TOPOLOGY CONSTRAINED RECTILINEAR BLOCK PACKING FOR LAYOUT REUSE

Maggie Zhi-Wei Kang [1] *Wayne Wei-Ming Dai* [2]

[1 2] Dept. of Computer Engineering
University of California, Santa Cruz
maggiek@cse.ucsc.edu
dai@cse.ucsc.edu

ABSTRACT

In this paper, we formulate the problem of *topology constrained rectilinear block packing* in layout reuse. A specific class of rectilinear shaped blocks, *ordered convex rectilinear blocks*, is represented in bounded slicing grid (BSG) structure. The Non-overlapped packing is guaranteed. Based on both sequence pair (SP) and BSG structures, we propose an algorithm to compact the ordered convex blocks under the topological constraints, in which the x and y directions are independently compacted. By augumenting or further partitioning the arbitrary rectilinear blocks into the ordered convex shapes, this method can be extended to handle the topology constrained rectilinear block packing. Furthermore, our recent theoretical progress is briefly reported at the end of this paper, in which arbitrarily rectilinear shaped blocks are represented in SP structure. Three necessary and sufficient constraints are derived on the sequence pair, such that the non-overlapping compaction is guaranteed.

1. BLOCK PACKING IN LAYOUT REUSE

Layout is one of the most complicate steps in IC design and therefore very resource consuming. The design for renewed fabrication processes usually maintains the layout technology but using different design rules. In order to avoid unnecessary waste of time and energy, it has become of practical importance to reuse the layout results accumulated so far in the old fabrication processes. As the increasing complexity of IC design, layout reuse becomes more important.

1.1. Topology Constrained Rectilinear Packing

First we extract devices and group them as a set of macro device blocks. After shrinking the devices and wires, the block sizes are shrunk and shapes are changed. A packing algorithm is required to eliminate the empty space in between without changing the topological relations, which is referred to as *topology constrained rectilinear block packing*. Fig. 1 shows that five blocks in the original placement are sized and compacted together. The topological relation between any two blocks is defined by their pre-placed positions. For example, block A is left to block B as shown in Fig. 1 (b). In this paper, we focus the rectilinear block packing, ignoring the interblock wiring. The incorporation of wiring will be presented in a separate paper.

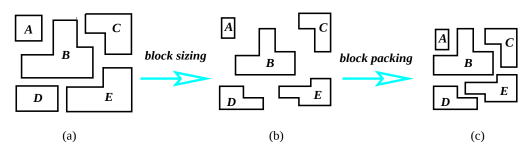


Figure 1. Given the original placement of five macro blocks in (a), the block sizes are shrunk and shapes are changed in (b). In (c), they are compacted together under the same topological relations.

1.2. Major Contribution of Our Work

The interdependency of x and y compaction is the key issue for the optimal packing solution. Recently, Nakatake et al. [1] introduced the bounded slicing grid (BSG) and Murata et al. [2] proposed the sequenced pair (SP) structures to represent the general rectangle packing. Both BSG and SP define the binary relationship for each pair of rectangles, and provide the way to independently compact x and y dimension. [3] applied BSG structure on the general floorplanning problem, in which the L-shaped, T-shaped and soft blocks were considered. [4] indicated the complicate relationship of rectilinear blocks and proposed a SP-based compaction algorithm. Unfortunately the algorithm may leads to overlaps in the final packing.

In this paper, the rectilinear shaped blocks are partitioned into a set of sub-rectangles such that each pair of adjacent sub-blocks form an L-shape. The sub-blocks are individually represented in BSG structure. An algorithm is derived to independently align x and y coordinates of the sub-blocks after BSG packing, such that the original rectilinear shape can be recovered. The related proof shows that the algorithm can recover the exact shape of blocks without causing overlaps if every polygon has *ordered convex* shape.

Furthermore the topological relations of rectilinear polygons can be simply but accurately described using the binary relations of the corresponding sub-rectangles. As such, the constrained packing problem becomes the *constrained BSG assignment problem* : find out a BSG assignment for the rectilinear blocks, in which the topological relations are same with the original placement. Based on both SP and BSG structures, an algorithm is derived to construct such assignment.

In the rest of the paper, Section 2 introduces both BSG and SP structures. Section 3 describes the partition and alignment of rectilinear blocks based on BSG structure. In

Section 4, the necessary and sufficient conditions for the constrained BSG assignment are discussed. A corresponding algorithm is developed, in which the SP structure is used as a easy way to control the topological relationship. Section 5 reports the experimental results and concludes the paper. Finally in the Appendix, we briefly report our recent theoretical progress on the arbitrarily rectilinear block packing using SP structure.

2. BSG AND SP STRUCTURES

Nakatake et al. [1] introduced a meta-grid structure, called bounded slicing grid (BSG), and Murata et al. [2] proposed an equivalent structure, called sequence pair, to represent the general rectangular dissection. Both structures can provide a finite solution space at least one of which is optimal.

2.1. Bounded Slicing Grid Structure (BSG)

BSG is the meta-grid structure as shown in Fig. 2 (a), in which the line segments are called *Bounded Slice Lines*, or BS-lines. The rectangular space surrounded by BS-lines is called *room*. BSG introduces the orthogonal relations to each pair of rooms uniquely. In BSG domain, a packing is represented by an assignment of rectangular blocks to rooms, called *BSG assignment*. This assignment is to map each block to a distinct room, by which the blocks inherit the relationship of the rooms.

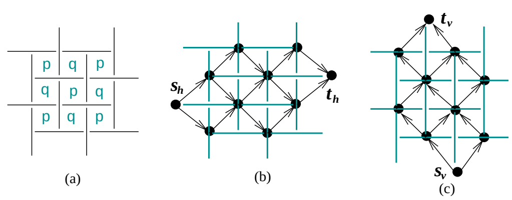


Figure 2. (a) a bounded slicing grid structure, (b) the horizontal acyclic graph G_h, and (c) the vertical acyclic graph G_v.

Two directed acyclic graphs, horizontal graph G_h and vertical graph G_v, are defined to represent the binary relations. The G_h puts vertex on the center of each vertical BS-line as shown in Fig. 2 (b). There is an arc from v_i to v_j if the vertical BS-line corresponding to v_j is right to the vertical BS-line corresponding to v_i and they share the same room. In particular, there is a source s_h connected to all the vertices representing the leftmost BS-lines, and a sink t_h connected from all the vertices corresponding to the rightmost BS-lines. The weight of arc is given by the width of the block assigned to the corresponding room, if the room is occupied. Otherwise the weight is zero. The vertical graph G_v is similarly defined as shown in Fig. 2 (c).

The x-coordinates of blocks are determined using the longest path algorithm. In particular, the overall width equals to the longest path length from the source to the sink in G_h. The y-coordinates and the overall height can be similarly determined in G_v. In such way, the BSG compaction is independently carried out in x and y dimension. Fig. 3 compares three kinds of packing in which the independent packing gives the optimal solution.

2.2. Sequence Pair (SP)

A *sequence pair* for a set of n blocks is a pair of sequences of n symbols which represent blocks. As shown in Fig. 4 (a), an oblique-grid can be constructed for a sequence pair. For every block, the plane is divided by the two crossing

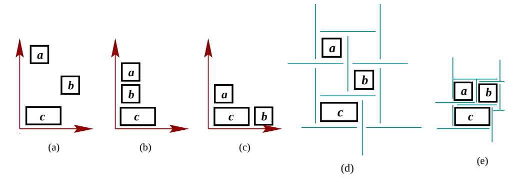


Figure 3. Given three blocks in (a), the x-y packing in (b) is achieved by first compacting x dimension followed by y dimension, while the packing in (c) is achieved by y-x order. Neither (b) nor (c) is the optimal solution. On the other hand, if the three blocks are assigned into BSG as in (d), the optimal packing in (e) can be achieved by independently compacting x and y dimension in BSG structure.

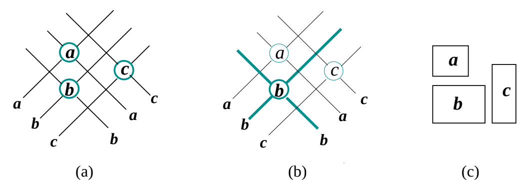


Figure 4. (a) the oblique-grid of sequence pair (abc, bac), (b) the four cones of block b, and (c) the corresponding placement of a, b and c.

slope lines into four cones as shown in Fig. 4 (b). Block a is in the upper cone of block b, then a is above b. Similarly, block c is in the right cone of block b, then c is right to b. In general, equivalent with BSG, SP imposes the binary relations for each pair of blocks :

$$
\begin{array}{lll}
(\cdots a \cdots b \cdots, \quad \cdots a \cdots b \cdots) & \Rightarrow & b \text{ is right to } a, \\
(\cdots b \cdots a \cdots, \quad \cdots a \cdots b \cdots) & \Rightarrow & b \text{ is above } a.
\end{array}
$$

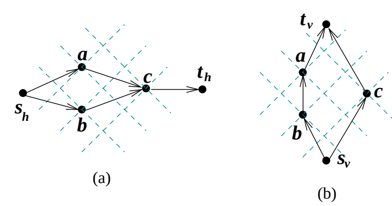


Figure 5. The two acyclic graphs of sequence pair (abc, bac).

Similar to BSG structure, two vertex weighted directed acyclic graphs can be constructed for SP. As shown in Fig. 5 (a), in the horizontal graph G_h, each vertex corresponds to a block, there is an arc from block a to block c if c is right to a. In particular, there is a source s_h connected to leftmost blocks and a sink t_h connected from rightmost blocks. The vertex weight equals to the width of the block. Similarly the vertical graph G_v is constructed as shown in Fig. 5 (b). The x and y coordinates can be determined using the same longest path algorithm.

3. ORDERED CONVEX RECTILINEAR BLOCKS IN BSG

In layout reuse, the blocks can be any rectilinear shape due to the device and wire sizing. We have studied the special cases : L-shaped and T-shaped blocks in BSG structure [3]. For general rectilinear blocks, the similar method could be applied : partitioning a rectilinear polygon into a set of sub-rectangles, each of them is assigned to a distinct BSG room individually. After BSG packing, the coordinates of sub-blocks are aligned to recover the exact shape of the original block. To derive the constraints on the partition and assignment, which are referred to as *aligning rules*, we first discuss the alignment algorithm.

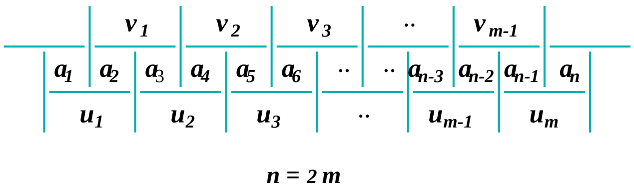


Figure 6. y alignment of sub-rectangles $A = \{a_1, a_2, \cdots, a_n\}$.

In BSG structure, there are two kinds of rooms : *p-typed* and *q-typed* rooms, which are located alternatively as shown in Fig. 2 (a). Each pair of adjacent rooms are alternatively *pq*- or *qp*-adjacent. The horizontal *pq*-adjacent rooms share the bottom BS-line, while *qp*-adjacent rooms share the top BS-line.

3.1. y Alignment

Given a rectilinear shaped block $A = \{a_1, a_2, \ldots, a_n\}$ is assigned into horizontally adjacent BSG rooms as shown in Fig. 6. When the room of a_1 is p-typed and a_n is q-typed, n must be even : $n = 2m$, where m is an integer.

Let y_{u_i} and y_{v_i} denote the y coordinate of BS-lines u_i and v_i, respectively, and h_i denote the height of rectangle a_i. The BSG compaction in y direction has the following relations :

$$\begin{aligned} y_{v_1} &= max(y_{u_1} + h_2, \;\; y_{u_2} + h_3) \\ y_{v_2} &= max(y_{u_2} + h_4, \;\; y_{u_3} + h_5) \\ &\cdots \\ y_{v_{m-1}} &= max(y_{u_{m-1}} + h_{n-2}, \;\; y_{u_m} + h_{n-1}) \end{aligned} \tag{1}$$

The y coordinate of sub-rectangles $a_1, a_2, \ldots, a_n$ can be aligned if the following relations are satisfied:

$$\begin{aligned} y_{u_1} + h_2 &= y_{u_2} + h_3 \\ y_{u_2} + h_4 &= y_{u_3} + h_5 \\ &\cdots \\ y_{u_{m-1}} + h_{n-2} &= y_{u_m} + h_{n-1} \end{aligned} \tag{2}$$

Let y'_{u_i} denote the aligned y coordinate of BS-line u_i, the non-overlapping constraint requires $y'_{u_i} \geq y_{u_i}$, i.e. BS-lines should never be moved downward. The aligned y coordinate of u_1 is given by :

$$\begin{aligned} y'_{u_1} = max(\;& y_{u_1}, \\ & y_{u_2} + h_3 - h_2, \\ & \cdots \\ & y_{u_m} + h_{n-1} - h_{n-2} + h_{n-3} - h_{n-4} + \cdots + h_5 - \\ & h_4 + h_3 - h_2 \;) \end{aligned} \tag{3}$$

Once y'_{u_1} is known, the aligned y coordinate of other BS-lines u_i, where $i > 1$, can be calculated as follows :

$$\begin{aligned} y'_{u_2} &= y'_{u_1} + h_2 - h_3 \\ y'_{u_3} &= y'_{u_2} + h_4 - h_5 \\ &\cdots \\ y'_{u_m} &= y'_{u_{m-1}} + h_{n-2} - h_{n-1} \end{aligned} \tag{4}$$

It can be proved that for each BS-line u_i: $y'_{u_i} \geq y_{u_i}$. Therefore no overlap will occur since the horizontal BS-lines never be moved downward. For the other three cases where both room of a_1 and a_n are p-typed, or a_1 is q-typed while a_n is p-typed, or both a_1 and a_n are q-typed, the similar equations can be derived. Fig. 7(a) shows an example of $y - alignment$.

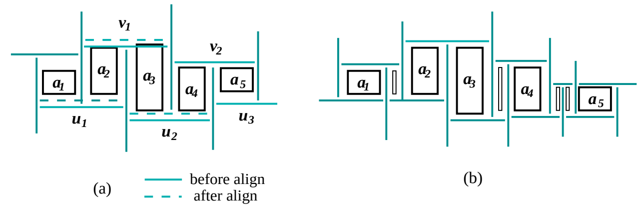


Figure 7. (a) shows an example of y alignment on sub-rectangles $A = \{a_1, a_2, \cdots, a_5\}$. and (b) shows an example of horizontally assignment of these sub-blocks, in which the dummy block with zero width is inserted in the empty room between two adjacent sub-blocks.

We can conclude that the y alignment is applicable for such an assignment that the rooms of sub-rectangles are in the same row, and there is no occupied room in between. The dummy blocks with zero width can be inserted into the empty room between two adjacent sub-blocks, as shown in Fig. 7 (b). Obviously y alignment will not affect the topological relations defined by the BSG structure.

3.2. x Alignment

Given block $A = \{a_1, a_2, a_3, a_4, a_5\}$ in Fig. 8 (a), the aligned x coordinates should satisfy:

$$\begin{aligned} x_{l_2} &= x_{l_1} + w_2 \\ x_{l_3} &= x_{l_2} + w_3 \\ x_{l_4} &= x_{l_3} + w_4 \end{aligned} \tag{5}$$

where x_{l_i} denote the x coordinate of BS-line l_i and w_i the width of block a_i. As such, BS-line l_3 must be exactly right to l_1 by $w_2 + w_3$. While in the horizontal graph G_h, $x_{l_3} = max(x_{l_1} + w_2 + w_3, \; x_{l_1} + w'_2 + w'_2)$, where w'_2 and w'_3 denote the width of block a'_2 and a'_3, respectively. The above condition may not be satisfiable. However if we move a_2 all the way to the right until hitting a_3, followed by a_1 to the right until hitting a_2 as shown in Fig. 8 (b), similarly move a_4 and a_5 to the left, the x coordinates can be aligned. No overlap is caused if the sub-rectangles satisfy :

$$h_1 \leq h_2 \leq h_3 \quad and \quad h_3 \geq h_4 \geq h_5 \tag{6}$$

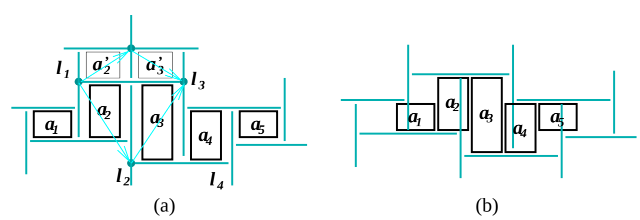


Figure 8. x alignment of sub-rectangles $\{a_1, a_2, a_3, a_4, a_5\}$.

The above property is required by x alignment. Since the blocks are moved only in horizontal direction, x alignment will not affect the vertical relations. For each right-aligned block a_i such as a_1, if any other block b is left to a_i, then b is still left to a_i after moving a_i to the right. On the other hand, $h_i \leq h_{i+1}$ according to Eq. 6. If b is right to a_i, then b is a_{i+1} itself or b is also right to a_{i+1} in the BSG packing. Thus b will be still right to a_i after the right moving of a_i. The similar situation exists for the left-aligned blocks. Therefore the topological relations of BSG packing is preserved by x alignment.

Overall, the x and y coordinates are independently aligned without causing overlaps or changing the relations of BSG packing. The symmetrical alignment is applicable for the vertically adjacent assignments of the sub-rectangles with the similar property as Eq. 6.

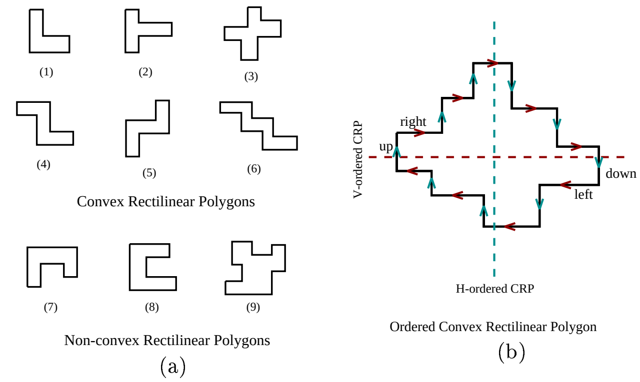


Figure 9. (a) Given a set of rectilinear polygons, in which (1)–(6) are convex shape, while (7)–(9) are concave. (b) When "down" edges are always right to "up" edges, such CRP is *H-ordered*. Similarly when "left" edges are always below "right" edges, such CRP is *V-ordered*. The CRP in (a) (1), (2) and (3) are both H-ordered and V-ordered, while the CRP shown in (a) (4) is only H-ordered and (a) (5) only V-ordered. The CRP in (a) (6) is neither H-ordered nor V-ordered.

3.3. Ordered Convex Rectilinear Polygon

A polygon A is referred to as *convex* rectilinear polygon (CRP) if and only if : given any two points inside A, there exists a shortest Manhatann path inside A, as shown in Fig. 9 (a) (1)–(6). Otherwise *concave* polygons, as shown in Fig. 9 (a) (7)–(9). Given a CRP A, traverse the vertices in clockwise direction and mark each edge by "up", "right", "down" and "left", respectively as shown in Fig. 9(b). A is called *H-ordered* CRP if and only if "down" edges are always right to "up" edges. Symmetrically, A is called *V-ordered* CRP if and only if "left" edges are always below "right" edges. The CRP shown in Fig. 9 (a) (1), (2) and (3) are both H-ordered and V-ordered CRP. On the other hand, the CRP shown in Fig. 9 (a) (4) is only H-ordered and Fig. 9 (a) (5) only V-ordered. However the CRP shown in Fig. 9 (a) (6) is neither H-ordered nor V-ordered.

3.3.1. Partition of Ordered CRPs

An H-ordered CRP A will be partitioned as follows :

1. Put a vertical slicing line on each vertical edge of A, the rectangular space bounded by any two adjacent slicing lines forms a sub-rectangle. In particular, the sub-rectangle bounded by two overlapped slicing lines has zero width as shown in Fig. 10 (a).
2. Visit sliced sub-rectangles from the left to right, and mark each sub-rectangle as shown in Fig. 10 (b).
3. If one sub-rectangle is marked by both p and q, bi-partition it such that the two new sub-rectangles are marked by p and q, respectively as shown in Fig. 10 (c).

We call such partition *H-partition*. Symmetrically the *V-partition* can be defined for V-ordered CRPs.

3.3.2. Property of Ordered CRPs

The following property of H-ordered CRP can be proved :

Lemma 1 *Given an H-ordered CRP is H-partitioned :* $A = \{a_1, a_2, \cdots, a_n\}$*, in which* a_i *is the* i^{th} *leftmost sub-rectangle,*

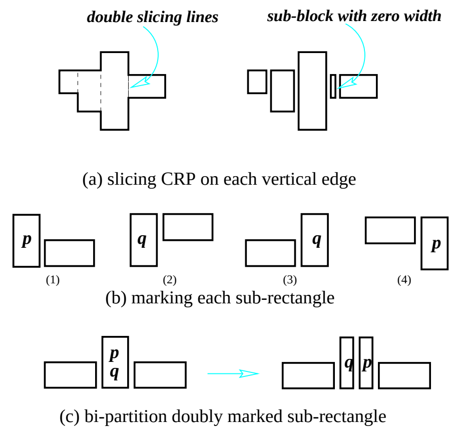


Figure 10. H-partition for an H-ordered CRP.

there exists a sub-rectangle a_k*,* $k \in [1, n]$*, which is referred to as dominant sub-rectangle :*

$$\begin{aligned} h_i &\le h_{i+1}, \quad \text{for } i \in [1,\ k) \\ h_i &\ge h_{i+1}, \quad \text{for } i \in [k,\ n) \end{aligned}$$

where h_i *denotes the height of block* a_i*.*

Similar property can be proved for V-ordered CRPs.

3.3.3. Assignment of Ordered CRPs

Given an H-partitioned CRP : $A = \{a_1,\ a_2,\ \cdots a_n\}$, in which a_i is the i^{th} left-most sub-rectangle. Let r_i denote the BSG room assigned to a_i. We call the BSG assignment of A *H-assignment* if and only if :

1. If a_i is marked by p, the room r_i is p-typed, and if a_i is marked by q, the room r_i is q-typed;
2. The room r_i is on the left of the room r_{i+1}, and they are in the same row;
3. There is no occupied room between r_i and r_{i+1}.

Similarly *V-assignment* can be defined for the V-partitioned CRP. Based on the alignment method discussed above, together with the property of Lemma 1, we can derive the following theorem:

Theorem 1 *Given a placement of a set of blocks with ordered convex rectilinear shape, the* x *and* y *dimension can be independently compacted without overlaps if each H-ordered block is H-partitioned and H-assigned, and each V-ordered block is V-partitioned and V-assigned in BSG structure.*

3.4. Constrained BSG Assignment Problem

The topological relationship of rectilinear blocks may become very complicated. Rather than enumerating all possible relations as done by [4], we can simply but accurately describe such relation using the binary relations of the corresponding sub-rectangles.

If a sub-rectangle of B is right to a sub-rectangle of A, we say B is right to A. Similarly we can define B below A. We call A and B have *consistent relationship* if and only if B is not both right to and left to A, and B is not both above and below A.

Lemma 2 *Any two convex rectilinear polygons have the consistent relationship.*

As such, the topology constrained rectilinear block packing can be transferred to a *constrained BSG assignment problem* : find out a BSG assignment, in which the H- and V-ordered CRPs are H- and V-assigned, respectively. The exact relationship of blocks are guaranteed.

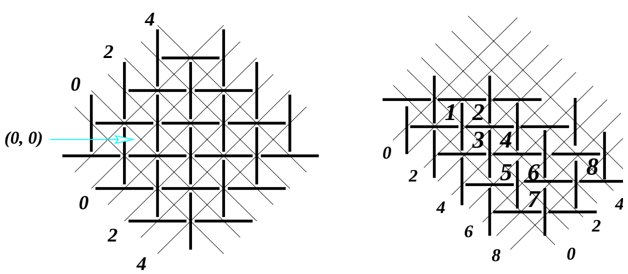
4. CONSTRAINED BSG ASSIGNMENT

We decompose the problem into two steps : (1) construct an initial BSG assignment which provides the equivalent relations with the given placement; (2) adjust the assignment such that each H- and V-partitioned CRP is H- and V-assigned, respectively. As introduced earlier, SP defines the binary relation between each pair of blocks by the order of their symbols in both sequences. Given n rectangular blocks and their topological relations, a sequence pair can be easily constructed in $O(n^2)$ time [2]. In the following, we state a method proposed by S. Nakatake and K. Fujiyoshi, which constructs a BSG assignment for a given SP such that they defines the exact same topological relationship.

4.1. SP-based BSG Assignment

Here we adopt a coordinate system composed by two sets of $+45^0$ and -45^0 slant integer axes, both ordered from the left side as shown in Fig. 11 (a). A room centered at the cross of i_+^{th} positive and i_-^{th} negative axes is referred to by $r(i_+, i_-)$:

Fact 1 *In the slant coordinate system, if $r(0,0)$ is assumed to be a p-typed BSG room, then $r(i_+,\ i_-)$ is a p-typed room if and only if both i_+ and i_- are even. On the other hand, $r(i_+,\ i_-)$ is a q-typed room if and only if both i_+ and i_- are odd.*



(a) (b)

Figure 11. (a) The slant coordinate system of BSG structure, (b) the BSG assignment for the given sequence pair : $,_+ = a_1\ a_2\ a_3\ a_4\ a_5\ a_6\ a_7\ a_8$ and $,_- = a_1\ a_3\ a_5\ a_7\ a_6\ a_4\ a_2\ a_8$.

Let $(,_+, ,_-)$ denote the given sequence pair, and $,_+(a_i)$ denote the index of block a_i in the first sequence $,_+$. Without loss of generality, we assume the first sequence $,_+ = a_1\ a_2\ \cdots\ a_n$, by relabeling if necessary, so $,_+(a_i) = i$ for $i \in [1, n]$. For example, $,_+ = a_1\ a_2\ a_3\ a_4\ a_5\ a_6\ a_7\ a_8$, and $,_- = a_1\ a_3\ a_5\ a_7\ a_6\ a_4\ a_2\ a_8$. The SP-based BSG assignment can be constructed as follows:

1. Placing a dummy block a_0 at the beginning of $,_-$: $a_0\ a_1\ a_3\ a_5\ a_7\ a_6\ a_4\ a_2\ a_8$, and assigning $,_+(a_0) = 0$.
2. Traversing $,_-$ from left to right and grouping every maximal sub-sequence which is either consecutive blocks whose $,_+()$ values are even and decreasing, or consecutive blocks whose $,_+()$ values are odd and increasing. In the above example, $,_- = [a_0]\ [a_1\ a_3\ a_5\ a_7]\ [a_6\ a_4\ a_2]\ [a_8]$. A grouped sub-sequence is called a *group*. The $,_+()$ values of blocks in a group are uniquely even or odd, thus the group is called *even* or *odd* accordingly. For example, $[a_1\ a_3\ a_5\ a_7]$ is an odd group, and $[a_6\ a_4\ a_2]$ is a even group.
3. Placing an empty group between every pair of consecutive even groups or consecutive odd groups : $[a_0]\ [a_1\ a_3\ a_5\ a_7]\ [a_6\ a_4\ a_2]\ [\ ]\ [a_8]$.
4. $,_-(a_i)$ denotes the number of groups in $,_-$ before the group that contains a_i. In this example, $,_-(a_1) = 1$ and $,_-(a_8) = 4$.
5. Assigning block a_i into BSG room $r(,_+(a_i),\ ,_-(a_i))$: a_1 will be assigned into room $r(1,1)$ while a_8 to room $r(8,4)$, as shown in Fig. 11 (b).

The following property can be proved :

Lemma 3 *Each cross $r(,_+(a_i),\ ,_-(a_i))$ is a BSG room, and the relation between each pair of rooms $r(,_+(a_i),\ ,_-(a_i))$ and $r(,_+(a_j),\ ,_-(a_j))$ is exactly the same relation between the corresponding blocks a_i and a_j defined in the given SP.*

Using this method, a BSG assignment of n blocks can be constructed such that it provides the same relations with the given placement. To challenge the second step of the assignment problem, we first derive the necessary and sufficient conditions for H-assignment. Since H- and V-assignment are symmetrical, the similar conditions can be derived for V-assignment.

4.2. Necessary and Sufficient Conditions for H-Assignment

Lemma 4 *In the SP-based assignment, an H-partitioned CRP $A = \{a_1,\ a_2,\ \cdots,\ a_n\}$ is H-assigned if and only if :*

1. *If a_i is p-marked, both $,_+(a_i)$ and $,_-(a_i)$ should be even; if a_i is q-marked, both $,_+(a_i)$ and $,_-(a_i)$ should be odd.*
2. *If a_i and a_j are adjacent sub-blocks in A, and a_i is left to a_j, then $,_+(a_j) - ,_+(a_i) = ,_-(a_j) - ,_-(a_i)$.*

Due to the Fact 1, the first condition above is equivalent to the first requirement of H-assignment defined in Section 3.2.3. In the slant coordinate system, room $r(i_+, i_-)$ and $r(j_+, j_-)$ are in the same row if and only if $j_+ - i_+ = j_- - i_-$. Therefore the second condition above is equivalent to the second requirement of H-assignment. As such, both conditions are necessary for H-assignment. On the other hand, if there is an occupied room $r(,_+(a_k), ,_-(a_k))$ between a_i and a_j :

$$,_+(a_i) < ,_+(a_k) < ,_+(a_j), \qquad ,_-(a_i) < ,_-(a_k) < ,_-(a_j)$$

then both sequences should be like : $a_i \cdot\cdot a_k \cdot\cdot a_j$, which implies that block a_k is right to a_i and left to a_j. If a_k belongs to the same CRP with a_i and a_j, then a_k must be between a_i and a_j, which conflicts to the assumption that a_i and a_j are adjacent. On the other hand, if a_k belongs to a distinct CRP, this CRP will be both left to and right to the CRP of a_i and a_j, which conflicts to the consistent relationship in Lemma 2. Therefore the rooms between a_i and a_j can not be occupied and the third requirement of H-assignment in Section 3.2.3 will be automatically satisfied in the SP-based assignment. As such, the above two conditions are sufficient for H-assignment. In the following, we will propose two operations on SP such that the SP-based assignment satisfies the two conditions of Lemma 4.

4.3. PQ-Adjustment

To satisfy the first condition of Lemma 4, we define an operation called *pq-adjustment*. In the SP-based assignment, $,_+(i)$ and $,_-(i)$ are both even or both odd. Without loss of generality, we assume a_i is a p-marked block, $,_+(i)$ and $,_-(i)$ are both odd. pq-adjustment is carried out by inserting two dummy blocks * into the first sequence $,_+$, one right before and the other right after a_i, respectively, and appending two empty groups at the end of the second sequence $,_-$, as shown in Fig. 12 (a).

After this operation, $,_+(i)$ is increased by one and becomes even. The $,_+()$ values of those blocks after a_i in the first sequence are increased by two. The parity of $,_+()$ values will not be affected except block a_i. Given a_j is the predecessor of a_i in the second sequence, overall there are four possible cases as shown in Fig. 12 (b) :

1. $,_+(j)$ is odd, a_j and a_i are originally grouped together as shown in Fig. 12 (b) (1). The group will split when $,_+(i)$ becomes even after the operation. So the number of groups between a_j and a_i is increased by one.

2. $, _+(j)$ is odd, a_j and a_i are grouped separately, an empty group must be in between as shown in Fig. 12 (b) (2). When $, _+(i)$ becomes even, the empty group is deleted, and the number of groups between a_j and a_i is decreased by one.

3. $, _+(j)$ is even, a_j and a_i are grouped separately, as shown in Fig. 12 (b) (3). When $, _+(i)$ becomes even, which is greater than $, _+(j)$, a_j and a_i will be grouped separately and one empty group is inserted in between, as shown in Fig. 12 (b) (3). The number of groups between a_j and a_i is increased by one.

4. $, _+(j)$ is even, a_j and a_i are grouped separately, as shown in Fig. 12 (b) (4). When $, _+(i)$ becomes even, which is smaller than $, _+(j)$, a_j and a_i will be grouped together, and the number of groups between a_j and a_i is reduced by one.

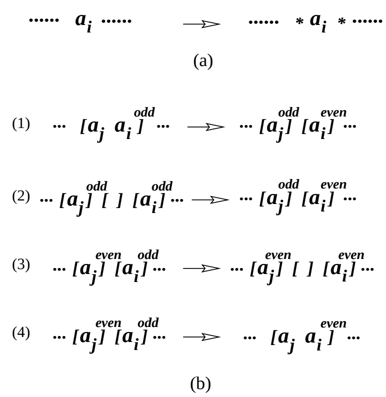


Figure 12. pq-adjustment inserts two dummy blocks in the first sequence, right before and after a_i, respectively as shown in (a). $, _+(a_i)$ is increased by one, and the $, _+()$ values of blocks after a_i will be increased by two. On the other hand, given a_j is the predecessor of a_i in the second sequence, overall there are four possible cases as shown in (b) (1) – (4). It can be derived that the number of groups between a_j and a_i will be increased or decreased by one.

Since the parity of $, _+()$ values are preserved for the other blocks, the groups before a_j in the second sequence will not be changed, and $, _-()$ values remain the same for those blocks before a_i in the second sequence. Due to the changed groups between a_j and a_i, $, _-(a_i)$ is increased or decreased by one and becomes even.

On the other hand, given a_k is the successor of a_i in the second sequence, the similar analysis derives that the number of groups between a_i and a_k will be increased or decreased by one, and the groups after a_k in the second sequence will not be affected. So together with the changed groups between a_j and a_i, the $, _-()$ values are changed by either 0 or 2 for those blocks after a_i in the second sequence. Overall we can conclude :

Lemma 5 *Given block a_i is p-marked, and the corresponding room ($, _+(a_i)$, $, _-(a_i)$) is q-typed, pq-adjustment can adjust the room of a_i to p-typed by simultaneously changing the parity of $, _+(a_i)$ and $, _-(a_i)$. Furthermore the pq-adjustment carried for block a_i will not affect the parity of $, _+()$ or $, _-()$ values of the other blocks.*

Similarly, the pq-adjustment can be applied for q-marked block. In such way, the first condition of Lemma 4 can be satisfied for all marked blocks by carrying out pq-adjustment for each of them, individually.

4.4. Δ-Adjustment

Given adjacent sub-blocks a_i and a_j, a_i is left to a_j, both sequences are $a_i\ \cdots a_j$. We define Δ_+^{ij} and Δ_-^{ij} as follows :

$$\Delta_+^{ij} = , _+(a_j) - , _+(a_i), \qquad \Delta_-^{ij} = , _-(a_j) - , _-(a_i).$$

The second condition of Lemma 4 is equivalent to $\Delta_+^{ij} = \Delta_-^{ij}$. The following Lemma can be proved :

Lemma 6 $|\ \Delta_+^{ij} - \Delta_-^{ij}\ | = 2m$*, where m is an integer.*

When $\Delta_-^{ij} - \Delta_+^{ij} = 2m > 0$, an operation called $, _+$-*adjustment* is applied by consecutively inserting $2m$ dummy blocks * in the first sequence $, _+$, somewhere between a_i and a_j (the exact position will be discussed later), while appending $2m$ empty groups at the end of the second sequence $, _-$.

In the example shown in Fig. 13, given a_1 and a_3 are adjacent sub-blocks, $\Delta_-^{13} - \Delta_+^{13} = 4$. Four dummy blocks are inserted in the first sequence while four empty groups attached at the end of the second sequence. Obviously, $, _+(3)$ is increased by four and accordingly Δ_+^{13} is increased by four. On the other hand, the parity of the $, _+()$ values are not affected due to the even number of dummy blocks, so the groups of the second sequence will not be affected, and Δ_-^{ij} remains the same. Therefore $\Delta_+^{13} = \Delta_-^{13}$ after the $, _+$-adjustment.

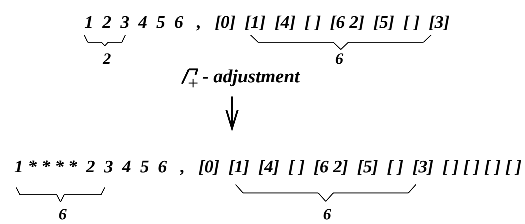


Figure 13. Given a sequence pair and adjacent sub-blocks a_1 and a_3 : $\Delta_-^{13} - \Delta_+^{13} = 4$. $, _+$-adjustment is applied: insert four dummy blocks $*$ into the first sequence between a_1 and a_3, while attach four empty groups [ ] at the end of the second sequence. As such, $\Delta_+^{13} = \Delta_-^{13}$.

Similarly when $\Delta_+^{ij} - \Delta_-^{ij} = 2m > 0$, another operation called $, _-$-*adjustment* is applied : consecutively inserting $2m$ empty groups [ ] in the second sequence $, _-$, somewhere between the groups contain a_i and a_j (the exact position will be discussed later). If a_i and a_j are originally grouped together, the group will split and $2m$ empty groups are inserted in between. On the other hand, $2m$ dummy blocks $*$ are appended at the end of the first sequence $, _+$. So the $, _+()$ values remain the same, while $, _-()$ values of those blocks after the empty groups are increased by $2m$. Therefore Δ_-^{ij} is increased by $2m$ and $\Delta_+^{ij} = \Delta_-^{ij}$. Overall we call both operations Δ-*adjustment.*

4.5. Two Basic Properties of SP

Given two pairs of adjacent blocks (a_i, a_j) and (b_i, b_j), their relative order in a sequence will be one of the following three cases :

- $a_i\ \cdots a_j\ \cdots b_i\ \cdots b_j \Rightarrow a$-pair *separates* from b-pair;
- $a_i\ \cdots b_i\ \cdots a_j\ \cdots b_j \Rightarrow a$-pair *interleaves* with b-pair;
- $a_i\ \cdots b_i\ \cdots b_j\ \cdots a_j \Rightarrow a$-pair *includes* b-pair.

The following two properties can be proved :

Lemma 7 *If a-pair includes b-pair in one sequence of SP, then a-pair separates from b-pair in the other sequence.*

Lemma 8 *If a-pair interleaves with b-pair in one sequence of SP, then a-pair separates from b-pair in the other sequence.*

Since the proofs of the above two Lemmas are very similar, we will only show the first one. Without loss of generality, we assume a_i is left to a_j, and b_i left to b_j. Then both Γ_+ and Γ_- will have : $a_i \cdot\cdot a_j$ and $b_i \cdot\cdot b_j$. If a-pair includes b-pair in the first sequence : $\Gamma_+ = a_i \cdot\cdot b_i \cdot\cdot b_j \cdot\cdot a_j$, then in the second sequence b_i will not be between a_i and a_j. Otherwise, $(a_i\ b_i\ a_j,\ a_i\ b_i\ a_j)$ implies b_i is right to a_i and left to a_j. With this relationship, if b_i belongs to the same CRP with a_i and a_j, b_i will be left to a_i while right to a_j, which conflicts to the assumption that a_i and a_j are adjacent. On the other hand, if b_i belongs to a distinct CRP, the CRP of b_i will be both left to and right to the CRP of a_i and a_j, which conflicts to the consistent relationship of Lemma 2. Therefore, the second sequence must be either $b_i \cdot\cdot a_i \cdot\cdot a_j$ or $a_i \cdot\cdot a_j \cdot\cdot b_i$. The same situation happens for b_j. Thus there are only three possible permutations for the second sequence Γ_-:

$$b_i \cdot\cdot a_i \cdot\cdot a_j \cdot\cdot b_j$$
$$a_i \cdot\cdot a_j \cdot\cdot b_i \cdot\cdot b_j$$
$$b_i \cdot\cdot b_j \cdot\cdot a_i \cdot\cdot a_j$$

If Γ_- is in the first case, we can derive the relation graph as shown in Fig. 14 (a). If a_i and a_j are adjacent sub-blocks as shown in Fig. 14 (b), b_i and b_j should be located at the two shadowed cones, respectively. So they could not be adjacent. Similarly a_i and a_j could not be adjacent given b_i and b_j are adjacent as shown in Fig. 14 (c). Therefore we can conclude Γ_- can only be one of the last two cases, in which a-pair separates from b-pair.

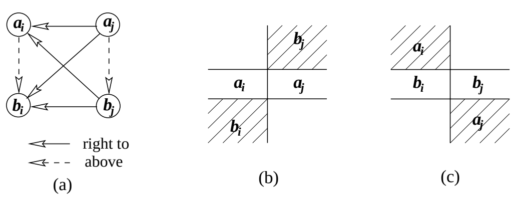


Figure 14. If the relations between blocks a_i, a_j, b_i, and b_j are as shown in (a), assume a_i and a_j are adjacent sub-blocks as shown in (b), the b_i and b_j would be located at the two shadowed cones, respectively. They could not be adjacent. If we assume b_i and b_j are adjacent sub-blocks as shown in (c), a_i and a_j could not be adjacent, either.

Based on the above two properties, every pair of adjacent sub-blocks can be ordered, such that the second condition of Lemma 4 will be satisfied after Δ-adjustment is carried out individually for each pair blocks in this order. Due to the limit of paper length, we skip the detail discuss. The readers can refer to it in [5].

Since Δ-adjustment doesn't change the parity of $\Gamma_+()$ or $\Gamma_-()$ values, the first condition of Lemma 4 will not be affected. Overall we can conclude the following theorem :

Theorem 2 *The necessary and sufficient conditions for H-assignment in Lemma* 4 *can be satisfied by applying pq-adjustment and Δ-adjustment in SP-based BSG assignment.*

The same operations can also be applied to SP-based BSG assignment such that the necessary and sufficient conditions for V-assignment are satisfied.

4.6. One Example of Constrained BSG Assignment

Given the placement of five blocks as shown in Fig. 15, in which four L-shaped blocks are either H-partitioned or V-partitioned. The sequence pair extracted from the placement is :

$$\Gamma_+ = a_1\ a_2\ a_3\ a_4\ a_5\ a_6\ a_7\ a_8\ a_9, \quad \Gamma_- = a_1\ a_8\ a_4\ a_7\ a_5\ a_3\ a_6 a_2\ a_9.$$

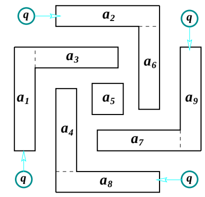


Figure 15. Given the placement of five blocks, in which four L-shaped blocks are either H-partitioned or V-partitioned.

To simplify the notation, we abbreviate a_i as i, and let $*^n$ and $[\ ]^n$ denote the n consecutive dummy blocks and empty groups, respectively. In addition, we ignore the dummy blocks or empty groups attached at the end of the sequences. The SP is grouped as follows :

$$(1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9, \quad [0]\ [1]\ [8\ 4]\ [7]\ [\ ]\ [5]\ [\ ]\ [3]\ [6\ 2]\ [9]).$$

The first condition of Lemma 4 has already been satisfied for blocks a_1 and a_9, so the pq-adjustment is carried out only for a_2 and a_8 :

$$(1\ \underline{*}\ 2\ \underline{*}\ 3\ 4\ 5\ 6\ 7\ \underline{*}\ 8\ \underline{*}\ 9, \quad [0]\ \underline{[1\ 8]\ [4]}\ [7]\ [\ ]\ [5]\ [\ ]\ [3]\ \underline{[6]\ [2\ 9]}).$$

Then Δ-adjustment is carried out for each pair of adjacent sub-blocks : four empty groups are inserted into Γ_- for pair (a_4, a_8), followed by another four empty groups are inserted into Γ_- for pair (a_2, a_6).

$$(1 * 2 * 3\ 4\ 5\ 6\ 7 * 8 * 9, \quad [0]\ [1\ 8]\ [\ ]^4\ [4]\ [7]\ [\ ]\ [5]\ [\ ]\ [3]\ [6]\ \underline{[\ ]^4}\ [2\ 9]).$$

In Γ_+, six dummy blocks are inserted for pair (a_1, a_3), and followed by another six dummy blocks inserted for pair (a_7, a_9).

$$(1 *^7 2 * 3\ 4\ 5\ 6\ 7 * 8 * \underline{*}^6\ 9, \quad [0]\ [1\ 8]\ [\ ]^4\ [4]\ [7]\ [\ ]\ [5]\ [\ ]\ [3]\ [6]\ [\ ]^4\ [2\ 9]).$$

The corresponding BSG assignment is as follows:

$$\begin{array}{lll} a_1 \to (1,\ 1) & a_2 \to (9,\ 17) & a_3 \to (11,\ 11) \\ a_4 \to (12,\ 6) & a_5 \to (13,\ 9) & a_6 \to (14,\ 12) \\ a_7 \to (15,\ 7) & a_8 \to (17,\ 1) & a_9 \to (25,\ 17) \end{array}$$

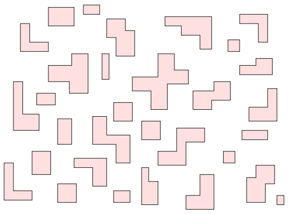
The necessary and sufficient conditions of both H-assignment and V-assignment are satisfied.

5. EXPERIMENTAL RESULTS AND CONCLUSION

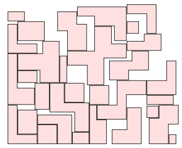
For the application of the layout reuse problem, the constraint of ordered convex shape may be too restrict. However the arbitrary rectilinear shaped block can be transferred or further partitioned into the ordered convex shape(s). And then the algorithm can be extended to handle the general rectilinear blocks.

5.1. Experimental Results

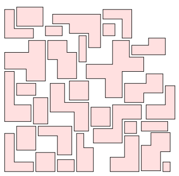
To demonstrate the efficiency of the algorithm presented in this paper, we randomly generated the example shown in Fig. 16 (a), in which all of 31 blocks have ordered convex rectilinear shapes. The packing result achieved by our algorithm is shown in Fig. 16 (b), in which the x and y dimension are independently compacted and the topological relations of blocks in (a) are preserved. On the other hand, we compact the 31 blocks without considering the relation constraints, the packing result shown in Fig. 16 (c) is achieved by first packing x dimension followed by y dimension, and Fig. 16 (d) is the result by first packing y dimension followed by x dimension. Obviously, our algorithm give the best result.



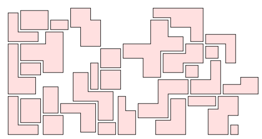
(a) $535 \times 392 = 209,720$



(b) $301 \times 296 = 78,260$



(c) $320 \times 310 = 99,200$



(d) $450 \times 240 = 108,000$

Figure 16. (a) shows the initial placement of 31 rectilinear blocks. By preserving the block relations, our algorithm achieves the packing (b). For comparison, 1-D compactor is applied on the same problem. Without considering the topological constraints, x-y compactor gets result of (c) while y-x compactor gets (d).

5.2. Conclusion

In this paper, we derived an efficient data representation for a special class of rectilinear polygons : ordered convex rectilinear polygons in BSG structure. As such, the x and y dimension can be independently compacted given every polygon is ordered convex shape. By transferring or partitioning arbitrary rectilinear polygons into the ordered convex shapes, the general rectilinear compaction can be dealt with. Furthermore the topology constrained rectilinear block packing is applied to the layout reuse problem. A SP-based BSG assignment is constructed such that the rectilinear blocks can be compacted under the topology constraints.

APPENDIX

OUR RECENT WORK ON ARBITRARILY RECTILINEAR BLOCK PACKING

In the following, we briefly report our recent theoretical results on the rectilinear block packing. The detailed proofs and optimization will be presented in another paper in the near future.

An arbitrarily rectilinear shaped block is partitioned into a set of rectangular sub-blocks: $A = \{a_1, a_2, \ldots a_m\}$, each of them is handled as an individual block in the sequence pair. We introduce two ways to partition a macro block A: slicing the block along every vertical boundary of A from the left to right as shown in Fig. 17 (a), or slicing the block along every horizontal boundary of A from the bottom to top as shown in Fig. 17 (b). We call the first way horizontal partition or *H-partition*, and the second way vertical partition or *V-partition*. In either way, the topology of sub-blocks in A is exactly defined. Corresponding to each partition, there exists one and only one pair of permutations on $\{a_1,\ a_2\ \cdot\cdot a_m\}$, which is referred to as *H-pair* or *V-pair* of A.

Given a sequence pair, the corresponding rectangle packing can be constructed using the longest path algorithm described in Section 1. A post process is then carried out on every macro block A, such that the sub-blocks of A are unioned together to form the original shape. Based on the two directed acyclic graphs of SP, the post process can recover the exact shape of every macro block without causing overlaps, **if and only if** the sequence pair satisfies three constraints. As such, the three constraints are

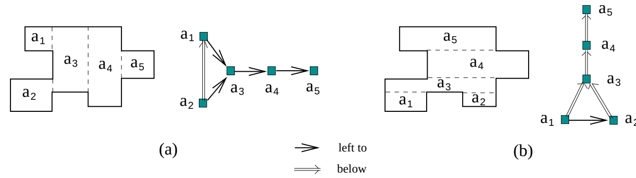


Figure 17. (a) H-partition of macro block A, the H-pair is $(a_1\ a_2\ a_3\ a_4\ a_5,\ a_2\ a_1\ a_3\ a_4\ a_5)$. (b) V-partition of macro block A, the V-pair is $(a_5\ a_4\ a_3\ a_1\ a_2,\ a_1\ a_2\ a_3\ a_4\ a_5)$.

both *necessary* and *sufficient*:

1. for every macro block $A = \{a_1,\ a_2,\ \cdot\cdot\ a_m\}$, the pair of permutations on $a_1,\ a_2,\ \cdot\cdot\ a_m$ equals to H-pair of A when A is H-partitioned, or V-pair of A when A is V-partitioned.
2. for any macro block A, $a_i,\ a_j \in A$ and $c \notin A$, if c is between a_i and a_j in one sequence, then c will not be between a_i and a_j in the other sequence.
3. for any macro block A and B, $a_i,\ a_j \in A$ and $b_k,\ b_l \in B$, if $a_i,\ a_j$ interleaves or includes $b_k,\ b_l$ in one sequence, $a_i,\ a_j$ must seperate $b_k,\ b_l$ in the other sequence.

We call a sequence pair *feasible* if and only if the above three constraints are satisfied.

The theoretical study shows that for any packing of convex rectilinear blocks, there always exists a feasible sequence pair corresponding to it. As such, the optimal packing of convex rectilinear blocks can be guaranteed by exhausting all the feasible sequence pairs.

Furthermore we apply simulated annealing on the rectilinear block packing optimization, in which three local moves are defined on sequence pair to search the solution space continuously. Starting from a feasible sequence pair, each move takes constant time and the modified sequence pair is guaranteed feasible.

Ack

The authors like to thank Shigetoshi Nakatake, H. Murata and K. Fujiyoshi for helpful discussion. Most of this work was done during the first author's summer internship at Ultima Interconnect Technology Inc.

REFERENCES

[1] S. Nakatake, H. Murata, K. Fujiyoshi, and Y. Kajitani, "Bounded-Slicing Structure for Module Placement," Tech. Rep. 313, Institute of Electronics, Information and Communication Engineers of Japan, 1994.

[2] H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani, "Rectangle-Packing-Based Module Placement," in *IEEE/ACM International Conf. on Computer Aided Design*, (San Jose, CA), pp. 472–479, November 1995.

[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. of Asia and South Pacific Design Automation Conf. 1997*, (Chiba, Japan), pp. 265–270, Feburary 1997.

[4] J. Dufour, R. McBride, P. Zhang, and C.-K. Cheng, "A Building Block Placement Tool," in *Proc. 1997 Aisa and South Pacific Design Automation Conf.*, (Chiba, Japan), pp. 271–276, January 1997.

[5] M. Z.-W. Kang and W. W.-M. Dai, "Topology constrained rectilinear block packing for layout reuse," Tech. Rep. UCSC-CRL-97-23, Dept. of Computer Engineering, University of California, Santa Cruz, 1998.