A New Approach to Simultaneous Buffer Insertion and Wire Sizing

Chris C. N. Chu and D. F. Wong
cnchu@cs.utexas.edu and wong@cs.utexas.edu
Department of Computer Sciences, University of Texas at Austin, Austin, TX 78712.

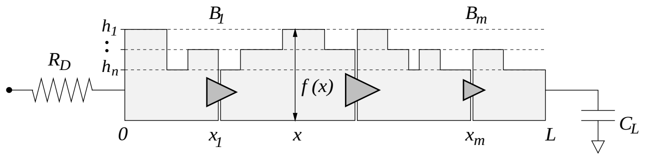
Abstract

In this paper, we present a completely new approach to the problem of delay minimization by simultaneous buffer insertion and wire sizing for a wire. We show that the problem can be formulated as a convex quadratic program, which is known to be solvable in polynomial time. Nevertheless, we explore some special properties of our problem and derive an optimal and very efficient algorithm to solve the resulting program. Given m buffers and a set of n discrete choices of wire width, the running time of our algorithm is $O(mn^2)$ and is independent of the wire length in practice. For example, an instance of 100 buffers and 100 choices of wire width can be solved in 3 seconds. Besides, our formulation is so versatile that it is easy to consider other objectives like wire area or power dissipation, or to add constraints to the solution. Also, wire capacitance lookup tables, or very general wire capacitance models which can capture area capacitance, fringing capacitance, coupling capacitance, etc. can be used.

PROBLEM 1: The Simultaneous Buffer Insertion and Wire Sizing Problem

Given: wire length L, driver resistance R_D, load capacitance C_L, a set $H = \{h_1, \dots, h_n\}$ of choices of wire width such that $h_1 > \cdots > h_n$, and m buffers $B_1, \dots, B_m$.

Determine: the positions $x_1, \dots, x_m$ at which the buffers are inserted and the wire width $f(x)$ at each point x along the wire such that the delay from source to sink is minimized.



1 Introduction

In the past, gate delay is the dominating factor in circuit design. However, as the feature size of VLSI devices continues to decrease, interconnect delay becomes increasingly important. Nowadays, feature size has been down to $0.25\mu m$ in advance technology. Interconnect delay has become the dominating factor in determining system performance. In many systems designed today, as much as 50% to 70% of clock cycle are consumed by interconnect delay [9]. It is predicted in [12] that the feature size will be reduced to $0.18\mu m$ by 1999 and $0.13\mu m$ by 2002. So we expect the significance of interconnect delay will further increase in near future.

In this paper, we will mainly consider the simultaneous buffer insertion and wire sizing problem as stated in Problem 1. (Elmore delay model [13] is used for delay calculation.) Both buffer insertion and wire sizing have been shown to be effective techniques to reduce interconnect delay and many work has been done during the past few years. See below for a brief overview or see [9] for a comprehensive survey.

For wire sizing alone, almost all the previous work gives an approximate solution by the approach of dividing the wire into small fixed-length segments and optimizing the width of each segment iteratively (e.g. [1, 3, 7, 8, 10, 11, 20, 21, 23]). In order to obtain accurate results, the wire usually needs to be divided into a large number of (much more than n) segments. [2, 4, 14, 15] consider a variant such that the set H of choices of wire width is a continuous interval. Therefore, the resulting wire width function $f(x)$ is continuous. However, a continuous shape is expensive to fabricate.

Not much work has been done on simultaneous buffer insertion and wire sizing. Recently, [18] generalizes the dynamic programming algorithm in [22] to handle buffer insertion and wire sizing simultaneously. Their algorithm also allows choices of buffers of different size and includes power consideration. However, their algorithm runs in pseudo-polynomial time and requires a substantial amount of memory. [6] also considers buffer insertion, buffer sizing and wire sizing simultaneously and a closed form optimal solution is obtained. However, in that paper, only wire area capacitance is considered (terms like wire fringing capacitance are ignored).

We present a completely new approach in this paper. Instead of solving Problem 1 directly, we will solve an equivalent problem which will be introduced later (see Problem 1′ in Section 3). Our new approach has many advantages over the previous approaches:

1. The problem of our approach has much less vari-

ables than the problem of the traditional approach of dividing a wire into small fixed-length segments. If m buffers are to be inserted and a set of n choices of wire width is given, the problem we formulated will have $(m+1)n$ variables no matter how long the wire is. As in practice, usually only a few buffers and a few choices for the wire width are allowed, $(m+1)n$ is a small number. Moreover, the problem of our approach is completely equivalent to Problem 1 (not an approximation).

2. The problem of our approach can be solved optimally and very efficiently even for large m and n. We will show that our problem is a convex quadratic program. Convex quadratic programming has been well studied and can be solved efficiently by many public domain or commercial software systems. Nevertheless, we explore some special properties of our problem and derive an even more efficient iterative algorithm such that each iteration needs only linear time. In practice, the algorithm runs in about n iterations only. For example, an instance of 100 buffers and 100 choices of wire width (i.e. 10100 variables) can be solved in 3 seconds by our algorithm.

3. Buffer insertion is generally considered a hard problem and usually some heuristics or dynamic programming are needed to handle it. However, it is interesting to note how naturally and easily buffer insertion is handled in our approach. We will see that it is no more difficult than wire sizing alone.

4. Besides delay, our formulation can be easily extended to consider other objectives like wire area or power dissipation. For example, we can optimally solve the problems of minimizing a weighted sum of delay and wire area, minimizing delay with bounded area, minimizing area with bounded delay, etc. Our formulation also allows adding constraints to the solution. Moreover, our efficient algorithm can still be applied to get optimal results.

5. We can use very general wire capacitance models which can capture area capacitance, fringing capacitance, coupling capacitance (the capacitance due to an adjacent parallel wire), etc. A wire segment is modeled as a π-type RC circuit as shown in Figure 1. The capacitance of a wire segment of width h and length l is given by $c(h)l$, where $c(h)$ is the unit length wire capacitance for a segment of width h. The only restriction on c is that it has to be an increasing function from $\mathcal{R}^+$ to $\mathcal{R}^+$. For example, to model wire area capacitance, wire fringing capacitance and coupling capacitance at the same time, suppose the distance to an adjacent parallel wire is $d-h$ when the wire width is h. Then we can set $c(h) = c_0 h + c_f + c_c/(d-h)$, where c_0 is the unit wire area capacitance, c_f is the unit wire fringing capacitance and c_c is the unit wire coupling capacitance. The values of $c(h)$ for each h in H can also be obtained from a lookup table.

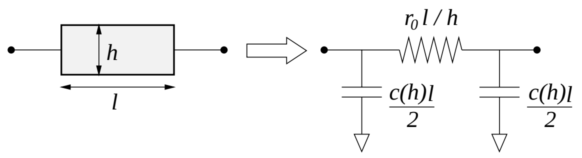


Figure 1: *The model of a wire segment of length l and width h by a π-type RC circuit. r_0 is the unit wire resistance. $c(h)$ is the wire capacitance per unit length for a segment of width h. We assume $c(h)$ is an increasing function in this paper.*

The paper is organized as follows. In Section 2, we will first consider the problem of wire sizing without buffer insertion. Once the formulation and the algorithm for wire sizing is understood, the extension to simultaneous buffer insertion and wire sizing will be easy and will be discussed in Section 3. In Section 4, we will further extend our results to consider other objectives like wire area or power, and to handle additional constraints to the solution. In Section 5, some experimental results to show the efficiency of our algorithm will be presented. In Section 6, we discuss some directions for future research.

2 Wire Sizing

For the rest of the paper, we will use uppercase boldface letters to denote matrices and lowercase boldface letters to denote vectors. We will use the convention that indices of matrices and vectors start from one. To simplify the presentation, if we refer to an element of a matrix or a vector such that the index is out of range, we assume that the value is zero.

2.1 Our New Approach

We want to solve the wire sizing problem as stated in Problem 2, where the optimal wire width is represented by a step function $f : [0, L] \to H$. Instead of solving Problem 2 directly, we approach the problem by looking at an equivalent problem. Before we introduce the new problem, we will first prove that f must be a decreasing function. A similar monotone property for simpler wire capacitance model and fixed-length segments has been proved in [11].

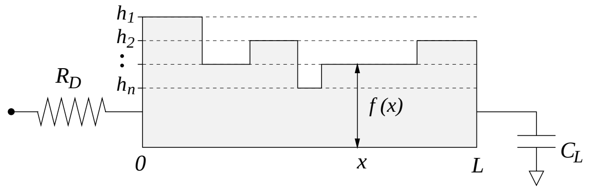
Lemma 1 *The optimal wire sizing function f is a decreasing function.*

Proof outline: Suppose f changes from a smaller value h to a larger value h' at a point of a distance l from the source. Therefore, $f(x) = h$ for $l - \delta \le x \le l$ and $f(x) = h'$ for $l < x \le l + \delta$ for some $\delta > 0$. Let g

PROBLEM 2: The Wire Sizing Problem

Given: wire length L, driver resistance R_D, load capacitance C_L, a set $H = \{h_1, \ldots, h_n\}$ of choices of wire width such that $h_1 > \cdots > h_n$.

Determine: the wire width $f(x)$ at each point x along the wire such that the delay from source to sink is minimized.



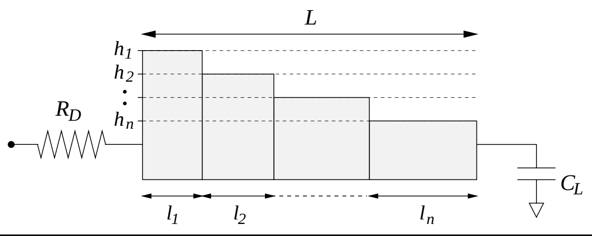
be another wire sizing function defined as $g(x) = h'$ for $l - \delta \leq x \leq l$, $g(x) = h$ for $l < x \leq l + \delta$ and $g(x) = f(x)$ otherwise. We can show that g is better than f, which contradicts to the fact that f is optimal. □

Instead of solving Problem 2 directly, we will solve Problem 2′ below. In Problem 2′, the wire is divided into n segments such that the width of the ith segment is h_i, and the length of each segment is to be determined. By Lemma 1, it is clear that Problem 2′ is equivalent to Problem 2. Note that our new approach divides the wire into only n segments and gives an optimal solution to the original problem. If we approach Problem 2 by dividing the wire into small fixed-length segments, the solution will not be exact. In order to obtain a good approximation, the wire needs to be divided into much more than n segments.

PROBLEM 2′: The Wire Sizing Problem of Our Approach (equivalent to Problem 2)

Given: wire length L, driver resistance R_D, load capacitance C_L, a set $H = \{h_1, \ldots, h_n\}$ of choices of wire width such that $h_1 > \cdots > h_n$.

Determine: the segment lengths $l_i \geq 0$ for $1 \leq i \leq n$ such that the delay from source to sink is minimized.



2.2 Problem formulation

We will show in this subsection that Problem 2′ can be formulated as a convex quadratic program.

Let $c_i = c(h_i)$ for $1 \leq i \leq n$. Then for Problem 2′, the delay from source to sink is

$$\begin{aligned}
D &= R_D(c_1 l_1 + c_2 l_2 + \cdots + c_n l_n + C_L) \\
&+ \frac{r_0 l_1}{h_1}\left(\frac{c_1 l_1}{2} + c_2 l_2 + \cdots + c_n l_n + C_L\right) \\
&+ \frac{r_0 l_2}{h_2}\left(\frac{c_2 l_2}{2} + c_3 l_3 + \cdots + c_n l_n + C_L\right) \\
&\vdots \\
&+ \frac{r_0 l_n}{h_n}\left(\frac{c_n l_n}{2} + C_L\right) \\
&= \frac{1}{2}\boldsymbol{l}^T \boldsymbol{\Phi} \boldsymbol{l} + \boldsymbol{\rho}^T \boldsymbol{l} + R_D C_L
\end{aligned}$$

where

$$\boldsymbol{\Phi} = \begin{pmatrix} c_1 r_0/h_1 & c_2 r_0/h_1 & c_3 r_0/h_1 & \cdots & c_n r_0/h_1 \\ c_2 r_0/h_1 & c_2 r_0/h_2 & c_3 r_0/h_2 & \cdots & c_n r_0/h_2 \\ c_3 r_0/h_1 & c_3 r_0/h_2 & c_3 r_0/h_3 & \cdots & c_n r_0/h_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_n r_0/h_1 & c_n r_0/h_2 & c_n r_0/h_3 & \cdots & c_n r_0/h_n \end{pmatrix},$$

$$\boldsymbol{\rho} = \begin{pmatrix} R_D c_1 + C_L r_0/h_1 \\ R_D c_2 + C_L r_0/h_2 \\ R_D c_3 + C_L r_0/h_3 \\ \vdots \\ R_D c_n + C_L r_0/h_n \end{pmatrix} \text{ and } \boldsymbol{l} = \begin{pmatrix} l_1 \\ l_2 \\ l_3 \\ \vdots \\ l_n \end{pmatrix}.$$

So Problem 2′ can be formulated as follows:

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\boldsymbol{l}^T \boldsymbol{\Phi} \boldsymbol{l} + \boldsymbol{\rho}^T \boldsymbol{l} \\ \text{subject to} & l_1 + \cdots + l_n = L \\ & l_i \geq 0 \text{ for } 1 \leq i \leq n \end{array} \qquad (1)$$

This is a quadratic program. In general, quadratic program is a mathematical program with a quadratic objective function subject to linear equality and inequality constraints. If the matrix $\boldsymbol{\Phi}$ is positive definite, it is called a convex quadratic program. Note that quadratic programming is NP-hard [16] but convex quadratic programming can be solved in polynomial time [17]. In the following, we will prove that the quadratic program (1) is convex. First, we make the following two definitions.

Definition 1 *(Symmetric Decomposable Matrix)* *Let $\boldsymbol{Q} = (q_{ij})$ be an $n \times n$ symmetric matrix. If for some $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)^T$ and $\boldsymbol{v} = (v_1, \ldots, v_n)^T$ such that $0 < \alpha_1 < \cdots < \alpha_n$, $q_{ij} = q_{ji} = \alpha_i v_i v_j$ for $i \leq j$, then $\boldsymbol{Q}$ is called a symmetric decomposable matrix. We denote $\boldsymbol{Q} = SDM(\boldsymbol{\alpha}, \boldsymbol{v})$.*

Definition 2 *(Upper Triangular Decomposable Matrix)* *Let $\boldsymbol{U} = (u_{ij})$ be an $n \times n$ upper triangular matrix. If for some $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_n)^T$ and $\boldsymbol{v} = (v_1, \ldots, v_n)^T$ such that $\beta_i > 0$ for all i, $u_{ij} = \beta_i v_j$ for $i \leq j$ and $u_{ij} = 0$ for $i > j$, then $\boldsymbol{U}$ is called an upper triangular decomposable matrix. We denote $\boldsymbol{U} = UTDM(\boldsymbol{\alpha}, \boldsymbol{v})$.*

Lemma 2 *$\boldsymbol{\Phi}$ in (1) is symmetric decomposable.*

Proof:
Let $\boldsymbol{\alpha} = (\frac{r_0}{c_1 h_1}, \ldots, \frac{r_0}{c_n h_n})^T$ and $\boldsymbol{v} = (c_1, \ldots, c_n)^T$. Note that $0 < \frac{r_0}{c_1 h_1} < \cdots < \frac{r_0}{c_n h_n}$. Then $\boldsymbol{\Phi} = SDM(\boldsymbol{\alpha}, \boldsymbol{v})$. □

Lemma 3 *If $\boldsymbol{Q}$ is symmetric decomposable, then $\boldsymbol{Q} = \boldsymbol{U}^T\boldsymbol{U}$ where $\boldsymbol{U}$ is upper triangular decomposable. In particular, if $\boldsymbol{Q} = SDM(\boldsymbol{\alpha}, \boldsymbol{v})$, then $\boldsymbol{Q} = \boldsymbol{U}^T\boldsymbol{U}$ where $\boldsymbol{U} = UTDM(\boldsymbol{\beta}, \boldsymbol{v})$, $\beta_i = \sqrt{\alpha_i - \alpha_{i-1}}$ for $1 \leq i \leq n$.*

Proof outline:
Can be verified by multiplying $\boldsymbol{U}^T$ and $\boldsymbol{U}$. □

Lemma 4 *If $\boldsymbol{Q}$ is symmetric decomposable, then $\boldsymbol{Q}$ is positive definite.*

Proof: Let $\boldsymbol{Q} = SDM(\boldsymbol{\alpha}, \boldsymbol{v})$. By Lemma 3, $\boldsymbol{Q} = \boldsymbol{U}^T\boldsymbol{U}$ where $\boldsymbol{U} = UTDM(\boldsymbol{\beta}, \boldsymbol{v})$ for some $\boldsymbol{\beta}$. For any $\boldsymbol{x} \neq \boldsymbol{0}$, let $\boldsymbol{y} = \boldsymbol{U}\boldsymbol{x}$. Note that $\boldsymbol{y} \neq \boldsymbol{0}$ as $\boldsymbol{U}$ is nonsingular and $\boldsymbol{x} \neq \boldsymbol{0}$. So $\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{x}^T\boldsymbol{U}^T\boldsymbol{U}\boldsymbol{x} = \boldsymbol{y}^T\boldsymbol{y} > 0$. In other words, $\boldsymbol{Q}$ is positive definite. □

By Lemma 2, Lemma 4 and [17], we have Theorem 1.

Theorem 1 *The quadratic program (1) is convex, and hence can be solved in polynomial time.*

2.3 Algorithm

In the following, we will make some interesting observations on the convex quadratic program (1) and then derive a very efficient algorithm to solve it based on the idea of active set method. Active set method is a classical technique for constrained optimization problems. It is also one of the most popular methods to solve quadratic programming. It has been shown to be efficient in practice. We will first give a brief outline on how to solve convex quadratic programming by active set method. See [19] for details.

If a convex quadratic program consists of equality constraints only, it is particularly easy to solve. Consider the following program:

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\boldsymbol{l}^T\boldsymbol{\Phi}\boldsymbol{l} + \boldsymbol{\rho}^T\boldsymbol{l} \\ \text{subject to} & \boldsymbol{\Gamma}\boldsymbol{l} = \boldsymbol{b} \end{array} \qquad (2)$$

where $\boldsymbol{\Phi}$ is positive definite and $\boldsymbol{\Gamma}$ is of full rank. Consider the associated Lagrangian:

$$\ell(\boldsymbol{l}, \boldsymbol{\lambda}) = \frac{1}{2}\boldsymbol{l}^T\boldsymbol{\Phi}\boldsymbol{l} + \boldsymbol{\rho}^T\boldsymbol{l} + \boldsymbol{\lambda}^T(\boldsymbol{\Gamma}\boldsymbol{l} - \boldsymbol{b})$$

The Lagrange necessary conditions of optimality are $\partial\ell(\boldsymbol{l}, \boldsymbol{\lambda})/\partial l_i = 0$ and $\partial\ell(\boldsymbol{l}, \boldsymbol{\lambda})/\partial\lambda_i = 0$ for all i. The conditions can be written in matrix form as follows:

$$\begin{array}{lcl} \boldsymbol{\Phi}\boldsymbol{l} + \boldsymbol{\Gamma}^T\boldsymbol{\lambda} + \boldsymbol{\rho} & = & \boldsymbol{0} \\ \boldsymbol{\Gamma}\boldsymbol{l} - \boldsymbol{b} & = & \boldsymbol{0} \end{array}$$

Since $\boldsymbol{\Phi}$ is positive definite and $\boldsymbol{\Gamma}$ is of full rank, it can be shown that the conditions can be uniquely solved:

$$\begin{array}{lcl} \boldsymbol{\lambda} & = & -(\boldsymbol{\Gamma}\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma}^T)^{-1}(\boldsymbol{\Gamma}\boldsymbol{\Phi}^{-1}\boldsymbol{\rho} + \boldsymbol{b}) \\ \boldsymbol{l} & = & -\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma}^T\boldsymbol{\lambda} - \boldsymbol{\Phi}^{-1}\boldsymbol{\rho} \end{array} \qquad (3)$$

Now consider a general convex quadratic program, which can have both equality and inequality constraints. If we solve the program by ignoring all the inequality constraints and the solution obtained is feasible with respect to all constraints, then it should be the optimal solution of the original program. If it is non-feasible, the optimal solution should be on the boundary of the set of feasible solutions. Active set method is a systematic way to search for the optimal solution on the boundary.

The idea underlying active set method is to partition the inequality constraints into two groups: active and inactive. In each iteration, those active constraints are treated as equality constraints and those inactive constraints are essentially ignored (i.e. the feasible space is restricted to some boundary). The resulting equality constrained program is solved. If the solution is infeasible with respect to the original program, some inactive constraints will be added to the set of active constraints. If the solution is feasible but not optimal (i.e. some Lagrange multipliers are negative), some constraints will be removed from the current active set. The process is repeated until the optimal solution is found (in that case, the active set will define the boundary the optimal solution is on).

Therefore if we apply active set method to (1), we need to solve an equality constrained problem in the form of (2) (i.e. compute (3)) in each iteration. Each iteration can be done in cubic time in general. However, we will show below that (3) can be solved in linear time in our case.

First, note that all the inequality constraints in (1) are of the form $l_i \geq 0$. If we set $l_i = 0$ and substitute it into (1), the resulting program is of exactly the same form as (1) but of smaller size. In fact, if for some iteration, all constraints except $l_{j_1} \geq 0, \ldots, l_{j_r} \geq 0$ are in the active set $\mathcal{A}$, then the equality constrained program corresponding to that iteration will be equivalent to the following reduced program:

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\boldsymbol{l}_{\mathcal{A}}^T\boldsymbol{\Phi}_{\mathcal{A}}\boldsymbol{l}_{\mathcal{A}} + \boldsymbol{\rho}_{\mathcal{A}}^T\boldsymbol{l}_{\mathcal{A}} \\ \text{subject to} & \boldsymbol{\Gamma}_{\mathcal{A}}\boldsymbol{l}_{\mathcal{A}} = L \end{array} \qquad (4)$$

where $\boldsymbol{l}_{\mathcal{A}} = (l_{j_1}, \ldots, l_{j_r})^T$, $\boldsymbol{\Gamma}_{\mathcal{A}} = (1\ 1\ \cdots\ 1)$, $\boldsymbol{\rho}_{\mathcal{A}} = (R_D c_{j_1} + C_L r_0/h_{j_1}, \ldots, R_D c_{j_r} + C_L r_0/h_{j_r})^T$, and $\boldsymbol{\Phi}_{\mathcal{A}}$ is the symmetric decomposable matrix corresponding to $\mathcal{A}$ (i.e. $\boldsymbol{\Phi}_{\mathcal{A}} = SDM(\boldsymbol{\alpha}_{\mathcal{A}}, \boldsymbol{v}_{\mathcal{A}})$ with $\boldsymbol{\alpha}_{\mathcal{A}} = (\frac{r_0}{c_{j_1}h_{j_1}}, \ldots, \frac{r_0}{c_{j_r}h_{j_r}})^T$ and $\boldsymbol{v}_{\mathcal{A}} = (c_{j_1}, \ldots, c_{j_r})^T$).

As before, the Lagrange necessary conditions of optimality for (4) are

$$\begin{aligned}\lambda_{\mathcal{A}} &= -(\mathbf{\Gamma}_{\mathcal{A}}\mathbf{\Phi}_{\mathcal{A}}^{-1}\mathbf{\Gamma}_{\mathcal{A}}^T)^{-1}(\mathbf{\Gamma}_{\mathcal{A}}\mathbf{\Phi}_{\mathcal{A}}^{-1}\boldsymbol{\rho}_{\mathcal{A}} + L) \\ \boldsymbol{l}_{\mathcal{A}} &= -\mathbf{\Phi}_{\mathcal{A}}^{-1}\mathbf{\Gamma}_{\mathcal{A}}^T\lambda_{\mathcal{A}} - \mathbf{\Phi}_{\mathcal{A}}^{-1}\boldsymbol{\rho}_{\mathcal{A}}\end{aligned}$$

The crucial observation is that $\mathbf{\Phi}_{\mathcal{A}}^{-1}$ is tridiagonal, as proved in the following lemma.

Lemma 5 *If $\boldsymbol{Q}$ is symmetric decomposable, then $\boldsymbol{Q}^{-1}$ is tridiagonal. In particular, if $\boldsymbol{Q} = SDM(\boldsymbol{\alpha}, \boldsymbol{v})$, then $\boldsymbol{Q}^{-1} = (\theta_{ij})$ where $\theta_{ii} = 1/((\alpha_i - \alpha_{i-1})v_i^2) + 1/((\alpha_{i+1} - \alpha_i)v_i^2)$, $\theta_{i,i+1} = \theta_{i+1,i} = -1/((\alpha_{i+1} - \alpha_i)v_i v_{i+1})$ for $1 \le i \le n-1$, $\theta_{nn} = 1/((\alpha_n - \alpha_{n-1})v_n^2)$, and $\theta_{ij} = 0$ otherwise.*

Proof outline: Can be verified by multiplying the matrices $\boldsymbol{Q} = SDM(\boldsymbol{\alpha}, \boldsymbol{v})$ and (θ_{ij}) defined above. □

By Lemma 2 and Lemma 5, we have the Theorem 2.

Theorem 2 *$\mathbf{\Phi}_{\mathcal{A}}^{-1}$ in (4) is tridiagonal.*

Let $\mathbf{\Phi}_{\mathcal{A}}^{-1} = (\theta_{ij})$ where θ_{ij} is given by Lemma 5 and let $\boldsymbol{\rho}_{\mathcal{A}} = (\rho_1, \ldots, \rho_r)^T$. Then the Lagrange optimality conditions for (4) can be written in closed form as follows:

$$\begin{aligned}\lambda_{\mathcal{A}} &= -\frac{L + \sum_{i=1}^{r}(\theta_{i-1,i} + \theta_{ii} + \theta_{i+1,i})\rho_i}{\sum_{i=1}^{r}(\theta_{i-1,i} + \theta_{ii} + \theta_{i+1,i})} \\ l_{j_i} &= -(\theta_{i-1,i}\rho_{i-1} + \theta_{ii}\rho_i + \theta_{i+1,i}\rho_{i+1}) \\ &\quad - (\theta_{i-1,i} + \theta_{ii} + \theta_{i+1,i})\lambda_{\mathcal{A}} \qquad \text{for } 1 \le i \le r\end{aligned}$$

Obviously, $l_j = 0$ for all $j \notin \{j_1, \ldots, j_r\}$. Once $\boldsymbol{l}$ is found, it is not difficult to see that $\boldsymbol{\lambda}$ in (3) can also be found in linear time. Hence (3) can be computed in linear time. The algorithm can be summarized as below.

```
Algorithm MASM (Modified Active Set Method)
 1. Set the active set $\mathcal{A} = \emptyset$.
 2. repeat
 3.      Solve for $\boldsymbol{\lambda}$ and $\boldsymbol{l}$ with respect to $\mathcal{A}$
 4.           by our special technique.
 5.      if ($\boldsymbol{l} \not\geq \mathbf{0}$) then    /* check for feasibility */
 6.           Add some inactive constraints to $\mathcal{A}$.
 7.      else if ($\boldsymbol{\lambda} \not\geq \mathbf{0}$) then    /* check for optimality */
 8.           Remove some active constraints from $\mathcal{A}$.
 9. until ($\boldsymbol{l} \geq \mathbf{0}$ and $\boldsymbol{\lambda} \geq \mathbf{0}$)
```

Theorem 3 *The wire sizing problem (Problem 2, or equivalently, Problem 2′) can be solved by algorithm MASM such that each iteration takes $O(n)$ time.*

The number of iterations of the algorithm depends on how constraints are added to and remove from $\mathcal{A}$. For step 6, we add all the inactive constraints corresponding to negative segment lengths. For step 8, we remove the one corresponding to the most negative λ_j. In Section 5, we will see for this implementation, the number of iterations of our algorithm MASM is less than n in practice.

3 Simultaneous Buffer Insertion and Wire Sizing

In this section, simultaneous buffer insertion and wire sizing will be discussed. m buffers $B_1, \ldots, B_m$ are given and they will be inserted into a wire in this order (with B_1 nearest to the source). A buffer is modeled as a switch-level RC circuit as shown in Figure 2. For convenience, we treat the source and the sink as buffers and we call the source B_0 and the sink B_{m+1}.

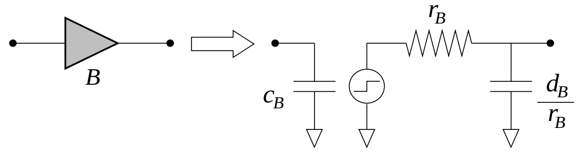


Figure 2: *The model of a buffer B by a switch-level RC circuit. c_B, r_B and d_B are the input capacitance, output resistance and the intrinsic delay of buffer B respectively.*

We want to solve Problem 1 introduced in Section 1. Note that the wire width is not necessary to be decreasing across a buffer. However, the sizing problem of the piece of wire between B_k and B_{k+1} for any k is basically the same as Problem 2 discussed in Section 2 (except that the length of that piece of wire is not fixed). So by Lemma 1, the optimal wire sizing function between two buffers will still be a decreasing step function. Hence, we can approach the problem as before by dividing the piece of wire between every pair of consecutive buffers into n segments of decreasing width, and determining the length of each segment. Instead of having a total length constraint for each piece of wire between buffers, we will have a single constraint specifying that the sum of all segment lengths equals L. See Problem 1′ below.

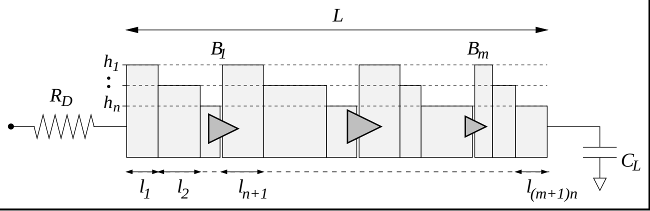
Let $\mathbf{\Phi}'$ be the matrix corresponding to the coefficients of the quadratic terms and $\boldsymbol{\rho}_{\mathbf{k}}$ be the vector corresponding to the coefficients of the linear terms for the delay from B_k to B_{k+1}. Let

$$\mathbf{\Phi} = \begin{pmatrix} \mathbf{\Phi}' & & & 0 \\ & \mathbf{\Phi}' & & \\ & & \ddots & \\ 0 & & & \mathbf{\Phi}' \end{pmatrix} \quad \text{and} \quad \boldsymbol{\rho} = \begin{pmatrix} \boldsymbol{\rho}_0 \\ \boldsymbol{\rho}_1 \\ \vdots \\ \boldsymbol{\rho}_m \end{pmatrix}.$$

PROBLEM 1$'$: The Simultaneous Buffer Insertion and Wire Sizing Problem of Our Approach (equivalent to Problem 1)

Given: wire length L, driver resistance R_D, load capacitance C_L, a set $H = \{h_1, \ldots, h_n\}$ of choices of wire width such that $h_1 > \cdots > h_n$, and m buffers $B_1, \ldots, B_m$.

Determine: the segment lengths $l_i \geq 0$ for $1 \leq i \leq (m+1)n$ such that the delay from source to sink is minimized.



Then the delay from source to sink is

$$D = \frac{1}{2}\boldsymbol{l}^T \boldsymbol{\Phi} \boldsymbol{l} + \boldsymbol{\rho}^T \boldsymbol{l} + \sum_{k=0}^{m} r_{B_k} c_{B_{k+1}} + \sum_{k=1}^{m} d_{B_k}$$

So Problem 1$'$ can be formulated as follows:

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\boldsymbol{l}^T \boldsymbol{\Phi} \boldsymbol{l} + \boldsymbol{\rho}^T \boldsymbol{l} \\ \text{subject to} & l_1 + \cdots + l_{(m+1)n} = L \\ & l_i \geq 0 \text{ for } 1 \leq i \leq (m+1)n \end{array}$$

which is of the same form as (1). $\boldsymbol{\Phi}$ is clearly positive definite as $\boldsymbol{\Phi}'$ is positive definite. Hence the quadratic program is again convex. In addition, for each iteration, we can find $\boldsymbol{l}$ and $\boldsymbol{\lambda}$ as before by reducing the equality constrained program as in (2) to one as in (4). The matrix $\boldsymbol{\Phi}_{\mathcal{A}}$ will be:

$$\boldsymbol{\Phi}_{\mathcal{A}} = \begin{pmatrix} \boldsymbol{\Phi}_{\mathcal{A}_0} & & & 0 \\ & \boldsymbol{\Phi}_{\mathcal{A}_1} & & \\ & & \ddots & \\ 0 & & & \boldsymbol{\Phi}_{\mathcal{A}_m} \end{pmatrix}$$

where $\boldsymbol{\Phi}_{\mathcal{A}_k}$ is the symmetric decomposable matrix corresponding to the set of active constraints for segments between B_k and B_{k+1}. So

$$\boldsymbol{\Phi}_{\mathcal{A}}^{-1} = \begin{pmatrix} \boldsymbol{\Phi}_{\mathcal{A}_0}^{-1} & & & 0 \\ & \boldsymbol{\Phi}_{\mathcal{A}_1}^{-1} & & \\ & & \ddots & \\ 0 & & & \boldsymbol{\Phi}_{\mathcal{A}_m}^{-1} \end{pmatrix}$$

Therefore $\boldsymbol{\Phi}_{\mathcal{A}}^{-1}$ is also tridiagonal as $\boldsymbol{\Phi}_{\mathcal{A}_0}^{-1}, \boldsymbol{\Phi}_{\mathcal{A}_1}^{-1}, \ldots, \boldsymbol{\Phi}_{\mathcal{A}_m}^{-1}$ are all tridiagonal. Hence, Problem 1$'$ can be solved as before.

Theorem 4 *The simultaneous buffer insertion and wire sizing problem (Problem 1, or equivalently, Problem 1$'$) can be solved by algorithm MASM such that each iteration takes $O(mn)$ time.*

We will see in Section 5 that the number of iterations of our algorithm MASM is about n in practice.

4 Extensions

In the following two subsections, we will extend our result for simultaneous buffer insertion and wire sizing to consider wire area (and hence power dissipation), and to handle additional constraints to the solution respectively. We will show that for both extensions, the resulting problem can still be solved by MASM such that each iteration can be done in linear time. Besides, it is easy to see that we can also handle a combination of the two extensions and the resulting program can again be solved by MASM such that each iteration takes linear time.

4.1 Wire Area Consideration

Besides delay, we sometimes want to consider some other objectives as well. In this subsection, we will use wire area as an example and we will consider three cases:

1. *Minimization of a weighted sum of delay and area:* First, note that

$$\text{wire area} = \sum_{k=0}^{m} h_1 l_{kn+1} + \cdots + h_n l_{(k+1)n} = \boldsymbol{h}^T \boldsymbol{l}$$

where $\boldsymbol{h} = (h_1, \ldots, h_n, \ldots\ldots, h_1, \ldots, h_n)^T$. Then the objective will be $\mu(\boldsymbol{l}^T \boldsymbol{\Phi} \boldsymbol{l}/2 + \boldsymbol{\rho}^T \boldsymbol{l}) + (1-\mu)\boldsymbol{h}^T \boldsymbol{l} = (\mu/2)\boldsymbol{l}^T \boldsymbol{\Phi} \boldsymbol{l} + (\mu \boldsymbol{\rho}^T + (1-\mu)\boldsymbol{h}^T)\boldsymbol{l}$ for some weight μ. So the problem can be formulated as follows:

$$\begin{array}{ll} \text{minimize} & (\mu/2)\boldsymbol{l}^T \boldsymbol{\Phi} \boldsymbol{l} + (\mu \boldsymbol{\rho}^T + (1-\mu)\boldsymbol{h}^T)\boldsymbol{l} \\ \text{subject to} & l_1 + \cdots + l_{(m+1)n} = L \\ & l_i \geq 0 \text{ for } 1 \leq i \leq (m+1)n \end{array}$$

Note that this program is of the same form as (1) and hence can be solved by MASM as in Section 2.3.

2. *Delay minimization with bounded area:* The problem can be formulated as follows:

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\boldsymbol{l}^T \boldsymbol{\Phi} \boldsymbol{l} + \boldsymbol{\rho}^T \boldsymbol{l} \\ \text{subject to} & l_1 + \cdots + l_{(m+1)n} = L \\ & \boldsymbol{h}^T \boldsymbol{l} \leq b_{area} \\ & l_i \geq 0 \text{ for } 1 \leq i \leq (m+1)n \end{array}$$

where b_{area} is the area bound. We can solve the program by active set method. If the area constraint is inactive, that iteration can be solved in closed form as before. If the area constraint is active, the matrix $\boldsymbol{\Gamma}_{\mathcal{A}}$ in (4) for that iteration will contain two rows. However, it is clear that it can still be solved in linear time. In fact, it is not difficult to see that if $\mathbf{\Gamma}_{\mathcal{A}}$ has $O(1)$ rows (i.e. we have $O(1)$ nontrivial equalities), the iteration still takes linear time.

3. *Area minimization with bounded delay:*
This case is not as simple since the resulting mathematical program is no longer a quadratic program:

$$\begin{array}{ll} \text{minimize} & \boldsymbol{h}^T\boldsymbol{l} \\ \text{subject to} & \frac{1}{2}\boldsymbol{l}^T\mathbf{\Phi}\boldsymbol{l} + \boldsymbol{\rho}^T\boldsymbol{l} \le b_{delay} \\ & l_1 + \cdots + l_{(m+1)n} = L \\ & l_i \ge 0 \text{ for } 1 \le i \le (m+1)n \end{array}$$

where b_{delay} is the delay bound. We can solve this problem by the Lagrangian relaxation technique as in [1]. The relaxed program will be as follows:

$$\begin{array}{ll} \text{minimize} & \boldsymbol{h}^T\boldsymbol{l} + \lambda(\frac{1}{2}\boldsymbol{l}^T\mathbf{\Phi}\boldsymbol{l} + \boldsymbol{\rho}^T\boldsymbol{l} - b_{delay}) \\ \text{subject to} & l_1 + \cdots + l_{(m+1)n} = L \\ & l_i \ge 0 \text{ for } 1 \le i \le (m+1)n \end{array}$$

where λ is the Lagrange multiplier. It is again of the form as (1) and hence can be solved by MASM.

4.2 Additional Constraints

Sometimes, we may want to have some constraints on the solution. We can do this by adding constraints to the convex quadratic program. For example, we may require that the section of the wire within a distance l' from the sink cannot be wider than h'. If t is the index such that $h_t > h' \ge h_{t+1}$, then the corresponding program will be:

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\boldsymbol{l}^T\mathbf{\Phi}\boldsymbol{l} + \boldsymbol{\rho}^T\boldsymbol{l} \\ \text{subject to} & l_1 + \cdots + l_{mn+t} \le L - l' \\ & l_1 + \cdots + l_{(m+1)n} = L \\ & l_i \ge 0 \text{ for } 1 \le i \le (m+1)n \end{array}$$

As we mentioned above, as long as $O(1)$ constraints are added, the resulting program can be handled by MASM such that each iteration can be done in linear time.

5 Experimental Results

In this section, we will show that the algorithm MASM is efficient in practice. We have implemented the version for delay minimization by simultaneous buffer insertion and wire sizing in C. We run it on an IBM PowerPC 25 using several different values for the number of choices of wire width n and for the number of buffers m. For each pair of values of n and m, we run our algorithm on 100 instances with R_D, C_L, $B_1, \ldots, B_m$, H, and L being randomly generated. The average number of iterations and CPU time over the 100 instances are reported in Table 1.

Nowadays, the values of n and m that actually used are usually less than 10. So the running time is negligible. Even for an instance of 100 choices of wire width and 100 buffers, the algorithm still takes only 3 seconds. An interesting observation is that the number of iterations is about n (except when $m = 0$) and is basically independent of m. So for Problem 1$'$, algorithm MASM runs in $O(mn^2)$ time in practice.

# width choices	# buffers	# variables	Algorithm MASM	
n	m	$(m+1)n$	# iter.	CPU(s)
10	0	10	3.93	0.00
10	10	110	11.34	0.00
10	40	410	12.74	0.01
10	70	710	13.18	0.03
10	100	1010	13.31	0.04
40	0	40	15.79	0.00
40	10	440	40.19	0.05
40	40	1640	41.89	0.20
40	70	2840	42.36	0.35
40	100	4040	42.64	0.49
70	0	70	31.22	0.01
70	10	770	68.39	0.15
70	40	2870	71.01	0.60
70	70	4970	71.65	1.05
70	100	7070	72.24	1.53
100	0	100	42.61	0.01
100	10	1100	96.37	0.31
100	40	4100	100.16	1.24
100	70	7100	101.21	2.14
100	100	10100	101.69	3.06

Table 1: *The average number of iterations and CPU time of the algorithm MASM for simultaneous buffer insertion and wire sizing.*

There are also many public domain or commercial software systems that can solve convex quadratic programs (e.g. LOQO, CPlex, OSL, MINO). LOQO is one of the fastest system available. So we compare the running time of LOQO with our algorithm MASM. We notice that MASM, being based on the observations in Section 2.3, is much faster than LOQO. For small problems ($n = 10$ and $m = 10$), our algorithm is about 15 times faster. For larger problems ($n = 100$ and $m = 100$), our algorithm is more than 30 times faster.

6 Discussion

Tree topology: For weighted delay objective, our algorithm can be applied to handle nets with tree topology by a similar technique as in [5]. That is we use an iterative algorithm to optimize the tree edges one at a time. At each time we manipulate an edge, we keep all the other edges fixed and apply our algorithm to that edge. For other objectives like minimizing maximum delay or minimizing area with delay bounds, we can apply the Lagrangian relaxation technique as in [5] to reduce the problems to a problem of minimizing weighted delay. For these cases, we need to fix the number of buffers used on each wire beforehand. If only the total number of buffers to be used is given, a possible approach would be to combine dynamic programming with our algorithm to distribute the buffers among the edges.

Simultaneous buffer insertion, buffer sizing and wire sizing: If choices of buffers of several different sizes are allowed, we can find the optimal solution by trying all possible combinations of buffer sizes. For each combination, we can use our algorithm to handle buffer insertion and wire sizing. As our algorithm is really fast, the total running time will still be very short. For example, if we have 10 choices of wire width, 6 choices of buffer size and 4 buffers to be inserted ($6^4 = 1296$ combinations), the total running time to find the optimal solution is only 2.05 seconds. So our algorithm can even be used for simultaneous buffer insertion, buffer sizing and wire sizing in practice.

Note that the number of segments by our approach is independent of the wire length. For the example above, our problem has only $(4+1) \times 10 = 50$ segments. For the traditional approach, in order to obtain comparable accuracy, a longer wire needs to be divided into more segments. For a wire of $3000\mu m$ long, if we divide it into $1\mu m$ segments, we will have 3000 segments. The running time will be long if we use dynamic programming to solve this problem.

Of course, it would be nice if we can have better technique to handle buffer sizing than trying all possibilities.

References

[1] Chung-Ping Chen, Yao-Wen Chang, and D. F. Wong. Fast performance-driven optimization for buffered clock trees based on Lagrangian relaxation. In *Proc. IEEE ICCAD*, pages 405–408, 1996.

[2] Chung-Ping Chen, Yao-Ping Chen, and D. F. Wong. Optimal wire-sizing formula under the Elmore delay model. In *Proc. ACM/IEEE DAC*, pages 487–490, 1996.

[3] Chung-Ping Chen and D. F. Wong. A fast algorithm for optimal wire-sizing under Elmore delay model. In *Proc. IEEE ISCAS*, volume 4, pages 412–415, 1996.

[4] Chung-Ping Chen and D. F. Wong. Optimal wire-sizing function with fringing capacitance consideration. In *Proc. ACM/IEEE DAC*, 1997.

[5] Chung-Ping Chen, Hai Zhou, and D. F. Wong. Optimal non-uniform wire-sizing under the Elmore delay model. In *Proc. IEEE ICCAD*, pages 38–43, 1996.

[6] Chris C. N. Chu and D. F. Wong. Closed form solution to simultaneous buffer insertion/sizing and wire sizing. In *Proc. Intl. Symp. on Physical Design*, pages 192–197, 1997.

[7] Jason Cong and Lei He. An efficient approach to simultaneous transistor and interconnect sizing. In *Proc. IEEE ICCAD*, pages 181–186, 1996.

[8] Jason Cong and Lei He. Optimal wiresizing for interconnects with multiple sources. *ACM Trans. Design Automation of Electronic Systems*, 1(4), October 1996.

[9] Jason Cong, Lei He, Cheng-Kok Koh, and Patrick H. Madden. Performance optimization of VLSI interconnect layout. *INTEGRATION, the VLSI Journal*, 21:1–94, 1996.

[10] Jason Cong and Cheng-Kok Koh. Simultaneous buffer and wire sizing for performance and power optimization. In *Proc. Intl. Symp. on Low Power Electronics and Design*, pages 271–276, August 1996.

[11] Jason Cong and Kwok Shing Leung. Optimal wiresizing under the distributed Elmore delay model. *IEEE Trans. on CAD*, 14(3):321–336, March 1995.

[12] Jim DeTar. Advances outpace SIA roadmap (Semiconductor Industry Association alters projections) (Industry Trend or Event). *Electronic News*, 42(2147):1, December 16 1996.

[13] W. C. Elmore. The transient response of damped linear network with particular regard to wideband amplifiers. *J. Applied Physics*, 19:55–63, 1948.

[14] J. P. Fishburn. Shaping a VLSI wire to minimize Elmore delay. In *Proc. European Design and Test Conference*, 1997.

[15] J. P. Fishburn and C. A. Schevon. Shaping a distributed-RC line to minimize Elmore delay. *IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications*, 42(12):1020–1022, December 1995.

[16] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, NY, 1979.

[17] M. K. Kozlov, S. P. Tarasov, and L. G. Khachiyan. Polynomial solvability of convex quadratic programming. *Soviet Mathematics Doklady*, 20:1108–1111, 1979.

[18] John Lillis, Chung-Kuan Cheng, and Ting-Ting Y. Lin. Optimal wire sizing and buffer insertion for low power and a generalized delay model. *IEEE Journal of Solid State Circuits*, 31(3):437–447, March 1996.

[19] D. G. Luenberger. *Linear and Nonlinear Programming.* Addison Wesley, second edition, 1984.

[20] N. Menezes, R. Baldick, and L. T. Pileggi. A sequential quadratic programming approach to concurrent gate and wire sizing. In *Proc. IEEE ICCAD*, pages 144–151, 1995.

[21] Sachin S. Sapatnekar. RC interconnect optimization under the Elmore delay model. In *Proc. ACM/IEEE DAC*, pages 387–391, 1994.

[22] L. P. P. P. van Ginneken. Buffer placement in distributed RC-tree networks for minimal Elmore delay. In *Proc. Intl. Symp. on Circuits and Systems*, pages 865–868, 1990.

[23] Qing Zhu, Wayne W.-M. Dai, and Joe G. Xi. Optimal sizing of high-speed clock networks based on distributed RC and lossy transmission line models. In *Proc. IEEE ICCAD*, pages 628–633, 1993.