# SIMULTANEOUS TRANSISTOR AND INTERCONNECT SIZING USING GENERAL DOMINANCE PROPERTY

*Jason Cong and Lei He*
*Department of Computer Science*
*University of California, Los Angeles, CA 90095* *


## ABSTRACT

We study the simultaneous transistor and interconnect sizing (*STIS*) problem in this paper. We define a class of optimization problems named *CH-posynomial programs* and show a general dominance property for all CH-posynomial programs (Theorem 1). Based on this property, a set of lower and upper bounds for the optimal solution of a CH-posynomial program can be computed through local refinement operations very efficiently (in polynomial time). We show that the STIS problem under a number of transistor and interconnect models is CH-posynomial programs, for example, when we use the RC tree model for interconnects and the step or the slope model for transistors. Preliminary experimental results show that in nearly all cases, the optimal solution is achieved because the recursive application of local refinement operations using the dominance property leads to the identical lower and upper bounds. Moreover, the STIS algorithm produces the more optimized solution when compared with optimal transistor sizing only.


## 1. INTRODUCTION

It is well recognized that interconnect delay has become the dominating factor in determining the circuit performance in deep submicron designs. We believe that the most effective way for performance optimization in deep submicron design is to consider both logic and interconnect designs throughout the entire design process. As part of our effort to develop a unified methodology and platform for simultaneous logic and interconnect design and optimization, we study the simultaneous transistor and interconnect sizing problem (*STIS*) for delay minimization in this paper.

Most previous works consider the *device sizing* and the *interconnect sizing* problems separately. The device sizing problem includes both *gate sizing* and *transistor sizing* problems. In [9], the transistor sizing problem was formulated as a posynomial programming problem under the Elmore delay formulation for the RC tree model and solved by a sensitivity-based heuristics. Later, the authors of [13] applied the delay model developed in [10] for their transistor sizing formulation and solved it by a convex programming technique. The gate sizing problem usually assumes that all transistor sizes within each gate scale isotropically [14] (i.e., all transistor sizes increase or decrease by an uniform factor), or each gate has a discrete set of implementations (cells) with different driving capabilities pre-designed as a given cell library for performance optimization. Recent gate sizing works can be found in [1, 2, 3]. However, all these works ignored the sizing of interconnects.

The interconnect sizing problem, often called the *wiresizing problem*, was first introduced in [6, 7] where the authors developed the first polynomial-time optimal wiresizing algorithm to minimize the weighted Elmore delay between the unique source and a set of critical sinks. Later on, the wiresizing problem to minimize the maximum delay along a RC tree was formulated as a posynomial programming problem and solved in [12] by the sensitivity-based algorithm similar to that in [9]. In addition, the optimal wiresizing problem for interconnects with multiple sources was formulated and solved optimally in [4]. All these works, however, did not consider the need to size the driving transistors again after interconnects have been changed.

Several recent studies consider both gate and interconnect sizing. The work in [5] formulated the simultaneous driver and wire sizing problem to size a single routing tree driven by a chain of cascaded drivers and developed an efficient optimal algorithm for both delay and power optimization. The authors of [11] studied the simultaneous wire sizing and buffer insertion problem for a single RC tree and solved it by a dynamic programming approach. Finally, the concurrent gate and wire sizing problem was studied in [15] using a sequential quadratic programming technique with the assumption that all transistor sizes in a gate are scaled isotropically.

However, if we assume that all transistor in a gate are sized isotropically, it often leads to suboptimal designs, especially in the full-custom layout. In this paper, we shall study the *simultaneous transistor and interconnect sizing (STIS) problem*, where the optimal size for *each* transistor (instead of a gate or cell) and the optimal wire width for *each* wire segment are computed independently in order to achieve more optimized designs. We show that the STIS problem under a number of transistor and interconnect models has the objective functions of the following form:

$$
\begin{aligned}
f(\mathbf{X}) \quad &= \quad \sum_{p,q,i,j} a_{pq,ij} \cdot \frac{x_j^q}{x_i^p} + \sum_{p,i} b_{p,i} \cdot \frac{1}{x_i^p} + \sum_{p,i} c_{p,i} \cdot x_i^p \\
\textit{where} \quad & \quad x_i, x_j \in \mathbf{X} = (x_1, \cdots, x_n) \\
& \quad p, q \in (1, 2, \cdots, m), \\
& \quad a_{pq,ij},\ b_{p,i}\ \textit{and}\ c_{p,i} \geq 0 \qquad (1)
\end{aligned}
$$

and its coefficients have the following symmetry:

1. $a_{pq,ij} > 0$ if and only if $a_{qp,ji} > 0$;
2. $b_{p,i} > 0$ only if $c_{p,i} > 0$ or $a_{pq,ij} > 0$ for any $q$ and $j$;

*This work is partially supported by ARPA/CSTO under contract J-FBI-93-112, the NSF Young Investigator Award MIP-9357582 and a grant from Intel Corporation under the NYI matching award program.

3. $c_{p,i} > 0$ only if $b_{p,i} > 0$ or $a_{pq,ij} > 0$ for any $q$ and $j$;

Following the notations in [8], we call the class of functions as *CH-posynomials*.

**Definition 1** *Eqn. (1) is a simple CH-posynomial if all coefficients are constants independent of* $\mathbf{X}$.

**Definition 2** *Eqn. (1) is a general CH-posynomial if coefficients are functions of* $x_i$ *and* $x_j$ *satisfying the following conditions:* $a_{pq,ij}$ *monotonically increases with respect to* $x_i$ *while monotonically decreases with respect to* $x_j$*, and* $b_{p,i}$ *monotonically increases with respect to* $x_i$ *while* $c_{p,i}$ *monotonically decreases with respect to* $x_i$.

Furthermore, we call an optimization problem to minimize a simple CH-posynomial as a *simple CH-posynomial program*, and to minimize a general CH-posynomial a *general CH-posynomial program.* We study the following *dominance property* for CH-posynomial programs.

**Definition 3 (Dominance Relation)** *For two vectors* $\mathbf{X}$ *and* $\mathbf{X}'$*, we say that* $\mathbf{X}$ *dominates* $\mathbf{X}'$ *(denoted as* $\mathbf{X} \geq \mathbf{X}'$*) if* $x_i \geq x'_i$ *for all* $i$.

**Definition 4 (Local Refinement Operation)** *The local refinement operation of a solution vector* $\mathbf{X}$*, with respect to any particular variable* $x_i$ *and function* $f(\mathbf{X})$*, is to minimize* $f(\mathbf{X})$ *subject to only evaluating* $x_i$ *while keeping the value of any* $x_j (j \neq i)$.

We say that the result solution vector is the *local refinement* of $\mathbf{X}$ (with respect to $x_i$). For simplicity of presentation, we shall use *solution* instead of solution vector.

**Definition 5 (Dominance Property)** *Let* $\mathbf{X}^*$ *be the optimal solution to minimize* $f(\mathbf{X})$*. If* $\mathbf{X}$ *dominates* $\mathbf{X}^*$*, a local refinement of* $\mathbf{X}$ *still dominates* $\mathbf{X}^*$*; If* $\mathbf{X}$ *is dominated by* $\mathbf{X}^*$*, a local refinement of* $\mathbf{X}$ *is still dominated by* $\mathbf{X}^*$

We proved the following important Theorem 1 that will lead to our STIS algorithm later on.

**Theorem 1** *The dominance property holds for both simple and general CH-posynomial programs.*

The authors of [7] first proposed the dominance property for their single-source optimal wiresizing formulation. Our results greatly generalize the concept of the dominance property and reveal that the dominance property holds for a large class of optimization problems instead of the single optimization problem in [7], which is an instance of the simple CH-posynomial program. The dominance property leads to an efficient algorithm to compute a set of lower and upper bounds of the optimal solution to a CH-posynomial program by local refinement operations very efficiently (in polynomial time). The algorithm has *guaranteed* convergence and optimality and can be applied to many optimization problems in VLSI CAD and other domains.

We will show that the STIS problem is CH-posynomial programs when we use the RC tree model for interconnects and the step or the slope model for transistors. Preliminary experimental results show that in nearly all cases, the optimal solution is achieved because the recursive application of local refinement operations using the dominance property leads to the identical lower and upper bounds. So the algorithm is optimal in the practical sense. Besides, the STIS algorithm produces the more optimized solution when compared with optimal transistor sizing only.

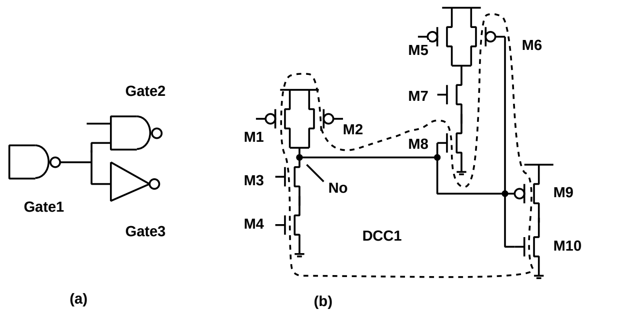


**Figure 1. (a) the gate-level circuit diagram where gate2 and gate3 are fanout of gate1. (b) the transistor-level circuit diagram where gate1 and the routing tree linked to its output comprise a *DC-Connected-Component* $DCC_1$. The output of gate1, denoted as $N_o$, is the root of the routing tree. The inputs of gate2 and gate3 are sinks of the routing tree.**

## 2. FORMULATIONS

We consider the CMOS static logic in this paper. Since the transistor channel length is usually fixed to the minimum feature size, we only allow the channel width to change during transistor sizing and shall use *width* to refer both the channel width of a transistor and the wire width of an interconnect. We will first formulate the delay as a function of the transistor/wire widths, then show the STIS problem for delay minimization is a CH-posynomial program.

### 2.1. Delay Formulation

Because an MOS transistor is a voltage-controlled device, an MOS circuit can be partitioned into a number of *DC-Connected Components (DCCs)* [18]. A DCC is a set of transistors and wires that are connected by DC-current paths containing only transistor channels or wires. The DC current can not cross the boundary of a DCC. In most cases, a DCC consists of a gate $G$ and a routing tree connecting the output (denoted as $N_o$) of $G$ to the inputs of all gates driven by $G$ (see Figure 1).

Our delay computation is similar to that in the switch level timing analysis tool Crystal [16]. The delay will be computed based on a *stage*, which is a DC-current path from a signal source (either the Vdd or the ground) to the gate of a transistor, including both transistors and wires (see Figure 2). The delay of a stage is the summary of delays through all transistors and wires in the stage.

#### A. Delay through transistors

A transistor can be modeled by the source-drain *effective resistance* $r$ and the gate, source and drain capacitances $c_g$, $c_s$ and $c_d$. Let $x$ be the transistor width, $r, c_g, c_s$ and $c_d$ can be written as the following:

$$
\begin{aligned}
r &= r_0/x \\
c_g &= c_{g0} \cdot x \\
c_s &= c_{s0} \cdot x \\
c_d &= c_{d0} \cdot x
\end{aligned}
$$

Where $c_{g0}, c_{s0}$ and $c_{d0}$ are the gate, source and drain capacitances for a unit-width transistor and can be viewed as constants without loss much accuracy. In addition, $r_0$ is called unit effective resistance that is defined in the following: Let

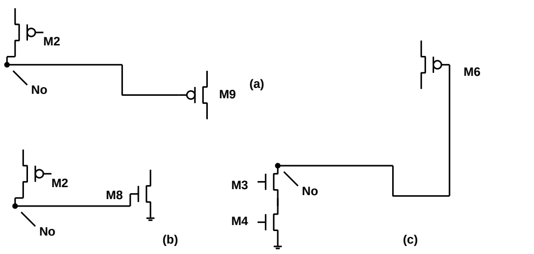


**Figure 2. For $DCC_1$ in Figure 1, (a) a stage from the Vdd to the gate of transistor $M_9$; (b) a stage from the Vdd to the gate of transistor $M_8$; (c) a stage from the ground to the gate of transistor $M_6$. Clearly, a transistor and a wire may belong to multiple stages.**

a rising input drive an inverter with total capacitance loading $C_L$. If the 50% delay is $\tau$, we say that $r = \tau / C_L$ is the effective resistance of the n-type transistor and the unit effective resistance $r_0$ is given by $r \cdot x$ with $x$ being the size of the n-type transistor. The unit effective resistance of the p-type transistor can be defined similarly by the delay under the falling input waveform. If $\mathbf{X} = \{\mathbf{x_1}, \cdots, \mathbf{x_n}\}$ is the sizing solution for all transistors and wires, in general, $r_0$ shall be a function of $\mathbf{X}$. For simplicity of presentation, we use $r_0$ instead of $r_0(\mathbf{X})$. However, it is worthwhile to mention that the unit effective resistance can consider the nonlinear characteristic of the transistor and the waveform slope effect. High accuracy can be achieved when using a table based method and more discussions will be given in Section 2.3.

Under this transistor model, the gate part of a DCC becomes an $RC$ network. Let $P^M(N_s, N_o)$ denote the path corresponding to all transistors in a stage. When computing the delay through these transistors, we only consider the effective resistances in path $P^M(N_s, N_o)$ and the capacitances connected to nodes in path $P^M(N_s, N_o)$. Let $c_k^s$ denote the total capacitance at node $N_k$ due to source/drain capacitances of all transistors linked to $N_k$ and $\mathcal{C}_o$ the total capacitance due to the routing tree and its sinks. If $R(P^M(N_s, N_k))$ is the total resistance of the partial path from the source $N_s$ to node $N_k$ in path $P^M(N_s, N_k)$, the Elmore delay through path $P^M(N_s, N_k)$, denoted as $t(P^M(N_s, N_o))$, is

$$\begin{aligned} t(P^M(N_s, N_o)) \quad &= \sum_{N_k \in P^M(N_s, N_o)} R(P^M(N_s, N_k)) \cdot c_k^s \\ &+ \quad R(P^M(N_s, N_o)) \cdot \mathcal{C}_o \qquad (2) \end{aligned}$$

### B. Delay through wires

A routing tree is modeled as a distribute RC tree, similarly to [7, 4]. Each sink has an extra capacitance due to the gate capacitance of a transistor in a gate driven by the routing tree. Each wire segment is divided into a sequence of sub-segments. Each sub-segment is treated as a $\pi$-type RC circuit. Since we assume that the wire width is uniform within a sub-segment, a sub-segment is called a *uni-segment*. Note that the segment division controls how aggressively we perform wiresizing optimization. Clearly, if the unit-width unit-length wire has wire resistance $r_0$, wire area capacitance $c_0$ and wire fringing capacitance $c_1$, the resistance $r$ and the capacitance $c$ for a uni-segment with width $x$ and length $l$ are

$$\begin{aligned} r &= r_0 \cdot l / x \\ c &= c_0 \cdot x \cdot l + c_1 \cdot l \end{aligned}$$

We treat node $N_o$ as the root of the routing tree. Consider node $N_k$ in the routing tree and let $T_k$ denote the subtree rooted at node $N_k$ (including node $N_k$). We denote the total capacitance within $T_k$ as $\mathcal{C}_k$, including the total wire capacitances and the total sink capacitances within $T_k$. Let $r_k$ be the resistance of the uni-segment $E_k$ with downstream node at $N_k$, the delay $t(P^I(N_o, N_t))$ along the unique path $P^I(N_o, N_t)$ between node $N_o$ and sink $N_t$ is

$$t(P^I(N_o, N_t)) = \sum_{N_k \in P^I(N_o, N_t)} r_k \cdot \mathcal{C}_k \qquad (3)$$

### C. Delay through a stage

With respect to Eqn. (2) and Eqn. (3), the delay $t(P(N_s, N_t), \mathbf{X})$ of stage $P(N_s, N_t)$ can be written as the following according to [19]:

$$\begin{aligned} & t(P(N_s, N_t), \mathbf{X}) \qquad (4) \\ = \; & t(P^M(N_s, N_o)) \; + \; t(P^I(N_o, N_t)) \\ = \; & \sum_{i,j} f_0^{st}(i,j) \cdot \frac{x_j}{x_i} \cdot l_i \cdot l_j \; + \; \sum_{i,j} f_1^{st}(i,j) \cdot \frac{1}{x_i} \cdot l_i \cdot l_j \; + \\ & \sum_i g^{st}(i) \cdot \frac{l_i}{x_i} \; + \; \sum_i h_0^{st}(i) \cdot l_i^2 \; + \; \sum_i h_1^{st}(i) \cdot \frac{l_i^2}{x_i} \end{aligned}$$

where $x_i$ is the width for a transistor $M_i$ or a wire uni-segment $E_i$, and $l_i$ is the length for a wire uni-segment $E_i$ or $l_i = 1$ for a transistor $M_i$. The coefficients $f_0^{st}, f_1^{st}, g^{st}, h_0^{st}$ and $h_0^{st}$ can be determined for stage $P(N_s, N_t)$ with respect to the given $r_0, c_{g0}, c_{s0}, c_{d0}, c_0$ and $c_1$ for either transistors or wires.

## 2.2. STIS Formulation to Minimize Delay for Multiple Critical Paths

In order to simultaneously minimize delay along multiple critical paths in a circuit, we propose to minimize the weighted delay $t(\mathbf{X})$ of all stages in these critical paths:

$$t(\mathbf{X}) \quad = \sum_{P(N_s, N_t) \in critical\ paths} \lambda^{st} \cdot t(P(N_s, N_t), \mathbf{X}) (5)$$

where the penalty weight $\lambda^{st}$ indicates the criticality of stage $P(N_s, N_t)$.

A simplified weight assignment scheme is used in this paper. The weight of a stage is 1 if the stage is in a critical path, otherwise, it is 0. Let

$$\begin{aligned} F_0(i,j) &= \sum_{P(N_s, N_t) \in critical\ paths} \lambda^{st} \cdot f_0^{st}(i,j) \\ F_1(i,j) &= \sum_{P(N_s, N_t) \in critical\ paths} \lambda^{st} \cdot f_1^{st}(i,j) \\ G(i) &= \sum_{P(N_s, N_t) \in critical\ paths} \lambda^{st} \cdot g^{st}(i) \\ H_1(i) &= \sum_{P(N_s, N_t) \in critical\ paths} \lambda^{st} \cdot h_1^{st}(i). \end{aligned}$$

and eliminate those terms independent of $\mathbf{X}$, Eqn. (5) becomes

$$\begin{aligned} & t(\mathbf{X}) \qquad (6) \\ = & \sum_{i,j} F_0(i,j) \cdot \frac{x_j}{x_i} \cdot l_i \cdot l_j + \sum_{i,j} F_1(i,j) \cdot \frac{1}{x_i} \cdot l_i \cdot l_j \\ + & \sum_i G(i) \cdot \frac{l_i}{x_i} + \sum_i H_1(i) \cdot \frac{l_i^2}{x_i} \end{aligned}$$

With respect to Eqn. (6), we define the following STIS problem to minimize delay through multiple critical paths:

**Formulation 1** *Given a circuit and the lower and upper bounds for the width of every transistor/wire, the STIS problem is to determine the width for every transistor/wire (or equivalently, a sizing solution* $\mathbf{X}$*) such that the weighted delay through multiple critical paths given by Eqn. (6) is minimized.*

In practice, it is often the case that we want to size the transistors and wires without increase in the layout area (using the *free* space in the current layout) or with bounded increase in the layout area. Therefore, there is an upper bound associated with each transistor and wire during the optimization. On the other hand, there is a lower bound associated with each transistor and wire due to the technology feature sizes and reliability concerns (like electromigration). Thus, the lower and upper bounds are used to handle these constraints. It will be seen later on that the lower and upper bounds are also the starting point for our *STIS* algorithm.

### 2.3. Dominance Property for STIS Problems

Coefficient functions $F_0, F_1, G$ and $H_1$ in Eqn. (6) are determined by parameters $r_0, c_{g0}, c_{s0}, c_{d0}, c_0$ and $c_1$. Since we assume that the unit resistance $r_0$ for wires and all capacitance parameters are constants independent of $\mathbf{X}$, the property of these coefficients will be determined by the unit effective resistance $r_0$ for the transistor.

The unit effective resistance $r_0$ for the transistor defined in Section 2.1.A is a function of the input waveform slope. The *step model* assumes that the input waveform is always a step so that $r_0$ is a constant independent of the sizing solution $\mathbf{X}$. As a result, all coefficients of Eqn. (6) are positive constants independent of $\mathbf{X}$. Thus, we have the following theorem.

**Theorem 2** *The STIS problem under the step model is a simple CH-posynomial program with the dominance property.*

The step model has been used in [9] for transistor sizing and in wiresizing works [7, 12, 4] to model the driver. It was also used in [5] for simultaneous driver and wire sizing. However, the step input is just an ideal assumption and it is well known that the delay under real waveforms will be larger than that under the step input. In order to consider the waveform slope effect on the unit effective resistance for the transistor, we define the following *DP-slope model* for the transistor.

**Definition 6** *The DP-slope model is a transistor model where the unit effective resistance for the transistor is an increasing function of its size.*

We proved that the STIS problem under a DP-slope model has the dominance property.

**Theorem 3** *The STIS problem under a DP-slope model is a general CH-posynomial program with the dominance property.*

An example of DP-slope model is the model developed in [10]. For an inverter, let $\tau_0$ be the delay under the step input and $\tau_s$ be the delay under the input with the transition time of $s$, [10] derived the following relation

$$\tau_s = \alpha \cdot s + \tau_0 \qquad (7)$$

where $\alpha$ is a constant determined by the technology. According to Eqn. (7), the unit effective resistance of a transistor is an increasing function of its input transition time. Because increasing the size of a transistor always increases the gate capacitance of the transistor, the input waveform will become slower due to a larger capacitance loading. Thus, the unit effective resistance of a transistor is an *increasing* function of its size.

Since Eqn. (7) is an accurate solution for the inverter delay, we believe that at least most models to consider the waveform slope effect are DP-slope models. It is worthwhile to mention that if Eqn. (7) is used to compute a gate delay by assuming that the input switch time is twice the Elmore delay in the previous stage, like the transistor sizing formulation in [13], the path delay is a simple CH-posynomial and the STIS problem to minimize the weighted delay of all stages in multiple critical paths is also a simple CH-posynomial program with the dominance property. However, Theorem 3 is much more general in the sense that it is applicable to a DP-slope model of any form and even without a closed form, as long as the effective unit resistance is an increasing function of the transistor size. A slope model without using a closed form will be discussed in Section 3.2. We would like to emphasize that the STIS algorithm will be developed for any DP-slope model which satisfies Theorem 3.

## 3. ALGORITHMS

### 3.1. Generic Algorithms to Exploit Dominance Property

For a CH-posynomial $f(\mathbf{X})$, let $\underline{\mathbf{X}}$ be the lower bound of its definition domain and $\overline{\mathbf{X}}$ the upper bound. Since its optimal solution $\mathbf{X}^*$ must be bounded by $\underline{\mathbf{X}}$ and $\overline{\mathbf{X}}$, a simple algorithm scheme, the *Local Refinement Algorithm (LRA)* scheme, can be used to compute a set of tighter lower and upper bounds for $\mathbf{X}^*$ by applying the local refinement operations beginning with either $\underline{\mathbf{X}}$ or $\overline{\mathbf{X}}$. We introduce the concept of *LR-tight* bound in the following.

**Definition 7** *A lower or upper bound is LR-tight if it can not be tightened any more by a local refinement.*

The LRA scheme is a greedy algorithm based on iterative local refinement operations. If beginning with a solution $\mathbf{X} = \underline{\mathbf{X}}$, the LRA scheme traverses every $x_i$ in certain order to perform a local refinement operation on it. Because $\mathbf{X}$ is dominated by $\mathbf{X}^*$, its local refinement is still dominated by $\mathbf{X}^*$. This process is repeated and $\mathbf{X}$ becomes increasingly closer to and remains dominated by $\mathbf{X}^*$. This process is stopped until no improvement is achieved on any $x_i$ in the last round of traversal and we obtain a LR-tight lower bound. A LR-tight upper bound can be obtained in the similar way by performing local refinement operations beginning with $\mathbf{X} = \overline{\mathbf{X}}$.

In essence, the LRA scheme generalize the greedy wiresizing algorithm GWSA first developed in [7]. If $r$ is the maximum number of the possible evaluations for any $x_i (i = \{1, \cdots, n\})$, the LRA scheme will converge in the polynomial time $O(r \cdot n^3)$.

### 3.2. Implementation of the DP-slope model

We assume a DP-slope model of the most general form with the only requirement to give the unit effective resistance $r_0$ for each discrete transistor size and apply a table based method in order to achieve the satisfactory trade-off between accuracy and complexity. In general, the effective unit resistance $r_0$ of a transistor is a function of its size, the input waveform slope and the capacitance loading. However, [17] proposed that all the three parameters could be combined into one factor called *slope ratio* to solely determine the unit effective resistance $r_0$ for the transistor. Thus, a one-dimensional table can be used instead of a three-dimensional table. We built a one-dimensional table for every type of transistors based on SPICE simulation results. Our implementation is similar to that in [16].

### 3.3. Overview of Near-Optimal STIS algorithm

The overall STIS algorithm includes three steps: to initialize the coefficient functions, to tighten lower and upper bounds of the optimal solution and to search the optimal solution between the LR-tight lower and upper bounds. Besides, the coefficient functions will be updated during the procedure to tighten lower and upper bounds because the unit effective $r_0$ for the transistor under the DP-slope model is a function of the current solution $\mathbf{X}$.

In order to efficiently initialize and update the coefficient functions, a circuit containing both transistors and interconnects is pre-partitioned into DCCs. Although the sizes of coefficient function $F_0$ and $F_1$ are $n \times n$, their operation complexities are reduced greatly by DCC partitioning because these coefficients are only needed to be computed for transistors and wires within a DCC.

It is worthwhile to mention that the LR-tight lower and upper bounds have the *zero* sensitivity. In other words, the sensitivity based method can not obtain a solution more optimized than the LR-tight lower or upper bound. If the LR-tight lower and upper bounds are identical for every transistor/wire, the optimal solution is achieved immediately, which happens almost all cases in practice. Because both the coefficient operations and the lower and upper bound operations are completed in polynomial-time, the optimal STIS solution is achieved by the polynomial-time in practice.

When the LR-tight lower and upper bounds do not meet, it is observed that the gap between the lower and upper bounds are very small, often just of one width in our experiments, and the percent of divergent transistors and wires is also very small, thus enumeration can be carried out in reasonable time.

## 4. EXPERIMENTAL RESULTS

We have implemented the STIS algorithm in ANSI C for the Sun SPARC station environment. Preliminary experiments will be presented in this section. The delays to be reported are computed using HSPICE. The use of HSPICE simulation results, instead of calculated Elmore delay values, not only shows the quality of our STIS solutions, but also verifies the validity of our transistor/interconnect modeling and the correctness of our STIS problem formulation.

The MCNC $0.5\mu m$ CMOS process technology is used. The wire width choices are $\{W_1, 2W_1, 3W_1, 4W_1, 5W_1\}$, where $W_1$ is the minimum wire width $(0.95\mu m)$ in the technology. The allowed transistor widths are $\{0.5\mu m, 0.6\mu m, \cdots, 200.0\mu m\}$ with step of $0.1\mu m$, which are determined by the design rules. Note that the local refinement operation does not depend on how feasible widths are defined, thus our algorithms are applicable to any width scheme for transistors and wires.

### 4.1. Transistor Sizing

The dominance property was only used for the optimal wire-sizing problem in the past. In this experiment, the STIS algorithm based on the dominance property is applied to the transistor sizing problem. The full adders from 2-bit to 16-bit implemented by complex gates are sized by assuming that all transistors are critical. Although we use an extreme upper bound ($200.0\ \mu m$) for each transistor, the performance of the STIS algorithm is still quite good. In Table 1, $nFET$ is the total transistor number in a circuit, *average* $nLoc$ is the average number of local refinement operations for a transistor, *time* is the total CPU time to size a circuit and *average width* is the average channel width for all transistors after transistor sizing. A transistor on average reaches its optimal width after 6–8 local refinement operations on it, and the time to size 514 transistors is just 19.78 seconds in a Sparc-10 workstation. The critical delay reduction by the optimal transistor sizing is up to 23.7%.

### 4.2. Comparison between STIS and Other Sizing Schemes

A 4-bit adder is used to driven a 1cm wire in IC technology. Three sizing schemes, i.e., minimum transistor/interconnect width, transistor sizing only, and STIS are used (see Table 2). When compared with the minimum transistor and wire sizing solution, transistor sizing only reduces the maximum delay by 13.58% and STIS reduces the maximum delay by 27.56%. Clearly, more optimized solution is achieved by the simultaneous transistor and wire sizing.

## 5. CONCLUSIONS

We formulated the *STIS* problem using a distributed RC tree model with consideration of the waveform slope effect for transistors, and developed efficient STIS algorithm based on recursive local refinement computations. The preliminary experiments have shown that the STIS algorithm produces the solution more optimized than that obtained by optimal transistor sizing only. We plan to integrate our STIS algorithm with a timing analysis tool like Crystal [16] and test more circuits in the future.

## ACKNOWLEDGMENTS

The authors would like to thank Professor C.-J. Richard Shi at University of Iowa, Kei-Yong Khoo and Cheng-Kok Koh at UCLA for their helpful discussions and assistance.

## REFERENCES

[1] M. Berkelaar, P. Buurman and J. Jess, "Computing the Entire Active Area/Power Consumption versus Delay Trade-off Curve for Gate Sizing with a Piecewise Linear Simulator", *Proc. IEEE Int'l. Conf. on Computer-Aided Design*, 1994, pp. 474-480.

[2] W. Chuang, S. S. Sapatnekar and I. N. Hajj, "Timing and Area Optimization for Standard-Cell VLSI Circuit Design", *IEEE Tran. on Computer-Aided Design*, March 1995, pp. 308-320.

| Adder | nFET | Average nLoc | Time (s) | Average Width | Critical Delay ($ns$) | |
|---|---|---|---|---|---|---|
| | | | | | min_width | opt_width |
| 2bit | 66 | 6.82 | 0.22 | 2.030 | 0.6021 | 0.4597 (-23.7%) |
| 4bit | 130 | 7.55 | 1.23 | 1.923 | 1.3511 | 1.0592 (-21.6%) |
| 8bit | 258 | 7.90 | 4.40 | 1.868 | 2.8681 | 2.4440 (-14.8%) |
| 16bit | 514 | 8.07 | 19.78 | 1.840 | 5.8878 | 4.6227 (-21.5%) |

**Table 1. Critical delay comparison between the minimum-width solution and the optimal-width solution.**

| | Total FET Area | Total Wire Area | Critical Delay |
|---|---|---|---|
| min FET + min Wire | 828 | 1000 | 2.3159ns(0.00%) |
| opt FET + min Wire | 1119 | 1000 | 2.0012ns(-13.58%) |
| STIS | 1268 | 4493 | 1.6776ns(-27.56%) |

**Table 2. Critical delay comparison among minimal sizing scheme, transistor sizing only, and the STIS solution.**


[3] G. Chen, H. Onodera and K. Tamaru, "An Iterative Gate Sizing Approach with Accurate Delay Evaluation", *Proc. IEEE Int'l. Conf. on Computer-Aided Design*, 1995, pp. 422-427.

[4] J. Cong and L. He, "Optimal Wiresizing for Interconnects with Multiple Sources", *Proc. IEEE Int'l. Conf. on Computer Design*, Nov. 1995 (full version as *UCLA Computer Science Dept. Tech. Report CSD-00031, 1995*).

[5] J. Cong, and C.-K. Koh, "Simulataneous Driver and Wire Sizing for Performance and Power Optimization", *IEEE Trans. on VLSI*, 2(4), December 1994, pp. 408-423.

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

[7] J. Cong and K. S. Leung, "Optimal Wiresizing Under the Distributed Elmore Delay Model", *IEEE Trans. on CAD*, 14(3), March 1995, pp. 321-336 (extended abstract in *Proc. IEEE Int'l. Conf. on Computer-Aided Design*, 1993, pp. 634-639).

[8] J. G. Ecker, "Geometric Programming: Methods, Computations and Appicaltions", *SIAM Review*, Vol. 22, No. 3, July 1980, pp. 338-362.

[9] J. P. Fishburn and A. E. Dunlop, "TILOS: A Psoynomial Programming Approach to Transistor Sizing", *Proc. IEEE Int'l. Conf. on Computer-Aided Design*, 1985, pp. 326-328.

[10] N. Hedenstierna, and K. O. Jeppson, "CMOS Circuit Speed and Buffer Optimization", *IEEE Tran. on CAD*, 1987, pp. 270-281.

[11] J. Lillis, C. K. Cheng and T. T. Y. Lin, "Optimal Wire Sizing and Buffer Insertion for Low Power and a Generalized Delay Model", *Proc. IEEE Int'l. Conf. on Computer-Aided Design*, Nov. 1995, pp. 138-143.

[12] S. S. Sapatnekar, "RC Interconnect Optimization Under the Elmore Delay Model", *Proc. ACM/IEEE Design Automation Conf.*, 1994, pp. 387-391.

[13] S. S. Sapatnekar, V. B. Rao, P. M. Vaidya, and S. M. Kang, "An Exact Solution to the Transistor Sizing Problem for CMOS Circuits Using Convex Optimization", *IEEE Tran. on CAD*, November 1993, pp. 1621-1634.

[14] N. Menezes, S. Pullela, and L. T. Pilegi, "Simultaneous Gate and Interconnect Sizing for Circuit-Level delay Optimization", *Proc. ACM/IEEE DAC*, 1995, pp. 690-695.

[15] N. Menezes, R. Baldick, and L. T. Pileggi, "A Sequential Quadratic Programming Approach to Concurrent Gate and Wire Sizing", *Proc. IEEE ICCAD*, 1995, pp. 144-151.

[16] J. K. Ousterhout, "A Switch-Level Timing Verifier for Digital MOS VLSI, " *IEEE Trans. on CAD*, 4(3) (1983) pp. 336-349.

[17] D. J. Pilling and J. G. Skalnik, "A Circuit Model for Predicting Transient Delays in LSI Logic Systems", *Proc. 6th Asilomar Conf. on Circuits and Systems*, 1972, pp. 424-428.

[18] V. B. Rao and I. Hajj, "Switch-Level Timing Simulation of MOS VLSI Circuits", *Proc. IEEE ISCAS*, 1985, pp. 229-232.

[19] J. Rubinstein, P. Penfield, and M. A. Horowitz, "Signal Delay in RC Tree Networks", *IEEE Trans. on CAD*, 2(3) (1983) pp. 202-211.