Accuracy and Fidelity of Fast Net Length Estimates

Joseph L. Ganley

Cadence Design Systems, Inc., 2615 John Milton Drive, Herndon, Virginia 20171

September 29, 1997

Abstract

Most popular tools for VLSI placement rely on some type of local search algorithm to iteratively refine a given placement solution. In such algorithms, it is necessary to evaluate the total amount of routing that a given placement will require. Typically, rectilinear Steiner tree heuristics are used to estimate the routing length of a placement. When evaluating heuristics, researchers typically focus on their *absolute* accuracy, i.e., how nearly optimal are their solutions? However, here a more pertinent statistic is their *relative* accuracy, i.e. how likely is it that a given heuristic will agree with the optimum on which of two instances has the shorter routing? In this paper, we experimentally evaluate four popular net length estimation heuristics, with respect to both their absolute and relative accuracy as well as their speed.

Keywords: rectilinear Steiner tree heuristics, accuracy, fidelity.

1 Introduction

In electronic physical design automation, the *placement* problem is that of choosing physical locations within some area for the devices that comprise a particular circuit. A variety of

objectives are typically to be optimized in placement, but a primary one is always the routability of the layout, which is in part measured by the total length of wire required to route the given placement. It is clearly far too expensive to compute an actual routing during the execution of a placement algorithm, so heuristics are used to approximate the routing length required by a placement solution.

Typically, these heuristics approximate the length of a *rectilinear Steiner tree*, since this is the shape often taken by the routing of a single net. The *rectilinear Steiner tree (RST)* problem is stated as follows: given a set T of n points called *terminals* in the plane, find a set S of additional points called *Steiner points* such that the length of a rectilinear minimum spanning tree of $T \cup S$ is minimized. Garey and Johnson [1] prove that the RST problem is NP-complete, indicating that a polynomial-time algorithm to compute an optimal RST is unlikely to exist.

When evaluating heuristics, researchers typically focus on their *absolute* accuracy, i.e., the magnitude of the discrepancy between the length measured by the heuristic and the length of an optimal RST. However, in the context of an iterative optimization algorithm, a more pertinent statistic is their *relative* accuracy, i.e. the probability that the heuristic will correctly order a given pair of instances according to the lengths of their optimal RSTs.

In this paper, we experimentally evaluate four popular net length estimation heuristics, with respect to both their absolute and relative accuracy, as well as their speed. Since these heuristics are to be applied within the inner loop of an iterative optimization algorithm, it is essential that they be very fast. Thus, we concentrate solely on heuristics with at most $O(n \log n)$ time complexity, where n is the number of terminals.

2 Definitions

We define the *accuracy* of a heuristic to be the average ratio between the length of an optimal RST and the length of one computed by the heuristic. In other words, if there

are m instances and h_i and s_i are respectively the objective values of the heuristic and optimal solutions to instance i, then the accuracy a is

$$a = \frac{\sum_{0 \leq i < m} \frac{s_i}{h_i}}{m}.$$

Boese, Kahng, McCoy, and Robins [2] define the term *fidelity* to describe the relative accuracy of a heuristic with respect to optimal. They define fidelity to be the average difference in rank between corresponding heuristic and optimal solutions.

We believe that since our interest is in pairwise comparisons between heuristic solutions, a more reasonable definition of fidelity is one used in the literature of approximate sorting; namely, the portion of the pairwise inequality relations among the optimal solutions that are correctly determined by the heuristic solutions. More formally, if there are m instances and h_i and s_i are respectively the objective values of the heuristic and optimal solutions to instance i, then the fidelity f is

$$f = \frac{|\{(i,j) : 0 \leq i < j < m, ((h_i - h_j)(s_i - s_j) > 0) \text{ or } (s_i = s_j)\}|}{\binom{n}{2}}.$$

3 Heuristics

This section describes the four heuristics that we examine, which are among the most popular for net length estimation in iterative placement applications.

3.1 Circumrectangle Semiperimeter

One popular net length estimate used in placement applications is the *circumrectangle semiperimeter (CRS)*, i.e. half of the perimeter of the smallest rectangle enclosing the input terminals. This is clearly a lower bound on the length of the optimal RST, and this is the only one of the estimates we examine that does not provide an actual heuristic RST.

The CRS can obviously be computed in linear time. Chung and Hwang [3] prove that the ratio between the length of an optimal RST and the length of the CRS is $O(\sqrt{n})$.

3.2 Spine Tree

Another net length estimate is the *spine tree*. Assume without loss of generality that the enclosing circumrectangle of the terminals has greater width than height. A spine tree consists of a backbone segment that spans the width of the circumrectangle, and whose y coordinate is the mean y coordinate of the terminals. Each terminal is then connected to the backbone by a vertical line segment.

The spine tree can also be computed in linear time. However, the worst-case ratio between the length of a spine tree and the length of an optimal RST is $O(n)$. (Consider a set of terminals that all lie on the top or bottom border of the circumrectangle.)

3.3 Minimum Spanning Tree

Another popular net length estimate is the *minimum spanning tree (MST)*. An MST is a minimum-length interconnection of the terminals if every edge has two terminals as end-points. For VLSI net length estimation purposes, typically the rectilinear (L_1) distance metric is used.

An MST can be computed in $O(n \log n)$ time, where n is the number of terminals (see, e.g., [4]). The worst-case ratio between the length of an MST and the length of an optimal RST was proven by Hwang [5] to be 3/2.

3.4 Prim-Steiner Tree

The MST can be easily improved upon by modifying Prim's MST algorithm [6] such that it can connect each new terminal to a point along an existing edge (creating a Steiner point), rather than solely to other terminals.

This Prim-Steiner tree can also be computed in $O(n \log n)$ time, and the worst-case ratio between the length of a Prim-Steiner tree and the length of an optimal RST is 3/2, just as for the MST [7].

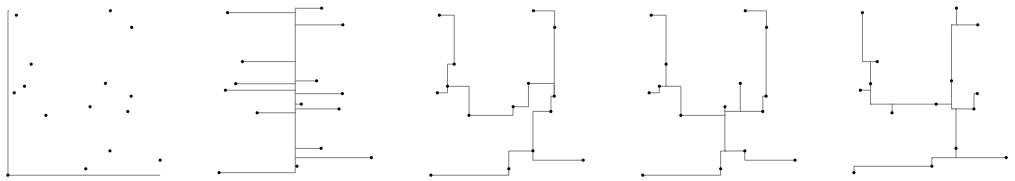

Figure 1: The four heuristic solutions and the optimal solution for a 15-terminal instance.

Figure 1 shows the four heuristic solutions and the optimal solution for a randomly generated 15-terminal instance.

4 Experimental Results

We have implemented the four heuristics and determined the accuracy, fidelity, and running time of each for 10,000 randomly generated instances of each size from $n = 3$ to $n = 15$. The optimal solutions used for comparison were generated by the FDP algorithm of Ganley and Cohoon [8, 9]. The results are summarized in Table 1.

There are several noteworthy features of these results.

The spine tree is often touted as a more accurate alternative to the CRS. However, in our tests, while the spine tree is more accurate for larger n, its fidelity is always worse than that of the CRS!

It has been suggested (e.g., by Kahng and Robins [10]) that one can improve the quality of the CRS estimate by multiplying the length of its solutions by the Chung-Hwang bound [3] of $O(\sqrt{n})$. Our experiments show that while this does improve the accuracy slightly for larger n, it obviously does not affect the fidelity of the heuristic (since it amounts to multiplying both sides of an inequality by the same value).

The MST is both more accurate and has higher fidelity than either of the linear-time heuristics, and these values appear to degrade much more slowly with increasing n. The same is true, only moreso, for the Prim-Steiner algorithm. However, one pays a substantial

	CRS			Spine Tree			MST			Prim-Steiner		
n	$1/a$	f	t	a	f	t	a	f	t	a	f	t
3	1.00	1.00	7.0	0.92	0.95	7.5	0.92	0.93	9.3	9.95	0.94	9.5
4	0.94	0.92	7.2	0.88	0.89	7.9	0.91	0.92	11.8	0.95	0.93	11.8
5	0.89	0.87	7.4	0.85	0.85	8.1	0.90	0.92	15.0	0.95	0.93	14.8
6	0.84	0.84	7.6	0.83	0.82	8.4	0.90	0.92	18.6	0.94	0.92	18.1
7	0.80	0.82	7.7	0.80	0.79	8.7	0.90	0.91	22.8	0.94	0.92	22.4
8	0.77	0.81	7.9	0.78	0.77	8.9	0.89	0.91	27.4	0.94	0.92	27.1
9	0.73	0.79	8.1	0.76	0.75	9.3	0.89	0.91	33.0	0.94	0.92	32.6
10	0.71	0.77	8.3	0.74	0.74	9.5	0.89	0.91	39.1	0.94	0.92	38.2
11	0.68	0.77	8.4	0.72	0.73	9.8	0.89	0.91	46.1	0.94	0.92	44.9
12	0.66	0.76	8.6	0.70	0.71	10.1	0.89	0.91	53.1	0.94	0.91	52.1
13	0.66	0.75	8.8	0.66	0.70	10.3	0.86	0.91	61.2	0.91	0.91	59.7
14	0.62	0.75	8.9	0.66	0.69	10.7	0.89	0.91	70.2	0.94	0.91	69.5
15	0.60	0.74	9.1	0.65	0.68	10.9	0.89	0.90	79.3	0.94	0.91	77.7

Table 1: Experimental results. n is the number of terminals, a is accuracy, f is fidelity, and t is running time in microseconds.

price in running time for this increased result quality.

5 Conclusions

The most interesting line of future work would be to devise a linear-time (or nearly so) heuristic whose fidelity does not decrease with increasing n as rapidly as do the CRS and spine tree. One potential candidate is Clarkson's nearly-linear approximate rectilinear minimum spanning tree algorithm [11].

One can easily show that unless P = NP, there does not exist a heuristic for the RST problem (or most any NP-complete problem) with fidelity 1.0, as such a heuristic could be used to solve the corresponding decision problem.

It would be interesting to extend these experiments to $n > 15$, by using a faster exact algorithm than the one used here (such as that of Salowe and Warme [12]).

It would also be interesting to test these heuristics against one another in an actual iterative placement algorithm.

References

[1] M. R. Garey and D. S. Johnson, "The rectilinear Steiner tree problem is NP-complete", *SIAM Journal on Applied Mathematics*, vol. 32, pp. 826–834, 1977.

[2] K. D. Boese, A. B. Kahng, B. A. McCoy and G. Robins, "Near-optimal critical sink routing tree constructions", *IEEE Transactions on Computer-Aided Design*, pp. 1417–1436, 1995.

[3] F. R. K. Chung and F. K. Hwang, "The largest minimal rectilinear Steiner trees for a set of n points enclosed in a rectangle with given perimeter", *Networks*, vol. 9, pp. 19–36, 1979.

[4] T. H. Cormen, C. E. Leiserson and R. L. Rivest, *Introduction to Algorithms*, MIT Press, Cambridge, Massachusetts, 1990.

[5] F. K. Hwang, "On Steiner minimal trees with rectilinear distance", *SIAM Journal on Applied Mathematics*, vol. 30, pp. 104–114, 1976.

[6] R. C. Prim, "Shortest connection networks and some generalizations", *Bell System Technical Journal*, vol. 36, pp. 1389–1401, 1957.

[7] A. B. Kahng and G. Robins, "On performance bounds for a class of rectilinear Steiner tree heuristics in arbitrary dimension", *IEEE Transactions on Computer-Aided Design*, vol. 11, pp. 1462–1465, 1992.

[8] J. L. Ganley and J. P. Cohoon, "Improved computation of optimal rectilinear Steiner minimal trees", *International Journal of Computational Geometry and Applications*, 1997, (to appear).

[9] J. L. Ganley and J. P. Cohoon, "A faster dynamic programming algorithm for exact rectilinear Steiner minimal trees", in *Proceedings of the Fourth Great Lakes Symposium on VLSI*, pp. 238–241, 1994.

[10] A. B. Kahng and G. Robins, "A new class of iterative Steiner tree heuristics with good performance", *IEEE Transactions on Computer-Aided Design*, vol. 11, pp. 893–902, 1992.

[11] K. L. Clarkson, "Fast expected-time and approximation algorithms for geometric minimum spanning trees", in *Proceedings of the Sixteenth Symposium on Theory of Computing*, pp. 342–348, 1984.

[12] J. S. Salowe and D. M. Warme, "35-point rectilinear Steiner minimal trees in a day", *Networks*, vol. 25, pp. 69–87, 1995.

Biography

Joseph L. Ganley received his Ph.D. in Computer Science in 1995 from the University of Virginia. His primary research interests are in VLSI physical design automation and graph and geometric algorithms. He is a member of ACM, SIGACT, SIGDA, and Tau Beta Pi. Since 1995 he has been a member of the research and development staff at Cadence Design Systems.