Simultaneous Buffer and Wire Sizing for Performance and Power Optimization*

Jason Cong, Cheng-Kok Koh
{cong, kohck}@cs.ucla.edu
Computer Science Dept., UCLA

Kwok-Shing Leung
ksleung@ichips.intel.com
Intel Corporation

Abstract

In this paper, we study the *simultaneous buffer and wire sizing (SBWS) problem* under two objective functions: (i) delay minimization only, and (ii) combined delay and power dissipation minimization. This paper extends our previous work on the simultaneous driver and wire sizing (SDWS) problem where no buffer is considered. Analogous to the DS/WS relation for optimal SDWS solutions, we prove the BS/WS relation for optimal simultaneous buffer and wire sizing solutions. This relation leads to a polynomial time algorithm for computing the lower and upper bounds of the optimal SBWS solutions and efficient optimal algorithms for computing optimal SBWS solutions. We have applied the optimal SBWS algorithms to the clock nets in a spread spectrum IF transceiver chip and accurate HSPICE simulations show that our algorithms can achieve power reduction by a factor of $2.6X$ and skew reduction by a factor of $3.5X$ when compared to the manual layout of the clock net in the original chip.

*This work is partially supported by ARPA/CSTO under Contract J-FBI-93-112, the National Science Foundation Young Investigator Award and a grant from Intel Corporation.

1 Introduction

In layout synthesis of high-performance low-power systems, it is important to consider and optimize both device design (i.e. transistor/cell design) and interconnect design simultaneously in order to achieve the objective of delay and power minimization. This new design methodology is especially important as the VLSI fabrication technology advances to submicron device dimension and gigahertz clock frequency, where interconnect delay becomes an important factor in determining circuit speed, and the *distributed* nature of the interconnect structure must be considered. Moreover, the bulk of the dynamic power dissipation is due to the interconnect capacitance. The objective of this paper is to study the problem of buffers and wire sizing for delay and power optimization. This technique is particularly applicable to buffered clock tree optimization since clock tree generally spans the entire chip and has significant interconnect delay, and the clock signals normally operate at the highest frequency of all signals and dissipate a significant amount of power.

Recent studies show that interconnect delay can be reduced by interconnect topology optimization for both general and clock nets. For example, interconnect topologies such as bounded-radius bounded-cost trees [10], AHHK trees [1], maximum performance trees [8], A-trees [15], low-delay trees [4], and IDW/CFD trees [19] have been proposed to optimize general nets. For clock nets, zero-skew tree (ZST) construction [3, 5, 17] and bounded-skew tree (BST) construction [13, 20, 11] have been studied extensively. Interconnect delay can be further reduced by sizing device and wire. The wiresizing algorithms in [15, 16, 9, 24, 21] can minimize interconnect delay by optimally assigning different wire width to each wire segment in the interconnect design. Both interconnect topology optimization and wiresizing optimization are effective when resistance ratio, i.e. the driver resistance versus unit wire resistance, is small in the design [15]. Very recently, [14, 12] explore the possibility of simultaneous driver and wire sizing (SDWS) for performance and power optimization. The follow-up work by [22] considers simultaneous gate and interconnect sizing to optimize performance. In the SDWS problem, the interconnect tree is driven by a chain of cascaded drivers. [12] reveals, among other interesting properties, the DS/WS relation between driver and wire sizing solutions. These results lead to polynomial time algorithms for computing the lower and upper bounds of the optimal SDWS solutions and efficient algorithms for computing optimal SDWS solutions. However, it is not difficult to see that this method is overly restrictive in that it did not consider buffers in the interconnect tree which are normally inserted to distribute the capacitive load among the buffers [2]. The works by [25, 21, 26] consider buffer insertion for either performance optimization or power minimization.

In this paper, we propose to solve the problem of *simultaneous buffer and wire sizing (SBWS)* for performance and power optimization. In the SBWS problem, we assume that the buffer locations are given and the variables in this problem are the buffer sizes and the wire widths. The major contribution of this paper is that we generalize the results on simultaneous driver and wire sizing where no buffer is considered. Analogous to the DS/WS relation, we prove the BS/WS relation between buffer sizing and optimal wire sizing. The relation leads to a polynomial time algorithm for computing the lower and upper bounds of the optimal SBWS solutions and efficient optimal algorithms for computing optimal SBWS solutions. We have applied the SBWS algorithms to the clock nets in a spread spectrum IF transceiver chip designed at UCLA for wireless multimedia information systems [7]. Accurate HSPICE simulations show that our algorithms can achieve power reduction by a factor of $2.6X$ when compared to the manual layout of the clock nets in the original chip. On top of lower power dissipation, the clock nets have smaller maximum clock delay, smaller maximum skew and sharper signal waveform.

The remainder of this paper is organized as follows: In Section 2, we present the formulation of the SBWS problem under the delay minimization objective. In Section 3, we present interesting properties of optimal SBWS solutions. In Section 4, we present a polynomial algorithm to compute the upper and lower bounds of the optimal SBWS solution followed by two polynomial optimal algorithms to compute the optimal SBWS solution. Section 5 shows the experimental results obtained by our SBWS algorithms. Section 6 concludes the paper with discussions of future work.

2 Problem Formulation

Assume that we are given a buffered routing tree T implementing a signal net $\mathcal{N}$ which consists of a set of m sinks $\{N_1, N_2, \cdots, N_m\}$, and a set of b buffers $\{D_2, D_3, \cdots, D_b\}$ at fixed locations in T. Furthermore, we assume that the signal net is driven by a driver D_1 of a given size d_1 at the source as shown in Figure 1. A sink N_i has a loading capacitance of $c^s_{N_i}$. A node in T refers to the source, or a buffer, or a sink, or a Steiner node, and a segment connects two nodes in T. Assume that $\{E_1, E_2, \cdots, E_n\}$ is the set of segments forming the tree T, where n is the total number of segments in the tree.

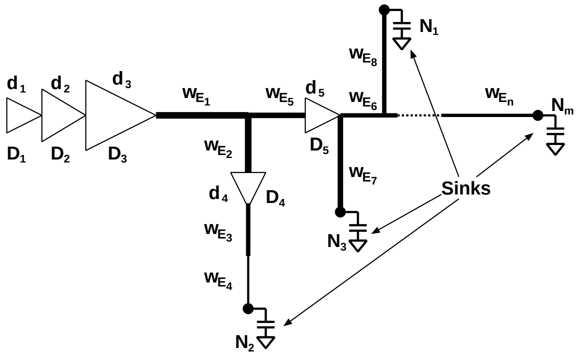

Figure 1: A buffered interconnect tree T with sinks $\{N_1, N_2, \cdots, N_m\}$. w_{E_i} denotes the width of the wire segment E_i, $i = 1..n$ and d_i denotes the size of buffer/driver D_i.

We assume that each wire segment has a set of discrete choices of wire widths $\{W_1, W_2, \cdots, W_r\}$ ($W_1 < W_2 < \cdots < W_r$). We use w_{E_i} to denote the width of the wire segment E_i, $i = 1..n$. We denote the the size of a buffer, say D_i, by d_i. Given the above definitions, the problem of simultaneous buffer and wire sizing (SBWS) for performance optimization can be defined as follows:

Definition 1 *Given a buffered routing tree T of a signal net $\mathcal{N}$ with prescribed buffer locations, the SBWS problem for delay minimization is to find a buffer sizing solution $\mathcal{D} = \{d_2, d_3, \cdots, d_b\}$ and wiresizing solution $\mathcal{W} = \{w_{E_1}, w_{E_2}, \cdots, w_{E_n}\}$ such that the performance measure $t_T(\mathcal{D}, \mathcal{W})$ is optimized.*

The performance measure $t_T(\mathcal{D}, \mathcal{W})$ evaluate the "signal delay" of the net from the source to one or several critical sinks, and it is expressed as a linear combination of the signal delays from the source to all sinks [16, 13, 4]:

$$t_T(\mathcal{D}, \mathcal{W}) = \sum_{1 \le i \le m} \lambda_{N_i} \cdot t_T^{N_i}(\mathcal{D}, \mathcal{W}) \tag{1}$$

where λ_{N_i} measures the criticality of sink N_i and $t_T^{N_i}(\mathcal{D}, \mathcal{W})$ is the signal delay from the source driver D_1 to sink N_i in the buffered tree T. Note that we use t_T without superscript to denote the weighted sum of delay and $t_T^{N_i}$ with superscript N_i to denote the Elmore delay [18] from source driver to sink N_i.

The weighted-sum formulation can be used in two scenarios. For performance optimization, large λ's are used for timing critical sinks. For clock skew minimization, clock pins with longer (shorter) delay should be assigned with higher (lower) criticality. In both cases, we *normalize* λ_{N_i}'s such that $\sum_{i=1..n} \lambda_{N_i} = 1$. The weighted-sum formulation leads to efficient optimal wiresizing algorithm due to the separability, the monotone property, and most importantly, the dominance property [16]. However, the separability property does not hold for minimizing maximum delay [24] and it is not clear if the dominance property still holds for maximum delay minimization. Although there is a polynomial time algorithm for minimizing maximum delay [21], the actual run-time is long and the memory requirement is prohibitively large (see experimental results in Section 5). [6] shows that by assigning appropriate weight of each sink based on Lagrangian relaxation, the weighted-sum formulation can be used iteratively to minimize maximum delay.

Although buffer and wire sizing are effective approaches to reduce interconnect delay, the drawback is that larger buffer size and additional routing area also increase the power dissipation. In practice, circuit design requires a careful tradeoff between performance and power. We define the SBWS problem for both delay and power optimization (SBWS-DP) as follows:

Definition 2 *Given a buffered routing tree T for net $\mathcal{N}$ with prescribed buffer locations, the SBWS problem for both delay and power optimization (SBWS-DP) is to determine a buffer sizing solution $\mathcal{D}$, and a wiresizing solution $\mathcal{W}$, such that the performance measure $obj_T(\mathcal{D}, \mathcal{W})$ defined below is minimized:*

$$obj_T(\mathcal{D}, \mathcal{W}) = \alpha \cdot Power_T(\mathcal{D}, \mathcal{W}) \; + \; \gamma \cdot t_T(\mathcal{D}, \mathcal{W}) \tag{2}$$

where $t_T(\mathcal{D}, \mathcal{W})$ is the performance measure and $Power_T(\mathcal{D}, \mathcal{W})$ is the power dissipation.

We estimate the performance measure and the power dissipation as follows.

2.1 Delay Models for Buffer and Interconnect

2.1.1 RC Delay Model for Buffers

In our formulation, we assume that a buffer/driver is an inverter. As in [12], we model a buffer as a switch-level RC device. Figure 2(a) shows a minimum-size inverter (buffer) with a p-transistor resistance R_p and a n-transistor resistance R_n. We assume that $R_p = R_n = R_{min}$. Let C_g denote the gate capacitance and C_d denote the capacitance due to the source and drain diffusion of the minimum-size buffer/driver. A buffer/driver with size d has an effective resistance of $\frac{R_{min}}{d}$, a gate capacitance of $C_g \cdot d$ and a diffusion capacitance of $C_d \cdot d$. If the total capacitance of the net driven by the driver is CL_T, then the delay due to the driver is $\frac{R_{min}}{d} \cdot (C_d \cdot d + CL_T)$.

Note however that our results and algorithms on SBWS is not restricted to this simple model only. As long as the effective resistance is monotonically decreasing and the input gate capacitance is monotonically increasing as the size increases, the algorithms proposed in this paper still apply since the BS/WS relation only assumes that as buffer/driver size increases, its input capacitance increases and resistance decreases.

2.1.2 Elmore Delay Model for Interconnect without Intermediate Buffers

As in the many works on wiresizing [16, 12, 9], we use the distributed Elmore delay model [18] for interconnect delay measure. In the following, we consider a routing tree connecting a driver to a set of sinks with no buffer inserted. We use T to denote the *unbuffered* routing tree, N_i to denote sinks, N_+ to denote the source and E_i to denote edges. The formulations used in this section are based on those in [16, 12]. We use c^s_u to denote the capacitance of sink u. If the sink u is actually a buffer D_i, then $c^s_u = C_g \cdot d_i$. Let the driver of the tree be D and the size of the driver be d. Then, the driver has resistance R_{min}/d and capacitance $C_d \cdot d$. We introduce a RC segment D–N_+ to capture the resistance and output capacitance of driver

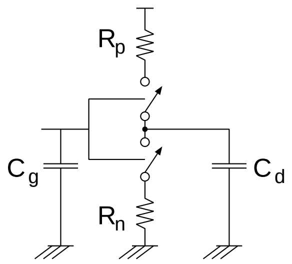

Figure 2: A switch-level RC model of a driver.

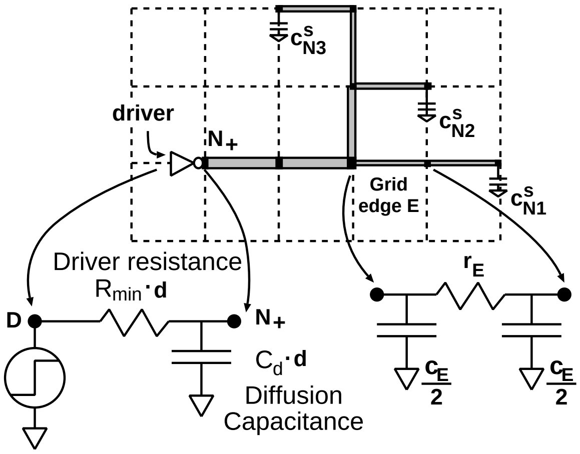

Figure 3: A uniform grid structure for the distributed RC interconnect model. (a) The layout of an interconnect tree T with 3 sinks N_1, N_2, and N_3. (b) The corresponding distributed RC interconnect model of T. Each grid edge E in T connecting two adjacent nodes is modeled as a π-type RC circuit containing a resistor of r_E and two capacitors of $\frac{c_E}{2}$ each. Each sink has an extra loading capacitance.

D as in Figure 3. With the derivation given in [12], the performance measure of the routing tree T, denoted $t_T(\mathcal{D}, \mathcal{W})$ is defined to be the linear combination of all driver-to-sink delays, i.e.,

$$\begin{aligned}
t_T(\mathcal{D}, \mathcal{W}) &= \sum_{N_i \in sink(T)} \lambda_{N_i} \cdot t_T^{N_i}(\mathcal{D}, \mathcal{W}) \\
&= \sum_{N_i \in sink(T)} \lambda_{N_i} \cdot \mathcal{K}_{1i} + \mathcal{K}_1 \cdot \sum_{E \in T} H(E) \cdot \frac{1}{w_E} + \mathcal{K}_2 + \mathcal{K}_3 \cdot \sum_{E \in T} w_E + \\
&\quad \mathcal{K}_4 + \mathcal{K}_5 \cdot \sum_{E,E' \in T, E \neq E'} F(E,E') \cdot \frac{w_{E'}}{w_E} + \mathcal{K}_6 \cdot \sum_{E,E' \in T, E \neq E'} F(E,E') \cdot \frac{1}{w_E} + \\
&\quad \mathcal{K}_7 \cdot \sum_{E \in T} G(E) \cdot \frac{1}{w_E}
\end{aligned} \tag{3}$$

where $\mathcal{K}$'s are coefficients dependent on the driver size d and other technology parameters such as unit wire resistance and capacitance, and the functions F, G and H are dependent on the topology. The details can be found in [12].

2.2 Performance Measure for Buffered Tree

Let T now be the *buffered* routing tree for net $\mathcal{N}$. A buffer in T decomposes the net into two subnets. In general, $b-1$ buffers in T decompose the net into a hierarchy of b subnets. By a top-down topological sorting of the subnets, we can order the subnets as $\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_b$ and define a parent-child relationship of the subnets. Each buffer D_i is a sink with capacitance $C_g \cdot d_i$ of the parent subnet and the driver with resistance R_{min}/d_i and output capacitance $C_d \cdot d_i$ of the child subnet. Clearly, subnet $\mathcal{N}_1$ is driven by driver D_1. Similarly, we order the buffers such that buffer D_i drives subnet $\mathcal{N}_i$. Also, we denote the routing subtree for subnet $\mathcal{N}_i$ by T_i. Note that the T_i is a unbuffered routing tree. We denote the size of driver D_i by d_i. Note that a chain of b cascaded drivers in the SDWS problem is formulated as b subnets under this new formulation.

Consider a sink N_i in subnet $\mathcal{N}_k$ ($N_i \in \mathcal{N}_k$). Suppose $\mathcal{N}_j$ is the parent subnet of $\mathcal{N}_k$, i.e. buffer D_k is considered a sink driven by D_j. Then, the loading capacitance of driver D_k in $\mathcal{N}_j$ is $C_g \cdot d_k$ and the delay from driver D_j to D_k, denoted $t_{T_j}^{D_k}(\mathcal{D}, \mathcal{W})$, is as follows:

$$t_{T_j}^{D_k}(\mathcal{D}, \mathcal{W}) = \sum_{u \in P(D_j, D_k)} r_u \cdot \left(\frac{c_u}{2} + C_u\right) \tag{4}$$

where the summation is taken over all the segments on the path $P(D_j, D_k)$ from the driver D_j to the node D_k in subnet $\mathcal{N}_j$. Therefore, the delay from driver D_j to sink N_i is $t_{T_j}^{D_k}(\mathcal{D}, \mathcal{W}) + t_{T_k}^{N_i}(\mathcal{D}, \mathcal{W})$.

Let $Parent(N_i)$ denote the buffer driving the sink N_i. Similarly, we define $Parent(D_i)$ to be the buffer driving the buffer D_i. Clearly, D_1 has no parent. Define recursively, the delay from the source driver D_1 to sink N_i in the buffered tree is as follows:

$$t_T^{N_i}(\mathcal{D}, \mathcal{W}) = \begin{cases} t_{T_1}^{N_i}(\mathcal{D}, \mathcal{W}) & \text{if } Parent(N_i) = D_1 \\ t_T^{Parent(N_i)}(\mathcal{D}, \mathcal{W}) + t_{T_{Parent(N_i)}}^{N_i}(\mathcal{D}, \mathcal{W}) & \text{otherwise} \end{cases} \tag{5}$$

Note that we use $t_T^{N_j}$ to denote the delay to N_j in the entire routing tree T and $t_{T_i}^{N_j}$ to denote the delay in subtree T_i to sink N_j which is in subnet $\mathcal{N}_i$. Consider a buffer D_i that drives sinks only, i.e. $\mathcal{N}_i$ contains only one buffer, D_i. Let $Children(D_i)$ denote the set of sinks driven by D_i and we define λ_{D_i} to be $\sum_{N_j \in Children(D_i)} \lambda_{N_j}$. Suppose D_i is not D_1, then

$$\begin{aligned} t_T(\mathcal{D}, \mathcal{W}) &= \sum_{N_j \in Children(D_i)} \lambda_{N_j} \cdot \left(t_T^{D_i}(\mathcal{D}, \mathcal{W}) + t_{T_i}^{N_j}(\mathcal{D}, \mathcal{W})\right) + \sum_{N_j \in \mathcal{N} - Children(D_i)} \lambda_{N_j} \cdot t_T^{N_i}(\mathcal{D}, \mathcal{W}) \\ &= \lambda_{D_i} \cdot t_T^{D_i}(\mathcal{D}, \mathcal{W}) + \sum_{N_j \in Children(D_i)} \lambda_{N_j} \cdot t_{T_i}^{N_j}(\mathcal{D}, \mathcal{W}) + \sum_{N_j \in \mathcal{N} - Children(D_i)} \lambda_{N_j} \cdot t_T^{N_j}(\mathcal{D}, \mathcal{W}) \end{aligned} \tag{6}$$

If $\lambda_{D_i} > 0$, then $\sum_{N_j \in Children(D_i)} \lambda_{N_j}/\lambda_{D_i} = 1$. Suppose within the subnet $\mathcal{N}_i$, we define the criticality of a sink N_j in subnet $\mathcal{N}_i$ to be $\lambda'_{N_j} = \lambda_{N_j}/\lambda_{D_i}$. Then, we can now write Eqn. (6) as

$$t_T(\mathcal{D}, \mathcal{W}) = \lambda_{D_i} \cdot t_T^{D_i}(\mathcal{D}, \mathcal{W}) + \lambda_{D_i} \cdot t_{T_i}(\mathcal{D}, \mathcal{W}) + \sum_{N_j \in \mathcal{N} - Children(D_i)} \lambda_{N_j} \cdot t_T^{N_j}(\mathcal{D}, \mathcal{W}) \tag{7}$$

Clearly, $\sum_{N_j \in \mathcal{N} - Children(D_i)} \lambda_{N_j} + \lambda_{D_i} = 1$. Now, we treat D_i as a sink with a loading capacitance of $C_g \cdot d_i$ in the net consisting of $\mathcal{N} - Children(D_i) + D_i$. Repeating the same process bottom-up, we can eventually write the delay measure of the buffered tree T as follows:

$$t_T(\mathcal{D},\mathcal{W}) \quad = \quad \sum_{\text{all } D_i} \lambda_{D_i} \cdot t_{T_i}(\mathcal{D},\mathcal{W}) \tag{8}$$

Note that when $\lambda_{D_i} = 0$, then $\lambda_{N_j} = 0$ for all $N_j \in Children(D_i)$. In this case, we relax the definition of $t_{T_i}(\mathcal{D},\mathcal{W})$ such that it is not necessary for the criticality values of sinks in the net to be normalized. We can conclude from the above equation that if we are given the buffer sizes, we can compute the optimal wiresizing solution using the technique in [16].

2.3 Performance and Power Optimization

2.3.1 Power Dissipation Formulation

There are two components that determine the amount of power dissipated in a CMOS circuit, namely, *static* dissipation due to leakage current, and *dynamic* dissipation due to switching transient current (short-circuit dissipation) and charging and discharging of load capacitances (capacitive dissipation) [27]. We consider only dynamic dissipation in our formulation since static dissipation is usually 2 to 3 order of magnitude smaller. For simplification, we consider only capacitive power in the following. However, the results still hold when short-circuit dissipation power is also considered. Given a total capacitance CL driven by the driver, we can write the capacitive dissipation of the single driver of size d as follows [27]:

$$Power_{cap} \quad = \quad f \cdot CL \cdot V_{dd}^2 \tag{9}$$

where f is the switching frequency of the input signal. To simplify the expression, we let $\mathcal{L} = f \cdot V_{dd}^2$ and write $Power_{cap} = \mathcal{L} \cdot CL$. Note that CL has three components: the diffusion capacitance of the driver, $C_d \cdot d$, the wire capacitance of the routing tree and the total capacitance of the sinks in the tree.

2.3.2 Trade-Off Between Performance and Power

Consider an unbuffered tree T driven by driver D with size d, we denote the total capacitance due to the routing tree by $CL_T(\mathcal{D},\mathcal{W})$ which is computed as

$$CL_T(\mathcal{D},\mathcal{W}) \quad = \quad \sum_{u \in sink(T)} c_u^s + c_0 \cdot \sum_{E \in T} w_E + \sum_{E \in T} c_1 \tag{10}$$

The total (capacitive) power dissipation by D is

$$Power_T(\mathcal{D},\mathcal{W}) \quad = \quad \mathcal{L} \cdot \left(C_d \cdot d + \sum_{u \in sink(T)} c_u^s + c_0 \cdot \sum_{E \in T} w_E + \sum_{E \in T} c_1 \right) \tag{11}$$

We can now write the trade-off between performance and power in Eqn. (2) as follows:

$$\begin{aligned} obj_T(\mathcal{D},\mathcal{W}) \quad = \quad & \alpha \cdot \mathcal{L} \cdot \left(C_d \cdot d + \sum_{u n sink(T)} c_u^s + c_0 \cdot \sum_{E \in T} w_E + \sum_{E \in T} c_1 \right) + \\ & \gamma \cdot \left(\sum_{N_i \in sink(T)} \lambda_i \cdot \mathcal{K}_{1i} \; + \; \mathcal{K}_1 \cdot \sum_{E \in T} H(E) \cdot \frac{1}{w_E} \; + \; \mathcal{K}_2 \; + \; \mathcal{K}_3 \cdot \sum_{E \in T} w_E \; + \; \mathcal{K}_4 \; + \right. \end{aligned}$$

$$\mathcal{K}_5 \cdot \sum_{E,E' \in T, E \neq E'} F(E,E') \cdot \frac{w_{E'}}{w_E} + \mathcal{K}_6 \cdot \sum_{E,E' \in T, E \neq E'} F(E,E') \cdot \frac{1}{w_E} +$$
$$\left. \mathcal{K}_7 \cdot \sum_{E \in T} G(E) \cdot \frac{1}{w_E} \right) \qquad (12)$$

Grouping the terms belonging to $CL_T(\mathcal{D}, \mathcal{W})$ and $t_T(\mathcal{D}, \mathcal{W})$, we obtain an expression $\hat{t}_T(\mathcal{D}, \mathcal{W})$ that is similar to $t_T(\mathcal{D}, \mathcal{W})$ except for differences in the coefficients. In other words, $obj_T(\mathcal{D}, \mathcal{W}) = \alpha \cdot \mathcal{L} \cdot C_d \cdot d + \hat{t}_T(\mathcal{D}, \mathcal{W})$. Therefore, when we consider a buffered tree T, the trade-off objective is

$$\begin{aligned} obj_T(\mathcal{D}, \mathcal{W}) &= \alpha \cdot \mathcal{L} \cdot C_d \cdot \sum_{\text{all } D_i} d_i + \sum_{\text{all } D_i} \lambda_{D_i} \hat{t}_{T_i}(\mathcal{D}, \mathcal{W}) \\ &= \alpha \cdot \mathcal{L} \cdot \cdot C_d \cdot d_1 + \sum_{\text{all } D_i} \left\{ \lambda_{D_i} \hat{t}_{T_i}(\mathcal{D}, \mathcal{W}) + \sum_{D_j \in Children(D_i)} \alpha \cdot \mathcal{L} \cdot C_d \cdot d_j \right\} \end{aligned} \qquad (13)$$

Note that the term $\lambda_{D_i} \hat{t}_{T_i}(\mathcal{D}, \mathcal{W}) + \sum_{D_j \in Children(D_i)} \alpha \cdot \mathcal{L} \cdot C_d \cdot d_j$ is again very similar to $\lambda_{D_i} t_{T_i}(\mathcal{D}, \mathcal{W})$ except for differences in the coefficients. Therefore, in the following, we will present the results which assumes optimization of $t_T(\mathcal{D}, \mathcal{W})$ in Eqn. (8) instead of $obj_T(\mathcal{D}, \mathcal{W})$.

3 Properties of Optimal SBWS Solutions for Performance and Power Optimization

The key to solving the simultaneous driver and wire sizing (SDWS) problem is the DS/WS relation between a driver sizing solution and a wiresizing solution [12]. The optimal SDWS solution is computed by first computing the upper and lower bounds of the optimal solution followed by, if necessary, exploring the extensively pruned solution space using dynamic programming. In most of the cases, the first step is able to yield the optimal solution since the upper and lower bounds meet. The key to solving the SDWS problem is the DS/WS relation between a driver sizing solution and a wiresizing solution [12]. In this paper, we will again use the same approach to solve the SBWS problem. We first review a few results stated in [12].

3.1 Review of Known Results on Optimal SDWS Solutions

3.1.1 Properties of Optimal Driver Sizing Solutions

Theorem 1 *[27] Given the loading capacitance C_L, the number of stages k and the minimum gate capacitance C_g, the optimal stage ratio is $s = (\frac{C_L}{C_g})^{1/k}$.* □

3.1.2 Properties of Optimal Wire Sizing Solutions

The results in [16, 12] showed several interesting properties of optimal wiresizing solutions for performance optimization, including the *separability*, the *monotone property*, and the *dominance property*.

Theorem 2 *[16, 12]* **(Separability):** *If the width assignment of a path P originated from the source is given, the optimal width assignment for each subtree branching off from P can be carried out independently.* □

Definition 3 (Monotone Property): *Given a routing tree T, a wiresizing solution $\mathcal{W}$ on T is a monotone assignment if $w_E \geq w_{E'}$ for any pair of segments E and E' such that $E \in Ans(E')$.*

Theorem 3 *[16, 12] For any given tree T, there exists a monotone optimal width assignment $\mathcal{W}^*$.* □

Definition 4 (Dominance Property): *Given two wire width assignments $\mathcal{W}$ and $\mathcal{W}'$, $\mathcal{W}$* dominates *$\mathcal{W}'$ if for any segment E, the width assignment of E in $\mathcal{W}$ is greater than or equal to that of E in $\mathcal{W}'$.*

Definition 5 *Given a routing tree T, a wire width assignment $\mathcal{W}$ on T, and any particular segment $E \in T$, a* local refinement *on E is the operation to optimize the width of E based on the objective function in (3), without changing the width on the other segments specified by $\mathcal{W}$.*

Theorem 4 *[16, 12] Let $\mathcal{W}^*$ be an optimal width assignment. If a width assignment $\mathcal{W}$ dominates $\mathcal{W}^*$, then any local refinement of $\mathcal{W}$ still dominates $\mathcal{W}^*$. Similarly, if a width assignment $\mathcal{W}$ is dominated by $\mathcal{W}^*$, then any local refinement of $\mathcal{W}$ is dominated by $\mathcal{W}^*$.* □

Given a routing tree T with one or more critical sinks. Let R_d be the resistance of the driver driving the routing tree and $\mathcal{W}^*$ be the corresponding optimal wire width assignment. Consider another driver such that R'_d $(> R_d)$ is the resistance of the driver and $\mathcal{W}'^*$ is the corresponding optimal wire width assignment. [12] shows the following relation between driver sizing and wire sizing solutions.

Theorem 5 *[12]* **(DS/WS Relation):** *For any tree T with one or more critical sinks, $R_d < R'_d$ implies $\mathcal{W}^*$ dominates $\mathcal{W}'^*$.* □

This result reveals the relation between driver sizing and wiresizing, and it plays an important role in determining the lower and upper bounds of the optimal SDWS solutions. Starting from minimum-width wires, $\mathcal{W}_0$, Theorem 1 is applied to find the corresponding optimal driver sizing solution, denoted $\mathcal{D}_1$. Based on $\mathcal{D}_1$, the corresponding optimal wire sizing solution, denoted $\mathcal{W}_1$ is then computed. The process of driver sizing followed by wiresizing continued until there is no change in the wiresizing solution. This process computes the lower bound of the optimal SDWS solution. The upper bound can be obtained by starting with maximum-width wires.

We note that since the drivers are restricted to be at the source of the interconnect tree in the SDWS problem, a change in the driver sizing solution affects the wiresizing solution from the *upstream only* and a change in the wiresizing solution affects the driver sizes from the *downstream only*. However, in the case of the SBWS problem, a change in the buffer sizing solution affects the buffer and wire sizing solution of both the upstream and downstream. Similarly, a change in the wiresizing solution has both upstream and downstream effect on the buffer and wire sizing solution. A significant portion of this section will be spent on establishing the BS/WS relation between buffer and wire sizing solutions.

3.2 Properties of Optimal SBWS Solutions

The above results still holds for the upstream effect of buffer sizing and the downstream effect of wiresizing. However, to solve the SBWS problems, we need to establish several new properties as follows:

3.2.1 Properties of Optimal Buffer Sizing Solutions

Consider a buffer D_i with a parent buffer D_j. Assume that we have a wiresizing solution $\mathcal{W}$ and we know the size of D_j and the loading capacitance of the sinks that D_i is driving. We want to compute the optimal size of driver D_i. The variable

d_i appears in both $t_{T_i}(\mathcal{D},\mathcal{W})$ and $t_{T_j}(\mathcal{D},\mathcal{W})$. We rewrite the two equations as follows to highlight the relationship between the two equations:

$$\begin{aligned} t_{T_i}(\mathcal{D},\mathcal{W}) &= \sum_{N_i\in sink(T_i)} \lambda_i\cdot\mathcal{K}_{1i} + \mathcal{K}_1\cdot\sum_{E\in T_i} H(E)\cdot\frac{1}{w_E} + R_{min}\cdot C_d + \frac{R_{min}}{d_i}\cdot\sum_{u\in sink(T_i)} c_u^s + \\ &\quad \frac{R_{min}\cdot c_0}{d_i}\cdot\sum_{E\in T_i} w_E + \frac{R_{min}}{d_i}\cdot\sum_{E\in T_i} c_1 + \mathcal{K}_5\cdot\sum_{E,E'\in T_i,E\neq E'} F(E,E')\cdot\frac{w_{E'}}{w_E} + \\ &\quad \mathcal{K}_6\cdot\sum_{E,E'\in T_i,E\neq E'} F(E,E')\cdot\frac{1}{w_E} + \mathcal{K}_7\cdot\sum_{E\in T_i} G(E)\cdot\frac{1}{w_E} \end{aligned} \tag{14}$$

$$\begin{aligned} t_{T_j}(\mathcal{D},\mathcal{W}) &= \sum_{N_i\in sink(T_j)} \lambda_i\cdot\mathcal{K}_{1i} + \mathcal{K}_1\cdot\sum_{E\in T_j} H(E)\cdot\frac{1}{w_E} + R_{min}\cdot C_d + \frac{R_{min}}{d_j}\cdot\sum_{u\in sink(T_j),u\neq D_i} c_u^s + \\ &\quad \frac{R_{min}}{d_j}\cdot C_g\cdot d_i + \frac{R_{min}\cdot c_0}{d_j}\cdot\sum_{E\in T_j} w_E + \mathcal{K}_4 + \mathcal{K}_5\cdot\sum_{E,E'\in T_j,E\neq E'} F(E,E')\cdot\frac{w_{E'}}{w_E} + \\ &\quad \mathcal{K}_6\cdot\sum_{E,E'\in T_j,E\neq E'} F(E,E')\cdot\frac{1}{w_E} + \mathcal{K}_7\cdot\sum_{E\in T_j,D_i\notin sink(E)} G(E)\cdot\frac{1}{w_E} + \\ &\quad \mathcal{K}_7\cdot\sum_{E\in T_j,D_i\in sink(E)} H(E)\cdot\frac{1}{w_E}\cdot\left(\sum_{u\in sink(E),u\neq D_i} c_u^s + C_g\cdot d_i\right) \end{aligned} \tag{15}$$

Note that we make use of the facts that (i) $G(E) = H(E)\cdot\sum_{u\in sink(E)} c_u^s$, (ii) $\mathcal{K}_2$, $\mathcal{K}_3$ and $\mathcal{K}_4$ contain the terms d_i and d_j, and (iii) $\mathcal{K}_7 = r_0$ in the above equations. Taking differential of $t_T(\mathcal{D},\mathcal{W})$ with respect to d_i and we obtain:

$$\begin{aligned} \frac{\partial t_T(\mathcal{D},\mathcal{W})}{\partial d_i} &= \lambda_{D_i}\frac{\partial t_{T_i}(\mathcal{D},\mathcal{W})}{\partial d_i} + \lambda_{D_j}\frac{\partial t_{T_j}(\mathcal{D},\mathcal{W})}{\partial d_i} \\ &= \lambda_{D_i}\cdot\frac{-R_{min}}{d_i^2}\cdot\left(\sum_{u\in sink(T_i)} c_u^s + c_0\cdot\sum_{E\in T_i} w_E + \sum_{E\in T_i} c_1\right) + \\ &\quad \lambda_{D_j}\cdot C_g\cdot\left(\frac{R_{min}}{d_j} + r_0\cdot\sum_{E\in P(D_j,D_i)} H(E)\cdot\frac{1}{w_E}\right) \end{aligned} \tag{16}$$

Setting the differential to zero and solving for d_i, we have the following theorem:

Theorem 6 *Given net $\mathcal{N}_i$ and its parent net $\mathcal{N}_j$. Assuming a wiresizing solution $\mathcal{W}$ and that both the loading capacitances of the sinks in $\mathcal{N}_i$ and the size d_j of the driver of $\mathcal{N}_j$, namely D_j, are given, the optimal size for the buffer D_i can be computed as follows:*

$$d_i = \left(\frac{\lambda_{D_i}\cdot R_{min}\cdot\left(\sum_{u\in sink(T_i)} c_u^s + c_0\cdot\sum_{E\in T_i} w_E + \sum_{E\in T_i} c_1\right)}{\lambda_{D_j}\cdot C_g\cdot\left(\frac{R_{min}}{d_j} + r_0\cdot\sum_{E\in P(D_j,D_i)} H(E)\cdot\frac{1}{w_E}\right)}\right)^{1/2} \tag{17}$$

In a similar manner in which the local refinement operation for wiresizing is defined, we define the local refinement operation for buffer sizing as follows: the local optimal size of a buffer is computed using the expression in Eqn. (17) by keeping other buffer sizes and wire sizes intact. Note that the above expression assumed that the buffer size is continuous. However, we can also make the buffer size discrete (e.g., multiples of the minimum-size) as in the case of assuming a finite set of wire widths $\{W_1,\cdots,W_r\}$ for wiresizing.

By manipulating the terms in Eqn. (17), we obtain the following equality:

$$\lambda_{D_i} \cdot \frac{R_{min}}{d_i} \cdot \left(\sum_{u \in sink(T_i)} c_u^s \, + \, c_0 \cdot \sum_{E \in T_i} w_E \, + \, \sum_{E \in T_i} c_1 \right) = \lambda_{D_j} \cdot C_g \cdot d_i \cdot \left(\frac{R_{min}}{d_j} \, + \, r_0 \cdot \sum_{E \in P(D_j, D_i)} H(E) \cdot \frac{1}{w_E} \right) \qquad (18)$$

We observe that the bracketed sum of terms on the left hand side of the equality is the total load that driver D_i (with resistance $\frac{R_{min}}{d_i}$) has to drive whereas the bracketed sum of terms on the right hand side gives the total (weighted) resistance that drives the load due to driver D_i, i.e., $C_g \cdot d_i$. The wire resistance from D_j to D_i are weighted by $H(E)$ to consider the criticality of other sinks that branches off the D_j–D_i path.

This is a generalization of the results for driver sizing of a chain of cascaded drivers in Theorem 1. In case of SDWS, $\lambda_{D_i} = \lambda_{D_j}$ and $r_0 \cdot \sum_{E \in P(D_j, D_i)} H(E) \cdot \frac{1}{w_E} = 0$ since we assume that the wire resistance between cascaded drivers is negligible. The constant stage ratio can be derived from the above equality holds. Effectively, in the SBWS formulation, D_i treats the total weighted resistance on the right hand side as the resistance of an imaginary cascaded driver just before it.

Another observation that we can draw from the local optimal computation of d_i is that if d_j dominates the optimal size of driver D_j, denoted d_j^*, and the wiresizing solution $\mathcal{W}$ dominates the optimal wiresizing solutions, denoted $\mathcal{W}^*$, then the size of D_i after the local refinement step also dominates the optimal size of D_i, denoted d_i^*. We can generalize the above observation and prove the following:

Theorem 7 (WS/BS Relation) *Consider two wire sizing solutions $\mathcal{W}$ and $\mathcal{W}'$, if $\mathcal{W}$ dominates $\mathcal{W}'$, then the optimal buffer sizing solution $\mathcal{D}$ for $\mathcal{W}$ dominates the optimal buffer sizing solution $\mathcal{D}'$ for $\mathcal{W}'$.*

3.2.2 Properties of Optimal Buffer and Wire Sizing Solutions

In section 2, we have written the performance measure $t_T(\mathcal{D}, \mathcal{W})$ such that it is a weighted sum of the performance measures $t_{T_i}(\mathcal{D}, \mathcal{W})$ of the subnets. As a result of the decomposition, we observe that if we are given the buffer sizes, we can simply apply the optimal wiresizing technique in [16]. However, changing the wiresizing solution will affect the buffer sizing solution. On top of that, as the buffers change their sizes, they affect the wiresizing of the interconnect that they are driving and *also* the wiresizing of the interconnect driven by their parent drivers.

In the following, we will establish the BS/WS relation of the buffer sizing and wiresizing solutions. This relation is a generalization of the results obtained in [12]. In the following, we consider a unbuffered routing tree T with one or more critical sinks. Let R_d be the resistance of the driver driving the routing tree and $\mathcal{W}$ be the corresponding optimal wire width assignment. Let c_v^s be the loading capacitance of one of the critical sink of T, i.e. $\lambda_v > 0$. This sink is actually a buffer and it drives a downstream subnet. Consider a similar routing tree except that the loading capacitance of the critical sink at v is $c_v^{s\prime}$ ($< c_v^s$) and $\mathcal{W}'$ is the corresponding optimal wire width assignment. While Theorem 5 shows the relation between the driver resistances and wiresizing solutions, we want to investigate the relation between the loading capacitances and wiresizing solutions. We want to show that there *exists* a wiresizing solution $\mathcal{W}$ such that it dominates $\mathcal{W}'$ if $c_v^s > c_v^{s\prime}$.

In the following, we deviate from the notation $t_T(\mathcal{D}, \mathcal{W})$ and use $t(\mathcal{W}, c_v^s)$ to denote the performance measure of the first routing tree and $t(\mathcal{W}', c_v^{s\prime})$ for the performance measure of the second routing tree. The new notation highlights that the variables are the sink capacitance c_v^s (the input capacitance of a buffer at v which changes due to buffer sizing), and the wire widths. We rewrite $t(\mathcal{W}, c_v^s)$ to highlight the variables c_v^s and w_E as follows:

$$\begin{aligned} t(\mathcal{W}, c_v^s) \quad = \quad & \sum_{N_i \in sink(T)} \lambda_{N_i} \cdot \mathcal{K}_{1i} \, + \, \mathcal{K}_1 \cdot \sum_{E \in T} H(E) \cdot \frac{1}{w_E} \, + \, R_{min} \cdot C_d \, + \, R_d \cdot \sum_{u \in sink(T), u \neq v} c_u^s \, + \\ & R_d \cdot c_v^s \, + \, \mathcal{K}_3 \cdot \sum_{E \in T} w_E \, + \, \mathcal{K}_4 \, + \, \mathcal{K}_5 \cdot \sum_{E, E' \in T, E \neq E'} F(E, E') \cdot \frac{w_{E'}}{w_E} \, + \end{aligned}$$

$$\mathcal{K}_6 \cdot \sum_{E,E' \in T, E \neq E'} F(E,E') \cdot \frac{1}{w_E} + \mathcal{K}_7 \cdot \sum_{E \in T} H(E) \cdot \frac{1}{w_E} \cdot \sum_{u \in Sink(E), u \neq v} c_u^s +$$
$$\mathcal{K}_7 \cdot \sum_{E \in P(N_+,v)} H(E) \cdot \frac{1}{w_E} \cdot c_v^s \tag{19}$$

In the following, proofs of the lemmas are omitted and are presented in the Appendix.

Lemma 1 *If $c_v^s > c_v^{s\prime}$, then*

$$\sum_{E \in P(N_+,v)} H(E) \cdot \left(\frac{1}{w_E} - \frac{1}{w'_E}\right) \leq 0 \tag{20}$$

□

Lemma 2 *Consider an edge E in the routing tree. If $w_E < w'_E$, then*

$$\sum_{E' \in Ans(E)} H(E') \cdot \left(\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}\right) + \frac{H(E)}{w_E \cdot w'_E} \sum_{E' \in Des(E)} (w'_{E'} - w_{E'}) > 0 \tag{21}$$

□

Lemma 3 *Consider the edge E directly above the loading capacitance at v. If $c_v^s > c_v^{s\prime}$, then $w_E \geq w'_E$.* □

Lemma 4 *Consider the edge E directly above the loading capacitance at v. If $c_v^s > c_v^{s\prime}$, then there exists an optimal wiresizing solution $\mathcal{W}$ such that it dominates $\mathcal{W}'$ for all edges in $Des(E)$.* □

The above two lemmas show that we can find an optimal wiresizing solution $\mathcal{W}$ such that the wire width assignment of E and its descendant edges, if any, dominates the wire width assignment $\mathcal{W}'$ of $\{E\} \cup Des(E)$ if $c_v^s > c_v^{s\prime}$. We can now establish the relationship between wiresizing solution and the capacitive load (CL) as follows.

Theorem 8 (WS/CL Relationship) *If $c_v^s > c_v^{s\prime}$, then there exists a wiresizing solution $\mathcal{W}$ such that for all edges $E \in T$, $w_E \geq w'_E$.*

Proof: Let T_E denote single-stem tree with E as the root stem [16]. Note that $E \in T_E$ under this definition. Due to the separability of the optimal wiresizing problem, we only need to consider the single-stem tree that contains the loading capacitance (c_v^s or $c_v^{s\prime}$). We can also conclude from the separability property that edges in other single-stem trees of the routing tree T will have identical wire width assignment. Therefore, in the following, we assume that T is a single-stem tree.

We will prove by induction that if $c_v^s > c_v^{s\prime}$, then for every edge $E \in P(N_+,v)$, all edges $E' \in T_E$ have widths $w_{E'} \geq w'_{E'}$. We have shown in Lemma 3 that the edge just before the loading capacitance has this property. Suppose this is true for all edges from E to v along the $P(N_+,v)$ path. Let the parent node of E be v'.

Let $C_{v'}$ be the total capacitance at v' due to wire capacitances and sink capacitances, including c_v^s, in T_E for wiresizing solution $\mathcal{W}$. Similarly, let $C'_{v'}$ be the total capacitance at v' due to wire capacitances and sink capacitances, including $c_v^{s\prime}$, in T_E for wiresizing solution $\mathcal{W}'$. Clearly, $C_{v'}$ is strictly larger than the $C'_{v'}$. Now, consider a new routing tree $T' = T - T_E$ with, either a loading capacitance of $C_{v'}$ or $C'_{v'}$ at the node v'. By applying Lemmas 3 and 4, we can show that E', the parent edge of E, also have the same property that $w_{E'} \geq w'_{E'}$. □

Combining the WS/BS and WS/CL relationships, we can deduce the following relation between buffer sizing and wire sizing solutions:

Theorem 9 (BS/WS Relation) *Consider two buffer sizing solution D and D', if D dominates D', then there exists an optimal wiresizing solution $\mathcal{W}$ for D and an optimal wiresizing solution $\mathcal{W}'$ for D' such that $\mathcal{W}$ dominates $\mathcal{W}'$.*

Analogous to the the DS/WS relation for the SDWS problem, the BS/WS relation plays an important role in determining the lower and upper bounds of the optimal SBWS solutions in the next section.

4 Optimal SBWS Algorithms

4.1 Lower and Upper Bound Computation for Optimal SBWS Solutions

We first present an algorithm called the SBWS-LU-Bound Algorithm to compute the upper and lower bounds of an optimal SBWS solution for a given buffered interconnect tree: Starting with an initial wire width assignment (say, all wires have the minimum width) and an initial buffer sizing solution (say, all buffers have the minimum size), we perform an arbitrary number of local refinement operations on the sizes of the intermediate buffers based on Theorem 6. Note that we may perform local refinement operations until there is no further changes in the buffer sizes, i.e. the buffer sizes converge for the current set of wiresizing solution (convergence is guaranteed since each local refinement reduces the performance measure), or we have reached a pre-determined upper limit of the number of local refinement operations. Now, we perform a delay optimal wiresizing algorithm [16] on the routing tree of each subnet based on the resistance of the driver of the subnet and the sink capacitances of downstream buffers or loading pins. A new iteration of local refinement operations for buffers are carried out to yield a new buffer sizing solutions. Then, the optimal wiresizing solution will be computed again based on the new buffer sizing solution. Let an iteration be a series of local refinement operations of buffers followed by applying optimal wiresizing algorithm on each subnet. The process is repeated until the wire width do not change in consecutive iterations. The algorithm is described formally in Figure 4.

Let $\mathcal{W}_0$ be the initial minimum width assignment and $\mathcal{W}_i$ denote the optimal wire width assignment obtained in iteration i. Similarly, let $\mathcal{D}_0$ be the initial buffer sizes and $\mathcal{D}_i$ denote the final buffer sizes at the end of iteration i. It is obvious that in the first iteration, local refinement of buffer size will only cause buffer sizes to revise upward since they are all of minimum size initially. Therefore, $\mathcal{D}_1$ dominates $\mathcal{D}_0$. Similarly, the wire widths in $\mathcal{W}_1$, which is computed based on $\mathcal{D}_1$ dominates $\mathcal{W}_0$ since $\mathcal{W}_0$ is the minimum-width assignment.

In the next iteration of computing $\mathcal{D}_2$ based on $\mathcal{W}_1$, by Theorem 6, the size of any buffer, say D_j, will only increase further if there is any change. By similar argument, other buffer sizes will also be revised upward if there is any change. Subsequently, by the BS/WS relation Theorem, the optimal wiresizing solution for $\mathcal{D}_1$ will also be revised upwards. In general, we can show by mathematical induction that $\mathcal{D}_{i+1}$ dominates $\mathcal{D}_i$ and $\mathcal{W}_{i+1}$ dominates $\mathcal{W}_i$ (according to Theorem 9 on BS/WS relation). Hence, the algorithm will terminate (since there is an upper bound on the maximum wire width). We can also observe that the optimal SBWS solution, denoted $(\mathcal{D}^*, \mathcal{W}^*)$, dominates $(\mathcal{D}_0, \mathcal{W}_0)$. Therefore, applying Theorems 7 and 9 and by induction, it will dominates all $(\mathcal{D}_i, \mathcal{W}_i)$ for all i. In other words, the lower bound is computed correctly. Similarly, we can start with a maximum wire width assignment and maximum buffer size assignment and compute the upper bound of buffers and wire sizes of the optimal SBWS solution.

The most computationally expensive operation in the algorithm is the wiresizing operations which run in $O(n^3 \cdot r)$ [16]. However, with the use of bundled refinement operation introduced in [9], the wiresizing algorithm runs much faster than $O(n^3 \cdot r)$. Experimental results show that the algorithm terminates after three or four iterations in most cases. In addition, the upper and lower bounds meet for most instances, which implies that the optimal SBWS solution is obtained. Note that the upper and lower bounds include both the buffer sizes and wire widths.

SBWS LU-Bound Algorithm

$\mathcal{W}_l \leftarrow Min\_Wire\_Width$; $\mathcal{D}_l \leftarrow Min\_Buffer\_Size$;

while `true`

Compute new buffer size by Theorem 6: $\mathcal{D}_l \leftarrow$ iterations of $buffer\_local\_refinement(\mathcal{D}_l, \mathcal{W}_l)$;

Compute new wire size by [16]: $\mathcal{W} \leftarrow Optimal\_Wiresizing(\mathcal{D}_l, \mathcal{W}_l)$;

if $\mathcal{W} > \mathcal{W}_l$ **then**

$\mathcal{W}_l \leftarrow \mathcal{W}$

else break;

end while

Output $(\mathcal{D}_l, \mathcal{W}_l)$ as the lower bound of the optimal SBWS solution;

$\mathcal{W}_u \leftarrow Max\_Wire\_Width$; $\mathcal{D}_u \leftarrow Max\_Buffer\_Size$;

while `true`

Compute new buffer size by Theorem 6: $\mathcal{D}_u \leftarrow$ iterations of $buffer\_local\_refinement(\mathcal{D}_u, \mathcal{W}_u)$;

Compute new wire size by [16]: $\mathcal{W} \leftarrow Optimal\_Wiresizing(\mathcal{D}_u, \mathcal{W}_u)$;

if $\mathcal{W} < \mathcal{W}_u$ **then**

$\mathcal{W}_u \leftarrow \mathcal{W}$

else break;

end while

Output $(\mathcal{D}_u, \mathcal{W}_u)$ as the upper bound of the optimal SBWS solution;

Figure 4: The SBWS-LU-Bound algorithm for computing lower and upper bounds of optimal SBWS solution for a given buffered tree T.

4.2 Optimal Algorithms for the SBWS Problem

In the case where the bounds computed by the SBWS-LU-Bound Algorithm do not meet, we can enumerate, with pruning, the solution space using the dynamic programming technique. For each buffer, if its bounds do not meet, we obtain a set of discrete buffer sizes within the bounds for the buffer.[1] Let the smallest grid-size possible be λ-unit (not to be confused with the criticality value λ_{D_i} or λ_{N_i}). Then, the buffer sizes for buffer D_i is of the form form $\{j \cdot \lambda, (j+1) \cdot \lambda, \cdots, k \cdot \lambda\}$ where $j \cdot \lambda \geq \mathcal{D}_l(D_i)$ and $k \cdot \lambda \leq \mathcal{D}_u(D_i)$, where $\mathcal{D}_l(D_i)$ and $\mathcal{D}_u(D_i)$ are the lower and upper bounds for D_i computed by the SBWS-LU-Bound Algorithm.

We present two approaches to optimally solve the SBWS problem. Note that we can actually apply either algorithm to find the optimal SBWS solution without going through the bound computation process if both buffer sizes and wire widths are discrete. However, in both cases, the original solution space is computationally too huge to be considered. The bound computation allows us to reduce the solution space significantly by narrowing the bounds. This speeds up the Optimal SBWS Algorithms tremendously. In fact, the bounds computed by the SBWS-LU-Bound Algorithm are very tight and they meet in most of the cases.

[1] As mentioned before, the SBWS-LU-Bound Algorithm can also be applied for the case where the buffer sizes are discrete and not continuous. In this case, the set of discrete sizes for each buffer are those sizes within the bounds computed by the SBWS-LU-Bound Algorithm. For the case of continuous buffer size, we can use the set of discrete buffer sizes based on the restriction imposed by the fabrication technology.

4.2.1 Optimal SBWS Algorithm-1

This algorithm makes use of the optimal wiresizing algorithm (OWSA) in [16]. We define a buffer-rooted tree to be a *buffered* tree rooted at a buffer. We use $BRT(D_i)$ to denote the buffer-rooted tree with buffer D_i being the root. We assume a set of possible wire widths $W = \{W_1, W_2, \cdots, W_r\}$ for each wire and a set of possible buffer sizes $D = \{B_1, B_2, \cdots, B_d\}$ for each buffer.[2] Consider a buffer D_i with no child buffer. After assigning a buffer size from $\{B_1, B_2, \cdots, B_d\}$ to D_i, we can determine the optimal wiresizing solution to the unbuffered subtree T_i by applying OWSA from [16]. Let $(\mathcal{D}, \mathcal{W})^*_{BRT(D_i)}(B_j)$ denote the optimal SBWS solution for the $BRT(D_i)$ when D_i is assigned with the size B_j and $t^*_{BRT(D_i)}(B_j)$ be the corresponding performance measure.

Optimal SBWS Algorithm-1

for each buffer D_i in bottom-up order
  for each size $B_l \in \{B_1, B_2, \cdots, B_d\}$
    $t^*_{BRT(D_i)}(B_l) \leftarrow \infty$;
    for each monotone wiresizing solution $\mathcal{W}_{T_i}$ (enumerated by OWSA in [16])
      $(\mathcal{D}, \mathcal{W}) \leftarrow (\{B_l\}, \mathcal{W}_{T_i})$;
      for each child buffer D_j
        $t^*_{P(D_i,D_j),BRT(D_j)} \leftarrow \infty$;
        for each size $B_k \in \{B_1, B_2, \cdots, B_d\}$
          $(\mathcal{D}', \mathcal{W}') \leftarrow (\mathcal{D}, \mathcal{W})^*_{BRT(D_j)}(B_k) \cup (\{B_l\}, \mathcal{W}_{T_i})$;
          if $t_{P(D_i,D_j),BRT(D_j)}(\mathcal{D}', \mathcal{W}') < t^*_{P(D_i,D_j),BRT(D_j)}$ **then**
            $d_j \leftarrow B_k$;
            $t^*_{P(D_i,D_j),BRT(D_j)} \leftarrow t_{P(D_i,D_j),BRT(D_j)}(\mathcal{D}', \mathcal{W}')$;
          end if
        end for
        $(\mathcal{D}, \mathcal{W}) \leftarrow (\mathcal{D}, \mathcal{W}) \cup (\mathcal{D}, \mathcal{W})^*_{BRT(D_j)}(d_j)$;
      end for
      if $t_{BRT(D_i)}(\mathcal{D}, \mathcal{W}) < t^*_{BRT(D_i)}(B_l)$ **then**
        $t^*_{BRT(D_i)}(B_l) \leftarrow t_{BRT(D_i)}(\mathcal{D}, \mathcal{W})$;
        $(\mathcal{D}, \mathcal{W})^*_{BRT(D_i)}(B_l) \leftarrow (\mathcal{D}, \mathcal{W})$;
      end if
    end for
  end for
end for

Figure 5: The Optimal SBWS Algorithm-1 for a buffered tree T.

Consider buffer D_i with child buffer D_j. Suppose we know $(\mathcal{D}, \mathcal{W})^*_{BRT(D_j)}(B_k)$ for any B_k. Consider D_i with a buffer size B_l. Due to Theorem 2 on separability, if we know the wire width assignment for the wires along the path from D_i to D_j, we can select the optimal d_j for D_j from $\{B_1, B_2, \cdots, B_d\}$ and the corresponding $(\mathcal{D}, \mathcal{W})^*_{BRT(D_j)}(d_j)$ *without* considering

[2] We use uniform bounds instead of the bounds computed by the SBWS-LU-Bound Algorithm to simplify the analysis of the time complexity. In practice, we use the bounds computed by the SBWS-LU-Bound Algorithm in the Optimal SBWS Algorithms.

other children buffers of D_i. Note that for each parent-child buffer pair D_i-D_j, we do not have to consider all possible wire width assignment for edges along the path $P(D_i, D_j)$ from D_i to D_j. Due to the monotone property (Theorem 3 [16, 12]) for the optimal wiresizing solution of each subtree T_i of the buffered tree T, we only have to consider all monotone wiresizing solutions in T_i. The OWSA algorithm in [16] gives details on how monotone wiresizing solutions can be enumerated. The *Optimal SBWS Algorithm-1* is given in Figure 5. In the algorithm, $t_{P(D_i,D_j),BRT(D_j)}(\mathcal{D}, \mathcal{W})$ corresponds to the performance measure of the subtree consisting of the path $P(D_i, D_j)$ and $BRT(D_j)$ according to Theorem 2 on separability in [16, 12].

Within each subtree, the total number of monotone wire width assignments is bounded by $O(n^{|W|})$, where n is the number of edges in the subtree [16]. For each parent-child buffer pair in the subtree, we only have to consider $|D|^2$ combinations of buffer sizes due to the separability of buffered tree. There are altogether $O(b)$ parent-child buffer pairs for b buffers. Therefore, the time complexity of the Optimal SBWS Algorithm-1 is $O(b \cdot |D|^2 \cdot n^{|W|})$.

4.2.2 Optimal SBWS Algorithm-2

We use a technique similar to that in [25, 21] to find the optimal SBWS solution. Note that in [25, 21], the objective is to minimize the maximum Elmore delay, whereas, our objective in this paper is to minimize the weighted delay. For an edge e in the buffered clock tree, we can define $\lambda_e = \sum_{N_i \in sink(e)} \lambda_{N_i}$ (much alike the definition of λ_{D_i}). [3] We define $t_e^{N_i}(\mathcal{D}, \mathcal{W})$ to be the Elmore delay from the upstream endpoint of e to sink N_i where N_i is a sink rooted at the downstream endpoint of e. Then, we define $t_e(\mathcal{D}, \mathcal{W}) = \sum_{N_i \in sink(v)} \lambda_{N_i} \cdot t_e^{N_i}(\mathcal{D}, \mathcal{W})$. Let T_e denote the subtree (within the unbuffered T_i containing e and not the buffered tree T) rooted at the downstream endpoint of e. Note that T_e is not the single-stem tree in the previous section and it does not contain the edge e. Let $CL_{T_e}(\mathcal{D}, \mathcal{W})$ be the total capacitance seen at the downstream endpoint of e. For a given $(\mathcal{D}, \mathcal{W})$, we note that $t_e(\mathcal{D}, \mathcal{W})$ can be computed in a bottom-up fashion as follows:

$$t_e(\mathcal{D}, \mathcal{W}) = \lambda_e \cdot r_e \cdot \left(\frac{c_e}{2} + CL_{T_e}(\mathcal{D}, \mathcal{W})\right) + \sum_{e' \in Child(e)} t_{e'}(\mathcal{D}, \mathcal{W}) \tag{22}$$

where $Child(e)$ contains all child edges of e. Note that if e is a driver edge, then r_e is the driver resistance and c_e is twice the diffusion capacitance of the driver. If e is an wire segment, then r_e is the wire resistance and c_e is the wire capacitance.

For each edge e and a buffer and wiresizing solution, we associate with e a (c,t) pair where c is the total capacitance at the upstream endpoint of e (i.e., $c = CL_{T_e}(\mathcal{D}, \mathcal{W}) + c_e$) and t is the performance measure given in Eqn. (22). Consider two buffer and wire sizing solutions, then we have (c,t) and (c', t') at e. We observe that if $c \le c'$ and $t' > t$, then (c', t') is sub-optimal. In other words, the corresponding buffer and wire sizing solution of (c', t') is *redundant* and can be ignored. Therefore, the *Optimal SBWS Algorithm-2* computes for each edge e in the buffered tree T, a set of irredundant (c,t) pairs for e (or correspondingly, a set of buffer and wire sizing solutions for the subtree $T_e + \{e\}$). At the end of the bottom-up computation of (c,t) pairs, the root may have a set of solutions. The optimal solution is achieved by choosing the (c,t) pair with the smallest t at the root. Figure 6 shows how the set of (c,t) pairs is computed for an edge e. In the computation, we use $\{s_e^l, \cdots, s_e^u\}$ to denote the set of sizes for e, where s_e^l is the smallest size of e and s_e^u denotes the largest size of e. The bounds are computed by the SBWS-LU-Bound Algorithm.

Next, we want to analyze the complexity of the Optimal SBWS Algorithm-2. Suppose each wire segment has at most W_R possible choices of wire widths (though the set of wire widths might be different) and each buffer has at most D_R possible buffer sizes (again, the set of buffer sizes might be different). Note that W_R and D_R are derived from the *reduced*

[3] Note that we distinguish E from e. E is a wire segment in the routing tree and we use e to denote both a wire segment in the routing tree and the RC-segment representing driver or buffer; see the discussion on the edge D–N_+ in Section 2.1.2.

(c,t)-Pair-Computation

```
Let e' be any child edge of e
S_1 ← set of (c,t) pairs of e
Mark e' visited;
for each unvisited child edge e' of e
    S_2 ← set of (c,t) pairs of e'; S ← ∅;
    /* S_1 and S_2 are ordered in increasing order of c */
    for i ← 1 to |S_1| and j ← 1 to |S_2|
        c = c_i + c_j; t = t_i + t_j; S ← S ∪ {(c,t)};
    end for
    S_1 ← irredundant (c,t) in S;
end for
S ← ∅;
for each size s of e in {s_e^l, ···, s_e^u}
    Calculate r_e and c_e of e accordingly /* whether e is a wire or buffer */;
    for i ← 1 to |S_1|
        t ← t_i + λ_e · r_e · (c_e/2 + c_i)
        if e is a wire segment then
            c ← c + c_e;
        else /* e is a buffer/driver */
            c ← C_g · s /* input gate capacitance */
        end if
        S ← S ∪ {(c,t)};
    end for
end for
return irredundant (c,t) pairs in S;
```

Figure 6: An algorithm to compute the set of irredundant (c,t) pairs of e.

sets of wire widths for each wire and buffer sizes for each buffer, respectively, after bound computation using the SBWS-LU-Bound Algorithm. Again, let the smallest grid-size possible be λ-unit (not to be confused with the criticality value λ_e). Then, it is not unrealistic to assume that $\{s_e^l, \cdots, s_e^u\}$ for edge e is actually of the form $\{j\cdot\lambda, (j+1)\cdot\lambda, \cdots, (j+W_R-1)\cdot\lambda\}$ or $\{j\cdot\lambda, (j+1)\cdot\lambda, \cdots, (j+D_R-1)\cdot\lambda\}$ depending on whether e is a wire or a buffer.

From the (c,t)-Pair-Computation Algorithm, we observe that for each buffer, there are at most D_R irredundant (c,t) pairs. For any wire edge e, the total capacitance at e (including e) consists of wire capacitances in T_e and capacitances of sinks (which may be buffers) in T_e. Now, we want to bound the number of irredundant (c,t) pairs of a wire edge. Consider a subnet T_i with n wire edges and m sinks. For any wire edge e, total wire capacitance rooted at e (including e) has at most $n\cdot W_R$ distinct values and the total sink capacitance rooted at e has at most $m\cdot D_R$ distinct values. Therefore, the total capacitance rooted at e (including e) has at most $m\cdot n\cdot W_R\cdot D_R$ distinct values. In other words, there are at most $m\cdot n\cdot W_R\cdot D_R$ irredundant (c,t) pairs at e.

The most computationally intensive process in the Optimal SBWS Algorithm-2 is to combine two sets S_1 and S_2 of

(c,t) pairs in the (c,t)-Pair-Computation procedure and return the set of irredundant (c,t) pairs. A straight forward method to compute the set of irredundant (c,t) pairs is to sort the pairs according to their c-value and then go through the sorted list once to discard redundant (c,t) pairs. Combining two sets S_1 and S_2 and then sorting the combined set requires $O((m \cdot n \cdot W_R \cdot D_R)^2 \cdot \log(m \cdot n \cdot W_R \cdot D_R))$. For m wire edges, there are $m-1$ such combinations. Therefore, the Optimal SBWS Algorithm-2 runs in polynomial time of $O(n \cdot (m \cdot n \cdot W_R \cdot D_R)^2 \cdot \log(m \cdot n \cdot W_R \cdot D_R))$. Note that the time complexity of the two optimal algorithms are analyzed based on different assumptions regarding the sets of possible buffer sizes and wire widths; we use uniform and non-uniform bounds in the analyses of Optimal SBWS Algorithm-1 and Algorithm-2, respectively. In practice, we use the non-uniform bounds computed by the SBWS-LU-Bound algorithm in both optimal algorithms.

5 Experimental Results

5.1 Results on IF Transceiver Chip

We apply the SBWS algorithms to the clock nets in a spread spectrum IF transceiver chip developed for wireless multimedia information systems at UCLA [7]. The integrated single-chip transceiver, which meets the stringent requirements in high-capacity wireless communication systems due to its code division multiple access capability, is designed for integration into portable products such as notebook computers. The chip has a die size of $7.9 \times 10.0 mm^2$, is designed under $1.2 \mu m$ 2-level metal SCMOS technology, and has a total power dissipation of $1.1W$. The chip is shown in Figure 7. There are two clock nets, namely, `CLK` and `DCLK`, in this design, and they are routed interactively with the Flint place-and-route tool from Berkeley. Although the clock skew within each net is important, the designers are more concerned with the inter-clock skew between `CLK` and `DCLK` among the registers in the upper and lower `Integrate-Dump-1` blocks in Figure 7. Both nets contain registers in the `Integrate-Dump-1` blocks.

Both clock signals originate from the `Clock-Generator` block and each clock source is driven by a chain of 4 cascaded inverters. Each clock signal passes through a buffer consisting of a chain of 4 cascaded inverters before arriving at registers in the two blocks of `Integrate-Dump-1`. Both the clock source drivers and buffers are designed manually. `DCLK` operates at $12.7MHz$ and `CLK` operates at $50.8MHz$. Figure 8 shows the ideal clock signals of `DCLK` and `CLK`, with a computation time of $9.8ns$ available for the data path in `Integrate-Dump-1`. However, simulation shows that the worst skew occurs between the maximum `DCLK` delay and the minimum `CLK` delay within the upper `Integrate-Dump-1` block. The maximum `DCLK` delay is $4.55ns$ and the minimum `CLK` delay is $4.05ns$, resulting in a inter-clock skew of $500ps$. The `DCLK` and `CLK` nets have a power dissipation of $0.0259W$ and $0.2947W$, respectively.

We apply the SBWS algorithm to speed up the `DCLK` net.[4] To do so, we assign to each sink (or register) a weight using the following assignment scheme: each sink gets a unnormalized weight which is simply its delay obtained from HSPICE simulation. The weights are then normalized and the SBWS algorithm is applied. Although we have no guarantee that this is a good weight assignment scheme, simulation result shows that it serves the purpose of reducing the inter-clock skew. After optimization, the inter-clock skew is still due to the maximum `DCLK` delay and the minimum `CLK` delay of registers in the upper `Integrate-Dump-1` block. While the minimum `CLK` delay remains unchanged, the maximum `DCLK` delay is reduced to $4.19ns$, translating to a factor of $3.5X$ skew reduction from $500ps$ to $140ps$. A more significant achievement of the SBWS algorithm is that the power dissipation of `DCLK` is reduced by a factor of $2.6X$ to $0.0097W$ after the optimization.

On the other hand, if we do not care about the inter-clock skew, we can optimize both nets separately. For example,

[4] In this experiment, the SBWS-LU-Bound Algorithm alone is able to output the optimal solutions since the bounds computed meet.

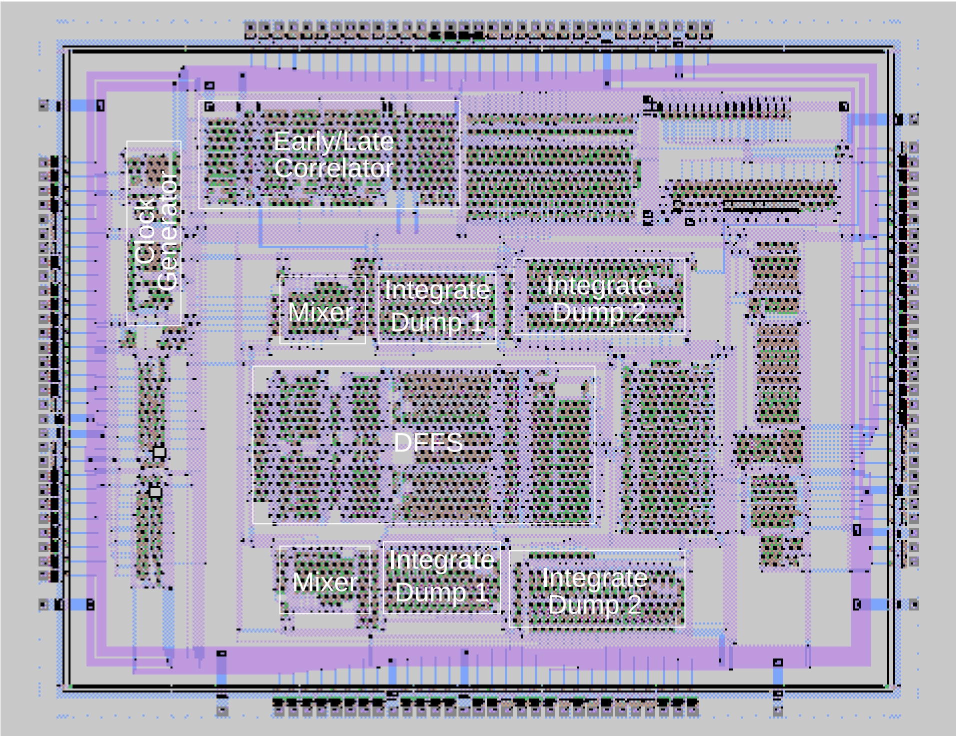

Figure 7: The 12.7 Mchip/s all-digital BPSK direct sequence spread-spectrum IF transceiver chip in $1.2\mu m$.

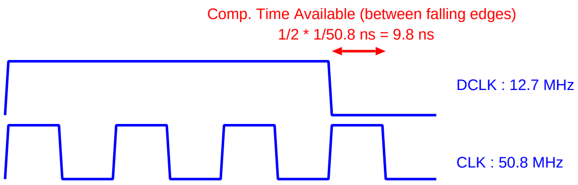

Figure 8: The ideal clock signals of DCLK and CLK.

we also apply the SBWS algorithm to optimize the CLK net, assuming that we only care about the skew within the CLK net. The maximum skew of the original CLK is $2018.7ps$ whereas the maximum skew of the optimized net is $1746.4ps$. Besides, the skew within each individual block of the CLK net remains at about the same order as shown in Table 1. Again, significant power reduction is achieved; the power dissipation of the optimized CLK net is reduced by a factor of $2.3X$ to $0.1272W$. The detailed results of the optimized DCLK net is also given in Table 2. Also note that all rise and fall times of the clock signals are within 5% of the CLK period. In fact, after SBWS optimization, the clock signals are sharper with smaller rise/fall times.

5.2 Application of Weighted-Delay Formulation

We also conduct experiments to evaluate the speed-up of run-time due to the lower and upper bound computation in Section 4.1 and the quality of the wiresizing solutions obtained by assigning appropriate weights to sinks by iterative Lagrangian relaxation [6] to minimize maximum delay, as compared to the solutions obtained by [21]. Table 3 reports the run-time and the maximum delay by HSPICE simulations of six routed nets in an Intel microprocessor design [9]. [5] The experimental

[5] These are actually multi-source nets. We randomly assign one pin to be the source driver.

Block	Original Design			Optimized Design		
	Max-Delay (ns)	Min-Delay (ns)	Skew (ps)	Max-Delay (ns)	Min-Delay (ns)	Skew (ps)
`Early/Late-Correlator`	3.9057	3.8843	21.4	3.4610	3.4349	26.1
Upper `Mixer`	4.1514	4.1173	34.1	3.7575	3.7266	30.9
Upper `Integrate-Dump-1`	4.0523	4.0450	7.3	3.6264	3.6082	18.2
Lower `Mixer`	4.1650	4.1310	34.0	3.7894	3.7590	30.4
Lower `Integrate-Dump-1`	4.0657	4.0581	7.6	3.6597	3.6429	16.8
`DDFS`	5.9030	5.5637	339.3	4.9910	5.1813	190.3
`CLK`	5.9030	3.8843	2018.7	5.1813	3.4349	1746.4
`CLK` Power Dissipation (W)	Original Power: 0.2947			Optimized Power: 0.1272		

Table 1: Comparison of skews and power dissipation of the manually layout `CLK` net with the optimized `CLK` net

Block	Original Design			Optimized Design		
	Max-Delay (ns)	Min-Delay (ns)	Skew (ps)	Max-Delay (ns)	Min-Delay (ns)	Skew (ps)
Upper `Integrate-Dump-1`	4.5553	4.5489	6.4	4.1897	4.1756	14.1
Upper `Integrate-Dump-2`	4.5630	4.5502	12.8	4.3088	4.2852	23.6
Lower `Integrate-Dump-1`	4.4856	4.4921	6.5	4.0979	4.0841	13.8
Lower `Integrate-Dump-2`	4.6050	4.6183	13.3	4.2257	4.2021	23.6
`DCLK`	4.4856	4.6183	132.7	4.3088	4.1756	133.2
`DCLK` Power Dissipation (W)	Original Power: 0.0259			Optimized Power: 0.0097		

Table 2: Comparison of skews and power dissipation of the manually layout `DCLK` net with the optimized `DCLK` net

results show that comparable maximum delay is achieved by our method with significantly shorter run-time. Note that the wires in the six nets are divided into $100\mu m$ segments before we apply the two algorithms. Finer division of wires leads to prohibitively large memory requirement for the algorithm in [21]. The SBWS algorithm with iterative Lagrangian relaxation does not have such problem. Further study will be conducted to consider buffer sizing as well.

Net	Optimal Algorithm in [21]		SBWS with Lagrange-Multiplier	
	Delay (ns)	Run-Time (s)	Delay (ns)	Run-Time (s)
net1	0.17145	3.729	0.17145	0.114
net2	0.23557	13.903	0.23562	0.067
net3	0.35121	207.194	0.35121	3.924
net4	0.39744	312.854	0.39374	0.266
net5	0.42730	253.091	0.45300	0.172
net6	0.95751	8966.805	0.95758	19.465

Table 3: Comparison between wiresizing solutions obtained by [21] and the Lagrangian-Based approach for minimizing maximum delay. The wires in the nets are divided into $100\mu m$ segments before we apply the two algorithms.

6 Conclusion and Future Work

The major contribution of our work is to establish the BS/WS relation between buffer sizing and wire sizing solution, which leads to a polynomial time algorithm to compute the lower and upper bounds of the optimal SBWS solutions and efficient algorithms to compute the optimal SBWS solutions. The results in this paper have shown convincingly that simultaneous buffer and wire sizing can lead to significant reduction in the power dissipation, while achieving better clock delays and clock signals with sharper rise and fall edges.

Ideally, our goal is to develop a practical clock routing algorithm which carries out simultaneous topology generation, buffer insertion and buffer and wire sizing to achieve bounded skew with minimum power dissipation under various layout constraints. We adopt a divide-and-conquer approach towards solving this very difficult problem. Together with the results on bounded-skew tree routing where the clock tree has a maximum skew no larger than the specified skew bound [13, 20, 11], the results presented in this paper and [12] will be the *enabling* tools for the aforementioned practical clock routing problem.

Acknowledgements

The authors would like to thank Dr. Charles Chien and Prof. Rajeev Jain at UCLA EE Department for providing the transceiver chip, and Mr. Takumi Okamoto and Mr. Lei He for their helpful discussions.

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 Routing", *Proc. IEEE Int'l Symp. on Circuits and Systems*, May 1993, pp. 1869-1872.

[2] H. B. Bakoglu, *Circuits, Interconnections, and Packaging for VLSI*, Addison-Wesley, 1990.

[3] K. D. Boese and A. B. Kahng, "Zero-Skew Clock Routing Trees With Minimum Wirelength", *Proc. IEEE 5th Int'l ASIC Conf.*, Rochester, September 1992, pp. 1.1.1 - 1.1.5.

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

[5] T.-H. Chao, Y.-C. Hsu and J.-M. Ho, "Zero Skew Clock Net Routing," *Proc. ACM/IEEE Design Automation Conf.*, 1992, pp. 518-523.

[6] C.-P. Chen, Y.-W. Chang and D. F. Wong, "Fast Performance-Driven Optimization for Buffered Clock Trees Based on Lagrangian Relaxation," *Proc. ACM/IEEE Design Automation Conf.*, 1996, pp. 405–408.

[7] C. Chien, P. Yang, E. Cohen, R. Jain, and H. Samueli, "A 12.7Mchip/s All-Digital BPSK Direct Sequence Spread-Spectrum IF Transceiver in 1.2μm CMOS," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 1994, pp. 30–31.

[8] J. P. Cohoon and L. J. Randall, "Critical Net Routing", *Proc. IEEE Int'l. Conf. on Computer Design*, 1991, pp. 174-177.

[9] J. Cong and L. He, "Optimal Wiresizing for Interconnects with Multiple Sources," *Int'l Conf. on Computer-Aided Design*, Nov. 1995, pp. 568–574.

[10] J. Cong, A. B. Kahng, G. Robins, M. Sarrafzadeh, and C. K. Wong, "Provably Good Performance-Driven Global Routing", *IEEE Trans. on CAD*, 11(6), June 1992, pp. 739-752.

[11] J. Cong, A. B. Kahng, C.-K. Koh and C.-W. A. Tsao, "Bounded-Skew Clock and Steiner Routing Under Elmore Delay," *International Conf. on Computer-Aided Design*, Nov. 1995, pp. 66–71.

[12] J. Cong and C.-K. Koh, "Simultaneous Driver and Wire Sizing for Performance and Power Optimization," *IEEE Trans. on VLSI Systems*, Dec. 1994, pp. 408-425.

[13] J. Cong and C.-K. Koh, "Minimum-Cost Bounded-Skew Clock Routing," *Proc. IEEE Int'l Symp. on Circuits and Systems*, Apr. 1995, vol. 1, pp. 215–218.

[14] J. Cong, C.-K. Koh, and K. S. Leung, "Wiresizing with Driver Sizing for Performance and Power Optimization", *Proc. 1994 Int'l Workshop on Lower Power Design*, 1994.

[15] 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.

[16] J. Cong and K.-S. Leung, “Optimal Wiresizing Under Elmore Delay Model,” *IEEE Trans. on Computer-Aided Design*, Mar. 1995, pp. 321-336.

[17] M. Edahiro, “Minimum Skew and Minimum Path Length Routing in VLSI Layout Design,” *NEC Research and Development*, 32(4), Oct. 1991, pp. 569–575.

[18] W. C. Elmore, “The Transient Response of Damped Linear Network with Particular Regard to Wideband Amplifier”, *J. Applied Physics*, 19(1948), pp. 55-63.

[19] X. Hong, T. Xue, E. S. Kuh, C. K. Cheng, and J. Huang, “Performance-Driven Steiner Tree Algorithms For Global Routing”, *Proc. ACM/IEEE Design Automation Conf.*, 1993, pp. 177-181.

[20] J. H. Huang, A. B. Kahng and C.-W. A. Tsao, “On the Bounded-Skew Routing Tree Problem”, *Proc. ACM/IEEE Design Automation Conf.*, Jun. 1995, pp. 508–513.

[21] 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. Int’l Conf. on Computer-Aided Design*, Nov. 1995, pp. 138–143.

[22] N. Menezes, S. Pullela, and L. T. Pillage, “Simultaneous Gate and Interconnect Sizing for Circuit-Level Delay Optimization,” in *Proc. ACM/IEEE Design Automation Conf.*, Jun. 1995, pp. 690–695.

[23] J. K. Ousterhout, “Switch-Level Delay Models for digital MOS VLSI,” in *Proc. ACM/IEEE Design Automation Conf.*, 1984, pp. 542–548.

[24] S. S. Sapatnekar, “RC Interconnect Optimization under the Elmore Delay Model,” in *Proc. ACM/IEEE Design Automation Conf.*, 1994, pp. 387–391.

[25] L. P. P. P. van Ginneken, “Buffer Placement in Distributed RC-tree Networks for Minimal Elmore Delay,” in *Proc. Int’l Symp. on Circuits and Systems*, 1990, pp. 865–868.

[26] A. Vittal and M. Marek-Sadowska, “Power Optimal Buffered Clock Tree Design,” in *Proc. ACM/IEEE Design Automation Conf.*, 1995, pp. 497–502.

[27] N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design: a Systems Perspective – 2nd ed*, Addison-Wesley, 1993.

Appendix

Lemma 1 *If $c_v^s > c_v^{s\prime}$, then*

$$\sum_{E \in P(N_+,v)} H(E) \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \le 0 \tag{23}$$

Proof:

We obtain the inequality by swapping the two wiresizing solutions. Let $\Delta 1$ denote the difference $t(\mathcal{W}, c_v^s) - t(\mathcal{W}', c_v^s)$ and $\Delta 2$ denote the difference $t(\mathcal{W}', c_v^{s\prime}) - t(\mathcal{W}, c_v^{s\prime})$. Clearly, both $\Delta 1$ and $\Delta 2 \le 0$. We can write $\Delta 1$ and $\Delta 2$ [6] as follows:

$$\begin{aligned}
\Delta 1 &= \mathcal{K}_1 \cdot \sum_{E \in T} H(E) \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \ + \ \mathcal{K}_3 \cdot \sum_{E \in T} (w_E - w'_E) \ + \ \mathcal{K}_5 \cdot \sum_{E,E' \in T, E \ne E'} (\frac{w_{E'}}{w_E} - \frac{w'_{E'}}{w'_E}) \ + \\
& \mathcal{K}_6 \cdot \sum_{E,E' \in T, E \ne E'} F(E,E') \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \ + \ \mathcal{K}_7 \cdot \sum_{E \in T} H(E) \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \cdot \sum_{u \in Sink(E), u \ne v} c_u^s \ + \\
& \mathcal{K}_7 \cdot c_v^s \cdot \sum_{E \in P(N_+,v)} H(E) \cdot (\frac{1}{w_E} - \frac{1}{w'_E})
\end{aligned}$$

$$\begin{aligned}
\Delta 2 &= \mathcal{K}_1 \cdot \sum_{E \in T} H(E) \cdot (\frac{1}{w'_E} - \frac{1}{w_E}) \ + \ \mathcal{K}_3 \cdot \sum_{E \in T} (w'_E - w_E) \ + \ \mathcal{K}_5 \cdot \sum_{E,E' \in T, E \ne E'} (\frac{w'_{E'}}{w'_E} - \frac{w_{E'}}{w_E}) \ + \\
& \mathcal{K}_6 \cdot \sum_{E,E' \in T, E \ne E'} F(E,E') \cdot (\frac{1}{w'_E} - \frac{1}{w_E}) \ + \ \mathcal{K}_7 \cdot \sum_{E \in T} H(E) \cdot (\frac{1}{w'_E} - \frac{1}{w_E}) \cdot \sum_{u \in Sink(E), u \ne v} c_u^s \ + \\
& \mathcal{K}_7 \cdot c_v^{s\prime} \cdot \sum_{E \in P(N_+,v)} H(E) \cdot (\frac{1}{w'_E} - \frac{1}{w_E})
\end{aligned}$$

Summing up $\Delta 1$ and $\Delta 2$, we obtain

$$(c_v^s - c_v^{s\prime}) \cdot \sum_{E \in P(N_+,v)} H(E) \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \ \le \ 0$$

Since $c_v^s > c_v^{s\prime}$, the result follows. □

Lemma 2 *Consider an edge E in the routing tree. If $w_E < w'_E$, then*

$$\sum_{E' \in Ans(E)} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) + \frac{H(E)}{w_E \cdot w'_E} \sum_{E' \in Des(E)} (w'_{E'} - w_{E'}) > 0 \tag{24}$$

Proof: Applying the same technique of swapping wire assignment, except that now we only swap w_E with w'_E while keeping the rest of the wire assignment intact.

We have two cases: (i) $E \in P(N_+, v)$, and (ii) $E \notin P(N_+, v)$. In case (i):

$$\begin{aligned}
\Delta 1 &= t(\mathcal{W}, c_v^s) - t(\mathcal{W} - w_E \cup \{w'_E\}, c_v^s) \\
&= \mathcal{K}_1 \cdot H(E) \cdot (\frac{1}{w_E^*} - \frac{1}{w_E'^*}) \ + \ \mathcal{K}_3 \cdot (w_E - w'_E) \ + \ \mathcal{K}_5 \cdot \sum_{E' \in Ans(E)} F(E', E) \cdot (\frac{w_E - w'_E}{w_{E'}}) \ + \\
& \mathcal{K}_5 \cdot \sum_{E' \in Des(E)} F(E,E') \cdot w_{E'} \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \ + \ \mathcal{K}_6 \cdot \sum_{E' \in Des(E)} F(E,E') \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \ +
\end{aligned}$$

[6] Due to similarity in their derivations, in subsequent proofs, only $\Delta 1$ will be derived.

$$
\begin{aligned}
&\mathcal{K}_7 \cdot H(E) \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \cdot \sum_{u \in Sink(E), u \neq v} c^s_u \; + \; \mathcal{K}_7 \cdot H(E) \cdot c^s_v \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \\
\leq \;\; & 0
\end{aligned}
$$

Similarly, we can obtain $\Delta 2 = t(\mathcal{W}', c^{s\prime}_v) - t(\mathcal{W}' - w'_E \cup \{w_E\}, c^{s\prime}_v) \leq 0$ and deduce the following:

$$
\begin{aligned}
\Delta 1 \; + \; \Delta 2 \;\; = \;\; & \mathcal{K}_5 \cdot \sum_{E' \in Ans(E)} F(E', E) \cdot (w_E - w'_E) \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \; + \\
& \mathcal{K}_5 \cdot \sum_{E' \in Des(E)} F(E, E') \cdot (w_{E'} - w'_{E'}) \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \; + \; \mathcal{K}_7 \cdot H(E) \cdot (c^s_v - c^{s\prime}_v) \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \\
\leq \;\; & 0
\end{aligned}
$$

Since $w_E - w'_E < 0$ and $F(E_1, E_2) = H(E_1)$ if $E_2 \in Des(E_1)$,

$$
\mathcal{K}_5 \cdot \sum_{E' \in Ans(E)} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \; + \; \mathcal{K}_5 \cdot \sum_{E' \in Des(E)} H(E) \cdot (w'_{E'} - w_{E'}) \cdot (\frac{1}{w_E \cdot w'_E}) \; + \; \mathcal{K}_7 \cdot H(E) \cdot (c^{s\prime}_v - c^s_v) \cdot \frac{1}{w_E \cdot w'_E} \; \geq \; 0
$$

Since $c^{s\prime}_v - c^s_v < 0$ and $\mathcal{K}_5 > 0$, the result follows.

In case (ii),

$$
\begin{aligned}
\Delta 1 \;\; = \;\; & t(\mathcal{W}, c^s_v) - t(\mathcal{W} - w_E \cup \{w'_E\}, c^s_v) \\
= \;\; & \mathcal{K}_1 \cdot H(E) \cdot (\frac{1}{w^*_E} - \frac{1}{w'^*_E}) \; + \; \mathcal{K}_3 \cdot (w_E - w'_E) \; + \; \mathcal{K}_5 \cdot \sum_{E' \in Ans(E)} F(E', E) \cdot (\frac{w_E - w'_E}{w_{E'}}) \; + \\
& \mathcal{K}_5 \cdot \sum_{E' \in Des(E)} F(E, E') \cdot w_{E'} \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \; + \; \mathcal{K}_6 \cdot \sum_{E' \in Des(E)} F(E, E') \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \; + \\
& \mathcal{K}_7 \cdot H(E) \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \cdot \sum_{u \in Sink(E), u \neq v} c^s_u \\
\leq \;\; & 0
\end{aligned}
$$

Again, the result holds when we sum $\Delta 1$ and $\Delta 2$. □

Lemma 3 *Consider the edge E directly above the loading capacitance at v. If $c^s_v > c^{s\prime}_v$, then $w_E \geq w'_E$.*

Proof: Suppose the edge E have no descendant edges. Suppose $w_E < w'_E$, we obtain the following from Eqn. (24) in Lemma 2:

$$
\sum_{E' \in Ans(E)} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \; > \; 0
$$

However, this contradicts Lemma 1 since $E \cup Ans(E) = P(N_+, v)$.

Suppose the edge E has descendant edges. Again, suppose $w_E < w'_E$. Since $\sum_{E' \in P(N_+, v)} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \leq 0$ (Lemma 1), we have $\sum_{E' \in Ans(E)} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}} < 0$. From Lemma 2, $\sum_{E' \in Des(E)} (w'_{E'} w_{E'}) > 0$.

Let T_E denote single-stem tree with E as the root stem [16]. Note that $E \in T_E$. Let $\mathcal{W}(T_E)$ be the wire width assignment of the subtree T_E. Again, we apply the technique of swapping the wire width assignment of the edges in T_E.

$$
\Delta 1 \;\; = \;\; t(\mathcal{W}, c^s_v) - t(\mathcal{W} - \mathcal{W}(T_E) + \mathcal{W}'(T_E), c^s_v)
$$

$$
\begin{aligned}
&= \mathcal{K}_1 \cdot \sum_{E' \in T_E} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \;+\; \mathcal{K}_3 \cdot \sum_{E' \in T_E} (w_{E'} - w'_{E'}) \;+ \\
&\quad \mathcal{K}_5 \cdot \sum_{E' \in Ans(E), E'' \in T_E} F(E',E'') \cdot (\frac{w_{E''}}{w_{E'}} - \frac{w'_{E''}}{w_{E'}}) \;+\; \mathcal{K}_5 \cdot \sum_{E',E'' \in T_E} F(E',E'') \cdot (\frac{w_{E''}}{w_{E'}} - \frac{w'_{E''}}{w'_{E'}}) \;+ \\
&\quad \mathcal{K}_6 \cdot \sum_{E',E'' \in T_E} F(E',E'') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \;+\; \mathcal{K}_7 \cdot \sum_{E' \in T_E} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \cdot \sum_{u \in Sink(E'), u \neq v} c_u^s \;+ \\
&\quad \mathcal{K}_7 \cdot H(E) \cdot (\frac{1}{w_E} - \frac{1}{w'_E}) \cdot c_v^s \\
&\leq 0
\end{aligned}
$$

Similarly, we define $\Delta 2$ to be $t(\mathcal{W}', c_v^{s\prime}) - t(\mathcal{W}' - \mathcal{W}'(T_E) + \mathcal{W}(T_E), c_v^{s\prime})$. Summing up $\Delta 1$ and $\Delta 2$, we obtain

$$
\mathcal{K}_5 \cdot \sum_{E' \in Ans(E), E'' \in T_E} F(E',E'') \cdot (w_{E''} - w'_{E''}) \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \;=\; \mathcal{K}_5 \cdot \sum_{E' \in Ans(E)} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \cdot \sum_{E'' \in T_E} (w_{E''} - w'_{E''}) \;\leq\; 0
$$

Since $\sum_{E'' \in T_E} (w_{E''} - w'_{E''}) < 0$, we have $\sum_{E' \in Ans(E)} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \geq 0$ which is a contradiction. □

Lemma 4 *Consider the edge E directly above the loading capacitance at v. If $c_v^s > c_v^{s\prime}$, then there exists an optimal wire-sizing solution $\mathcal{W}$ such that it dominates $\mathcal{W}'$ for all edges in $Des(E)$.*

Proof: Suppose not, let $\tilde{E} \in Des(E)$ be the first edge in a top-down topological order such that $w_{\tilde{E}} < w_{\tilde{E}}$. Clearly, $\sum_{E' \in Ans(\tilde{E})} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \leq 0$ since for all edges E' from v to the parent of $\tilde{E}$, $w_{E'} \geq w'_{E'}$. From Lemma 2, we therefore have $\sum_{E' \in Des(\tilde{E})} (w'_{E'} - w_{E'}) \geq 0$. Let $T_{\tilde{E}}$ denote the single-stem tree rooted at $\tilde{E}$ (including $\tilde{E}$). Then $\sum_{E' \in T_{\tilde{E}}} (w'_{E'} - w_{E'}) > 0$.

Now, we swap wire width assignment of $T_{\tilde{E}}$ as before,

$$
\begin{aligned}
\Delta 1 &= t(\mathcal{W}, c_v^s) - t(\mathcal{W} - \mathcal{W}(T_{\tilde{E}}) \cup \mathcal{W}'(T_{\tilde{E}}), c_v^s) \\
&= \mathcal{K}_1 \cdot \sum_{E' \in T_{\tilde{E}}} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \;+\; \mathcal{K}_3 \cdot \sum_{E' \in T_{\tilde{E}}} (w_{E'} - w'_{E'}) \;+ \\
&\quad \mathcal{K}_5 \cdot \sum_{E' \in Ans(\tilde{E}), E'' \in T_{\tilde{E}}} F(E',E'') \cdot (\frac{w_{E''}}{w_{E'}} - \frac{w'_{E''}}{w_{E'}}) \;+\; \mathcal{K}_5 \cdot \sum_{E',E'' \in T_{\tilde{E}}} F(E',E'') \cdot (\frac{w_{E''}}{w_{E'}} - \frac{w'_{E''}}{w'_{E'}}) \;+ \\
&\quad \mathcal{K}_6 \cdot \sum_{E',E'' \in T_{\tilde{E}}} F(E',E'') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \;+\; \mathcal{K}_7 \cdot \sum_{E' \in T_{\tilde{E}}} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \cdot \sum_{u \in Sink(E'), u \neq v} c_u^s \\
&\leq 0
\end{aligned}
$$

Again, we obtain for $\Delta 2 = t(\mathcal{W}', c_v^{s\prime}) - t(\mathcal{W}' - \mathcal{W}'(T_{\tilde{E}}) \cup \mathcal{W}'(T_{\tilde{E}}), c_v^{s\prime}) \leq 0$ and sum them up. We obtain an identical expression as in the previous lemma:

$$
\mathcal{K}_5 \cdot \sum_{E' \in Ans(\tilde{E}), E'' \in T_{\tilde{E}}} F(E',E'') \cdot (w_{E''} - w'_{E''}) \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \;=\; \mathcal{K}_5 \cdot \sum_{E' \in Ans(\tilde{E})} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \cdot \sum_{E'' \in T_{\tilde{E}}} (w_{E''} - w'_{E''}) \;\leq\; 0
$$

Since $\sum_{E' \in T_{\tilde{E}}} (w_{E'} - w'_{E'}) < 0$, we have $\sum_{E' \in Ans(\tilde{E})} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \geq 0$. However, $\sum_{E' \in Ans(\tilde{E})} H(E') \cdot (\frac{1}{w_{E'}} - \frac{1}{w'_{E'}}) \leq 0$. Therefore $\Delta 1 + \Delta 2 = 0$, implying that $\Delta 1 = 0$ and $\Delta 2 = 0$. In other words, swapping the wire width assignment in $T_{\tilde{E}}$ does not affect the optimality of the wiresizing solutions, and we have found a wiresizing solution such that the wire width assignment of $T_{\tilde{E}}$ dominates $\mathcal{W}'(T_{\tilde{E}})$. □