# Constructing Lower and Upper Bounded Delay Routing Trees Using Linear Programming*

Jaewon Oh[1], Iksoo Pyo[2] and Massoud Pedram[1]

[1]University of Southern California
Department of Electrical Engineering
Los Angeles, CA 90089
joh@danube.usc.edu, massoud@zugros.usc.edu

[2]Intel Corporation
JF1-61
Hillsboro, OR 97124
ipyo@ichips.intel.com

## Abstract

*This paper presents a new approach for solving the Lower and Upper Bounded delay routing Tree (LUBT) problem using linear programming. LUBT is a Steiner tree rooted at the source node such that delays from the source to sink nodes lie between the given lower and upper bounds. We show that our proposed method produces minimum cost LUBT for a given topology under a linear delay model. Unlike recent works which control only the difference between the maximum and the minimum source-sink delay, we construct routing trees which satisfy distinct lower and upper bound constraints on the source-sink delays. This formulation exploits all the flexibility that is present in low power and high performance clock routing tree design.*

## 1 Introduction

Routing affects various aspects of design such as chip area, performance and power dissipation. In the performance driven global routing problem, the routing cost is minimized while the maximum delay from the source to any sink is kept within a given bound. In the zero skew clock routing problem, the routing cost is minimized while the skew of the routing tree, which is the difference between the minimum and maximum delay from the source to any sink, is made zero. In practice, exact zero skew is not an actual design requirement. We can allow some tolerable skew with which the system can function correctly. Bounded skew clock routing methods are presented in [4] and [5] to reduce the routing cost over zero skew routing. These methods however only consider the skew bound and do not control the maximum source-sink delay. Long wires require more buffers and cause slower rise and fall time. More buffers and slower switching result in higher power dissipation. A power optimizing clock routing algorithm with bounded skew and bounded maximum source-sink delay under the Elmore delay model is presented in [6]. In this algorithm, delays are controlled by buffer sizing rather than by controlling the wire lengths, the clock tree is an equal source-sink path length Steiner tree (zero skew tree under linear delay) regardless of skew bounds, and finally the routing cost may become large when non-zero skew is required.

*This research was funded in part by ARPA under contract No. F33615-95-C1627 and by NSF NYI under contract No. MIP-9457392.

Our proposed method allows the user to specify different delay bounds for each individual sink, which can lead to a further reduction of the routing cost. In the case of clock routing, clock arrival time to each flip/flop (FF) can be made different while in the case of global routing, the required signal arrival time among sinks are made different. In addition, if the combinational delay between two FFs violates the short path delay constraint, common practice is to insert a delay element on the short path. Instead, one can increase the wire length to meet the short path delay constraint. Since nowadays routing delays dominate gate delays, wire-length control is a more effective mean of introducing delays compared to adding buffers. However, one cannot arbitrarily increase the length a wire since it may violate required arrival time of other sinks. These observations motivated us to develop a method for controlling the path lengths such that any delays lie between given upper and lower bounds.

Mathematical programming in Manhattan metric has been rarely used in practice. The reason is that the Manhattan distance between two points $(x_1, y_1)$, $(x_2, y_2)$ is $|x_1-x_2|+|y_1-y_2|$ and a mathematical programming problem with these absolute valued functions is not easy to solve. These functions are not differentiable and the correct sign of absolute valued terms in the formulation, whether they appear in the objective function or in the constraints, must be maintained during the search for a solution. This slows down the optimization speed significantly. Many researchers have thus replaced Manhattan distance with the less accurate Euclidean or Quadratic distance or other approximations.

Our method, Edge-Based Formulation (EBF) overcomes this problem. Variables of the mathematical programming are not the positions of the Steiner points. Instead, the variables are edge lengths of the tree, eliminating any absolute valued terms in the formulation. The proposed formulation leads to a simple linear programming problem under the linear delay model which can be solved optimally in polynomial time. Once the edge lengths are determined, the position of Steiner points are determined from geometric considerations.

## 2 Terminology and Problem Definition

Let $T(S, E)$ be a given rooted tree topology. Let $S = \{s_0, s_1, s_2, \ldots, s_n\}$ be the vertices of $T$. Among these vertices, $s_0$ is the root (source) of $T$ whose location may or may not be

**33rd Design Automation Conference ®**

DAC 96 - 06/96 Las Vegas, NV, USA

given. $\{s_1, s_2, \ldots, s_m\}$ are the sinks whose locations are given and $\{s_{m+1}, \ldots, s_n\}$ are the Steiner points whose locations are to be determined. The term *points* refer to the source, sinks and Steiner points. Since $T$ is a tree, there is a unique path between any two points. We say $s_i$ is the *parent* of $s_j$ if in the path from $s_0$ to $s_j$, $s_i$ comes immediately before $s_j$. Inversely, $s_j$ is a child of $s_i$.

Let $E = \{e_1, e_2, \ldots, e_n\}$ be the set of edges in the tree. We associate each point $s_i$, except $s_0$, of the rooted topology $T$ with edge $e_i$ that connects $s_i$ to its parent in $T$. When there is no confusion, we use $e_i$ to refer to the $i$th edge or the *edge length* of the $i$th edge.

Let $dist(s_i, s_j)$ be the Manhattan distance between $s_i$ and $s_j$. The cost of a tree $T$ is the sum of the edge lengths, i.e. $cost(T) = \sum_{i=1}^{n} e_i$. Let $path(s_i, s_j)$ be the set of edges in the path from $s_i$ to $s_j$ in $T$. We define the *delay of a sink* $s_i$ as the delay from the source to that sink and denote it as $delay(s_i)$. Since we are using a linear delay model, $delay(s_i)$ is defined by:

$$delay(s_i) = \sum_{e_k \in \; path(s_0, s_i)} e_k \tag{1}$$

Under a linear delay model, *diameter* is the distance between the farthest two sinks. If the source location is not given, the *radius* is defined as half of the diameter. Otherwise, radius is the distance from the source to the farthest sink. We now define *Lower/Upper Bounded routing Tree* (LUBT) problem.

**Definition 2.1 Lower/Upper Bounded routing Tree (LUBT) Problem:** *Given a rooted tree topology T(S,E) and two sets of bounds* $L = \{l_1, l_2, \ldots, l_m\} \subset \mathcal{R}$, $U = \{u_1, u_2, \ldots, u_m\} \subset \mathcal{R}$, *find a tree embedding in the Manhattan plane, i.e., find the locations of Steiner points and the values of* $e_i$*'s, such that the tree cost is minimal and the delay from the source* $s_0$ *to any sink* $s_i$ *satisfies the following inequalities:*

$$l_i \leq delay(s_i) \leq u_i \qquad i = 1, \ldots, m \tag{2}$$

*where* $l_i$*'s and* $u_i$*'s satisfy the following:*

$$0 \leq l_i \leq u_i \quad and \quad u_i \geq dist(s_0, s_i) \tag{3}$$

$$or \quad 0 \leq l_i \leq u_i \quad and \quad u_i \geq radius. \tag{4}$$

*Equation 3 holds when the source location is given while Equation 4 holds when this location is not given.*

It can be shown that for a given topology $T$, the solution to a LUBT problem may not exist depending on the bounds. However, if every sink is a leaf node in the topology, then, under the linear delay model, it is always possible to find a LUBT for any lower and upper bounds given by Equation (3) or (4) [8].

## 3 Edge-Based Formulation(EBF)

We present an EBF formulation for the LUBT problem. The edge lengths are determined such that they satisfy both the *Steiner constraints* and the *delay constraints* as described next.

### 3.1 Steiner Constraints

When we determine the edge lengths, it is important that there exist valid locations for Steiner points that achieve those edge lengths. The following is a necessary condition for edge lengths.

$$\sum_{e_k \in \; path(s_i, s_j)} e_k \geq dist(s_i, s_j) \qquad \text{for every pair of sinks } s_i, s_j \tag{5}$$

Otherwise, the two sinks $s_i, s_j$ will get separated, breaking the tree into two components. The above equation is also a sufficient condition as described next. All proofs can be found in [8].

**Theorem 3.1** *Let* $e_1^*, \ldots, e_n^*$ *be a solution to the following set of linear inequalities.*

$$\sum_{e_k \in \; path(s_i, s_j)} e_k \geq dist(s_i, s_j) \quad \textit{for every pair of sink } s_i, s_j \tag{6}$$

*Then there exist placements of steiner points* $s_{m+1}, \ldots, s_n$ *such that*

$$e_k^* \geq dist(s_k, s_p) \qquad k = 1, \ldots, n \tag{7}$$

*where* $s_p$ *is the parent of* $s_k$*.*

### 3.2 Delay constraints

The delay constraints dictate that the delays from the source to any sink are bounded. Under the linear delay model, we have:

$$l_i \leq \sum_{e_k \in \; path(s_0, s_i)} e_k \leq u_i \qquad \text{for all sinks } s_i \tag{8}$$

### 3.3 Summary of the Formulation

Our objective is to minimize the total sum of edge lengths. Together with the Steiner constraints and the delay constraints, we have the following mathematical formulation.

$$\begin{array}{ll} \text{Min} & \sum_{k=1}^{n} e_k \\ \text{Subject to} & \sum_{e_k \in \; path(s_i, s_j)} e_k \geq dist(s_i, s_j) \quad \forall \text{ sinks } s_i, s_j \\ & l_i \leq delay(s_i) \leq u_i \qquad \forall \text{ sinks } s_i \end{array} \tag{9}$$

### 3.4 Optimality of Our Method

Our method constructs minimum cost LUBT for a given topology since it uses mathematical programming.

**Theorem 3.2** *Our method gives minimum cost LUBT for a given topology.*

### 3.5 An example

Consider the tree topology of Figure 1. We want to find a LUBT with a lower bound of 4 and an upper bound of 6 for all the sinks. Assuming the source position is not given, we have the following formulation.

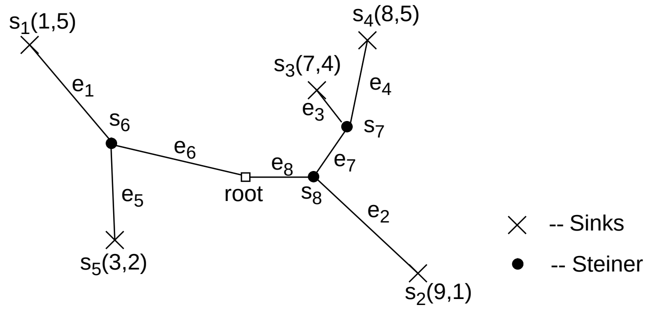


Figure 1: A 5 point example

$$
\begin{array}{ll}
\text{Min} & e_1+e_2+e_3+e_4+e_5+e_6+e_7+e_8 \\
\text{Subject to} & \\
& \left.\begin{array}{l}
e_1+e_6+e_8+e_2 \geq 12 \\
e_1+e_6+e_8+e_7+e_3 \geq 7 \\
e_1+e_6+e_8+e_7+e_4 \geq 7 \\
e_1+e_5 \geq 5 \\
e_2+e_7+e_3 \geq 5 \\
e_2+e_7+e_4 \geq 5 \\
e_2+e_8+e_6+e_5 \geq 7 \\
e_3+e_4 \geq 2 \\
e_3+e_7+e_8+e_6+e_5 \geq 6 \\
e_4+e_7+e_8+e_6+e_5 \geq 8
\end{array}\right\}\text{Steiner Constraints} \\
& \left.\begin{array}{l}
4 \leq e_1+e_6 \leq 6 \\
4 \leq e_2+e_8 \leq 6 \\
4 \leq e_3+e_7+e_8 \leq 6 \\
4 \leq e_4+e_7+e_8 \leq 6 \\
4 \leq e_5+e_6 \leq 6
\end{array}\right\}\text{Linear delay Constraints}
\end{array}
$$

## 4 Placement of Steiner points

Once the edge lengths are determined, the actual positions of Steiner points (and the root if its position is not given) should be determined. Our method for placement of Steiner points is similar to the *Deferred Merge Embedding* (DME) [3] algorithm exploited by most zero skew and bounded skew clock routing algorithms. In the DME algorithm, the feasible regions for Steiner points and the edge lengths are found in a bottom up fashion, and then Steiner points are placed in the feasible regions in a top down fashion. Our method is different in that edge lengths are predetermined and the feasible regions are rectangular regions instead of simple line segments (in case of zero skew algorithms) or octilinear regions (in case of bounded skew algorithms). Details can be found in [8].

## 5 Extensions of EBF to other problems

In this section, we consider extensions of the EBF method to some interesting problems. In all cases, the Steiner constraints remain the same. We need only make modifications to the delay constraints or the objective function to obtain other problems.

### Using Elmore delay

The Elmore delay is defined as follows. Let $T_k$ be the subtree of the routing tree rooted at $s_k$. We use $C_k$ to denote the total tree capacitance at $s_k$, namely the sum of sink and edge capacitances of $T_k$. Let the unit resistance and unit capacitance of routing wire be $r_w$ and $c_w$, respectively. Then the delay at a sink $s_j$ is defined by:

| bench | [5] skew bound | shortest delay | longest delay | tree cost | LUBT tree cost |
|---|---|---|---|---|---|
| prim1 | 0.000 | 1.000 | 1.000 | 132565.0 | 132539.75 |
| | 0.010 | 0.995 | 1.005 | 130060.2 | 129872.23 |
| | 0.100 | 0.910 | 1.020 | 113805.0 | 112887.03 |
| | 0.500 | 0.648 | 1.148 | 93650.0 | 93647.38 |
| | 1.000 | 0.439 | 1.439 | 84915.0 | 84915.00 |
| | ∞ | 0.000 | ∞ | 79810.0 | 79810.00 |
| prim2 | 0.000 | 1.000 | 1.000 | 315630.0 | 315628.20 |
| | 0.010 | 0.990 | 1.000 | 305332.0 | 303963.30 |
| | 0.100 | 0.954 | 1.054 | 251540.0 | 249448.30 |
| | 0.500 | 0.741 | 1.241 | 206140.0 | 205783.60 |
| | 1.000 | 0.382 | 1.382 | 182490.0 | 182457.20 |
| | ∞ | 0.000 | ∞ | 173200.0 | 173200.00 |
| r1 | 0.000 | 1.000 | 1.000 | 1312498.0 | 1311913.38 |
| | 0.020 | 0.996 | 1.023 | 1356429.8 | 1343863.00 |
| | 0.100 | 0.947 | 1.032 | 1797884.9 | 1750177.80 |
| | 0.500 | 0.714 | 1.214 | 932256.5 | 931271.69 |
| | 1.000 | 0.444 | 1.444 | 848555.5 | 847653.00 |
| | ∞ | 0.000 | ∞ | 780100.0 | 780100.25 |
| r3 | 0.000 | 1.000 | 1.000 | 3331097.5 | 3330921.00 |
| | 0.010 | 0.996 | 1.006 | 3227565.5 | 3212405.00 |
| | 0.100 | 0.918 | 1.018 | 2732820.5 | 2709491.30 |
| | 0.500 | 0.741 | 1.241 | 2261973.0 | 2254820.50 |
| | 1.000 | 0.566 | 1.566 | 2137096.0 | 2135432.00 |
| | ∞ | 0.000 | ∞ | 1929421.0 | 1929421.00 |

All bounds are normalized to the radius.

Table 1: Routing costs for [5] and for the LUBT method

$$delay(s_j) = \sum_{e_k \in\, path(s_0, s_j)} r_w e_k \left(\frac{c_w e_k}{2} + C_k\right). \qquad (10)$$

The delay equation is quadratic with respect to $e_k$'s (Note that $C_k$ itself is also a function of edge lengths). Since the Elmore delay function is quadratic and the sum of the quadratic terms is positive (i.e. the function is *posynomial* in $e_k$), the delay function is strictly convex. The feasible set defined by a convex function with both lower and upper bounds is however not a convex set. So the EBF with Elmore delay constraints is not a convex programming problem. However, if we don't impose the lower bounds ($l_i = 0$), then our formulation will remain a convex programming problem.

### Different weights on edges

In the EBF method, the objective is the cost of the tree where each edge is equally weighted. However, some edges may be given higher weights to account for wireability concerns, blockage, type of metals used, crosstalk or switching activities. In that case, we can give different weights $w_1, w_2, \ldots, w_n$ to edges in the objective function. The resulting problem is still a linear programming problem.

## 6 Experimental Results

The EBF is a Linear Programming problem which can be solved efficiently using a number of commercially available LP solvers. Especially we have chosen LOQO [7] as our solver. LOQO uses the *interior point method* which is known to be faster than Simplex method for large problems. We have implemented our algorithm in C for SPARC and HPPA workstations. To reduce the complexity of the problem, some techniques are used to reduce the number of Steiner constraints [8]. Those techniques are based on geometric considerations.

The topology generator is taken from [5]. This topology generator is based on the nearest neighbor merge technique

| bench | skew bound | lower bound | upper bound | tree cost |
|---|---|---|---|---|
| prim1 | 0.3 | 0.70 | 1.00 | 103219.5 |
| | | 0.80 | 1.10 | 102122.9 |
| | | *0.89 | *1.19 | 103051.8 |
| | | 0.95 | 1.25 | 103671.0 |
| | 0.5 | 0.50 | 1.00 | 98120.7 |
| | | 0.60 | 1.10 | 93152.0 |
| | | *0.65 | *1.15 | 93647.4 |
| | | 0.75 | 1.25 | 94700.0 |
| prim2 | 0.3 | 0.70 | 1.00 | 247834.4 |
| | | 0.80 | 1.10 | 237720.3 |
| | | *0.85 | *1.15 | 225650.0 |
| | | 0.95 | 1.25 | 230756.0 |
| | 0.5 | 0.50 | 1.00 | 212068.8 |
| | | 0.60 | 1.10 | 211034.6 |
| | | *0.74 | *1.24 | 205783.6 |
| | | 0.85 | 1.35 | 207344.5 |

All bounds are normalized to the radius.
*: bounds produced by [5].

Table 2: Routing cost of LUBT for the same skew but different upper bounds

and dynamically changes the topology during the construction phase based on the skew. The topologies are full binary trees in which every sink is a leaf node. Therefore as hinted in Section 2, a solution will exist for any lower and upper bound constraints. We tested our method on benchmark data prim1, prim2 [1] and r1, r3 [2]. Our results are compared to those reported in [5] in Table 1. Algorithm of [5] produces optimal solutions for infinite skew bounds and suboptimal solutions for finite skew bounds. Since [5] accepts only the skew bounds and does not allow the user to specify lower/upper bounds, we first ran their algorithm with a skew bound and extracted the topology and the actual shortest/longest sink delays from the solution. Then we ran our algorithm with those shortest/longest sink delays as our lower/upper bounds of LUBT for the same topology.

To show that our algorithm can produce different lower/upper bounds for the same skew, prim1 and prim2 were tested. Results are shown in Table 2. Note that for the same skew, the longest delay can be reduced with little increase in the tree cost. Trees with zero lower bounds and some finite upper bounds are useful for global routing. Table 3 shows results for some other interesting bound combinations useful for global routings and bounded skew - bounded longest delay routings. Note also that as the skew bound is tightened, the tree cost increases.

## 7 Conclusion

We proposed a new method for solving lower/upper bounded delay routing tree (LUBT) problems. The method is based on linear programming in which variables are the edge lengths of the tree. The LUBT problem is a generalization of global routing and clock routing. Our method produces an optimal LUBT for a given topology under the linear delay model. Due to optimality of our method, we can immediately know the existence of a solution for a given topology and bounds since, in case there is no solution, there will be no initial feasible solution to EBF.

Implementation of the EBF method under the Elmore delay model is currently being investigated. Under the Elmore delay, the optimality of the LUBT cost is assured only when the lower bound is zero. When the lower bound is not zero, a sequential quadratic optimization is needed to solve the EBF.

| bench | lower bound | upper bound | tree cost |
|---|---|---|---|
| prim1 | 0.99 | 1.00 | 129818.3 |
| | 0.95 | 1.00 | 121833.6 |
| | 0.90 | 1.00 | 113728.9 |
| | 0.50 | 1.00 | 98120.7 |
| | 0.00 | 1.00 | 97234.1 |
| | 0.00 | 2.00 | 79840.0 |
| prim2 | 0.99 | 1.00 | 304058.7 |
| | 0.95 | 1.00 | 269495.2 |
| | 0.90 | 1.00 | 248388.0 |
| | 0.50 | 1.00 | 212068.8 |
| | 0.00 | 1.00 | 213276.0 |
| | 0.00 | 2.00 | 173300.0 |
| r1 | 0.99 | 1.00 | 1284095.9 |
| | 0.95 | 1.00 | 1218575.8 |
| | 0.90 | 1.00 | 1215419.9 |
| | 0.50 | 1.00 | 963928.4 |
| | 0.00 | 1.00 | 1099360.8 |
| | 0.00 | 2.00 | 780288.8 |
| r3 | 0.99 | 1.00 | 3211281.5 |
| | 0.95 | 1.00 | 2924382.0 |
| | 0.90 | 1.00 | 2707221.8 |
| | 0.50 | 1.00 | 2374080.0 |
| | 0.00 | 1.00 | 2197381.0 |
| | 0.00 | 2.00 | 2025446.0 |

All bounds are normalized to the radius.

Table 3: Routing cost of LUBT for various other bounds

Our method requires an input tree topology. The topology generator we have taken from [5] uses the amount of skew to guide the topology generation, rather than the explicit lower/upper bounds. So future work will include better topology generation which is guided by both lower and upper bounds, and at the same time, results in lower tree cost.

Finally, the EBF method is a general-purpose approach for solving optimization problem in Manhattan space. We are also considering an application of the EBF to the placement/floor-planning problems.

## References


[1] M. A. B. Jackson, A. Srinivasan, and E. S. Kuh, "Clock routing for high-performance ICs," *27th Design Automation Conference* , pp. 573-579, 1990.

[2] R-S Tsay, "Exact zero skew," *International Conference on Computer-Aided Design* , pp. 336-339, 1991.

[3] K. D. Boese and A. B. Kahng, "Zero-Skew Clock Routing Trees With Minimum Wirelength," *Proc. IEEE International Conference on ASIC*, pp. 1.1.1-1.1.5, 1992.

[4] J. Cong and C-K. Koh, "Minimum-Cost Bounded-Skew Clock Routing," *International Symposium on Circuits and Systems*, pp. 215-218, 1995.

[5] D. J.-H. Huang, A. B. Kahng and C-W. Tsao, "On the Bounded-Skew Clock and Steiner Routing Problems," *32nd Design Automation Conference*, pp. 508-513, 1995.

[6] J. Xi, W-M. Dai, "Buffer Insertion and Sizing Under Process Variations for Low Power Clock Distribution," *32nd Design Automation Conference*, pp. 491-496, 1995

[7] R. J. Vanderbei, "LOQO User's Manual," *Program and the manual are available free for academic users at ftp://elib.zib-berlin.de/pub/opt-net/software/loqo.*

[8] J. Oh, I. Pyo and M. Pedram, "Contructing Lower and Upper Bounded Delay Routing Trees Using Linear Programmin," *USC Electrical Engineering Department - Systems, Technical Report 96-05*, Mar. 1996.