國立交通大學

電機學院與資訊學院 電子與光電學程

碩 士 論 文

運用增廣拉格朗日方法的多階層式混合尺寸置放器

A Multi-level Mixed-size Placer Using Augmented Lagrangian Method

研 究 生：周幼奇

指導教授：周景揚 博士

陳宏明 博士

中 華 民 國 九 十 五 年 九 月

運用增廣拉格朗日方法的多階層式混合尺寸置放器

A Multi-level Mixed-size Placer Using Augmented Lagrangian Method

研 究 生：周幼奇 Student：Yu-Chi Chou

指導教授：周景揚 Advisor：Jing-Yang Jou

陳宏明 Hung-Ming Chen

國 立 交 通 大 學

電機學院與資訊學院專班 電子與光電學程

碩 士 論 文

A Thesis

Submitted to Degree Program of Electrical Engineering and Computer Science

College of Electrical Engineering and Computer Science

National Chiao Tung University

in Partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Electronics and Electro-Optical Engineering

September 2006

Hsinchu, Taiwan, Republic of China

中華民國九十五年九月

運 用 增 廣 拉 格 朗 日 方 法 的 多 階 層 式 混 合 尺 寸 置 放 器

學生：周紡奇　　　　　　　　　　　　　　　　　指導教授：周景揚 博士

陳宏明 博士

國立交通大學 電機學院與資訊學院 電子與光電學程（研究所）碩士班

摘　　　要

由於矽智財的普遍使用以及越來越多的不同元件整合成系統單晶片，混合尺寸的電路元件置放成為實體電路設計中關鍵的一環。然而在處理置放問題時，對於大電路單元和標準電路元，兩者演算法在本質上差異甚大，因此要在一套流程中一起完成混合尺寸的置放是十分具有挑戰性的問題。在這篇論文中，我們將原本的置放問題轉化為一套新的最佳化模型，並以數學解析的方法配合多階層式架構求解。由實驗結果可知此套模型確實可用於實作全域式置放器，以應用增廣拉格朗日方法做為非線性最佳化求解的核心，能夠對整體電路繞線長度得到良好的成果。

A Multi-level Mixed-size Placer Using Augmented Lagrangian Method

Student：Yu-Chi Chou

Advisors：Dr. Jing-Yang Jou
Dr. Hung-Ming Chen

Degree Program of Electrical Engineering and Computer Science
National Chiao Tung University

ABSTRACT

Due to the trends of IP re-use and the SOC integration, mixed-size designs are very common now, and the quality of mixed-size placement becomes a critical step in the VLSI physical design. However, because the algorithms of macro placement and standard-cell placement are fundamentally distinct, placing the mixed-size design in a single flow is actually a challenging problem. In this thesis, we formulate the general placement problem as a nonlinear constrained optimization problem and solve it by the analytical approach incorporating with a multi-level scheme. The experimental results clearly show that our model can be employed as a global placer. By applying the augmented Lagrangian method to perform nonlinear programming, the result of the total half-perimeter wire length is comparable to current state-of-the-art placers.

ACKNOWLEDGEMENTS

I would like to express my sincerest appreciation to my advisors, Dr. Jing-Yang Jou and Dr. Hung-Ming Chen, for their guidance of this thesis. On every review Dr. Jou always highlighted the critical issues I never think of. Dr. Chen gave me consistent help and invaluable suggestion during my program development. Without their help I could never get this work done. I also would like to thank all the friends and colleagues who ever helped me during my thesis writing. At last I wish to thank my wife and my mother for the endless support and encouragement throughout my study years.

Contents

List of Figures

List of Tables

Chapter 1

Overview

1.1 Introduction

The complexity of circuit design continues to increase as the deep sub-micron IC technology keeps scaling down the feature size. This trend makes IP reuse become a necessary strategy to tame the design complexity and ensure time-to-market. Current SOC designs usually contain hundreds of thousands of standard cells, mixed with a number of IP, analog blocks, embedded memories, and pre-designed legacy blocks. To place so many different cells simultaneously is actually a great challenges. An obvious issue is the design scale. The placement algorithm must be very efficient so that it can handle huge amount of placement instances within an acceptable runtime and memory size. In addition, the variation in cell dimensions introduces significant discontinuity in the solution space of the placement.

As indicated in [1], the cell size of the hard blocks may ranges from 1x to 10000x or more, comparing with that of a standard cell. That is the root cause why traditional standard-cell placers usually either fail to process these mixed-size designs or produce results with unsatisfied quality. Although floorplanners are suitable to place the cells with arbitrary sizes, the design scale makes the use of traditional floorplanners impractical. In existing commercial tools, placing the hard macros of a modern SOC design still requires helps from human engineers and is a time-consuming task during the design floorplanning phase. Since placement plays a critical role in determining the circuit performance and layout resources, an efficient and effective algorithm which focuses on such mixed-size large-scale design would be very helpful to SOC developments. This thesis presents a new algorithm to tackle the large-scale mixed-size placement problem.

1.2 Our Contribution

- A new formulation is proposed to model the placement problem. Our formulation focuses on wire length minimization and ensures an even cell distribution. Because of its universal form, the formulation can be applied on general designs and is suitable for mixed-size problem.

- We introduce the augmented Lagrangian method to implement the optimization solver and describe the programming details of the whole multi-level engine.

- The experimental results show that our algorithm is comparable to current state-of-the-art placers.

- A novel floorplanning technique is studied as the macro legalization.

1.3 Organization

The remaining part of this paper is organized as follows. Chapter 2 describes the background of the placement problem and our problem formulation. The details of our method is discussed in Chapter 3, and the experimental results are summarized in Chapter 4. We conclude the paper in Chapter 5.

Chapter 2

Preliminaries

2.1 Placer Classification

Cell placement is the process to arrange the circuit components onto a layout surface. A placer that performs cell placement is usualy required to minimize the interconnects of the whole design as well as other objectives, depending on the design requirements. According to the design style, the placement problem can be classified as block placement or macro placement, standard-cell placement or row-based placement, and mixed-size placement.

Generally, block placement focuses on full-custom design, in which the circuit components can be arbitrary sizes and shapes. Due to its high complexity, it can only deal with the design that contains a small number of placement objects. Most algorithms employs the floorplanning techniques to solve the block placement problem.

Standard-cell placement targets the digital design which is synthesized based on standard cells or gate array cells. Such design has the characteristics that the cell heights are unique and the cell sizes are relatively similar. The difficulty of standard-cell placement is the extremely large problem size. Current advanced designs which contain million of cells are very common. To ensure a reasonable runtime can be achieved for such large designs, the heuristic algorithms of most standard-cell placers often exploit the design characteristic of equal cell height. After decades of work, now the state-of-the-art standard-cell placers are able to deliver very optimized results with excellent efficiency for the pure standard-cell designs. These placers apply various heurstics such as recursive partitioning (Capo[2], FengShui[3]), recursive clustering (mPG[4]), analytical techniques (Kraftwerk[5], fastPlace[6]), and the hybrid algorithms that combine partitioning and analytical methods (GORDIAN[8], GORDIAN-L[9]). However, for those designs that the required design characteristic does not exist, most standard-cell placers either cannot function normally or produce unacceptable results. Unfortunately, pure standard-cell designs are rarely seen in the SOC era. Most SOC designs often consists of a number of hard macros, and they must be fixed first in order to make standard-cell placement work. An obvious drawback of such flow is the loss of optimality when placing hard macros without considering the effect of standard cells.

Recently the concept of mixed-size placement is proposed to solve the challenges from these SOC designs. It emphasizes that large macro cells and standard cells can be handled in a single flow without human efforts.

2.2 Previous Work of Mixed-size Placer

Some of the standard-cell placers mentioned above also propose sophisticated approaches to handle large-scale mixed-size designs. A three-stage placement-floorplanning-placement flow that incorporates Capo and the Parquet floorplanner was demostrated in [10]. Khatkhate et al. improved the placer FengShui and introduced a special fractional cut bisection that allows off-row alignment for horizontal cuts[11]. mPG-MS[1], the new version of mPG, clusters the standard cells to form big blocks and eventually clusters with big macros which have similar sizes. With multi-level simulated annealing scheme and careful treatment of macros in the legalization stage, mPG-MS can provide results with comparable quality as Capo. On the other hand, the analytical placer FDP[12] which is based on the method of Kraftwerk, minimizes the quadratic wire length objective and spreads cells by adjusting the extra spreading force. Another analytical placer, APlace, turns to a log-sum-exp wire length model and utilizes nonlinear programming to obtain outstanding wire length results for mixed-size designs. Among these mixed-size placers, the analytical placers exhibit their inherent flexibilities on handling the various design constraints in [5, 14, 15]. In this thesis, we propose a new analytical placement algorithm based on the augmented Lagrangian method to perform the global placement for large-scale mixed-size designs.

Among various analytical placement algorithms, the force-directed method attracted the most attention in recent years. The force-directed method simulates the design netlist as an spring system and solves the classic mechanics problem to determine the cell locations that minimize the total wire length. Such model is proved to be very efficient for large-scale problems. However, by reason that it does not take the cell dimension into consideration, the solution usually contains a large amount of cell overlaps. How to eliminate those cell overlaps without much impact on wire length is actually the key point in force-directed methods. In Kraftwerk[5], additional forces are applied on cells to pull them away from dense regions. The new cell locations are obtained by solving the Poisson's equation. Viswanathan et al.[6] proposed a simple cell shifting technique which determines the magnitude and direction of the new forces by expanding the bins with high utilization and shrinking those with low utilization. This method exhibits outstanding performance due to the fact that only the unconstrained minimization is needed in each iteration. In APlace[7], though a log-sum-exp wire length model is used to substitute the quadratic force model, a similar bin structure is constructed to represent the local information of cell distribution within the placement area. The regions with either too high or too low bin utilization would be penalized in the quadratic penalty function. By performing nonlinear optimization with increasing penalty weight an even cell distribution can be achieved eventually.

2.3 Problem Formulation

2.3.1 Concept

In order to simplify the problem complexity, we only target the wire length minimization as our objective in this thesis. The classic quadratic wire length is chosen as our wire length model for the sake of its simplicity. We also adopt the bin structure to represent rough cell distribution. The utilization of each bin stands for the cell density of this local region. In a global manner, if we can generate a placement which the bin utilizations are all close to the chip density, an even cell distribution is obtained. Hence, we formulate the constraints based on the differences between the chip density and the exact utilizations of the bins. By combining the objective and the constraints, finding a placement with minimized wire length is modeled as a constrained optimization problem. Such constrained optimization can be solved effectively through the augmented Lagrangian method. The details of our problem formulation are described as below.

2.3.2 Quadratic Objective

Given a netlist with n cells and m nets, the placement problem is to find the locations and orientations of all cells so that the total wire length is minimized. Since we do not need to consider the data flow, the netlist can be modeled as a non-directive graph $G = (V, E)$ where each vertex $v_i \in V$ corresponding to a cell with the cell area as the vertex weight. Each edge $e_{ij} \in E$ represents the connectivity between each pair of cells. In this thesis, we model a net as a clique which follows the formula proposed in [13], and thus the edge weight of e_{ij} can be determined by summing up the weights of all the clique edges between v_i and v_j. By constructing this connectivity graph we can calculate the quadratic objective which is actually the sum of the weighted squares of the Euclidean distances between two cells:

$$f(x, y) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}[(x_i - x_j)^2 + (y_i - y_j)^2] \tag{2.1}$$

The objective function can be rewritten in matrix notation as illustrated in [18]:

$$f(\vec{\mathbf{x}}, \vec{\mathbf{y}}) = \frac{1}{2}\vec{\mathbf{x}}'\mathbf{C}\vec{\mathbf{x}} + \vec{\mathbf{d}_\mathbf{x}}'\vec{\mathbf{x}} + \frac{1}{2}\vec{\mathbf{y}}'\mathbf{C}\vec{\mathbf{y}} + \vec{\mathbf{d}_\mathbf{y}}'\vec{\mathbf{y}} + const. \tag{2.2}$$

The vectors $\vec{\mathbf{x}}$ and $\vec{\mathbf{y}}$ denote the coordinates of the movable cells, and the prime denotes vector transposition. The matrix $\mathbf{C}$ represents the connectivity of movable cells. The vectors $\vec{\mathbf{d_x}}$ and $\vec{\mathbf{d_y}}$ are contributed by the connectivity between the movable cells and the fixed cells, and the constant term is contributed by the connectivity between the fixed cells. Solving this unconstrained minimization problem is equivalent to solving the two linear equations:

$$\mathbf{C}\vec{\mathbf{x}} + \vec{\mathbf{d_x}} = 0 \tag{2.3}$$

$$\mathbf{C}\vec{\mathbf{y}} + \vec{\mathbf{d_y}} = 0 \tag{2.4}$$

Such equivalence relationship requires that the connectivity matrix $\mathbf{C}$ is positive definite. Fortunately, this property is always true for general circuits because none of the connectivity would be negative. Hence, the quadratic objective can be easily minimized by any linear equation solvers. In this thesis, we solve Equation (2.3) and (2.4) by conjugate gradient method in order to obtain the initial placement. Note that the fixed cells such as the I/O pads must be provided to guarantee non-zero vectors for both $\vec{\mathbf{d_x}}$ and $\vec{\mathbf{d_y}}$. Otherwise, a trivial solution that $\vec{\mathbf{x}} = \mathbf{0}(\vec{\mathbf{y}} = \mathbf{0})$ will be obtained, which is not what we desire.

2.3.3 Nonlinear Constraints Based on Bin Utilization

The major issue of the quadratic objective function in Equation (2.2) and (2.3) is that it would generate a placement with a large amount of cell overlaps. The reason is evident since there is no information about cell dimension at all in the equation. Due to that the number of internal connections is usually larger than that of the connections to fixed I/O pads, in most cases, the unconstrained optimization would result in the placement which the cell density at the center of is much higher than that at the chip boundary. Figure 2.1 demostrates such result for a 100-block circuit and compares it with

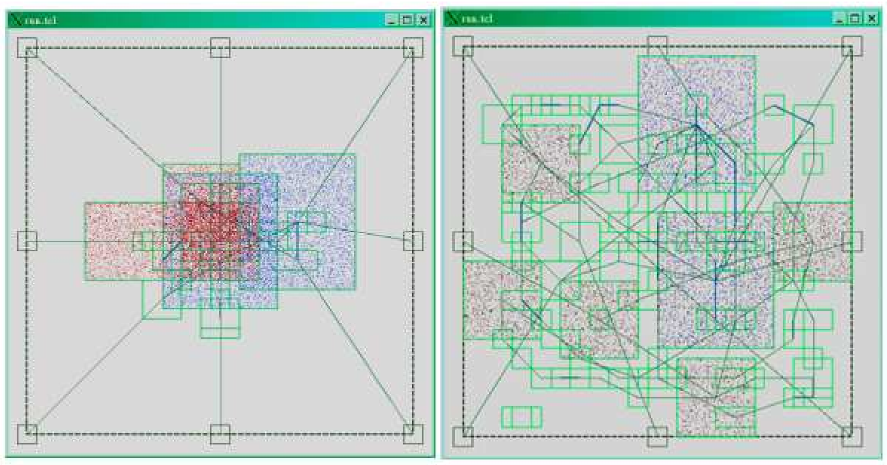


Figure 2.1: The comparison of unconstrained and constrained optimization for a simple case

another more even solution generated by constrained optimization. In order to eliminate the cell overlaps, we must evaluate the cell distribution to guide us how to push cells toward the vacant region. A straightforward model to measure the cell distribution is the bin structure. Let the chip area be divided by k bins, and all the bins have the same width w_b and height h_b. The bin utilization of bin b_j is denoted as u_j, which is defined as the total cell areas within b_j over the bin area $w_b \times h_b$. Given a placement, the local cell density can be measured by the utilizations of the k bins. If one bin has the utilization over 100%, it is impossible to find a feasible placement inside this bin, and thus some of the cells must be removed to decrease the bin utilization. Since our goal is to generate the placement which every bin utilization is close to the average chip density, we can define the constraints as

$$u_j - U = 0, \text{for } j = 1 \dots k \tag{2.5}$$

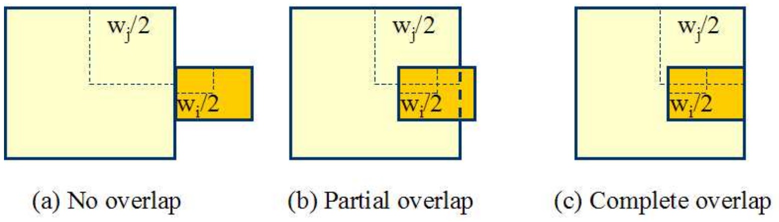


Figure 2.2: The three types of x-dimension overlap for small cells

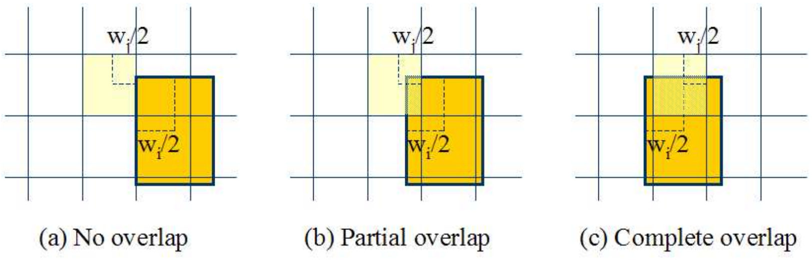


Figure 2.3: The three types of x-dimension overlap for large cells

, where U is the average chip density, that is, the total cell area over the chip area. In this thesis, we adopt a unified model to calculate the exact area that a cell contributes to each bin, no matter the cell size is larger or smaller than the bin size. The model is described as follows.

Consider a cell c_i with width w_i and height h_i, and a bin b_j with width w_j and height h_j. Also assume that the cell width is smaller than the bin width. Let d_x and d_y denote the center-to-center distance between c_i and b_j in x-dimension and y-dimension, respectively. The overlapping in x-dimension can be classified into three cases as illustrated in Figure 2.2. If d_x is larger than or equal to $(w_j + w_i)/2$, there is no overlapping. As d_x is decreased to be less than $(w_j + w_i)/2$, c_i starts to overlap with b_j in x-dimension, and the length of overlapping is $(w_j + w_i)/2 - d_x$, which grows as d_x decreases. When d_x is further decreased to be less than or equal to $(w_j - w_i)/2$, the length of overlapping will saturate at its maximum value, that is, $min(w_j, w_i)$. Another scenario is that the cell width is larger than the bin width, which may happens while dealing with the big macro cells. As shown in Figure 2.3, the overlapping is similar with that of the former scenario, except $(w_i - w_j)/2$ substitutes with $(w_j - w_i)/2$ and $min(w_j, w_i)$ changes to w_j. The overlapping in y-dimension can also be calculated in the similar manner. Combine the two scenarios of different cell sizes and consider the x-dimension and the y-dimension together, the exact overlap area of cell c_i and bin b_j can be generalized as below :

$$a_{ij} = w_s h_s M_x(d_x) M_y(d_y) \tag{2.6}$$

, where $w_s = min(w_i, w_j)$, $h_s = min(h_i, h_j)$, M_x and M_y are actually two piecewise-linear functions of d_x and d_y, respectively.

$$M_x(d_x) = \begin{cases} 1, & \text{if } d_x \leq \delta_x \\ \dfrac{-d_x + \delta_x + w_s}{w_s}, & \text{if } \delta_x < d_x < \delta_x + w_s \\ 0, & \text{otherwise} \end{cases} \tag{2.7}$$

$$M_y(d_y) = \begin{cases} 1, & \text{if } d_y \leq \delta_y \\ \dfrac{-d_y + \delta_y + h_s}{h_s}, & \text{if } \delta_y < d_y < \delta_y + h_s \\ 0, & \text{otherwise} \end{cases} \tag{2.8}$$

,where δ_x denotes $|w_i - w_j|/2$, and δ_y denotes $|h_i - h_j|/2$. With this unified model, the exact overlap between cells and bins can be easily computed. Now the bin utilization of each bin can be rephrased as follows :

$$u_j = \frac{1}{w_b h_b} \sum_{i=1}^{n} a_{ij}, \text{for } j = 1 \ldots k \tag{2.9}$$

Combine the quadratic objective and the nonlinear constraints, the placement problem is restated in the following form :

$$\begin{aligned} &\text{Minimize} \quad f(\vec{\mathbf{x}},\vec{\mathbf{y}}) = \tfrac{1}{2}\vec{\mathbf{x}}'\mathbf{C}\vec{\mathbf{x}} + \vec{\mathbf{d_x}}'\vec{\mathbf{x}} + \tfrac{1}{2}\vec{\mathbf{y}}'\mathbf{C}\vec{\mathbf{y}} + \vec{\mathbf{d_y}}'\vec{\mathbf{y}} \\ &\text{subject to} \quad c_j(\vec{\mathbf{x}},\vec{\mathbf{y}}) = u_j(\vec{\mathbf{x}},\vec{\mathbf{y}}) - U = 0, \text{for } j = 1 \ldots k \end{aligned} \tag{2.10}$$

Here we describe the notations again. $\mathbf{C}$ denotes the matrix of the connectivities between any two movable cells. Each element in $\vec{\mathbf{d_x}}$ denotes the sum of product of the connectivity between the movable cell to each fixed cell and the X coordinate of the fixed cell. Similarly, $\vec{\mathbf{d_y}}$ is a vector composed of the sum of product of the connectivity between the movable cell to each fixed cell and the Y coordinate of the fixed cell. u_j denotes the utilization of bin b_j, and U denotes the average chip density. For the chip area divided into k bins, there are k constraints.

In Equation (2.10), the objective to be minimized is quadratic, but the constraints are nonlinear to $\vec{\mathbf{x}}$ and $\vec{\mathbf{y}}$. This equation implies that any solver can solve this nonlinear constrained optimization problem effectively can be used as a global placer for mixed-size placement. The details of our approach is discussed in the next chapter.

Chapter 3

Algorithm and Implementation

3.1 Main Flow

The constrained optimization is usually attacked by solving a sequence of unconstrained problems. In each unconstrained problem, the merit function, which is composed of the original objective function and the constaints, is to be minimized. Since our formulation contains a quadratic objective function and nonlinear constraints, the merit function is nonlinear. In our work, a nonlinear minimizer based on the conjugate gradient method is implemented as the fundation of the solver. The merit function is formulated following the augmented Lagrangian (ALAG) method[19], which can be viewed as a combination of the quadratic penalty function and the ordinary Lagrangian method. Unlike the quadratic penalty function, the ALAG method usually generates a solution without an extremely large penalty, and thus avoids ill-conditioning effects. It also provides better convergence rate than ordinary Lagrangian method. Although the ALAG method has such advantages over other constrained optimization methods, it still does not guarantee to obtain the minimum from any starting point in the solution space. In order to make

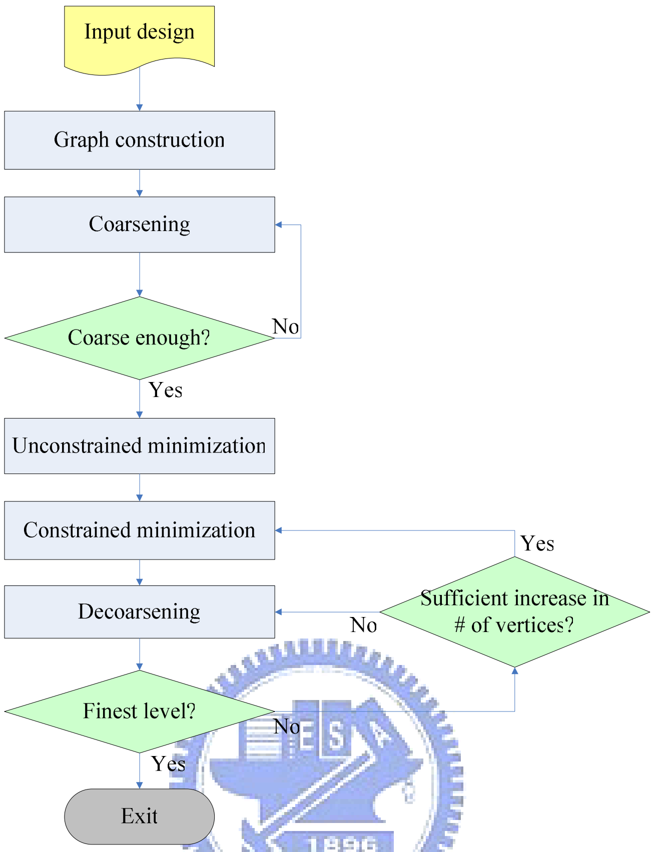


Figure 3.1: The main flow of our algorithm

a good choice of the starting point, the multi-level scheme is approached, as the procedure shown in Figure 3.1. Our algorithm begins with a multi-level graph coarsening, which is followed by a unconstrained minimization to determine the initial point for the coarsest graph. Then a sequence of constrained minimizations and graph uncoarsenings are performed. In each level, first the ALAG method is executed, and the solution is used as the starting point for the next finer level. Such process repeats until the finest graph is uncoarsened, and a solution of the global placement is finally obtained. For the purpose of improving the efficiency, an extra decision branch is added to skip the ALAG method when the number of vertices is not sufficiently larger than that in last coarser level in which the ALAG method was performed. We set the vertex ratio as 1.4. In other words, every time the ALAG method is performed, the graph size is at least 1.4x larger than that of last run. The implementation details are described in the following sections.

3.2 Augmented Lagrangian Method

The augmented Lagrangian method introduces one penalty parameter μ for the quadratic penalty term and explicit Lagrange multiplier estimate λ_j for each constraint $c_j(\vec{\mathbf{x}}, \vec{\mathbf{y}})$. By relaxing the constraints $c_j(\vec{\mathbf{x}}, \vec{\mathbf{y}})$ into the original objective $f(\vec{\mathbf{x}}, \vec{\mathbf{y}})$, the constrained problem in Equation (2.10) is transformed into a sequence of unconstrained problems shown in the following equation:

$$\mathcal{L}(\vec{\mathbf{x}}, \vec{\mathbf{y}}) = f(\vec{\mathbf{x}}, \vec{\mathbf{y}}) - \sum \lambda_j c_j(\vec{\mathbf{x}}, \vec{\mathbf{y}}) + \frac{1}{2\mu} \sum c_j^2(\vec{\mathbf{x}}, \vec{\mathbf{y}}), \text{for } j = 1 \ldots k \tag{3.1}$$

At the n-th iteration, the ALAG method fixes the penalty parameter μ^n and all the Lagrange multiplier estimates λ_j^n, and performs unconstrained minimization with respect to $\vec{\mathbf{x}}$ and $\vec{\mathbf{y}}$. After an approximate minimizer is found, we check the convergence of the merit value. If the convergence criterion is satisfied, the ALAG method is terminated with approximate solution $(\vec{\mathbf{x}}^n, \vec{\mathbf{y}}^n)$. Otherwise, we update the Lagrange multiplier estimates and the penalty parameter for the next iteration. The algorithm is shown in Figure 3.2.

```
given initial solution $(\mathbf{x}^{0_s}, \mathbf{y}^{0_s})$
repeat
    find $(\mathbf{x}^k, \mathbf{y}^k)$ that minimizes $L_k(\mathbf{x},\mathbf{y})$ from $(\mathbf{x}^{k_s}, \mathbf{y}^{k_s})$
    if $(\|\mathbf{c}(\mathbf{x}^k, \mathbf{y}^k)\| <$ the tolerence $\eta_k)$
        if (no obvious reduction on $L_k(\mathbf{x},\mathbf{y})$)
            break with solution $(\mathbf{x}^k, \mathbf{y}^k)$
        foreach Lagrange multiplier
            $\lambda_j^{k+1} = \lambda_j^k + c_j(\mathbf{x}^k, \mathbf{y}^k)/\mu^k$
    else
        $\mu_{k+1} = 0.5\mu_k$
    $(\mathbf{x}^{k+1_s}, \mathbf{y}^{k+1_s}) = (\mathbf{x}^k, \mathbf{y}^k)$
```

Figure 3.2: The algorithm of ALAG method

3.3 Nonlinear Conjugate Gradient Minimization

The subproblem stated in Equation (3.1) indicates that a nonlinear unconstrained minimization is required within every ALAG iteration. While choosing the minimizer, its performance is the major concern since it is located in the loop and will be called many times. Here we choose the conjugate gradient method[19]. The conjugate gradient method is an iterative method and is very popular in large-scale problems due to its efficiency. In each iteration, only several vector operations are needed to determine the search direction, and a one-dimensional minimization is performed along the search direction to find the suitable step size. With successive line searchs, this algorithm can gradually reach the local minimum.

At the k-th iteration, the search direction $\mathbf{p}_k$ is a linear combination of the negative gradient $-\mathbf{g}_k$ and the previous search direction $\mathbf{p}_{k-1}$. When $k = 0$, the negative gradient at the initial solution $\mathbf{x}_0$ is an intuitive choice for the initial search direction, since the previous search direction does not exist. In general, the search direction can be expressed in the following form :

$$\mathbf{p}_k = \begin{cases} -\mathbf{g}_k, & \text{if } k = 0 \\ -\mathbf{g}_k + \beta_k \mathbf{p}_{k-1}, & \text{otherwise} \end{cases} \tag{3.2}$$

where β_k is a scalar such that $\mathbf{p}_k$ is conjugate to $\mathbf{p}_{k-1}$. There are several variants in the choice of β_k, and here we follows the Polak-Ribiere formula[20] :

$$\beta_k = \frac{\mathbf{g}'_k(\mathbf{g}_k - \mathbf{g}_{k-1})}{\mathbf{g}'_{k-1}\mathbf{g}_{k-1}} \tag{3.3}$$

After the search direction $\mathbf{p}_k$ is obtained, the step size α_k is determined by finding the approximate one-dimensional minimizer along the direction, and thus the new solution for next iteration is given by :

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \tag{3.4}$$

Our one-dimensional minimization combines the line seacrah method following the Armijo's rule and the Fibonacci search method. Both details can be found in [20]. The Armijo's rule is also known as the Wolfe condition, which uses the first-order approximation to decide whether a step makes enough cost reduction in the merit function. If the step makes enough reduction, we accept it and further expand the step size to examine the Armijo's rule again. On the contrary, if the reduction is not satisfied, we repeatly contract the step size until either the Armijo's rule is satisfied or the step size becomes too small. In each line search, first we determine the interval that contains the minimum by examining the Armijo's rule, and then we perform the Fibonacci search within the interval to find the most suitable step size.

The stopping criteria of our conjugate gradient method are : (1) the gradient is too small, (2) the merit value cannot be further improved after several iterations, or (3) a maximum number of iterations is reached.

3.4 Negative Gradient Evaluation

An important factor in the conjugate gradient method is how to compute the negative gradient for the complex merit function presented in Equation (3.1). In our work, we develop a special approximation of the negative gradient. Let $(\vec{\mathbf{x}}^k, \vec{\mathbf{y}}^k)$ denotes the current design point of the k-th iteration of the conjugate gradient method, (x_i^k, y_i^k) denotes the i-th element of $(\vec{\mathbf{x}}^k, \vec{\mathbf{y}}^k)$, and δ denotes a unit distance. In the evaluation of the negative gradient at the k-th iteration, for each element (x_i^k, y_i^k) eight different directions are examined with moving a δ stepsize. More specifically, (x_i^k, y_i^k) is replaced by $(x_i^k + \delta, y_i^k)$, $(x_i^k + \delta, y_i^k + \delta)$, $(x_i^k, y_i^k + \delta)$, $(x_i^k - \delta, y_i^k + \delta)$, $(x_i^k - \delta, y_i^k)$, $(x_i^k - \delta, y_i^k - \delta)$, $(x_i^k, y_i^k - \delta)$, and $(x_i^k + \delta, y_i^k - \delta)$, respectively, and thus eight different design points are evaluated to obtain the difference of the merit value, comparing with the original design point. The difference actually represents the local information that how the merit value is affected if the cell moves along this direction. Obviously, the movement that causes the most decrease in the merit value should be chosen, and the corresponding element of the negative gradient can be determined by the definition of the gradient, that is, the difference of the merit value over the distance. If this move only causes the change in the X-direction, for example, then the Y-component of the element of the negative gradient is 0, and vice versa.

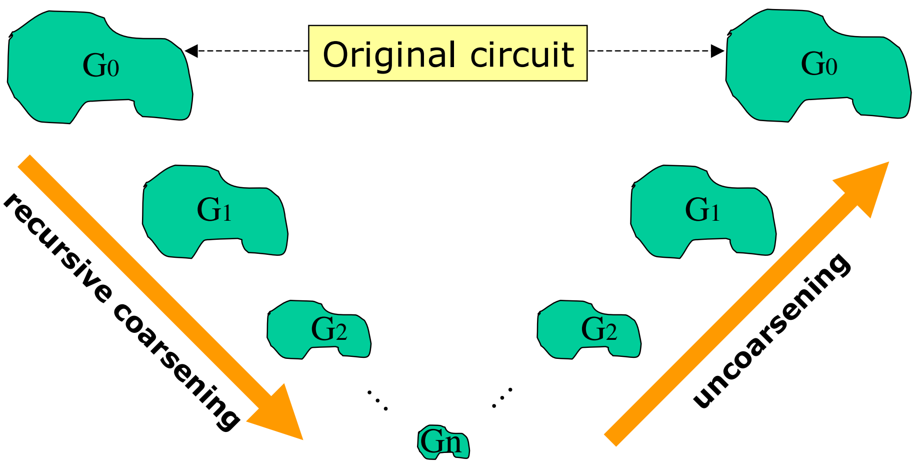


Figure 3.3: The coarsening/uncoarsening flow

It is clear that the evaluation of the negative gradient would take much more time then that of the merit value. However, with careful implementation, the time spent on one evaluation of the negative gradient can be less than eight times of that spent on one evaluation of the merit value.

3.5 Multi-Level Scheme

Multi-level scheme is widely adopted in the domain of physical design. A typical implementation consists of the coarsening phase and the uncoarsening phase, as shown in Figure 3.3. In the beginning the problem is recursively coarsened to reduce the problem size. The optimization is performed at the coarsest level, and then in each of the following finer levels the local refinement is performed during the uncoarsening phase. The mult-level scheme is very attractive in the runtime reduction without too much quality loss. Placers with such scheme can be found in [1, 7, 13].

In this thesis, the implementation of a multi-level framework is mainly due to the lack of a good initial design point for the ALAG method. It is well-known that the inital design point significantly affects the result and the performance in nonlinear constrained optimization[21]. But how to decide a reasonable initial design point? We think that the optimization result of the coarser level should be a proper choice. During the uncoarsening phase, the solution obtained in the coarser level is inherited as the initial solution of the ALAG method in the current level. The only exception is at the coarest level, where a unconstrained optimization is first performed to obtained the required initial solution. By testing a simple case which contains 107 cells and 151 nets, we confirmed that such multi-level framework is superior to the direction optimization with the initial design point given by random guess. The row 2 to row 9 in Table 3.1 show the placement results with 8 random initial design points, and their average results are shown in the row 10. The row 11 shows the placement result with the same optimization parameters except for the initial design point obtained by a 7-level optimization. It is clear that both the half-perimeter wire length and the amount of cell overlapping obtained by the multi-level scheme are better than those obtained in the random cases.

Our coarsening/uncoarsening approach follows the algorithms proposed by metis and hMetis[16, 17]. The randomized "First-Choice" matching is performed to cluster the adjacent vertices based on the connectivity and vertex weights. Here the vertex weight is the cell area. When two vertices are clustered together, their weights are summed up to form the new vertex. We recursively coarsen the graph until (1) the graph size is small enough that the number of vertices is less than a given lower bound, saying 10. (2) there

Table 3.1: The placement results with different initial design points

	HPWL	**Cell overlap**
random 1	242.00	388.88
random 2	376.00	183.13
random 3	444.00	156.63
random 4	453.00	155.38
random 5	423.00	182.38
random 6	390.00	167.50
random 7	378.00	167.38
random 8	413.00	177.63
avg.	389.88	197.36
multi-level	344.00	157.50

exists some vertex whose weight exceeds a given maximum weight.

3.6 Legalization

3.6.1 Legalization Flow

After the ALAG method is performed at the finest level, we can obtain a placement solution that every bin has similar cell utilization. It is obvious that such result doesn't guarantee overlap-free, and thus an extra legalization step is necessary. This step can be further divided into three stages: (1)macro legalization, (2)row legalization, and (3)detailed placement.

In the first stage, only the macro cells are legalized without considering the standard cells. After a solution can be obtained that none of the macro cells overlaps with each other, these macro cells are set fixed, and we go to next stage to legalize the standard cells. The key point of the stage 1 and stage 2 is that the legalization must honor the solution of our global placement and only moves cells locally. After stage 2, the placement is a legal solution, but the total wire length may increase. Hence, the final stage is to refine the standard cells locally to further reduce the wire length as well as to meet other design constraints without cell overlaps. In our work, we evaluate a new macro packing algorithm to perform macro legalization, but we do not cover the row-based legalization and detailed placement.

3.6.2 Implementation of Macro Legalization

Our macro legalization is based on the block packing technique which is widely used in design floorplanning. The fundation of block packing is a floorplan representation, with which the geometric relationship of the blocks can be well-defined. By disturbing the representation sequentially, we can evaluate different packings and choose the one with the best cost. We developed a representation which is actually a modification of B*-Tree[23]. B*-Tree is an ordered binary tree in which a node stands for a block to be packed. The root node is initially placed as the bottom-left block, and the leaf nodes are placed by the rules as follows.

- If node n_j is the left child of node n_i, module b_j must be located on the right-hand side and adjacent to module b_i.
- If node n_j is the right child of node n_j, module b_j must be located above and adjacent to module b_i.

After traversing the whole tree, all the blocks are placed, and a nonslicing floorplan is constructed. However, the packing along the bottom-left direction is not what we desired. In most commercial ICs, the macro cells are often placed around the chip boundary, and the chip center can forms a complete region for standard cell placement. To ensure the macro cells can be packed in such style, a four-B*-Tree stretegy is employed. We pack the four trees from the four chip corners. The bottom-left tree is actuall a B*-tree, and the top-left tree is a clockwise 90-degree rotation of B*-Tree. Similarily, the top-right tree and the bottom-right tree are also rotated B*-Trees.

In the beginning the legalization, the macro cells are assigned to the four trees by their locations which are determined by the global placement. For each tree, the sortings along the X-direction and Y-direction are performed to decide the geometric relationships of these cells. Thus, the trees can be constructed following the B*-Tree definition. Then we start the SA process to disturb the trees. In each iteration, only one tree is selected, and one of the three possible operations is applied on the node which is chosen by random.

- ROTATE - Rotate the cell by 90-degree, 180-degree or 270-degree.
- MOVE - Insert a node n_i to be the child of another node n_j. The child trees of both n_i and n_j may be affected recursively.
- SWAP - Switch two nodes n_i and n_j in the tree.

The cost function of our SA process is composed of two terms. The first term is the sum of the distance from the block to the chip corner of the tree. The second is the related HPWL of the blocks. We also make the weight of the first term larger than that of the HPWL term to ensure a compact packing. The result of macro legalization is discussed in Section 4.2.

Chapter 4

Experimental Results

We implemented our placer in C/C++ and compiled it using g++ on cygwin, which emulates Linux environment on Windows XP. We ran the program on a 1.5GHz PC with 512MB memory, and used the IBM ICCAD'04 benchmark set in LEF/DEF format as testcases, which can be downloaded from [21].

Table 4.1: Some ICCAD'04 benchmark cases

	# of inst(core/macro/pad)	**# of nets**	**# of masters**
ibm01	12752 (12260 / 246 / 246)	14111	2846
ibm02	19601 (19071 / 271 / 259)	19584	3057
ibm03	23136 (22563 / 290 / 283)	27401	3757
ibm04	27507 (26925 / 295 / 287)	31970	4587
ibm05	39347 (28146 / 0 / 1201)	28446	4911
ibm06	32498 (32154 / 178 / 166)	34826	4598
ibm07	45926 (45348 / 291 / 287)	48117	5748
ibm08	51309 (50722 / 301 / 286)	50513	4235

4.1 Some IBM ICCAD'04 Results

We compared our placement results with other academic placers by applying the ICCAD'04 benchmark set. This benchmark set contains 18 large-scale designs and most of them include large movable macros. These macros are all hard blocks with fixed aspect ratios and pin locations. In addition, the locations of I/O pads are fixed around the chip boundary. Such characteristic is a must for force-directed methods such as our placer, because the fixed cells provides the external energy to expand the movable cells. Due to the short of resources, we only ran the first 8 cases. The design chararcteristics of the cases we tested are listed in Table 4.1, and the comparison results are shown in Table 4.2.

Table 4.2: The comparison of the placement results

	APlace gpWL	APlace dpWL	FengShui dpWL	Capo dpWL	MPG-MS dpWL	Ours gpWL	Ours runtime
ibm01	2.17	2.14	2.56	2.67	3.01	2.74	12.35
ibm02	4.83	4.61	6.05	5.54	7.42	5.60	26.50
ibm03	6.94	6.72	8.77	8.67	11.20	8.13	34.00
ibm04	7.70	7.60	8.38	9.79	10.50	9.83	46.58
ibm05	9.82	9.70	9.94	10.82	10.90	10.01	36.46
ibm06	6.31	5.99	6.99	7.35	9.21	8.79	65.25
ibm07	10.04	10.02	11.37	11.23	13.70	11.96	72.78
ibm08	12.65	12.34	13.51	16.02	16.40	15.52	91.93

The first two columns show the HPWL of APlace global placement and APlace detailed placement. By normalizing the gpWL to 1, we found that dpWL is 0.975 smaller than gpWL in average. It indicates that with a strong correlation between global placement and detailed placement, the wirelength

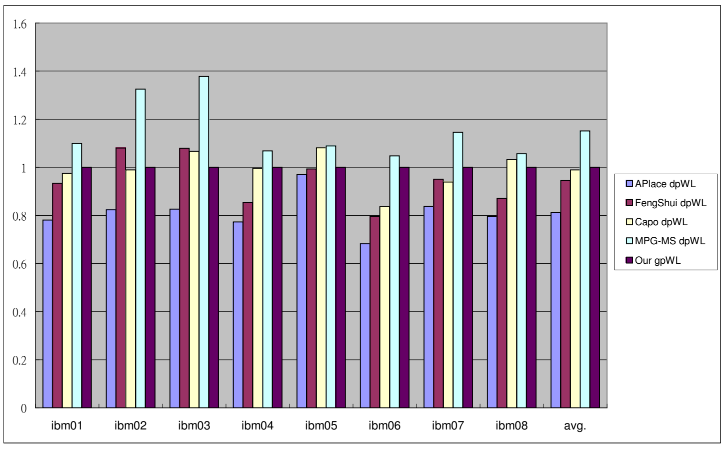


Figure 4.1: The normalized HPWL comparison

could be further reduced after legalization and detailed placement, and the HPWL of global placement and detailed placement should be very close. The 3th, 4th, and 5th column are the detailed placement results of FengShui v2.6, Capo v9.0, and MPG-MS, respectively. The data of the first four columns were reported in [22], and the data of the 5th column was reported in [1]. The last two column show the global placement HPWL and the runtime of our placer. We also normalized our HPWL as 1 to see the differences with the other placers, as depicted in Figure 4.1. The last set is the average from ibm01 to ibm08, and the values are 0.81, 0.94, 0.99, 1.15 and 1, respectively. Our results for the eight cases are depicted in Figure 4.2 to Figure 4.9.

4.2 Macro Legalization

We implemented the block packing that mentioned above in our multi-level framework. Every time a finer level begins, we check whether there exists any macro cell as a single node in the current graph. If it does, the SA-based block packing is performed before the ALAG method. The parameters of our

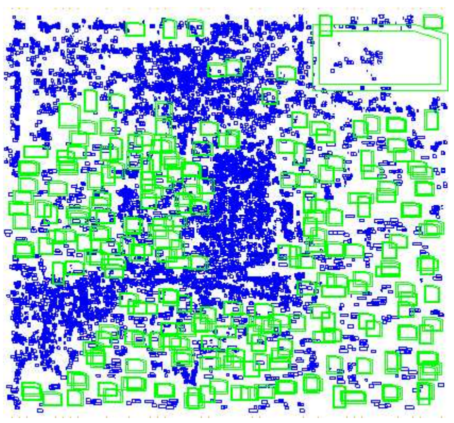


Figure 4.2: IBM01 result

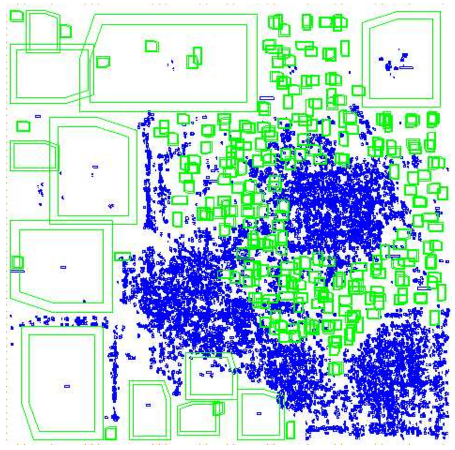


Figure 4.3: IBM02 result

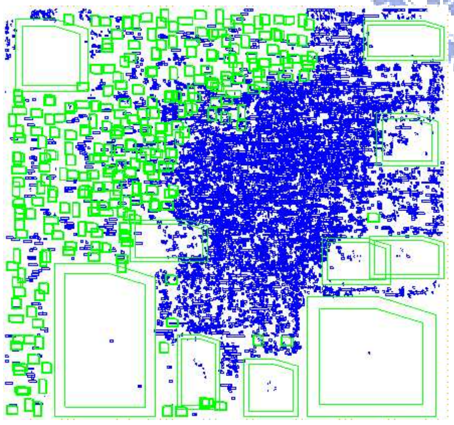


Figure 4.4: IBM03 result

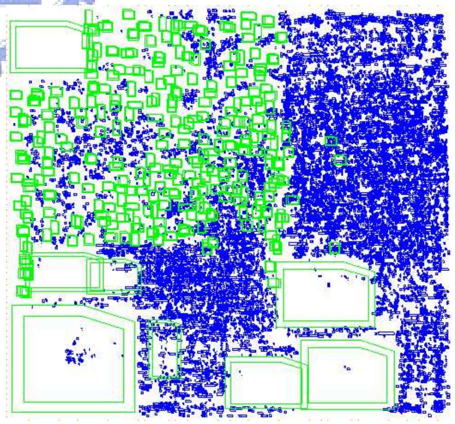


Figure 4.5: IBM04 result

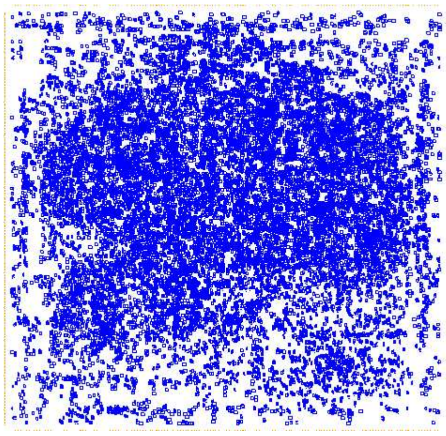


Figure 4.6: IBM05 result

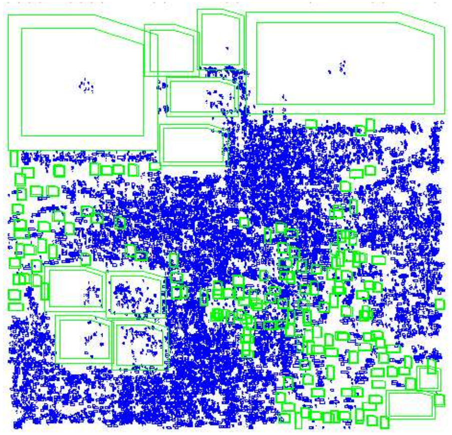


Figure 4.7: IBM06 result

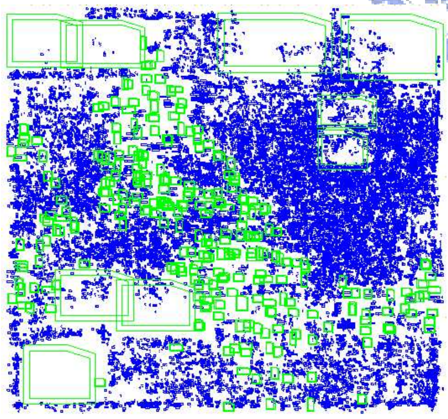


Figure 4.8: IBM07 result

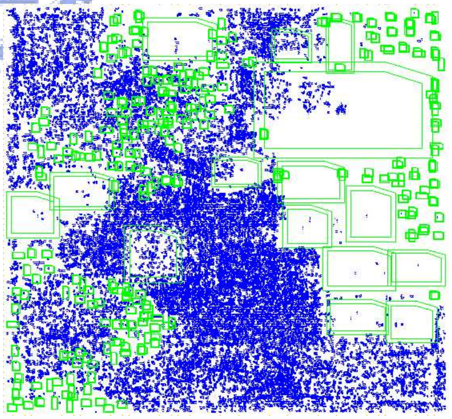


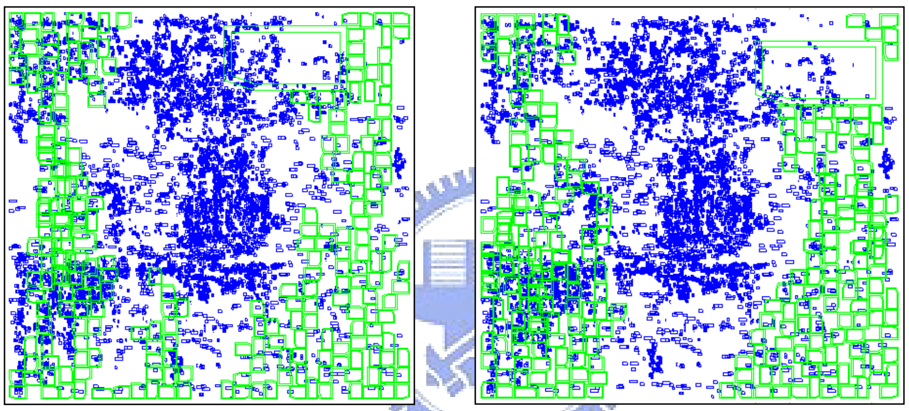
Figure 4.9: IBM08 result

SA process are described as follows. We set the temperature-decreasing rate as 0.85 and the stopping temperature T_{stop} as 5.0^{-4} of the initial temperature T_0. At each temperature, the number of moves is set as $300 * thenumofblocks$. And the uphill probability is 0.85 at the initialization. Following such parameter setting to test the ibm01 case, the experimental result is shown in the 2nd column of Table 4.3. Comparing with the original result which we list again in the first column, it clearly indicates that such block packing impacts the HPWL greatly. The 3rd column shows the result when we further decrease T_{stop} to 1.5^{-4}. The HPWL is improved slightly at the cost of runtime. The corresponding placements for the two packings are depicted in Figure 4.10. We can see that the cell distribution is quite uneven. This is because the block packing disturbs the solution too much. Even we try to maintain the geometic relationships between blocks at the tree construction, the following SA process would destroy the relationships eventually. Since the solution for the macro cells becomes very different, it also affects the standard cells and escapes the result which we obtained through the multi-level scheme.

From the experimental results, we realize that the SA-based block packing is not suitable to incorperate with our algorithm. We should seek for a legalizer which can honor the global placement solution with minimum movement toward the chip boundary.

Table 4.3: The HPWL of ibm01 with different legalization

	No Packing	**SA with $T_{stop} = T_0 * 5.0^{-4}$**	**SA with $T_{stop} = T_0 * 1.5^{-4}$**
HPWL	2.74	3.79	3.54



(a) $T_{stop} = 5.0^{-4}T_0$ (b) $T_{stop} = 1.5^{-4}T_0$

Figure 4.10: IBM01 placement after macro legalization

Chapter 5

Conclusion

A new multi-level mixed-size placer is proposed in this thesis. It is based on a unified model of the placement problem and utilizes the augmented Lagragian method to optimize the HPWL. The multi-level scheme not only improves the performance but also helps decide good initial points for nonlinear programming. The experimental results show that our placer generates good global placements, and the quality is comparable to current state-of-the-art placers.

Many features are necessary to form a complete placement framework. The future works include a robust legalizer for both macro cells and standard cells, and an effective algorithm for detailed placement. The performance of our placer can be further improved. And more design constraints such as congestion and power should be supported to make our placer more practical.

Bibliography

[1] C.-C. Chang, J. Cong, and X. Yuan, Multi-level Placement for Large-Scale Mixed-Size IC Designs , In *Proc. Asia South Pacific Design Automation Conference*, pp. 325-330, 2003.

[2] A. E. Caldwell, A. B. Kahng, and I. L. Markov, Can Recursive Bisection Alone Produce Routable Placements? , In *Proc. Design Automation Conference*, pp. 477-482, 2000.

[3] A. Agnihotri, M. C. Yildiz, A. Khatkhate, A. Mathur, S. Ono, and P. H. Madden, Fractional cut: Improved recursive bisection placement , In *Proc. International Conference on Computer-Aided Design*, pp. 307-310, 2003.

[4] C.-C. Chang, J. Cong, D. Pan and X. Yuan, Physical hierarchy generation with routing congestion control, In *Proc. International Symposium on Physical Design*, pp. 36-41, 2002

[5] H. Eisenmann and F. M. Johannes, Generic Global Placement and Floorplanning , In *Proc. Design Automation Conference*, pp. 269-274, 1998

[6] N. Viswanathan and Chris C.-N. Chu, FastPlace: Efficient Analytical Placement using Cell Shifting, Iterative Local Refinement and a Hybrid

Net Model, In *Proc. International Symposium on Physical Design*, pp. 26-33, 2004

[7] Andrew B. Kahng and Q. Wang, Implementation and Extensibility of an Analytic Placer, In *Proc. International Symposium on Physical Design*, pp. 18-25, 2004

[8] J. M. Kleinhans, Georg Sigl, F. M. Johannes and K. J. Antreich, GORDIAN: VLSI Placement by Quadratic Programming and Slicing Optimization, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol.10, No.3, Mar 1991

[9] Georg Sigl, K. Doll and F. M. Johannes, Analytical Placement: A Linear or a Quadratic Objective Function? , In *Proc. ACM/IEEE Design Automation Conference*, pp. 427-431, 1991

[10] S. N. Adya and I. L. Markov, Consistent Placement of Macro-Blocks Using Floorplanning and Standard-Cell Placement , In *Proc. International Symposium on Physical Design*, pp. 12-17, 2002

[11] A. Khatkhate, Chen Li, A. R. Agnihotri, M. C. Yildiz, S. Ono, C.-K. Koh and P. H. Madden, Recursive Bisection Based Mixed Block Placement , In *Proc. International Symposium on Physical Design*, pp. 84-89, 2004

[12] K. Vorwerk, A. Kennings and A. Vannelli, Engineering Details of a Stable Force-Directed Placer, In *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pp. 573-580, 2004

[13] B. Hu and M. Marek-Sadowska, Fine Granularity Clustering-Based Placement, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol.23, No.4, Apr 2004

[14] A. B. Kahng and Q. Wang, An Analytical Placer for Mixed-Size Placement and Timing-Driven Placement, In *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pp. 565-572, 2004.

[15] Y.-C. Chou and Y.-L. Lin, A Performance-Driven Standard Cell Placer Based on a Modified Force-Directed Algorithm, In *Proc. International Symposium on Physical Design*, pp. 24-29, 2001.

[16] G. Karypis and V. Kumar, A fast and high quality multilevel scheme for partitioning irregular graphs, In *SIMA Journal on Scientific Computing*, Vol.20, No.1, pp. 359-392, 1999

[17] G. Karypis, V. Kumar and S. Shekhar, Multilevel Hypergraph Partitioning: Application in VLSI DomainIn *Proc. ACM/IEEE Design Automation Conference*, pp. 526-529, 1997

[18] K. M. Hall, A r-dimensional quadratic placement algorithm, In *Management Science*, pp. 219-229, 1970

[19] D. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*, Academic Press, 1982

[20] M. S. Bazaraa, H. D. Sherali and C. M. Shetty, *Nonlinear Programming - Theory and Algorithms, 2nd Edition*, John Wiley & Sons, 1993

[21] S. N. Adya, S. Chaturvedi, A. Roy, D. Papa and I. L. Markov, Unification of Partitioning, Floorplanning and Placement, In *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pp. 550-557, 2004 (URL: http://vlsicad.eecs.umich.edu/BK/ICCAD04bench/)

[22] A. B. Kahng, S. Reda and Qinke Wang, Architecture and Details of a High Quality, Large-Scale Analytical Placer, In *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pp. 890-897, 2005

[23] Y.-C. Chang, Y.-W. Chang, G.-M. Wu and S.-W. Wu, B*-Trees: A New Representation for Non-Slicing Floorplans, In *Proc. IEEE/ACM Design Automation Conference*, pp.458-463, 2000