On Wirelength Estimations for Row-Based Placement*

Andrew E. Caldwell, Andrew B. Kahng, Stefanus Mantik, Igor L. Markov and Alex Zelikovsky
UCLA Computer Science Department, Los Angeles, CA 90095-1596
{caldwell,abk,stefanus,imarkov,alexz}@cs.ucla.edu

Abstract

Wirelength estimation in VLSI layout is fundamental to any pre-detailed routing estimate of timing or routability. In this paper, we develop new wirelength estimation techniques appropriate for top-down floorplanning and placement synthesis of row-based VLSI layouts. Our methods include accurate, linear-time approaches, often with sublinear time complexity for dynamic updating of estimates (e.g., for annealing placement). The new techniques offer advantages not only for early on-line wirelength estimation during top-down placement, but also for *a posteriori* estimation of routed wirelength given a final placement. In developing these new estimators, we have made several theoretical contributions. Notably, we have resolved the long-standing discrepancy between *region-based* and *bounding box-based* RSMT estimation techniques; this leads to new estimates that are functions of instance size n and aspect ratio AR.

1 Introduction

Wirelength estimation in VLSI layout is fundamental to any pre-detailed routing estimate of timing or routability. Accordingly, wirelength estimation has been studied in such contexts as gate-array routability [5], hierarchical top-down layout [4] [6] [18], floorplanning [9], and growth rates of rectilinear Steiner minimal trees [16] [17] [3]. Our present work is aimed at wirelength estimation *during* the synthesis of row-based placement.

We distinguish three basic types of wirelength estimations associated with placement: *a priori*, *a posteriori* and *on-line* estimations.[1]

- *A priori* estimation seeks to estimate the total wirelength of a layout design in advance, before placement. For example, a floorplanner may use such estimates to obtain rough measures of routability, RC parasitics and circuit performance; these in turn drive floorplan changes and circuit optimizations. For such estimates to provide leverage, they must be faster than the actual placement or routing constructions, at the cost of reduced accuracy. Such estimates are typified by the "wireload models" used in RTL floorplanning and logic optimization.

- *A posteriori* estimation occurs when we are given a fixed placement and want to estimate the post-routing wirelength. This is of value whenever routing requires significantly more CPU time than placement or wiring estimation. Typical applications include predicting the routability of gate array layouts [5] [6], estimating channel height in standard-cell layouts [13] [14], choosing between two competing placements, etc. Again, such estimates must be faster than actual construction of the routing. (Note that accuracy need not be perfect if the estimate has good "fidelity", i.e., for any two solutions the estimator correctly predicts which one is better even if the estimate of relative solution costs may be inaccurate.

- *On-line* estimation occurs when we want to estimate the wirelength *during* top-down hierarchical floorplanning or placement. This has many applications. For example, the estimate can be used to stop the placement process early, as soon as it becomes obvious that the placement process is leading to a bad solution. Early estimates of wirelength can also be used to shorten the feedback loops in timing- and wirelength-driven placement: clock tree synthesis, scan ordering, gate sizing, etc. may all be done earlier in the flow when good wiring estimates are available. Finally, wirelength estimates can be useful in determining the merit of local perturbations to the current solution. For example, the inner loop of a simulated annealing placer requires us to accurately estimate the quality of a proposed move. Since this incremental cost estimation is one of the main contributors to annealing placement runtime, an on-line estimator must be very fast. The accuracy of on-line wirelength estimation should be between those for *a priori* and *a posteriori* regimes, reflecting the available information (more information than *a priori*, less than *a posteriori*).

Previous Wirelength Estimation Techniques

In the present work, we equate "wirelength estimation" with "estimation of the rectilinear Steiner minimal tree (RSMT) cost".[2] The input to the RSMT problem is a pointset P of size $|P| = n$. P can be chosen randomly from a uniform distribution over a rectangular region R having width w_R and height h_R. Alternatively, we may know the minimum bounding box enclosing all points of P, having width w_{bb} and height h_{bb}. We now review the most relevant literature for this problem.

Growth rates of subadditive functionals in the Euclidean plane. The work of [17] [15], in a literature that stems from the seminal work of Beardwood et al. [1], shows that the expected cost (total tree length) of a Euclidean Steiner minimal tree over n uniformly random points chosen within a bounded plane region R of area $w_R \cdot h_R$ is proportional to $\sqrt{w_R \cdot h_R \cdot n}$ for sufficiently large n. The constant of proportionality, denoted by β, is dependent on the functional of the pointset (e.g., the minimum spanning tree cost, or the minimum traveling salesperson tour cost, have similar growth rates but different constants of proportionality β_{EMST}, β_{ETSP}, etc.).

MST-based methods. Hwang [10] shows that the rectilinear Steiner ratio (worst-case ratio of rectilinear minimum spanning tree (RMST) cost to RSMT cost) is $3/2$. Hence, $2/3$ times the RMST cost is a lower bound on RSMT cost. Since the RMST cost is an upper bound on RSMT cost, one might propose, e.g., $5/6$ times the RMST cost as an

*This research was supported by a grant from Cadence Design Systems, Inc.

[1] We do not discuss the class of *constructive* wiring estimators, which essentially construct the layout down to global or detailed routing in order to obtain an "estimate" of the wiring. Constructive estimators can be relevant to certain design methodologies, but we are more concerned with early and fast predictions that afford the leverage essential to forward synthesis.

[2] We understand that router outputs may not be the same as RSMTs (due to the routing heuristic, congestion, timing or noise constraints, and obstacles), and we understand that minimum wirelength is not perfectly correlated with minimum delay or maximum routability. Nevertheless, pure RSMT cost estimation remains a core technology within today's industry floorplanners, I/O pin optimizers, global and detailed placers, and related tools. Our ongoing work is developing extensions to model the effects of routability and performance optimization.

RSMT estimator that guarantees at most 16% error. Alternatively, empirical studies show that $\frac{cost(RSMT)}{cost(RMST)}$ averages around 0.88; see [12] for a review. While MST-based estimators are excellent, we avoid them because their implementation requires $\Omega(n \log n)$ runtime with fairly high constant, or else $\Omega(n^2)$ runtime.[3]

Bounding box based methods. In iterative improvement placers, the objective is typically based on the half-perimeter of the bounding box of pin locations for each net, i.e., the RSMT estimate is $w_{bb} + h_{bb}$. This is computed in linear time; given appropriate data caching, it can be updated in expected sublinear time when a cell is moved. The bounding box half-perimeter exactly gives the RSMT cost for 2- and 3-pin nets, and can be fairly accurate for larger nets if the bounding box aspect ratio becomes large (see below). However, during top-down placement or floorplanning, the pin locations are typically snapped to the centers of the *regions* in which they are located, after which the bounding box computation occurs. This can be an unbounded factor smaller than the correct value[4]; corrections for special cases have been proposed by Donath [4] and subsequent authors.

Chung and Hwang [3] study the worst-case cost of the rectilinear Steiner minimal tree (RSMT) over n points with bounding box dimensions w_{bb}, h_{bb}. The maximum value of $\frac{cost(RSMT)}{w_{bb}+h_{bb}}$ tends to $\frac{\sqrt{n}+1}{2}$ as $n \to \infty$. Several authors (e.g., Sechen) have noted that this result implies a correction factor to the bounding box half-perimeter estimate for nets with $|P| > 3$.

Hamada et al. [8] propose a purely *a priori* wirelength estimation based on local neighborhood analysis. Nets are expanded into cliques, and 2-neighborhoods of each are analyzed to obtain parameters of branching within the circuit. Multi-pin net wirelength estimates are inferred from these parameters and a physical model in which neighbors of a given cell compete for locations close to that cell.

Cheng [2] empirically estimates the probability of having a wire pass through any given point within a net bounding box when the net is routed. His methodology is equivalent to estimating the horizontal and vertical components of the RSMT cost, as a correction factor to the sum $(w_{bb} + h_{bb})$. The correction factor is a function of the net size n; [2] provides a table of such correction factors obtained by Monte Carlo methods.[5]

Our Contributions

In this paper, we develop new on-line wirelength estimation techniques appropriate for top-down floorplanning and placement synthesis of row-based VLSI layouts. Our methods include accurate, linear-time approaches (typically with sublinear time complexity for dynamic updating of estimates) to guide both iterative and top-down placement methods.

We make the following theoretical contributions.

- First, we develop new bounding box estimators for on-line wirelength estimation in top-down layout. Specifically, we give both an exact $O(n^2)$ algorithm and two $O(n)$ heuristics for computing the expected bounding box of n points with known distribution among k regions of a floorplan or hierarchical placement. Importantly, our heuristics are much faster that the exact algorithm even for small values of n, which makes them better choices in many applications.
- Second, we make new insights into the discrepancy between asymptotic results of Steele et al. (that expected RSMT cost is proportional to $\sqrt{w_R \cdot h_R \cdot n}$) and the accepted practice (that expected RSMT cost is proportional to $\sqrt{n} \cdot (w_{bb} + h_{bb})$).
- Third, we demonstrate why a practical estimator of expected RSMT cost cannot simply be based on a single constant β. Rather, such an estimator must be based on a set of values $\beta(n, AR)$ (where AR is the aspect ratio of the region within which points are randomly chosen, or else the aspect ratio of the pointset's bounding box).
- Fourth, we develop new wirelength estimators using pure analytic techniques (specifically, a direct combination of the first and third results above), empirical (table-lookup) approaches[6], and a combination of the table-lookup and analytic techniques.

We also make the following practical contributions.

- We show that our new estimators are substantially more accurate than previous methods that are used in industry tools, including bounding box methods and the method of Cheng [2].
- We validate our new wirelength estimators in the on-line context, using a top-down partitioning-based placement tool. We show that our new estimators can stably and accurately predict eventual total RSMT cost or total bounding box half-perimeter early in the top-down placement process. Thus, unpromising solutions can be pruned earlier in the top-down layout process.

2 The Bounding Box of n Random Points With Given Distribution Among k Rectangles

Hierarchical partitioning- and annealing-based placers maintain lists of rectangular regions and cells assigned to each region. Until the bottom level of the placement, cells may have no particular location, yet wirelength cost estimates are needed to drive further partitioning or annealing. We therefore estimate the expected Manhattan wirelength under the assumption that cells are uniformly distributed within the rectangular regions to which they are assigned. [7] In particular, for each net we estimate the expected half-perimeter of the bounding box, assuming each pin is uniformly distributed in the region to which its cell belongs. Formally, we are given N rectangles R_i, $i = 1, \ldots, N$, in the rectilinear plane, and we are given that the i^{th} rectangle contains n_i uniformly random points. We seek estimates of the expected width and height of the bounding box of all $n = \sum_{i=1}^{N} n_i$ random points.

The straightforward and often used heuristic – assuming that each cell is placed in the center of its rectangle (i.e., at its expected location) – can be smaller than the correct answer by an unbounded factor. In the example shown after Fact 4, this heuristic underestimates the expected distance between two pins by one third of the bounding box size, and can return a zero estimate when the true expected distance is large.[8]

Since the $x-$ and the $y-$coordinates of each pin are independent random variables with ranges in segments, we can estimate the two sides of the bounding box separately and add the results to obtain the expected half-perimeter. Since the expected side of the bounding box is simply the expected distance between the maximal and the minimal random point, finding these two expectations (i.e., of the maximal and

[3] As we will emphasize below, our work seeks *linear-time* estimators that fit a dynamic / on-line use model. We do recognize that MST-based estimators have the interesting feature that they return an actual topology, as opposed to just a cost estimate. Our experience with industrial deep-submicron libraries and process technologies is that with well-balanced and well-sized circuits the resistive interconnect effects are not dominant, e.g., lumped-capacitance or simple C_{eff} estimates as in [11] are adequate. Whether any routing estimate (as opposed to a detailed Steiner embedding) can be used for noise-avoidance is yet unclear.

[4] Consider two tall and thin regions next to each other; the distance between their centers is an unbounded factor smaller than the expected distance between two random points chosen from the two regions.

[5] A detailed survey of all RSMT estimates is beyond the scope of this draft. For example, we omit discussion of methods (e.g., the improvement of Donath's technique given in [18], or [9]) that estimate hierarchical interconnections as opposed to RSMTs.

[6] The empirical approach easily allows practical variants, e.g., a given routing tool's characteristics can be captured by using the router's results, rather than the output of an RSMT heuristic, to create the lookup tables.

[7] The techniques that we develop below apply to the case where there is a known non-uniform probability distribution for pin locations within a given region (e.g., [9]).

[8] We are not the first to notice this error, e.g., [9] cite Donath [4] as a source for simple correction factors in the cases of $N = 1, 2$. Our purpose in this section is to develop a more complete theory than in previous works, and to use it as the basis for novel wirelength estimators in subsequent sections.

minimal coordinates) will solve the problem. Let us specify a given rectangle R_i by its lower-left and upper-right corners $\{(a_i^x, a_i^y), (b_i^x, b_i^y)\}$ with $a_i^x \le b_i^x$ and $a_i^y \le b_i^y$. Then, the computation of the expected bounding box of the n points is given in Figure 1.

Computation of the Expected Bounding Box
Input: Rectangles $R_i = \{(a_i^x, a_i^y), (b_i^x, b_i^y)\}, i = 1 \ldots N$ each with n_i random points
Output: The expected width E_{width} and the expected height E_{height} of the bounding box of all points
For horizontal segments $[a_i^x, b_i^x]$ with n_i random points each: find E_{left}, the expected location of the leftmost point; find E_{right}, the expected location of the rightmost point
For vertical segments $[a_i^y, b_i^y]$ with n_i random points each: find E_{top}, the expected location of the topmost point; find E_{bottom}, the expected location of the bottommost point
Output $E_{width} = E_{right} - E_{left}$ and $E_{height} = E_{top} - E_{bottom}$

Figure 1: Computing the expected bounding box for n points distributed over N rectangles.

In the remainder of this section, we will deal with computing the expected location of the *leftmost* point, because computing any of E_{right}, E_{top}, or E_{bottom} obviously reduces to computing E_{left}:

The Expected Minimum Problem. *Given N segments $[a_i, b_i], i = 1, \ldots, N$ on the real line with n_i points distributed uniformly in the i^{th} segment, find the expected location of the point with minimum coordinate.*

The following subsection gives an exact $O(n^2)$ algorithm, and subsection 2.2 gives $O(n)$ and $O(n \log n)$ heuristics that we use as the basis of new estimators in later sections.

2.1 Exact Solution of the Expected Minimum Problem

We work with a random point P_i on a segment in terms of its *cumulative distribution function* $p_i(t) : [a_i, b_i] \to [0, 1]$, which gives the probability of the point appearing to the left of t. Thus, $1 - p_i(t)$ gives the probability of the point appearing to the right of t. The uniform distribution corresponds to the cumulative distribution $p_i(t) = \frac{t - a_i}{b_i - a_i}$.

We can extend cumulative distribution functions by 0 to the left from a_i and by 1 to the right from b_i, allowing us to deal with random points *supported* on different segments (i.e., taking non-zero and non-one values of the cumulative distribution only on their respective segments).

Fact 1 *For n independent random points with cumulative distributions $p_i(t)$, the distribution of the minimum is $1 - \prod_{i=1}^{n}(1 - p_i(t))$.* □

Fact 2 *The expected location of a random point with distribution $\tau(t)$ supported within $[A, B]$ is $E = B - \int_A^B \tau(t)\, dt$.*[9] □

Since a point distributed on $[a_i, b_i]$ is also distributed on any containing segment (but not vice versa), one can enlarge $[a_i, b_i]$ to any $[A, B]$ when considering products in Fact 1 and the theorems below. Facts 1 and 2 imply

Fact 3 *For n independent random points with cumulative distributions $p_i(t)$ supported within the segment $[A, B]$ (i.e. having its non-zero and non-one values within $[A, B]$), the expected minimum is*

$$E_{min} = B - \int_A^B (1 - \prod_{i=1}^{n}(1 - p_i(t)))\, dt = A + \int_A^B (\prod_{i=1}^{n}(1 - p_i(t)))\, dt$$

□

[9] Idea of the proof: Represent the cumulative distribution as the derivative of the distribution and integrate by parts.

Fact 4 *The expected minimum for k independent random points uniformly distributed on the segment $[0, 1]$ is $\frac{1}{k+1}$.* □

Example. If two points are uniformly distributed on $[0, 1]$ the leftmost is expected at $\frac{1}{3}$ and the rightmost — at $\frac{2}{3}$. Consequently, the straightforward estimate for the expected distance between two random points as the distance between their expectations (0 in this case) will be wrong by $\frac{1}{3}$ of the bounding box size for the region over which the points are distributed (or 100% of the correct result).

Theorem 2.1 *Consider n random points, each of which is independently and uniformly distributed on segment $[a_i, b_i]$, $i = 1, \ldots, n$. Let $a_i \le a_{i+1}$ and let $A = a_1$ and $B = \min b_i$. Then the expected minimum $E_{\Updownarrow\backslash}$ is*

$$E_{min} = A + \int_A^B \left(1 - \frac{t - a_1}{b_1 - a_1}\right) dt - \sum_{i=2}^{n} \int_{a_i}^{B} \frac{t - a_i}{b_i - a_i} \prod_{j=1}^{i-1} \left(1 - \frac{t - a_j}{b_j - a_j}\right) dt$$

□

The formula above can be computed in n steps, spending $O(n)$ time on each step to multiply $P(t)$ by a linear polynomial and integrate the result.

The Expected Minimum Algorithm
Input: segments $[a_i, b_i], i = 1..N$ each containing n_i random points
Output: The expected position E_{min} of the minimum point
Set $M = \min b_i$, the smallest of the right segment endpoints Discard all segments with left endpoint greater than M Sort the segments by left endpoints, such that $a_1 \le \ldots \le a_n$ Set $A = a_1$ $E = A + \int_A^M (1 - \frac{t - a_1}{b_1 - a_1}) dt$; $P(t) = 1$ **For each** $i = 2, \ldots, n$ $P(t) = P(t)(1 - \frac{t - a_{i-1}}{t - b_{i-1}})$ $E = E - \int_{a_i}^{B} \frac{t - a_i}{b_i - a_i} P(t) dt$
Output $E_{min} = E$

Figure 2: An $O(n^2)$ exact algorithm for expected minimum.

Theorem 2.2 *The Expected Minimum Algorithm (Figure 2) finds the expected minimum of n points uniformly distributed in segments $[a_i, b_i]$ in time $O(n^2)$.* □

Theorem 2.1 shows that the expectation of the minimum is computed starting from the expectation of one point, with a series of apparently exponentially decreasing negative corrections. This motivates the question of designing linear or near-linear time heuristics: *if we allow for small decreasing errors to the above corrections, the cumulative error will be small*.

2.2 Fast Estimation of the Expected Minimum

We now present two heuristics for finding the expected minimum which are significantly faster than the exact algorithm, not only asymptotically, but also for small values of n.

The linear-time heuristic starts with the segment for the first random point and gradually shifts both endpoints of this segment to the left as it goes sequentially through the list of all segments. The midpoint of the resulting segment gives an approximation of the expected minimum (see Figure 3).

The second heuristic is more accurate but slower, with $O(n \log n)$ runtime. It sorts all segments in decreasing order of their left endpoints

The Fast Expected Minimum Heuristic

Input: Segments $[a_i, b_i], i = 1 \ldots N$, each with one random point
Output: Approximate expected location of the leftmost point

1. $A = a_1$, $B = b_1$
2. **For each** of the $n-1$ remaining segments $[a_i, b_i]$ **do**
   if $a_i < A$, then swap the two segments $[A, B]$ and $[a_i, b_i]$
   if $a_i < B$
   then
   if $b_i \geq B$
   then $B = B - \frac{(B-a_i)^3}{3(b_i-a_i)(B-A)}$
   else $B = B - \frac{\frac{1}{3}(b_i-a_i)^2+(B-a_i)(B-b_i)}{(B-A)}$
3. **Output** $E_{min} = \frac{A+B}{2}$

Figure 3: A linear-time heuristic for expected minimum.

The Expected Minimum Heuristic

Input: Segments $[a_i, b_i], i = 1 \ldots N$ each with one random point
Output: Approximate expected location the leftmost point

1. Sort segments by left endpoints, such that $a_1 \geq \ldots \geq a_n$
2. Find the leftmost right endpoint $M = \min b_i$
3. Omit all segments with $a_i > M$
4. Apply Fast Expected Minimum Heuristic to remaining segments

Figure 4: A more accurate, $O(n \log n)$ expected minimum heuristic.

and finds the leftmost right endpoint $M = \min b_i$. The Fast Expected Minimum Heuristic is then applied to segments whose left endpoints are not greater than M (see Figure 4).

Step 2 of the Fast Expected Minimum Heuristic is based on the following

Proposition 2.3 *If $[a_1, b_1]$ and $[a_2, b_2]$ ($a_1 \leq a_2$) are two segments each containing one random point, then the expected minimum is equal to*

$$\frac{a_1+b_1}{2} - \frac{(b_1-a_2)^3}{6(b_1-a_1)(b_2-a_2)} \quad \text{if } b_1 \leq b_2,$$

or otherwise

$$\frac{a_1+b_1}{2} - \frac{\frac{1}{3}(b_2-a_2)^2-(b_2-a_2)(b_1-a_2)+(b_1-a_2)^2}{2(b_1-a_1)}$$

□

We replace a pair of segments with a new segment such that its middle approximates the expected minimum of random points in the two original segments. This can be viewed as approximating the cumulative distribution of the minimum (over the union of original segments) with a linear cumulative distribution (over the new segment). The approximation error of one such step is the difference between the original expected minimum and the middle of the new segment ("new expected minimum"). However, when the step is applied many times, additional error is incurred by our removing higher momenta of the expected minimum.

To show that the sorting step (Step 1) in the Expected Minimum Heuristic improves accuracy, consider segments $[a_1, b_1] = [0, 1]$ and $[a_i, b_i] = [\frac{1}{2}, \frac{1}{2}]$, $i = 2, \ldots, n$. For this input, the Fast Expected Minimum Heuristic correctly determines the expected minimum for the random points in the first two segments as $\frac{3}{8}$, but the right endpoint of the resulting segment is placed at $\frac{3}{4}$. All other segments $[a_i, b_i]$, $i = 3, \ldots, n$, further shift the right endpoint to the left. For sufficiently large n, the right endpoint will be at $\frac{1}{2}$ and the approximate expected minimum will be at $\frac{1}{4}$. On the other hand, the exact expected minimum is still $\frac{3}{8}$.

To assess the relative error of the Fast Expected Minimum Heuristic, we compute both the approximate and the exact expected *maximum* values as well. Then evaluate the relative error between the heuristic's approximated expected distance between maximum and minimum values, and the exact expected distance between maximum and minimum values. In the above example, the relative error of the Fast Expected Minimum Heuristic is 100% since the heuristic's approximated expected distance is $\frac{3}{4} - \frac{1}{4} = \frac{1}{2}$, while the exact expected distance is $\frac{5}{8} - \frac{3}{8} = \frac{1}{4}$); we believe that this is the worst case. Our Monte-Carlo experiments indicate that the average error of the Fast Expected Minimum Heuristic for random input[10] is about 1.1%.

On the other hand, for the input described above, the Expected Minimum Heuristic finds the exact expected minimum. Monte-Carlo experiments also confirm the benefit of the sorting step: for 10000 random inputs for each value of $n = 3, \ldots, 30$ the expected relative error of the Expected Minimum Heuristic was always less than 0.6%, and we never encountered any instance with relative error greater than 5%. Based on the symmetry of the problem, we also believe that the maximum relative error of the Expected Minimum Heuristic occurs in the case when all segments are the same. An error bound for this case is given by the following

Fact 5 *If all segments are identical, the maximum possible error of the Expected Minimum Heuristic is* $\approx 5.15\%$. □

This discussion suggests that the Expected Minimum Heuristic has small worst-case error. We leave determining the exact performance ratio as an open problem.

3 Expected RSMT Cost for n Random Points Distributed in a Plane Region

A literature on growth rates of subadditive functionals of pointsets, originating with Beardwood et al. [1] and continuing through works of Steele and Snyder [16] [17], establishes bounds on the expected RSMT cost, $E[c(RSMT)]$, for n points chosen uniformly at random in a region R. Specifically, we know that $E[c(RSMT)] \propto \sqrt{area(R) \cdot n}$ when n grows sufficiently large. The constant of proportionality β does not depend on the shape of the region.[11] Empirical evidence suggests that the expected value of the ratio $\frac{c(RSMT)}{\sqrt{area(R) \cdot n}}$ converges to the constant $\beta \approx 0.76$. If R is a rectangle, which is often appropriate in layout applications, then $area(R) = w_R \cdot h_R$ and $E[c(RSMT)] \propto \sqrt{w_R \cdot h_R \cdot n}$.

Our work in this section is motivated by an apparent contradiction. If the expected RSMT cost is proportional to the square root of the area $w_R \cdot h_R$ of a given region, why are all practical estimates based on the half-perimeter $w_R + h_R$ of the region? Put another way, if the theory suggests use of a *geometric mean* estimate, why have practitioners always used an *arithmetic mean* estimate?[12] In this section, we resolve this puzzle both theoretically and experimentally. We show that there is very substantial deviation from the $\sqrt{w_R \cdot h_R \cdot n}$ expected RSMT cost when the pointset is small and/or when the region R is non-square (i.e., has aspect ratio > 1). As it happens, these are precisely the conditions of interest for VLSI layout applications. The results of this section, along with those of the previous section, together allow us to develop new and highly accurate RSMT cost estimators in Section 4.

[10] We generated n ($n = 3, \ldots, 30$) random segments in the $(0,1)$-interval and found both the exact and the approximate expected distance between the maximum and minimum. We ran 10000 experiments for each n and found that the average relative error was largest (i.e., about 1.2%) for $n = 12$ while the maximum relative error was always less than 10%.

[11] The argument is simple. Tile the region with uniform small squares. Apply the known result for $R =$ the unit square to each small square, then join the "trees" in each small square together. The cost of joining is asymptotically negligible.

[12] The half-perimeter of the region is twice the arithmetic mean $\frac{w_R+h_R}{2}$. We know that many estimates scale as square-root of the number of. pins, whose average for nets in VLSI circuits is close to 2.

$c(RSMT)/\sqrt{n \cdot area}$									
	Aspect Ratio (AR)								
n	1	2	4	8	16	32	64	128	256
4	0.64	0.67	0.78	0.98	1.29	1.76	2.44	3.44	4.82
5	0.67	0.70	0.80	0.99	1.30	1.76	2.43	3.39	4.76
6	0.69	0.72	0.81	0.99	1.27	1.73	2.41	3.36	4.68
7	0.71	0.73	0.81	0.98	1.26	1.69	2.33	3.25	4.56
8	0.72	0.74	0.82	0.97	1.24	1.66	2.28	3.16	4.44
9	0.73	0.75	0.81	0.96	1.21	1.62	2.21	3.07	4.33
10	0.74	0.75	0.81	0.95	1.19	1.57	2.15	2.99	4.18
15	0.75	0.76	0.80	0.90	1.10	1.42	1.91	2.62	3.67
20	0.76	0.77	0.80	0.87	1.03	1.30	1.73	2.37	3.29
30	0.76	0.76	0.79	0.84	0.95	1.16	1.51	2.03	2.81

Table 1: Average values of $\frac{cost(RSMT)}{\sqrt{n \cdot area}}$ over 10000 random n-point samples in a rectangular region with aspect ratio AR.

We first show that the convergence of the ratio $\frac{E[c(RSMT)]}{\sqrt{area(R) \cdot n}}$ to β strongly depends on the shape of the region R even though the value of β is asymptotically independent of the shape. We confine our discussion to the relevant case of rectangular regions; this allows the shape of the region R to be expressed as an *aspect ratio* $AR = \frac{w_R}{h_R}$, where we assume without loss of generality that $w_R > h_R$.

Theorem 3.1 *$E[c(RSMT)]$ for n random points chosen uniformly in a rectangular region R is proportional to the aspect ratio $\frac{w(R)}{h(R)}$, when the aspect ratio is sufficiently large.*

Our theorem implies that we can reformulate the result from [1] as: $E[c(RSMT)] \propto \sqrt{area(R) \cdot n}$ for a sufficiently large number $n > N_0$ of random points chosen uniformly in a region R, where N_0 depends on the shape of the region R. We experimentally validate the theorem, as well as the original result from the literature, by the following experiment. We first generate $N = 10000$ random instances of n points, for $n = 4, 5, \ldots, 30$ (note that $n = 2, 3$ are not interesting), chosen from a uniform distribution in the rectangular region $[0,1] \times [0, AR]$, for values of aspect ratio $AR = 1, 2, 4, 8, ..., 512$. We then find the cost of a heuristic RSMT over the generated n points using the Batched Iterated 1-Steiner implementation of Griffith et al. [7], and divide this cost by $\sqrt{AR \cdot n}$.[13] Table 1 presents the resulting values $\beta(n, AR)$, which we know should converge to $\beta \approx 0.76$ by the theory of Beardwood, Steele et al. The plot of Figure 5 presents a portion of the Table 1 data in an alternate way; we give individual curves depicting the convergence of $\beta(n, AR)$ for different values of the aspect ratio AR. Notice that the convergence is slower for larger values of AR, and that the deviation of $\beta(n, AR)$ from β is larger when n is small. The wide separation of the curves for small n, and the slow convergence for large AR, explains the discrepancy between the theoretical result and the observed use in practice of half-perimeter based RSMT estimators.

4 Expected RSMT Cost of n Random Points Distributed in a Specified Bounding Box

While the results of the previous section allow improved estimates of $c(RSMT)$ for pointsets in a given (rectangular) region, such estimates can often be very rough. In practice, e.g., for *a posteriori* estimation, the bounding box of the RSMT instance (indeed, the entire pointset) is known. Intuitively, the more we know about the pointset, the better our estimate of $c(RSMT)$ should be. In this section, we theoretically and experimentally determine improvements in RSMT cost estimation that can be obtained when we know the pointset bounding box.

[13] In what follows, we will always use this Batched Iterated 1-Steiner implementation to approximate the (NP-hard) RSMT solution. Results in [7] indicate that this will overestimate the true RSMT cost by an average of less than a quarter percent for the instance sizes that we discuss.

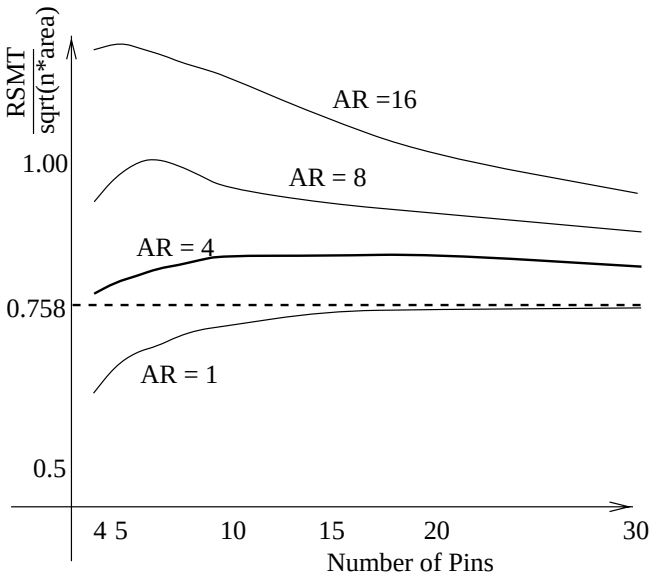

Figure 5: Plots of $\beta(n) = E[c(RSMT)]/\sqrt{area(R) \cdot n}$ for different aspect ratios AR of the rectangular region.

Avg. RSMT Cost for Unit Square								
	#points (n)							
	4	5	6	8	10	15	20	30
RSMT	1.28	1.50	1.69	2.04	2.33	2.91	3.38	4.15
90%	40.6	34.8	31.0	25.1	20.6	16.1	13.0	10.0
95%	48.1	40.8	36.8	29.7	24.3	18.9	15.4	12.0
98%	57.2	48.0	43.2	34.8	29.3	22.6	18.4	14.5

Table 2: Average RMST costs over 10000 random n-point samples in a unit square. Maximum relative deviation from average (expressed as a percentage) is computed for "best" 90%, 95% and 98% of samples.

We begin with an empirical demonstration of the gain from knowing the bounding box. We generate $N = 10000$ random instances of n points ($n = 4, 5, \ldots, 30$) chosen from a uniform distribution in the unit (1×1) square. The first row of Table 2 shows average values of RSMT cost over the N samples for each value of n. (These correspond the first column of Table 1, scaled by factors of $\sqrt{n}$.) The table also shows the maximum relative deviation from this average (expressed as a percentage) among the 90%, 95% and 98% of the instances.[14]

Similarly, in Table 3, the first row gives averages over $N = 10000$ samples of the *ratio* of the RSMT cost *divided by the half-perimeter of the pointset's bounding box*. Each column again gives maximum relative deviations in the middle 90%, 95% and 98% of the data, expressed as percentages. We see that normalizing to the bounding box half-perimeter yields a greatly improved estimate.

Finally, we make a small digression to indicate how far off these estimates are from "best possible" non-constructive estimates, namely, those based on the rectilinear MST construction. Recall that RMST cost is known to average around 12% greater than RSMT cost (cf. analyses of Bern and de Carvalho, as reviewed in [12]). Table 4 gives the aver-

[14] The exact calculation is as follows. For each of 10000 samples we find the relative deviation of $c(RSMT)$ from the average cost of RSMT for a given n. The we rank all relative deviations for each given value of n. To find, say, the maximum relative deviation over the 90% of samples, we determine the $90th$ percentiles of the rank order. For example, we see from the table that the middle 90% of all 10000 7-point instances have $c(RSMT)$ within 27.003% of the average value of $c(RSMT)$, which is 1.8748.

Average of RSMT Cost / Bounding Box Half-Perim.								
	#points (n)							
	4	5	6	8	10	15	20	30
NRSMT	1.06	1.13	1.19	1.31	1.42	1.66	1.87	2.22
90%	10.5	11.8	14.2	14.4	13.4	11.6	10.2	8.5
95%	14.5	15.2	15.8	16.7	15.6	13.8	12.1	10.1
98%	20.0	18.3	18.7	18.9	18.6	16.4	14.4	12.2

Table 3: First line (NRSMT) gives averages of RSMT cost divided by the half-perimeter of the pointset's bounding box. The 10000 n-point samples are taken from a uniform distribution in the unit square. Maximum relative deviation (expressed as a percentage) is computed for "best" 90%, 95% and 98% of the samples, respectively.

Average of RMST Cost / RSMT Cost in Unit Square								
	#points (n)							
	4	5	6	8	10	15	20	30
MST	1.10	1.11	1.11	1.11	1.12	1.12	1.12	1.12
90%	9.4	9.3	8.2	6.7	6.2	4.8	4.2	3.3
95%	12.5	10.5	9.6	7.9	7.3	5.8	5.0	3.9
98%	16.5	13.2	11.8	9.3	8.7	7.0	5.8	4.7

Table 4: First line (MST) gives average ratios of RMST cost divided by RSMT cost. The 10000 n-point samples are taken from a uniform distribution in the unit square. Maximum relative deviation (expressed as a percentage) is computed for "best" 90%, 95% and 98% of the samples.

age ratio of RMST cost over RSMT cost, for the same 10000 random instances for each value of n. This allows us to compare the bounding box estimator with the minimum spanning tree estimator. We conclude that the bounding box-based estimator is much better than the region-based estimator, and that the MST-based estimator is somewhat better still. However, as we noted earlier, the MST is too expensive to be used in practice – we require linear-time on-line wirelength estimators with sublinear update costs (based on reasonable storage) in the iterative placement context.

A New Connection Between Region-Based and Bounding Box-Based Estimation

From the results of Section 3 we know there is a dependency of RSMT cost on the aspect ratio of the *region* from which points are chosen. When we also consider the previous results of this section, we are motivated to seek a dependency of RSMT cost on the the aspect ratio of the *pointset bounding box* as well. Intuitively, if we can estimate this dependency, we will be able to more accurately predict the RSMT cost for random points with a known bounding box. In the remainder of this section, we will give a formal basis for such an estimation, then confirm our ideas experimentally.

The results of Section 3 allow us to estimate RSMT cost for random pointsets chosen in a *region* with given aspect ratio, based on empirical values of $\beta(n, AR)$. From the previous results of this section, we know that estimates that exploit knowledge of the pointset bounding box are more accurate than estimates that use only knowledge of the region from which the pointset was chosen. The difficulty is that the $\beta(n, AR)$ values from Section 3 cannot be directly applied when we have a specific pointset with a specific bounding box. We resolve this difficulty as follows.

- From Fact 4, we know that if we choose k random points in a rectangular region with sides w and h, the expected sides of the bounding box of these points are $w' = w(1 - \frac{2}{k+1})$ and $h' = h(1 - \frac{2}{k+1})$.
- Therefore, if we are given k random points having a known *bounding box* with sides w_{bb} and h_{bb}, we may predict that the expected RSMT cost is the same as the expected RSMT cost of k random points chosen in the *region* having sides $w_R = \frac{k+1}{k-1} \cdot w_{bb}$ and $h_R = \frac{k+1}{k-1} \cdot h_{bb}$.
- We can then apply lookup in Table 1 to estimate the RSMT cost over the k points.[15]

The above recipe defines a new type of estimator that is the first to combine the classic region-based and bounding box-based approaches. We have confirmed that wirelength predictors based on the approach of computing $w_R = \frac{k+1}{k-1} \cdot w_{bb}$, $h_R = \frac{k+1}{k-1} \cdot h_{bb}$ differ from empirically constructed (i.e., via Monte Carlo experiments) predictors by less than 1%.

In the empirical approach (which is analogous to the construction of Table 1, we generate a random set of n points within a bounding box of prescribed width w and height h in the following way:

1. generate a random set of n points in the 1×1 square;
2. find the bounding box of this set of points, with dimensions w' and h'; and
3. multiply all x- and y-coordinates of the points by w/w' and h/h', respectively.

We use this construction in finding RSMT costs over $N = 10000$ samples of n random points with bounding box aspect ratio $AR = 1, 2, ..., 32$, and dividing by the half-perimeter of the bounding box. Table 5 shows the average values for this ratio of RSMT cost to bounding box half-perimeter, as a function of n and AR. A similar empirical analysis was performed by Cheng [2], but the corresponding coefficients did not depend on aspect ratio. Additionally, the author of [2] used a worse code for finding heuristic RSMTs. As a result, the entries of the first row of Table 5, which we reproduce from [2], are larger than our entries by approximately 2% even in the case of $AR = 1$. In practice, Cheng's method will produce even worse overestimates of RSMT cost because it ignores the effect of bounding box aspect ratio.[16]

5 Practical On-Line Wirelength Estimation

To see the practical value of our new estimators, we first observe that they are extremely efficient.

- In the on-line (top-down placement) context, we have n pins distributed among various regions. Instead of returning the bounding box of the region centers, we apply the linear-time heuristic of Figure 3 in Section 2 to obtain the expected bounding box of the pins, then perform lookup (with linear interpolation as appropriate) in Table 5 of Section 4.
- In the *a posteriori* context, we have n pins in exact locations. Instead of returning any of the previous bounding box based estimates, we perform lookup (again with linear interpolation as appropriate) in Table 5.

The time complexity of our estimates in each context is $O(n)$. Updating the estimate when a pin is moved, as long as we have exact locations and do not need to execute the heuristic of Figure 3, requires

[15] This simple approach is very accurate, as we will discuss next. It may be possible to improve its accuracy by considering all possible region dimensions w_R and h_R, weighted by the likelihood that the pointset was actually chosen from within those region dimensions. However, we believe that the advantages of such a complicated extension will be minimal.

[16] N.B.: Readers may notice that the entries for $AR = 1$ in Table 5 are slightly different from the entries in the first row of Table 3. This is because the Table 3 entries are averaging over all bounding box aspect ratios, not just $AR = 1$. The expected bounding box aspect ratio for random points is somewhere between 1 and 2, and this is consistent with the data in the two tables.

Average RSMT Cost for Pointsets With BBox Half-Perimeter = 1								
	#points (n)							
AR	4	5	6	8	10	15	20	30
1*	1.08	1.15	1.22	1.34	1.45	1.69	1.89	2.23
D	12.2	13.4	15.5	15.2	14.4	12.7	11.4	9.52
1	1.06	1.13	1.19	1.32	1.42	1.66	1.87	2.22
D	11.2	12.5	14.5	14.1	13.5	11.6	10.3	8.64
2	1.05	1.11	1.16	1.27	1.36	1.59	1.78	2.10
D	9.83	10.3	12.3	12.6	12.6	11.2	10.1	8.64
4	1.03	1.07	1.11	1.18	1.25	1.41	1.57	1.84
D	6.87	6.93	8.50	9.54	9.98	9.87	9.35	8.13
10	1.01	1.03	1.05	1.08	1.12	1.21	1.29	1.45
D	3.37	3.44	4.39	5.05	5.53	6.18	6.51	6.53

Table 5: Each entry represents an average, over 10000 samples of n random points having prescribed bounding box aspect ratio, of RSMT cost divided by bounding box half-perimeter. The first row reproduces coefficients from the paper by Cheng [2]. Each row marked with D gives the maximum relative deviation from the average in 90% of the samples, expressed as a percentage.

Test Case	Number of Cells	Number of Nets
Test1	1756	1492
Test2	3286	2902
Test3	6692	6527
Test4	12133	11828
Test5	12857	10880

Table 6: Parameters of five standard-cell test cases from industry.

Average Relative Error of Wire Length Estimates										
	% of levels completed									
Estimator	10	20	30	40	50	60	70	80	90	100
CBB	.243	.185	.138	.110	.086	.059	.031	.011	.001	.000
HBB	2.36	1.59	.911	.511	.298	.169	.071	.029	.002	.000
HBB0	.109	.071	.038	.032	.020	.016	.011	.008	.000	.000
HBB6	.191	.110	.050	.025	.025	.022	.015	.006	.001	.000
Cheng	.208	.139	.085	.051	.038	.027	.039	.057	.065	.065
CBBtab	.259	.188	.132	.104	.080	.057	.032	.016	.010	.010
HBBtab	2.37	1.61	.920	.509	.293	.164	.066	.025	.010	.010
HBB0tab	.140	.097	.072	.047	.026	.020	.011	.008	.010	.010
HBB6tab	.120	.069	.061	.045	.046	.040	.032	.018	.010	.010

Table 7: Relative errors of estimated sums of bounding box half-perimeters and RSMT costs during the top-down placement. Data for 5 testcases are normalized and averaged.

only a constant number of operations after the net bounding box has been *updated*. Thus, speedups of bounding box *updates* that are used in practice (see, e.g., TimberWolf-related papers of Sechen et al) hence transfer directly into our methods. Having established efficiency, we next show that our methods lead to improved estimation accuracy in the on-line context.

We have incorporated our new estimates, along with the previous center-based, bounding box-based, and Cheng [2] estimates, into an internal placement testbed that includes both top-down partitioning and annealing engines.[17] Specifically, we compare the following nine wirelength estimates (the first four algorithms estimate half-perimeters of net bounding boxes and the rest five algorithms estimate Steiner tree lengths taking in account the net sizes and the aspect ratios of bounding boxes):

- CBB : Standard bounding box estimate using the center coordinates of the region in which the given pin is located
- HBB : Heuristic bounding box estimate using the linear-time heuristic of Figure 3
- HBB0 : Heuristic BBox with length scaled down to zero for nets completely contained in a region
- HBB6 : Heuristic BBox assuming that regions have dimensions that are scaled to 1/6 of their actual values (with the scaled regions having the same centers as the original regions)
- Cheng : CBB estimate, scaled by the coefficients of Cheng [2] (reproduced in the first line of Table 5)
- CBBtab : CBB estimate, followed by lookup in Table 5
- HBBtab : HBB estimate, followed by lookup in Table 5
- HBB0tab : HBB0 estimate, followed by lookup in Table 5
- HBB6tab : HBB6 estimate, followed by lookup in Table 5

Notice that if we blindly follow the uniform distribution assumption, we will tend to overestimate wirelengths during the early stages of the top-down placement process. This is because cells that are connected by a net will end up closer together than predicted by the random model, due to the contribution of the net to the placement objective. Thus, we have proposed two corrections to our basic models. (1) In the HBB0 and HBB0tab estimates, we assign a zero bounding box half-perimeter to nets that still belong to the same block (i.e., region) of the top-down partitioning. This captures the fact that most of these internal nets will end up being placed with their cells close together (a consequence of Rent's rule). If a net already has its cells in different regions, we apply our regular estimate since we already know these cells must end up in separated locations. (2) An alternate way of modeling the fact that cells that share a net should end up closer together than random cells is to restrict the possible cell locations. To do this, we scale the dimensions of each region by one-sixth[18], while keeping the region centered at the same coordinates. This leads to the HBB6 and HBB6tab estimates.

Our experiments thus far have evaluated the accuracy of on-line wirelength estimation in a top-down partitioning-based placer. We have used five standard-cell test cases Test1,...,Test5, obtained from industry; their parameters are given in Table 6.

Our results are given in Table 7. For each test case, we run the top-down partitioning based placer to completion, then measure both total net bounding box half-perimeter and total Iterated 1-Steiner heuristic RSMT cost of the result. For the bounding box estimators (the first four rows of each table), each table entry gives the relative error of the estimated sum of bounding box half-perimeters after the first $i \cdot 10\%$ ($i = 1, \ldots, 10$) levels of the top-down partitioning based placement, versus the final sum of bounding boxes. (We report data for every 10% of the levels because the number of placement levels varies according to instance size.) For the wirelength estimators (the last five lines of each table), each table entry gives the relative error of the estimated sum of Steiner tree costs, versus the final sum of I1S heuristic RSMT costs. Table 7 gives the average of all the values. We see that our new estimators are *substantially* better in the on-line context than previous methods of estimating either sum of bounding box half-perimeters or sum of RSMT costs. In other words, we can obtain accurate estimates of the final values of these objectives, relatively early in the top-down placement process. This allows pruning of bad solution paths, and large potential runtime savings. At the same time, even for *a posteriori* estimation our new methods are superior to previous approaches.

[17] Placements that we obtain from our internal testbed are competitive – in terms of runtime, various solution metrics, and routability by industrial routers – with placements from industry placers that we are aware of.

[18] The empirical factor of $1/6$ is a result of fine-tuning our estimates.

6 Conclusions and Future Work

We have developed new wirelength estimation techniques appropriate for top-down floorplanning and placement synthesis of row-based VLSI layouts. Our methods give accurate, truly linear-time approaches, typically with sublinear time complexity for dynamic updating of estimates (e.g., for annealing placement). The new techniques offer advantages not only for early on-line wirelength estimation during top-down placement, but also for *a posteriori* estimation of routed wirelength given a final placement. In developing these new estimators, we have made several theoretical contributions. Notably, we have resolved the long-standing discrepancy between *region-based* and *bounding box-based* RSMT estimation techniques; this leads to new estimates that are functions of instance size n and aspect ratio AR.

We have validated our new techniques experimentally using test cases from industry; the HBB0 and HBB0tab are substantially superior to previous methods. Our ongoing research addresses such issues as (1) confirming that our new cost estimates can successfully drive partitioning- and annealing-based placers to improved solutions (preliminary results are quite promising); and (2) improving the intuitions that led to the HBB0/6 and HBB0/6tab refinements of our original estimators.

The assumption about uniform distribution of cells in bounding boxes may not hold in some VLSI CAD applications. We believe that our methods can be applied if actual distributions are available, in particular, our exact algorithm for expected minimum can accommodate many piece-wise polynomial distributions.

Extending our present results to non-uniform cell distributions appearing in top-down placement presents an intriguing direction for future work.

REFERENCES

[1] J. Beardwood, J. H. Halton, and J. M. Hammersley. The shortest path through many points. In *Proceedings of the Cambridge Philosophical Society 55*, pages 299–327, 1959.

[2] C.-L. E. Cheng. Risa: Accurate and efficient placement routability modeling. In *Proceedings IEEE International Conf. on Computer-Aided Design*, pages 690–695, 1994.

[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*, 9(1):19–36, Spring 1979.

[4] W. E. Donath. Placement and average interconnection lengths of computer logic. *IEEE Transactions on Circuits and Systems*, CAS-26(4):272–277, April 1979.

[5] A. A. El Gamal. Two-dimensional stochastic model for interconnections in master slice integrated circuits. *IEEE Transactions on Circuits and Systems*, CAS-28(2):127–138, February 1981.

[6] M. Feuer. Connectivity of random logic. *IEEE Transactions on Computers*, C-31(1):29–33, January 1982.

[7] J. Griffith, G. Robins, J. S. Salowe, and T. Zhang. Closing the gap: Near-optimal steiner trees in polynomial time. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 13(11):1351–1365, November 1994.

[8] T. Hamada, C.-K. Cheng, and P. M. Chau. A wire length estimation technique utilizing neighborhood density equations. In *Proc. ACM/IEEE Design Automation Conf.*, pages 57–61, 1992.

[9] W. Hebgen and G. Zimmermann. Hierarchical netlength estimation for timing prediction. In *Proceedings of the ACM/SIGDA Physical Design Workshop*, pages 118–125, 1996.

[10] F. K. Hwang. On steiner minimal trees with rectilinear distance. *Siam Journal of Applied Mathematics*, pages 104–114, January 1976.

[11] A. B. Kahng and S. Muddu. Efficient gate delay modeling for large interconnect loads. In *Proceedings of the IEEE Multi-Chip Module Conference*, pages 202–207, 1996.

[12] A. B. Kahng and G. Robins. *On Optimal Interconnections for VLSI*. Kluwer, 1994.

[13] M. Pedram and B. Preas. Interconnection length estimation for optimized standard cell layouts. In *Proceedings IEEE International Conf. on Computer-Aided Design*, pages 390–393, 1989.

[14] C. Sechen. Average interconnection length estimation for random and optimized placements. In *Proceedings IEEE International Conf. on Computer-Aided Design*, pages 190–193, 1987.

[15] T. L. Snyder and J. M. Steele. A priori bounds on the euclidean traveling salesman. *SIAM Journal of Computing*, 24(3):665–671, June 1995.

[16] J. M. Steele. Growth rates of euclidean minimal spanning trees with power weighted edges. *Annals of Probability*, 16(4):1767–1787, 1988.

[17] J. M. Steele and T. L. Snyder. Worst-case growth rates of some classical problems of combinatorial optimization. *SIAM Journal of Computing*, 18(2):278–287, 1989.

[18] D. Stroobandt. Improving Donath's technique for estimating the average interconnection length in computer logic. Technical report, Univ. Ghent ELIS Dept., June 1996.