
Net3	10.07	0.78	0.71 (-9.0%)	0.71 (-9.0%)	0.61 (-22%)	1.03	0.96 (-6.8%)	0.95 (-7.8%)	0.83 (-19%)
Net4	10.57	0.54	0.41 (-24%)	0.39 (-28%)	0.27 (-50%)	0.84	0.71 (-16%)	0.65 (-23%)	0.44 (-48%)
Net5	31.98	3.19	2.57 (-19%)	2.57 (-19%)	1.73 (-46%)	4.92	4.26 (-13%)	4.27 (-13%)	3.26 (-34%)

TABLE II  
COMPARISON OF WEIGHTED DELAY AND AVERAGE WIRE WIDTH USING DIFFERENT ALGORITHMS

PS	Weighted Delay (ns)						Average Width ( $\mu\text{m}$ )					
	MIN	OWS	SISS	[17]	[20]	GISS	MIN	OWS	SISS	[17]	[20]	GISS
2 $\times$	1.51	1.26	0.80	0.81	0.80	0.76	1	0.64	0.75	0.68	0.69	0.66
3 $\times$	1.33	0.73	0.56	0.57	0.53	0.50	1	1.02	1.16	1.05	1.11	0.94
4 $\times$	1.28	0.46	0.46	0.46	0.45	0.40	1	1.36	1.54	1.34	1.49	1.21
5 $\times$	1.25	0.38	0.38	0.39	0.36	0.35	1	1.52	1.74	1.46	1.65	1.44
6 $\times$	1.23	0.35	0.33	0.36	0.32	0.32	1	1.52	1.75	1.48	1.64	1.63

TABLE III  
CONVERGENCE RATE (CR), AVERAGE GAP (GAP), AVERAGE NUMBER OF LOCAL REFINEMENT OPERATIONS PER EDGE (#BR), AND TOTAL RUNTIMES (CPU) USING GISS-S AND GISS-A ALGORITHMS

PS	GISS-S				GISS-A			
	CR	Gap ( $\mu\text{m}$ )	#BR	CPU(s): BR/DP	CR	Gap ( $\mu\text{m}$ )	#BR	CPU(s): BR/DP
2 $\times$	90.3%	0.011	2.95	0.41/0.10	85.2%	0.016	6.92	1.50/0.15
3 $\times$	61.9%	0.042	8.69	1.25/0.25	64.5%	0.040	13.8	4.35/0.62
4 $\times$	48.4%	0.061	9.76	1.64/0.44	66.6%	0.047	13.5	8.50/0.70
5 $\times$	48.4%	0.067	9.73	1.90/0.53	60.2%	0.057	14.3	11.4/2.30
6 $\times$	77.5%	0.025	6.96	1.56/0.17	83.4%	0.021	10.0	10.8/0.40
Avg	65.3%	0.041	7.62	1.35/0.30	72.0%	0.036	11.7	7.31/0.83

randomly assign some surrounding wire segments with spacing from the net being  $1-5 \times \text{min\_pitch}$ .

In Table I, we summarize the weighted delay and the maximum delay from different algorithms: minimum wire sizing (MIN); OWS algorithm without considering the coupling capacitance (but coupling capacitance is included in the HSPICE simulations for delay comparison); symmetric SISS algorithm (SISS-S) and asymmetric SISS algorithm (SISS-A).<sup>5</sup> Note that for all five nets, the bound computation leads to 100% convergence for both symmetric and asymmetric SISS optimization. In the parentheses under the columns of OWS, SISS-S, and SISS-A, we list the percentage of improvement over MIN. From the table, we can see that SISS-A consistently outperforms all other algorithms. In terms of its weighted delay, the improvement is up to 52% over the MIN solution (Net2), 34% over OWS (Net4), and 33% over SISS-S (Net5). Note that although the weighted delay is our objective, this formulation also help to reduce the maximum delay as well. From the table, we can see that SISS-A reduces the maximum delay from MIN, OWS, and SISS-S by up to 48%, 38%, and 32% (Net4), respectively.

## B. GISS

To demonstrate the effectiveness of our GISS algorithm on multiple nets, we have performed the experiments on a 16-bit parallel bus structure of 10-mm long with equal weight for every bus line and the PS between adjacent lines are  $\{2, 3, 4, 5, 6\} \times \text{min\_pitch}$ , respectively. Each wire segment is set to be 500  $\mu\text{m}$ .

Table II shows the weighted delay and average wire width obtained after HSPICE simulations from running different algorithms: MIN just

uses the minimum wire width, OWS performs OWS without explicit consideration of the coupling capacitance, SISS runs the the SISS optimization in Section III in a net by net manner, [17] and [20] are two previously reported algorithms for wire sizing and spacing, and GISS is our global optimization algorithm in Section IV. From Table II, we can see that GISS consistently obtains the best performance than all other algorithms by as much as 74% (for bus 6 $\times$ ), 40% (for bus 2 $\times$ ), 13% (for bus 4 $\times$ ), 13% (for bus 4 $\times$ ), and 11% (for bus 4 $\times$ ) than MIN, OWS, SISS, [17], and [20], respectively. Meanwhile, it is interesting to observe that better delay does not necessarily consume larger wiring area. The GISS algorithm has less wiring area than SISS and [20] in all cases. It also has less area than OWS and [17] in most cases.

Table III reports some statistics from the bound computations from the symmetric and asymmetric GISS algorithm, including the convergence rate, the average gap between the final lower and upper bounds for each wire segment, the average number of BR operations for each edge, and the runtime. We define a wire segment to be *convergent* if its lower and upper bounds are exactly the same. We can see that our bound computation is very effective as on average, 65% wire segments in symmetric GISS and 72% wire segments in asymmetric GISS are convergent. Even for those wire segments that are not convergent, the gaps between lower and upper bounds are very small, in most cases, just the wire width increment (0.11  $\mu\text{m}$  in our experimental setting). The average gap between the lower and upper bounds for all wire segments is only from 0.011 to 0.067  $\mu\text{m}$ . Note that since there are very few wire-width choices for each wire segment, the DP algorithm runs very fast (see the CPU breakup between BR and DP in Table III).<sup>6</sup>

<sup>5</sup>We use the suffix “-A” and “-S” to denote the asymmetric and symmetric wire sizing throughout this section. If not stated, the default is the asymmetric wire sizing.

<sup>6</sup>In fact, we also directly assigned the lower bound wire widths to be the final wire-sizing solution (i.e., without the DP step) and found very little delay/area difference from the GISS results in Table II.

TABLE IV  
COMPARISON OF WEIGHTED DELAY, AVERAGE WIRE WIDTH, AND CPU FOR  
TWO PARALLEL BUS LINES WITH WEIGHTS OF 1 AND 10000

PS	Weighted Delay (ns)			Average Width ( $\mu\text{m}$ )			CPU (s)		
	[17]	[20]	GISS	[17]	[20]	GISS	[17]	[20]	GISS
2 $\times$	0.463	0.395	0.394	1.309	1.172	1.155	5.93	4.27	0.12
3 $\times$	0.401	0.341	0.340	1.507	1.260	1.249	9.88	7.44	0.21
4 $\times$	0.377	0.315	0.315	1.683	1.353	1.342	14.7	11.9	0.26
5 $\times$	0.333	0.300	0.300	1.793	1.425	1.425	17.8	14.2	0.31
6 $\times$	0.306	0.295	0.295	1.793	1.425	1.425	18.0	14.4	0.32

As mentioned in the beginning of Section IV, the algorithm in [17] may fail to give correct lower and upper bound wire widths, especially when net weights differ drastically. To show such an example, we take two parallel bus lines with weights to be 1 and 10000, respectively. Other parameters are the same as those in the 16-bit bus example. The results by using [17] and [20] and GISS algorithms are listed in Table IV. It shows that [17] cannot take the net weights into consideration and leads to incorrect lower and upper bounds. In all cases, [17] leads to larger weighted delay (e.g., 20% for bus 2 $\times$ ) and wire width (e.g., 25% for bus 2 $\times$ ) than GISS. In this example, [20] obtains delay/width comparable to GISS. However, it uses significantly more CPU time by a factor of 40 $\times$  of that by GISS. This is because the bound computation stage by [20] does not give good convergence and it has to rely heavily on our bottom-up DP to pick the best solution between lower and upper bounds. Our bound-computation stage of GISS, however, leads to 100% convergence rate for this weighted bus example. Thus, it is much more effective than [20].

## VI. CONCLUSION

In this paper, we have developed efficient bound-computation algorithms and show their optimality under general interconnect resistance and capacitance models for wire-sizing and spacing optimizations in DSM designs. In practice, by bound computation alone, we can achieve optimal or near-optimal solutions. Our experimental results show significant performance improvement over previous wire-sizing/spacing algorithms.

It shall be noted that besides wire sizing and spacing, some other interconnect optimization techniques, such as driver sizing, buffer insertion, and sizing, can be used to improve interconnect performance as well. Also in practice, one may not need extensive wire tapering (i.e., using a lot of discrete wire widths, or even continuous wire shaping) to get near-optimal results. Some simplified wire-sizing schemes, such as uniform-width sizing [21], [22] and two-width sizing [21], may be good enough to get near-optimal solution, especially when optimal buffer insertion and sizing are performed so that all wires are not too long. In general, more freedom of wire sizing will lead to longer critical length for buffer insertion [23], which means less number of buffers. So there is a design tradeoff between buffer insertion/sizing versus wire-sizing/spacing.

In this paper, the focus is on the performance side of the coupling capacitance. Another important effect of the coupling capacitance is the crosstalk noise. In the future, we plan to extend the bound-computation and/or DP algorithm for simultaneous performance and noise optimization.

## ACKNOWLEDGMENT

The authors would like to thank M. K. Mohan, K. Soumyanath, and G. Srinivasa from the Intel Corporation for their input and support of this project and the anonymous reviewers for their helpful comments.

## REFERENCES

- [1] J. Cong, L. He, K.-Y. Khoo, C.-K. Koh, and D. Z. Pan, "Interconnect design for deep submicron ICs," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 1997, pp. 478–485.
- [2] J. Cong, K. S. Leung, and D. Zhou, "Performance-driven interconnect design based on distributed RC delay model," in *Proc. Design Automation Conf.*, June 1993, pp. 606–611.
- [3] J. Cong and K. S. Leung, "Optimal wire sizing under the distributed Elmore delay model," *IEEE Trans. Computer-Aided Design*, vol. 14, pp. 321–336, Mar. 1995.
- [4] J. Cong and L. He, "Optimal wire sizing for interconnects with multiple sources," *ACM Trans. Design Automat. Electron. Syst.*, vol. 1, no. 4, pp. 478–511, Oct. 1996.
- [5] C. P. Chen, Y. W. Chang, and D. F. Wong, "Fast performance-driven optimization for buffered clock trees based on Lagrangian relaxation," in *Proc. Design Automation Conf.*, June 1996, pp. 405–408.
- [6] J. P. Fishburn and C. A. Schevov, "Shaping a distributed-RC line to minimize Elmore delay," *IEEE Trans. Circuits Syst I*, vol. 42, pp. 1020–1022, Dec. 1995.
- [7] C.-P. Chen and D. F. Wong, "Optimal wire-sizing function with fringing capacitance consideration," in *Proc. Design Automation Conf.*, June 1997, pp. 604–607.
- [8] Y. Gao and D. F. Wong, "Optimal shape function for a bi-directional wire under Elmore delay model," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 1997, pp. 622–627.
- [9] N. Menezes, S. Pullela, F. Dartu, and L. T. Pillage, "RC interconnect synthesis—A moment fitting approach," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 1994, pp. 418–425.
- [10] T. Xue, E. S. Kuh, and Q. Yu, "A sensitivity-based wiresizing approach to interconnect optimization of lossy transmission line topologies," in *Proc. IEEE Multi-Chip Module Conf.*, Jan. 1996, pp. 117–121.
- [11] L. P. P. van Ginneken, "Buffer placement in distributed RC-tree networks for minimal Elmore delay," in *Proc. IEEE Int. Symp. Circuits and Systems*, May 1990, pp. 865–868.
- [12] J. Lillis, C. K. Cheng, and T. T. Y. Lin, "Simultaneous routing and buffer insertion for high performance interconnect," in *Proc. 6th Great Lakes Symp. VLSI*, Mar. 1996, pp. 148–153.
- [13] J. Cong, C.-K. Koh, and K.-S. Leung, "Simultaneous buffer and wire sizing for performance and power optimization," in *Proc. Int. Symp. Low Power Electronics and Design*, Aug. 1996, pp. 271–276.
- [14] J. Cong, L. He, C.-K. Koh, and D. Z. Pan, (2000) Interconnect Sizing and Spacing with Consideration of Coupling Capacitance. Tech. Rep. CSD-200011, Comput. Sci. Dept., Univ. California, Los Angeles, CA. [Online]. Available: <http://cadlab.cs.ucla.edu/~pan/publications/>.
- [15] W. C. Elmore, "The transient response of damped linear networks with particular regard to wide-band amplifiers," *J. Appl. Phys.*, vol. 19, no. 1, pp. 55–63, Jan. 1948.
- [16] C.-P. Chen, C. C. N. Chu, and D. F. Wong, "Fast and exact simultaneous gate and wire sizing by Lagrangian relaxation," *IEEE Trans. Computer-Aided Design*, vol. 18, pp. 1014–1025, July 1999.
- [17] J. Cong, L. He, C.-K. Koh, and Z. Pan, "Global interconnect sizing and spacing with consideration of coupling capacitance," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 1997, pp. 628–633.
- [18] *National Technology Roadmap for Semiconductors*, Semiconductor Industry Association, San Jose, CA, 1994.
- [19] J. Cong, L. He, A. B. Kahng, D. Noice, N. Shirali, and S. H.-C. Yen, "Analysis and justification of a simple, practical 2 1/2-d capacitance extraction methodology," in *Proc. ACM/IEEE Design Automation Conf.*, June 1997, pp. 40.1.1–40.1.6.
- [20] J. Cong and L. He, "Theory and algorithm of local-refinement based optimization with application to device and interconnect sizing," *IEEE Trans. Computer-Aided Design*, vol. 18, pp. 406–420, Apr. 1999.
- [21] J. Cong and D. Z. Pan, "Interconnect estimation and planning for deep submicron designs," in *Proc. Design Automation Conf.*, June 1999, pp. 507–510.
- [22] C. J. Alpert, A. Devgan, and S. T. Quay, "Is wire tapering worthwhile?," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 1999, pp. 430–435.
- [23] J. Cong and D. Z. Pan, "Interconnect delay estimation models for synthesis and design planning," in *Proc. Asia South Pacific Design Automation Conf.*, Jan. 1999, pp. 97–100.