An Analytical Approach to Floorplanning for Hierarchical Building Blocks Layout

CHANG-SHENG YING AND JOSHUA SOOK-LEUNG WONG, MEMBER, IEEE

Abstract—**This paper describes an *analytical* approach for the floorplanning of rectangular blocks with various constraints on their *connection* and *dimension* such that the total wire length and area of the resulting floorplan are minimized.**

The approach consists of two phases: *relative placement* and *spacing*. Relative placement places the building blocks in a plane such that the topology of the blocks satisfies the combined goal of short interconnection and small bounding area for a given aspect ratio. The spacing phase removes the overlap in the floorplan resulting from relative placement by moving and reshaping the blocks. Both phases are modeled heuristically as unconstrained minimization problem. The constraint on the shape of floorplan is met using a bounding penalty function. Indirect connections between large blocks are taken into account for more efficient area utilization.

The floorplanning algorithm has been implemented. Experimental results show that the approach is effective for handling floorplanning under various kinds of constraints.

I. Introduction

As THE complexity of VLSI circuit grows, the difficulty of design increases. A *hierarchial design* method is considered as an effective way to manage the VLSI layout design problem [13]. In hierarchical design, chip area is partitioned into several large blocks, and each block is recursively partitioned into blocks at lower levels, until each block contains single circuit module.

Given the dimension of a set of blocks and how they communicate with each other, the problem of laying them out optimally is referred as *floorplanning*. This hierarchical layout design can be viewed as a procedure of hierarchical floorplanning of building blocks [4], [9]. We call the building blocks at some level of the hierarchy *son blocks*, and the block created by combining son blocks together a *father block*. To create a father block, the floorplanning lays out its son blocks such that the combined constraints on area and connection are met. Since it is impossible to determine the pins' position before going down to a lower level of hierarchy, pins are assumed located at the centers of son blocks [4]. In a top-down method, the floorplan of a block determines the dimensions of the floorplan of its son blocks. Whenever the floorplan of a block at some level of hierarchy can not meet the constraints imposed by its father block, a *backtrace* is required to update the floorplan of all the ancestral blocks affected.

Floorplanning is traditionally done at the chip level with a few large functional blocks. Many algorithms for chip floorplanning have been reported [5], [6], [12], [16]. But the adaptation of most of them to the hierarchical design methodology is very difficult, for they do not have a global perspective which is an important component of hierarchical design [9].

In this paper, we propose an approach for floorplanning in a hierarchical building block layout design environment. The floorplanning consists of two phases: *relative placement* and overlap-free *spacing* (including decompaction and compaction). Relative placement places the blocks within a bounded area (with prespecified aspect ratio) having regards to the dimensions and the interconnections of blocks. The spacing phase removes overlap between blocks to yield a final layout solution. The constraint on the shape of layout area is considered. This is important for obtaining a globally optimal hierarchical layout design. Another feature is that both the relative placement and spacing phases are modeled heuristically as unconstrained minimization problem and solved by the modified *Self-Scaling Variable Metric* (SSVM) method [10], [11].

The remainder of this paper is organized as follows. We review some well-known strategies for floorplanning in Section II. Section III presents the notations and definitions. The model of relative placement and spacing are described in detail in Sections IV and V, respectively. Section VI gives the floorplanning algorithm and its implementation. Section VII discusses some specific issues related to floorplanning. The experimental results with performance analysis of the floorplanning algorithm are shown in Section VIII. Finally, Section IX contains the conclusions.

II. The State of the Art

There exists many methods for solving the floorplanning problem. Among them are *slicing embedding* [12], *dual graph formation* [2], [5], [8], *force-directed* with *slicing* [21], *analytic* with *packing* [6], [16], *partition* with *shape determination* [7], [9], *hierarchical enumeration* [4], *simulated annealing* [20] and *graphtheoretic* [18].

Manuscript received August 5, 1988; revised November 28, 1988. This work was supported by Research Sub-Committee, Hong Kong Polytechnic, Hung Hom, Kowloon, Hong Kong. The review of this paper was arranged by associate Editor A. E. Dunlop.

C.-S. Ying is with the Department of Electronic Engineering, Hong Kong Polytechnic, Kowloon, Hong Kong, on leave from Tsinghua University, Beijing, China.

J. S.-L. Wong is with the Department of Electrical Engineering, Hong Kong Polytechnic, Kowloon, Hong Kong.

IEEE Log Number 8826349.

In [12], a point embedding is first determined relying on the netlist information, and then a floorplan is obtained by cutting the embedding into a slicing structure. The approach neglects the actual building block dimensions.

In the method of dual graph formation, a layout structure graph is first transformed into a plannar graph by deleting a minimum number of connections and adding crossover vertices, and then an optimal rectangular dual is sought for the planar graph. This approach may not be readily modified to take into account the various constraints imposed by practical applications.

The *divide-and-conquer* scheme has been widely used in floorplanning. Lapotin and Director chose a combined min-cut and slicing approach [7] to floorplanning. The approach is composed of three major parts: a combined bipartitioning and slicing step, a conversion to geometric layout and a floorplan evaluation step. The PIONEER system proposed by Woo *et al.* [21] for floorplanning provided two capabilities: initial layout from user-provided data and interactive graphics for improving the initial layout. The initial layout proceeds in three steps: determination of the macro centers, generation of the slicing structure and expansion of the layout. In [6] and [16], floorplanning is divided into subtasks: initial placement and a block packing process. The initial placement in [6] used the potential energy method and that in [16] used the force method.

The slicing structure is restrictive in area utilization. When there are constraints on the shape of building blocks, the wasted space may turn out to be very large in some instances.

Dai *et al.* [4] presented a hierarchical enumeration method for floorplanning, which is used in the BEAR building block layout system. Although it can create some *non-slicing* floorplan, the limit on the number of blocks (not larger than 5) at each level of hierarchy will restrict the area utilization.

The simulated annealing (SA) technique is used in [20] to obtain a floorplan. One of the major disadvantages of the SA approach is that it is computing intensive and may not be readily adapted to deal with various constraints on floorplan.

III. Definitions and Notations

Before we describe the problem formula of relative placement and spacing, some notations and definitions are given in this section. The input of floorplanning is a set of *building blocks:*

$$b_i, \quad i = 1, m$$

and a set of *nets:*

$$n_i, \quad i = 1, n.$$

Each net is assigned with a *weight*

$$c_i, \quad i = 1, n$$

which specifies the *connectivity* (a concept combining the number of connections and the importance of the connections) between blocks in the net.

Note that for a net (say n_i) connecting more than two blocks, say k, it is represented by $(k(k-1))/2$ pairs of connections, each pair of connection is weighted $2 \times c_i/k$ [1], [15].

For each rectangular block b_i, we have the following variables and constants:

- x_i X coordinate of center of block i.
- y_i Y coordinate of center of block i.
- w_i *Half* of the width of block i.
- h_i *Half* of the height of block i.
- α_i *Aspect ratio* of block i, defined as h_i/w_i.
- s_i Estimated or actual area of block i.

The area of each block is prespecified. The position of block b_i in the floorplan is given by (x_i, y_i). To simplify the discussion, we always center the floorplan at the origin of coordinates. For *fixed-shape* blocks, w_i, h_i, and α_i are constants; for *variable-shape* blocks, they are variables. The parameters w_i, h_i, α_i, and s_i are related by

$$s_i = 4 \times w_i \times h_i, \quad \alpha_i = h_i/w_i$$

$$w_i = \tfrac{1}{2}\sqrt{s_i/\alpha_i}, \qquad h = \tfrac{1}{2}\sqrt{s_i \times \alpha_i}.$$

For each pair of blocks b_i and b_i, we have the following variables and constants:

- $d_{i,j}$ distance between the centers of b_i and b_j, $= \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$
- $\Delta x_{i,j} = |x_i - x_j|$
- $\Delta y_{i,j} = |y_i - y_j|$
- $cc_{i,j}$ the total *connectivity* between i and j
- $cd_{i,j}$ the *disconnectivity* between i and j

$$\begin{cases} = 1, & \text{if } cc_{(i,j)} = 0 \\ = 0, & \text{else} \end{cases}$$ i.e., no connection between b_i and b_j

In addition, we have following measures to the entire floorplan:

- w_0 the *half width* of the floorplan,
- h_0 the half *height* of the floorplan,
- α_0 the expected aspect ratio of the floorplan.

The parameters (or weight) λ_1 and λ_2 are used in some formula to define the relative importance of different terms.

IV. The Model for Relative Placement

Relative placement places the blocks in a plane such that a *good topology* between building blocks is obtained which best reflects the *interconnections* and the *dimensions* of the blocks.

There are many heuristics for relative (also called *initial*) placement. These may be *constructive* or *analytic*, *discrete* or *continuous*. Among them the point model was widely used such as in the Force-directed approach [14], [20]. It works well for regular layouts (such as gate array layout and standard cell layout), where points in the relative placement can be properly mapped to the slots (basic cells) in each row of the layout while maintaining the rel-

ative positions. However, the same method, when applied to building block layout, does not usually lead to efficient area utilization due to the irregular dimensions of the blocks. Therefore, we choose the *potential energy* model with each block modeled as a circle instead of a point for our implementation of the relative placement such that the size of each block is, to a large extent, taken into account.

Blocks can be of variable-shape or fixed-shape. In the relative placement both the interconnection and the size of blocks are the main concerns. The actual shape of each block is determined in the spacing phase. But the constraint on the aspect ratio of the whole layout area is done in the relative placement.

4.1. The Potential Energy Model

By potential energy between blocks we mean: when two blocks (connected or unconnected) overlap, a *potential* field exists which produces a *force* tending to separate them. The more they overlap, the larger the force. When two connected blocks are separated there is a force which tends to pull the two blocks closer. Without separation or overlap, blocks are said to be in equilibrium condition [6].

In order to formulate the potential energy model, each block is modeled as a *circle* which has a prespecified radius, r_i,

$$\text{set } r_i = w_i, \qquad i = 1, m.$$

The overlap, $p_{i,j}$, between blocks b_i and b_j is defined as

$$p_{i,j} = \max\left\{0, (r_i + r_j) - d_{i,j}\right\}$$

The energy $e_{i,j}$ between two blocks b_i and b_j is defined as follows:

(a) if block i and j are *connected*,

$$e_{i,j} = cc_{i,j}\left(\frac{\delta_{(i,j)}}{d_{(i,j)}} p_{i,j} + \frac{d_{(i,j)}}{\delta_{(i,j)}} - 1\right).$$

(b) if block i and j are *disconnected*,

$$e_{i,j} = cd_{i,j} \frac{\delta_{(i,j)}}{d_{(i,j)}} p_{i,j}$$

where $\delta_{i,j} = r_i + r_j$.

When the distance $d_{i,j}$ between two connected blocks b_i and b_j are equal to the sum of radius $\delta_{i,j}$, $e_{i,j} = 0$, i.e, b_i and b_j are in equilibrium condition; when there is an overlap ($p_{i,j} > 0$) or separation ($d_{i,j}/\delta_{i,j} > 1$), $e_{i,j} > 0$, i.e., b_i and b_j are in non-equilibrium condition. For unconnected blocks b_i and b_j, $e_{i,j} = 0$ if and only if there is no overlap between them.

The objective is to minimize the overall energy among blocks.

$$\text{Minimize } E = \sum_{i,j=1}^{m} e_{i,j} \tag{4.1}$$

Hence the relative placement is modeled as a *nonlinear unconstrained optimization* problem. The solution of the problem gives the position of each block.

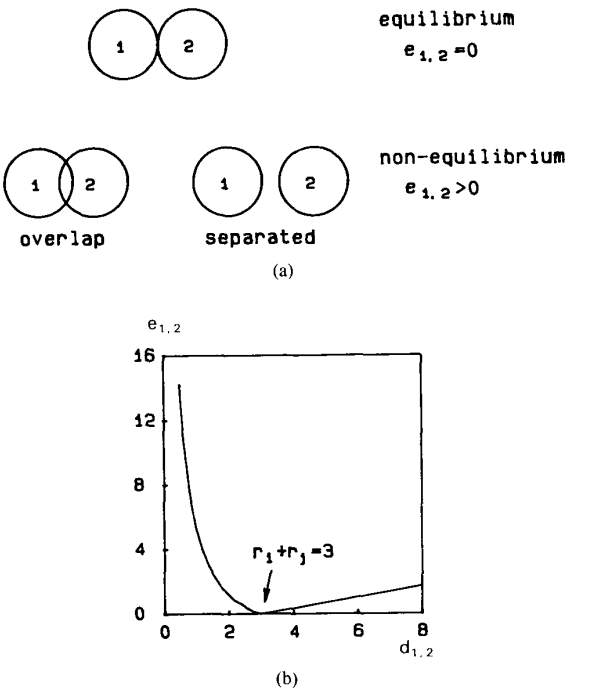


Fig. 1. (a) Potential energy model. (b) Graph of $e_{i,j}$ versus $d_{i,j}$.

Fig. 1 shows the potential energy model and the graph of energy $e_{(i,j)}$ versus distance $d_{(i,j)}$ of two connected blocks b_i and b_j. There is a rapid increase of $e_{(i,j)}$ in the case of overlap and a slow increase in the case of separation.

The model reflects the connection and overlap between blocks directly. It has heavier penalty on overlap than that on separation. Such a model is more reasonable than that in [1] and [6]. In [6], Hsu and Kubitz proposed a potential energy model which treats overlap and separation equally, but in the final floorplan solution, two connected blocks may be separated but cannot be overlap. Using the potential energy model proposed by Alon and Ascher in [1], two connected blocks may tend to be far away from each other.

4.2. Bounding Function

To meet the constraint on the dimension of the layout area, we introduce a *bounding function* as a *penalty* function to limit the location of blocks within a prespecified boundary. The boundary is defined by w_0 and h_0, and the constraint on each block is

$$|x_i| \le w_0 - w_i, \; |y_i| \le h_0 - h_i, \qquad i = 1, m.$$

The expected aspect ratio of layout area (α_0) is reflected in parameter w_0 and h_0. If there is no constraint on α_0, we assume $\alpha_0 = 1$. We define bounding function as

$$B = \sum_{i=1}^{m} (Bdx_i + Bdy_i) \tag{4.2}$$

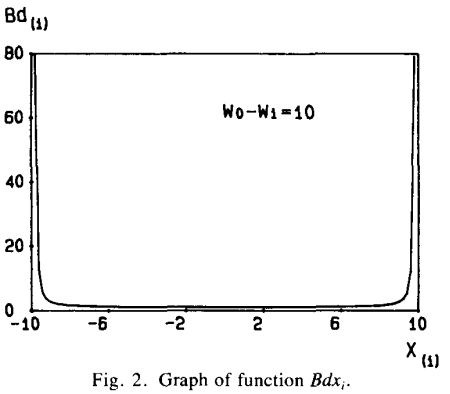


Fig. 2. Graph of function Bdx_i.

and

$$Bdx_i = \begin{cases} e^{1/(w_0 - w_i - |x_i|)}, & \text{if } |x_i| < w_0 - w_i \\ \infty, & \text{else} \end{cases}$$

$$Bdy_i = \begin{cases} e^{1/(h_0 - h_i - |y_i|)}, & \text{if } |y_i| < h_0 - h_i \\ \infty, & \text{else} \end{cases}$$

where e is the exponential function.

Fig. 2 shows the feature of the bounding function. There is a *rapid increase* of the function value when a block is close to the boundary and a *slow decrease* when it is close to the center of the layout area. Minimizing the function value can keep blocks close to the center of layout area.

Combining the potential energy model and bounding penalty function, we obtain an improved relative placement model:

$$\text{Minimize } Z = E + \lambda_1 B. \tag{4.3}$$

This is an unconstrained minimization problem. The model says that while the interconnection of blocks is reflected adequately, the constraint on the dimension of the layout area is met also. There may be a conflict between the minimization of the energy E and the value of the bounding function B, but the model permits a globally optimal solution.

4.3. The Calibration Function

According to the potential energy model defined above, block i and j are in equilibrium condition in both Fig. 3(a) and (b). But, by considering the actual rectangular shape of each block, we prefer Fig. 3(b) to Fig. 3(a), since Fig. 3(a) conceals an overlap between block i and j.

To make up for the *approximation* of modeling a block as a circle, we use a penalty function C, called the *calibration* function, to improve the model for relative placement.

The calibration function C is formulated as

$$C = \sum_{i,j=1}^{m} cc_{i,j} \times CB_{i,j} \tag{4.4}$$

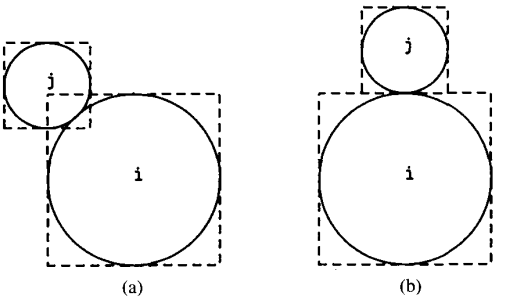


Fig. 3. In (a) and (b), $e_{e,j} = 0$, but (a) conceals an overlap.

where

$$CB_{i,j} = \frac{\Delta x_{i,j} \times \Delta y_{i,j}}{\delta_{i,j}}$$

To minimize $CB_{i,j}$ means to align blocks b_i and b_j horizontally or vertically, i.e., make $\Delta x_{i,j} \to 0$ or $\Delta y_{i,j} \to 0$. We obtain our final version of relative placement model as follows:

$$\text{Minimize } Z = E + \lambda_1 B + \lambda_2 C. \tag{4.5}$$

Generally, $0 < \lambda_2 < \lambda_1 < 1$.

The calibration function helps to improve the quality of the relative placement. The problem as depicted in Fig. 3 can then be automatically solved.

V. The Model for Spacing

After relative placement, one may claim that blocks are placed in a good topology as far as the constraints on connections and area are concerned. But each block is modeled as a circle in the relative placement. When each block takes on its rectangular shape, there may exist overlap between blocks. The spacing procedure will determine the final floorplan solution by removing overlap and ensure the least bounding area.

In the spacing phase, blocks may be moved a small distance and/or *reshaped* while satisfying their *shape constraints*. For a block (say b_i), there may be three parameters to be determined, x_i, y_i, and α_i. Variable-shape blocks take on the square shape initially while fixed-shape blocks take their prespecified shapes (their relative orientations can be so chosen as to minimize the overlap with the nearby blocks).

Before we introduce the spacing model, two definitions should be given first. In the relative placement of the blocks in actual or assumed rectangular shape, we say that blocks b_i and b_j are *horizontally related* (or *vertically related*) if the overlap (minus value in the case of separation) in the X direction (or Y direction) between blocks i and j is less than that in the Y direction (or X direction), as illustrated in Fig. 4. Two variables "*hr*" and *vr*" are defined as follows to represent the horizontal related and

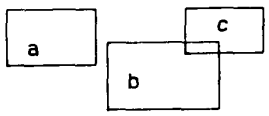


Fig. 4. Horizontally and vertically related blocks, $hr_{a,b} = 1$, $hr_{a,c} = 1$, and $vr_{b,c} = 1$.

vertical related relationship among blocks respectively,

$$hr_{i,j} = \begin{cases} = 1, & \text{if } b_i \text{ and } b_j \text{ are horizontally related} \\ = 0, & \text{else} \end{cases}$$

$$vr_{i,j} = \begin{cases} = 1, & \text{if } b_i \text{ and } b_j \text{ are vertically related} \\ = 0, & \text{else.} \end{cases}$$

The physical meaning behind these two relations is that: if two blocks are horizontally related, it is easier to remove the overlap which may exist between them in the horizontal direction than in the vertical direction.

5.1. The Overlap Freeing Model

If two blocks are horizontally related (or vertically related), there may be an overlap between them in horizontal (or vertical) direction. The overlap in horizontal direction and vertical direction (*HO* and *VO* for short, respectively) between blocks b_i and b_j can be formulated as follows and illustrated in Fig. 5.

$$HO_{i,j} = hr_{i,j} \times \max\left\{0, (w_i + w_j) - \Delta x_{i,j}\right\} \quad (5.1)$$

$$VO_{i,j} = vr_{i,j} \times \max\left\{0, (h_i + h_j) - \Delta y_{i,j}\right\}. \quad (5.2)$$

Substituting w_i and h_i and α_i (see Section III), (5.1) and (5.2) becomes

$$HO_{i,j} = hr_{i,j} \times \max\left\{0, \tfrac{1}{2}\left(\sqrt{s_i/\alpha_i} + \sqrt{s_j/\alpha_j}\right) - \Delta x_{i,j}\right\} \quad (5.3)$$

$$VO_{i,j} = vr_{i,j} \times \max\left\{0, \tfrac{1}{2}\left(\sqrt{s_i \times \alpha_i} + \sqrt{s_i \times \alpha_j}\right) - \Delta y_{i,j}\right\}. \quad (5.4)$$

The objective is to minimize the overall overlap in the horizontal and vertical directions.

$$\text{Minimize } L = \sum_{i,j=1}^{m} (HO_{i,j} + VO_{i,j}) \quad (5.5)$$

Each block is specified by a set of three parameters x_i, y_i, and α_i. The complete sets of parameters corresponding to all the blocks such that L is minimum are to be determined. This constitutes the analytical model of the spacing phase. It is a constrained minimization problem with constraint on the aspect ratios α_i

$$\alpha_i > 0, \qquad i = 1, m$$

since by definition the value of α_i cannot be less than or equal to zero.

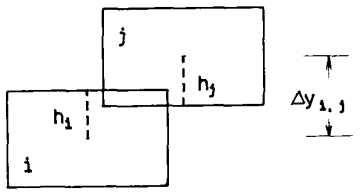


Fig. 5. Vertical overlap *VO* (similar for *HO*) $VO_{i,j} = (h_i + h_j) - \Delta y_{i,j}$.

We make a *transformation* by introducing a new variable t_i for each block. Let

$$t_i^2 = \alpha_i, \qquad i = 1, m$$

and the overlap formula becomes

$$HO_{i,j} = hr_{i,j} \times \max\left\{0, \tfrac{1}{2}\left(\sqrt{s_i}/|t_i| + \sqrt{s_j}/|t_j|\right) - \Delta x_{i,j}\right\} \quad (5.6)$$

$$VO_{i,j} = vr_{i,j} \times \max\left\{0, \tfrac{1}{2}\left(\sqrt{s_i} \times |t_i| + \sqrt{s_j} \times |t_j|\right) - \Delta y_{i,j}\right\}. \quad (5.7)$$

Replace *HO* and *VO* in (5.5) with their new definition, the spacing model becomes an unconstrained minimization problem.

According to (5.5), it is obvious that the overall overlap will be minimized to zero if all of the blocks are far away from each other. Such a solution is not what we want, for the entire layout area should be kept as small as possible. Here the *boundary function* defined in Section IV is used again to limit the location of each block. And the spacing model becomes

$$\text{Minimize } Z = L + \lambda_1 B. \quad (5.8)$$

5.2 Bounding Aspect Ratios

In (5.8), the shapes of the blocks are assumed to be arbitrary and flexible. Actually, there are constraints on them, at least, the width or height of a block cannot be less than a minimum feature size. So the aspect ratio of each block must be in a limited range. We introduce a parameter β, let

$$\beta_i = \alpha_i + 1/\alpha_i, \quad (\text{and } \alpha_i = t_i^2), \qquad i = 1, m$$

where β_i shows how thin or narrow the block b_i is. We then have a constraint on β_i:

$$\beta_i < f_i, \qquad i = 1, m. \quad (5.9)$$

Here parameter f_i represents a feasible range of variation of the shape of block b_i, the larger the f_i, the more flexible the block b_i. It is prespecified or set internally,

$$f_i = s_i + 1/s_i, \qquad \text{where } s_i \text{ is the area of } b_i.$$

In Section IV we use a bounding function B for constraint on the locations of the blocks. To meet the constraint (5.9), we use a similar bounding function B':

$$B' = \sum_{i=1}^{m} Bdr_i \quad (5.10)$$

where

$$Bdr_i = \begin{cases} e^{1/(f_i - \beta_i)}, & \text{if } \beta_i < f_i \\ \infty, & \text{else.} \end{cases}$$

Hence, we have the final spacing model as

$$\text{Minimize } Z = L + \lambda_1 B + \lambda_2 B'. \qquad (5.11)$$

5.3 Overlap Transfer

Within a given layout boundary, a floorplan solution without overlap may not be found. In this case, the layout boundary has to be *expanded* by increasing w_0 and h_0, and the final floorplan is obtained after a few *iterative* applications of the spacing procedure.

To *speed up* the spacing iteration, we use an "*overlap transfer*" technique. When overlap cannot be avoided within current layout area, we seek to transfer the overlap such that it occurs *away* from the center of the layout area. When the area is expanded, the overlap can then be removed easily. This can be automatically implemented by *weighting the overlap* in the model. The modified overlap formula is

$$HO_{i,j} = \frac{hr_{i,j}}{|x_i + x_j|} \times \max\left\{0, \tfrac{1}{2}(\sqrt{s_i}/|t_i| + \sqrt{s_j}/|t_j|) - \Delta x_{i,j}\right\} \qquad (5.12)$$

$$VO_{i,j} = \frac{vr_{i,j}}{|y_i + y_j|} \times \max\left\{0, \tfrac{1}{2}(\sqrt{s_i} \times |t_i| + \sqrt{s_j} \times |t_j|) - \Delta y_{i,j}\right\}. \qquad (5.13)$$

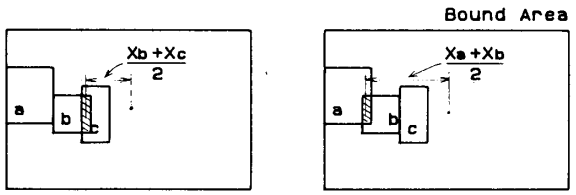
The weight "$1/|x_i + x_j|$" (and "$1/|y_i + y_j|$") reflects approximately the distance between the location where horizontal (and vertical) overlap occurs and the center of the layout area. Fig. 6 shows the overlap transfer.

5.4 Fixed-Shape Constraint

We have mentioned that the input building blocks can be of fixed-shape or variable-shape. The constraints on variable-shape blocks are met by using the bounding function B'. For fixed-shape blocks, their aspect ratios are constants during the spacing phase.

5.5 Forbidding Large Movement

In the context of our floorplanning approach, the position of each block will not change drastically (relative to its dimension) in the spacing procedure, because the relative placement procedure takes into account the area of each block and tends to keep the overlap between blocks as small as possible. Therefore, it is reasonable to *forbid large movement* on blocks during the spacing procedure. This can speed up the convergency of the spacing and help to maintain the good topology between blocks. In addition, we have the new definitions of "*hr*" and "*vr*,"



(a) (b)

Fig. 6. Overlap transfer to the boundary in the horizontal direction.

$$hr_{i,j} = \begin{cases} = 1, & \text{if } b_i \text{ and } b_j \text{ are horizontally related and in each other's proximity} \\ = 0, & \text{else} \end{cases}$$

$$vr_{i,j} = \begin{cases} = 1, & \text{if } b_i \text{ and } b_j \text{ are vertically related and in each other's proximity} \\ = 0, & \text{else.} \end{cases}$$

It is obvious that "*hr*" and "*vr*" become very sparse and thus help speed up the spacing phase.

VI. Implementations

6.1 The Floorplanning Algorithm

Based on the relative placement model and the spacing model mentioned above, we develop a *floorplanning algorithm* as follows.

S1: ***Locate*** each block on the layout area (with prespecified size or set equal to the sum of the size of all building blocks) in its *gravitational-center position* determined from the connectivities with *I/O pads* and with *other blocks*.

S2: Model each block as a circle, and obtain the ***relative placement*** of the blocks by evaluating the analytical model for the relative placement.

S3: ***Replace*** the circle model of each block with rectangular shape in the relative placement, square shape for variable-shape blocks and prespecified shape for fixed-shape blocks.

S4: ***Remove overlap*** between blocks by manipulating the spacing model.

S5: If a floorplan without overlap is obtained, then ***stop;*** else increase the w_0 and h_0, ***goto Step 4).***

6.2 Solution of Minimization Problem

We used the modified Self-Scaling Variable Metric (SSVM) method [10], [11] for the unconstrained minimization of nonlinear functions derived from the relative placement model and the spacing model. The appendix contains a brief introduction of the SSVM method and some of the issues considered in our implementation. For this implementation, the calculations of the functional values and the gradient vector for the variables take into account the constraints of "fixed-shape" and "forbidding large movement" described in last section. The method converges quickly and thus ensures reasonable running time.

VII. Some Useful Remarks

In our discussion of floorplanning, we have focused on one level of the layout hierarchy. In a hierarchical layout design, floorplanning will be performed within each block of the design, and at each level of the hierarchy. To create a block, we assume that the dimension and connection of its son blocks are given; the floorplanner then lays the son blocks out such that the constraints on the dimension and connection of the son blocks and the bounding dimension constraint imposed by its father block are met.

It is recognized that routing area estimation is crucial to a good floorplanner. In our approach to the hierarchical layout, a process of bottom-up floorplanning and global routing is performed to estimate the dimension (include routing area) of blocks at each level of hierarchy before going on the top-down floorplanning. During the process of bottom-up floorplanning and global routing the overall aspect ratio and the I/O pad positions are not yet considered. Only the size requirement of blocks at each level of hierarchy is of interest. A fast global routing algorithm [22] has been proposed which could be used in routing area estimation. The bottom-up estimation of the dimension of the blocks ensures a limited number of design backtraces.

If the routing area is taken into account at each level or hierarchy, two more steps are required in the floorplanning algorithm presented in Section 6.1, they are as follows.

S6: Perform the global routing and obtain the routing density on the boundaries of each block.

S7: Enlarge each block according to the routing density on its boundaries, repeat Step 4 and 5 until a floorplan without overlap is obtained.

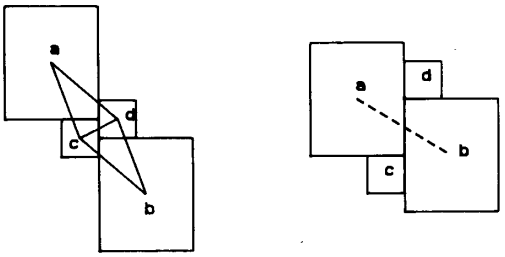
When a floorplan of a son block at some level cannot meet the constraint imposed by the floorplan of its father block, a backtrace is needed to modify the floorplan of its father block. This process can be done easily in our floorplanning approach as follows: ***first***, replace the son block in the floorplan of its father block with its actual dimension. ***Then***, take this new floorplan containing overlap as the input to the spacing phase and call Steps 4 and 5 in the floorplanning algorithm; Step 3 is not necessary since relative placement is still regarded as reasonable. A new floorplan of its father block without overlap can be obtained.

In this paper we assume that the pins are located at the center of the block in the context of hierarchical layout design. At the lowest level of hierarchy the information about the pins position is readily known, a block orientation refinement procedure may be used as a postprocess of the floorplanning.

VIII. Experiments

The floorplanning algorithm has been implemented in C language on a VAX 11/750 running UNIX and the results are displayed on a *Tektronic* 4113 graphic terminal.

Consideration was given to the definition of the connectivity between blocks. If the connectivity is defined simply as the number of interconnections, the results are found to be unpredictable, i.e., a small change in the number of connection in a net may lead to a quite different relative placement. Therefore, we use five connectivity ranks of positive values to account for both the number of connections and the importance of connections between blocks. A detailed experiment on the connectivity was found in [17].



(a) (b)

Fig. 7. Considering the connectivity between blocks: (a) not taking into account the indirect connections; (b) taking into account the indirect connections between large blocks.

A further investigation was made on the relationship between the connectivity and the sizes of the blocks. Two blocks are indirectly connected if they are connected to some common block(s) instead of being connected to each other directly. For a better layout area utilization, two large indirectly connected blocks should be located nearby if their common neighbors are small. To account for this situation, a connectivity is assigned to each pair of large blocks which are indirectly connected. We modify the connectivity as following: given a connectivity matrix cc as mentioned in Section III (and set $cc_{ii} = 0$), perform a self-multiplication of matrix cc and thus obtain a new connectivity matrix involving the connection between indirectly connected blocks. The transformations on cc are

$$cc = cc \times cc$$

$$cc_{i,j} = s_i \times s_j / avs^2$$

where "avs" is the average area of all blocks. The indirect connection between small blocks is ignored during the transformations.

Fig. 7 shows an example, where a, b, c, and d are four blocks, a and b being much larger than c and d. There are connections between a and c, a and d, b and c, b and d; while a and b are indirectly connected. A solution of relative placement is shown in Fig. 7(a) (other blocks are ignored); however, a better topology which tends to utilize the layout area more efficiently can be expected if the indirect by connected blocks a and b are assigned a connectivity, as shown in Fig. 7(b).

Various floorplanning problems were tested. Table I presents the detailed experimental results on two problems "$ex1$" and "$ex2$" from [23] and [5], respectively, one with 8 blocks and 38 nets and the other with 21 blocks and 53 nets. It is difficult to compare the results with those of BBL's [23] and Heller's [5] since the final layout so-

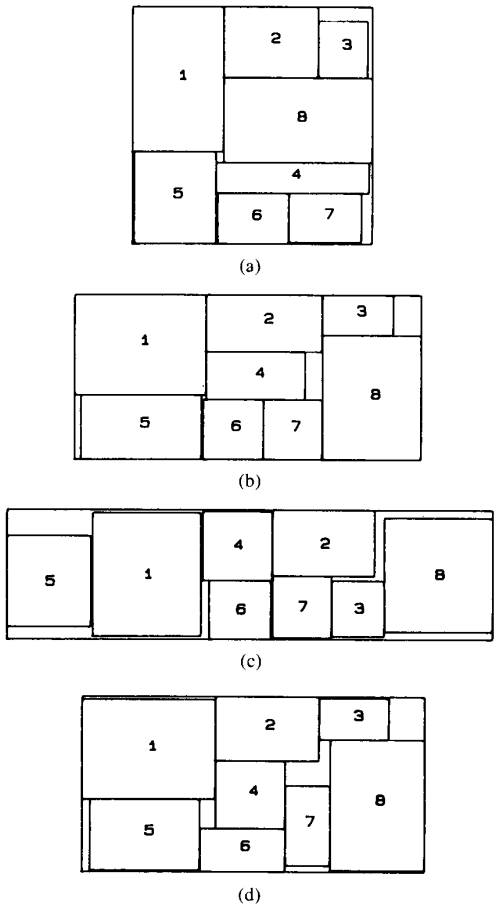


Fig. 8. Floorplan example from [24], with additional constraints on aspect ratio. (a) $\alpha_0 = 1$. (b) $\alpha_0 = 0.5$. (c) $\alpha_0 = 0.25$. (d) $\alpha_0 = 0.5$ and $\alpha_4 = 1$.

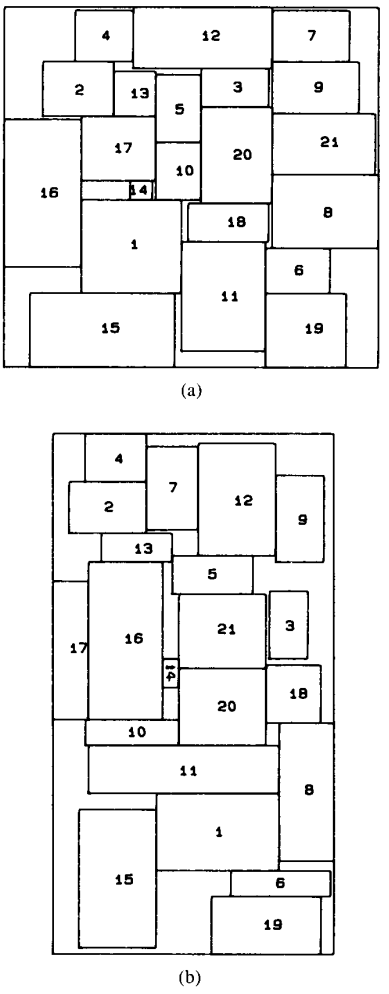


Fig. 9. Another floorplan example from [5] with 21 blocks and 53 nets, the imposed constraint is (a) $\alpha_0 = 1$, (b) $\alpha_0 = 2$.

TABLE I
EXPERIMENTAL RESULTS OF TWO FLOORPLANNING EXAMPLES

Input	Constraint	Area	Wire Length	CPU Time(sec.) T1	T2	See Fig
ex1	αo=1	23822	1899	8.12	30.6	Fig.8a
"	αo=0.5	24394	1938	4.12	51.6	Fig.8b
"	αo=0.5 α_4=1	25048	2090	4.12	48.8	Fig.8d
	αo=0.25	26369	2380	2.06	41.2	Fig.8c
"	αo=3.0	25890	2441	2.18	33.6	
	αo=4.0	25931	2552	2.3	37.6	
ex2	αo=1	13083	21796	116.4	301.2	Fig.9a
"	αo=2	14044	24871	46.7	371.4	Fig.9b
"	αo=0.5	13445	23916	50.2	416.3	
	αo=0.66	13210	20214	114.4	263.2	

Note : example "ex1" is from [23] and has 8 blocks and 38 nets, example "ex2" is from [5] and has 21 blocks and 53 nets. T1 and T2 in the table are the CPU times for the relative placement and the spacing phases respectively in a VAX 11/750 running UNIX.

lutions are not available, and the constraints on the blocks are not the same.

In Table I, the term "wire length" is defined as

$$\sum_{i,j=1}^{m} cc_{i,j} \times (\Delta x_{i,j} + \Delta y_{i,j})$$

it reflects approximately the quality of the floorplan solution with respect to the short connection goal.

Fig. 8 presents four solutions of problem "*ex1*" under various constraints on the shape of the layout area α_0 and blocks (α_i). Fig. 9 shows two solutions of problem "*ex2*."

Plots in Fig. 10 demonstrate the performance of the floorplanning algorithm, where the X coordinate is "$\alpha_0 + 1/\alpha_0$" which is the variation of the shape of the layout area and the Y coordinate are "wire length" or "area" which are the main measurement of the performance of the floorplanning. The plots show that the algorithm yields stable performance with respect to a large variation of contraint on the aspect ratio of the layout area.

IX. CONCLUSIONS

An *analytical* approach for floorplanning has been presented in the context of hierarchical building block layout design. The floorplanning is divided into two phases, relative placement and overlap-free spacing. Several floor-

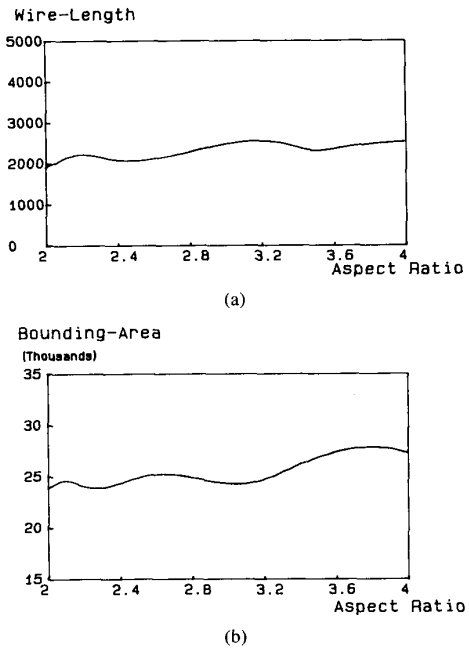


Fig. 10. Plots of the performance of the floorplanning algorithm versus the variation of constraint on the shape. The X coordinate is the variation of the shape ($\alpha_0 + 1/\alpha_0$). The Y coordinate is the area and the wire length which are the main measurements of the performance. (a) The plot of *wire-length* versus ($\alpha_0 + 1/\alpha_0$). (b) The plot of *area* versus ($\alpha_0 + 1/\alpha_0$).

plan examples have been tested. The primary experimental results are quite encouraging.

The key features of the present method are summarized as follows:

a) Fixed-shape and variable-shape blocks can be accommodated on the same floorplan.
b) Prespecified constraint on the aspect ratio of layout area is always satisfied.
c) The relative placement model takes into account the overlaps and the sizes of blocks directly, and the calibration function helps improve the quality of the relative placement.
d) The overlap-free spacing procedure keeps the layout compact while removing the overlap between the blocks.
e) Both procedures of relative placement and spacing are modeled as nonlinear unconstrained minimization problem and are solved with the same modified SSVM method.
f) Two bounding functions help meet the constraint on the dimensions of the layout area and the blocks. At the same time, they help speed up the convergency of the floorplanning algorithm.
g) Taking into account the indirect connections between large blocks leads to more efficient area utilization.

The next step is to incorporate this floorplanning approach to hierarchical layout design.

Appendix

The SSVM method is briefly described in this section. An important class of widely applicable algorithms for solving smooth unconstrained minimization problems are the quasi-Newton methods (also referred to as variable metric algorithms). To minimize a functional f depending on the n-dimensional vector x, these algorithms have the form

$$x_{i+1} = x_i - \lambda_i D_i g_i$$

where x_i is the ith approximation to the minimum point, g_i is the gradient of f at x_i, D_i is an $n \times n$ matrix that approximates the inverse Hessian of f at x_i and λ_i is a positive step size parameter whose value is selected according to some rule depending on the specific method.

Oren [10] proposed a new class of quasi-Newton algorithms called the SSVM for unconstrained minimization in which no line search is necessary and the inverse Hessian approximations are positive definite. In the SSVM methods, D is updated by the following two-parameter formula at every iteration:

$$D_{i+1} = \left(D_i - \frac{D_i q_i q_i' D_i}{q_i' D_i q_i} + \theta v_i v_i'\right)\gamma_i + \frac{p_i p_i'}{p_i' q_i}$$

where

$$v_i = \sqrt{q_i' D_i q_i}\left(\frac{p_i}{p_i' q_i} - \frac{d_i q_i}{q_i' D_i q_i}\right)$$

$$\gamma_i = \varphi \frac{g_i' p_i}{g_i' D_i q_i} + (1 - \varphi)\frac{p_i' q_i}{q_i' D_i q_i}$$

$$q_i = g_{i+1} - g_i; \quad p_i = -\lambda_i D_i G_i$$

$$\varphi, \theta \in [0, 1].$$

As concluded in [10], in a quadratic case, the SSVM algorithm converges at least weak superlinearly; it has substantial advantage for functions with a large number of variables (>10) in comparison to the previous quasi-Newton methods. Therefore, we use the SSVM method for the minimization problems at hand. In our implementation, we test the effect of varying the parameters φ and θ and select them appropriately, since different minimization problems may correspond to different values of φ and θ for good performance.

Acknowledgment

The suggestions and comments of the referees were greatly appreciated. Their remarks contributed to the quality of discussions in several sections of this paper. The first author wishes to thank Prof. E. Q. Wang and Prof. X. L. Hong of Tsinghua University, Beijing, China, for their valuable support and encouragement during this work.

References

[1] A. Alon and U. Ascher, "Model and solution strategy for placement of rectangular blocks in the Euclidean plane," *IEEE Trans. Computer-Aided Design*, vol. 7, pp. 378–386, March 1988.

[2] J. Bhaske and S. Sahni, "A linear algorithm to find a rectangular dual of a planar triangulated graph," in *Proc. 23rd DAC*, pp. 108–114, 1986.
[3] C. Chen and E. S. Kuh, "Module placement, based on resistive network optimization," *IEEE Trans. Computer-Aided Design*, vol. CAD-3, pp. 218–225, July 1984.
[4] W. M. Dai and E. S. Kuh, "Hierarchical floor planning for building block layout," in *IEEE Proc. ICCAD*, pp. 454–457, Nov. 1986.
[5] W. P. Heller, G. Sorking, and K. Maling, "The planar package planner for system designers," in *Proc. 19th DAC*, pp. 252–259, 1982.
[6] Y. C. Hsu and W. J. Kubitz, "A procedure for chip floor planning," in *IEEE Proc. ISCAS*, pp. 568–571, 1987.
[7] D. P. LaPotin and S. W. Director, "MASON: A global floorplanning approach for VLSI design," *IEEE Trans. Computer-Aided Design*, vol. CAD-5, pp. 477–489, Oct. 1986.
[8] S. M. Leinman and Y. T. Lai, "An algorithm for building rectangular floor plans," in *IEEE Proc. 21 DAC*, pp. 663–664, 1984.
[9] H. Modarres and A. Kelapure, "An automatic floorplanner for up to 100,000 gates," *VLSI System Design*, pp. 38–44, Dec. 1987.
[10] S. S. Oren, "Self-scaling variable metric algorithms without line search for unconstrained minimization," *Math. Comp.*, vol. 27, no. 124, 1973.
[11] S. S. Oren and D. G. Luenberger, "Self-scaling variable metric (SSVM) algorithms. Part II: Implementation and experiments," *Manag. Sci.*, vol. 20, no. 5, pp. 863–874, Jan. 1974.
[12] R. H. J. M. Otten, "Automatic floorplan design," in *Proc. 19th DAC*, pp. 261–267, 1982.
[13] B. T. Preas and C. W. Gwyn, "Method of hierarchical automatic layout of custom LSI circuit masks," in *Proc. 14th DAC*, pp. 206–211, 1978.
[14] N. R. Quinn Jr. and M. A. Breuer, "A force directed component placement procedure for printed circuit boards," *IEEE Trans. Circuits Syst.*, vol. 26, pp. 377–388, June 1979.
[15] L. Sha and R. W. Dutton, "An analytic algorithm for placement of arbitrarily sized rectangular blocks," in *Proc. 22nd DAC*, pp. 602–608, 1985.
[16] K. Ueda, H. Kitazawa, and I. Harada, "CHAMP: Chip floor plan for hierarchical VLSI layout design," *IEEE Trans. Computer-Aided Design*, CAD-4, pp. 12–22, Jan. 1986.
[17] T. Watanabe and H. Baba, "A floor-plan design system for LSI layout," in *Proc. ISCAS*, pp. 9–12, 1985.
[18] S. Wimer, I. Koren, and I. Cederbaum, "Floorplans, planar graphs, and layouts," *IEEE Trans. Circuits Syst.*, vol. 35, pp. 267–278, Mar. 1988.
[19] G. J. Wipfler, M. Weisel, and D. A. Mlynski, "A combined force and cut algorithm for hierarchical VLSI layout," in *Proc. 21st DAC*, pp. 671–676, 1982.
[20] D. F. Wong and C. L. Liu, "A new algorithm for floor plan design," in *Proc. 23rd DAC*, pp. 101–107, 1986.
[21] L. S. Woo, C. K. Wong, and D. T. Tang, "PIONEER: A macro-based floor-planning design system," *VLSI System Design*, pp. 32–43, Aug. 1986.
[22] C. S. Ying, J. S. L. Wong, X. L. Hong, and E. Q. Wang, "A converging search algorithm for Steiner tree on global routing graph," in preparation.
[23] C. Chen *et al.*, "Appendix B of BBL system user manual—Input data format of placement," Dept. of EECS, Univ. of California, Berkeley, 1984.

*

Chang-Sheng Ying received the B.S. degree in computer engineering from the Hefei Polytechnic University, Hefei, China, in 1984.

Since 1984 he has been in Tsinghua University, Beijing, where he is currently working toward the Ph.D. degree in computer science.

At present he is on leave as a Research Assistant in the Department of Electronic Engineering, Hong Kong Polytechnic, Hong Kong. His research interests lie in the area of computer-aided design of integrated circuits, with emphasis on data structure and building block layout.

*

Joshua Sook-Leung Wong received the bachelor's degree in electrical engineering from the University of Hong Kong in 1961, and the Ph.D. degree from the University of Leeds, Leeds, UK, 1965.

He was chief designer with Racal Research Ltd., UK, 1967, associate professor of engineering with the California State University at Los Angeles 1968–1974, and with the Jet Propulsion Laboratory of the California Institute of Technology 1972–1974 and summers of 1976, 1980, and 1984. He has been Head of the Department of Electronic Engineering, Hong Kong Polytechnic since 1974 and Chairman of the Division of Engineering, Hong Kong Polytechnic, since 1987. His current interest is in VLSI CAD tools.