Sequence-Pair Approach for Rectilinear Module Placement

Jin Xu, *Member, IEEE,* Pei-Ning Guo, *Student Member, IEEE,* and Chung-Kuan Cheng, *Senior Member, IEEE*

***Abstract*— With the recent advent of deep sub-micron technology and new packaging schemes such as multichip modules, integrated circuit components are often not rectangular. Most existing block placement approaches, however, only deal with rectangular blocks, resulting in inefficient area utilization. New approaches which can handle arbitrarily shaped blocks are essential to achieve high-performance design. In this paper, we extend the sequence-pair approach for rectangular block placement to arbitrarily sized and shaped rectilinear blocks. Experimental results show that our algorithm achieves results with excellent area utilization.**

***Index Terms*—Deep submicron, layout, placement, very large scale integration (VLSI).**

I. Introduction

WIT THE recent advent of deep submicron technology and new packaging schemes such as multichip modules, integrated circuit components are often not rectangular. Most existing block placement approaches, however, only deal with rectangular blocks, resulting in inefficient area utilization. New approaches which can handle arbitrarily shaped blocks are essential to achieve high-performance design.

Kang *et al.* [3] proposes a genetic simulated annealing algorithm for L-shaped, T-shaped and soft blocks. Based on bounded-slicing structure [4], the algorithm combines the simulated anealing (SA)-based local search and genetic algorithm (GA)-based global crossover for general nonslicing floorplanning.

Lee [5] extends the zone refinement technique introduced by Shin *et al.* [8] to arbitrarily shaped rectilinear and soft blocks. A rectilinear block is represented by four linear profiles viewed from four directions. A profile is specified by a series of line segments, each of which is defined by two breaking points. A bounded two-dimensional contour searching algorithm is proposed to find the best position for the block.

Preas *et al.* [10] proposes a graph model for the topological relationship between rectangular blocks. An iterative improvement algorithm is presented to reduce both area and interconnections. Wong *et al.* [9] extends the Polish expression to represent floorplans of rectangular and L-shaped blocks. A simulated annealing method is used to search for optimal floorplan.

Manuscript received September 30, 1998. This work was supported in part by grants from the National Science Foundation (NSF) under Project MIP-9529077 and the California MICRO Program. This paper was recommended by Associate Editor M. Wong.

J. Xu is with Cadence Design Systems, Inc., San Jose, CA 95134 USA (e-mail: jinxu@cadence.com).

P.-N. Guo is with Mentor Graphics Corp., San Jose, CA 95131 USA (e-mail: pn-guo@mentor.com).

C.-K. Cheng is with Mentor Graphics Corp., San Jose, CA 95131 USA, while on leave from the University of California at San Diego, La Jolla, CA 92093-0114 USA (e-mail: ok-cheng@mentor.com).

Publisher Item Identifier S 0278-0070(99)02323-4.

Murata *et al.* [1] proposes the sequence-pair approach for rectangular block placement. The general idea is to first place the blocks on a grid, and then use the longest path algorithm to generate a compacted placement. To determine the block placement, two block name sequences are derived, which correspond to the horizontal and vertical grid lines. Then they introduce a P-admissible solution space of size $(n!)^2 8^n$, where n is the total number of blocks, and apply a simulated annealing method to search for a good solution.

In this paper, we extend the sequence-pair approach described above to arbitrarily sized and shaped rectilinear blocks. The major contribution of our work is to identify admissible sequence-pairs for rectilinear block placement and decompact the compacted placement to be feasible. First, we explore the properties of L-shaped blocks, then decompose arbitrarily shaped rectilinear blocks into a set of sub-L-shaped-blocks. The properties of L-shaped blocks, therefore, are applied to general rectilinear blocks.

To demonstrate the efficiency of our algorithm, we apply it to a randomly generated test case and the modified Microelectronics Center of North Carolina (MCNC) benchmark circuit $ami49$. The experiment results show that the algorithm achieves placements with excellent area utilization.

The rest of the paper is organized as follows. Section II gives preliminaries of sequence-pair approach for rectangular block placement. We examine the properties of rectilinear blocks and then formulate the problem of rectilinear block placement in Section III. Section IV presents the decompaction operations. Section V describes how to identify admissible sequence-pairs for rectilinear block placement. An simulated annealing algorithm and the experimental results are given in Section VI. Finally, Section VII concludes the paper.

II. Preliminaries—Sequence-Pair

The topological relationships between rectangular blocks can be represented by two sequences of the block names. And given the dimensions of each block, a placement can be generated by a compaction operation. For more details, please refer to [1].

III. Problem Formulation

A. Rectilinear Block Partitioning

To handle rectilinear blocks, we partition rectilinear blocks into rectangular blocks.

0278–0070/99$10.00 © 1999 IEEE

(a) (b)

Fig. 1. (a) Feasible versus (b) infeasible partitioning of a T-shaped block.

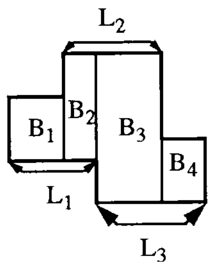


Fig. 2. Partitioning a rectilinear block into sub-L-shaped-blocks.

Definition 1—Subblock: A rectilinear block can be partitioned into a few rectangular blocks. These are called subblocks, to distinguish them from individual rectangular blocks.

In this paper, we label subblocks of rectilinear blocks in upper-cases, and individual rectangular blocks in lower-cases. And suppose $B_1, B_2, \cdots, B_k$ are the subblocks of a rectilinear block, the rectilinear block is denoted by $\{B_1, B_2, \cdots, B_k\}$.

We will explore properties of L-shaped blocks, the most basic rectilinear blocks, first later on. To apply these properties to general rectilinear blocks, we decompose arbitrarily shaped rectilinear blocks into a set of L-shaped-blocks.

A general rectilinear block can be partitioned into a few rectangular subblocks. The partitioning can be done in more than one way, vertically or horizontally. The number of subblocks would be the same. And in order to employ the properties of L-shaped blocks, the partitioning has to be done in such a way that two neighboring subblocks can be grouped into a L-shaped block. Therefore, we partition rectilinear blocks in only one direction, and any two neighboring subblocks should align in the orthogonal direction.

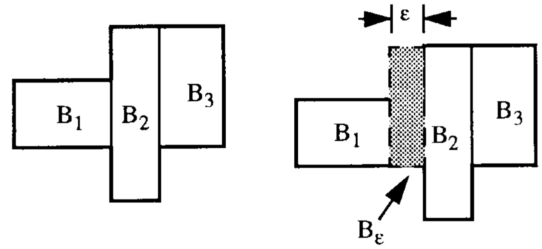
Definition 2—Feasibility of Rectilinear Block Partitioning: A rectilinear block partitioning is feasible if any two neighboring subblocks align in the direction orthogonal to the partition line.

Fig. 1 illustrates an example. Only the partitioning in Fig. 1(a) is feasible.

Fig. 2 shows another example. A rectilinear block is partitioned vertically into four subblocks B_1, B_2, B_3, and B_4. Every two neighboring subblocks align one of their horizontal edges and abut against each other, thus form three L-shaped blocks L_1, L_2, and L_3.

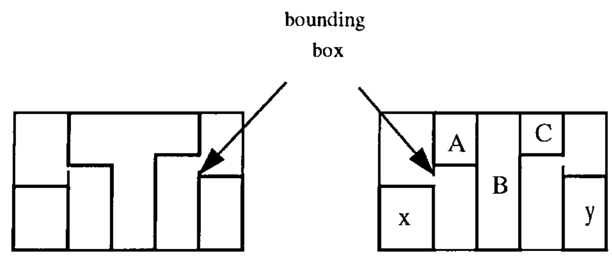
Definition 3—Sub-L-Shaped-Block: A rectilinear block can be represented as a set of L-shaped blocks. These are called *sub-L-shaped-blocks*, to distinguish them from individual L-shaped blocks.

Let B_i be the subblocks, and L_i be the sub-L-shaped-blocks of a rectilinear block, where $L_i = \{B_i, B_{i+1}\}$, $1 \le i < k$, the rectilinear block can be denoted by $\{L_1, L_2, \cdots, L_{k-1}\}$. Thus, the rectilinear block shown in Fig. 2 can be represented by $\{L_1, L_2, L_3\}$.



(a) (b)

Fig. 3. (a) Original and (b) ε-approximation of rectilinear block partitioning.



(a) (b)

Fig. 4. Place a rectilinear block as a rectangular block. (a) Before partitioning and (b) after partitioning.

Some rectilinear blocks, however, cannot be partitioned into sub-L-shaped-blocks, neither vertically nor horizontally, as illustrated in Fig. 3(a). In this case, we can select a side of the rectilinear block and expand it by the size of ε, and add a block B_ε between B_1 and B_2, as shown in Fig. 3(b), so that the rectilinear block becomes L-shape dividable. We call this operation ε-approximation.

Clearly a rectilinear block is in its original shape, if all its sub-L-shaped-blocks are in their original L-shape.

B. Rectilinear Block Placement

We now define a feasible rectilinear block placement as the following.

Definition 4—Feasibility of Rectilinear Block Placement: A rectilinear block placement is feasible iff: 1) it is nonoverlapping; 2) all rectilinear blocks are in their original shape.

Obviously, if all rectilinear blocks are just placed as rectangular blocks, i.e., by their bounding boxes, feasible placements always exist. Fig. 4(a) illustrates a placement of a T-shaped block with two rectangular blocks. By definition, the placement is feasible.

The T-shaped block can be partitioned into three subblocks and denoted by $\{A, B, C\}$, as shown in Fig. 4(b). The sequence-pair corresponding to the placement is $(xABCy, xABCy)$.

Now let us consider L-shaped blocks. A L-shaped block can be partitioned into two subblocks A and B, as shown in Fig. 5. To maintain its original L-shape, subblocks A and B should always align their bottom edges and abut against each other, as stated in the following observation.

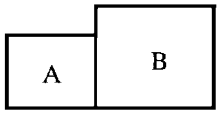


Fig. 5. Partition a L-shaped block into two subblocks.

Observation 1—Feasible Property: In a feasible placement, subblocks of a L-shaped block possess the following property: 1) alignment—the two subblocks align with each other; 2) abutment—the two subblocks abut against each other.

We say a L-shaped block is in its original L-shape if its two subblocks possess the feasible property and a general rectilinear block is in its original shape if all of its sub-L-shaped-blocks possess the feasible property.

Then similar to rectangular block placement, given a sequence-pair of the subblocks, as well as their dimensions, a placement can be generated by a compaction operation. However, unlike rectangular block placement, the placement is not always feasible, even though nonoverlapping, since the rectilinear blocks might not be in their original shape. We thereby formulate the problem of rectilinear block placement as the following.

Rectilinear Block Placement Problem: Given a sequence-pair of rectilinear block subblocks, a compacted placement can be generated. Rectilinear block placement is to find a feasible placement within the compacted chip area.

This implies that: 1) additional operations are needed to restore the original shape of rectilinear blocks in the compacted placement. We call this operation "decompaction" to contrast it with the compaction operation which generates a placement as compacted as possible from a sequence-pair and 2) we need to identify sequence-pairs whose compacted placements can be decompacted to a feasible placements within the initial compacted chip area. We call those sequence-pairs "admissible" sequence-pairs.

In the remaining of the paper, we will present how to decompact a compacted placement to make it feasible and how to identify admissible sequence-pairs which lead to feasible placements with efficient area utilization.

IV. Decompaction

We first examine the admissible property which allows decompaction operations to decompact a compacted placement to a feasible placement within the original compacted chip area, then present the decompaction algorithm.

A. Admissible Property

Fig. 6(a) shows a compacted placement of T-shaped block $\{A, B, C\}$. Evidently, the placement is not feasible.

We are to decompact the placement to make it feasible. We first define the alignment edge and abutment edge of a subblock as the following.

Definition 5—Alignment Edge: The edge along which a subblock of a rectilinear block is to be aligned is called its *alignment edge*.

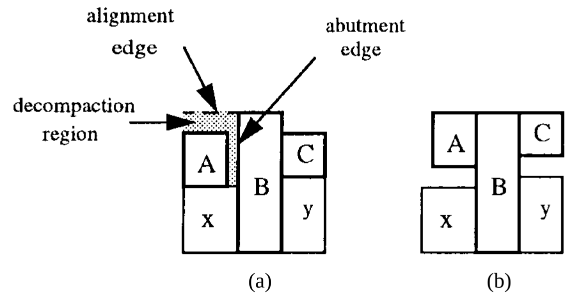


Fig. 6. Decompact an infeasible placement to be a feasible placement. (a) Before decompaction and (b) after decompaction.

Definition 6—Abutment Edge: The edge against which a subblock of a rectilinear block is to be abutted is called its *abutment edge*.

Fig. 6(a) shows the alignment edge and abutment edge of subblock A in the compacted placement. We are to pull the block up to align with, and right to abut against, subblock B. Now we define the decompaction region as the following.

Definition 7—Decompaction Region: Suppose a subblock is to be decompacted to its alignment and/or abutment edges, its decompaction region is the area between its current position and those edges.

The decompaction region of subblock A is illustrated by the shaded area in the figure. We can pull the block up and right, so that the original L-shape of sub-L-shaped-block $\{A, B\}$ is restored. Fig. 6(b) shows the final placement. Clearly, it is feasible.

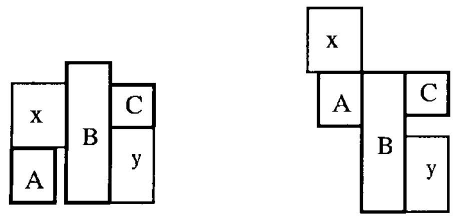
Note that the decompaction operation does not change the overall block topology and chip size. Evidently, this is because there is no other block in the decompaction region and thereby there is always room for decompaction operation to pull a subblock to restore its feasible properties.

Observation 2—Admissible Property: To assure a compacted placement can be decompacted to be a feasible placement within the original compacted chip area, there should not be any block in the decompaction regions.

Consequently, we define an admissible placement and sequence-pair as the following.

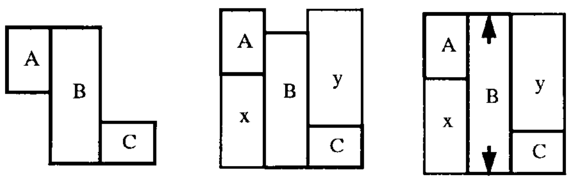
Definition 8—Admissible Placement and Sequence-Pair: A compacted placement is said to be *admissible* if there is no block in the decompaction regions. A sequence-pair is *admissible* if its corresponding compacted placement is admissible.

In general, we can always pull and align the subblocks of rectilinear blocks to restore their original shape and make the placement feasible. The block topology and chip size, however, might be changed. For example, let us consider sequence-pair $(xABCy, AxByC)$. Fig. 7(a) shows the compacted placement. Since block x is in the decompaction region of subblock A, the placement and sequence-pair are inadmissible. If we pull subblock A up to align with subblock B at their top alignment edges and push block x all the way up, the chip size is changed [Fig. 7(b)]. Furthermore, the block topology of the decompacted placement corresponds to another sequence-pair $(xABCy, AByCx)$. Thus the inadmissible sequence-pair is also redundant in the sense that there exists another admissible sequence-pair which leads to



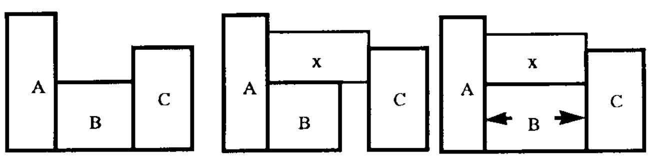
(a) (b)

Fig. 7. Inadmissible compacted placement and sequence-pair. (a) Before decompaction and (b) after decompaction.



(a) (b) (c)

Fig. 8. Expand a subblock to its alignment edge. (a) Z-shape $\{A, B, C\}$, (b) before expansion, and (c) after expansion.



(a) (b) (c)

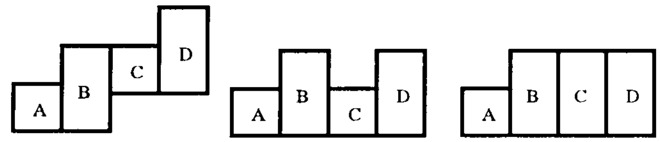
Fig. 9. Expand a subblock to its abutment edge. (a) Concave $\{A, B, C\}$, (b) before expansion, and (c) after expansion.

the exact solution—the same placement. Therefore, excluding inadmissible sequence-pairs does not reduce the solution space for rectilinear block placement. All feasible placements are covered by admissible sequence-pairs.

Now let us look at a placement of a Z-shaped block $\{A, B, C\}$ [Fig. 8(a)] and two rectangular blocks x and y as shown in Fig. 8(b). Clearly the placement is admissible. To maintain the original chip size, however, we must expand subblock B in y-direction [Fig. 8(c)] so that subblock B aligns with subblock A at the top and with subblock C at the bottom. Since the feasible property of subblocks A, B, and C is satisfied after block expansion, the original Z-shape of block $\{A, B, C\}$ is restored and thus the placement becomes feasible.

Similarly, subblock B of a concave rectilinear block $\{A, B, C\}$ is expanded in x-direction (Fig. 9) so that its abutment property is satisfied and the placement becomes feasible. Fig. 10 shows the placement of a W-shaped block $\{A, B, C, D\}$. The placement also becomes feasible after block expansion since the alignment property of the subblocks is satisfied.

We can improve the W-shaped block after expansion by adding extra feasible constraints to restrict subblocks C and D to be above subblock A. However, we demonstrate only pairwise relation for adjacent subblocks instead of triple or higher relation in this paper.



(a) (b) (c)

Fig. 10. W-shaped block placement. (a) W-shape $\{A, B, C\}$, (b) before expansion, and (c) after expansion.

We deem expansion operations reasonable for soft block placement in a hierarchical top-down design. And since the operation would only increase the block sizes, there would always be enough rooms for placement at the next block level in the hierarchy.

Now a decompaction operation can be stated as the following.

Decompaction Operation: Decompaction is a local operation which moves and/or expands a subblock of a rectilinear block within its decompaction region to its alignment and/or abutment edges such that the subblock satisfies the feasible properties.

We will present the decompaction operation in detail in the next section.

B. L-Shaped Block Placement Decompaction

In this paper, we assume that blocks are compacted to the left and bottom sides, i.e., to the lower-left corner, without loss of generality. This will guarantee the alignment or abutment of subblocks at their left and bottom edges in admissible compacted placements—for example, the alignment and abutment of subblocks B and C in Fig. 8(a) are inherently ensured by the compaction operation. For blocks that align or abut at their right or upper edges, however, decompaction operations are needed to restore their original shape.

Definition: Let $\{B_1, B_2, \cdots, B_k\}$ be a rectilinear block, and $M_l = \{B_1, B_2, \cdots, B_l\}$, where $1 < l < k$, be a part of the rectilinear block constituted of the first l subblocks.

Let (w_i, h_i) be the width and height of subblock B_i, and (x_i, y_i) be the coordinates of its lower-left corner in an admissible compacted placement.

1) Alignment Move: Assume we have an L-shaped block $\{B_1, B_2\}$ which should align on the right edges but after compaction

$$x_1 + w_1 \neq x_2 + w_2.$$

The decompaction will move one of the subblocks in x-direction to satisfy the alignment property. We have

- move B_1 right at x-direction by $x_2 + w_2 - x_1 - w_1$ if $x_1 + w_1 < x_2 + w_2$;
- move B_2 right at x-direction by $x_1 + w_1 - x_2 - w_2$ if $x_1 + w_1 > x_2 + w_2$.

Assume we have an L-shaped block $\{B_1, B_2\}$ which should align on the top edges but after compaction

$$y_1 + h_1 \neq y_2 + h_2.$$

The decompaction will move one of the subblocks in y-direction to satisfy the alignment property. We have

- move B_1 up at y-direction by $y_2 + h_2 - y_1 - h_1$ if $y_1 + h_1 < y_2 + h_2$;
- move B_2 up at y-direction by $y_1 + h_1 - y_2 - h_2$ if $y_1 + h_1 > y_2 + h_2$.

2) Abutment Move: Assume we have an L-shaped block $\{B_1, B_2\}$ which should abut in x-direction but after compaction

$$x_1 + w_1 < x_2.$$

The decompaction will move one of the subblocks in x-direction to satisfy the abutment property. We have

- move B_1 right at x-direction $x_2 - x_1 - w_1$.

Assume we have an L-shaped block $\{B_1, B_2\}$ which should abut in y-direction but after compaction

$$y_1 + h_1 < y_2.$$

The decompaction will move one of the subblocks in y-direction to satisfy the abutment property. We have

- move B_1 up at y-direction $y_2 - y_1 - h_1$.

C. Rectilinear Block Placement Decompaction

We can enhance the decompaction operation above for rectilinear block placement by adding expansion operations as follows.

1) Alignment Expansion: Assume we have a shape $\{M_{i-1}, B_i\}$ which should align on the right edges but after compaction

$$x_{i-1} + w_{i-1} \neq x_i + w_i.$$

The decompaction operation finds there is not enough space to move M_{i-1} or B_i in x-direction to satisfy the alignment property. We have

- expand B_{i-1} by stretching its right edge at x-direction by $x_i + w_i - x_{i-1} - w_{i-1}$.

Similarly, assume we have a shape $\{M_{i-1}, B_i\}$ which should align on the top edges but after compaction

$$y_{i-1} + h_{i-1} \neq y_i + h_i.$$

The decompaction operation finds there is not enough space to move M_{i-1} or B_i in y-direction to satisfy the alignment property [Fig. 11(a)]. We have

- expand B_{i-1} by stretching its top edge at y-direction by $y_i + h_i - y_{i-1} - h_{i-1}$ [Fig. 11(b)].

2) Abutment Expansion: Assume we have a y-sliced shape $\{M_{i-1}, B_i\}$ which should abut in x-direction but after compaction

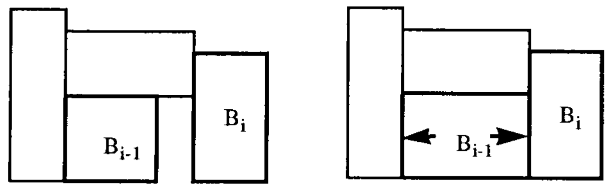
$$x_{i-1} + w_{i-1} < x_i.$$

The decompaction operation finds there is not enough space to move M_{i-1} or B_i in x-direction to satisfy the abutment property [Fig. 12(a)]. We have

- expand B_{i-1} by stretching its right edge at x-direction by $x_i + w_i - x_{i-1} - w_{i-1}$ [Fig. 12(b)].

(a) (b)

Fig. 11. Expand a subblock to its alignment edge. (a) Before expansion and (b) after expansion.



(a) (b)

Fig. 12. Expand a subblock to its abutment edge. (a) Before expansion and (b) after expansion.

```
Procedure Decompaction
    for each rectilinear block R = { B1, ..., Bk
        for each Bi in R - {Bk}
            AL := align_edge(Bi, Bi+1)
            if not move(AL, {B1, ..., Bi}, Bi+1)
                than expand(AL, Bi)
            AB := abut_edge(Bi, Bi+1)
            if not move(AB, {B1, ..., Bi}, Bi+1)
                than expand(AB, Bi)
        end
    end
```

Fig. 13. Decompaction algorithm.

Similarly, assume we have a x-sliced shape $\{M_{i-1}, B_i\}$ which should abut in y-direction but after compaction

$$y_{i-1} + h_{i-1} < y_i.$$

The decompaction operation finds there is not enough space to move M_{i-1} or B_i in y-direction to satisfy the abutment property. We have

- expand B_{i-1} by stretching its top edge at y-direction by $y_i + h_i - y_{i-1} - h_{i-1}$.

Fig. 13 shows the decompaction algorithm.

In the remaining part of the paper, we will present how to identify admissible sequence-pairs which lead to feasible placements with efficient area utilization. We first explore the properties of L-shaped blocks, and then apply them to general rectilinear blocks with sub-L-shaped-block partitioning.

V. Admissable Sequence-Pair

A. L-Shaped Block Placement

Let us consider L-shaped blocks first. In the following discussion we will show that, the admissibility of a placement and the coESIGNrresponding sequence-pair depends on both the orientations of L-shaped blocks and the relative positions of other rectangular blocks in the pair.

We first define the orientation of a L-shaped block as the following.

Definition 9—Orientation of L-Shaped Block: A L-shaped block is in "up" orientation, or "up" oriented, if its subblocks align their bottom edges. Similarly, "down," "left," and "right" orientations are defined.

Therefore, the L-shaped block shown in Fig. 5 is in its "up" orientation. Now let us consider the relative positions of blocks in a sequence-pair. Obviously, with respect to the two subblocks of a L-shaped block in a sequence, a rectangular block may have three positions—before, between or after, as defined as follows.

Definition 10—Relative Position of Rectangular Block: Suppose subblocks A and B of a L-shaped block are the ith and jth blocks in a sequence Π, and a rectangular block is in the kth position of the sequence, the rectangular block is before, between, or after A, B in sequence Π, as shown in (1) at the bottom of the page.

For example, in the sequence $\Pi = (yAxBz)$, blocks x, y, and z are between, before and after subblocks A and B, respectively.

Furthermore, with respect to the two subblocks in a sequence-pair, the rectangular block may have nine configurations, as shown in Fig. 14. Clearly, the admissible property always holds for the first four configurations, because the rectangular block is just placed outside the bounding box of the L-shaped block and will not be placed within the decompaction region of any of the subblocks. Therefore, the first four configurations are always admissible no matter which orientation block $\{A, B\}$ is in. Note that the rectangular block is either before or after subblocks A and B in the corresponding sequence-pairs. Therefore, the following theorem holds.

Theorem 1: If all other blocks are not between the two subblocks of a L-shaped block in either of the two sequences, the sequence-pair is admissible with respect to this L-shaped block.

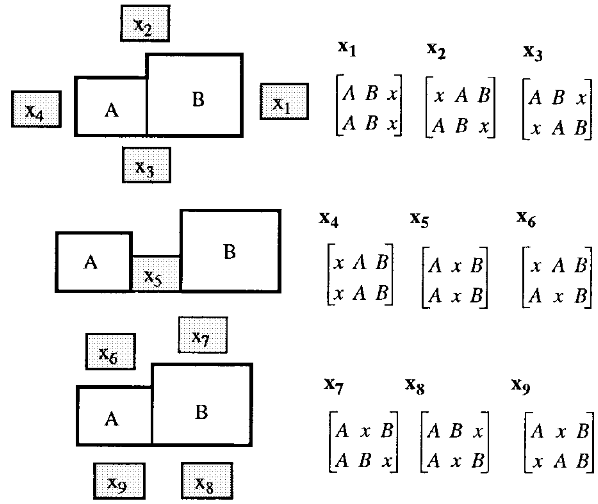


Fig. 14. Nine possible configurations of a rectangular block.

Also clearly, configuration x_5 is always inadmissible since the rectangular block is placed between subblocks A and B and the abutment property will never be satisfied. Therefore, the following theorem holds.

Theorem 2: If a rectangular block is between the two subblocks of a L-shaped block in both of the sequences, the sequence-pair is inadmissible with respect to this L-shaped block.

The other four configurations, where the rectangular block is between subblocks A and B in one of the sequences, are defined as the four classes in the following definition.

Definition 11—Class of Rectangular Block: Given a sequence pair (Π_+, Π_-) and a L-shaped block $\{A, B\}$, a rectangular block is class I, II, III, or IV, as shown in (2) at the bottom of the page.

To illustrate the four classes of a rectangular block defined above, we can draw directed lines between its two positions in the sequence-pair, starting from the one which is between $\{A, B\}$, as shown in Fig. 15(a). If we place these lines in a coordinate system starting from the origin, these four classes are named according to the quadrants in which the directed lines end, as illustrated in Fig. 15(b).

$$\begin{array}{lll} \text{before} & & k < \min(i, j); \\ \text{between} & \{A, B\} \text{ in sequence } \Pi, \text{ if} & \min(i, j) < k < \max(i, j); \\ \text{after} & & k > \max(i, j) \end{array} \tag{1}$$

$$\begin{array}{lllll} \text{class I,} & & \text{after} & & \text{between} \\ \text{class II,} & & \text{before} & & \text{between} \\ & \text{if it is} & & \{A, B\} \text{ in sequence } \Pi_+ \text{ and} & & \{A, B\} \text{ in sequence } \Pi_- \\ \text{class III,} & & \text{between} & & \text{before} \\ \text{class IV,} & & \text{between} & & \text{after} \end{array} \tag{2}$$

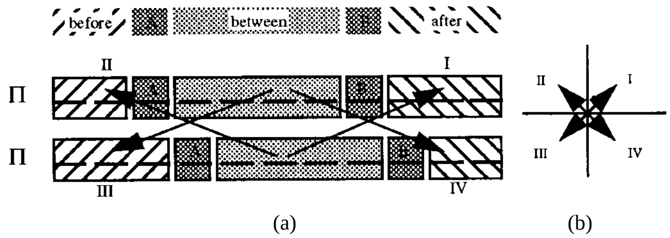


Fig. 15. Classes of a rectangular block with respect to a L-shaped block in a sequence-pair.

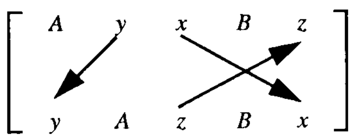


Fig. 16. Classes of rectangular blocks—an example.

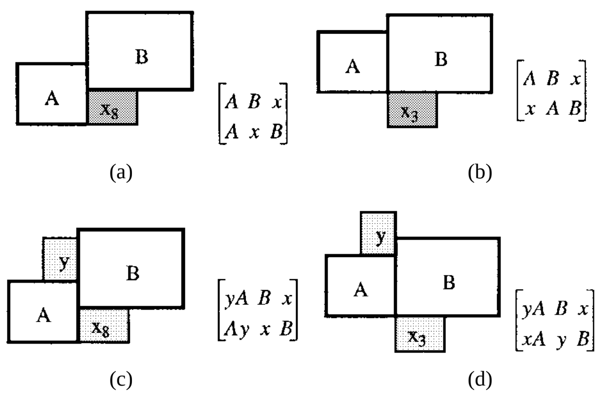


Fig. 17. (a), (c) Inadmissible versus (b), (d) admissible sequence-pairs

Therefore, rectangular block x is class II, IV, I, and III, respectively in the four configurations x_6, x_7, x_8, and x_9 shown in Fig. 14. Fig. 16 shows another example where blocks x, y, and z are class IV, III, and I, respectively.

Now let us consider configuration x_8 in Fig. 14. L-shaped block $\{A, B\}$ is "up" oriented. Fig. 17(a) shows the compacted placement. If we "decompact" the placement—pull subblock A up to restore its alignment property—the configuration is changed to be the same as configuration x_3 [Fig. 17(b)]. Thus the decompacted placement is redundant in the sense that there exists an admissible sequence-pair which leads to the exact solution. Furthermore, sequence-pair (ABx, xAB) guarantees the resulting placement is always feasible, while configuration x_8 does not assure that the admissible property is always satisfied. For example, if there is another block placed on top of subblock A [Fig. 17(c)], we may pull subblock A up to align with subblock B [Fig. 17(d)]. The overall chip size, however, is changed and the resulting placement is also covered by the admissible sequence-pair shown in Fig. 17(d). Therefore, configuration x_8 and the sequence-pair (ABx, AxB)—class I—are inadmissible.

Obviously, block y in Fig. 17 is in configuration x_6—class II. Since the block is placed at the other side of subblocks A and B and does not affect their bottom edge alignment, its configuration is admissible. Similarly we can show that configuration x_9 (class III) is inadmissible while configuration x_7 (class IV) is admissible. Consequently, we have the following observation.

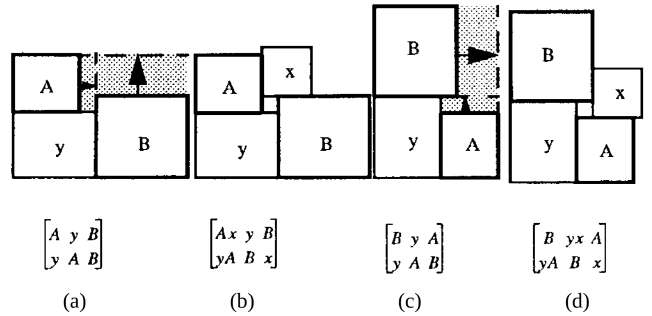


Fig. 18. Inadmissible sequence-pairs of "down" and "right" oriented L-shaped blocks.

Observation 3: Given the "up" orientation of a L-shaped block, a sequence-pair is inadmissible if any other block is either class I or III.

If the L-shaped block is "down" oriented, Fig. 18(a) shows the decompaction regions of subblocks A and B by the shaded areas. To assure their admissible property, no blocks should be in the regions—right to subblock A and above subblock B. Therefore, block x—class IV—results in an inadmissible sequence-pair $(AxyB, yABx)$ [Fig. 18(b)], although sequence-pair (AyB, yAB)—class III—is admissible [Fig. 18(a)].

Similarly, when the L-shaped block is "right" oriented, sequence-pair $(ByxA, yABx)$—class IV—is inadmissible [Fig. 8(d)] while sequence-pair (ByA, yAB)—class III—is admissible [Fig. 8(c)].

Therefore, the following lemmas holds for the other three orientations of a L-shaped block.

Observation 4: Given the "down" orientation of a L-shaped block, a sequence-pair is inadmissible if any other block is either class II or IV.

Observation 5: Given the "left" orientation of a L-shaped block, a sequence-pair is inadmissible if any other block is either class II or III.

Observation 6: Given the "right" orientation of a L-shaped block, a sequence-pair is inadmissible if any other block is either class I or IV.

Therefore, the following theorem holds.

Theorem 3: For placement of a L-shaped block, a sequence-pair is admissible if it satisfies one of the observations above with respect to the L-shaped block.

The theorem can be extent to general L-shaped block placement as the following.

Theorem 4: For L-shaped block placement, a sequence-pair is admissible if it satisfies Theorem 4 for all the L-shaped blocks.

B. General Rectilinear Block Placement

Sequence-Pair of Rectilinear Block Subblocks:

Let us look at the subblocks first. Obviously, no matter how other blocks are placed, these subblocks should always possess the feasible properties in any feasible placements. And

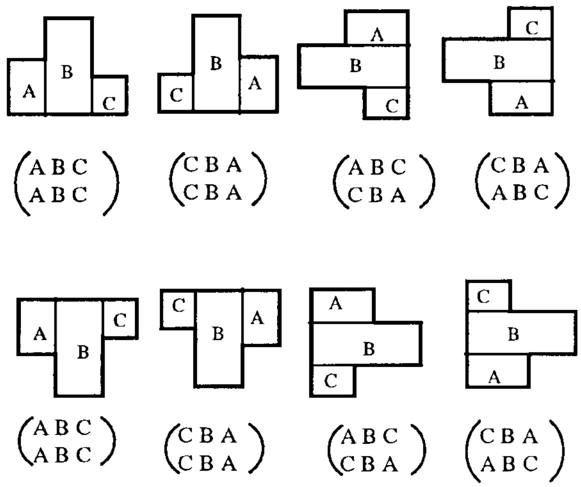


Fig. 19. Admissible sequence-pairs for the subblocks of a T-shaped block.

no matter how the rectilinear block is rotated or reflected, these subblocks should always maintain their original relative positions. Fig. 19 shows the eight reflections and rotations of a T-shaped block and the corresponding sequence-pairs of its three subblocks. As we can see, block B is always between block A and C in all of the placements and sequence-pairs. All other sequences, such as "ACB" and "BCA," are inadmissible. Consequently, the number of admissible sequence-pairs for the three subblocks A, B, and C is reduced substantially. There are only four admissible pairs, each with two reflections, instead of $(3!)^2 = 36$ [1].

In general, we have the following theorem.

Theorem 5: The number of admissible sequence-pairs for the subblocks of a rectilinear block is always only four, instead of $(k!)^2$, where k is the number of the subblocks.

Since general rectilinear blocks are decomposed into a set of sub-L-shaped blocks. The properties of L-shaped blocks can be applied to general rectilinear blocks based on the following transitivity properties.

Transitivity Property for Alignment: Assume $\{B_{i-1}, B_i\}$ and $\{B_i, B_{i+1}\}$ are two adjacent sub-L-shaped-blocks of a rectilinear block. If a sequence-pair is admissible to both $\{B_{i-1}, B_i\}$ and $\{B_i, B_{i+1}\}$, by definition, the alignment property can be restored for both of the L-shaped blocks by applying appropriate decompaction operations on each of them separately. Thus, the alignment property is restored for rectilinear block $\{B_{i-1}, B_i, B_{i+1}\}$ and the sequence-pair is admissible to $\{B_{i-1}, B_i, B_{i+1}\}$.

Transitivity Property for Abutment: The transitivity property is also true for abutment. The abutment property is restored for rectilinear block $\{B_{i-1}, B_i, B_{i+1}\}$ and a sequence-pair is admissible to $\{B_{i-1}, B_i, B_{i+1}\}$, provided that the sequence-pair is admissible to both $\{B_{i-1}, B_i\}$ and $\{B_i, B_{i+1}\}$.

```
Procedure PLACE
    s := initial configuration with random sequence pairs
    T := T0;
    repeat
        count := 0;
        repeat
            count := count + 1;
            nexts := generate(s);
            if cost(nexts) <= cost(s) or f(cost(s), cost(nexts), T) > random(0, 1)
            then s := nexts;
        until count > |s|^1.5;
        T := update(T);
    until the time reaches the limit or Frozen(T);
    output the best placement found;

Function Generate(s)
    apply one of the following move operations to perturb the sequences in s;
        swap two blocks in one of the sequence;
        swap two blocks in both the first and the second sequences;
        rotate one block;
    return modified s;

Function Cost(s)
begin
    C1 := Area_evaluation(s);
    C2 := Wire_length_estimation(s);
    C3 := Inadmissible_penalty(s);
    return W1 * C1 + W2 * C2 + W3 * C3;
end
```

TABLE I
THE EXPERIMENTAL RESULTS TEST CASE

test case	# rectangular	# L-shaped	# others	area	dead space(%)
artificial	2	4	--	2304 (48 x 48)	5.16
ami49	7	21	--	37,391,508 (6314 x 5922)	5.20
ami49	8	17	2	37,897,776 (5,502 x 6,888)	6.47

Therefore, the following theorem holds.

Theorem 6: For placement of an arbitrarily shaped rectilinear block, a sequence-pair is admissible if it is admissible with respect to all the sub-L-shaped-blocks of the rectilinear block.

In general, the following theorem holds.

Theorem 7: For rectilinear block placement, a sequence-pair is admissible if it satisfies Theorem 6 for all the rectilinear blocks.

VI. EXPERIMENTAL RESULTS

Similar to Murata's approach, we also apply a simulated annealing strategy to search the solution space. A carefully selected cooling schedule will converge the annealing process and reach an optimized configuration. We set up the initial temperature to have more than 90% of acceptance rate, and lower the temperature exponentially every step. In each iteration, the number of configuration is 1.5 power of the number of blocks.

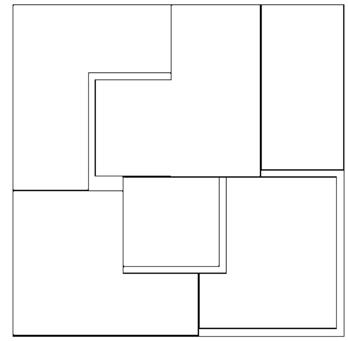
Since inadmissible sequence-pairs might be generated for rectilinear block placement, we make the feasibility of sequence-pairs as part of the cost function used in the annealing process. If an inadmissible pair is identified, we add a penalty to the cost function according to the number of the inadmissible blocks. The larger the function value is, the more admissible violations the configuration has.

In the cost function, the weights for each term is trying to balance the effects each other. The weight of inadmissible penalty is increased exponentially so that we have admissible configuration at the final stage of annealing process. Note that the cost function reflects the distance of an inadmissible block from its desired position. More distance from feasible position, more number of inadmissible blocks counted.

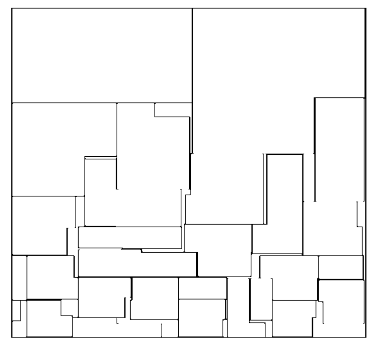
The outline of the algorithm is shown at the bottom of the previous page.

To show the efficiency of our approach, we implement the above algorithms in C language, and run the test cases on Sun Ultra 1 200-MHz workstation. The test cases are generated randomly, where the second and the third test cases are based on the MCNC benchmark circuit $ami49$ by merging blocks to rectilinear shape randomly.

The experimental results are summarized in Table I. The first three columns are the number of block classification for each test case. For example, the third test case contains eight rectangular blocks, 17 L-shaped blocks, and two rectilinear blocks which are neither rectangular nor L-shaped. The area, dimensions and dead space of the final placements are shown in the last two columns in Table I.



(a)



(b)

(c)

Fig. 20. Experimental results. (a) artificial test case (5.16%), (b) modified $ami49$(5.20%), and (c) modified $ami49$(6.47%)

The final placements are shown in Fig. 20. All the results for the test cases are obtained within the given 20-min CPU time limit.

VII. CONCLUSIONS

In this paper, we extend the sequence-pair approach introduced in [1] for rectangular block placement to arbitrarily sized and shaped rectilinear blocks. The properties of L-shaped blocks are examined first, and then arbitrarily shaped rectilinear blocks are decomposed into a set of L-shaped-blocks. The properties of L-shaped blocks, therefore, are applied to general rectilinear blocks. The experiment results show that the algorithm achieves placements with excellent area utilization.

References

[1] H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani, "Rectangular-packing-based module placement," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, 1995, pp. 472–479.

[2] J. Xu, P. N. Guo, and C. K. Cheng, "Rectilinear block placement using sequence-pair," in *Proc. 1998 ACM/IEEE Int. Symp. on Physical Design,* Monterey, CA, Apr. 6–8, 1998, pp. 173–178.

[3] M. Kang and W. W.-M. Dai, "General floorplanning with L-shaped, T-shaped and soft blocks based on bounded slicing grid structure," in *Proc. 1997 ASP-DAC,* 1997, pp. 265–270.

[4] S. Nakatake, H. Murata, K. Fujiyoshi, and Y. Kajitani, "Bounded-slicing structure for module placement," Institute of Electronics, Information and Communication Engineers of Japan, Japan, Tech. Rep., 1994, vol. VLD94, pp. 19–24.

[5] T. C. Lee, "A bounded 2D contour searching algorithm for floorplan design with arbitrarily shaped rectilinear and soft modules," in *Proc. 30th Design Automation Conf.,* 1993, pp. 525–530.

[6] C. K. Cheng and E. S. Kuh, "Module placement based on resistive network optimization," *IEEE Trans. Computer-Aided Design,* vol. CAD-3, pp. 218–225, July 1984.

[7] H. Onodera, Y. Taniguchi, and K. Tamaru, "Branch-and-bound placement for building block layout," in *Proc. 28th Design Automation Conf.,* 1991, pp. 433–439.

[8] H. Shin, A. L. Sangiovanni-Vincentelli, and C. H. Sequin, "'Zone-refining' techniques for IC layout compaction," *IEEE Trans. Computer-Aided Design,* vol. 9, Feb. 1990.

[9] D. F. Wong and C. L. Liu, "Floorplan design for rectangular and L-shaped modules," in *Proc. IEEE Int. Conf. on Computer-Aided Design,* 1987, pp. 520–523.

[10] B. T. Preas and W. M. vanCleemput, "Placement algorithms for arbitrarily shaped blocks," in *Proc. ACM/IEEE Design Automation Conf.,* 1979, pp. 474–480.

Jin Xu (M'98) received the B.S. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 1985, and the M.S. degree from Tsinghua University, Beijing, China, in 1988, both in electrical engineering.

She is a Senior Member of Technical Staff at Cadence Design Systems, Inc., San Jose, CA. Her research interest are in VLSI physical design algorithms, especially in floorplanning and placement.

Pei-Ning Guo (S'98) received the B.S. degree in computer science from the National Taiwan University, Taiwan, the M.S. degree in computer science from New York University, New York, and the Ph.D. degree in computer science and engineering from University of California, San Diego, in 1998.

He joined Mentor Graphics Corp., San Jose, CA, as a Senior Engineer in 1999. His research interests include floorplan and placement approaches for VLSI physical design.

Chung-Kuan Cheng (S'82–M'84–SM'95) received the B.S. and M.S. degrees in electrical engineering from the National Taiwan University, Taiwan, and the Ph.D. degree in electrical engineering and computer science from the University of California, Berkeley, in 1984.

From 1984–1986 he was a Senior CAD Engineer at Advanced Micro Devices Inc., Sunnyvale, CA. In 1986, he joined the University of California, San Diego (UCSD), where he was a Professor in the Computer Science and Engineering Department. Currently, he serves as a Chief Scientist at Mentor Graphics while taking leave from the university. His research interests include network optimization and design automation on microelectronic circuits.

Dr. Cheng has been an Associate Editor of IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS since 1994. He is a recipient of the best paper award, IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, 1997, the NCR Excellence in Teaching award from the School of Engineering, UCSD in 1991.