Minimum Density Interconnection Trees

C. J. Alpert,J. Cong,A. B. Kahng,G. Robins,M. Sarrafzadeh

Technical Report No. CS-92-35
October 1, 1992

Minimum Density Interconnection Trees*

C. J. Alpert, J. Cong, A. B. Kahng, G. Robins†, and M. Sarrafzadeh‡

CS Dept., University of California at Los Angeles, Los Angeles, CA 90024-1596
† CS Dept., University of Virginia, Charlottesville, VA 22903-2442
‡ EECS Dept., Northwestern University, IL 60208-3118

Abstract

We discuss a new *minimum density* objective for spanning and Steiner tree constructions in the plane. This formulation is motivated particularly by the need for balanced usage of routing resources to achieve minimum-area VLSI layouts. We present two efficient heuristics for constructing low-density spanning trees, and prove that their outputs are within small constants of optimal with respect to both tree weight and density. Our proof techniques suggest a non-uniform lower bound schema, which may be used to establish the quality of the heuristic solution for any given instance. More interesting is the fact that the minimum density objective can be *transparently combined* with a number of previous interconnection objectives, without affecting the solution quality with respect to these previous metrics. As examples, we show how sets of competing measures (e.g., cost, density, radius; or cost, density, skew) can be *simultaneously* optimized. Extensive simulation results suggest that applications to VLSI global routing are promising.

1 Introduction

In this paper, we address a new problem formulation for spanning and Steiner tree constructions in the Manhattan plane. Our objective is to minimize the *density* of such interconnection trees, where density reflects a concept that is closely related to the stabbing number studied in computational geometry [8]. The motivation for our work lies in the area minimization requirement that is implicit in the global routing phase of VLSI layout; global routing entails building (spanning or Steiner) interconnection trees over given *signal nets* which correspond to point sets in the Manhattan plane. Traditionally, minimization of total edgelength in the tree is used to approximately capture the goal of minimizing the total area of the chip, since wires have a fixed width and must be routed at a fixed separation from each other. However, the structure of integrated circuit routing resources allows us to more precisely

*Partial support for this work was provided by a Department of Defense Graduate Fellowship; by NSF MIP-9110511; by NSF MIP-9110696, NSF Young Investigator Award MIP-9257982, ARO DAAK-70-92-K-0001, and ARO DAAL-03-92-G-0050; and by an IBM Graduate Fellowship.

determine the impact of a given interconnection topology on the chip area. For a detailed treatment of integrated circuit routing methodologies, the reader is referred to [14].

1.1 Problem Formulation

Consider the example of Figure 1, which depicts four *terminals* of a signal net. The interconnection tree in Figure 1(a) forces at least three wires to cross the dashed line, meaning that the vertical dimension of the chip must be large enough accomodate at least this many so-called *routing tracks* [14]. In contrast, the routing of Figure 1(b) only forces the vertical chip dimension to grow by one routing track (although the horizontal dimension grows by one track, as shown by the horizontal dashed line). In general, the most effective use of chip area will be attained if the chip dimensions are roughly equal; this suggests a *balancing* of the horizontal and vertical routing requirements induced by the interconnection tree. With the example of Figure 1 in mind, we develop the *Minimum Density Interconnection Tree* problem as follows.

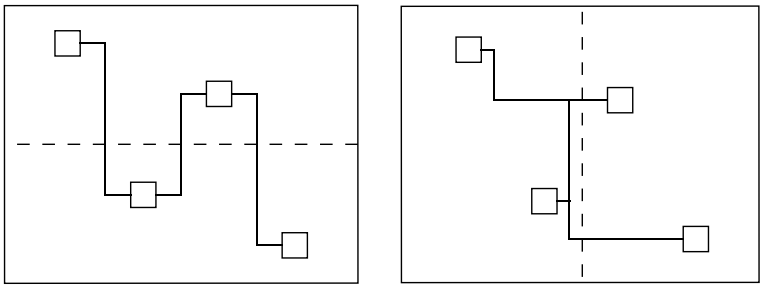

Figure 1: A four-point example for which the tree on the left has density 3, but the tree on the right has density 2.

We say that a *signal net* N is a set of n points, or *terminals*, $p_1, p_2, \ldots, p_n \in N$ in the Manhattan plane. If necessary, we may distinguish p_1 as a *source* terminal, with the remaining members of N being *sink* terminals. An *interconnection tree* of a net N, denoted $T(N)$, is a tree which spans N. The *cost* of a routing tree T is the sum of the costs of its edges, where the cost of an edge is the Manhattan distance between its endpoints. Without loss of generality, we assume that the terminal coordinates are scaled so that the entire signal net lies within the unit square.

Definition: The *density* of an interconnection tree is the maximum number of tree edges properly

intersected by any horizontal or vertical line in the plane (Figure 2).[1]

Definition: For a given net N, the *minimum density* of N is the minimum density achievable by an interconnection tree $T(N)$, and a *minimum density interconnection tree* is any $T(N)$ that achieves this density.

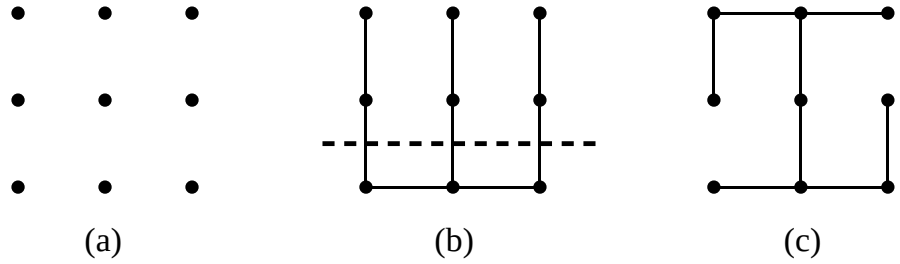

Figure 2: (a) Example of a signal net, along with (b) an interconnection tree with density 3, and (c) a minimum density tree with density 2.

In our work, we will address the following:

Minimum Density Interconnection Tree (MDIT) Problem: Given a net N, find a minimum density interconnection tree $T(N)$ that has minimum cost.

1.2 Related Formulations

It is interesting to note that a number of objectives for interconnection trees have been examined in the VLSI CAD routing literature. Three in particular are in some sense motivated by the issue of system *performance*, i.e., improving the maximum speed at which a digital system may be clocked. To be specific, these objectives include (i) minimizing the total cost of an interconnection tree (this reflects the RC delay of the wiring in addition to chip area), (ii) minimizing the maximum pathlength in the tree from the identified source to any sink terminal, i.e., the tree *radius* (this reflects the maximum signal delay, particularly for newer interconnect technologies such as those for multi-chip module design [9]), and (iii) minimizing the maximum *difference*, or *skew*, between source-sink pathlengths within a given interconnection tree (this reflects the clock skew minimization problem, which also arises due to system performance considerations).

Each of these issues has engendered an extensive literature, and all three are currently under active

[1] A line *properly* intersects an edge if and only if it intersects the edge at exactly one point (i.e., a line whose intersection with an edge is a (non-degenerate) segment, does not constitute a proper intersection).

investigation throughout the research community. The first corresponds to the well-known minimum rectilinear Steiner tree problem; recent surveys may be found in [13] [15] [19]. The second issue has been treated in the "bounded-radius, bounded-cost" interconnection tree algorithms of [4] [5] [6]; see also the discussion of [2], which surveys previous work on the "timing-driven interconnection" problem. Finally, the minimum clock skew problem has been extensively treated in such recent works as [11] [12] [18] [3]. We make note of these existing formulations because our proposed algorithms for minimum-density interconnection trees afford unique multiple optimizations – indeed, "triple optimizations" – wherein more than one competing objective may be optimized *simultaneously*. Section 4 below describes how, for example, tree cost, radius, and density can be simultaneously optimized; we also show how tree cost, skew and density can also be simultaneously addressed.

The remainder of this paper is organized as follows. In Section 2, we give two efficient heuristic constructions, along with several simple variants. Section 3 then establishes a number of performance bounds: we show that these methods have good performance in the sense that they on average produce interconnection trees with both cost and density bounded by constants times optimal. Section 4 integrates the minimum density objective with current performance-driven objectives via our "triple optimizations". Finally, Section 5 gives experimental results and notes several directions for future research.

2 Heuristics for Minimum Density Interconnection Trees

In the following, we assume that the x and y coordinates of the net terminals are all distinct, so that all intersections between routing tree edges and horizontal (or vertical) lines will be proper.[2] We also assume that the net contains exactly $n = k^2$ terminals, for some positive integer k.

2.1 Spanning and Steiner Trees Via the COMB Construction

Our first basic algorithm partitions the terminals of net N into $\sqrt{n}$ vertical strips, each containing $\sqrt{n}$ terminals (Figure 3a). We then connect all the terminals in each strip in a chain in order of decreasing y coordinate (Figure 3b). At this point we have $\sqrt{n}$ chains; we form a routing tree by joining the bottom terminals of all the chains (i.e., those with lowest y coordinates in each strip) from left to

[2] If the input contains non-distinct x and y coordinates, some coordinates can be perturbed slightly to achieve uniqueness.

right (Figure 3c). This process is described in Figure 4.[3] The complexity of this algorithm, which we call COMB, is clearly dominated by the sorting/partitioning step (line number 1 in Figure 4), and is therefore $O(n \log n)$.

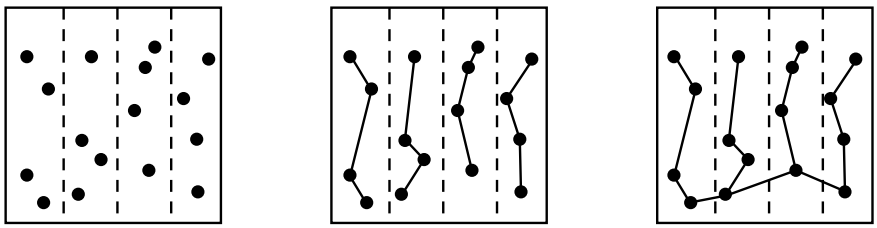

Figure 3: Execution of the COMB algorithm on a net of size $n = 16$.

Algorithm: COMB
Input: Net N containing $n = k^2$ terminals **Output:** a heuristic minimum-density interconnection tree $T(N)$
1: **Partition** N into $\sqrt{n}$ vertical strips each containing $\sqrt{n}$ terminals 2: **Connect** as a chain the terminals within each strip, sorted by their y-coordinates 3: **Connect** as a chain the bottom terminals of all strips, sorted by their x-coordinates 4: **Output** resulting spanning tree

Figure 4: Algorithm COMB: heuristic minimum density tree construction.

If we are allowed to introduce Steiner points in constructing the interconnection tree, we can reduce the worst-case density as well as the worst-case cost of our construction via the following method: (1) partition the net N into $\frac{\sqrt{n}}{\sqrt{2}}$ vertical strips, each containing $\sqrt{2} \cdot \sqrt{n}$ terminals (Figure 5a); (ii) connect all the terminals in each strip to a central *spine*[4] within the strip (Figure 5b); then (iii) join these spines using a single horizontal edge (Figure 5c). This variant, which we call COMB_ST, is described in Figure 6 and also has complexity $O(n \log n)$, again reflecting the complexity of the sorting/partitioning step at Line 1.

2.2 A Chain Peeling Method

A different approach to density minimization entails iterative superposition of chains over the net, where a *chain* is defined to be a set of terminals through which a staircase routing exists (i.e., a

[3] It is interesting to note that a similar strip-partitioning technique has been used in [17] for the performance analysis of traveling salesman and minimum matching heuristics.

[4] Formally, the *spine* is the vertical line which passes through the median terminal of the strip when the terminals are considered sorted by x-coordinate.

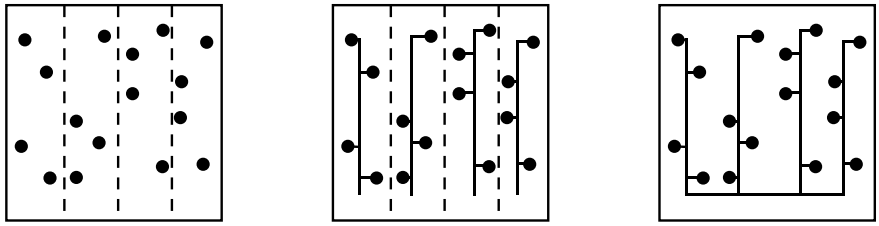

Figure 5: Execution of the COMB_ST algorithm on a net of size $n = 16$.

Algorithm: COMB_ST
Input: Net N containing $n = k^2$ terminals **Output:** a heuristic minimum-density Steiner interconnection tree $T(N)$
1: **Partition** N into $\frac{\sqrt{n}}{\sqrt{2}}$ vertical strips each containing $\sqrt{2} \cdot \sqrt{n}$ terminals 2: **Connect** the terminals within each strip to a middle spine 3: **Connect** the bottoms of all spines, sorted by their x-coordinates 4: **Output** resulting Steiner tree

Figure 6: Algorithm COMB_ST: heuristic minimum density (Steiner) tree construction.

sequence of terminals whose coordinates are monotone in both x and y). Each chain contributes at most 1 to the overall density, and once a maximal chain is detected, it is "removed" from the net and the process is iterated over the remaining terminals, until the net is covered. According to Dilworth's theorem from lattice theory [7], every partially ordered set of size n must contain a chain (or an anti-chain) of size at least $\sqrt{n}$, and such maximal chains can be computed efficiently in $O(n \log n)$ time. We call this algorithm PEEL (Figure 7). As we shall see in Section 3.1, the time complexity of this method is $O(n^{\frac{3}{2}} \log \log n)$.

3 Performance Bounds

We can show that both the density and the total tree cost of our constructions are on average only small constant factors away from optimal.

3.1 Density Bounds

A lower bound of $\Omega(\sqrt{n})$ can easily be established for the density of any spanning tree $T(N)$:

Algorithm: PEEL
Input: a net N, containing $\|N\| = n = k^2$ terminals **Output:** a low-density low-cost tree spanning N
1: $S = N$ 2: $T = \emptyset$ 3: **While** $S \neq \emptyset$ **Do** 4: $C =$ maximum chain (or antichain) of S 5: $T = T \cup C$ 6: $S = S - C$ 7: Join all the chains of T and output the resulting tree

Figure 7: Algorithm PEEL produces a low-density tree by iteratively finding long chains or antichains.

Lemma 3.1 *A net N consisting of n terminals regularly arranged in a $\sqrt{n} \times \sqrt{n}$ lattice cannot be connected by an interconnection tree $T(N)$ of density less than $\frac{\sqrt{n}}{2} - 1$.*

Proof: Consider the set of all horizontal and vertical lines between the rows and columns of terminals, as shown in Figure 8. In any spanning tree $T(N)$, each terminal must have an incident edge of the tree (to ensure connectedness), and this edge will cross some horizontal or vertical line. Since there are $2(\sqrt{n} - 1)$ horizontal and vertical lines, and there are $n - 1$ crossings of these lines, by the pigeonhole principle at least one of the lines is crossed $\frac{n-1}{2(\sqrt{n}-1)} \geq \frac{\sqrt{n}}{2} - 1$ times. □

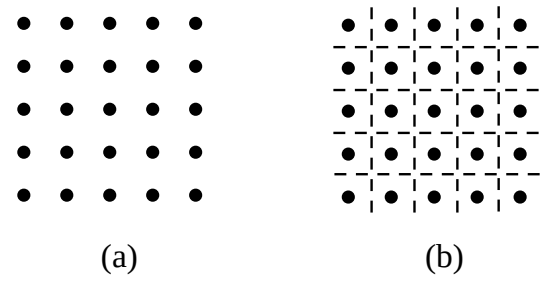

Figure 8: A lower bound for worst-case value of the minimum density.

Theorem 3.2 *For a net of n terminals whose locations are chosen randomly from the uniform distribution in the unit square, the minimum density interconnection tree has expected density $\Theta(\sqrt{n})$.*

Proof: Partition the unit square into n identical square cells, each of size $\frac{1}{\sqrt{n}}$ by $\frac{1}{\sqrt{n}}$ (Figure 9a). Each non-empty cell of the partition contains at least one terminal N_i which has an adjacent minimum

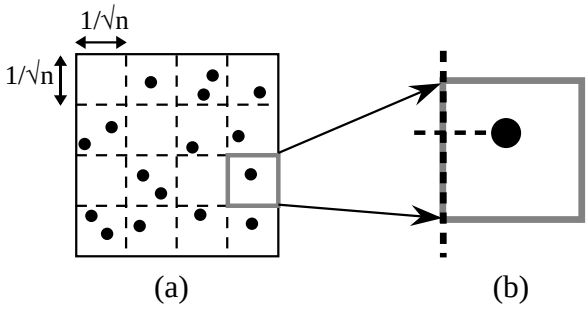

Figure 9: Expected minimum routing density of a net: (a) the unit square is partitioned into n congruent cells; (b) each non-empty cell contributes at least one edge that crosses a cell boundary.

density tree edge E_j that crosses one of the four sides of that cell (otherwise the terminals in this cell will not be spanned by the minimum density tree). Thus, the edge E_j increases the density count of the line containing the cell side crossed by E_j (Figure 9b). Note that the fraction of cells that are non-empty is $1 - \frac{1}{e}$ on average,[5] and that each of these will force an edge in $T(N)$ to cross at least one of the $2\sqrt{n} - 2$ vertical and horizontal lines which define our partition of the unit square into cells (the four outermost lines which define the boundary of the unit square are not crossed by any edges of the interconnection tree). By the pigeonhole principle, at least one of these lines will intersect at least $(1 - \frac{1}{e}) \cdot n/(2\sqrt{n} - 2) > (1 - \frac{1}{e}) \cdot \frac{\sqrt{n}}{2}$ of the routing tree edges, implying a lower bound of $(1-\frac{1}{e}) \cdot \frac{\sqrt{n}}{2} = \Omega(\sqrt{n})$ for the expected density of the minimum density tree. Since our algorithms always yield interconnection trees with density $O(\sqrt{n})$ (see the following sequence of results), the expected minimum routing density for a net of n terminals uniformly distributed in the unit square is $\Theta(\sqrt{n})$.

□

The density bounds for our heuristics are established as follows.

Theorem 3.3 *Algorithm COMB constructs a spanning interconnection tree with density* $\leq 2 \cdot \sqrt{n}$.

[5] Consider the expected fraction of non-empty boxes (i.e., cells) after n indistinguishable balls (i.e., terminals) have been placed independently and uniformly into n indistinguishable boxes. The probability of a given ball hitting a given box B_i is $\frac{1}{n}$, and so the probability of the ball missing box B_i is $1 - \frac{1}{n}$. By the independence of the trials the the probability of all n balls missing box B_i is $(1 - \frac{1}{n})^n$. Therefore, as n grows, symmetry implies that the probability of any box remaining empty is $\lim_{n \to \infty} (1 - \frac{1}{n})^n = \frac{1}{e}$. It follows by independence and linearity of expectation that for large n, a constant fraction $\frac{1}{e} \approx 0.368$ of the boxes is expected to remain empty, and hence $(1 - \frac{1}{e}) \cdot n$ boxes on average will be non-empty.

Proof: Since each strip contains no more than $\sqrt{n}$ terminals, a vertical line passing through any strip cannot intersect more than $\sqrt{n}$ tree edges. Any given horizontal line cannot intersect more than two edges within each strip. From this latter fact, we see that the density of the COMB output is at most $\sqrt{n}$. □

Theorem 3.4 *Algorithm COMB_ST constructs a Steiner interconnection tree with density* $\leq \frac{\sqrt{n}}{\sqrt{2}} + 1$.

Proof: Since each strip in the construction of Figure 5 contains no more than $\frac{\sqrt{n}}{\sqrt{2}}$ terminals on each side of its spine, no vertical line passing through any strip will intersect more than $\frac{\sqrt{n}}{\sqrt{2}}$ tree edges. Similarly, no horizontal line will intersect any of the $\frac{\sqrt{n}}{\sqrt{2}}$ vertical spines more than once. Thus, the density of $T(N)$ is at most $\frac{\sqrt{n}}{\sqrt{2}} + 1$ when we consider the added horizontal line which joins the spines together. □

A density bound for the chain-peeling algorithm PEEL follows from the following two results, namely, (i) that at most $O(sqrtn)$ chains or antichains will be "peeled" during the construction, and (ii) that these chains/antichains can be connected to form a single component which has density of at most the number of chains/antichains.

Lemma 3.5 *Algorithm PEEL computes at most* $2 \cdot \sqrt{n}$ *chains and / or antichains before terminating.*

Proof: Let a_i denote the number of points left in the point after we have peeled of the i-th chain/antichain. Assume that the algorithm stops when we have peeled off k chains and/or antichains, i.e., $a_k = 0$. We want to show that $k \leq 2 \cdot \sqrt{n}$. According to Dilworth's Theorem [7], the size of the $(i+1)$-th chain/antichain is at least $\sqrt{a_i}$. Thus, $a_{i+1} \leq a_i - \sqrt{a_i}$. Moreover, it is easy to verify that $\sqrt{x - \sqrt{x}} \leq \sqrt{x} - \frac{1}{2}$. Therefore,

$$\sqrt{a_k} \leq \sqrt{a_{k-1} - \sqrt{a_{k-1}}} \leq \sqrt{a_{k-1}} - \frac{1}{2} \leq (\sqrt{a_{k-2}} - \frac{1}{2}) - \frac{1}{2} \leq \ldots \leq \sqrt{a_0} - \frac{k}{2}$$

This implies that $k \leq 2 \cdot (\sqrt{a_0} - \sqrt{a_k}) = 2 \cdot \sqrt{(n)}$. □

Lemma 3.6 *If algorithm PEEL outputs a total of j chains and k antichains, these can be joined into an interconnection tree that has density no greater than $j + k$.*

Proof: Consider the j chains output by the algorithm. Even if we extend each chain to the top-right corner of the unit square, the total density of the resulting set of (extended) chains will not exceed j (see Figure 10). Similarly, the k antichains may be connected together by extending them all to the top-left corner of the unit square. A simple case analysis shows that the set of chains can then be connected to the set of antichains with no further increase in density, thus yielding a bound of $j+k$. □

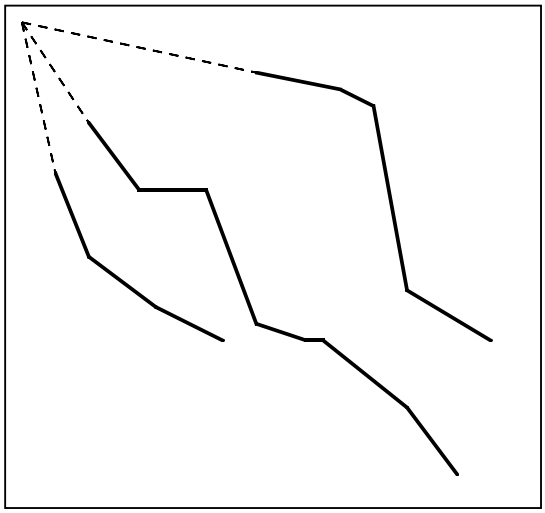

Figure 10: Combining chains into a low-density tree.

The maximum chain or antichain of a pointawt in the can be computed in time $O(n \log\log n)$ [10]. Since by Lemma 3.5 the total number of iterations of PEEL is $O(\sqrt{n})$, the time complexity of PEEL is $O(n^{\frac{3}{2}} \log\log n)$.

3.2 Cost Bounds

We now use probabilistic arguments to show that on average, all of our heuristic algorithms will also enjoy good performance with respect to the cost of the interconnection tree.

Theorem 3.7 *Given a net of n terminals with locations chosen randomly from a uniform distribution in the unit square, the expected cost of the minimum spanning tree is $\Theta(\sqrt{n})$.*

Proof: While this result is well-known from the theory subadditive functionals in the L_p plane [1] [16], we present the following simple proof. As in the proof of Theorem 3.2, we partition the unit square into an array of n identical square cells, each of size $\frac{1}{\sqrt{n}}$ by $\frac{1}{\sqrt{n}}$. In any interconnection tree $T(N)$,

each terminal will have at least one incident tree edge, and this edge must cross the boundary of the cell. It is easy to show that the expected distance from a terminal to the side of its containing cell is lower-bounded by some constant times the length of the side of this cell (in the Manhattan norm, this constant is 1/6). Recall that the expected number of cells that will contain at least one terminal is $(1 - \frac{1}{e}) \cdot n$. We therefore have an $\Omega(\sqrt{n})$ bound on the expected total cost of the interconnection tree. Since our algorithms always yield routing trees with cost $O(\sqrt{n})$ (see the following sequence of results), the minimum spanning tree cost for a set of n terminals uniformly distributed in the unit square is $\Theta(\sqrt{n})$ on average.[6] □

Theorem 3.8 *Given a net of n terminals with locations randomly chosen in the unit square, algorithm COMB constructs a spanning interconnection tree with cost* $\leq 2 \cdot \sqrt{n}$.

Proof: In the COMB construction, the sum of the vertical components of the edges within each strip is bounded by one (the height of each strip is one). Thus, the sum of the vertical components of all edges in the routing tree is bounded by $\sqrt{n}$. Consider the tree edges in each strip to be sorted by y-coordinate of the lower endpoint. Then, for the $\sqrt{n}$ first edges in these strips, the sum of horizontal components is bounded by one; similarly, the sum of the horizontal components of all the second edges in the strips is at most one, etc. Since the horizontal components of all of the edges therefore contribute at most $\sqrt{n}$ to the tree cost, the total tree cost is at most $2 \cdot \sqrt{n}$. □

Theorem 3.9 *Given a net of n terminals with locations randomly chosen in the unit square, algorithm COMB_ST constructs a Steiner interconnection tree with cost* $\leq \sqrt{2} \cdot \sqrt{n}$.

Proof: In the COMB_ST construction, the vertical spines contribute a total of $\frac{\sqrt{n}}{\sqrt{2}}$ to the tree cost. Sorting edges within each half-strip as in the proof of Theorem 3.8, we see that the horizontal components of the first edges in all half-strips is bounded by one, as is the sum of horizontal components of all second edges, etc. Therefore, the horizontal components of all edges contribute at most $\frac{\sqrt{n}}{\sqrt{2}}$ to the tree cost, and the desired bound follows. □

From these results, we have:

Corollary 3.10 *Given a set of n terminals whose locations are chosen randomly from a uniform distribution in the unit square, the algorithms COMB, COMB_ST and PEEL all return interconnection*

[6] Note that this result is also known from the theory of subadditive functionals in the Manhattan plane [1] [16].

trees which on average have both density and cost bounded by constants times optimal.

4 Triple Optimization

Often in VLSI design, we seek to simultaneously optimize more than one parameter; unfortunately, even optimizing each parameter of a design in isolation is usually intractable, and hence treating a composition of such optimizations would be even more complex. It is therefore quite unusual to be able to successfully optimize even two competing measures (a good example where this has been achieved is [6], where both total wirelength and radius are simultaneously minimized to within constants times their optimal values in the worst case, respectively).

In this section we show how to effectively combine density minimization with other, "performance-driven" objectives, as to simultaneously optimize up to three separate and competing measures of a routing tree.

4.1 Minimizing Skew, Density, and Total Wirelength

Minimizing clock skew is important in the design of high performance VLSI systems. Recall from the discussion of Section 1 that this may be viewed as construction of an interconnection tree with minimum difference among the various source-sink pathlengths. The work of [12] gives a general interconnection scheme that achieves extremely small clock skews, while keeping the total wirelength on average within a constant factor of optimal, and always bounded by $O(\sqrt{n})$. The clock routing construction of [12] egins with a forest of n isolated terminals, each of which is considered to be a tree with clock entry point (CEP) equal to the location of the terminal itself. The optimal geometric matching on these n CEPs yields $\frac{n}{2}$ segments, each of which defines a subtree with two nodes. The optimal CEP for each subtree of two nodes is the midpoint of the corresponding segment. In general, the matching operation will pair up the CEPs (roots) of all trees in the current forest. At each level, the root of each new merged tree is chosen to be the balance point which minimizes pathlength skew to the leaves of its two subtrees. Thus, at each level of this construction, we only match half as many points as in the previous level, and thus after $\lceil \log n \rceil$ matching iterations, we obtain the complete tree topology.

In order to construct clock routing trees with low density, our minimum density tree construction of Section 2 may be modified to yield a geometric matching with low density, as follows: partition

the net into $\sqrt{n}$ strips of $\sqrt{n}$ terminals each and connect the terminals of each strip in a long chain, as before (Figure 11a). However, instead of connecting the bottoms of all the chains, alternatingly connect the two bottoms of one adjacent pair of chains, and the two tops of the next adjacent chain pair, etc. (Figure 11b). It is easy to see (using arguments identical to those used in Theorems 3.8 and 3.3) that this procedure (which we call SERPENTINE) will connect all of the terminals in a single long chain having both total cost and overall density simultaneously bounded by $O(\sqrt{n})$ in the worst case.

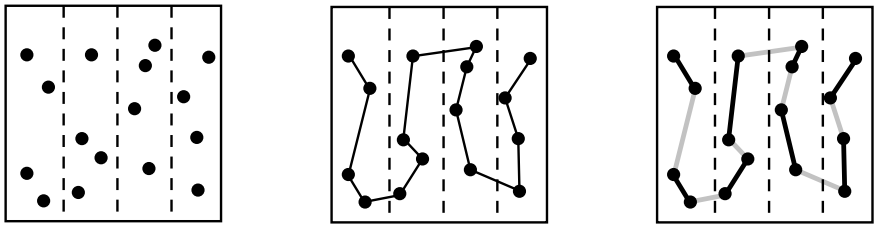

Figure 11: (a) Partitioning a net into strips/chain; (b) a tour with good average cost and low density; and (c) an embedded geometric matching with good average cost and low density.

Taking only every other edge of the resulting long chain will constitute a geometric matching (Figure 11c) having both total cost and overall density simultaneously bounded by $O(\sqrt{n})$. Next, we use such matchings within the minimum-skew routing method of [12], to yield clock routing trees that simultaneously minimize three (mutually competing) measures: pathlength skew, total wirelength, and density, with the latter two quantities being bounded on average by constants times the optimal values, respectively.

4.2 Minimizing Radius, Density, and Total Wirelength

Another example of a triple optimization is obtained if we combine our present bounded density formulation with the radius/cost tradeoff formulation of [6], where a method was proposed to uniformly trade off total routing tree cost with radius, achieving a simultaneous optimization of both cost and radius to within constants times the optimal respective values in the worst case. The bounded-radius tree construction of [6] starts with a low-cost tour of the net terminals, which is then augmented by adding shortest paths to the root from certain regularly spaced locations along the initial tour. The

final output routing tree is the shortest paths tree over the resulting augmented graph.

We can combine our present bounded radius method of Section 2 with the radius/cost tradeoff method of [6], by starting the basic algorithm of [6] with a tour having total cost and density both bounded by $2\sqrt{n}$, as described in Section 4.1, and then proceed normally with the rest of the construction of [6]. For an arbitrary given real parameter ϵ, the resulting routing tree will have radius bounded by $(1+\epsilon)\cdot R$, cost bounded by $(1+\frac{1}{\epsilon})\cdot 2\sqrt{n}$, and density bounded by $(1+\frac{1}{\epsilon\cdot R})\cdot 2\sqrt{n}$, where R is the distance from some distinguished source pin to the farthest sink ($R=\Theta(1)$ for a uniform distribution in the unit square). Note that for any fixed value of ϵ, all three of the above measures (i.e., radius, cost, and density) are on average constants times the respective optimal values (with the radius bound being constant times optimal in the *worst case* as well).

5 Experimental Results

We have implemented the COMB, COMB_ST and PEEL algorithms using ANSI C for both the Macintosh and Sun Sparc environments. Results are presented in Tables 1-2. For each pointset cardinality, each algorithm was executed on 100 point sets randomly chosen from a uniform distribution in the unit square. We computed the minimum, average, and maximum densities of the resulting interconnection trees. Statistics with respect to the routing tree density and given in Table 1.

The data indicates that the average density of the tree produced by the COMB algorithm is on par (indeed, up to XXX percent better) than the density of the minimum spanning tree. It should be noted that the density of the minimum spanning tree has considerably higher variance, and can be as great as $\Omega(n)$. On the other hand, the average density of the trees produced by the COMB_ST algorithm is considerably better than the average density of the corresponding minimum spanning trees. For example, for nets of size 10, COMB_ST yields trees with average density 2.89, while the average minimum spanning tree density is 3.65.

6 Future Work

It is still an open question whether there exists a polynomial-time algorithm that constructs a routing tree with both cost and density bounded by constants times the optimal values in the *worst case*. It is also unknown whether the MDIT problem is NP-complete. The chain-peeling method, PEEL, offers

net size	MST						SERPENTINE					
	density			lower bound			density			lower bound		
	min	ave	max	min	ave	max	min	ave	max	min	ave	max
3	1	1.69	2	1.00	1.69	2.00	1	1.69	2	1.00	1.69	2.00
5	2	2.57	4	1.00	1.35	3.00	2	2.70	3	1.00	1.41	3.00
7	2	2.97	5	1.00	1.52	3.00	2	3.64	4	1.00	1.88	3.00
10	2	3.82	6	1.00	1.85	3.00	3	3.71	4	1.00	1.80	2.00
15	3	4.35	6	1.33	2.08	3.00	3	4.95	5	1.50	2.38	2.50
20	4	4.98	8	1.33	2.15	4.00	4	4.98	5	1.67	2.14	2.50
30	4	5.99	8	1.67	2.05	3.50	6	6.00	6	3.00	2.00	2.11
;50	5	7.11	10	3.33	1.50	2.28	7	7.79	8	2.67	2.00	2.50
;100	7	9.48	12	1.75	2.37	3.00	10	10.01	11	2.50	2.50	2.75
;300	12	14.59	17	1.86	2.31	2.83	18	18.00	18	2.57	2.85	3.00
;500	15	17.79	22	1.88	2.23	2.75	23	23.00	23	2.88	2.88	3.29

net size	CHAIN						COMB_ST					
	density			lower bound			density			lower bound		
	min	ave	max	min	ave	max	min	ave	max	min	ave	max
3	1	1.69	2	1.00	1.00	1.00	1	1.00	1	1.00	1.00	1.00
5	2	2.00	2	0.50	0.94	2.00	2	2.00	2	1.00	1.05	2.00
7	2	2.66	3	0.50	1.03	2.00	3	3.00	3	1.00	1.46	1.50
10	2	3.08	4	1.00	1.85	3.00	3	3.00	3	1.00	1.46	1.50
15	3	3.93	5	1.00	1.58	2.00	3	3.00	3	1.00	1.44	1.50
20	4	4.76	6	1.00	1.74	2.50	4	4.00	4	1.33	1.72	2.00
30	5	5.88	7	1.33	1.81	3.00	5	5.00	5	1.67	1.76	2.50
50	7	7.85	9	1.50	2.27	2.67	6	6.00	6	1.50	1.92	2.00
;100	10	11.48	13	2.25	2.66	3.00	8	8.00	8	2.00	2.00	2.00
;300	19	20.69	22	2.57	3.12	3.50	13	13.00	13	1.86	2.06	2.17
;500	??	??.??	??	3.00	3.26	3.86	17	17.00	17	2.12	2.13	2.43

Table 1: Minimum density routing tree statistics.

some promise in the sense that there exist examples where this method outperforms the algorithms COMB and COMB_ST by a factor of $\Theta(\sqrt{n})$ (see Figure 12); we believe that PEEL can be shown to yield worst-case density that is within a constant factor of optimal. Indeed, we offer two closely related conjectures: (i) that the minimum density of a spanning tree over net N is at least the minimum of the number of chains or the number of antichains needed to cover N; and (ii) the PEEL algorithm will use at most two times the minimum possible number of chains/antichains that cover N.

In conclusion, we have proposed a new spanning and Steiner tree formulation, based on a minimum density criterion. We have also presented several efficient heuristics for constructing low-density spanning and Steiner trees. We prove that on average the performance of our algorithm is bounded by small constants away from optimal, in terms of both tree cost and density. We also show how our techniques can be used to unify the new density criterion with previous "performance-driven" interconnection

net	MST			SERPENTINE		
size	min	ave	max	min	ave	max
3	417	1103.66	2227	466	1210.13	2561
5	804	1658.39	2554	1010	2154.82	4233
7	1322	2039.34	2983	1520	2851.53	4427
10	1781	2662.36	3462	2635	3857.65	5334
15	2296	3224.41	4045	3253	4960.36	6231
20	2766	3789.89	4558	4351	5639.92	6845
30	4107	4651.00	5403	6273	7122.61	8469
50	5190	5945.47	6668	8003	9415.73	10478
;100	7481	8384.32	8887	12237	13079.22	14027
;300	13850	14318.99	14865	22488	23319.39	24164
;500	17840	18438.74	19079	29272	30065.21	31029

net	CHAIN			COMB_ST		
size	min	ave	max	min	ave	max
3	86	736.72	1577	366	1164.96	1882
5	758	1495.57	2481	1063	2260.09	3229
7	770	1852.91	3209	1992	3009.31	4141
10	1595	2776.06	4080	2307	3224.01	3974
15	2562	3721.27	5071	3143	4216.83	4941
20	2871	4720.69	6350	3692	4823.63	5649
30	4873	6318.27	8085	5594	6570.46	7740
50	7447	9298.73	11629	7070	8029.99	8945
;100	12552	14717.75	16579	10390	11083.93	11911
;300	26123	28960.44	31549	17963	18681.10	19614
;500	35249	39176.52	42265	24233	24930.63	25690

Table 2: Tree cost for different algorithms.

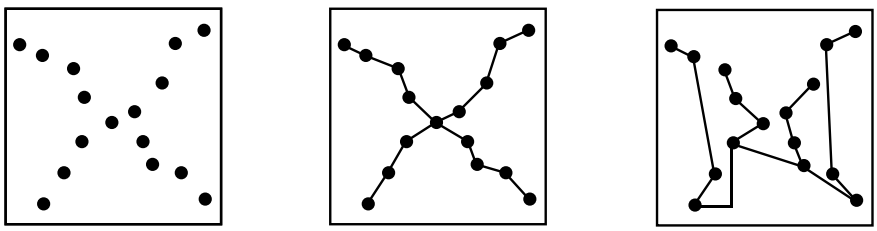

Figure 12: An example (a) on which PEEL (b) performs an unbounded factor better than either COMB or COMB_ST (c).

objectives in order to achieve simultaneous optimization of up to three separate interconnection tree measures. Extensive simulations indicate that this approach is effective in practice, and holds promise for applications to balanced-resource routing in VLSI layout. The main open questions remaining are whether there exists a polynomial-time algorithm that constructs a routing tree with both cost and

density bounded by constants times the optimal values in the *worst case*, and whether the MDIT problem is NP-complete.

References

[1] J. Beardwood, H. J. Halton, and J. M. Hammersley, *The Shortest Path Through Many Points*, Proc. Cambridge Philos. Soc., 55 (1959), pp. 299–327.

[2] K. D. Boese, J. Cong, A. B. Kahng, K. S. Leung, and D. Zhou, *On High-Speed VLSI Interconnects: Analysis and Design*, to appear in Proc. Asia-Pacific Conf. on Circuits and Systems, (1992).

[3] K. D. Boese and A. B. Kahng, *Zero-Skew Clock Routing Trees with Minimum Wirelength*, in Proc. IEEE Intl. ASIC Conf., Rochester, NY, September 1992, pp. 17–21.

[4] J. Cohoon and J. Randall, *Critical Net Routing*, in Proc. IEEE Intl. Conf. on Computer Design, Cambridge, MA, October 1991, pp. 174–177.

[5] J. Cong, A. B. Kahng, G. Robins, M. Sarrafzadeh, and C. K. Wong, *Performance-Driven Global Routing for Cell Based IC's*, in Proc. IEEE Intl. Conf. on Computer Design, Cambridge, MA, October 1991, pp. 170–173.

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

[7] R. P. Dilworth, *A Decomposition Theorem for Partially Ordered Sets*, Ann. Math, 51 (1950), pp. 161–166.

[8] H. Edelsbrunner, *Algorithms in Combinatorial Geometry*, Springer-Verlag, Berlin, 1987.

[9] C. Hilbert and C. Rathmell, *Multichip Modeuls: System Advantages, Major Constructions, and Materials Technologies*, IEEE Press, 1991.

[10] J. Hunt and S. Szymanski, *A Fast Algorithm for Computing Longest Common Subsequence*, Comm. of ACM, 20 (1977), pp. 350–353.

[11] M. A. B. Jackson, A. Srinivasan, and E. S. Kuh, *Clock Routing for High-Performance IC's*, in Proc. ACM/IEEE Design Automation Conf., 1990, pp. 573–579.

[12] A. B. Kahng, J. Cong, and G. Robins, *High-Performance Clock Routing Based on Recursive Geometric Matching*, in Proc. ACM/IEEE Design Automation Conf., June 1991, pp. 322–327.

[13] A. B. Kahng and G. Robins, *A New Class of Iterative Steiner Tree Heuristics With Good Performance*, IEEE Trans. on Computer-Aided Design, 11 (1992), pp. 893–902.

[14] B. T. Preas and M. J. Lorenzetti, *Physical Design Automation of VLSI Systems*, Benjamin/Cummings, Menlo Park, CA, 1988.

[15] D. Richards, *Fast Heuristic Algorithms for Rectilinear Steiner Trees*, Algorithmica, 4 (1989), pp. 191–207.

[16] J. M. Steele, *Growth Rates of Euclidean Minimal Spanning Trees With Power Weighted Edges*, Annals of Probability, 16 (1988), pp. 1767–1787.

[17] K. J. Supowit, E. M. Reingold, and D. A. Plaisted, *The Travelling Salesman Problem and Minimum Matching in the Unit Square*, SIAM J. Computing, 12 (1983), pp. 144–156.

[18] R. S. Tsay, *Exact Zero Skew*, in Proc. IEEE Intl. Conf. on Computer-Aided Design, Santa Clara, CA, November 1991, pp. 336–339.

[19] P. Winter, *Steiner Problem in Networks: A Survey*, Networks, 17 (1987), pp. 129–167.