Non-Rectangular Shaping and Sizing of Soft Modules for Floorplan Design Improvement

ABSTRACT

Many previous works on rectilinear block packing [7; 12; 4; 5; 6; 3; 9; 10; 11; 1; 14; 2] assume that some input modules are pre-designated to have a particular rectilinear shape, e.g., L-shape, T-shape, etc. However, this may not be the case in practice. Many modules still have large flexibilities in shapes and dimensions in the early floorplanning step. Instead of restricting some modules to be L-shaped or T-shaped at the beginning, the non-rectangular modules can indeed be generated by the floorplanning step, in order to improve the packing quality. After a preliminary floorplan is designed based on the major criteria like interconnect cost, delay and area, we can modify the shapes (from rectangular to non-rectangular) and dimensions of the modules to fill up the unused area, while keeping the relative spatial relationship between the modules unchanged. (This relationship may come from the optimization in the initial floorplanning step.) In this paper, we study this problem of changing the shapes and dimensions of the flexible modules in a best fit way to fill up the unused area of a preliminary floorplan, while keeping the relative positions between the modules unchanged. This feature will also be useful in fixing up small and incremental changes during ECO modifications. We formulate the problem as a mathematical program. The formulation is in such a perfect way that the corresponding Lagrangian relaxation subproblem can be solved optimally, and the optimal dimensions of all the rectangular and non-rectangular modules can be computed by closed form equations in $O(m)$ time where m is the total number of edges in the constraint graphs. We tested our method using some benchmark data and the experimental results show that an average reduction of about 3.6% of deadspace is achievable by shaping and sizing.

1. INTRODUCTION

Floorplanning has become increasingly important in the physical design of VLSI circuits due to the advance in the deep sub-micron technology. New packaging schemes such as Multi-Chip Modules (MCMs) and integrated circuit components may sometimes have their shapes more complex than a simple rectangle. A lot of works have then been reported on floorplanning with rectilinear blocks [12; 3; 4; 5; 9; 10; 11; 6; 1; 14; 2]. The papers [13; 3] extend the Polish expression representation for slicing floorplans to handle L-shaped and T-shaped modules. The works on non-slicing floorplans are either based on the Bounded Sliceline Grid (BSG) structure [4; 5; 9; 10; 11] or the Sequence Pair (SP) representation [6; 14; 1; 2]. Most of these works explore the rules to restrict the placement of the rectangular sub-blocks of a rectilinear module, so that the sub-blocks will be placed adjacent to one another in an appropriate way to form back its required rectilinear shape in the final packing.

However, in all these previous works, it is assumed that some input modules are pre-designated to have a particular rectilinear shape, e.g., T-shaped, L-shaped, etc, and full optimizations are performed by considering the objectives such as area, timing and power consumption as a whole. This may not be the case in practice. In fact, rectangular shapes are more preferable in many designing steps. They are easier to be managed not only in floorplanning algorithms, but also in downstream pin assignment, placement, routing and timing analysis algorithms. Non-rectangular shapes are often considered only when they can improve the packing further.

One strategy is that we can perform a preliminary floorplan design with all the soft-blocks in rectangular shapes at the first stage. After a preliminary floorplan is obtained based on some important criteria like interconnect cost, delay and area, we explore the idea of allowing the modules to change in shapes and dimensions to improve the packing at the final stage, while keeping the relative spatial relationships between the modules unchanged. This step is useful especially when the preliminary floorplan is designed mainly to optimize the interconnect cost and delay, which is the case for many nowadays interconnect driven floorplanners. By keeping the adjacency and closeness relationship between the modules unchanged, the effects of this step on the optimization of interconnect is little while the deadspace can be reduced by allowing the modules in its neighborhood to change in shapes in a best fit way to fill up the unused area. The process of selecting modules to be non-rectangular shapes could be done automatically or could be controlled interactively by mouse-click like selections. One advantage of this technique is that the packing algorithm can be very fast compared with the full optimization. Users can immediately obtain the result at real time. This feature and technique is also useful in fixing up small and incremental changes during ECO modifications.

In this paper, we formulate the problem as a mathematical program. All the flexible modules can change in dimension under their respective area and *aspect ratio* (width to height ratio) constraints, and those lying in the neighborhood of an unused area can further change in shape (to become non-rectangular) to fill up the unused area in a best fit way. We use the Lagrangian relaxation technique [8; 15] to solve the problem. The formulation is in such a perfect way that the corresponding Lagrangian relaxation subproblem can be solved optimally, and the optimal dimensions of all the rectangular and non-rectangular modules can be computed by closed form equations in $O(m)$ time where m is the total number of edges in the constraint graphs. We tested our method using some benchmark data with 10 to 49 modules. For each data set, a preliminary floorplan is first generated using a simulated annealing technique with a goal to optimize both the interconnect cost and the total chip area. We then apply our techniques to change the modules in shapes and dimensions. Experimental results show that about 3.6% deadspace can be reduced on average.

The rest of this paper is organized as follow: We will define the problem in the next section. Section three will give an overview of our approach and our formulation of the problem as a mathematical program. We will explain in details the Lagrangian relaxation technique and the optimality conditions to help solving the problem efficiently in section four. Experimental results will be shown in section five and some remarks will be given in the last section.

2. PROBLEM DEFINITION

In this problem, we are given a preliminary floorplan design, and our goal is to give a tighter packing by changing the shapes and dimensions of the flexible modules to fill up the unused area, while keeping the original spatial relationships between the modules unchanged. A simple example is shown in Figure 1. In this example, the packing on the left is a preliminary floorplan and our goal is to obtain a tighter packing (the one on the right) by changing the shapes (e.g., module 2 and 7) and dimensions of the flexible modules. There are two kinds of input modules: *hard* modules and *soft* modules. A hard module is a module whose dimension is fixed. A soft module is one whose area is fixed but its shape and dimension can be changed as long as its aspect ratio (and the aspect ratio of its sub-blocks if there is any) is within a given range. We are given n modules of areas A_1, A_2, ..., A_n, their aspect ratio ranges $[r_1, s_1]$, $[r_2, s_2]$, ..., $[r_n, s_n]$ and their initial dimensions. (In case of a hard module, its maximum and minimum aspect ratio will be the same.) We are also given the netlist information: $net_1, net_2, ..., net_m$ and the relative positions of the I/O pins $p_1, p_2, ..., p_q$ along the boundary of the chip. For each net net_i where $1 \leq i \leq m$, we are given its weight, the I/O pin and the set of modules it is connected to.

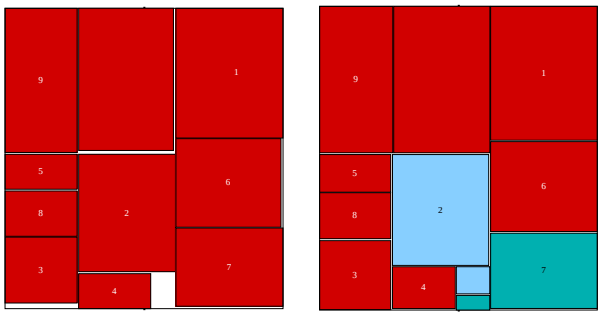


Figure 1: A simple example of changing the shapes and dimensions of the flexible modules to fill up the unused area.

A *packing* of a set of modules is a non-overlap placement of the modules. We measure the area of a packing as the area of the smallest rectangle enclosing all the modules. A *feasible packing* is a packing in which the widths and heights of all the modules and the sub-blocks are consistent with their aspect ratio constraints and their area constraints. For example, if a soft module i is L-shaped, the dimensions of its sub-blocks can be changed as long as their aspect ratios are within the given ranges and their total area is equal to A_i. A preliminary floorplan is given in the form of a pair of vertical and horizontal constraint graphs. (We can also generate a pair of constraint graphs given any slicing or non-slicing floorplan in any representation.) Our objective is to change the shapes and dimensions of the soft modules to fill up the unused area, while keeping the relative positions between the modules as described by the constraint graphs unchanged. (Notice that if a module becomes non-rectangular in shape, its spatial relationship with the other modules will be measured with respect to its *main block*, i.e., its largest sub-block.) The problem is defined as follows:

Problem Floorplanning/Sizing and Shaping (FP/SS)
Given a preliminary floorplan design in the form of a pair of horizontal and vertical constraint graphs, and a set of hard and soft modules with their initial dimensions and their areas and aspect ratio constraints, change the shapes (from rectangular to non-rectangular) and dimensions of the soft modules to produce a tighter feasible packing in which the relative positions between the modules as described by the constraint graphs are maintained.

3. AN OVERVIEW OF OUR APPROACH

We are given a preliminary floorplan of a set R of n modules M_1, M_2, ..., M_n with areas A_1, A_2, ..., A_n respectively. For each module $M_i \in R$, the minimum and maximum aspect ratios are r_i and s_i respectively. The preliminary floorplan is given as a pair of constraint graphs G_h and G_v together with the initial dimensions of the modules. From this information, we can determine the packing, the positions of the unused area and the positions of the modules. We will then select some soft modules lying in the neighborhood of an unused area into a set S. These selected modules are "eligible" to become non-rectangular in shape. An example is shown in Figure 2. In this example, module A and B are selected, and they can be changed to non-rectangular in shape to fill up the unused space (Figure 2(b)). After this selection step, every module in the set S will have one additional sub-block of variable size. The constraint graphs G_h and G_v will also be updated (become G'_h and G'_v) to include these new sub-blocks. New edges will be added to the constraint graphs to restrict the positions of these sub-blocks so that they will fill up the unused area and will be abut with their corresponding main blocks. Figure 3 shows the changes made to the constraint graphs for the example in Figure 2. Notice that every selected module $M_k \in S$ will have one sub-block M'_k but the area of the sub-block may become zero at the end and the selected module will then remain rectangular if this can optimize the packing better.

After selecting a set of modules into S and modifying the constraint

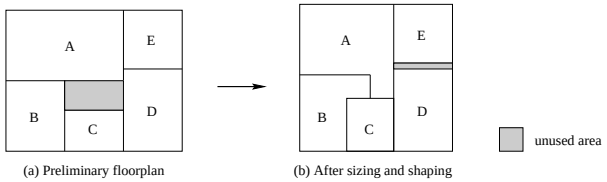


Figure 2: Module A and B are selected and they can change to non-rectangular in shape to fill up the unused area.

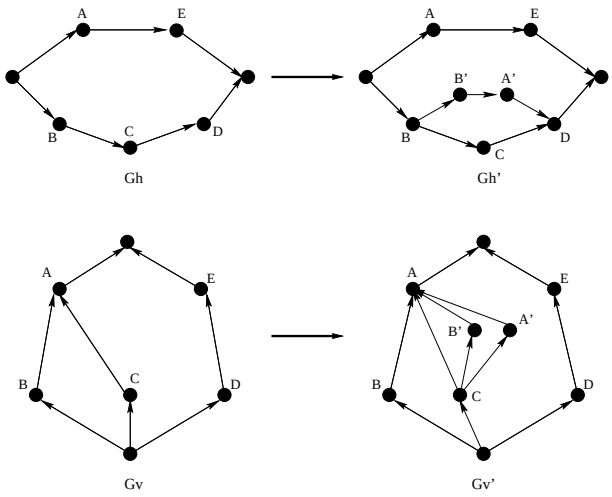


Figure 3: Modify the constraint graphs to include the new sub-blocks and their associated edges.

graphs, we can treat the sub-blocks as individual soft modules. (They will be automatically abut with their main blocks because of the new constraint edges added into the constraint graphs.) Let the size of S be p, i.e., p modules are selected to be possibly changed to non-rectangular in shape. W.l.o.g., we can assume that module $M_1, M_2, ..., M_p$ are in S and their corresponding sub-blocks are $M_{n+1}, M_{n+2}, ..., M_{n+p}$. Let S' denote this set of sub-blocks. Now, we have a new set R' of total $n' = n + p$ modules $M_1, M_2, ..., M_{n'}$. Consider the packing topology described by the constraint graphs G'_h and G'_v. Let x_i denote the smallest x position of the lower left corner of module M_i satisfying all the horizontal constraints in the horizontal constraint graph G'_h. Similarly, y_i denotes the smallest y position of the lower left corner of module M_i satisfying all the vertical constraints in the vertical constraint graph G'_v. Then for each edge $e(i, j)$ from M_i to M_j in G'_h, we have the following constraint:

$$x_i + w_i \leq x_j$$

where w_i is the width of M_i. Similarly, for each edge $e(i, j)$ from M_i to M_j in G'_v, we have the following constraint:

$$y_i + h_i \leq y_j$$

For each module $M_i \in R - S$, i.e., a rectangular module, the following relationship between w_i and h_i holds:

$$h_i = A_i / w_i$$

For each module $M_i \in S$, i.e., a non-rectangular module, we have a constraint on the total area of M_i and its sub-block M_{n+i}:

$$w_i h_i + w_{n+i} h_{n+i} = A_i$$

In the horizontal constraint graph G'_h, we denote the set of sources and sinks by s_h and t_h respectively where a source is a vertex without any in-coming edge and a sink is a vertex without any out-going edge. Similarly, we use s_v and t_v to denote the set of sources and sinks in G'_v respectively. Then for each M_i in s_h:

$$x_i = 0;$$

and for each M_i in s_v:

$$y_i = 0;$$

For simplicity, we add one dummy vertex labeled $n'+1$ to each of G'_h and G'_v. The dummy vertex in G'_h and G'_v represents the rightmost and the topmost boundary of the chip respectively. Edge $e(i, n'+1)$ with weight w_i is added to G'_h for each $M_i \in t_h$ because the rightmost chip boundary should be at a distance of at least w_i from each module $M_i \in t_h$. Similarly, $e(i, n'+1)$ with weight h_i is added to G'_v for each $M_i \in t_v$. From now onwards, we assume that the constraint graphs G'_h and G'_v contain these additional vertices and edges. The problem can be formulated as the following mathematical program PP (Primal Problem):

Minimize	$x_{n'+1} y_{n'+1}$		
Subject to	$x_i + w_i \le x_j$	$\forall\, e(i,j) \in G'_h$	(A)
	$y_i + A_i/w_i \le y_j$	$\forall\, e(i,j) \in G'_v \cap M_i \in R - S$	(B)
	$y_i + h_i \le y_j$	$\forall\, e(i,j) \in G'_v \cap M_i \in S + S'$	(C)
	$w_i h_i + w_{n+i} h_{n+i} = A_i$	$\forall\, 1 \le i \le p$	(D)
	$r_i h_i \le w_i$	$\forall\, 1 \le i \le n+p$	(E)
	$w_i \le s_i h_i$	$\forall\, 1 \le i \le n+p$	(F)

4. LAGRANGIAN RELAXATION

We will apply the Lagrangian relaxation technique [8] to solve the primal problem PP. According to the Lagrangian relaxation procedure, we can introduce multipliers, called Lagrange multipliers, to the constraints and move the constraints into the objective function. Let $\lambda_{i,j}$ denotes the multiplier for the constraint $x_i + w_i \le x_j$ in (A), and $\mu_{i,j}$ denotes the multiplier for the constraint $y_i + A_i/w_i \le y_j$ in (B) or the constraint $y_i + h_i \le y_j$ in (C) depending on the value of i.

Let $\vec{\lambda}$ and $\vec{\mu}$ be vectors of all the Lagrange multipliers introduced into the constraints. Then the Lagrangian relaxation subproblem associated with the multiplier, denoted by $LRS/(\vec{\lambda}, \vec{\mu})$, becomes:

Minimize	$x_{n'+1} y_{n'+1} +$ $\sum_{e(i,j)\in G'_h} \lambda_{i,j}(x_i + w_i - x_j) +$ $\sum_{(e(i,j)\in G'_v)\cap(M_i\in R-S)} \mu_{i,j}(y_i + \frac{A_i}{w_i} - y_j) +$ $\sum_{(e(i,j)\in G'_v)\cap(M_i\in S+S')} \mu_{i,j}(y_i + h_i - y_j)$	
Subject to	$w_i h_i + w_{n+i} h_{n+i} = A_i$	$\forall\, 1 \le i \le p$
	$r_i h_i \le w_i$	$\forall\, 1 \le i \le n+p$
	$w_i \le s_i h_i$	$\forall\, 1 \le i \le n+p$

Let $Q(\vec{\lambda}, \vec{\mu})$ denote the optimal value of the problem $LRS/(\vec{\lambda}, \vec{\mu})$. We define the Lagrangian dual problem LDP of PP as follows:

Maximize	$Q(\vec{\lambda}, \vec{\mu})$
Subject to	$\vec{\lambda} \ge 0$ and $\vec{\mu} \ge 0$

Now we are going to use the Lagrangian relaxation technique to solve the primal problem.

4.1 Simplification of the Lagrangian Relaxation Subproblem

The Lagrangian relaxation subprogram $LRS/(\vec{\lambda}, \vec{\mu})$ can be greatly simplified by the Kuhn-Tucker conditions [8; 15]. Consider the Lagrangian ζ of PP [8]:

$$\begin{aligned}\zeta &= x_{n'+1} y_{n'+1} + \sum_{e(i,j)\in G'_h} \lambda_{i,j}(x_i + w_i - x_j) \\ &+ \sum_{(e(i,j)\in G'_v)\cap(M_i\in R-S)} \mu_{i,j}(y_i + A_i/w_i - y_j) \\ &+ \sum_{(e(i,j)\in G'_v)\cap(M_i\in S+S')} \mu_{i,j}(y_i + h_i - y_j) \\ &+ terms\ independent\ of\ x'_i s\ and\ y'_i s\end{aligned}$$

The Kuhn-Tucker conditions imply that $\partial\zeta/\partial x_i = 0$ and $\partial\zeta/\partial y_i = 0$ for all $1 \le i \le n'+1$ at the optimal solution of PP. Therefore, in searching for the multipliers to optimize LDP, we only need to consider those multipliers such that $\partial\zeta/\partial x_i = 0$ and $\partial\zeta/\partial y_i = 0$ hold for all $1 \le i \le n'$. We obtain the following conditions by rearranging the terms in ζ and taking derivatives:

$$\sum_{e(j,i)\in G'_h} \lambda_{j,i} = \sum_{e(i,j)\in G'_h} \lambda_{i,j} \quad (1)$$

$$\sum_{e(j,i)\in G'_v} \mu_{j,i} = \sum_{e(i,j)\in G'_v} \mu_{i,j} \quad (2)$$

for all $1 \le i \le n'$, and

$$y_{n'+1} = \sum_{e(i,n'+1)\in G'_h} \lambda_{i,n'+1} \quad (3)$$

$$x_{n'+1} = \sum_{e(i,n'+1)\in G'_v} \mu_{i,n'+1} \quad (4)$$

We use Ω to denote the set of $(\vec{\lambda}, \vec{\mu})$ satisfying the above relationships (1) - (4) for the given pair of horizontal and vertical constraint graphs G'_h and G'_v. If $(\vec{\lambda}, \vec{\mu}) \in \Omega$, the objective function F of $LRS/(\vec{\lambda}, \vec{\mu})$ becomes:

$$\begin{aligned}F &= k + \sum_{p+1\le i\le n} ((\sum_{e(i,j)\in G'_h} \lambda_{i,j}) w_i + (\sum_{e(i,j)\in G'_v} \mu_{i,j}) A_i/w_i) \\ &+ \sum_{1\le i\le p} ((\sum_{e(i,j)\in G'_h} \lambda_{i,j}) w_i + (\sum_{e(i,j)\in G'_v} \mu_{i,j}) h_i) \\ &+ \sum_{n+1\le i\le n+p} ((\sum_{e(i,j)\in G'_h} \lambda_{i,j}) w_i + (\sum_{e(i,j)\in G'_v} \mu_{i,j}) h_i)\end{aligned}$$

where $k = -(\sum_{e(i,n'+1)\in G'_h} \lambda_{i,n'+1})(\sum_{e(i,n'+1)\in G'_v} \mu_{i,n'+1})$ is a constant for a fixed pair of $\vec{\lambda}$ and $\vec{\mu}$.

4.2 Optimality Condition for Rectangular Blocks

Consider a module M_i where $p+1 \le i \le n$, i.e., a rectangular module. We can differentiate F with respect to w_i in order to get the optimal value of w_i to minimize F:

$$\frac{\partial F}{\partial w_i} = 0$$

$$w_i = \sqrt{\frac{A_i \times \sum_{e(i,j)\in G'_v} \mu_{i,j}}{\sum_{e(i,j)\in G'_h} \lambda_{i,j}}}$$

Recall that w_i must lie within the range $[L_i, U_i]$ where $L_i = \sqrt{A_i/s_i}$ and $U_i = \sqrt{A_i/r_i}$. Let w_i^* denote $\sqrt{\frac{A_i \times \sum_{e(i,j)\in G'_v} \mu_{i,j}}{\sum_{e(i,j)\in G'_h} \lambda_{i,j}}}$. Since $\partial F/\partial w_i$ is positive for $w_i < w_i^*$ and negative for $w_i > w_i^*$, the optimal w_i can be computed as:

$$w_i = \min\{U_i, \max\{L_i, w_i^*\}\}$$

4.3 Optimality Conditions for Rectilinear Modules

Let λ_i, μ_i, λ_{n+i} and μ_{n+i} denote $\sum_{e(i,j)\in G'_h} \lambda_{i,j}$, $\sum_{e(i,j)\in G'_v} \mu_{i,j}$, $\sum_{e(n+i,j)\in G'_h} \lambda_{n+i,j}$ and $\sum_{e(n+i,j)\in G'_v} \mu_{n+i,j}$ respectively. Consider a module M_i where $1 \le i \le p$, i.e., a module which can possibly become non-rectangular. According to the Kuhn-Tucker conditions [8], the first-order optimality conditions for $LRS/(\vec{\lambda}, \vec{\mu})$ are as follows:

$$\lambda_i + \sigma_i h_i - \alpha_i + \beta_i = 0 \quad (5)$$
$$\mu_i + \sigma_i w_i + r_i\alpha_i - s_i\beta_i = 0 \quad (6)$$
$$\lambda_{n+i} + \sigma_i h_{n+i} - \alpha_{n+i} + \beta_{n+i} = 0 \quad (7)$$
$$\mu_{n+i} + \sigma_i w_{n+i} + r_{n+i}\alpha_{n+i} - s_{n+i}\beta_{n+i} = 0 \quad (8)$$
$$\alpha_i(r_i h_i - w_i) = 0 \quad (9)$$
$$\beta_i(w_i - s_i h_i) = 0 \quad (10)$$
$$\alpha_{n+i}(r_{n+i}h_{n+i} - w_{n+i}) = 0 \quad (11)$$
$$\beta_{n+i}(w_{n+i} - s_{n+i}h_{n+i}) = 0 \quad (12)$$
$$w_i h_i + w_{n+i}h_{n+i} = A_i \quad (13)$$

where $\sigma_i \in \Re$, $\alpha_i \ge 0$, $\beta_i \ge 0$, $\alpha_{n+i} \ge 0$ and $\beta_{n+i} \ge 0$. For a given pair of $\vec{\lambda}$ and $\vec{\mu}$, λ_i, μ_i, λ_{n+i} and μ_{n+i} are known. Therefore, we need to solve a system Γ of nine non-linear equations with nine unknowns (h_i, w_i, h_{n+i}, w_{n+i}, σ_i, α_i, β_i, α_{n+i} and β_{n+i}). Fortunately, they can still be solved by closed form equations according to the following theorem. Because of the limitation in space, we will only give a brief outline of its proof here.

Theorem 1 The above system of eqations Γ can be solved by closed form equations.

Proof There are 3 cases for the values of h_i and w_i according to the values of α_i and β_i:

Case 1 $\alpha_i = 0$ and $\beta_i = 0$. This case occurs when $r_i \le \frac{\mu_i}{\lambda_i} \le s_i$. From equation (5) and (6), we can deduce that:

$$h_i = \frac{-\lambda_i}{\sigma_i} \quad w_i = \frac{-\mu_i}{\sigma_i}$$

Case 2 $\alpha_i \ne 0$ and $\beta_i = 0$. This case occurs when $\frac{\mu_i}{\lambda_i} \le r_i$. From equation (5), (6) and (9), we can deduce that:

$$h_i = \frac{-\lambda_i r_i - \mu_i}{2\sigma_i r_i} \quad w_i = \frac{-\lambda_i r_i - \mu_i}{2\sigma_i}$$

Case 3 $\alpha_i = 0$ and $\beta_i \ne 0$. This case occurs when $\frac{\mu_i}{\lambda_i} \ge s_i$. From equation (5), (6) and (10), we can deduce that:

$$h_i = \frac{-\lambda_i s_i - \mu_i}{2\sigma_i s_i} \quad w_i = \frac{-\lambda_i s_i - \mu_i}{2\sigma_i}$$

Note that the case that $\alpha_i \ne 0$ and $\beta_i \ne 0$. is impossible since equation (9) and (10) cannot be satisfied simultaneously. Similarly, we can write w_{n+i} and h_{n+i} in terms of λ_{n+i}, μ_{n+i}, r_{n+i}, s_{n+i} and σ_i according to the values of α_{n+i} and β_{n+i}:

Case 1 $\alpha_{n+i} = 0$ and $\beta_{n+i} = 0$. This case occurs when $r_{n+i} \le \frac{\mu_{n+i}}{\lambda_{n+i}} \le s_{n+i}$. From equation (7) and (8), we can deduce that:

$$h_{n+i} = \frac{-\lambda_{n+i}}{\sigma_i} \quad w_{n+i} = \frac{-\mu_{n+i}}{\sigma_i}$$

Case 2 $\alpha_{n+i} \ne 0$ and $\beta_{n+i} = 0$. This case occurs when $\frac{\mu_{n+i}}{\lambda_{n+i}} \le r_{n+i}$. From equation (7), (8) and (11), we can deduce that:

$$h_{n+i} = \frac{-\lambda_{n+i}r_{n+i} - \mu_{n+i}}{2\sigma_i r_{n+i}} \quad w_{n+i} = \frac{-\lambda_{n+i}r_{n+i} - \mu_{n+i}}{2\sigma_i}$$

Case 3 $\alpha_{n+i} = 0$ and $\beta_{n+i} \ne 0$. This case occurs when $\frac{\mu_{n+i}}{\lambda_{n+i}} \ge s_{n+i}$. From equation (7), (8) and (12), we can deduce that:

$$h_{n+i} = \frac{-\lambda_{n+i}s_{n+i} - \mu_{n+i}}{2\sigma_i s_{n+i}} \quad w_{n+i} = \frac{-\lambda_{n+i}s_{n+i} - \mu_{n+i}}{2\sigma_i}$$

Similarly, the case that $\alpha_{n+i} \ne 0$ and $\beta_{n+i} \ne 0$ is impossible since equation (11) and (12) cannot be satisfied simultaneously. Therefore, in any combination of the above cases, we can write h_i, w_i, h_{n+1} and w_{n+1} in terms of σ_i. (Note that λ_i, λ_{n+i}, μ_i, μ_{n+i}, r_i, s_i, r_{n+i} and s_{n+i} are known.) Then we can substitute these expressions into equation (13) and compute σ_i. Finally, we will substitute back the value of σ_i into the expressions for h_i, w_i, h_{n+1} and w_{n+1} and compute their values. □

4.4 Solving LRS

The algorithm *LRS* below outlines the steps to solve the Lagrangian subproblem $LRS/(\vec{\lambda}, \vec{\mu})$ given a pair of $\vec{\lambda}$ and $\vec{\mu}$ satisfying the optimality condition.

Algorithm *LRS*
/* This algorithm solves $LRS/(\vec{\lambda}, \vec{\mu})$
optimally given $(\vec{\lambda}, \vec{\mu}) \in \Omega$ */
Input: Areas A_1, A_2, ..., A_n
Lower bounds on aspect ratios r_1, r_2, ..., $r_{n'}$
Upper bounds of aspect ratios s_1, s_2, ..., $s_{n'}$
Constraint graphs G'_v and G'_h
Lagrange multipliers $(\vec{\lambda}, \vec{\mu}) \in \Omega$
Output: w_1, w_2, ..., $w_{n'}$, h_1, h_2, ..., $h_{n'}$
1. For $i = p + 1$ *to* n
2. Compute $L_i = \sqrt{A_i/s_i}$ and $U_i = \sqrt{A_i/r_i}$
3. Compute $sum_\lambda = \sum_{e(i,j)\in G'_h} \lambda_{i,j}$
4. Compute $sum_\mu = \sum_{e(i,j)\in G'_v} \mu_{i,j}$
5. If $(sum_\lambda \ne 0)$ and $(sum_\mu/sum_\lambda \ge 0)$
6. Compute $w^* = \sqrt{A_i * sum_\mu/sum_\lambda}$
7. $w_i = \min\{U_i, \max\{L_i, w^*\}\}$ $h_i = A_i/w_i$
8. For $i = 1$ *to* p
9. Compute $sum_{\lambda_1} = \sum_{e(i,j)\in G'_h} \lambda_{i,j}$
10. Compute $sum_{\mu_1} = \sum_{e(i,j)\in G'_v} \mu_{i,j}$

11. Compute $sum_{\lambda_2} = \sum_{e(n+i,j) \in G'_h} \lambda_{n+i,j}$
12. Compute $sum_{\mu_2} = \sum_{e(n+i,j) \in G'_v} \mu_{n+i,j}$
13. Compute σ_i, h_i, w_i, h_{n+i} and w_{n+i} from the values of sum_{μ_1}, sum_{λ_1}, sum_{μ_2} and sum_{λ_2} according to Theorem 1 of Section 4.3.

4.5 Solving LDP

As explained above, we only need to consider those $(\vec{\lambda}, \vec{\mu}) \in \Omega$ in order to maximize $Q(\vec{\lambda}, \vec{\mu})$ in the LDP problem. We used a subgradient optimization method to search for these optimal pairs of $\vec{\lambda}$ and $\vec{\mu}$. Starting from an arbitrary $(\vec{\lambda}, \vec{\mu}) \in \Omega$ in step k, we will move to a new pair $(\vec{\lambda'}, \vec{\mu'})$ by following the subgradient direction:

$$\begin{aligned} \lambda'_{i,j} &= [\lambda_{i,j} + \rho_k(x_i + w_i - x_j)]^+ \\ \mu'_{i,j} &= [\mu_{i,j} + \rho_k(y_i + \frac{A_i}{w_i} - y_j)]^+ \end{aligned}$$

where

$$[x]^+ = \begin{cases} x & \text{if } x > 0, \\ 0 & \text{if } x \leq 0. \end{cases}$$

and ρ_k is a step size such that $lim_{k \to \infty} \rho_k = 0$ and $\sum_{k=1}^{\infty} \rho_k = \infty$. After updating $\vec{\lambda}$ and $\vec{\mu}$, we will *project* $(\vec{\lambda'}, \vec{\mu'})$ back to the nearest point $(\vec{\lambda^*}, \vec{\mu^*})$ in Ω using a 2-norm measure and solve the Lagrangian relaxation subproblem $LRS/(\vec{\lambda^*}, \vec{\mu^*})$ using the method described above. This procedure is repeated until the solution converges. The following algorithm summarizes the steps to solve LDP:

Algorithm *LDP*
/* This algorithm solves the LDP problem optimally. */
Input: Areas A_1, A_2, ..., A_n
Lower bounds of aspect ratios r_1, r_2, ..., $r_{n'}$
Upper bounds of aspect ratios s_1, s_2, ..., $s_{n'}$
Constraint graphs G'_v and G'_h
Output: w_1, w_2, ..., $w_{n'}$, h_1, h_2, ..., $h_{n'}$
1. Initialize $(\vec{\lambda}, \vec{\mu})$ and ρ_1
2. Project $(\vec{\lambda}, \vec{\mu})$ to $(\vec{\lambda^*}, \vec{\mu^*})$ such that $(\vec{\lambda^*}, \vec{\mu^*}) \in \Omega$
3. $k = 1$
4. $(\vec{\lambda}, \vec{\mu}) = (\vec{\lambda^*}, \vec{\mu^*})$
5. Repeat
6. Call *LRS()* for this pair of λ and μ
7. Compute (x_i, y_i) $\forall 1 \leq i \leq n' + 1$ using the longest path algorithm
8. Compute $\lambda'_{i,j} = [\lambda_{i,j} + \rho_k(x_i + w_i - x_j)]^+$ $\forall e(i,j) \in G'_h$
9. Compute $\mu'_{i,j} = [\mu_{i,j} + \rho_k(y_i + h_i - y_j)]^+$ $\forall e(i,j) \in G'_v$
10. Project $(\vec{\lambda'}, \vec{\mu'})$ to $(\vec{\lambda^*}, \vec{\mu^*})$ s.t. $(\vec{\lambda^*}, \vec{\mu^*}) \in \Omega$
11. $k = k + 1$
12. $(\vec{\lambda}, \vec{\mu}) = (\vec{\lambda^*}, \vec{\mu^*})$
13. Until w_i's converge

5. EXPERIMENTAL RESULTS

We tested our methd using four MCNC benchmarks with 10, 11, 33 and 49 modules. The size of each benchmark data is shown in Table 1. We repeated the experiment five times for each benchmark data using different seed values for the random number generator. In each experiment, we first generated a preliminary floorplan using a simulated annealing method. In this initial floorplanning step, equal weighting were given to the area term and the wirelength term in the cost function. After obtaining such a preliminary floorplan, we selected some soft modules lying in the neighborhood of the unused areas into the set S. These were the modules which might become non-rectangular in shape to fill up the unused areas. In our current implementation, the selection process was done manually. Among all the modules lying in the neighborhood of an unused area, we picked large ones which were also lying on a critical path in the constraint graphs. The constraint graphs were then modified to include the new sub-blocks of these selected modules and to restrict their positions so that they would fill up the unused area and would also be abut with their corresponding main blocks. After these pre-processing steps, we performed shaping and sizing on the modules using the Lagrangian relaxation technique described in section 4.

In all the experiments, the minimum and maximum aspect ratios of the soft modules were 0.5 and 2.0 respectively, while those for the sub-blocks were 0.1 and 10.0 respectively. We limited the aspect ratio of the final packing to be within the range of 0.9 to 1.1 inclusively. For comparison purposes, we also generated the results for the cases when no non-rectangular blocks was allowed, i.e., all the soft modules only changed in dimensions. The results were generated using a 600MHz Pentium III processor, and is shown in Table 2. The experimental results show that our shaping and sizing step is useful in improving the packing quality of the preliminary floorplan by reducing the deadspace by about 3.6% on average, while keeping the relative positions between the modules unchanged. Figure 4, 5, 6, 7 and 8 show the preliminary floorplans and the floorplans after sizing and shaping in some of the experiments.

Data Set	Number of Modules	Number of Nets
xerox	10	203
hp	11	83
ami33	33	123
ami49	49	408

Table 1: Testing data sets

Benchmark	Deadspace (%) in preliminary floorplan	Deadspace (%) after sizing only	Deadspace (%) after sizing & shaping	Time (sec)
xerox (1)	3.12	1.05	0.40	0.17
xerox (2)	3.15	0.88	0.84	0.15
xerox (3)	6.38	5.46	0.46	0.17
xerox (4)	2.10	1.18	0.71	0.15
xerox (5)	2.78	1.07	0.00	0.17
Average	3.51	1.93	0.48	0.16
hp (1)	2.80	2.42	1.58	0.17
hp (2)	4.05	2.87	1.73	0.16
hp (3)	3.89	2.94	1.62	0.17
hp (4)	6.40	4.98	1.60	0.18
hp (5)	2.21	1.18	0.47	0.17
Average	3.87	2.88	1.40	0.17
ami33 (1)	6.73	3.62	2.27	1.49
ami33 (2)	7.38	4.25	2.13	1.30
ami33 (3)	5.21	2.64	2.39	1.17
ami33 (4)	8.72	4.94	2.98	1.52
ami33 (5)	8.99	4.12	2.12	1.47
Average	7.41	3.91	2.38	1.39
ami49 (1)	6.48	4.08	2.31	5.36
ami49 (2)	7.60	5.50	5.20	3.85
ami49 (3)	10.42	7.70	5.93	4.65
ami49 (4)	7.51	5.48	4.28	5.57
ami49 (5)	13.33	11.70	9.26	6.29
Average	9.07	6.89	5.40	5.14

Table 2: Shaping and Sizing Results

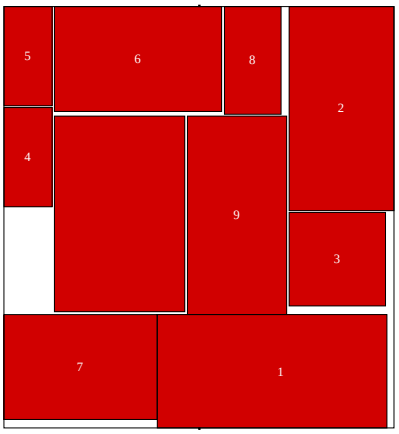
6. REMARKS

6.1 Modules with More than Two Sub-blocks

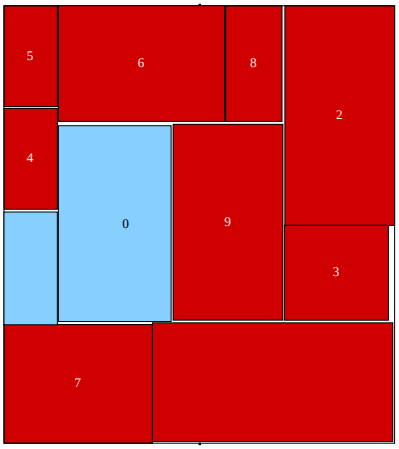
In this paper, we only handle the case when the non-rectangular modules have at most two rectangular sub-blocks. However our approach can be extended to more than two sub-blocks directly. If each non-rectangular module has up to k rectangular sub-blocks, the system of equations Γ will have $4k+1$ unknowns in $4k+1$ non-linear equations, and can still be solved by closed form equations by considering three possible cases for the size of each rectangular sub-blocks.

6.2 Module Selection

In our current implementation, the modules in the set S are selected manually. Our strategy is to pick, among all the modules in the neighborhood of an unused area, a large module which is lying on a critical path in the constraint graphs. This manual and interactive selection process gives the users a lot of flexibility and control over the final packing. However, the set of modules selected will affect the quality of the final results and an automatic selection process will have an advantage of picking the best set of modules to be re-shaped in order to obtain the tightest possible final packing.

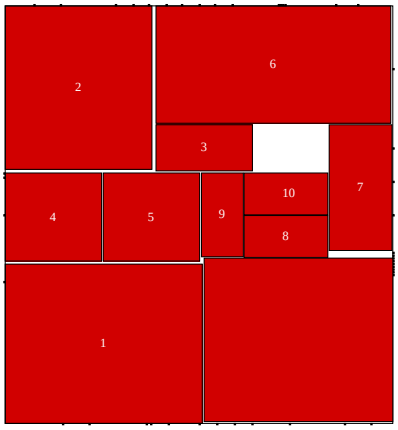


(a) Preliminary floorplan. Deadspace = 6.38%

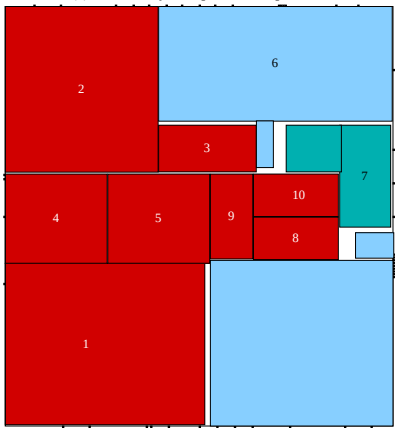


(b) After sizing and shaping. Deadspace = 0.46%

Figure 4: Data set xerox (3)



(a) Preliminary floorplan. Deadspace = 2.80%



(b) After sizing and shaping. Deadspace = 1.58%

Figure 5: Data set hp (1)

7. REFERENCES

[1] J. Dufour, R. McBride, P. Zhang, and C. K. Cheng. A Building Block Placement Tool. *IEEE Asia and South Pacific Design Automation Conference*, pages 271–276, 1997.

[2] K. Fujiyoshi and H. Murata. Arbitrary Convex and Concave Rectilinear Block Packing Using Sequence-Pair. *International Symposium on Physical Design*, pages 103–110, 1999.

[3] D.F. Wong F.Y. Young and Hannah H. Yang. On extending slicing floorplans to handle L/T-shaped modules and abutment constraints. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2000. To appear (Also appeared in ICDA 2000).

[4] M. Kang and W. W.M. Dai. General Floorplanning with L-shaped, T-shaped and Soft Blocks Based on Bounded Slicing Grid Structure. *IEEE Asia and South Pacific Design Automation Conference*, pages 265–270, 1997.

[5] M. Z.W. Kang and W. W.M. Dai. Topology Constrainted Rectilinear Block Packing. *International Symposium on Physical Design*, pages 179–186, 1998.

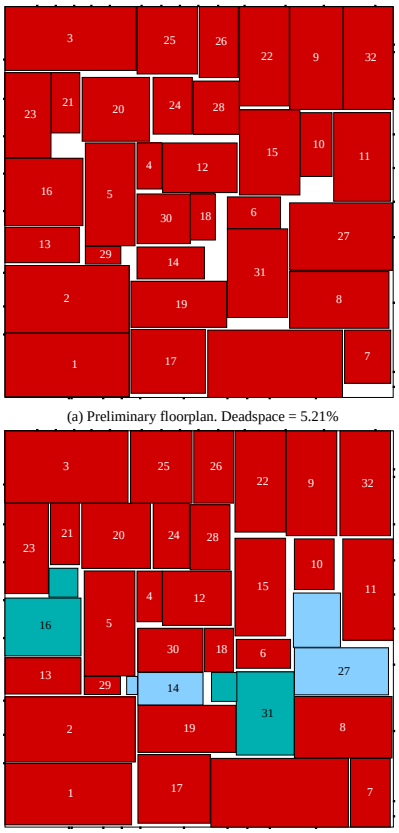
[6] Maggie Zhiwei Kang and Wayne Wei-Ming Dai. Arbitrary Rectilinear Block Packing Based on Sequence Pair. *Proceedings IEEE International Conference on Computer-Aided Design*, pages 259–266, 1998.

[7] T. Chang Lee. A Bounded 2D Contour Searching Algorithm for Floorplan Design with Arbitrarily Shaped Rectilinear and Soft Modules. *Proceedings of the 30th ACM/IEEE Design Automation Conference*, pages 525–530, 1993.

[8] M.S. Bazaraa and H.D. Sherali and C.M. Shetty. *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, Inc., second edition, 1997.

[9] S. Nakatake, K. Fujiyoushi, H. Murata, and Y. Kajitani. Module Placement on BSG-Structure and IC Layout Applications. *Proceedings IEEE International Conference on Computer-Aided Design*, pages 484–491, 1996.

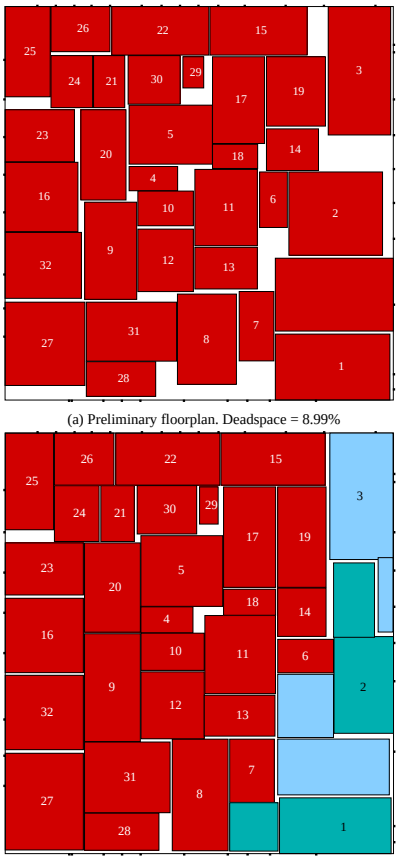
[10] S. Nakatake, M. Furuya, and Y. Kajitani. Module Placement on BSG-Structure with Preplaced Modules and Rectilinear Modules. *IEEE Asia and South Pacific Design Automation Conference*, pages 571–576, 1998.



(a) Preliminary floorplan. Deadspace = 5.21%

(b) After sizing and shaping. Deadspace = 2.39%

Figure 6: Data set ami33 (3)



(a) Preliminary floorplan. Deadspace = 8.99%

(b) After sizing and shaping. Deadspace = 2.12%

Figure 7: Data set ami33 (5)

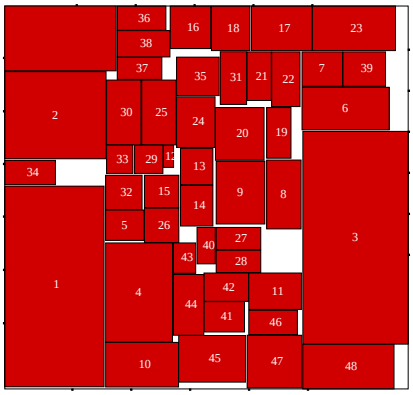
[11] Keishi Sakanushi, Shigetoshi Nakatake, and Yoji Kajitani. The Multi-BSG: Stochastic Approach to an Optimal Packing of Convex-Rectilinear Blocks. *Proceedings IEEE International Conference on Computer-Aided Design*, pages 267–274, 1998.

[12] D.F. Wong and C.L. Liu. A New Algorithm for Floorplan Design. *Proceedings of the 23rd ACM/IEEE Design Automation Conference*, pages 101–107, 1986.

[13] D.F. Wong and C.L. Liu. Floorplan Design for Rectangular and L-shaped Modules. *Proceedings IEEE International Conference on Computer-Aided Design*, pages 520–523, 1987.

[14] Jin Xu, Pei-Ning Guo, and Chung-Kuan Cheng. Rectilinear Block Placement Using Sequence-Pair. *International Symposium on Physical Design*, pages 173–178, 1998.

[15] F.Y. Young, Chris C.N. Chu, W.S. Luk, and Y.C. Wong. Floorplan Area Minimization using Lagrangian Relaxation. *International Symposium on Physical Design*, pages 174–179, 2000.



(a) Preliminary floorplan. Deadspace = 6.48%

(b) After sizing and shaping. Deadspace = 2.31%

Figure 8: Data set ami49 (1)