Performance-Driven Global Routing for Cell Based IC's

J. Cong, A. Kahng, G. Robins, M. Sarrafzadeh (†), C. K. Wong (‡)

Dept. of Computer Science, UCLA, Los Angeles, CA 90024
(†) Dept. of EE & CS, Northwestern University, Evanston, IL 60208
(‡) IBM T. J. Watson Research Center, Yorktown Heights, NY 10598

Abstract

Advances in VLSI technology and the increased complexity of circuit designs cause performance to become an increasingly important constraint for layout. In this paper, we address the issue of delay optimization during the global routing phase. We formulate this problem as the construction of a bounded-radius spanning tree for a given pointset in the plane, and present a family of effective heuristics. Our approach has very good empirical performance with respect to total wirelength, and can be smoothly tuned between the competing requirements of minimum delay and minimum total netlength, as confirmed by extensive computational results which confirm this. Extensions can be made to the graph and Steiner versions of the problem, and a number of open problems are described.

1 Introduction

As VLSI fabrication technology advances, interconnection delay becomes increasingly significant in determining overall circuit speed. Recently, it has been reported that interconnection delay contributes up to 50% to 70% of the clock cycle in the design of very dense and high performance circuits [2] [16]. Thus, with submicron device dimensions and over a million transistors integrated on a single microprocessor, on-chip and chip-to-chip interconnections play a major role in determining the performance of digital systems.

Due to this trend, performance-driven layout design has received considerable attention in the past few years. However, most of the work in this area has been on the timing-driven placement problem. A number of methods have been developed to generate good placements wherein the blocks or cells in timing-critical paths are placed close together. The so-called zero-slack algorithm was proposed by Hauge, Nair and Yoffa [5]; fictitious facilities and floating anchors methods were used by Marek-Sadowska and Lin [12], and a linear programming approach was used by Jackson, Srinivasan and Kuh [7] [8]. Several other approaches, including simulated annealing, have also been studied [2] [11] [16]. Since no global routing solution is generally available at the placement step, most of these placement algorithms use the net bounding box semiperimeter to estimate the interconnection delay of a net.

While such techniques have been developed for timing-driven placement, only limited progress has been reported for the timing-driven interconnection problem. In [3], net priorities are determined based on static timing analysis; nets with high priorities are processed earlier using fewer feedthroughs. In [9], a hierarchical approach to timing-driven routing was outlined. In [13], a timing-driven global router based on the A* heuristic search algorithm was proposed for building-block design. However, these results do not provide a general formulation of the timing-driven global routing problem. Moreover, their solutions are not flexible enough to provide a trade-off between interconnection delay and routing cost.

In this paper, we give a solution for timing-driven global routing in cell-based design regimes. The method is motivated by considering the similar problem of finding minimum spanning trees of bounded radius. In particular, when given a lower bound R for the spanning tree radius, we search for spanning trees with radius $(1+\epsilon)\cdot R$. Such a formulation offers a very natural, smooth trade-off between the tree radius (maximum signal delay) and the tree cost (total interconnection length). This in turn affords the circuit designer a great deal of algorithmic flexibility, as the parameter ϵ can be varied depending on performance constraints. The timing-driven global router that we propose is based on several simple yet very effective heuristic algorithms for computing bounded radius minimum spanning trees. Extensive experimental results show that our global router reduces maximum source-sink pathlength by an average of 28% for 10-pin nets when compared with conventional minimum spanning tree based global routers. Moreover, our method indeed produces an entire class of routing solutions which embody the trade-off between minimum delay and minimum wire cost.

The remainder of this paper is organized as follows. In Section 2, we present a general formulation of the performance-driven global routing problem. In Section 3, we present a simple yet very effective heuristic algorithm for computing bounded radius minimum spanning trees, and analyze performance bounds for the algorithm. Section 4 describes several variations and improvements of this basic algorithm, and experimental results are reported in Section 5.

CH3040-3/91/0000/0170$01.00 © 1991 IEEE

2 Formulation of the Problem

A *net* N is a set of terminals to be connected where one of the terminals is a *source* and the rest are *sinks*. A routing solution of a net is a spanning tree T (called the *routing tree* of the net) which connects all of the terminals in the net. Since the routing tree may be treated as a distributed RC tree, we may use the first-order moment of the impulse response (also called Elmore's delay) to approximate interconnection delay [4] [15]. A more accurate approximation can be obtained using the upper and lower bounds on delay in an RC tree derived in [15]. However, although both the formula for Elmore's delay and those in [15] are very useful for simulation or timing verification, they involve sums of quadratic terms and are difficult to compute and optimize during the layout design process. Thus, a linear RC model (where interconnection delay between a source and a sink is proportional to the wire length between the two terminals) is often used to derive a simpler approximation for interconnection delay (e.g., [11] [14]). In this paper, we shall also use wire length to approximate interconnection delay in the construction of routing solutions. In practice, a subsequent iterative improvement step, based on a more accurate RC delay model, may be used to enhance the routing solutions.

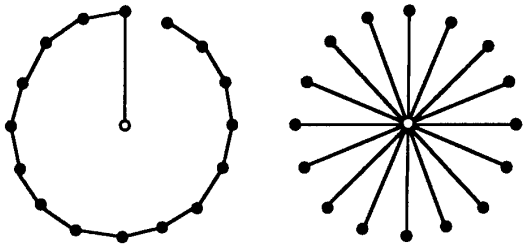

Figure 1: An example where the cost of a minimum-radius routing tree (right) is $\Omega(n)$ times larger than the cost of a minimum spanning tree (left).

The *radius* R of a *net* is the maximum Manhattan distance from the source to any sink in the net. The length or cost of an edge between points x and y is the Manhattan distance $dist(x, y)$. The *shortest path* in the routing tree T between any two terminals x and y, denoted by $minpath_T(x, y)$, is the sum of the lengths of all edges in the unique path from x to y in T. The cost of $minpath_T(x, y)$ is the sum of the costs of its edges, and is denoted $dist_T(x, y)$. We define the *radius* of a *routing tree*, $r(T)$, to be the maximum pathlength from the source to any sink. Clearly, $r(T) \geq R$ for any routing tree T. According to the linear RC delay model, in order to minimize the interconnection delay of a net, we want to minimize the radius of the routing tree since it measures the maximum interconnection delay between the source and any sink. If minimizing the radius was our only consideration, the global routing problem is trivial: simply connecting the source to every sink using the shortest path yields $r(T) = R$, which is the best possible result. However, the cost of this routing tree, i.e., the total edge length, might be very high. In fact, we can show that the cost of the tree can be $\Omega(n)$ times larger than the cost of the minimum spanning tree (MST), where n in the number of terminals in the net, as illustrated in Figure 1.

A routing tree with high cost may increase the overall routing area. Moreover, high cost also contributes to the interconnection delay which is not captured in the linear RC model. Therefore, neither tree shown in Figure 1 is particularly desirable. In order to balance the radius and the cost in the routing tree construction, we formulate the timing-driven global routing problem as follows:

The Bounded Radius Minimum Spanning Tree (BRMST) Problem: Given a parameter $\epsilon \geq 0$ and a signal net with radius R, find a minimum-cost routing tree T with radius $r(T) \leq (1 + \epsilon) \cdot R$.

The parameter ϵ controls the trade-off between the radius and the cost of the tree. When $\epsilon = 0$, we minimize the radius of the routing tree, and when $\epsilon = \infty$ we minimize the total cost of the tree. In general, as ϵ grows, there is less restriction on the radius, so we can further minimize the cost of the tree. For the example of Figure 2, we show three spanning trees obtained using different values of ϵ. Figure 2(a) shows the minimum radius spanning tree corresponding to the case $\epsilon = 0$, with maximum pathlength $r(T) = 6$; Figure 2(b) shows a solution with $r(T) = 10$ corresponding to the case $\epsilon = 1$; and Figure 2(c) shows the minimum cost spanning tree corresponding to the case $\epsilon = \infty$, with $r(T) = 14$. We can see that the tree cost decreases as the radius increases.

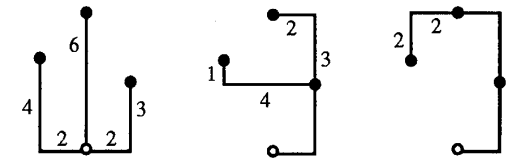

Figure 2: Examples showing how increasing the value of ϵ may result in decreased tree cost but increased radius.

The bounded-radius minimum spanning tree formulation provides a great deal of flexibility for our timing-driven global router. In practice, for nets in the timing-critical paths, the router uses small ϵ so that the interconnection delay is minimized. For nets not in any timing-critical path, the router uses large ϵ so that the total wire length is minimized.

According to the result in [6], constructing a minimum spanning tree with bounded radius in a general graph is NP-complete. However, this result does not imply that the BRMST problem is NP-complete since in our formulation, terminals are in the Manhattan plane (whose underlying graph is a grid graph) rather

than a general graph. So far, the exact complexity of the BRMST problem is still unknown, and the objective of this paper is to present a heuristic algorithm for the BRMST problem. Our goal is to construct a bounded-radius spanning tree with small cost.

3 Algorithm for Computing Bounded Radius Minimum Spanning Trees

Our basic algorithm for a net N finds a routing solution by growing a single connected component, following the general scheme of Prim's classical minimum spanning tree construction. We grow a tree $T = (V, E)$ which initially contains only the source s. At each step, we choose $x \in V$ and $y \in N - V$ such that $dist(x, y)$ is minimum. If adding (x, y) to T does not violate the radius constraint, i.e., $dist_T(s, x) + dist(x, y) \leq (1+\epsilon) \cdot R$, we include the edge (x, y) in T. Otherwise, we "backtrace" along the path from x to s to find the first point x' such that (x', y) is *appropriate* (i.e., $dist_T(s, x') + dist(x', y) \leq R$), and add (x', y) to the tree. In the worst case, the backtracing will terminate with $x' = s$, since the edge(s, y) is always appropriate. Note that in backtracing we could choose x' such that $dist_T(s, x') + dist(x', y) \leq (1 + \epsilon) \cdot R$. However, our choice of appropriate edges leads to fewer backtracing operations, while guaranteeing that backtracing is still always possible. In other words, we intentionally introduce some "slack" at y so that points within an ϵR neighborhood of y will not cause additional backtracing. Limiting the amount of backtracing in this way will keep the cost of the resulting tree close to that of the minimum spanning tree.

We call this algorithm the **Bounded Prim (BPRIM)** construction. The high-level description is given in Figure 3. We can show that the radius of the resulting tree is never greater than the radius of the MST whenever the MST is unique.

$T = (V, E) = (\{s\}, \emptyset)$
while $|V| < |N|$
  Select two points $x \in V$ and $y \in N - V$
    minimizing $dist(x, y)$
  if $dist_T(s, x) + dist(x, y) \leq (1 + \epsilon) \cdot R$
  then $V = V \cup \{x\}; E = E \cup \{(x, y)\}$
  else find first x' along path from x to s in T
      such that $dist_T(s, x') + dist(x', y) \leq R$
    $V = V \cup \{x'\}; E = E \cup \{(x', y)\}$

Figure 3: Computing a bounded-radius spanning tree T for a set of terminals N, with source $s \in N$ and radius R, using parameter ϵ.

With regard to total tree cost, we note that the difference between the BPRIM and MST tree costs will depend on the parameter ϵ. In practice, most nets will have between two and four pins. Furthermore, it is unlikely that a single gate will be used to drive more than six gates in CMOS design. In this case, we can show that the cost of the resulting tree is within a small constant factor of the cost of the MST for nets of practical size [1]. In fact, the experimental results of Section 5 suggest that the routing tree radius is still bounded by a small constant even for very large nets. However, examples exist which show that the worst-case performance ratio of BPRIM is not bounded by any constant for any value of ϵ.

4 Extensions of the Basic Algorithm

As it turns out, the bounded-radius construction can also be applied to minimum spanning tree methods other than Prim's algorithm. A more general algorithm template could be as follows:

$T = (V, E) = (\{s\}, \emptyset)$
while $|V| < |N|$
  Select two points $x \in V$ and $y \in N - V$,
    with $dist_T(s, x) + dist(x, y) \leq (1 + \epsilon) \cdot R$
  $V = V \cup \{x\}$; $E = E \cup \{(x, y)\}$

Figure 4: A more general bounded-radius spanning tree algorithm template.

This general template gives rise to a number of distinct variants, depending upon how the pair of points x and y are selected inside the inner loop. Several variants give significant performance improvements over the BPRIM algorithm:

- **H1** - Find x and y as in BPRIM, and select a terminal x' along $minpath_T(s, x)$ yielding a *minimum* appropriate edge (x', y).
- **H2** - Find a point $y \in N - V$ minimizing $dist(x, y)$ for any $x \in V$, and select a terminal $x' \in V$ yielding a minimum appropriate edge (x', y).
- **H3** - Find a pair of terminals $x \in V$ and $y \in N - V$ yielding a minimum appropriate edge (x, y).

It is easy to show that for BPRIM and for each of these variants, the radius of the routing tree is never greater than the radius of the MST whenever the MST is unique. However, when the MST is *not* unique, it is possible for the radius of the BPRIM (or any variant) construction to be arbitrarily larger than the radius of some MST. The time complexity of variants H1 and H2 is $O(n^2)$, while variant H3 can be easily implemented within time $O(n^3)$. Each of the above variants will also has unbounded worst-case performance ratio.

5 Experimental Results

The BPRIM algorithm and variants H1, H2, and H3 were implemented in ANSI C for the Sun-4, Macintosh and IBM environments; code is available from the authors. The algorithms were tested on a large number of random nets of up to 50 terminals, generated from a uniform distribution in the 1000 x 1000 grid. As noted in [10], any set of approximation heuristics induces a *meta-heuristic* which returns the best solution found by any heuristic in the set and has asymptotic complexity equal to that of the slowest original heuristic; we also implemented the meta-heuristic over H1, H2, and H3, with respect to minimum radius.

Although there exist examples where the BPRIM algorithm outperforms the more complicated variants, the data indicates that *on average*, variant H1 dominates BPRIM, H2 dominates H1, and H3 dominates H2. The smooth tradeoff between radius and cost is illustrated in figures 5 and 6 for typical values of ϵ.

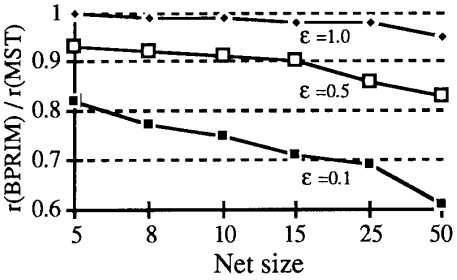

Figure 5: Average BPRIM routing tree radius, represented as a fraction of MST radius.

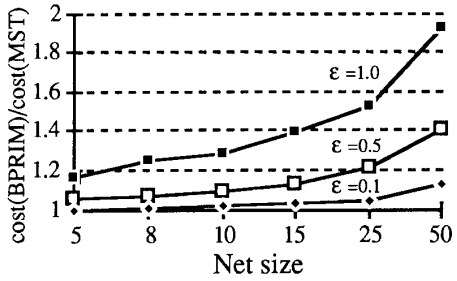

Figure 6: Average BPRIM routing tree cost, represented as a fraction of MST cost.

6 Extensions and Open Problems

Our basic algorithm and all of its variants readily extend to other norms and to alternate geometries (e.g., 45- or 30-60-90-degree routing regimes). In addition, the algorithm can be applied to arbitrary weighted graphs with non-metric edge weights, as in building-block and mixed-mode design. Extensions to performance-driven Steiner routing are also straightforward.

There are several interesting open problems. First, the complexity of the BRMST problem is still unknown. Also, it is open whether there is a polynomial time approximation algorithm for the BRMST problem with constant performance ratio (the constant would be independent of the problem size but depend on the value of ϵ). Moreover, for a given net, if the minimum spanning tree is not unique, it is unknown how to choose one with the minimum radius. Further studies of these problems are in progress.

7 Conclusion

We presented a family of global routing heuristics which construct a bounded-radius spanning tree for a given net. Our approach allows a smooth tradeoff between delay minimization and total wirelength, and has good empirical performance as well as efficient time complexity.

References

[1] J. Cong, A. Kahng and G. Robins, "Performance-Driven Global Routing for Cell Based IC's", UCLA CSD TR-900052, December 1990.

[2] W. E. Donath, R. J. Norman, B. K. Agrawal, and S. E. Bello, "Timing Driven Placement Using Complete Path Delays", *Proc. IEEE DAC*, 1990, pp. 84-89.

[3] A. E. Dunlop, V. D. Agrawal, D.N. Deutsch, M. F. Jukl, P. Kozak, and M. Wiesel, "Chip Layout Optimization Using Critical Path Weighting", *Proc. IEEE DAC*, 1984, pp. 133-136.

[4] W. C. Elmore, "The Transient Response of Damped Linear Networks with Particular Regard to Wide-Band Amplifiers", *J. Appl. Phys.* 19(1) (1948), pp. 55-63.

[5] P. S. Hauge, R. Nair, and E. J., Yoffa, "Circuit Placement for Predictable Performance", *Proc. IEEE ICCAD*, 1987, pp. 88-91.

[6] J. Ho, D. T. Lee, C. H. Chang, and C. K. Wong, "Bounded-Diameter Spanning Trees and Related Problems", *Proc. ACM Symp. on Computational Geometry*, 1989, pp. 276-282.

[7] M. A. B. Jackson, and E. S. Kuh, "Performance-Driven Placement of Cell-Based IC's", *Proc. IEEE DAC*, 1989, pp. 370-375.

[8] M. A. B. Jackson, A. Srinivasan, and E. S. Kuh, "Clock Routing for High-Performance IC's", *Proc. IEEE DAC*, 1990, pp. 573-579.

[9] M. A. B. Jackson, E. S. Kuh, and M. Marek-Sadowska, "Timing-Driven Routing for Building Block Layout", *Proc. IEEE Intl. Symp. on Circuits and Systems*, 1987, pp. 518-519.

[10] A. Kahng and G. Robins, "A New Family of Steiner Tree Heuristics With Good Performance: The Iterated 1-Steiner Approach", *Proc. IEEE ICCAD*, 1990, pp. 428-431.

[11] I. Lin, and D. H. C. Du, "Performance-Driven Constructive Placement", *Proc. IEEE DAC*, 1990, pp. 103-106.

[12] M. Marek-Sadowska, and S. P. Lin, "Timing Driven Placement", *Proc. IEEE ICCAD*, 1989, pp. 94-97.

[13] Y. Ogawa, M. Pedram, and E. S. Kuh, "Timing-Driven Placement for General Cell Layout", *Proc. International Symp. on Circuits and Systems*, 1990, pp. 872-876.

[14] P. Ramanathan and K. G. Shin, "A Clock Distribution Scheme for Non-Symmetric VLSI Circuits", *Proc. IEEE ICCAD*, 1989, pp. 398-401.

[15] J. Rubinstein, P. Penfield and M. A. Horowitz, "Signal Delay in RC Tree Networks", *IEEE Trans. on CAD* 2(3) (1983), pp. 202-211.

[16] S. Sutanthavibul, and E. Shragowitz, "An Adaptive Timing-Driven Layout for High Speed VLSI", *Proc. IEEE DAC*, 1990, pp. 90-95.