An Effective Routability-driven Placer for Mixed-size Circuit Designs

Shuai Li
School of Electrical and Computer Engineering, Purdue University, li263@purdue.edu

Cheng-Kok Koh
School of Electrical and Computer Engineering, Purdue University, chengkok@purdue.edu

Follow this and additional works at: http://docs.lib.purdue.edu/ecetr

http://docs.lib.purdue.edu/ecetr/446

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries. Please contact epubs@purdue.edu for additional information.
An Effective Routability-driven Placer for Mixed-size Circuit Designs

Shuai Li
Cheng-Kok Koh

TR-ECE-13-06
April 18, 2013

Purdue University
School of Electrical and Computer Engineering
465 Northwestern Avenue
West Lafayette, IN 47907-1285
ABSTRACT

We propose a routability-driven analytical placer that aims at distributing pins evenly. This is accomplished by including a group of pin density constraints in its mathematical formulation. Moreover, for mixed-size circuits, we adopt a scaled smoothing method to cope with fixed macro blocks. As a result, we have fewer cells overlapping with fixed blocks after global placement, implying that the optimization of the global placement solution is more accurate and that the global placement solution resembles a legal solution more. Routing solutions obtained by a commercial router show that for most benchmark circuits, better routing results can be achieved on the placement results generated by our pin density oriented placer.

Keywords

routability-driven placement, pin density, mixed-size circuit

1. INTRODUCTION

The quality of placement results has been greatly improved in the past decade, and many effective analytical placers such as mPL [1], Aplace [2], NTUplace [3], etc., have been developed. However, using total half parameter wirelength (HPWL) as the main objective, these placers may generate routing congested regions. Indeed, routability is one of the most critical issues in modern circuit design [4]. Placement, the immediate step before routing, should play an important role in alleviating routing congestion.

With the promotion of the ISPD11 and DAC12 routability-driven placement contests [4] [5] (and earlier ISPD contests), a number of routability-driven placers, including SimpleR [6], Ripple [7], NTUplace4 [8], etc., have been proposed in recent years. Generally, most existing routability-driven placers resort to placement refinement for the alleviation of possible routing congestions. After an initial solution is generated by a traditional placement algorithm, congested regions are located with the guidance of global routing overflows and/or pin penalty. The placement result is then refined accordingly by means of techniques such as white space allocation [9], cell bloating [10] [6], net-based movement [7], etc. This process usually is iterative and continues until no significant improvement can be made.

As mentioned, most existing routability-driven placers usually use global routing overflow as the main congestion estimation factor. The shortcomings of such a metric is that it ignores all the local nets (i.e., nets connecting only pins inside a cell and thus contributing nothing to global routing overflow), which may cause detrimental local congestions, too. Thus, the congestion information provided by global routing may not be exact. If guided by global routing information that does not account for local congestion information, placers are likely to generate placement results with fewer global routing overflows but poor routability in reality. Such a mismatch between the quality of a global routing solution and the quality of a detailed routing solution has been highlighted in [11–14].

However, it is infeasible to include in a placer a detailed routing step to guide the placement optimization process. In other words, it remains a big question mark how to generate placement results that can be (detailed) routed easily.

In this paper, we present a new analytical placer for routability-driven placement of mixed-size circuits. The goal is to achieve routability without invoking a global router or a detailed router. Two main contributions we make are:

1. Pin density oriented (or more precisely, pin count oriented) formulation for analytical placer.

The importance of pin density in guiding routability-driven placement has been addressed in many previous works [10] [8]. Instead of spreading cells, our placer just aims at even pin distribution across the chip. The key idea here is that a global placement solution that has even pin distribution most likely would also have cells sufficiently spread out.

What is more important is that the pin count within a region provides a good estimate of the routing requirement within that region. Every pin is involved in some net. The pin count can simultaneously account for global nets and local nets in that region, i.e., it can estimate both global routing demand and local routing demand. A global placer guided by pin density thus may be more effective in eliminating routing congestions.

Instead of resorting to placement refinement methods, we accomplish even pin distribution by applying a new formulation for analytical placer. The formulation is defined by including a group of effective pin density constraints.

2. Scaled smoothing method to cope with fixed macro blocks.

Fixed blocks may have many adverse effects on placement and routing. They are obstacles that affect cells’ movement in global placement. In particular, if there are many overlaps between cells and fixed blocks in a global placement result, legalizing these cells will greatly perturb the even distribution of pins. For a more accurate and effective optimization of the global placement solution for routability, the global placement solutions must resemble a legal solution.

To solve this problem of significant overlaps between cells
and macro blocks, we adopt a scaled smoothing method. By smoothing and properly scaling up the allocation of fixed blocks, cells are kept away from macro blocks in global placement such that cell displacement in the legalization step is reduced.

Experimental results show that our pin density oriented placer is effective in eliminating congested regions with high pin density, and even pin distribution turns out to be greatly helpful in improving the routability of circuits. We have obtained the routing solutions for our placer’s results on ISPD11 benchmark circuits by using a commercial router, and a comparison is made with the results generated by Ripple [7] and NTUplace [8] in ISPD11 contest. For seven out of eight ISPD11 benchmark circuits, our placer’s results can be routed in a shorter time and the routing solutions have fewer routing violations. Meanwhile, on average, the routing wirelength of our placer’s results is 8.84% and 5.42% shorter compared to Ripple and NTUplace, respectively. Moreover, the number of vertical vias is also reduced by over 5%.

The rest of the paper is organized as follows. Section 2 introduces the pin density oriented formulation. Section 3 discusses the scaled smoothing method, as well as the implementation details of our placer. Section 4 briefly presents the legalization and detailed placement techniques used in our placer. Section 5 demonstrates experiment results: Section 5.1 makes a comparison of the placement results generated by pin density oriented placer and cell density oriented placer; Section 5.2 shows the routing solutions of different placers’ results. Section 6 concludes the paper.

2. FORMULATION

As in other analytical placers, the problem of global placement is formulated as a constrained optimization problem in our analytical placer. The objective is to minimize wirelength estimated with the summation of HPWL for all the nets. Also, the whole circuit is partitioned into uniform nonoverlapping rectangle bins, and a constraint is defined for each bin so that not too many pins would be placed in it.

The definitions used in the formulation are as follows:

- \( \mathcal{N} \) set of all nets;
- \( \mathcal{C} \) set of all cells;
- \( \mathcal{B} \) set of all uniform rectangle bins;
- \( P_b \) area occupied by fixed macro blocks in bin \( b \in \mathcal{B} \);
- \( S_b \) available area for placing cells in \( b \in \mathcal{B} \);
- \( P_{bc} \) overlapping portion of cell \( c \in \mathcal{C} \) in \( b \in \mathcal{B} \);
- \( d_c \) number of pins on cell \( c \in \mathcal{C} \);
- \( w_c, h_c \) width, height of cell \( c \in \mathcal{C} \);
- \( w_b, h_b \) width, height of bin \( b \in \mathcal{B} \);
- \( (x_c, y_c) \) coordinates of the center of cell \( c \in \mathcal{C} \);
- \( (x_b, y_b) \) coordinates of the center of bin \( b \in \mathcal{B} \);
- \( t_{den} \) target placement density;
- \( \text{avg}_{pd} \) average pin density of all cells

With the definitions above, we formulate global placement problem as follows:

\[
\begin{align*}
\min_{x, y} & \quad \text{HPWL}(\vec{x}, \vec{y}) \\
\text{s.t.} & \quad \sum_{c \in \mathcal{C}} r_c P_{bc}(x_c, y_c) \leq \text{avg}_{pd} S_b, \quad \forall b \in \mathcal{B}. 
\end{align*}
\]

All the non-differentiable functions in the formulation must be estimated with smooth functions so that the minimization problem could be solved with Newton-like methods. The HPWL function is defined below, with non-differentiable \( \max \) and \( \min \) functions:

\[
\text{HPWL} = \sum_{n \in \mathcal{N}} (\max_{c \in \mathcal{C}} (x_n - x_c) + \max_{c \in \mathcal{C}} (y_n - y_c)).
\]

Instead of the widely-used log-sum-exp function [15], we use a different smoothing function to smooth HPWL function:

\[
C_{CHKS}(x_1, x_2) = (\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2})/2.
\]

The two-variable function above can be used as a smooth function for \( \max(x_1, x_2) \), and its smoothness can be tuned with factor \( \alpha \). In a nested way, multiple-variable \( \max, \min \) functions can also be smoothed with it. Details are discussed in [16] about the properties of CHKS function.

The left hand side of the pin density constraint (2) gives the potential of the number of pins that are placed in bin \( b \). The potential function \( P_{bc}(x_c, y_c) = h_{bc}(x_c)v_{bc}(y_c) \), which is defined by the amount of overlap between cell \( c \) and bin \( b \) is also non-differentiable. Here, \( h_{bc} \) and \( v_{bc} \) are the overlapping portion of cell \( c \) and bin \( b \) in horizontal and vertical dimensions, respectively. \( h_{bc}(x_c) \) can be smoothed with \( p(x-x_c) \), where \( p(x) \) is a continuous differentiable “bell-shape” function defined below [2]:

\[
p(x) = \begin{cases} 
1 - x^2/(2w_b^2) & \text{if } 0 \leq |x| \leq w_b, \\
(|x| - 2w_b)^2/2w_b^2 & \text{if } w_b \leq |x| \leq 2w_b, \\
0 & \text{if } |x| \geq 2w_b. 
\end{cases}
\]

\( v_{bc} \) can be smoothed in the same way, too. Besides, \( r_c \) in (2) is a normalization factor such that \( \sum_c (r_c P_{bc}) = d_c \), i.e. cell \( c \) contributes a total potential that is equal to the number of pins on it.

The right hand side of (2) denotes the number of pins that bin \( b \) is supposed to accommodate at most. It is defined as the product of the average pin density of all the cells, \( \text{avg}_{pd} \), and the available area in bin \( b \), \( S_b \):
blocks in each bin, \( F \) technique proposed in [3]. The smoothed allocation of fixed blocks, our placer adopts the Gaussian smoothing (3) is always the same. The contour of the normalized \( F \) shows the contour of \( F' \), which is smoother than \( F \). Also, the smoothness can be tuned by \( \sigma \) in the Gaussian function. Larger \( \sigma \) leads to smoother \( F' \). Conversely, when \( \sigma \) is sufficiently small, \( F' \) is nearly equivalent with \( F \), as shown in Fig. 1(d). As in [3], we make use of Gaussian smoothing by replacing \( F \) with \( F' \) in (3). At the beginning, \( F' \) is calculated with a large \( \sigma \) equal to the half width of the chip. Then \( \sigma \) decreases gradually in the optimization process.

With the application of Gaussian smoothing, cells are spread more smoothly during optimization. However, the other challenge caused by the presence of fixed blocks is not totally overcome. Many cells may still end up overlapping with blocks, as shown in Fig. 2(a). One of the reasons is that after smoothing, many fully occupied bins (i.e., bins with \( F_b = w_bh_b \)) have temporary empty spaces as \( F'_b < w_bh_b \). Thus, many cells may be attracted to the top of blocks, and then trapped after \( \sigma \) is decreased.

To solve this problem, in our implementation, we choose to properly scale up \( F'_b \) during optimization, so that most fully occupied bins have no empty space available for cells. To maintain the smoothness of fixed-block allocation, we scale up \( F'_b \) of all bins with the same factor. After Gaussian smoothing, most empty bins (i.e., bins with \( F_b = 0 \)) have \( F'_b \) close to 0, except those around fixed blocks. Thus, the scale-up has little influence on most empty bins, and the placement of cells in block-free areas will not be affected significantly. Meanwhile, the increased \( F'_b \) of bins around macro blocks may also be positive in helping cells away from future routing obstacles.

On the other hand, the scale-up factor cannot be set too large to influence too many bins around blocks. Also, the scaled smoothing does no good to cell spreading. Therefore, our global placement algorithm is implemented as a two-stage work:

**Stage 1:** Cell spreading. In this stage, we use large \( \sigma \) and do not scale up \( F'_b \)s. Instead, we also apply another smoothing technique, called level smoothing [3]. This stage continues until cells are well spread across the chip.

**Stage 2:** Relocating cells overlapping with blocks. This stage is composed with several iterations. In the first iteration, \( \sigma \) is set to be large enough so that no “flat” top would form on large blocks. Then, based on the Gaussian smoothing results, a scale factor is set such that \( \% F_b \) fully occupied bins have no empty space. With the scaled-up \( F'_b \) optimization problem (1)-(2) is solved. Then, \( \sigma \) is halved and a new iteration starts. This iterative process continues until \( \sigma \) is small enough. Note that after each iteration, \( \beta \) is increased, and at the end, it is close to 100.

An example is given in Fig. 2 to illustrate the effect of scaled smoothing. Both global placement results in it are generated with the two-stage global placement algorithm. The only difference is that the scale-up factor is always set to be 1 for (a). Apparently, much fewer cells end up overlapping with blocks in (b). Only 0.08% of cells still overlaps with blocks in (b), whereas in (a), 7.71% of cells overlaps with blocks. As a consequence, the average displacement of each cell after legalization is only 22.9 for (b), nearly half of that in (a). The percentages of cells overlapping with macro blocks in our global placement results of eight ISPD11
Figure 2: Illustration of the effect of scaled smoothing on superblue4. Plots of global placement results (a) Without scaled smoothing, cells on blocks 7.71%, average displacement 45.6; (b) With scaled smoothing, cells on blocks 0.08%, average displacement 22.9.

Table 1: Percentages of cells overlapping with macros blocks in our global placement results of ISPD11 benchmark circuits

<table>
<thead>
<tr>
<th>sb1</th>
<th>sb2</th>
<th>sb4</th>
<th>sb5</th>
<th>sb10</th>
<th>sb12</th>
<th>sb15</th>
<th>sb18</th>
</tr>
</thead>
<tbody>
<tr>
<td>%</td>
<td>0.38</td>
<td>0.13</td>
<td>0.08</td>
<td>0.29</td>
<td>0.69</td>
<td>0.03</td>
<td>0.43</td>
</tr>
</tbody>
</table>

benchmarks are listed in Table 1.

4. LEGALIZATION & DETAILED PLACEMENT

Our legalizer is similar to the Abacus algorithm [18]. From left to right, cells are placed into legal positions one by one. Each cell is placed in a row such that inserting the cell in the row leads to the smallest displacement. Instead of packing cells tightly from left to right in each row, in our implementation, we place cells as close to their original position as possible, so that the perturbance to pin distribution is reduced. We apply the sliding window technique [9] for detailed placement to further optimize the placement.

5. EXPERIMENTAL RESULTS

By applying the two-stage global placement algorithm discussed in Section 3, followed by the legalizer & detailed placer introduced in Section 4, we have generated placement results for the eight ISPD11 benchmark circuits [19]. Target density $t_{den}$ is set to be 0.8 for all benchmark circuits.

All placement results generated are evaluated by the commercial router Wroute [20] (version 3.1.61) for their routability quality. To do that, we used the translator developed in [21]. Placement results are translated into LEF/DEF files, which are then fed into Wroute for global and detailed routing under some 28nm design rules (also specified in [21]). We ran Wroute for two iterations to obtain the final detailed routing solutions. In the first iteration, Wroute is run in the default routing mode. Wroute may report that some nets as unroutable after global routing, and ignores these unroutable nets in detail routing. In the second iteration, Wroute is run in the post routing mode, which tries to route all nets and repairs routing violations. In each iteration, the detailed routing continues until all violations are repaired, no further improvement can be made, or a time limit of 24 hours is reached.

5.1 Comparison of formulations

We compare the cell density oriented formulation (1)(4) and the pin density oriented formulation (1)(2) using ISPD11 benchmark circuit superblue4. The placement result in Fig. 3(a) is generated with cell density oriented formulation, whereas the placement result in Fig. 3(b) is generated with pin density oriented formulation. The same placement steps introduced in Sections 2–4 are used in the generation of both results.

Detailed routing solutions of both results differ a lot in terms of the number of routing violations. (b) has only 201 violations whereas (a) has over 8800 violations, most of which appear in the red box marked in the figure.

Evidently, cells in (a) are more evenly distributed than (b). However, in reality, pins are not evenly distributed in it. The pin distribution in the red box highlighted in (a), which has 100×100 bins, is given in Fig. 3(c). There is one large congested region with high pin density near the bottom of the box. In fact, most violations in the routing solution of (a) occur in this congested region.

In contrast, as shown in Fig. 3(d), the pin count contour of the same highlighted area in (a) reveals that pins are more evenly distributed. There are also some small congested regions. However, because they are scattered, the negative effect on routing is not significant.

5.2 Routing solutions

Now, we present a comparison with Ripple [7] and NTU-place [8]. Ripple is chosen because it won the 2011 ISPD
Table 2: Comparison of detailed routing solutions for the placement results generated by our placer, Ripple [7], and NTUplace [8].

<table>
<thead>
<tr>
<th>Our placer</th>
<th>Ripple</th>
<th>NTUplace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vio (e8)</td>
<td>WL(e8)</td>
<td>Via(e6)</td>
</tr>
<tr>
<td>sb4</td>
<td>201</td>
<td>2.686</td>
</tr>
<tr>
<td>Imp.</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

Table 3: Comparison of global routing metrics of the placement results generated by different placers.

<table>
<thead>
<tr>
<th>Our placer</th>
<th>Ripple</th>
<th>NTUplace</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUN</td>
<td>OC(%)</td>
<td>NUN</td>
</tr>
<tr>
<td>sb1</td>
<td>32</td>
<td>5.29</td>
</tr>
<tr>
<td>sb2</td>
<td>199</td>
<td>3.57</td>
</tr>
<tr>
<td>sb4</td>
<td>0</td>
<td>6.00</td>
</tr>
<tr>
<td>sb5</td>
<td>661</td>
<td>3.96</td>
</tr>
<tr>
<td>sb10</td>
<td>389</td>
<td>4.64</td>
</tr>
<tr>
<td>sb12</td>
<td>0</td>
<td>15.34</td>
</tr>
<tr>
<td>sb15</td>
<td>3</td>
<td>12.37</td>
</tr>
<tr>
<td>sb18</td>
<td>524</td>
<td>8.72</td>
</tr>
<tr>
<td>Imp.</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

for seven out of eight benchmark circuits, the routing run-time of our placer’s results is shorter, and the routing solutions have fewer violations. Also, the bottom row in Table 2 shows the average improvement our placement results achieve over others in routing wirelength and the number of vias. On average, the routing wirelength of our placement results is 8.85%, 5.15% smaller than those of Ripple and NTUplace, respectively, whereas the number of vertical vias is also reduced by over 5%. Note that for each benchmark, the smallest number of violations and the shortest wirelength are shown in bold in table.

For six out of eight benchmark circuits, our placer’s results can be routed with only a few routing violations, while superblue5 and superblue18 are two exceptions. However, by applying different parameters in scaled smoothing according to the features of macro blocks on both circuits, we can also get better results for superblue5 and superblue18 with only 520, 3508 routing violations, respectively.

Table 3 also shows a comparison of global routing metrics for all placement results. “NUN” gives the number of unroutable nets that Wroute reports after global routing in the first iteration. “OC” gives the percentage of overcapacity gcells after both iterations. In terms of both metrics, the placement results of our placer also outperform others for most benchmarks.

So far our analytical placer does not adopt the multilevel placement technique [1] [3]. As a result, for the eight benchmark circuits, our placer has longer placement run-times, which are listed in the “Tp” column (hour:minute) in Table 2. While the placement run-times may look long, the total place-and-route run-time may be shorter compared with other placers, while the routability quality of the results improves, too. One of our future work is to increase the scalability of our placer by adopting the multilevel placement technique.

### 6. CONCLUSION

A new analytical placer applicable for mixed-size circuits is proposed in this paper. To help alleviate possible routing congestions, our placer makes use of a new analytical placement formulation defined with pin density constraints. In addition, our placer adopts a scaled smoothing technique to avoid cells overlapping with macro blocks in global placement results, due to which the perturbance to pin distribution in legalization stage can be reduced.

Experiment results on ISPD11 benchmark circuits show that our pin density oriented placer manages to generate placement results with good routability. Compared with the placement results of other routability-driven placers, in most cases, detailed routing solutions with fewer violations can be generated for our placer’s results in a shorter time. On average, the routing solutions also have smaller routing wirelength and fewer vertical vias.

### 7. REFERENCES

