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

Chris C.N. Chu and Evangeline F.Y. Young

Abstract

Many previous works on floorplanning with non-rectangular modules [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12] assume that the modules are pre-designated to have particular non-rectangular shapes, e.g., L-shaped, T-shaped, etc. However, this is not common in practice because rectangular shapes are more preferable in many designing steps. Those non-rectangular shapes are actually generated during floorplanning in order to further optimize the solution. In this paper, we study this problem of changing the shapes and dimensions of the flexible modules 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 small incremental changes during ECO modifications. We formulate the problem as a mathematical program. The formulation is such that the 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. As a result, the total time for the whole shaping and sizing process is $O(k \times m)$ time where k is the number of iterations on the Lagrangian Relaxation Subproblem. Experimental results show that the amount of area re-used is 3.7% on average while the total wirelength can be reduced by 0.43% on average because of the more compacted result packing.

1 Introduction

A lot of previous works have then been reported on floorplanning with non-rectangular blocks [2, 6, 3, 4, 7, 8, 9, 5, 10, 11, 12, 13]. The papers [14, 6] extend the Polish expression representation for slicing floorplans to handle L-shaped and T-shaped modules. The works on non-slicing floorplans are mostly based on the Bounded Sliceline Grid (BSG) structure [3, 4, 7, 8, 9] or the Sequence Pair (SP) representation [5, 11, 10, 12]. Most of these works explore the rules to restrict the placement of the rectangular sub-blocks of a rectilinear module, so that these sub-blocks will be placed adjacent to one another in an appropriate way to form back its original rectilinear shape in the final packing. However, in all these previous works, it is assumed that some modules are pre-designated to have particular non-rectangular shapes, e.g., T-shaped or L-shaped, etc., but this is not common in practice since rectangular shapes are more preferable in many designing steps. They are easier to be managed not only in floorplanning, but also in downstream pin assignment, placement, routing and timing analysis. Non-rectangular shapes are often considered only when their shapes can improve the floorplan solution.

We can perform a preliminary floorplan design with all the soft-blocks in rectangular shapes at the beginning. After a preliminary floorplan is obtained based on the important criteria like interconnect delay, routing congestion and area, etc., we can allow the flexible modules to change in shapes and dimensions slightly to further improve the floorplan solution as a post-processing step, while keeping the relative spatial relationships between the modules unchanged. By keeping the adjacency and closeness relationship between the modules unchanged, the effects of this step on the original optimization in interconnect is little while the area usage can be improved by allowing the flexible modules to change in shapes in the best fit way to fill up the unused area. The total interconnect length can usually be reduced by this post-processing step because of the more compacted result packing. It is true that re-using all the empty spaces may not be good sometimes because empty spaces will be useful for buffer insertion or for other routing purposes. However, the selection of soft modules and empty spaces can be made by the users according to their needs and our method allows the users to further optimize a floorplan solution in a flexible way. This technique will also be useful in fixing small and incremental changes during ECO modifications.

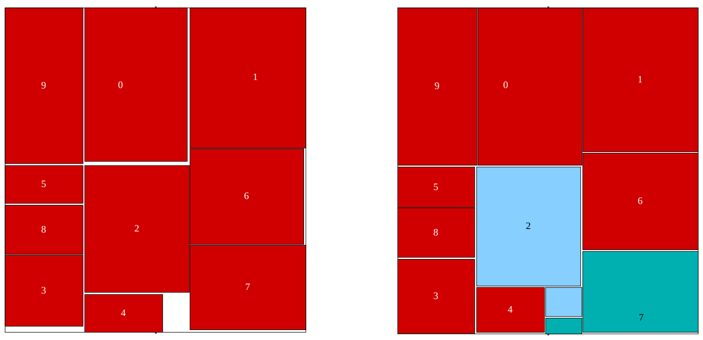
In this paper, we formulate the problem as a mathematical program on floorplanning. Moh et. al [15] formulated the floorplanning problem as a geometric program and find its global minimum using some standard convex optimization techniques. Murata et. al [16] extend the work of [15] to non-slicing floorplan and try to reduce the number of variables and functions when formulating the problem so as to improve the efficiency. However, the execution time of their method to find an exact solution is still very long, and they consider soft rectangular modules only. In our formulation, all the flexible modules can change in dimension under their area and *aspect ratio* (width to height ratio) constraints. Those lying in the neighborhood of an empty space can change in shape (to become non-rectangular) to fill up the unused area in the best fit way. We use the Lagrangian relaxation technique [17, 18] to solve the problem. The formulation is such that the 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. As a result, the total time for the whole shaping and sizing process is $O(k \times m)$ time where k is the number of iterations on the Lagrangian Relaxation Subproblem. The paper [16] has also used convex-programming to solve the floorplanning problem with soft blocks and pre-placed blocks.

We tested our method using some MCNC benchmarks. For each data set, a preliminary floorplan is first generated with an objective to optimize both the interconnect cost and total chip area. We then apply our mathematical programming technique to change the flexible modules in shapes and dimensions to fill up the empty spaces.

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 remarks and conclusion 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 change the shapes and dimensions of some flexible modules to fill up the empty spaces, while keeping the module areas constant and 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 given preliminary floorplan and our goal is to change the shapes and dimensions of the flexible modules to fill up the empty spaces. The final packing is shown on the right in which module 2 and 7 becomes L-shaped after this post-processing step. 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 bounds $[r_1, s_1]$, $[r_2, s_2]$, ..., $[r_n, s_n]$ and their initial dimensions. (In case of a hard module, its minimum and maximum 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.



(a) Preliminary floorplan. Deadspace = 2.78%

(b) After sizing and shaping. Deadspace = 0%

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

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 their sub-blocks (if there is any) satisfy their aspect ratio constraints and their area constraints. For example, if a soft module i is L-shaped, the dimensions of its two sub-blocks can be changed as long as their aspect ratios are within the given bounds 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. 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 formally as follows:

Problem Floorplanning with Shaping and Sizing (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 reduce the total area of the floorplan such that the relative positions between the modules (as described by the constraint graphs) are maintained and all the area and aspect ratio constraints are satisfied.

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 that lie in the neighborhood of some empty spaces 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 spaces (Figure 2(b)). Every module in this 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 empty spaces and will 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 design better.

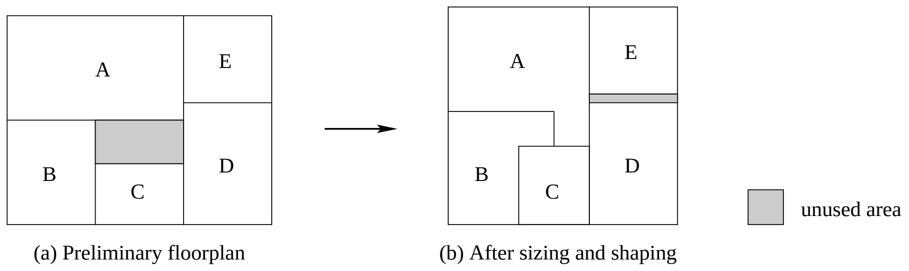


Figure 2: Module A and B are selected to be eligible to be changed into non-rectangular in shape to fill up the empty spaces.

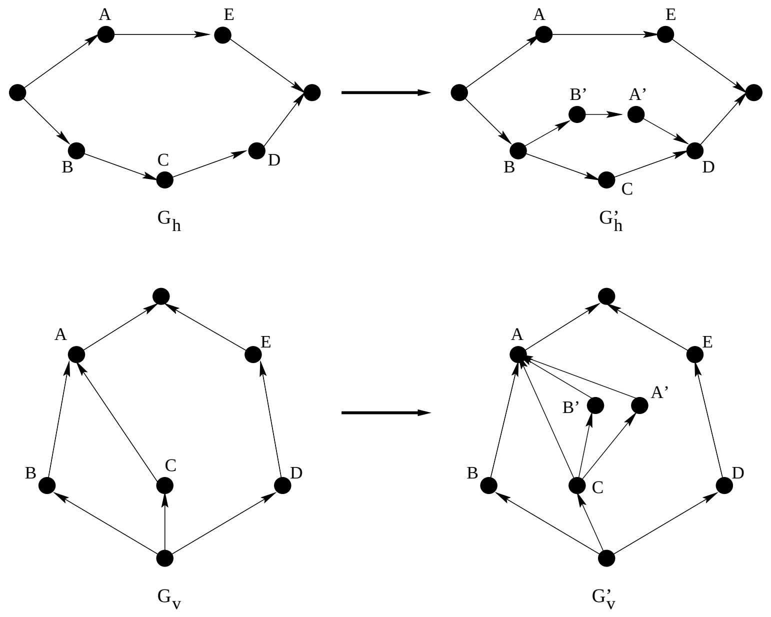


Figure 3: Modify the constraint graphs to include the new sub-blocks and their associated edges.

After selecting a set of modules into S and modifying the constraint graphs correspondingly, we can treat the sub-blocks as individual soft modules. (They will automatically abut with their main blocks because of the 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 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 and M_j in s_v:

$$\begin{aligned} x_i &= 0 \\ y_j &= 0 \end{aligned}$$

For simplicity, we add one dummy vertex labeled $n' + 1$ to each of G'_h and G'_v. The dummy vertices in G'_h and G'_v represent the rightmost and 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:	$x_i + w_i \leq x_j$	$\forall e(i, j) \in G'_h$	(A)
	$y_i + h_i \leq y_j$	$\forall e(i, j) \in G'_v$	(B)

$$\begin{array}{lll} w_i h_i = A_i & \forall p+1 \le i \le n & \text{(C)} \\ w_i h_i + w_{n+i} h_{n+i} = A_i & \forall 1 \le i \le p & \text{(D)} \\ r_i h_i \le w_i & \forall 1 \le i \le n+p & \text{(E)} \\ w_i \le s_i h_i & \forall 1 \le i \le n+p & \text{(F)} \end{array}$$

4 Solving the Problem by Lagrangian Relaxation

We will apply the Lagrangian relaxation technique [17] to solve the primal problem PP. Lagrangian relaxation is a general technique for solving constrained optimization problems. Constraints that are difficult to handle are "relaxed" and incorporated into the objective function by multiplying each constraint with a constant called Lagrange multipler. To solve the problem PP, we relax the constraints (A) and (B). 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 $y_i + h_i \le y_j$ in (B). Let $\vec{\lambda}$ and $\vec{\mu}$ be vectors of all the Lagrange multipliers introduced. Then the Lagrangian relaxation subproblem associated with the multipliers $\vec{\lambda}$ and $\vec{\mu}$, denoted by $LRS/(\vec{\lambda}, \vec{\mu})$, becomes:

$$\begin{array}{lll} \text{Minimize:} & \multicolumn{2}{l}{x_{n'+1} y_{n'+1} + \sum_{e(i,j) \in G'_h} \lambda_{i,j}(x_i + w_i - x_j) + \sum_{e(i,j) \in G'_v} \mu_{i,j}(y_i + h_i - y_j)} \\ \text{Subject to:} & w_i h_i = A_i & \forall p+1 \le i \le n \\ & w_i h_i + w_{n+i} h_{n+i} = A_i & \forall 1 \le i \le p \\ & r_i h_i \le w_i & \forall 1 \le i \le n+p \\ & w_i \le s_i h_i & \forall 1 \le i \le n+p \end{array}$$

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:

$$\begin{array}{ll} \text{Maximize:} & Q(\vec{\lambda}, \vec{\mu}) \\ \text{Subject to:} & \vec{\lambda} \ge 0 \text{ and } \vec{\mu} \ge 0 \end{array}$$

Notice that PP is a geometric program [19]. It can be transformed into a convex problem. Hence, if $\vec{\lambda}$ and $\vec{\mu}$ is the optimal solution of LDP, then the optimal solution of $LRS/(\vec{\lambda}, \vec{\mu})$ will also optimize PP [17, Theorem 6.2.4]. In the following, we will show how to solve $LRS/(\vec{\lambda}, \vec{\mu})$ and LDP optimally. In other words, PP also is solved optimally.

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 [17, 18]. Consider the Lagrangian ζ of PP [17]:

$$\begin{aligned} \zeta \quad = \quad & x_{n'+1} y_{n'+1} + \sum_{e(i,j) \in G'_h} \lambda_{i,j}(x_i + w_i - x_j) + \sum_{e(i,j) \in G'_v} \mu_{i,j}(y_i + h_i - y_j) \\ & + \text{terms independent of } {x_i}'s \text{ and } {y_i}'s \end{aligned}$$

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 $1 \leq i \leq n'+1$. 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 $1 \leq i \leq n'$, 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. When $(\vec{\lambda}, \vec{\mu}) \in \Omega$, the objective function F of $LRS/(\vec{\lambda}, \vec{\mu})$ becomes:

$$F = k + \sum_{1\leq i\leq n+p} \left((\sum_{e(i,j)\in G'_h} \lambda_{i,j})w_i + (\sum_{e(i,j)\in G'_v} \mu_{i,j})h_i \right)$$

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 $\vec{\mu}$. Let $\lambda_i = \sum_{e(i,j)\in G'_h} \lambda_{i,j}$ and $\mu_i = \sum_{e(i,j)\in G'_v} \mu_{i,j}$, for $1 \leq i \leq n+p$. Then $LRS/(\vec{\lambda}, \vec{\mu})$ can be simplified to:

$$\begin{array}{lll}
\text{Minimize:} & \sum_{1\leq i\leq n+p} (\lambda_i w_i + \mu_i h_i) & \\
\text{Subject to:} & w_i h_i = A_i & \forall p+1 \leq i \leq n \\
 & w_i h_i + w_{n+i} h_{n+i} = A_i & \forall 1 \leq i \leq p \\
 & r_i h_i \leq w_i & \forall 1 \leq i \leq n+p \\
 & w_i \leq s_i h_i & \forall 1 \leq i \leq n+p
\end{array}$$

To solve this simplified Lagrangian relaxation subproblem, we first write down its Lagrangian ξ:

$$\begin{aligned}
\xi = & \sum_{1\leq i\leq n+p} (\lambda_i w_i + \mu_i h_i) + \sum_{p+1\leq i\leq n} \theta_i (w_i h_i - A_i) + \sum_{1\leq i\leq p} \sigma_i (w_i h_i + w_{n+i} h_{n+i} - A_i) \\
& + \sum_{1\leq i\leq n+p} \alpha_i (r_i h_i - w_i) + \sum_{1\leq i\leq n+p} \beta_i (w_i - s_i h_i)
\end{aligned}$$

where $\theta_i \in \Re$, $\sigma_i \in \Re$, $\alpha_i \geq 0$ and $\beta_i \geq 0$ denote the Lagrangian multipliers for the constraints in (C), (D), (E) and (F), respectively. According to the Kuhn-Tucker conditions [17], the first-order optimality conditions for $LRS/(\vec{\lambda}, \vec{\mu})$ are as follows:

$$\begin{aligned} \partial\xi/\partial w_i &= 0 && \text{for all } 1 \le i \le n+p && (5)\\ \partial\xi/\partial h_i &= 0 && \text{for all } 1 \le i \le n+p && (6)\\ \alpha_i(r_i h_i - w_i) &= 0 && \text{for all } 1 \le i \le n+p && (7)\\ \beta_i(w_i - s_i h_i) &= 0 && \text{for all } 1 \le i \le n+p && (8)\\ w_i h_i &= A_i && \text{for all } p+1 \le i \le n && (9)\\ w_i h_i + w_{n+i} h_{n+i} &= A_i && \text{for all } 1 \le i \le p && (10) \end{aligned}$$

4.2 Optimality Condition for Rectangular Blocks

Consider a module M_i where $p+1 \le i \le n$, i.e., a rectangular module. The first-order optimality conditions for this module are:

$$\lambda_i + \theta_i h_i - \alpha_i + \beta_i = 0 \quad (11)$$
$$\mu_i + \theta_i w_i + r_i\alpha_i - s_i\beta_i = 0 \quad (12)$$
$$\alpha_i(r_i h_i - w_i) = 0 \quad (13)$$
$$\beta_i(w_i - s_i h_i) = 0 \quad (14)$$
$$w_i h_i = A_i \quad (15)$$

where (11) and (12) follow from (5) and (6), respectively.

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 $\beta_i = 0$. From (11), $h_i = -\lambda_i/\theta_i$. From (12), $w_i = -\mu_i/\theta_i$. Eliminating θ_i, $h_i = w_i\lambda_i/\mu_i$. Substituting h_i into (15), $w_i^2\lambda_i/\mu_i = A_i$. Therefore, $w_i = \sqrt{A_i\mu_i/\lambda_i}$.

Case 2: $\alpha_i \ne 0$ and $\beta_i = 0$. Equation (13) implies that $w_i = r_i h_i$. Substituting h_i into (15), $w_i = \sqrt{A_i r_i}$.

Case 3: $\alpha_i = 0$ and $\beta_i \ne 0$. Equation (14) implies that $w_i = s_i h_i$. Substituting h_i into (15), $w_i = \sqrt{A_i s_i}$.

Note that the case $\alpha_i \ne 0$ and $\beta_i \ne 0$ is impossible since equation (13) and (14) cannot be satisfied simultaneously.

By combining the three cases, it is not difficult to see that

$$w_i = \min\{\sqrt{A_i s_i}, \max\{\sqrt{A_i r_i}, \sqrt{A_i\mu_i/\lambda_i}\}\}$$

Once w_i is found, h_i is given by A_i/w_i.

4.3 Optimality Conditions for Non-Rectangular Modules

Consider a module M_i where $1 \le i \le p$, i.e., a module that can possibly become non-rectangular in shape. The first-order optimality conditions for this module and its sub-block M_{n+i} are:

$$\lambda_i + \sigma_i h_i - \alpha_i + \beta_i = 0 \quad (16)$$
$$\mu_i + \sigma_i w_i + r_i\alpha_i - s_i\beta_i = 0 \quad (17)$$

$$\lambda_{n+i} + \sigma_i h_{n+i} - \alpha_{n+i} + \beta_{n+i} = 0 \tag{18}$$
$$\mu_{n+i} + \sigma_i w_{n+i} + r_{n+i}\alpha_{n+i} - s_{n+i}\beta_{n+i} = 0 \tag{19}$$
$$\alpha_i(r_i h_i - w_i) = 0 \tag{20}$$
$$\beta_i(w_i - s_i h_i) = 0 \tag{21}$$
$$\alpha_{n+i}(r_{n+i}h_{n+i} - w_{n+i}) = 0 \tag{22}$$
$$\beta_{n+i}(w_{n+i} - s_{n+i}h_{n+i}) = 0 \tag{23}$$
$$w_i h_i + w_{n+i}h_{n+i} = A_i \tag{24}$$

where (16) and (17) follow from (5) and (6), respectively. 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 be solved by closed form equations as described in the following.

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 $r_i \leq \frac{\mu_i}{\lambda_i} \leq s_i$. From equation (16) and (17),

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

Case 2: $\alpha_i \neq 0$ and $\beta_i = 0$. This case occurs when $\frac{\mu_i}{\lambda_i} \leq r_i$. From equation (16), (17) and (20),

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

Case 3: $\alpha_i = 0$ and $\beta_i \neq 0$. This case occurs when $\frac{\mu_i}{\lambda_i} \geq s_i$. From equation (16), (17) and (21),

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

Note that the case that $\alpha_i \neq 0$ and $\beta_i \neq 0$ is impossible since equation (20) and (21) 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 $r_{n+i} \leq \frac{\mu_{n+i}}{\lambda_{n+i}} \leq s_{n+i}$. From equation (18) and (19),

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

Case 2: $\alpha_{n+i} \neq 0$ and $\beta_{n+i} = 0$. This case occurs when $\frac{\mu_{n+i}}{\lambda_{n+i}} \leq r_{n+i}$. From equation (18), (19) and (22),

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

Case 3: $\alpha_{n+i} = 0$ and $\beta_{n+i} \neq 0$. This case occurs when $\frac{\mu_{n+i}}{\lambda_{n+i}} \geq s_{n+i}$. From equation (18), (19) and (23),

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

Similarly, the case that $\alpha_{n+i} \neq 0$ and $\beta_{n+i} \neq 0$ is impossible since equation (22) and (23) cannot be satisfied simultaneously. Therefore, in any combination of the above cases, we can write h_i, w_i, h_{n+i} and w_{n+i} 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.) We can substitute these expressions into equation (24) and solve σ_i. Finally, we will substitute back the value of σ_i into the expressions for h_i, w_i, h_{n+i} and w_{n+i} and compute their values.

4.4 Solving LRS

The algorithm *LRS* below outlines the steps to solve the Lagrangian relaxation 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 a pair of $(\vec{\lambda}, \vec{\mu}) \in \Omega$ */
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 $L_i = \sqrt{A_i r_i}$ and $U_i = \sqrt{A_i s_i}$
3.         Compute $\lambda_i = \sum_{e(i,j) \in G'_h} \lambda_{i,j}$
4.         Compute $\mu_i = \sum_{e(i,j) \in G'_v} \mu_{i,j}$
5.         If $(\lambda_i \neq 0)$ and $(\mu_i / \lambda_i \geq 0)$
6.              Compute $w^* = \sqrt{A_i \mu_i / \lambda_i}$
7.              $w_i = \min\{U_i, \max\{L_i, w^*\}\},\ h_i = A_i / w_i$
8.    For $i = 1$ to $p$
9.         Compute $\lambda_i = \sum_{e(i,j) \in G'_h} \lambda_{i,j}$
10.        Compute $\mu_i = \sum_{e(i,j) \in G'_v} \mu_{i,j}$
11.        Compute $\lambda_{n+i} = \sum_{e(n+i,j) \in G'_h} \lambda_{n+i,j}$
12.        Compute $\mu_{n+i} = \sum_{e(n+i,j) \in G'_v} \mu_{n+i,j}$
13.        Compute $\sigma_i$, $h_i$, $w_i$, $h_{n+i}$ and $w_{n+i}$ from the values of $\lambda_i$, $\mu_i$, $\lambda_{n+i}$ and $\mu_{n+i}$
           according to the cases described in 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:

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

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 $lim_{k\rightarrow\infty}\rho_k = 0$ 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 $LRS/(\vec{\lambda^*}, \vec{\mu^*})$ again using the algorithm *LRS*. These steps are repeated until the solution converges. The following algorithm summarizes the steps to solve the LDP problem:

```
Algorithm LDP
/* This algorithm solves the $LDP$ problem optimally. */
Input:   Areas $A_1, A_2, ..., A_n$
         Lower bounds on aspect ratios $r_1, r_2, ..., r_{n'}$
         Upper bounds on 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 $\rho_1$
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() with $(\vec{\lambda}, \vec{\mu})$
7.       Compute $(x_i, y_i)$ $\forall 1 \leq i \leq n' + 1$ from $G'_v$ and $G'_h$ 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 $h_i$ and $w_i$ $\forall 1 \leq i \leq n' + 1$ converge
```

5 Experimental Results

We tested our method using the MCNC benchmarks. The size of each benchmark data is shown in Table 1. 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

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 (%)	Area Re-used (%)	Change in Total Wirelength (%)	Time (sec)
xerox	3.51	3.01	-0.87	0.16
hp	3.87	2.55	-0.88	0.17
ami33	7.41	5.01	-0.97	1.44
ami49	9.07	4.08	+1.00	4.88

Each result is obtained by taking the average of running an experiment five times.

Table 2: Shaping and sizing results

function, where the wirelength is computed using the half-perimeter estimation method assumming that the pins are located at the center of the module. After obtaining a preliminary floorplan, we selected some soft modules lying in the neighborhood of some large empty spaces into the set S. These were the modules that would possibly become non-rectangular in shape to fill up the spaces. In our current implementation, we would select those modules which were also lying on the critical paths of the constraint graphs. The constraint graphs were then modified to include the new sub-blocks of the selected modules and to restrict their positions so that they would fill up the empty spaces and abut with their corresponding main blocks. After these pre-processing steps, we performed shaping and sizing on the modules using the Lagrangian relaxation technique as described in section 4.

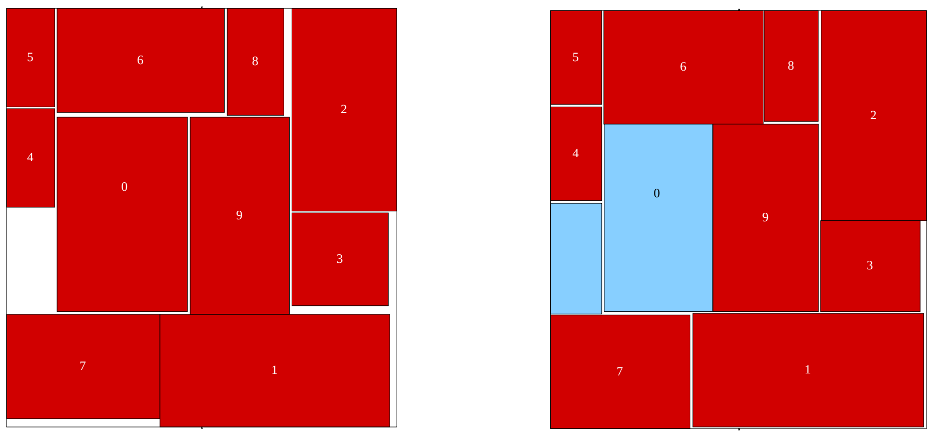
In all the experiments, the minimum and maximum aspect ratios of the soft modules were 0.5 and 2.0 respectively, while those for the sub-blocks were 0.1 and 10.0 respectively. We limited the aspect ratio of the final packing to the range of 0.9 to 1.1. All the results were generated using a 600MHz Pentium III processor, and were shown in Table 2. Experimental results show that our shaping and sizing technique is useful in re-using empty spaces by changing some soft modules to non-rectangular in shape, while keeping the relative positions between the modules unchanged. We can see that the total wirelength is reduced on average because of the more compacted result packing. Notice that after shaping and sizing, the pins of an L-shaped module are assummed to be located at the center of the major sub-block. Figure 4, 5, 6, 7 and 8 show the preliminary floorplans and the floorplans after shaping and sizing for some of the experiments.

6 Remarks

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-block.

7 Conclusion

We presented an efficient method to post-process a floorplan solution to further optimize its area usage by changing some soft modules to non-rectangular in shape to fill up the empty spaces. The total wirelength can also be reduced because of the more compacted result packing. This technique will also be useful in fixing small incremental changes during ECO modifications. Our approach is based on an elegant closed-form solutions to a mathematical program using the Lagrangian relaxation technique. Experimental results on the MCNC benchmarks have demonstrated its effectiveness in post-processing a floorplan solution in a very flexible way.



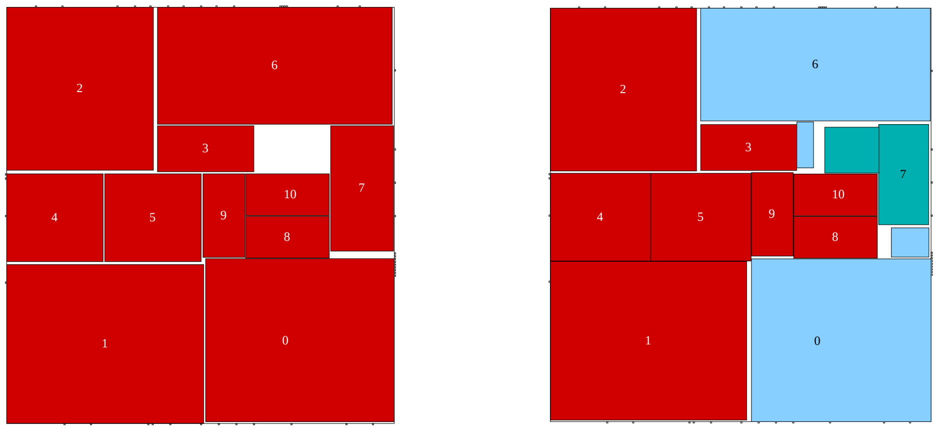
(a) Preliminary floorplan. Deadspace = 6.38% (b) After sizing and shaping. Deadspace = 0.60%

Figure 4: Data set xerox (3)

References

[1] 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.

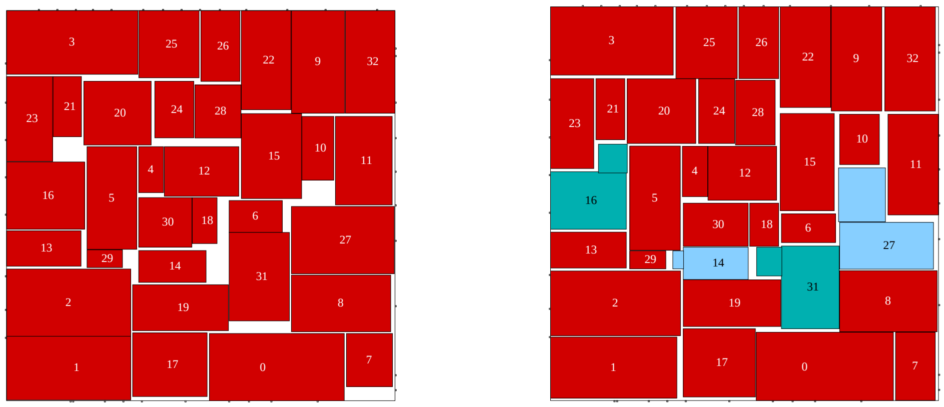
[2] D.F. Wong and C.L. Liu. A New Algorithm for Floorplan Design. *Proceedings of the 23rd ACM/IEEE Design Automation Conference*, pages 101–107, 1986.



(a) Preliminary floorplan. Deadspace = 2.80%

(b) After sizing and shaping. Deadspace = 1.50%

Figure 5: Data set hp (1)



(a) Preliminary floorplan. Deadspace = 5.21%

(b) After sizing and shaping. Deadspace = 2.40%

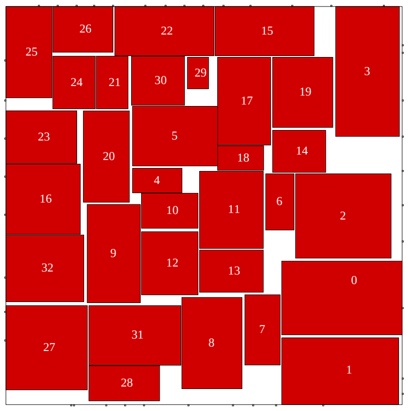
Figure 6: Data set ami33 (3)

[3] 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.

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

[5] 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.

[6] 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).


(a) Preliminary floorplan. Deadspace = 8.99%

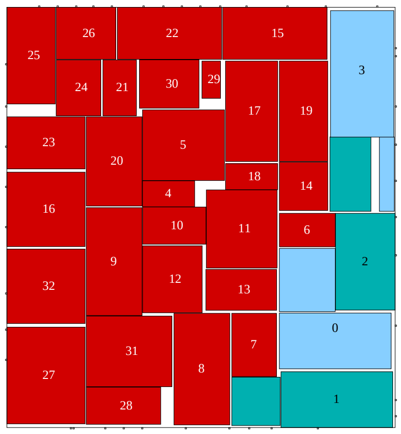
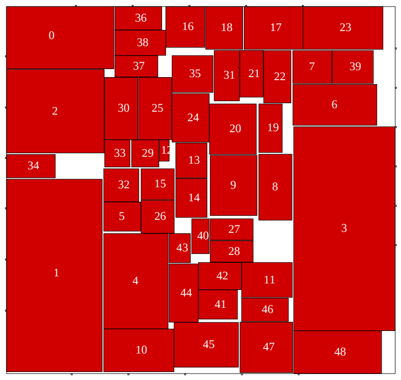

(b) After sizing and shaping. Deadspace = 2.09%

Figure 7: Data set ami33 (5)


(a) Preliminary floorplan. Deadspace = 6.48%

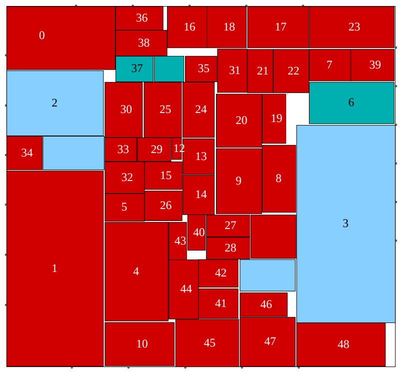

(b) After sizing and shaping. Deadspace = 1.72%

Figure 8: Data set ami49 (1)

[7] 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.

[8] 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.

[9] 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.

[10] 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.

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

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

[13] Chih-Hung Lee, Yu-Chung Lin, Wen-Yu Fu, Chun-Chiao Chang, and Tsai-Ming Hsieh. A New Formulation for SOC Floorplan Area Minimization Problem. *Design, Automation and Test in Europe Conference and Exhibition 2002*, page 1100, 2002.

[14] 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.

[15] T.-S. Moh, T.-S. Chang, and S. L. Hakimi. Globally Optimal Floorplanning for a Layout Problem. *IEEE Transaction on Circuit and Systems - I: Fundamental Theory and Applications*, 43(9):713–720, 1996.

[16] H. Murata and Ernest S. Kuh. Sequence-Pair Based Placement Method for Hard/Soft/Pre-placed Modules. *International Symposium on Physical Design*, pages 167–172, 1998.

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

[18] 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.

[19] R. J. Duffi n, E. L. Peterson, and C. Zener. *Geometric Programming – Theory and Application*. John Wiley & Sons, Inc., NY, 1967.