[35] C. Sechen and K.-W. Lee, "An improved simulated annealing algorithm for row-based placement," in *Dig. Papers, Int. Conf. Computer-Aided Design*, Santa Clara, CA, Nov. 1987, pp. 478–481.
[36] E. H. L. Aarts, F. M. J. de Bont, E. H. A. Habers, and P. J. M. van Laarhoven, "Parallel implementations of the statistical cooling algorithm," *Integr. VLSI J.*, vol. 4, pp. 209–238, Sept. 1986.
[37] A. Casotto, F. Romeo, and A. Sangiovanni-Vincentelli, "A parallel simulated annealing algorithm for the placement of macro-cells," *IEEE Trans. Computer-Aided Design*, vol. CAD-6, pp. 838–847, Sept. 1987.
[38] A. Casotto and A. Sangiovanni-Vincentelli, "Placement of standard cells using simulated annealing on the connection machine," in *Dig. Papers, Int. Conf. Computer-Aided Design*, Santa Clara, CA, Nov. 1987, pp. 350–353.
[39] C.-P. Wong and R.-D. Fiebrich, "Simulated annealing-based circuit placement algorithm on the connection machine system," in *Proc. Int. Conf. Computer Design*, Rye Brook, NY, Oct. 1987, pp. 78–82.
[40] W.-J. Sun and C. Sechen, "A loosely coupled parallel algorithm for standard cell placement," in *Dig. Papers, Int. Conf. Computer-Aided Design*, San Jose, CA, Nov. 1994, pp. 137–144.
[41] E. H. L. Aarts and P. J. M. van Laarhoven, "Simulated annealing: Theory and applications," in *Mathematics and Its Applications.* Boston, MA: Kluwer, 1987.
[42] K.-G. Lee and S.-Y. Lee, "Efficient parallelization of simulated annealing using multiple Markov chains: An application to graph partitioning," in *Proc. Int. Conf. Parallel Processing*, St. Charles, IL, Aug. 1992, pp. III:177–III:180.
[43] A. Sohn, "Parallel speculative computation of simulated annealing," in *Proc. Int. Conf. Parallel Processing*, St. Charles, IL, Aug. 1994, pp. III:8–III:11.

Performance-Driven Routing with Multiple Sources

Jason Cong and Patrick H. Madden

***Abstract*— Existing routing problems for delay minimization consider the connection of a *single* source node to a number of sink nodes, with the objective of minimizing the delay from the source to all sinks, or a set of critical sinks. In this paper, we study the problem of routing nets with *multiple* sources, such as those found in signal busses. This new model assumes that each node in a net may be a source, a sink, or both. The objective is to optimize the routing topology to minimize the total weighted delay between *all* node pairs (or a subset of critical node pairs). We present a heuristic algorithm for the multiple-source performance-driven routing tree problem based on efficient construction of minimum-diameter minimum-cost Steiner trees. Experimental results on random nets with submicrometer CMOS IC and MCM technologies show an average of 12.6% and 21% reduction in the maximum interconnect delay, when compared with conventional minimum Steiner tree based topologies. Experimental results on multisource nets extracted from an Intel processor show as much as a 16.1% reduction in the maximum interconnect delay, when compared with conventional minimum Steiner tree based topologies.**

***Index Terms*— Interconnections, interconnect topology optimization, layout, minimum diameter routing tree, multisource routing tree, rectilinear arborescence, Steiner tree.**

Manuscript received May 8, 1995; revised April 29, 1996 and December 17, 1996. This work was supported in part by DARPA/ITO under Contract J-FBI-93-112, an NSF Young Investigator Award MIP9357582, and a grant from Intel Corporation. This paper was recommended by Associate Editor C.-K. Cheng.

The authors are with the Computer Science Department, University of California, Los Angeles, CA 90024-1596 USA.

Publisher Item Identifier S 0278-0070(97)05151-8.

I. Introduction

The competitive nature of the very large scale integration (VLSI) industry has created a strong demand for techniques to improve the performance of integrated circuits. Methods to increase speed, and to reduce area or power consumption, are of great interest.

Scaling of device dimensions has resulted in changes to many fundamental design goals: where previously the bulk of system delay had been generated by the switching times of devices, it is now common that the interconnecting wires between devices accounts for the dominating portion of the delay. These changes have created new areas in need of optimization, and new measures by which we gauge solution quality.

With smaller minimum feature size comes a reduction in transistor channel width and length, resulting in relatively constant transistor on resistance; the reduction in wire width, on the other hand, results in higher unit wire resistance [2]. As a result, the *resistance ratio* [8], defined to be the driver resistance divided by the unit length wire resistance, is reduced significantly. This shift produces a situation where the length of the path between a driver and sink can have comparable resistance to that of the transistor channel. Thus, changes to the interconnect length and topology can have a significant impact on delay. The result in [11] showed convincingly that interconnect topology optimization has a considerable effect on interconnect delay reduction when the resistance ratio is small.

A number of optimized interconnect topologies have been proposed, including bounded-radius bounded-cost trees [9], AHHK trees [1], LAST trees [21], maximum performance trees [7], A-trees [11], low-delay trees [5], and IDW/CFD trees [18]. These methods consider both the traditional concern of low total wire length, and also the path length or Elmore delay between the source node and the timing-critical sink nodes.

Although many of these methods effectively reduce the interconnect delay, all of them assume that there is a single source node driving one or more sink nodes and minimize the delay from the unique source to all sinks, or a set of critical sinks.

In practice, many timing-critical nets may have multiple sources, each of them controlled by a tri-state gate and driving the net at a different time. Signal busses are instances of such nets. In these cases, the existing performance-driven routing algorithms for single source nets may perform poorly, as a topology optimized for one source may result in high interconnect delay when some other source becomes active.

Fig. 1 presents a pair of four-node routing trees with the same wire length. The first routing tree, optimized for node p_1, has relatively high delay when node p_2 drives the net. The second routing tree provides a lower overall maximum delay when all four nodes might be sources or sinks. Delay times with respect to the driving nodes are shown in Table I.

Note that the second routing tree, which minimizes the maximum linear delay, does not fall entirely on the Hanan grid [16]. For the single source model under Elmore delay, [4] showed that an optimal tree which minimizes the maximum delay to any sink may not be contained by the Hanan grid, but also observed that these cases were rare. For problems with multiple sources, a solution restricted to the Hanan grid may be far from the optimal solution, as shown in Fig. 1. Therefore, we cannot restrict our search for solutions to this grid.

In this paper, we study the problem of routing nets with multiple sources. This new model assumes that each node in a net may be a source, a sink, or both. The objective is to optimize the routing

0278–0070/97$10.00 © 1997 IEEE

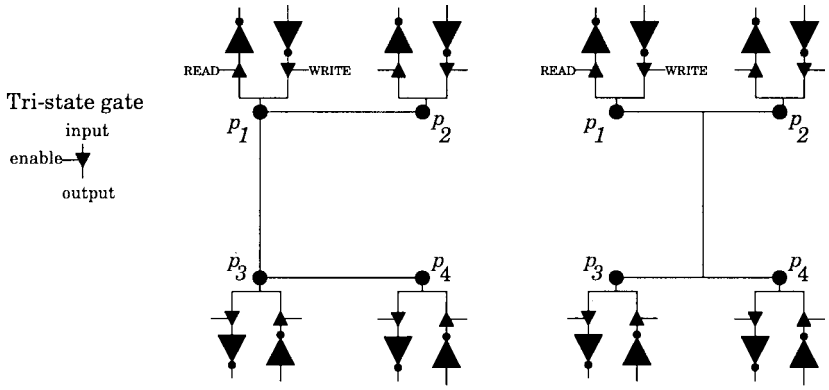


Fig. 1. An example of the impact of path length on delay. The first routing tree is optimized for node p_1. When p_2 drives the net, however, performance suffers. The second routing tree provides a lower maximum delay when both p_1 and p_2 can drive the net.

TABLE I
TOPOLOGY EFFECTS ON DELAY. FOR A NET WITH MULTIPLE SOURCES, THE DELAY TO A GIVEN SINK DEPENDS ON WHICH NODE DRIVES THE NET

	Routing Tree 1				Routing Tree 2			
Driver	p_1	p_2	p_3	p_4	p_1	p_2	p_3	p_4
p_1	-	1.99	2.93	3.18	-	3.19	4.11	4.11
p_2	4.34	-	5.21	5.45	3.19	-	4.11	4.11

topology to minimize the total weighted delay between *all* node pairs, where the weight between a node pair indicates the priority of delay minimization between this pair of nodes. We present an algorithm for the performance-driven multiple source routing tree problem based on construction of *minimum diameter A-trees*. Some preliminary results of our work were presented in ISCAS '95 [12].

II. PROBLEM FORMULATION

Given a set of points $P = \{p_1, p_2, \cdots, p_n\}$ on the Manhattan plane, and a nonnegative weight $W(p_i, p_j)$ as the *weight* between each pair of source p_i and sink p_j to indicate the timing criticality between this pair of points, the *performance-driven multiple source routing tree (PD-MSRT) problem* is defined as finding a Steiner tree T which connects all points in P and minimizes the following two objectives:

- total weighted delay $WD(T)$ between pairs of nodes p_i and p_j; i.e., $WD(T) = \sum_{p_i,p_j} W(p_i, p_j) \times delay(p_i, p_j)$;
- total tree length $L(T)$, defined as the sum of the lengths of each tree edge.

We assume that the first objective has higher priority than the second one. For simplicity, one may assume that $W(p_i, p_j) \in [0, 1]$, i.e., noncritical pairs of points have weight zero, and critical pairs have weight one. Values between 0 and 1 provide a greater degree of freedom in "tuning" for performance optimization, although the heuristic presented here can make only limited use of this. The delay between a pair of points, $delay(p_i, p_j)$, may be estimated using an appropriate model, such as the linear delay model (where delay is proportional to path length), the Elmore delay model [15], or calculated using SPICE.

Given a point $p_i \in P$, we use (x_i, y_i) to denote the x and y coordinates of point p_i. We will utilize an additional point q in some proofs, and denote its location with (x_q, y_q). For any two points p_i and p_j, we define the distance $d(p_i, p_j)$ between them as their Manhattan distance, $|x_i - x_j| + |y_i - y_j|$. Given a tree T, we define the distance between nodes p_i and p_j in T as $d_T(p_i, p_j)$, the sum of edge lengths along the unique path between the points. The *diameter* D of tree T over a set of points P, $D_T(P)$, is defined as the maximum $d_T(p_i, p_j)$ over all pairs p_i, p_j. Given a point set P, we define the diameter $D(P)$ of the set to be the maximum distance between any pair of points in the set. Clearly, we always have $D_T(P) \geq D(P)$.

Note that if the weights of all pairs are zero, the PD-MSRT problem as we have formulated it becomes the classical minimum Steiner tree problem, which is NP-hard. Therefore, the PD-MSRT problem is also NP-hard. However, if we do not minimize the total wire length, and only wish to minimize the total weighted Elmore delay, the complexity of the problem is not known.

When the delay bound of each pair of timing-critical nodes is given, one can also formulate the *constrained multiple source routing tree problem* as finding a Steiner routing tree which satisfies the delay constraint between every timing-critical pair and minimizes the total tree length.

In the following, we consider a simplified version of the general problem. We treat the sources and sinks of the routing problem as nodes in a graph, or points on a plane, and restrict path weighting to $\{0, 1\}$. Our approach to this problem is through the construction of minimum diameter trees with minimized total wire length.

The analysis in [11] indicated that total wirelength minimization under a shortest path constraint is an appropriate objective for single source routing problems in submicrometer design. We use that result as the motivation for our diameter-based objectives.

III. MINIMUM DIAMETER TREE CONSTRUCTION

For our algorithm, we minimize the maximum path length between any pair of critical source and sink, in order to minimize the maximum linear delay between any pair of critical source and sink.

In the case of a single driver, such minimization can be obtained by *radius minimization*, with direct paths between the driver and all sink nodes. Shortest path trees rooted at the source achieve this goal. A number of works address the radius objectives, both for general path length minimization, and also for skew minimization in clock nets [3], [6], [13], [20]. A minimum radius construction with a suitable root point may be also be a minimum diameter construction.

When there are multiple sources and sinks, path length minimization can be achieved by minimizing the maximum distance between any pair of nodes, which leads to *diameter minimization*. Our goal is to construct a minimum diameter routing tree with minimum total tree cost, as measured by a combination of maximum path length, average path length, and total tree length.

A number of results for minimum diameter trees on the Euclidean plane were presented in [17]. In particular, it was shown that the diameter of the smallest enclosing circle for a set of points also gives the minimum diameter for a tree connecting those points. After determination of this minimum diameter circle, a star topology

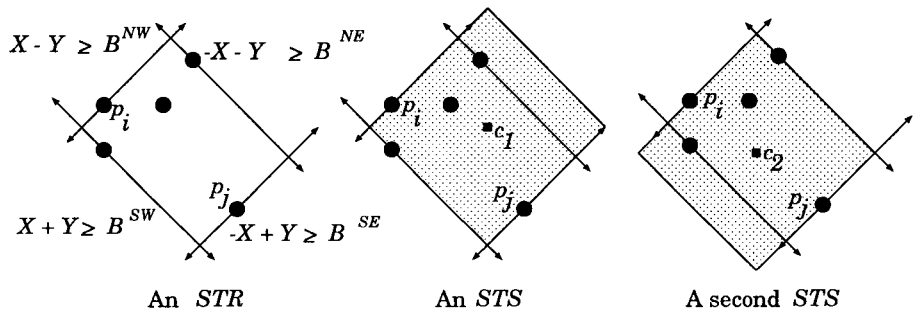


Fig. 2. The smallest tilted rectangle (STR) containing the points, and a pair of smallest tilted squares STS's which contain the rectangle.

connecting the center of the circle to each point in the set was shown to have the minimum diameter possible of any Steiner tree over the points. We follow their general approach, but address the Manhattan plane and also pursue tree length minimization.

The work in [3], [6], [13], and [20] can be used to construct minimum diameter trees, but they are concerned mainly with skew minimization instead of total tree length minimization. The Manhattan minimum diameter Steiner tree problem has not been explicitly studied in the literature. Our work studies the construction of minimum diameter Steiner trees in the Manhattan plane with minimized tree length.

In the next two subsections, we discuss general constraints related to Manhattan minimum diameter trees, present a pair of facts which give the lower bound for tree diameter, and then present a simple method, the *Minimum Diameter A-Tree (MD A-Tree)* algorithm, to construct trees that obtain this lower bound. A third subsection presents the *Minimum Cost Minimum Diameter A-Tree (MC MD A-Tree)* algorithm, which provides a method to optimize the tree construction.

A. Manhattan Tree Diameter Minimization

We define a *tilted rectangle (TR)* as a region defined by a rectangle with sides at 45° angles with respect to the X and Y axes. Such a region may be defined by a set of four equations, named boundary equations

$$-X + Y \geq B^{SE} \quad (1)$$
$$X - Y \geq B^{NW} \quad (2)$$
$$-X - Y \geq B^{NE} \quad (3)$$
$$X + Y \geq B^{SW}. \quad (4)$$

The constants B^{SE}, B^{NW}, B^{NE}, and B^{SW} represent the Southeastern, Northwestern, Northeastern, and Southwestern boundaries, respectively.

In the Manhattan plane, the analog of the Euclidean circle is the *tilted square (TS)*, the set of points p such that $d(p,c) \leq D/2$ for some center point c and diameter D. A *TS* is a special case of a *TR*, where the distances between opposite sides are equal. Obviously, the maximum distance between any pair of points contained in a *TS* of diameter D is less than or equal to D. The distance between points on opposite sides of a *TS* will be D.

Given a point set P, $STR(P)$, and $STS(P)$ are the *smallest tilted rectangle* and a *smallest tilted square* enclosing P, respectively. Clearly, $STS(P)$ contains $STR(P)$. Fig. 2 shows an STR and two STSs for a point set. Note that $STR(P)$ is unique, but there may be a set of STS's for a given point set.

Fact 1: For a point set P, the diameter of the point set $D(P)$ is equal to the diameter D of an STS containing the points.

Fact 2: A shortest path tree T rooted at the center c of an STS for a point set P is a minimum diameter tree.

Both of these facts can be derived from earlier works on zero-skew clock routing [3], [13], [14]. The center points defined by an STS (the STS may not be unique) can be equivalent to the final "merging segment" used in [20].

For the Manhattan plane, an STS can easily be found in linear time. We first find the STR enclosing the points by computing the B^{SE}, B^{NW}, B^{NE}, and B^{SW} values of the boundary equations in (1)–(4) as follows:

$$B^{SE} = \min(-x_i + y_i) \qquad \forall p_i \in P \quad (5)$$
$$B^{NW} = \min(x_i - y_i) \qquad \forall p_i \in P \quad (6)$$
$$B^{NE} = \min(-x_i - y_i) \qquad \forall p_i \in P \quad (7)$$
$$B^{SW} = \min(x_i + y_i) \qquad \forall p_i \in P. \quad (8)$$

We can then adjust one of the boundary equations to produce an STS covering the points.

Also note that the center of an STS can be obtained directly from the B^{SE}, B^{NW}, B^{NE}, and B^{SW} values. The two lines $-X + Y = (B^{SE} + B^{NW})/2$ and $X + Y = (B^{NE} + B^{SW})/2$ bisect the sides of the STR and will therefore intersect at the center of the STR. The center of the STR is also the center of an STS.

B. Minimum Diameter A-Tree Algorithm

Our algorithm presented in this paper for Manhattan minimum diameter tree construction consists of two basic steps (as shown in Fig. 3). The first step is to identify the root point r of the tree, which could be the center of an STS, or some other point obtained by methods we will describe in Section III-C4). The second step is to construct a shortest path tree rooted at r with low tree length. A rectilinear shortest path Steiner tree is also called a rectilinear *arborescence* [22], or an *A-Tree* in short. Since the *A-Tree algorithm* [11] has proven to be very effective in generating a shortest path Steiner tree on the Manhattan plane with near minimum total wirelength, it is used in the second step of our algorithm.

Alternatively, for large problems where run time is a consideration, a simpler heuristic by Rao *et al.* [22] (which was the basis for the A-Tree algorithm) may be used instead of the A-Tree algorithm.

The most basic variation of our algorithm is called the *Minimum Diameter A-Tree (MD A-Tree)* algorithm. It simply computes the STS for the point set P, and then constructs an A-tree rooted at the center of the STS. In an A-tree T rooted at point r, for any point p_i in P, $d_T(p_i, r) = d(p_i, r)$, as the routing within the tree is guaranteed to be a shortest path. As Fact 2 indicates, the resulting routing tree will have minimum diameter.

As was noted in Fig. 2, the STS of a set of points is not necessarily unique, and so there may be a number of acceptable "root" points for the center of a minimum diameter tree. The set of centers of STS's

1. Identify a root point r for the point set P.
2. Construct a shortest path tree connecting the root point r with each point in P.

Fig. 3. Basic steps of the *Minimum Diameter A-Tree (MDAT)* algorithm.

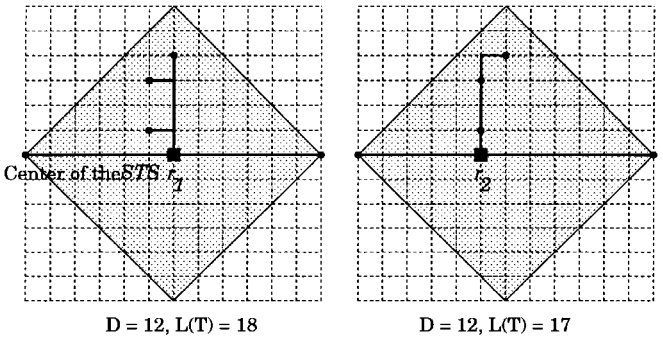


Fig. 4. Two shortest path trees in the Manhattan plane which satisfy $d(p_i, r) + d(r, p_j) \leq D$. The point r_1 is at the center of the STS; point r_2 is within the *feasible region* for the set of points. Both trees have a maximum diameter of 12, but the tree rooted at r_2, a point which is not the center of an STS, has lower tree length.

form a diagonal line and have been observed previously [20]. It is not always necessary to place the root of the A-tree at the center of an STS, however. This freedom leads to an optimization approach that is discussed in the next subsection.

C. Minimum Cost Minimum Diameter A-Tree Algorithm

By using an A-tree construction, we ensure that the tree distance between any point and the root is equal to the Manhattan distance. Having this, it is easy to see that as long as the root point r satisfies the constraint $d(p_i, r) + d(r, p_j) \leq D$ for all distinct p_i and p_j, the tree will have minimum diameter. This freedom allows for further optimization, with the possibility of reductions in tree length and weighted path length. An example of such an instance for the Manhattan plane is given in Fig. 4. By shifting the "center" point slightly, a reduction in tree length is obtained without an increase in the maximum diameter of the tree.

1) Feasible Region: As there can be more that one location that can serve as the root of a minimum diameter tree, we would like to compute this region precisely.

We define the *feasible region* (FR) of a set of points P as

$$FR(P) = \{r | d(p_i, r) + d(r, p_j) \leq D \qquad \forall p_i, p_j \in P\}.$$

If only a subset of point pairs are critical, we can define $P_c \subseteq P$ as the subset of points which are part of a nonzero weighting, i.e., $P_c = \{p_i \in P | W(p_i, p_j) \neq 0 \ \vee \ W(p_j, p_i) \neq 0$ for some $p_j\}$. D_c is defined to be the diameter of point set P_c. We then define the *critical feasible region* (FR_c) of P_c as

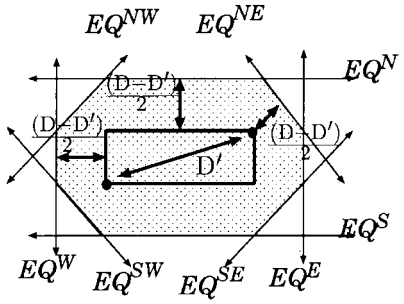
$$FR_c(P_c) = \{r | d(p_i, r) + d(r, p_j) \leq D_c \qquad \forall p_i, p_j \in P_c \text{ and } W(p_i, p_j) \neq 0\}.$$

The diameter for the FR_c is less than or equal to the diameter for the FR. Note that the FR_c does not place constraints on path lengths between noncritical pairs, so the path length between a noncritical pair may be greater than D_c. When all node pairs are critical, the FR and FR_c are equivalent; but, in general, the two regions are not equivalent and may even be disjoint.

In the Euclidean plane, the constraint $d(p_i, r) + d(p_j, r) \leq D$ defines an ellipse, with points p_i and p_j as focii. The feasible regions

TABLE II
OCTILINEAR BOUND EQUATIONS USED TO DEFINE EITHER AN OE OR AN OR

Equation	Constraint		
EQ^N:	$-Y$	$\geq$	B^N
EQ^S:	Y	$\geq$	B^S
EQ^E:	$-X$	$\geq$	B^E
EQ^W:	X	$\geq$	B^W
EQ^{NE}:	$-X + -Y$	$\geq$	B^{NE}
EQ^{NW}:	$X + -Y$	$\geq$	B^{NW}
EQ^{SE}:	$-X + Y$	$\geq$	B^{SE}
EQ^{SW}:	$X + Y$	$\geq$	B^{SW}



An octilinear ellipse, and its eight octilinear bound equations.

Fig. 5. An *octilinear ellipse (OE)* on the Manhattan plane, the set of points which satisfy $d(p_i, r) + d(r, p_j) \leq D$. This region can be defined as either a polygon bounded by no more than eight sides, or as the intersection of eight octilinear bound equations.

can be formed simply by intersecting a set of ellipses. A similar property holds for the Manhattan plane.

We define an *octilinear segment* to be a segment that is either horizontal, vertical, or has slope ± 1; an *octilinear bound equation* is an equation of the form $\alpha X + \beta Y \geq B$ $(\alpha, \beta \in \{0, 1, -1\}$, with at least one being nonzero), for some constant B. In the Manhattan plane, the set of points satisfying $d(p_i, r) + d(r, p_j) \leq D$ is an *octilinear ellipse (OE)*. An *OE* is bounded by no more than eight octilinear segments, and can also be represented by the intersection of eight *octilinear bound equations* with constants $B^N, B^S, B^E, B^W, B^{NE}, B^{NW}, B^{SE}$, and B^{SW} denoting the Northern, Southern, Eastern, Western, Northeastern, Northwestern, Southeastern, and Southwestern boundaries of the *OE*, respectively; the octilinear bound equations define half-plane regions, and are shown in Table II. If $d(p_i, p_j) = D'$, the *OE* contains the bounding box of the points with a "fringe" of $(D - D')/2$. An example is shown in Fig. 5.

An *octilinear region (OR)* is defined to be a convex region that is bounded by no more than eight octilinear segments; it can also be represented by the intersection of no more than eight octilinear bound equations. For the following, we will utilize this property to find the intersection of a number of OR's. An OE is an instance of an OR

If we wish to identify the intersection of a pair of OR's, we can do so by combining the bounds of each. Without loss of generality, let $E_1^f \equiv \alpha X + \beta Y \geq B_1^f$ be an octilinear bound equation of $OR\ R_1$, where $f \in \{N, S, E, W, NE, NW, SE, SW\}$, and $E_2^f \equiv \alpha X + \beta Y \geq B_2^f$ an octilinear bound equation of $OR\ R_2$. Then $\alpha X + \beta Y \geq \max(B_1^f, B_2^f)$ defines an octilinear bound equation of the $OR\ R_1 \cap R_2$. We will use the term "constrictive" in relation to these equations; if an equation E_1^f of $OR\ R_1$ defines a half-plane region that is a strict subset of E_2^f from $OR\ R_2$, then E_1^f is more constrictive than E_2^f.

In order to determine the set of points which may serve as the center of a minimum diameter tree, we find the intersection of all OE's for the point pairs. As the OE's are convex, their intersections (and the feasible regions) will also be convex. There are no more

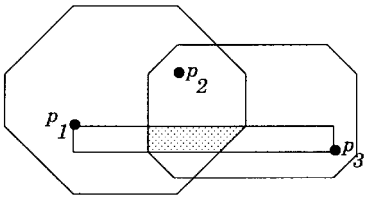


Fig. 6. The feasible region $FR(P)$ for the center of a minimum diameter A-tree, the intersection of a number of octilinear ellipses.

than $O(n^2)$ point pairs, resulting in the same number of OE's. It can be shown that the intersection of any number of OE's can be represented by a single OR. The shaded OR in Fig. 6 shows the feasible region for the root of a shortest path tree that will result in a minimum diameter tree.

Theorem 1: The $FR(P)$ and $FR_c(P)$ for any set of points P and any set of critical pairs are nonempty.

Proof: Fact 1 showed that the minimum diameter of a tree connecting the points was equal to the diameter of an STS; let c be the center of such an STS. Fact 2 showed that a shortest path tree rooted at c has minimum diameter. Clearly, c satisfies $d(p_i, c) + d(c, p_j) \leq D$ for any p_i and p_j, and is therefore contained by each OE. As this point is contained by each OE, it is also contained by the intersection, and therefore the $FR(P)$ and $FR_c(P)$ are nonempty. □

Theorem 2: Any shortest-path tree rooted at point $r \in FR(P)$ is a minimum diameter tree. Any shortest-path tree rooted at a point $r \in FR_c(P)$ is a minimum diameter tree over the critical points.

Proof: This arises directly from the definitions of $FR(P)$ and $FR_c(P)$. For any point r in the feasible region, $d(p_i, r) + d(r, p_j) \leq D$ for all pairs p_i and p_j; a shortest path tree T rooted at r ensures that $d_T(p_i, r) = d(p_i, r)$, so $d_T(p_i, r) + d_T(r, p_j) \leq D$. □

As $FR(P)$ and $FR_c(P)$ are nonempty, and points within these sets allow for the construction of minimum diameter trees, we will use them to guide our search for root points of low cost (in terms of path length or tree length) trees.

Clearly, construction of $FR(P)$ and $FR_c(P)$ can be performed in $O(n^2)$ time, by simply intersecting the $\binom{n}{2}$ OE's formed by all point pairs. In cases where n is large, it may be desirable to use a low complexity method to construct a shortest path tree (i.e., [22]). In the next subsection, we will present a method for linear time computation of these regions, preventing feasible region construction from dominating the run time.

Note that, in general, the feasible region defines an *area*. The final "merging segment" obtained by the planar zero-skew clock routing algorithm of Kahng and Tsao [20] may be a subset of the feasible region.

2) Linear Time Computation of Feasible Region: In this subsection we show how to compute $FR(P)$ in linear time; $FR_c(P)$ can be computed similarly by considering only critical points. We approach the problem by determining which pairs of points generate the most constrictive bounds for each of the eight octilinear bound equations that may define the feasible region. We will consider two cases: one for the uppermost horizontal bound of $FR(P)$, EQ^N, and one for the upper right diagonal bound EQ^{NE}. Other bounds may be obtained by similar methods.

Pseudocode for the algorithms to compute these bounds is given in Fig. 7. Proofs that these algorithms are correct are given in the next two Lemmas.

Lemma 1: For the two points $p_i, p_k \in P, x_i \leq x_k$, which form the most constrictive upper horizontal bound EQ^N *of* $FR(P)$, the point p_i will have minimal $x_i + y_i$ value, and the point p_k will have

```
find-EQ^N()
    find points p_i, p_j with smallest and second smallest
        x_i + y_i and x_j + y_j values respectively
    find points p_k, p_l with largest and second largest
        x_x - y_k and x_l - y_l values respectively
    if  (p_i ≠ p_k)
        Bound is obtained from OE(p_i, p_k)
    else
        Bound is obtained from most constrictive of
        OE(p_i, p_l) and OE(p_k, p_j).

find-EQ^NE()
    find points p_i, p_j with smallest and second smallest
        x_i + y_i and x_j + y_j values respectively
    Bound is obtained from OE(p_i, p_j)
```

Fig. 7. Pseudocode for algorithms which find the boundaries for the feasible regions in linear time. Other bounds may be obtained by simply replacing the smallest or largest value criteria with those specified in Tables III and IV.

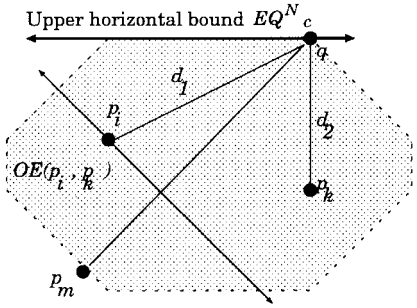


Fig. 8. Computation of the upper horizontal bound of the feasible region. If $OE(p_i, p_k)$ generates the most constrictive upper horizontal bound on $FR(P)$, then point p_i must have minimal $x_i + y_i$ value.

maximal $x_k - y_k$ value.

Proof: Let p_i and p_k be the two points which generate the most constrictive upper horizontal bount EQ^N, and $x_i \leq x_k$. Consider Fig. 8.

We choose a point q along the upper horizontal bound directly above point p_k, and have $d(p_i, q) = d_1, d(p_k, q) = d_2$, and $d_1 + d_2 = D$.

Now suppose that there exists a point p_m which satisfies $x_m + y_m < x_i + y_i$. These conditions result in the following:

$$d(p_m, q) > d(p_i, q) = d_1$$
$$d(p_m, q) + d_2 > D.$$

This implies that the upper horizontal bound for the OE of p_m and p_k is more constrictive than the one of p_i and p_k, and contradicts the initial assumption. Thus, the p_i element of the pair which generates the most constrictive bound is clearly one with smallest the $x_i + y_i$ value. Points with equal values will generate equivalent bounds. The p_i component of the pair can be found with a single pass over the points.

By a similar set of arguments, it is clear that the p_k element of the pair must have the largest $x_k - y_k$ value.

In some instances, the point with the smallest $X + Y$ and the point with the largest $X - Y$ may be the same; in this case, it is clear that the point will either be a "p_i" or a "p_k." We consider the second smallest $x_j + y_j$ and second largest $p_l \cdot x - p_l \cdot y$ points as well, with the most constrictive bound being formed from the point that the criteria select in common, and one of the secondary points. □

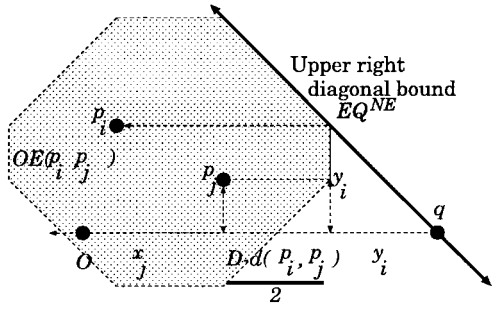


Fig. 9. Computation of the upper diagonal bound of the feasible region. The most constrictive bound on $FR(P)$ will be generated by the linear inequality of $OE(p_i, p_j)$ that intersects the X axis at the minimum q.

TABLE III
CRITERIA USED TO DETERMINE PAIRS OF POINTS WHICH DEFINE THE HORIZONTAL AND VERTICAL BOUNDS OF $FR(P)$. WHEN $p_i \neq p_k, OE(p_i, p_k)$ GENERATES A MOST CONSTRICTIVE BOUND ON THE FEASIBLE REGION. IF THE $p_i = p_k$, A MOST CONSTRICTIVE BOUND ON THE FEASIBLE REGION IS GENERATED BY EITHER $OE(p_i, p_l)$ OR $OE(p_j, p_k)$

Bound	p_i, p_j	p_k, p_l
EQ^N	smallest, second smallest $x+y$	largest, second largest $x-y$
EQ^S	smallest, second smallest $x-y$	largest, second largest $x+y$
EQ^E	smallest, second smallest $x+y$	smallest, second smallest $x-y$
EQ^W	largest, second largest $x-y$	largest, second largest $x+y$

TABLE IV
CRITERIA USED TO DETERMINE THE PAIRS OF POINTS WHICH DEFINE THE DIAGONAL BOUNDS OF $FR(P)$. FOR EACH DIAGONAL BOUND EQUATION, $OE(p_i, p_j)$ GENERATES THE MOST CONSTRICTIVE BOUND ON THE FEASIBLE REGION

Bound	p_i, p_j
EQ^{NE}	smallest and second smallest $x+y$
EQ^{NW}	largest and second largest $x-y$
EQ^{SE}	smallest and second smallest $x-y$
EQ^{SW}	largest and second largest $x+y$

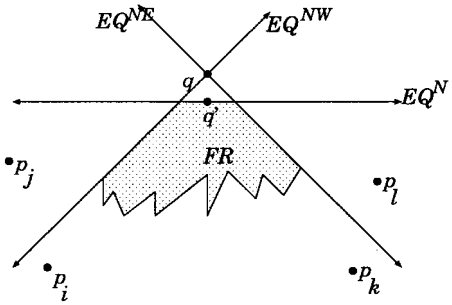


Fig. 10. The upper horizontal bound EQ^N of the feasible region must cross the diagonal bounds EQ^{NE} and EQ^{NW} at their intersection, or below their intersection.

Lemma 2: The two points $p_i, p_j \in P$ which form the most constrictive upper right diagonal bound EQ^{NE} *of* $FR(P)$ will have the two smallest $x_i + y_i$ and $x_j + y_j$ values.

Proof: Assume p_i and p_j are the point pair which forms the most constrictive upper right diagonal bound EQ^{NE} of $FR(P)$.

Consider Fig. 9. Assume that we shift the input point set so that it is contained in the first quadrant (this will not change the shape of the feasible region, or the points which provide the bounds). We label the intersection of EQ^{NE} and the X axis as point q; the bound EQ^{NE} which generates the minimal q will also generate the most constrictive bound on the FR.

Without loss of generality, we will assume that point p_i is to the left of p_j. For this proof, we will assume that $y_i \geq y_j$, the other case being solved similarly. We now have the following set of equations:

$$\begin{aligned} x_q &= y_i + \frac{D - d(p_i, p_j)}{2} + x_j \\ &= y_i + \frac{D}{2} - \frac{(x_j - x_i)}{2} - \frac{(y_i - y_j)}{2} + x_j \\ &= \frac{D}{2} + \frac{x_i + y_i}{2} + \frac{x_j + y_j}{2}. \end{aligned}$$

Since p_i and p_j generate the most constrictive upper right diagonal bound, q is minimum. Therefore, $x_i + y_i$ and $x_j + y_j$ are the smallest and second smallest among all $p \in P$. □

Theorem 3: $FR(P)$ and $FR_c(P)$ can be computed in $O(n)$ time, where n is the number of points in set P.

Proof: Fact 1 gave a linear time method to obtain diameter $D(P)$, while Lemmas 3 and 4 gave linear time methods to identify the pairs of points which generate the most constrictive bounds for EQ^N and EQ^{NE}. Other bounds can be obtained in a similar manner, using the criteria shown in Tables III and IV. □

3) Necessity of Feasible Region: The lemmas given above for the linear time construction of the feasible region are useful in proving another property of the feasible region: any minimum diameter tree *must* intersect $FR(P)$. To prove this, we will need an additional lemma.

Lemma 3: The upper horizontal bound EQ^N of $FR(P)$ will intersect $FR(P)$.

Proof: Consider Fig. 10. Let p_i and p_j be the two points with smallest and second smallest $X + Y$ values, respectively, Lemma 4 showed that these points generate the most constrictive upper right diagonal bound EQ^{NE}. Similarly, the points which define the upper right diagonal bound EQ^{NW} are labeled p_k and p_l. We identify the intersection of these two bounding lines as the point q, resulting in

$$\begin{aligned} d(p_i, q) + d(p_j, q) &= D \\ d(p_k, q) + d(p_l, q) &= D. \end{aligned}$$

Lemma 1 showed that the two points which generate the upper horizontal bound are p_i and p_k. As $d(p_i, q) \geq d(p_j, q)$, and $d(p_k, q) \geq d(p_l, q)$, the distances of p_i and p_k to point q are clearly at least $D/2$. A point q' along the upper horizontal bound EQ^N cannot have larger Y value than q, as this would result in $d(p_i, q') + d(p_k, q') > D$. □

Similarly the lower horizontal bound EQ^S, and the two vertical bounds EQ^E and EQ^W also intersect the feasible region (if only at a single point). The diagonal bounds cannot restrict the feasible region away from a horizontal or vertical bound (leaving a "gap" between feasible region and one of these bounds).

Theorem 4: Any minimum diameter tree connecting a set of points P must intersect $FR(P)$.

Proof: We will prove this by creating a contradiction: if a tree has minimum diameter over the points and does not intersect $FR(P)$, it contains a cycle.

Assume we have a minimum diameter tree, and that it does not intersect $FR(P)$. Consider Fig. 11, with the four most extreme points (smallest $X+Y$, largest $X-Y$, smallest $X-Y$, and largest $X+Y$) labeled p_i, p_j, p_k, and p_l respectively. Lemma 1 showed that these points are the ones which generate the horizontal and vertical bounds of $FR(P)$.

In a minimum diameter tree, there must be a path from p_i to p_j of length no greater than D. This path cannot go above the upper horizontal bound EQ^N, as this bound is generated by these two p_i and p_j. As Lemma 3 showed, EQ^N must intersect the feasible region, if only at a single point, so if the path is to avoid the feasible region, not only must it go beneath the EQ^N boundary, but beneath

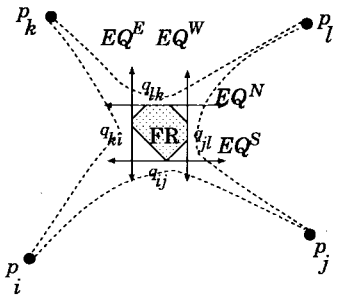


Fig. 11. Paths between the extreme points must form a cycle if they do not intersect the feasible region.

the entire feasible region, passing through some point q_{ij}. Similar constraints are placed on the p_i to p_k path, the p_k to p_l path, and the p_j to p_k path.

Thus, we have a set of paths $p_i \rightarrow q_{ij} \rightarrow p_j, p_j \rightarrow q_{jl} \rightarrow p_l, p_l \rightarrow q_{lk} \rightarrow p_k$, and $p_k \rightarrow q_{ki} \rightarrow p_i$. Clearly, there is a cycle $q_{ij} \rightarrow q_{jl} \rightarrow q_{lk} \rightarrow q_{ki} \rightarrow q_{ij}$, and the construction cannot be a tree. □

Thus, we show that not only will selection of a root point from the $FR(P)$ allow for the construction of a minimum diameter tree, but also that if a minimum diameter tree is required, the tree must pass through this region.

4) Summary of the MC MD A-Tree *Algorithm:* As was shown in Fig. 4, some locations of shortest path tree root points will lead to lower cost trees (in terms of path length or tree length) than other points. The MC MD A-Tree algorithm follows the basic outline given in Fig. 3. We first determine a set of candidate tree root locations, then select one root point from the set of candidates, and then finally construct an A-Tree rooted at that point.

A number of variations on tree root restriction and selection are possible for the algorithm, and we summarize them here.

First, there is a choice of which feasible region to use—either $FR(P)$ or $FR_c(P)$. The "appropriate" selection is dependent on both the locations of the critical sources and sinks, and their number. Table VIII in Section IV compares the two approaches.

From within the feasible region, we may have a number of points that are acceptable roots for a minimum diameter shortest path tree; in fact, if there is no underlying grid, there may be an infinite number of acceptable points. To make root selection tractable, we restrict *candidate root points* to be either Hanan grid points within the feasible region, points at the intersection of the Hanan grid lines with the boundaries of the feasible region, and corner points of the feasible region. It should be noted that the feasible region does not always contain points on the Hanan grid; this is the case for Fig. 1. In this figure, the feasible region consists of the single point at the center of the STS for the point set, and the only candidate root point is a "corner point" of the feasible region.

We can easily show that points which are not *candidate root points*, but are in the feasible region, do not need to be considered as possible roots (for linear delay and tree length objectives). It was shown in [22] that an optimal arboresence can be found on the Hanan grid lines, and we assume that all trees considered are so constrained. We define a *Hanan cell* to be a rectangular region bounded by a pair of vertically adjacent Hanan grid lines and a pair of horizontally adjacent Hanan grid lines.

Lemma 4: Given an arboresence with fixed topology and root point r: shifting the position of r horizontally (vertically) within a Hanan cell results in linear changes in tree length.

Proof: Without loss of generality, assume that the tree contains a vertical segment passing through the root at (x_r, y_r), and that a series of a horizontal segments branch off toward the left, and b branch toward the right. A shift in the root position by δx_r, while maintaining the same topology, results in a change in tree length of $(a - b) \times \delta x_r$. Thus, for a given topology, and within a Hanan cell, we have tree length as a linear function of the x_r. □

Within a Hanan cell, and with a fixed topology, we have tree cost as a linear function of the X and Y coordinates of the root. Given this, we have the following theorem.

Theorem 5: There exists a length optimal minimum diameter A-Tree, rooted at a candidate root point.

Proof: Assume we are given a length optimal minimum diameter A-Tree $T*$, with a root point (x_r, y_r) within a Hanan cell, but not at a candidate root point. By the previous lemma, we can shift the root location of this optimal tree, and its cost will change linearly.

This results in a simple linear programming problem, with the root location constrained by intersection of the Hanan cell C and the feasible region FR. Since both C and FR are convex, $C \cap FR$ is also convex. The minimum cost of $T*$ will be achieved at a corner point of $C \cap FR$, which is in the candidate root set. □

Similar properties hold if linear delay is our cost objective.

From the set of candidate root points, one must be selected to serve as the root of the A-Tree. We have performed experiments for root selection using two objective functions. One computes $\Sigma\ d(p_i, r)$ or $\Sigma\ W(p_i, p_j) \times (d(p_i, r) + d(p_j, r))$ as an estimation. The other performs actual A-Tree construction to obtain accurate distance measurements.

Experimentally, we have found the best solution performance to be that of the minimum diameter tree rooted within $FR_c(P)$, with the root minimizing the total tree length of an actual A-Tree construction. Detailed results are given in the next section.

The time complexity of our algorithm is comprised of two components. One is feasible region construction, which is $O(n)$. The other is the complexity of the shortest path tree construction. If we utilize the A-Tree algorithm, the complexity for constructing each A-Tree is $O(n^3)$, and the total complexity of the step is $O(kn^3)$, where k is the number of candidate root points [at worst $O(n^2)$].

If we use a faster A-Tree heuristic such as a variation of that by Rao *et al.* [22], the complexity of each A-tree construction is reduced to $O(n \log n)$. The total complexity of the step is $O(kn \log n)$. In this case, the overall complexity of MCMD A-Tree is $O(kn \log n)$, and the reduction of complexity for feasible region construction from $\Omega(n^2)$ to $O(n)$ is significant.

IV. Experimental Results

To evaluate the performance of the routing topologies, we used HSPICE to simulate a sized inverter driving a minimum-width wired network. The transistor model used for both 0.5-μm CMOS IC and MCM technologies was the 0.5-μm CMOS IC technology "nominal" model supplied by MCNC. Interconnect resistance and capacitance parameters are shown in Table V; these values are the same as those used in [11] and [4]. For each technology, we generated 100 test sets for each set size of 4, 8, and 16 randomly placed nodes.

For the 0.5-μm IC technology experiments, the surface area spanned was 1 cm^2, with a grid size of 10 μm (resulting in a 1000 by 1000 grid). Net segments were modeled as "π" circuits, with the capacitance of each wire segment divided between its endpoints. Segments longer than 1000 μm were broken into smaller segments (for example, a segment 1500 μm long is modeled as two π-model segments, one of 1000 μm, and the other of 500 μm). Experimentally, we found that shorter segment sizes did not improve accuracy, but did increase simulation time. Transistor sizes for the inverters were 20.0 μm $\times$ 0.5 μm and 19.0 μm $\times$ 0.5 μm for NMOS and PMOS, respectively; these sizes were selected to provide roughly equal rise

TABLE V
TECHNOLOGY PARAMETERS BASED ON ADVANCED MCM DESIGNS

Technology:	0.5μm CMOS IC	Multi-Chip Modules (MCMs)
Driver (NMOS, PMOS):	20.0μm $\times 0.5\mu$m, 19.0μm $\times 0.5\mu$m	200.0μm $\times 0.5\mu$m, 170.0μm $\times 0.5\mu$m
Unit Wire Resistance:	$0.0463\ \Omega/\mu m$	$0.02\ \Omega/\mu m$
Unit Wire Capacitance:	$0.189\ fF/\mu m$	$.085\ fF/\mu m$
Loading Capacitance:	$2.68\ fF$	$1000\ fF$
Total Area:	$1cm \times 1cm$	$10cm \times 10cm$
Wire Width:	$0.95\mu m$	$10\mu m$

TABLE VI
COMPARISON OF 1-STEINER, MD A-TREE, AND MC MD A-TREE FOR MAXIMUM DELAY (MD), AVERAGE MAXIMUM DELAY (AMD), AND AVERAGE DELAY (AD) THROUGH HSPICE SIMULATION WITH 0.5-μm CMOS IC TECHNOLOGY PARAMETERS. AVERAGE TREE LENGTHS AND DIAMETERS ARE IN CENTIMETERS

n	Topology	MD (ns)	AMD (ns)	AD (ns)	Length	Diameter
4	1-Steiner	.92	.87	.74	1.201	1.093
	MD A-Tree	.92	.87	.75	1.212	1.021
	MC MD A-Tree	.91	.86	.73	1.201	1.021
8	1-Steiner	1.72	1.51	1.17	1.822	1.395
	MD A-Tree	1.63	1.50	1.23	1.878	1.223
	MC MD A-Tree	1.62	1.48	1.20	1.826	1.223
16	1-Steiner	2.94	2.38	1.74	2.937	1.988
	MD A-Tree	2.58	2.25	1.83	3.170	1.507
	MC MD A-Tree	2.57	2.23	1.80	3.075	1.507

TABLE VII
COMPARISON OF 1-STEINER, MD A-TREE, AND MC MD A-TREE ALGORITHMS FOR MAXIMUM DELAY (MD), AVERAGE MAXIMUM DELAY (AMD), AND AVERAGE DELAY (AD) THROUGH HSPICE SIMULATION WITH MCM TECHNOLOGY PARAMETERS. AVERAGE TREE LENGTHS AND DIAMETERS ARE IN CENTIMETERS

n	Topology	MD (ns)	AMD (ns)	AD (ns)	Length	Diameter
4	1-Steiner	1.44	1.19	0.90	12.01	10.92
	MD A-Tree	1.39	1.19	0.90	12.12	10.21
	MC MD A-Tree	1.39	1.17	0.90	12.01	10.21
8	1-Steiner	3.44	2.46	1.62	18.22	13.95
	MD A-Tree	2.98	2.33	1.69	18.78	12.22
	MC MD A-Tree	3.00	2.28	1.62	18.26	12.22
16	1-Steiner	7.07	4.66	2.88	29.37	19.88
	MD A-Tree	5.56	4.00	2.87	31.70	15.07
	MC MD A-Tree	5.58	3.95	2.79	30.75	15.07

and fall times, and to provide delay values that were comparable to that of a 270-Ω resistor (as was used in [5] and [11]).

For the MCM experiments, the surface area spanned was 10 cm^2, with a grid size of 100 μm (resulting in a 1000 by 1000 grid). Segments longer than 10 000 μm were broken into smaller segments. Transistor sizes for the inverters were 200.0 μm $\times$ 0.5 μm and 170.0 μm μm $\times$ 0.5 μm for NMOS and PMOS, respectively; these sizes were selected to provide roughly equal rise and fall delays, and to provide delay values that were comparable to that of a 25-Ω resistor (as was used in [11], [5]).[1]

Delay for all cases was measured as the time between the input of the driver reaching 50% of the target value and the sink reaching 50% of its final value. Rise and fall times for the inputs to the drivers were 0.1 ns for all tests.

For each tree, we compute three types of delay. The first, *maximum delay* (MD) is the maximum delay of any source node to any sink node in a tree. The second, *average maximum delay* (AMD), is the average of the maximum source-sink delays for each source. The third, *average delay* (AD), is the average of all source-sink delays. Delay results are in nanoseconds, while tree and path lengths are in centimeters.

The topologies compared are as follows.

Minimum Diameter A-Trees: The STS for the weighted points is obtained, and an A-tree spanning all points is constructed.

Minimum Cost Minimum Diameter A-Trees: The $FR(P)$ or $FR_c(P)$ is constructed, and then possible root points from this region are evaluated by the construction of a tree at each point. The minimum length tree is selected as the final topology.

1-Steiner Trees: To obtain a tree with low total length, the 1-Steiner algorithm of Kahng and Robins [19] is used. These trees place no bounds on path length, but have very good performance for tree length minimization.

Note that existing performance driven interconnect optimization algorithms, listed in Section I, assume a fixed single driver, making comparisons with these methods inappropriate. As the optimal multisource routing solution does not lie entirely on the Hanan grid in general, the branch and bound method (as used in [4]) is difficult to apply. We are unaware of any existing multisource routing algorithm which can be used for a fair comparison. Also, there is no known method to compute an *optimal* multisource routing solution to evaluate the optimality of our algorithm.

Simulation results for 0.5-μm CMOS IC parameters are given in Table VI. While the 1-Steiner algorithm produces wire lengths that are from 1% to 8% lower, the average diameter of trees produced by the Minimum Diameter A-Tree and Minimum Cost Minimum Diameter A-Tree algorithms was from 6.6% to 24.2% lower. MCM technology examples, shown in Table VII, produced identical length results (the MCM test sets are scaled versions of the CMOS IC test sets). When given the freedom to select a root point, the Minimum Cost Minimum Diameter A-Tree algorithm obtained tree lengths that were from 1% to 2% lower than those of the Minimum Diameter A-Tree algorithm.

MD A-Tree and *MC MD A-Tree* produced similar delay results. Using 0.5-μm CMOS IC parameters, the two algorithms produced as much as an 12.6% maximum delay reduction, and a 6.3% average maximum delay reduction compared to the 1-Steiner algorithm. Using MCM parameters, the two algorithms produced as much as a 21% maximum delay reduction, and a 15.2% average maximum delay reduction.

For these experiments, it is assumed that the weight between all pairs is one, indicating that every path is critical.

Surprisingly, the *MD A-Tree* algorithm produced slightly better maximum delay values for the MCM examples than the *MC MD A-Tree* algorithm. In these cases, the "shifted" center location of the tree root results in some branches having disproportionally high capacitance, and the tree is "unbalanced." As has been observed in

[1] Note that we have sized the transistors to equalize rise and fall *delay*. This sizing results in unequal rise and fall times, i.e., the time for a sink to transition from 10% to 90% of the desired value. A more traditional 2 : 1 sizing ratio of P-transistor to N-transistor,which gives equal rise and fall times but unequal delay times, also showed similar improvement by our minimum-diameter A-tree algorithm. For example, under the 2 : 1 sizing using the same 0.5-μm CMOS IC parameters, the improvement in maximum delay of the MDAT topology over the 1-Steiner topology for eight randomly placed bidirectionaly nodes is 11.2% (as compared to 13.4% using the equal-delay sizing shown in Table VI).

TABLE VIII

COMPARISON OF 1-STEINER, MD A-TREE, AND MC MD A-TREE ALGORITHMS FOR MAXIMUM DELAY (MD), AVERAGE MAXIMUM DELAY (AMD), AND AVERAGE DELAY (AD) THROUGH HSPICE SIMULATION WITH 0.5-μm CMOS IC TECHNOLOGY PARAMETERS. ALL TEST SETS CONSISTED OF EIGHT NODES, WITH ONLY A GIVEN NUMBER OF NODE PAIRS BEING CRITICAL. ALSO INCLUDED IN THIS TABLE ARE THE AVERAGE WEIGHTED PATH LENGTHS (WPL), WEIGHTED DIAMETER (WD), AND REQUIRED PATH LENGTH (RPL). THE REQUIRED PATH LENGTH PROVIDES A LOWER BOUND FOR THE WEIGHTED PATH LENGTH, AND IN SOME CASES CANNOT BE OBTAINED. EXPERIMENTS CONSIDERING BOTH THE FEASIBLE REGION $[FR(P)]$ AND THE CRITICAL FEASIBLE $[FR_c(P)]$ WERE PERFORMED

pairs	Topology	MD *(ns)*	AMD *(ns)*	AD *(ns)*	Length	WPL	WD	RPL
1	1-Steiner	1.69	1.69	1.69	1.822	.777	.777	.645
	MD A-Tree, $FR(P)$	1.80	1.80	1.80	1.878	.803	.803	.645
	MD A-Tree, $FR_C(P)$	1.46	1.46	1.46	1.904	.645	.645	.645
	MC MD A-Tree, $FR(P)$	1.57	1.57	1.57	1.826	.719	.719	.645
	MC MD A-Tree, $FR_C(P)$	1.45	1.45	1.45	1.832	.645	.645	.645
2	1-Steiner	1.85	1.43	1.42	1.822	1.347	.849	1.201
	MD A-Tree, $FR(P)$	1.99	1.63	1.62	1.878	1.443	.881	1.201
	MD A-Tree, $FR_C(P)$	1.73	1.45	1.43	1.884	1.289	.763	1.201
	MC MD A-Tree, $FR(P)$	1.78	1.51	1.50	1.826	1.365	.808	1.201
	MC MD A-Tree, $FR_C(P)$	1.76	1.42	1.40	1.816	1.282	.767	1.201
3	1-Steiner	1.90	1.34	1.33	1.822	1.943	.895	1.751
	MD A-Tree, $FR(P)$	2.04	1.56	1.55	1.878	2.114	.925	1.751
	MD A-Tree, $FR_C(P)$	1.83	1.39	1.38	1.889	1.907	.823	1.751
	MC MD A-Tree, $FR(P)$	1.84	1.45	1.43	1.826	1.988	.854	1.751
	MC MD A-Tree, $FR_C(P)$	1.83	1.34	1.33	1.817	1.838	.813	1.751
10	1-Steiner	2.35	1.37	1.36	1.822	6.475	1.182	5.802
	MD A-Tree, $FR(P)$	2.31	1.50	1.49	1.878	6.777	1.054	5.802
	MD A-Tree, $FR_C(P)$	2.23	1.45	1.44	1.862	6.621	1.014	5.802
	MC MD A-Tree, $FR(P)$	2.19	1.41	1.40	1.826	6.514	1.029	5.802
	MC MD A-Tree, $FR_C(P)$	2.21	1.39	1.37	1.814	6.466	1.031	5.802

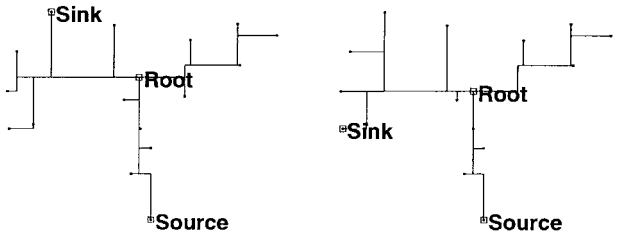


Fig. 12. An example in which a *MD A-Tree* topology has slightly lower maximum delay than a *MCMD A-Tree*, despite slightly higher tree length.

other high performance routing studies, the location of branching has an effect on the total delay. An example is shown in Fig. 12; the first topology is a simple *MD A-Tree* construction which, while having higher tree length, has lower maximum delay. While the source involved in the maximum delay path is the same for both topologies, the sink which has maximum delay changes.

In the Table VIII, we provide results for test sets of 8 nodes in the 0.5-μm CMOS IC technology, with 1, 2, 3, and 10 randomly selected pairs of critical nodes. For this table, we include the *weighted path length* (WPL) and *required path length* (RPL); the weighted path length is $\Sigma\ W(p_i, p_j) \times d_T(p_i, p_j)$ for all critical pairs, while the required path length is $\Sigma\ W(p_i, p_j) \times d(p_i, p_j)$. The required path length provides a lower bound (which in some instances cannot be achieved with a tree topology).

When only a subset of paths are critical, the *MC MD A-Tree* algorithm produced maximum delay improvements ranging from 6% to 14.2%. Trees rooted within the critical feasible region $FR_c(P)$ had slightly lower delay than those rooted in the feasible region $FR(P)$. Tree lengths were comparable.

Note that performance improvements for multisource routing problems are not as dramatic as was observed with single source problems [11]. An optimization which may be beneficial for source p_i may result in poor performance for a second source p_j; thus, the multisource problem has a greater number of constraints, resulting in less improvement in general. When only a small number of pin pairs are critical, the minimum diameter constraint may be overly restrictive.

Finally, we present an example of the application of these algorithms to a net extracted from an industry circuit. The Intel Corporation provided pin locations for six multisource nets from one of their processors; for five of these nets, the pin counts were very low. For nets with few pins, the topologies and delay characteristics of the minimum cost minimum diameter A-Tree topologies were nearly identical to the 1-Steiner topologies. The sixth and largest net contained 12 nodes, consisting of three inputs, five outputs, and five bidirectional nodes. Some nodes of this net were extremely close together, resulting in eight discrete groups. The optimized MC MD A-Tree topology had a significant impact on this net. We evaluated the topologies using HSPICE and the 0.5-μm CMOS IC technology parameters. The 1-Steiner and MC MD A-Tree topologies for this net, as well as delay and lengths, are shown in Fig. 13, For the single largest net, MC MD A-Tree provided a 16.1% reduction in the maximum delay, while the average reduction over the six nets was 2.7%. For all nets, the MC MD A-Tree topology had a maximum delay that was less than or equal to the maximum delay of the 1-Steiner topology.

In all cases, the time required for topology construction was much smaller than the time required to perform the HSPICE simulations, and the bulk of topology construction time was consumed by the *A-Tree* algorithm. The most complex variation, *MC MD A-Tree*, had run times ranging from 0.6 to 6 s for examples with 16 nodes and all pairs critical. The run time of the algorithm was strongly influenced by the size of the feasible region, as this impacts the number of candidate root locations; a larger feasible region (which can occur when fewer points are critical) resulted in a larger run time.

V. CONCLUSIONS AND FUTURE WORK

We have formulated a new performance-driven routing problem which considers interconnect topology optimization when there are

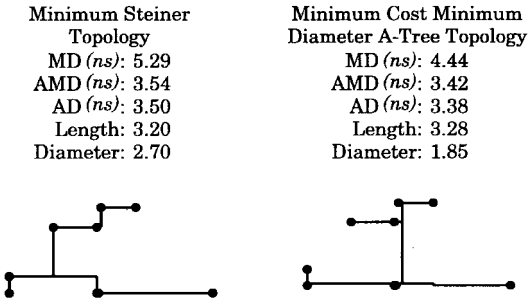


Fig. 13. Two possible topologies for a multisource net from an Intel processor. The first topology is a minimum length Steiner tree, while the second is a minimum cost minimum diameter A-Tree. The shorter maximum path length of the MC MD A-Tree results in a lower maximum signal delay between a source and sink. All lengths and diameters are in centimeters.

multiple sources in a single net, and have presented a heuristic solution for the problem through the construction of *Minimum Cost Minimum Diameter A-Trees*.

We have also given a linear time algorithm to determine the set of possible root locations for a minimum diameter shortest path tree, and shown that any minimum diameter tree must pass through this region. For most problems, the run time of the A-Tree algorithm dominates total run time; if we adopt a lower complexity tree construction approach, linear time construction of the feasible region becomes beneficial.

When compared with the 1-Steiner algorithm, minimum diameter A-trees produced lower maximum and average delays, with slightly higher tree length. Consideration of the feasible region resulted in reductions in tree length, and also in delay for most cases.

We are currently investigating a number of related performance driven multiple source routing problems. We are interested in the use of path length bounds for critical pairs while minimizing total wire length, possibly through the extension of previous bounded radius bounded cost algorithms such as BRBC [9] and AHHK [1]. Driver and wire sizing has proven effective for single source routing problems, and we are extending these to the multiple source domain. For practical VLSI routing applications, we are also considering routing in the presence of obstacles and nontree topologies with low total tree length. All of these problems will need extension to support more accurate delay models, so that optimization can be done for delay constraints rather than geometric constraints.

Acknowledgment

The authors are grateful to C.-K. Koh and the anonymous reviewers for their helpful suggestions on an earlier draft. They wish to thank H. Chan of Intel for supplying benchmark data.

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," in *Proc. IEEE Int. Symp. Circuits Syst.*, 1993, pp. 1869–1872.

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

[3] K. D. Boese and A. B. Kahng, "Zero-skew clock routing trees with minimum wirelength," in *Proc. IEEE Int. Conf. ASIC*, 1992, pp. 1.1.1–1.1.5.

[4] K. D. Boese, A. B. Kahng, B. A. McCoy, and G. Robins, "Rectilinear Steiner trees with minimum Elmore delay," in *Proc. 31st ACM/IEEE Design Automation Conf.*, 1994, pp. 381–386.

[5] K. D. Boese, A. B. Kahng, and G. Robins, "High-performance routing trees With identified critical sinks," in *Proc. ACM/IEEE Design Automation Conf.*, 1993, pp. 182–187.

[6] T.-H. Chao, Y.-C. Hsu, J.-M. Ho, K. D. Boese, and A. B. Kahng, "Zero skew clock routing with minimum wirelength," *IEEE Trans. Circuits Syst.*, vol. 39, pp. 799–814, Nov. 1992.

[7] J. P. Cohoon and L. J. Randall, "Critical net routing," in *Proc. IEEE Int. Conf. Computer Design*, pp. 174–177, 1991.

[8] K. D. Boese, J. Cong, A. B. Kahng, K. S. Leung and D. Zhou, "On high-speed VLSI interconnects: Analysis and design," in *Proc. Asia–Pacific Conf. Circuits Syst.*, Dec. 1992, pp. 35–40.

[9] J. Cong, A. B. Kahng, G. Robins, and M. Sarrafzadeh, "Provably good performance-driven global routing," *IEEE Trans. Computer-Aided Design*, vol. 11, no. 6, pp. 739–752, June 1992.

[10] J. Cong and K.-S. Leung, "Optimal wiresizing under the distributed Elmore delay model," in *Proc. Int. Conf. Computer-Aided Design*, 1993, pp. 110–114.

[11] J. Cong, K.-S. Leung, and D. Zhou, "Performance-driven interconnect design based on distributed *RC* delay model," in *Proc. 30th ACM/IEEE Design Automation Conf.*, 1993, pp. 606–611.

[12] J. Cong and P. H. Madden, "Performance driven routing with multiple sources," in *Proc. Int. Symp. Circuits Syst.*, vol. 1, 1995, pp. 203–206.

[13] M. Edahiro, "Minimum skew and minimum path length routing in VLSI layout design," *NEC Res. Develop.*, vol. 32, no. 4, pp. 569–575, Oct. 1991.

[14] ——, "Minimum path-length equi-distant routing", in *Proc. IEEE Asia–Pacific Conf. Circuits Syst.*, 1992, pp. 41–46.

[15] W. C. Elmore, "The transient response of damped linear network with particular regard to wideband amplifier," *J. Appl. Phys.*, no. 19, pp. 55–63, 1948.

[16] M. Hanan, "On Steiner's problem with rectilinear distance," *SIAM J. Appl. Math.*, vol. 14, no. 2, pp. 255–265, Feb. 1966.

[17] J.-M. Ho, D. T. Lee, C.-H. Chang, and C. K. Wong "Bounded-diameter minimum spanning trees and related problems," in *Proc. Comptat. Geometry Conf.*, 1989, pp. 276–282.

[18] X. Hong, T. Xue, E. S. Kuh, C. K. Cheng, and J. Huang, "Performance-driven Steiner tree algorithms for global routing," in *Proc. ACM/IEEE Design Automation Conf.*, 1993, pp. 177–181.

[19] A. B. Kahng and G. Robins, "A new family of Steiner tree heuristics with good performance: The iterated 1-Steiner approach," in *Proc. IEEE Int. Conf Computer-Aided Design*, 1990, pp. 428–431.

[20] A. B. Kahng and C.-W. A. Tsao, "Planar-DME: Improved planar zero-skew clock routing with minimum pathlength delay," in *Proc. European Design Automation Conf.*, 1994, pp. 440–445.

[21] S. Khuller, B. Raghavachari, and N. Young, "Balancing minimum spanning trees and shortest-path trees," in *Proc. ACM/SIAM Symp. Discrete Algorithms*, Jan. 1993, pp. 243–250.

[22] S. K. Rao, P. Sadayappan, F. K. Hwang, and P. W. Shor, "The rectilinear Steiner arborescence problem," *Algorithmica*, vol. 7, pp. 277–288, 1992.

[23] D. Zhou, S. Su, F. Fsui, D. S Gao, and J. Cong, "A simplified synthesis of transmission lines with a tree structure," *J. Analog Integrated Circuits Signal Processing*, (Special Issue on High-Speed Interconnects), vol. 5, no. 1, pp. 19–30, Jan. 1994.