Guiding Global Placement with Wire Density

Kalliopi Tsota, Cheng-Kok Koh and Venkataramanan Balakrishnan
School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907–2035
{ktsoa, chengkok, ragu}@ecn.purdue.edu

Abstract—This paper presents an efficient technique for the estimation of the routed wirelength during global placement using the wire density of the net. The proposed method identifies congested regions of the chip and incorporates the model of the routed wirelength into the objective function in order to effectively alleviate these regions from congestion. The method is integrated in the analytical placement framework and the two-level structure improves the scalability of the placer and speeds up the algorithm. The proposed analytical placer provides the best-so-far average routed wirelength in the IBM version2 benchmark suite.

I. INTRODUCTION

The placement of cells on the chip area is a difficult task in VLSI physical design and the minimization of total routed wirelength is one of the fundamental goals. Moreover, routability is a key measurement of quality. A placed design that results in congested regions on the chip often leads to routing detours and may result in a higher routed wirelength. In the worst case, it may even not allow successful routing.

Several approaches have been used in the past to address the issue of routability. To reduce congestion, in [1] the author modeled the routing supply versus routing demand and incorporated the routing congestion into the objective function of simulated annealing. In [2], [3] the routability of a placement solution was improved using allocation of the white space in congested region during global placement. In [4], [5] the congestion-driven placement was based on the inflation of cells that were located in the congested regions. In [6] the authors made use of linear programming to resolve the conflicts of multiple congested regions. In [7], [8], Rent’s rule was used to estimate the routing demand. In [9] the cell placement problem was formulated as a minimal-cost maximum-flow problem. The algorithm proposed in [10] obtained a model for the wire distribution on the chip, using the wire density of the net.

The paper presents a routability-driven global placement flow. The global placement is performed in a two-level framework and a wirelength estimation technique is used to guide the entire global placement process. At the first level, the clusters are placed on the chip area using an effective global placement algorithm. Then, the clusters are assigned to the updated positions, obtained from the placement algorithm, and every cell of the original netlist is located at the center of its corresponding cluster. At the second level, the cells of the original netlist are placed on the chip area to further reduce overlap. An efficient routed wirelength estimation method is used to determine the wirelength inside the regions of the chip and guide the placement of the clusters at both levels. The main contributions of the paper are: (i) The estimation of wirelength in a region; and (ii) The balancing of the cell density and the wirelength across the placement regions.

In this paper, the wire density of a net is defined as the ratio of the bounding-box perimeter over the bounding-box area of the net. If the bounding box of a net overlaps with a region, it adds to the total wirelength within the region by a factor equal to the product of the wire density of the net and the overlap area between the region and the bounding box of the net. The proposed method makes use of the wire density of a net to estimate the wire congestion at different regions of the chip area during the global placement. The objective is to reduce the wire congestion by modifying the cell density and, at the same time, distributing the wire uniformly inside the regions of the chip. The regions characterized as congested are assigned to lower cell density, whereas the uncongested regions are allowed to have higher cell density.

During the global placement phase, the placement area is divided into an array of uniform rectangular global bins. The wire distribution is independent of the number of wires that cross the boundary of the bin and therefore not affected by the bin structure. At the same time, it is not easy to calculate the wirelength inside the bin, as there are nets that either have multiple pins or do not reside entirely inside the bin. The technique for the estimation of wirelength inside the bin is based on the fact that the wirelength of a multi-pin net at the interior of a bin is determined from the overlap area between the two and the wire density of the net. The wirelength inside the bin is calculated and incorporated into the objective function. Therefore, the wire density of the net guides the placement process as the tool of estimating the wirelength. By penalizing the bins having wirelength higher or lower than the average, the formulation provides a placement of cells that do not contain congested regions and obtains shorter average routed wirelength.

The placer, guided by wire density to estimate wirelength, obtains shorter average routed wirelength compared to the existing placement tools. The improvement is between 2.2% and 14.6% compared to RUDY [10], APlace [11], CHKS+WSA [12], mPL-R+WSA [13], mPL6 [14] and ROOSTER [15], for the circuits of the IBM version2 benchmark suite. The rest of the paper is organized as follows: The two-level framework of the algorithm is covered in Section II. Section III gives the analytical placement formulation. In Section IV, the estimation technique for the routed wirelength is analyzed. Section V includes the implementation details and Section VI shows the experimental results. Section VII concludes the paper.
II. A TWO-LEVEL FRAMEWORK

The placement algorithm presented in the paper is performed in a two-level framework. The first level deals with a netlist of clusters. Initially, the cluster locations are randomly assigned. The algorithm updates their locations to remove cluster overlap and uniformly distribute wirelength across the chip area. At the second level, the cells of the original netlist are assigned to the locations obtained from the first level (each cell is located at the center of its corresponding cluster), and the placement algorithm places the cells on the chip to further reduce cell overlap and uniformly distribute wirelength. Therefore, the wirelength estimation technique is incorporated in the global placement and applied in both levels of the two-level framework. The two-level framework of the placer can be easily extended to a multilevel framework.

III. GLOBAL PLACEMENT

The input to the global placer is a circuit netlist, where the vertices represent the cells and the hyperedges represent the nets in the circuit. The global placement can be applied to the original netlist, at the second level of the two-level framework, or the netlist of clustered cells, at the first level of the two-level framework. As the goal of a global placement is to determine the approximate locations of the cells, it is not necessary to enforce the non-overlap constraints strictly. Instead, a popular approximation as in [12], [18] constitutes an alternative smoothing method to the HPWL, and it is as effective as the LSE approximation in terms of wirelength minimization.

A. Wirelength formulation

The half-perimeter wirelength of a net \( e \in E \) is expressed
\[
HPWL_e = \max\{x_i\} - \min\{x_i\} + \max\{y_i\} - \min\{y_i\}.
\]
(2)

The total HPWL is
\[
HPWL = \sum_{e \in E} HPWL_e.
\]

The CHKS function [12] is used to smooth the two-variable maximum function
\[
CHKS(x_1, x_2) = \sqrt{(x_1 - x_2)^2 + \alpha^2} + x_1 + x_2
\]
where \( \alpha > 0 \) is the smoothing parameter. The multi-variable maximum function, proposed in [18] and used in [12], is obtained by recursive calls to the two-variable maximum function
\[
\max\{x\} = \max\{\max\{x^{(1)}\}, \max\{x^{(2)}\}\},
\]
where \( x^{(1)}, x^{(2)} \) is a disjoint partitioning of \( x \). Then, \( \min\{x\} \) was obtained from \( \max\{x\} \) approximation as
\[
\min\{x\} = -\max\{-x\}.
\]

This paper follows the definition given in [12]: For a convex function \( f : \mathcal{R}^n \rightarrow \mathcal{R} \) define \( \forall 1 \leq i < n-1 \), the function \( f_{i,i+1} : \mathcal{R}^n \rightarrow \mathcal{R} \) by \( f_{i,i+1}(x) = f(x_i, x_{i+1}) \), and \( \forall 1 \leq i < j \leq n \), the function \( f_{i,j} : \mathcal{R}^n \rightarrow \mathcal{R} \) by \( f_{i,j}(x) = x_i \). Moreover, \( \forall 1 \leq i < j \leq n \) and \( j - i + 1 > 2 \), let \( f_{i,j} \) be defined by
\[
f_{i,j}(x) = f(f_{i,k}(x), f_{k+1,j}(x)),
\]
where \( k = \lfloor \frac{i+j}{2} \rfloor \).

Therefore, the multi-variable maximum and minimum functions are smoothed and approximated by \( f_{1,n} \), which, for sake of simplicity, is denoted as \( f \) in the rest of the paper. The CHKS function is convex and differentiable, therefore can be handled by general optimization algorithms. Then, the \( HPWL_e \) (2) is expressed as
\[
HPWL_e = f_{x,e} + f_{y,e},
\]
where \( f_{x,e} \) approximates \( \max_{i \in \mathcal{E}}\{x_i\} - \min_{i \in \mathcal{E}}\{x_i\} \) and is obtained by recursive calls to CHKS for the maximum and minimum x-coordinates of the pins of net \( e \). In the same way, \( f_{y,e} \) approximates \( \max_{i \in \mathcal{E}}\{y_i\} - \min_{i \in \mathcal{E}}\{y_i\} \).
B. Cell density formulation

The cell potential is a rectangular function that is smoothed by the Bell-shaped function as in APlace [11]. Let $w_v, h_v$ be the width, height of cell $v$ and $w_g, h_g$ be the width, height of bin $g$. If cell $v$ is located at a horizontal distance $d_x$ from the center of bin $g$, then the $x$-overlap between $g$ and $v$.

$$
p_{x}(g,v) = \begin{cases} 
1 - a \cdot d_x^2 & \text{if } d_x \in [0, \frac{w_v + 2w_g}{2}], \\
b \cdot (d_x - \frac{w_v + 4w_g}{2})^2 & \text{if } d_x \in \left[\frac{w_v + 2w_g}{2}, \frac{w_v + 4w_g}{2}\right], \\
0 & \text{otherwise.}
\end{cases}
$$

where

$$a = \frac{4}{(w_v + 2w_g)(w_v + 4w_g)}, \quad b = \frac{2}{w_v(w_v + 4w_g)}.$$  

so that the function is continuous when $d_x = \frac{w_v}{2} + w_g$. Similarly, a smoothed cell potential function is defined for the y-direction. The potential function $SD_g$ has the form

$$SD_g(x,y) = \sum_v C_v \cdot p_x(g,v) \cdot p_y(g,v),$$

where $C_v$ is a normalization factor such that

$$A_v = \sum_g C_v \cdot p_x(g,v) \cdot p_y(g,v).$$

In other words, each cell has a total area potential equal to its area.

IV. ESTIMATION OF ROUTED WIRELENGTH

A highly congested region in the placement may result in routing detours around the region and a larger routed wirelength. Congested areas may also degrade the performance of the router and even result in an unroutable design. In order to obtain a global placement that is eventually routable without increasing the runtime, it is important to efficiently evaluate the congestion and identify congested regions of the design. That rules out performing actual routing to determine wirelength and routing congestion, as routing itself is a difficult problem.

During the global placement stage, the path that will be used by the router to route the nets remains unknown and the routing pattern has not yet been obtained. In this paper, the routed-wirelength within a region of the chip is estimated based on the bounding box of each net that overlaps the region. In order to identify the congested regions, it is necessary to determine the wirelength within each one of these regions. For a two-pin net, the router will connect the pins by following the shortest path between them. However, the path the router will follow in the case of a multi-pin net is not as easy to predict. There are possibly many paths that all have the same minimum length for the router to follow. Moreover, the routing paths of the other nets in the region determine the routing path of the net.

Let $SD_g(x,y)$ denote the estimated routed wirelength inside the bin and $avg_{d_g}(x,y)$ the average estimated routed wirelength. Then $D_g(x,y)$ is a function of the estimated wirelength inside bin $g$ and

$$D_g(x,y) = (1 + \gamma \cdot (1 - \frac{SD_g(x,y)}{avg_{d_g}(x,y)})) \cdot D,$$

where $\gamma$ is a user defined constant that determines how much $D_g(x,y)$ deviates from the average $D$. The optimization problem (1) can now be rewritten as

$$\min_{x,y} HPWL(x,y) + \frac{1}{2 \cdot ut} \cdot \sum_g (SD_g(x,y) - D_g(x,y))^2,$$

where the average cell density $D$ used in APlace has now been replaced by $D_g(x,y)$.

From (3), if $SD_g(x,y) = avg_{d_g}, D_g(x,y) = D$. If $SD_g(x,y) < avg_{d_g}$, the estimated routed wirelength inside bin $g$ is less than the average and the cell density allowed inside the bin is higher. Similarly, if $SD_g(x,y) > avg_{d_g}$, the estimated routed wirelength is more than the average and the cell density allowed inside the bin is lower. Having estimated the wirelength inside each global bin, the bins that have high (low) congestion are assigned a lower (higher) cell density.

The wirelength of a net can be approximated by the HPWL. Since there is a strong correlation between the HPWL and the routed wirelength of a net, in the case of a net being entirely inside a global bin, the routed wirelength can be calculated satisfactorily. However, in the case of a net being only partially inside the bin, the routed wirelength inside the bin cannot be determined using the HPWL of the net. There are many paths outside the bin that connect the pins of the net and the router may not follow a path passing through the bin.

An example is illustrated in IV, where the bounding box of net $A$ partially overlaps the global bin $g_1$ (shown in gray) and the router follows a path (shown with a solid line) that is located inside the bin. The bounding box of net $B$ is routed across the global bin $g_2$. For net $B$, the path is located outside the bin. Either one of the two cases may occur and, therefore, the exact wirelength cannot be calculated in advance. The wire density of a net is defined

$$d_{n}(x,y) = \frac{HPWL_{n}}{Area_{n}},$$

where

$$HPWL_{n} = \max_{i \in e} \{x_i\} - \min_{i \in e} \{x_i\} + \max_{i \in e} \{y_i\} - \min_{i \in e} \{y_i\}.$$
is the HPWL of the bounding box of the net \( e \), and

\[
\text{Area}_n = (\max_{i \in e} \{x_i\} - \min_{i \in e} \{x_i\}) \cdot (\max_{i \in e} \{y_i\} - \min_{i \in e} \{y_i\})
\]

is the area of the bounding box of the net.

The algorithm proposed in [10] estimates the routing demand \( D_{\text{req}}(x, y) \) as the superposition of the rectangle functions of all nets, weighted by the wire density of each net. The rectangle function of a net is equal to one for \((x, y)\) inside the bounding box of the net and equal to zero otherwise. The algorithm adapts the demand to the supply at each position; essentially, the algorithm seeks a placement solution where the wire density is uniform across the chip. In this paper, however, the formulation uses the wire density of the net only to determine the wirelength inside a global bin and, therefore, it is the wirelength within the bin that is incorporated into the objective function, not the wire density. In addition, the optimization is independent of the shape of the bounding box of the net and the wirelength is independent of the routing model following the placement.

Taking into consideration the formula for the wire density of a net (4), it is possible to estimate the routed wirelength inside bin \( g \) for every possible configuration. The estimated routed wirelength is

\[
d_{g,e}(x,y) = d_n(x,y) \cdot A_{g,e}(x,y),
\]

where \( d_n(x,y) \) is the wire density of net \( e \) and \( A_{g,e}(x,y) \) is the overlap area between bin \( g \) and the bounding box of net \( e \). The total wirelength inside bin \( g \) is the sum of the wirelength of each net which overlaps the bin either fully or partially. Thus

\[
Sd_g(x,y) = \sum_e d_{g,e}(x,y).
\]

### A. Function smoothing

The estimation of the wirelength inside the bin using the wire density of a net, includes determining the overlap area \( A_{g,e}(x,y) \). As shown in IV-A, there are different ways in which the bounding box of a net and the bin overlap. For example, when the bounding box of the net is located partially inside and on the right side of the bin as illustrated in IV-A, the overlapped area between the specified net and the bin is calculated as

\[
A_{g,e} = (\text{horiz overlap}) \cdot (\text{vertical overlap}),
\]

where

\[
\text{horiz overlap} = \text{CHKS}(x_{\text{right}}, \min_{i \in e} x_i) + \text{CHKS}(x_{\text{right}}, -\min_{i \in e} x_i),
\]

\[
\text{vertical overlap} = \text{CHKS}(\max_{i \in e} y_i, \min_{i \in e} y_i) + \text{CHKS}(\max_{i \in e} -y_i, \min_{i \in e} -y_i)
\]

and \((x_{\text{left}}, y_{\text{bot}}), (x_{\text{right}}, y_{\text{top}})\) are the coordinates of the bin.

To smooth the function of the estimated routed wirelength inside bin \( g \), \( Sd_g(x,y) \), we use the recursive CHKS function.

Following the analysis performed in Section III, let \( f_{x,e} \) be the smoothed function that approximates \( \max_{i \in e} \{x_i\} - \min_{i \in e} \{x_i\} \) and \( f_{y,e} \) be the smoothed function that approximates \( \max_{i \in e} \{x_i\} - \min_{i \in e} \{x_i\} \). The wire density of a net is written as

\[
d_{n,e}(x,y) = f_{x,e}(x,y) + f_{y,e}(x,y).
\]

The total wirelength within bin \( g \) is

\[
Sd_g(x,y) = \sum_e (d_{n,e}(x,y) \cdot A_{g,e}(x,y)).
\]

The average bin wirelength is

\[
\text{avg}_{g}(x,y) = \text{avg}_{g}(Sd_g(x,y)).
\]

Therefore

\[
D_g(x,y) = (1 + \gamma \cdot (1 - \frac{\text{avg}_{g}(Sd_g(x,y))}{\text{avg}_{g}d(x,y)})) \cdot D,
\]

where

\[
Sd_g(x,y) = \sum_e (f_{x,e}(x,y) + f_{y,e}(x,y) \cdot A_{g,e}(x,y)).
\]

Also,

\[
\text{avg}_{g}(x,y) = \text{avg}_{g}\left(\sum_e (f_{x,e}(x,y) + f_{y,e}(x,y) \cdot A_{g,e}(x,y))\right).
\]

Most Newton-like optimization solvers require evaluations of the function value \( f(x) \) and its gradient \( g(x) = \left[\frac{\partial f_1(x)}{\partial x_1}, \frac{\partial f_1(x)}{\partial x_2}, \ldots, \frac{\partial f_1(x)}{\partial x_n}\right]^T \). To calculate the gradient of \( f(x) \)

\[
\frac{\partial f_{i,j}(x)}{\partial x_l} = \frac{\partial f_{i,k}(x)}{\partial x_l} \cdot \frac{\partial f_{k,j}(x)}{\partial x_l} + \frac{\partial f_{i,j}(x)}{\partial x_{k+1,j}} \cdot \frac{\partial f_{k+1,j}(x)}{\partial x_l},
\]

where

\[
k = \left\lfloor \frac{i+j}{2} \right\rfloor \text{ and } 1 \leq l \leq j. \]

For \( i \leq l \leq k \) and CHKS smoothing,

\[
\frac{\partial f_{i,j}(x)}{\partial x_l} = \frac{1}{2} \cdot \frac{\partial f_{i,k}(x)}{\partial x_l} + \frac{1}{2} \cdot \sqrt{(f_{i,k}(x) - f_{k+1,j}(x))^2 + \alpha^2} \cdot \frac{\partial f_{i,k}(x)}{\partial x_l},
\]
To solve the unconstrained optimization problem, one needs to calculate the gradient of $D_g$, $d_n$. The gradient of $D_g$ is calculated as
\[
\frac{\partial D_g(x,y)}{\partial x_l} = -\gamma \cdot D \cdot \frac{\partial SD_x(x,y)}{\partial x_l} \cdot \frac{\partial max(x,y)}{\partial x_l},
\]
\[
\frac{\partial D_g(x,y)}{\partial y_l} = -\gamma \cdot D \cdot \frac{\partial SD_y(x,y)}{\partial y_l} \cdot \frac{\partial max(x,y)}{\partial y_l}.
\]
The gradient of $d_n(x,y)$ is calculated as
\[
\frac{\partial}{\partial x_l} \left( f_{x,e}(x,y) \cdot f_{y,e}(x,y) \right) = -\frac{\partial f_{x,e}(x,y)}{\partial x_l} \cdot f_{y,e}(x,y),
\]
\[
\frac{\partial}{\partial y_l} \left( f_{x,e}(x,y) \cdot f_{y,e}(x,y) \right) = -\frac{\partial f_{y,e}(x,y)}{\partial y_l} \cdot f_{x,e}(x,y).
\]

V. IMPLEMENTATION

The placement algorithm is applied in a two-level framework as explained in Section II. The global placer places the clusters at the first level and the cells at the second level. For this work, the clusters were obtained from the $K$-way partitioning of the original circuit netlist using the publicly available software hMetis [19]–[21]. The hMetis algorithm performs $K$-way partitioning of the circuit netlist with the objective of reducing the number of edgecuts among the partitions. The hybrid first choice scheme in hMetis is used for the grouping of the cells. The partitioning of the netlist is obtained after $N = 5$ runs of the hMetis algorithm and the partition with the smallest edgecut is selected. The original netlist is clustered to a number of clusters such that the number of cells inside each cluster is between 1% and 5% of the number of cells in the netlist. Each cluster is shaped as a square, with its area being equal to the sum of the areas of the cells that belong to the cluster. The cells are confined inside the cluster area of their corresponding cluster and the input to the placement algorithm is the partitioning of the circuit netlist.

At both levels of the framework, the routed wirelength estimation technique guides the placement process. The unconstrained optimization problem is solved using L-BFGS-B, a publicly available quasi-Newton solver with boundary constraints, as described in [22]. The implementation of the solver is in the default mode. The two-level framework includes a multi-grid structure similar to the one in [12] and the same stopping criterion $\sqrt{\sum (SD_y - D_y)^2}$ for each bin. The initial weight of the density penalty is
\[
u = \frac{1}{2} \sum_{x_i,y_j} \sum_x |SD_x - D_x| \cdot \left( \left| \frac{\partial SD_x}{\partial x_l} \right| + \left| \frac{\partial SD_y}{\partial y_l} \right| \right)
\]
\[
+ \sum_{x_i,y_j} \left( \left| \frac{\partial WL}{\partial x_l} \right| + \left| \frac{\partial WL}{\partial y_l} \right| \right) \sum_x |SD_x - D_x| \cdot \left( \left| \frac{\partial SD_x}{\partial x_l} \right| + \left| \frac{\partial SD_y}{\partial y_l} \right| \right)
\]
\[
+ \sum_{x_i,y_j} \left( \left| \frac{\partial WL}{\partial x_l} \right| + \left| \frac{\partial WL}{\partial y_l} \right| \right) \sum_x \left( \left| \frac{\partial SD_x}{\partial x_l} \right| + \left| \frac{\partial SD_y}{\partial y_l} \right| \right)
\]
\[
\sum_{x_i,y_j} \left( \left| \frac{\partial WL}{\partial x_l} \right| + \left| \frac{\partial WL}{\partial y_l} \right| \right),
\]
(6)

After the initialization, $\nu$ is halved in every iteration. At the end of the global placement, the detailed placer mPL-R+WSA [13] is used to legalize the placement.

VI. EXPERIMENTAL RESULTS

The results are based on sixteen IBM version 2 easy and hard benchmarks and show the impact of routability-driven standard-cell placement on the routed wirelength. All experiments were performed on a 3GHz Pentium 4 CPU with 3GHz memory. Table 1 includes the routed wirelengths, as well as the CPU times for placement compared to mPL-R+WSA [13], mPL6 [14], CHKS+WSA [12], RUDY [10], ROOSTER [15] and APlace [11]. For each placement tool, the first column, labeled “r-WL”, shows the routed wirelength on each benchmark after routing the placed design using Cadence WRoute. The second column, labeled “p-CPU”, shows the CPU times for placement.

The placement solutions of the proposed approach were routed with CADENCE WRoute 5.3 in default configuration. The routed wirelengths of circuits placed with mPL-R+WSA, mPL6 and CHKS+WSA are as reported in [12] (routed with CADENCE WRoute 5.3) and the routed wirelengths of circuits placed with RUDY, ROOSTER and APlace (routed with CADENCE WRoute 2.3.32) are as reported in [10]. The placement solutions obtained from the proposed approach result in reductions in the total routed wirelength by 14.56%, 5.48%, 5.55%, 2.17%, 9.68% and 8.53%, compared to mPL-R+WSA, mPL6, CHKS+WSA, RUDY, ROOSTER and APlace respectively. Compared to RUDY, our algorithm obtained shorter routed wirelength in 12 out of the 16 benchmarks. In Table 1, the row labeled “r-WL%improv.” shows the improvements in the total routed wirelength obtained from our placer over those of other placement tools.

The placement time of every placement tool is defined as the total CPU time to perform global/detailed placement and legalization. The placement CPU times for mPL-R+WSA, mPL6, CHKS+WSA and APlace are obtained from our runs of the respective executables. For our placement algorithm, the placement CPU times also include the time to run hMetis. The proposed algorithm results in reductions in the placement CPU time by 52.13%, 8.29%, 45.61% and 63.50% respectively. The CPU times for ROOSTER were not reported in [10] and are, therefore, not included in Table 1. The CPU time for RUDY is obtained on a 2.2 GHz AMD Athlon Opteron 248 machine in [10]; therefore, no direct comparison with the runtime of RUDY and ROOSTER can be made at this point.

VII. CONCLUSION

The paper proposes a global placement flow guided by an efficient routed wirelength estimation method. The overlap between the bounding box of a net and a global bin, weighted by the wire density of the net, estimates the wirelength within the corresponding global bin and overcomes the difficulty of determining the wirelength inside a bin for the case of multipin nets. At the same time, the two-level framework reduces the total runtime of the placer. The result is an algorithm that obtains the best average routed wirelengths in the IBM version2 benchmark suite.
<table>
<thead>
<tr>
<th>Proposed</th>
<th>TABLE I</th>
<th>ROUTABILITY RESULTS ON IBM VS.2 STANDARD CELL PLACEMENT BENCHMARKS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**ACKNOWLEDGMENT**

This research was supported in part by SRC (Task ID 1822.001).

**REFERENCES**


