Arbitrary Rectilinear Block Packing Based On Sequence Pair Structure


Maggie Kang Wayne Dai
maggiek@cse.ucsc.edu dai@cse.ucsc.edu
Dept. of Computer Engineering
University of California, Santa Cruz

April 28, 1998

Abstract

Due to layout or specific physical requirements, macro blocks can be in an arbitrary rectilinear shape. Block packing problem will no longer be limited to the rectangle packing. So far no efficient algorithm has been proposed to solve the general rectilinear block packing problem. This paper presents a novel representation method for arbitrary shaped rectilinear blocks in a sequence pair structure. A sequence pair is feasible if an optimal packing of arbitrary rectilinear blocks can be guaranteed for the given sequence pair regardless of the dimensions of the blocks. In this paper, three conditions are derived on a sequence pair which are necessary and sufficient for a sequence pair to be feasible.

Furthermore this paper shows that there always exists a feasible sequence pair for a packing of convex rectilinear blocks. As such, the optimal solution for convex rectilinear block packing can be found by exhausting the finite number of feasible sequence pairs. Three sequence pair operations are developed to incrementally change a solution. Each operation takes linear time and generates a feasible sequence pair. An important theoretical result demonstrates that the optimal solution can always be reachable through a finite times of the sequence pair operations. Therefore a stochastic search based on the three operations can search the feasible solution space both continuously and exhaustively. In such a way, for the first time, the arbitrary shaped rectilinear block packing is solved by using a sequence pair structure.

1 Introduction

Most of the traditional floorplanning or placement algorithms considered only rectangular shaped macro blocks. The slicing structure was proposed to represent the block placement by recursively dissecting the rectangular plane into two parts using either horizontal or vertical line [1]. Corresponding to the slicing structure, Wong and Liu proposed a data representation called normalized Polish expression, which enables the efficient local search [2]. As the increase of routing layers, most of channel routing is being replaced by area routing. A block placement becomes more like a block packing problem, and the wasted
area introduced by a slicing structure becomes more evident. As such, the non-slicing block packing becomes more attractive.

Murata et al. [3] introduced a sequence pair (SP), and Nakatake et al. [4] proposed a bounded slicing grid structure (BSG) to represent the general rectangle packing. Both SP and BSG define the binary relationship for each pair of rectangular blocks, and provide a way to independently compact the $x$ and $y$ direction.

Due to layout or specific physical requirements, macro blocks can be in an arbitrary rectilinear shape. Few previous works have studied the rectilinear shaped blocks where the most noticeable one is the bounded 2D contour searching algorithm proposed by [5]. The arbitrarily rectilinear shaped blocks are represented by a set of four linear profiles, each one of them specifies the profile viewed from one side of the block. Given the original placement of blocks, the algorithm iteratively compacts the blocks along a certain direction, in which the 2D contour searching is carried out on the profiles of the compacted design. As we can see, this compaction method cannot be applied on block packing problem due to the complicated geometrical calculation.

A BSG-based method was proposed by [6] to pack rectangular, L-shaped, T-shaped, and soft blocks. The complicated relationship among rectilinear blocks was indicated and a SP-based algorithm was presented by [7]. Unfortunately the algorithm does not converge to the feasible solutions, overlaps may exist in the packing solution. Most recently, an algorithm based on both BSG and SP structures was proposed by [8], in which the topology constrained block packing can be handled given that blocks belong to a specific class of convex rectilinear shapes. An SP-based algorithm was proposed by [9] to pack the mountain-shaped blocks. However it was not guaranteed that the algorithm could always terminate. Based on BSG structures, an algorithm was proposed by [10] where each block is sliced into a set of rectangular sub-blocks, one of them is selected to be a master and the others slaves. Only the master sub-block is assigned into BSG domain. After the compaction, the slaves are attached with the master. A post-process eliminates overlaps by pushing the neighboring blocks away. Obviously the optimal solution is not guaranteed to be included, and the post process takes much more time than the BSG compaction itself.

![Diagram](image_url)

Figure 1: Each rectilinear shaped macro block, $A$, is partitioned into a set of rectangular sub-blocks: $a_1 a_2 \ldots a_n$, each of them is individually represented in a sequence pair as a unit block. After the unit blocks are compacted in the $x$ and $y$ directions, a post process aligns the $x$ and $y$ coordinates of the sub-blocks such as to recover the original macro shapes.

In this paper, a novel representation method is proposed for arbitrary shaped rectilinear blocks in a sequence pair structure. A sequence pair is feasible if an optimal packing of arbitrary rectilinear blocks for the given sequence pair can always be guaranteed regardless of the dimensions of the blocks. Three conditions on sequence pair are derived which are necessary and sufficient for a sequence pair to be feasible. The theoretical proof points
out that there always exists a feasible sequence pair corresponding to a packing of the convex rectilinear blocks. As such, an optimal solution for the convex block packing can be found by enumerating the finite feasible sequence pairs. Stochastic search is applied, and three sequence pair operations are defined such as to continuously search the feasible solution space. An important theorem shows that starting from any feasible sequence pair, the feasible sequence pair which yields the optimal packing can always be reachable through a finite steps of the operations. Therefore the stochastic search based on the three operations searches the feasible solution space both continuously and exhaustively.

In the following, the sequence pair structure is first introduced in Section 2. Then Section 3 describes the representation method for arbitrary rectilinear blocks in a sequence pair. The concept of feasible sequence pairs is defined, and three conditions on a sequence pair are presented. Section 4 shows that the three conditions are necessary and sufficient for a sequence pair to be feasible. In Section 5, it is proven that there always exists a feasible sequence pair for a packing of convex rectilinear blocks. Based on this fact, Section 6 applies a stochastic search on the convex block packing. The experimental results are reported in Section 7 followed by the conclusion of this paper.

## 2 Sequence Pair (SP) Structure

To clarify the notation, we call the rectangular blocks represented in sequence pair unit blocks. A sequence pair for a set of $n$ unit blocks is a pair of sequences of $n$ symbols which represent the unit blocks: $(\Gamma_1, \Gamma_2)$. Given a sequence pair $(a\ b\ d\ e\ c\ f, \ c\ b\ f\ a\ d\ e)$, an oblique grid can be constructed as shown in Fig. 2 (a): $45^0$ slope lines are named from left to right by the symbols in the first sequence $\Gamma_1$, and $-45^0$ slope lines are similarly named by the symbols in the second sequence $\Gamma_2$. Each unit block is placed at the cross of the two slope lines which are named by the same symbol corresponding to the unit block. As shown in Fig. 2 (b), the plane can be divided by the two crossing slope lines into four cones for any unit block $b$. Unit $a$ is in the upper cone of $b$, then $a$ is above $b$. Similarly, unit $d$, $e$ and $f$ is in the right cone of $b$, then they are right to $b$. In general, the sequence pair imposes the relationship between each pair of unit blocks as follows:

$$\begin{align*}
  (\cdots a \cdots b \cdots, \cdots a \cdots) & \Rightarrow a \text{ is left to } b, \\
  (\cdots b \cdots a \cdots, \cdots a \cdots) & \Rightarrow a \text{ is below } b.
\end{align*}$$

Given the sequence pair, a horizontal directed graph $G_h$ is derived as shown in Fig. 3 (a). Each vertex corresponds to a unit block, there is an arc from unit $a$ to $d$ if and only if $a$ is left to $d$. In particular, there is a source $s_h$ connected to each leftmost unit and a sink $t_h$ connected from each rightmost unit. Each vertex has a weight which equals to the width of the corresponding unit block. The vertical graph $G_v$ can be similarly derived as shown in Fig. 3 (b).

Both $G_h$ and $G_v$ are vertex weighted directed acyclic graphs, the packing of unit blocks can be obtained by simply applying the well-known longest path algorithm on both graphs. The $x$ and $y$ coordinates of each unit block are determined by the longest path from source to the vertex of the unit in $G_h$ and $G_v$, respectively. Similarly, the width and height of the overall packing can be determined by the source-to-sink longest path of $G_h$ and $G_v$.

For any two unit blocks, there is always an arc in either $G_h$ or $G_v$, but not both. Due to this fact, the $x$ and $y$ coordinates can be independently determined, and the resultant packing is guaranteed not to contain any overlap. Since the width and the height are independently minimum, the resultant packing is optimal for the given sequence pair. The longest path calculation can be done in the time proportional to the number of arcs in the graphs.
Figure 2: (a) Oblique grid of \((a\ b\ d\ e\ c\ f,\ c\ b\ f\ a\ d\ e)\), (b) the four cones of block \(b\), and (c) the corresponding packing of the six blocks.

Figure 3: The two vertex weighted directed acyclic graphs derived from the sequence pair \((a\ b\ d\ e\ c\ f,\ c\ b\ f\ a\ d\ e)\), in which the transitive arcs are deleted for the simplification.
As a coding scheme, sequence pair becomes an ideal data representation of rectangular block packing due to following facts [3]:

- Quick evaluation of packing area based on sequence pair: $O(n^2)$ time where $n$ is the number of unit blocks.
- Easy incremental change of packing topology using sequence pair: switch two units in the first or second sequence.
- There always exists a sequence pair corresponding to a rectangle packing, and vice versa.

Therefore the optimal packing can always be found by exhausting the finite number $(n!)^2$ of sequence pairs.

## 3 Arbitrarily Rectilinear Blocks in Sequence Pair

### 3.1 H, V-Partition

Let $A$ denote an arbitrary shaped rectilinear macro block. $A$ can be partitioned into a set of rectangular sub-blocks by slicing $A$ from the left to right along every vertical boundary of $A$. As shown in Fig. 4 (a), the partition is referred to as a horizontal partition or H-partition. Similarly $A$ can be vertically partitioned or V-partitioned as shown in Fig. 4 (b). Given that $A$ is H-partitioned or V-partitioned into $a_1, a_2, \cdots a_n$, each sub-block $a_i \in A$ is individually represented in SP as a unit block. We call the pair of permutations on $a_i \in A$ in the sequence pair a permutation pair of $A$.

![H-Pair and V-Pair](image)

Figure 4: (a) H-partition slices block $A$ on every vertical boundary from the left to right. (b) V-partition slices block $A$ on every horizontal boundary from the bottom to top.

For any two unit blocks $a$ and $b$ placed in a plane without overlaps, let $u_a$ and $l_a$ denote the upper and lower boundary of block $a$, respectively. Block $a$ is left of $b$ if the right boundary of $a$ is left of the left boundary of $b$. If $[l_a, u_a] \cap [l_b, u_b] \neq \emptyset$ as shown in Fig. 5 (a), $a$ is strictly left of $b$. The permutation pair of $a, b$ is $(a, b, a)$. Otherwise $[l_a, u_a] \cap [l_b, u_b] = \emptyset$ as shown in Fig. 5 (b), $a$ is both left of and below $b$, and two permutation pairs of $a, b$ are possible.

Given that macro block $A$ is either H-partitioned or V-partitioned, the topology between the sub-blocks of $A$ is strictly defined. Accordingly there is exactly one permutation pair of $A$ corresponding to the partition. The following Lemma is true:

**Lemma 1** Given an arbitrary rectilinear macro block $A$, there exists only one permutation pair of $A$ corresponding to the H-partition of $A$, which is referred to as an H-pair of $A$. Similarly
Figure 5: For any two unit blocks $a$ and $b$ placed in a plane without overlaps, $a$ is left of $b$ if the right boundary of $a$ is left of the left boundary of $b$. In (a), $[l_a, u_a] \cap [l_b, u_b] \neq \emptyset$, $a$ is strictly left of $b$. The permutation pair of $a$, $b$ is $(a \ b \ a \ b)$. In (b), $[l_a, u_a] \cap [l_b, u_b] = \emptyset$, $a$ is both left of and below $b$, and two permutation pairs are possible.

there exists only one permutation pair of $A$ corresponding to the V-partition of $A$, which is referred to as a V-pair.

In the example shown in Fig. 4, the H-pair of $A$ is $(a_1 \ a_2 \ a_3 \ a_4 \ a_5, \ a_2 \ a_1 \ a_3 \ a_4 \ a_5)$ and V-pair is $(a_5 \ a_4 \ a_3 \ a_1 \ a_2, \ a_1 \ a_2 \ a_3 \ a_4 \ a_5)$.

3.2 X, Y-Alignment

Given a sequence pair representing topological relationships among unit blocks of rectilinear macro blocks, the unit blocks are compacted left and downward using the longest path algorithm in the $x$ and $y$ directions, respectively. If any unit block is further moved left or downward after the compaction, overlaps will occur. In other words, $X$, $Y$-alignment can only move unit blocks right and upward. Given a macro block $A$ is H-partitioned as shown in Fig. 6 (a), and the sequence pair is $(c \ a_1 \ a_2 \ e \ a_3 \ a_4 \ d \ a_5 \ f, \ e \ a_2 \ a_1 \ a_3 \ f \ c \ a_4 \ a_5 \ d)$, the unit blocks are compacted as shown in Fig. 6 (b). In the $x$ direction, $a_5$ is the right-most sub-block of $A$. $X$-Alignment ($A$) freezes $a_5$ and moves $a_4$ to the right until it hits $a_5$, then moves $a_3$ to the right until it hits $a_4$, and so on. After the right move of $a_1$ and $a_2$, the sub-blocks of $A$ are aligned together in the $x$ direction as shown in Fig. 6 (c). In the $y$ direction, the highest sub-block of $A$ is $a_4$. $Y$-Alignment ($A$) freezes $a_4$, and moves the other sub-blocks of $A$ upward to align with $a_4$ as shown in Fig. 6 (d). When macro block $A$ is V-partitioned, $X$, $Y$-alignment can be symmetrically carried out. Obviously the following Theorem is true:

**Theorem 1** Given a sequence pair representing topological relationships among unit blocks of rectilinear macro blocks, $X$, $Y$-Alignment yields an optimal packing for the sequence pair.

Given a sequence pair, the shapes of macro blocks may not be recovered by $X$, $Y$-Alignment.

**Definition 1** A sequence pair is feasible if after $X$, $Y$-Alignment, the shapes of rectilinear blocks are guaranteed to be recovered regardless of the dimensions of the blocks.

**Definition 2** A sequence pair is infeasible if it is not feasible, e.g. the shapes of rectilinear blocks may not be recovered after $X$, $Y$-Alignment.
Figure 6: (a) Macro block $A$ is H-partitioned into $a_1$ $a_2$ $a_3$ $a_4$ $a_5$. (b) Given a sequence pair $(c$ $a_1$ $a_2$ $e$ $a_3$ $a_4$ $d$ $a_5$ $f$, $e$ $a_2$ $a_1$ $a_3$ $f$ $c$ $a_4$ $a_5$ $d)$, the unit blocks are compacted left and downward using the longest path algorithm in the $x$ and $y$ directions, respectively. (c) In the $x$ direction, $a_5$ is the right-most sub-block of $A$. X-Alignment ($A$) freezes $a_5$ and moves $a_4$ to the right until it hits $a_5$, then moves $a_3$ to the right until it hits $a_4$, and so on. After the right move of $a_1$ and $a_2$, the sub-blocks of $A$ are aligned together in the $x$ direction. (d) In the $y$ direction, the highest sub-block of $A$ is $a_4$. Y-Alignment ($A$) freezes $a_4$, and moves the other sub-blocks of $A$ upward to align with $a_4$. 
Accordingly we define the necessary and sufficient conditions on sequence pair as follows:

**Definition 3** A condition is necessary iff when the condition is not satisfied, the sequence pair is infeasible.

**Definition 4** A set of conditions are sufficient iff when the set of conditions are satisfied, the sequence pair is feasible.

![Diagram](image)

FSP : feasible sequence pairs.
NSP : sequence pairs which satisfy the necessary conditions.
SSP : sequence pairs which satisfy the sufficient conditions.
NSSP : sequence pairs which satisfy the necessary and sufficient conditions.

Figure 7: (a) Let FSP denote a set of feasible sequence pairs, NSP the sequence pairs which satisfy the necessary conditions, and SSP the sequence pairs which satisfy the sufficient conditions, then $SSP \subseteq FSP \subseteq NSP$. (b) If we can derive a set of conditions which are necessary and sufficient, the solution space for feasible sequence pairs can be completely characterized by the necessary and sufficient conditions: a sequence pair is feasible if the conditions are satisfied, otherwise infeasible.

Let FSP denote a set of feasible sequence pairs, NSP the sequence pairs which satisfy the necessary conditions, and SSP the sequence pairs which satisfy the sufficient conditions, then $SSP \subseteq FSP \subseteq NSP$ as shown in Fig. 7 (a). If the necessary conditions are not met, the sequence pair is infeasible. If the necessary conditions are met, the sequence pair may or may not be feasible. On the other hand, if the sufficient conditions are met, the sequence pair is feasible. If the sufficient conditions are not met, the sequence pair may or may not be feasible. As such, necessary conditions can not completely characterize the solution space for feasible sequence pairs, neither can sufficient conditions.

If we can derive a set of conditions which are necessary and sufficient, as shown in Fig. 7 (b), the solution space for feasible sequence pairs can be completely characterized by the necessary and sufficient conditions: a sequence pair is feasible if the conditions are satisfied, otherwise infeasible. In the following, we first present three conditions on sequence pair, and later we will prove that the three conditions are both necessary and sufficient.

### 3.3 Three Conditions on Sequence Pairs

Before presenting the detailed conditions, we first define four relations for the unit blocks in a sequence pair.
- Given three unit blocks $a_i, a_j \in A$ and $c \notin A$, if $c$ is between $a_i$ and $a_j$ in both sequences, for example $(a_i \ c \ a_j, \ a_i \ c \ a_j)$, we call $c$ interrupts $a_i$ and $a_j$.

- Given two pairs of unit blocks $a_i, a_j \in A$ and $b_i, b_j \in B, A \neq B$. In the first sequence:
  
  $a_i \cdots a_j \cdots b_i \cdots b_j$, we call $(a_i, a_j)$ and $(b_i, b_j)$ as separates of each other.
  
  $a_i \cdots b_i \cdots a_j \cdots b_j$, we call $(a_i, a_j)$ and $(b_i, b_j)$ as interleaves of each other.
  
  $a_i \cdots b_i \cdots b_j \cdots a_j$, we call $(a_i, a_j)$ as covering $(b_i, b_j)$.

- Similarly, the separate, interleave and cover relations can be defined in the second sequence.

The first condition is referred to as condition-1:

For any H-partitioned macro $A$, the permutation pair of $A$ equals the H-Pair of $A$, similarly for any V-partitioned $A$, the permutation pair of $A$ equals the V-pair of $A$.

The second condition is referred to as condition-2:

Any two units $a_i, a_j \in A$ are not interrupted by a unit $c \notin A$.

Finally the third condition is referred to as condition-3:

Any two pairs of unit blocks $a_i, a_j \in A$ and $b_i', b_j' \in B, A \neq B$. $(a_i, a_j)$ separates $(b_i', b_j')$ in the first or second sequence.

Overall the three conditions are called 3-conditions.

4 3-Conditions Are Necessary and Sufficient

4.1 3-Conditions Are Necessary

Given the sequence pair which does not satisfy condition-1, without loss of generality, we assume the permutation pair of an H-partitioned macro block $A$ does not equal the H-pair of $A$. There must exist two sub-blocks $a_i, a_j \in A$, whose permutations in SP are different from those in the H-pair of $A$. For example, sub-block $a_1$ is left of $a_2$ in the H-partitioned $A$, the H-pair of $A$ is $(a_1 \ a_2, \ a_1 \ a_2)$. On the other hand, the permutations of $a_1, a_2$ in the sequence pair is $(a_1 \ a_2, \ a_2 \ a_1)$. X, Y-Alignment preserve the topological relationships defined in the sequence pair, block $a_1$ will be above $a_2$ after the alignment and the shape of macro block $A$ cannot be recovered. Therefore condition-1 is a necessary condition.

Figure 8: (a) Unit block $c$ is right of $a_i$ while left of $a_j$, $a_i$ and $a_j$ may not be aligned in the $x$ direction. (b) Block $c$ is below $a_i$ while above $a_j$, $a_i$ and $a_j$ may not be aligned in the $y$ direction.
Given the sequence pair does not satisfy condition-2, there must exist two unit blocks \( a_i, a_j \in A \) interrupted by a unit block \( c \notin A \). The sequence pair will be either \((a_i \cdots c \cdots a_j, a_i \cdots c \cdots a_j)\) or \((a_i \cdots c \cdots a_j, a_j \cdots c \cdots a_i)\). In the first case, \( c \) is right of \( a_i \) while left of \( a_j \), \( a_i \) and \( a_j \) may be not aligned in the \( x \) direction as shown in Fig. 8 (a). While in the second case, \( c \) is below \( a_i \) while above \( a_j \), \( a_i \) and \( a_j \) may be not aligned in the \( y \) direction as shown in Fig. 8 (b). Therefore condition-2 is also a necessary condition.

![Diagram](image)

**Figure 9:** (a) In the horizontal graph \( G_h \), if unit block \( a_i \in A \) is left of unit \( b_v \in B \), \( A \neq B \), and unit block \( a_j \in A \) is right of unit \( b_v \in B \), \( a_i, a_j \) and \( b_v \), \( b_v \) are \( H \)-crossed each other. (b) In the vertical graph \( G_v \), if \( a_i \) is below \( b_v \), and \( a_j \) is above \( b_v \), \( a_i, a_j \) and \( b_v \), \( b_v \) are \( V \)-crossed each other.

In the following, we refer two unit blocks \( a_i, a_j \in A \) as \( a \)-pair. Similarly two unit blocks \( b_v, b_v \in B \) as \( b \)-pair, where \( A \neq B \). If \( a_i \) is left of \( b_v \) while \( a_j \) is right of \( b_v \) as shown in Fig. 9 (a), we call \( a \)-pair and \( b \)-pair \( H \)-crossing each other. In \( X \)-Alignment (A), the right move of \( a_i \) may push unit \( b_v \) to the right. Then \( b_v \) has to be moved to the right along with \( b_v \), which may push unit \( a_j \) to the right. Again \( a_i \) is moved to the right along with \( a_j \) and so on ... , the \( x \) alignment might continue infinitely. When the similar situation happens in the vertical dimension as shown in Fig. 9 (b), we call \( a \)-pair and \( b \)-pair \( V \)-crossing each other, and \( Y \)-Alignment might continue infinitely. We can conclude the following fact:

**Lemma 2** If two pairs of unit blocks are \( H \)-crossed or \( V \)-crossed to each other, the shapes of the corresponding macro blocks may not be recovered after \( X, Y \)-Alignment.

Given the sequence pair does not satisfy condition-3, there must exist \( a \)-pair and \( b \)-pair which do not separate each other in either sequence. Without loss of generality, we assume \( a_i \) is before \( a_j \), and \( b_v \) is before \( b_v \) in the first sequence \( \Gamma_1 \). Then \( \Gamma_1 \) will be either \( a_i b_v a_j b_v \) or \( a_i b_v b_v a_j \). When the first sequence is \( a_i b_v a_j b_v \), the second sequence \( \Gamma_2 \) is enumerated in Fig. 10. Since \( a \)-pair does not separate \( b \)-pair in \( \Gamma_2 \), the second sequence can not be the case (1.1.1), (1.1.2), (1.3.3), (1.3.4), (2.1.3), (2.1.4), (2.3.1) or (2.3.2). On the other hand, if the second sequence \( \Gamma_2 \) is ordered as the case (1.1.4), (1.3.1), (1.3.2), (2.1.1), (2.3.3) or (2.3.4), condition-2 will be violated. Therefore \( \Gamma_2 \) can only be case (1.1.3) or (2.1.2). The two corresponding sequence pairs are:

1. \((a_i b_v a_j b_v, b_v a_i b_v a_j)\)
2. \((a_i b_v a_j b_v, a_j b_v a_i b_v)\)
When the first sequence \( \Gamma_1 \) is \( a_i \ b_{i'} \ b_j \ a_{j'} \), the second sequence is enumerated in Fig. 11. Since \( -\)-pair does not separate \( -\)-pair in \( \Gamma_2 \), the second sequence cannot be the case (1.1.1), (1.1.2), (1.3.3), (1.3.4), (2.1.3), (2.1.4), (2.3.1) or (2.3.2). On the other hand, if the second sequence is the case (1.1.3), (1.3.2), (2.1.2) or (2.3.3), condition-2 will be violated in the sequence pair. Therefore \( \Gamma_2 \) can only be case (1.1.4), (1.3.1), (2.1.1) or (2.3.4). The four corresponding sequence pairs are:

3. \((a_i \ b_{i'} \ b_j \ a_{j'}, \ b_j \ a_i \ a_j \ b_{j'})\)
4. \((a_i \ b_{i'} \ b_j \ a_{j'}, \ b_{j'} \ a_i \ a_j \ b_{j'})\)
5. \((a_i \ b_{i'} \ b_j \ a_{j'}, \ b_{j'} \ a_j \ a_i \ b_{j'})\)
6. \((a_i \ b_{i'} \ b_j \ a_{j'}, \ b_{j'} \ a_j \ a_i \ b_{j'})\)

![Sequence pairs diagram](image)

Figure 10: Given \( -\)-pair does not separate \( -\)-pair in either sequence, and the first sequence is \( a_i \ b_{i'} \ a_{j'} \). The second sequence \( \Gamma_2 \) is enumerated in this Figure. Since \( -\)-pair does not separate \( -\)-pair in \( \Gamma_2 \), the second sequence cannot be the case (1.1.1), (1.1.2), (1.3.3), (1.3.4), (2.1.3), (2.1.4), (2.3.1) or (2.3.2). On the other hand, if the second sequence is the case (1.1.4), (1.3.1), (1.3.2), (2.1.1), (2.3.3) or (2.3.4), condition-2 will be violated. Therefore \( \Gamma_2 \) can only be case (1.1.3) or (2.1.2).

In each of the above six sequence pairs, \( -\)-pair and \( -\)-pair are either H-crossed or V-crossed to each other, the corresponding sequence pair is infeasible. Therefore condition-3 is a necessary condition.

**Lemma 3** The 3-conditions are necessary for a sequence pair to be feasible.

Based on the similar analysis, we can derive the following property:

**Lemma 4** Given condition-3 is satisfied in the sequence pair, any two pairs of unit blocks \( a_i, a_j \in A \) and \( b_{i'}, b_{j'} \in B, A \neq B, \) are neither H-crossed nor V-crossed to each other.

### 4.2 3-Conditions Are Sufficient

Given 3-conditions are satisfied in the sequence pair, the following Lemma is true:
Lemma 5 If a unit block $a_i \in A$ is left of a unit $b_j \in B$, $A \neq B$, then no unit of $A$ is right of a unit of $B$. Similarly, if $a_i$ is below $b_j$, no unit of $A$ is above a unit of $B$.

Lemma 5 can be proven by the contradiction. Let’s assume that $a_i$ is left of $b_j$, and a unit block $a_j \in A$ is right of a unit $b_j \in B$. If $a_i = a_j$ or $b_i = b_j$, we can easily derive that $a_i$ interrupts $(b_i, b_j)$ or $b_j$ interrupts $(a_i, a_j)$ in the sequence pair, which contradicts the assumption. Thus $a_i \neq a_j$, $b_i \neq b_j$. As such $a$-pair and $b$-pair are H-crossed to each other, then $a$-pair does not separate $b$-pair in either sequence. Again the assumption that sequence pair satisfies the 3-conditions is contradicted. Therefore Lemma 5 is true. Based on this property, we can sort the macro blocks in $x$ and $y$ direction, respectively:

**x-order**: macro block $A$ is before macro block $B$ if a unit of $A$ is left of a unit of $B$;

**y-order**: macro block $A$ is before macro block $B$ if a unit of $A$ is below a unit of $B$.

In the following, we show that the shapes of macro blocks can be exactly recovered by applying $X$, $Y$-Alignment on each macro block in $x$ and $y$-order, respectively. Based on the horizontal graph $G_h$, $X$-Alignment $(A)$ is carried out, a unit $a_i \in A$ is moved right, the unit $b_j \in B$ is pushed to the right by $a_i$ only when $b_j$ is right of $a_i$ and $B \neq A$, as shown in Fig. 12 (a). The right move of $b_j$ may further affect the unit blocks of $A$ if and only if:

1. a unit block $a_j \in A$ is right of $b_j$, or

2. a unit block $a_j \in A$ is right of another unit block $b_j \in B$.

In the first case, $b_j$ is right of $a_i$ while left of $a_j$ as shown in Fig. 12 (b). Then $b_i$ interrupts $a_i$, $a_j$ in the sequence pair, condition-2 is violated. In the second case, $a_i$, $a_j$ and $b_i$, $b_j$ are H-crossed to each other as shown in Fig. 12 (c). Then $a$-pair does not separate $b$-pair in either sequence.
and condition-3 is violated. Due to the assumption that sequence pair satisfies the 3-conditions, neither case 1 nor case 2 will happen. In other words, the right move of $a_i$ in $x$-align($A$) will not affect any other unit of $A$. $X$-Alignment($A$) exactly aligns the $x$ coordinates of $A$.

On the other hand, when $X$-Alignment is carried out on macro $B$ after $X$-Alignment($A$), no unit of $B$ can be left of a unit of $A$ due to the $x$-order. As such, the right move of unit blocks in $X$-Alignment($B$) will not affect macro block $A$. It implies that once $X$-Alignment($A$) is carried out, the $x$ coordinates of $A$ will not be affected later. We can conclude the following Lemma:

**Lemma 6** The 3-conditions are sufficient for a sequence pair to be feasible.

Followed by Lemma 3 and Lemma 6, we can conclude:

**Theorem 2** The 3-conditions are necessary and sufficient for a sequence pair to be feasible.

### 5 Convex Rectilinear Block Packing

A rectilinear block is called *convex* if any two points in the block have a shortest Manhattan path inside the block, as shown in Fig. 13 (a). Otherwise the block is called *concave*, as shown in Fig. 13 (b). It can be observed that some packing of concave blocks cannot be represented by the feasible sequence pair. For example, in the packing of Fig. 13 (c), block $c$ is right of block $a_1$ while left of $a_3$, then $c$ must interrupt $a_1$, $a_3$ in the corresponding sequence pair.

Let $\Pi$ denote an arbitrary convex block packing. By $H$- or $V$-partitioning the rectilinear macro blocks, $\Pi$ becomes a rectangle packing. The corresponding sequence pair $SP(\Pi)$ can be easily derived, in which the sub-blocks are treated as individual unit blocks. Obviously condition-1 is satisfied in $SP(\Pi)$.

If condition-2 is not satisfied in the sequence pair, there exist two unit blocks $a_i$, $a_j \in A$ interrupted by a unit $c \not\in A$. It can be easily derived that $A$ has concave shape, which contradicts the convex assumption. Therefore condition-2 is also satisfied in $SP(\Pi)$.

If the condition-3 is not satisfied in the sequence pair $SP(\Pi)$, there exist $a_i$, $a_j \in A$ and $b_\ell$, $b_j \in B$ such that $a$-pair does not separate $b$-pair in either sequence. According to the
Figure 13: (a) Any two points of the convex rectilinear block have a shortest Manhattan path inside the block. (b) At least two points in the concave block have no shortest Manhattan path located inside the block. (c) The packing of concave blocks can not be representable by feasible sequence pairs.

Figure 14: Let $l_i$ and $u_i$ denote the lower and upper boundary of unit $a_i$, respectively. Given sequence pair $SP(\Pi) = (a_i b_{ij} a_j b_{ij}, b_{ij} a_i b_{ij} a_j)$, $a_i$ is above $b_{ij}$ and $a_j$ is above $b_{ij}$. Then in the packing $\Pi$, $l_i \geq u_{ij}$ and $l_j \geq u_{ij}$. (a) $l_i \geq u_{ij}$: block $a_i$ is both left of and above $b_{ij}$. Then $SP(\Pi)$ can be transformed to $(a_i b_{ij} a_j b_{ij}, b_{ij} a_i b_{ij} a_j)$. (b) $l_j < u_{ij}$: $l_j \geq u_{ij} > l_i \geq u_{ij}$, then $l_j > u_{ij}$. Block $a_j$ is both right of and above $b_{ij}$ in the packing $\Pi$. $SP(\Pi)$ can be transformed to $(a_i a_j b_{ij} b_{ij}, b_{ij} b_{ij} a_i a_j)$. 
Then $(//)$ can be transformed as follows:

\[
(a_i \ b_i \ a_j \ b_j \cdot \ b_i \ a_i \ b_j \ a_j) \Rightarrow (a_i \ b_i \ a_j \ b_j \cdot \ b_i \ b_j \ a_i \ a_j)
\]  

(1)

Otherwise $l_i < u_j'$ as shown in Fig. 14 (b), $l_j \geq u_j' > l_i \geq u_i'$, then $l_j > u_i$. Block $a_j$ is both right of and above $b_i'$ in the packing II. Thus $SP(II)$ can be transformed as follows:

\[
(a_i \ b_i' \ a_j \ b_j' \cdot \ b_i' a_i \ b_j' a_j) \Rightarrow (a_i \ a_j \ b_i' \ b_j' \cdot b_i' b_j' a_i a_j).
\]

(2)

Obviously the transformation will not cause any violation of 3-conditions, $a$-pair will separate $b$-pair in at least one sequence of $SP(II)$, meanwhile the topology defined by the sequence pair is consistent with the packing II. For case 2 through case 6, $SP(II)$ can be similarly transformed. In such a way, a feasible sequence pair can be eventually achieved, that is

**Lemma 7** There always exists a feasible sequence pair corresponding to a packing of convex rectilinear blocks.

For $M$ convex macro blocks, each of them includes at most $n$ rectangular sub-blocks, the optimal solution for convex rectilinear block packing can be found by exhausting the finite number $O((nM)!^2)$ of feasible sequence pairs.

When a convex macro block $A$ is H-partitioned as shown in Fig. 15 (a), $a_i$ denotes the $i^{th}$ leftmost sub-block, the H-pair of $A$ is $(a_1 \cdots a_i \ a_i+1 \cdots a_m, \ a_1 \cdots a_i \ a_i+1 \cdots a_m)$. When $A$ is V-partitioned as shown in Fig. 15 (b), $a_i$ denotes the $i^{th}$ lowest sub-block, the V-pair of $A$ is $(a_m \cdots a_i+1 \ a_i \cdots a_1, \ a_1 \cdots a_i \ a_i+1 \cdots a_m)$.

Figure 15: (a) Convex block $A$ is H-partitioned where $a_i$ is the $i^{th}$ leftmost sub-block of $A$. The H-pair of $A$ equals to $(a_1 \cdots a_i \ a_i+1 \cdots a_m, \ a_1 \cdots a_i \ a_i+1 \cdots a_m)$. (b) When $A$ is V-partitioned, $a_i$ is the $i^{th}$ lowest sub-block of $A$. The V-pair $A$ equals to $(a_m \cdots a_i+1 \ a_i \cdots a_1, \ a_1 \cdots a_i \ a_i+1 \cdots a_m)$.

### 6 Stochastic Search on Convex Block Packing

In the following, a stochastic search is applied to the optimization of convex block packing. Three sequence pair operations are defined to incrementally change the feasible sequence pair: rotation, $\Gamma_1$-mutation, and $\Gamma_2$-mutation.
6.1 Rotation

Rotation rotates a macro block by $90^\circ$ in the clockwise direction as shown in Fig. 16. Given macro block $A = \{a_1, a_2, ..., a_n\}$, $\text{rotate}(A)$ switches the height with the width for each unit block of $A$. The sequence pair is accordingly changed by switching unit $a_i$ with $a_{m+1-i}$, $i \in [1, n]$, in the first sequence ($90^\circ$ to $180^\circ$, $270^\circ$ to $0^\circ$) or the second sequence ($0^\circ$ to $90^\circ$, $180^\circ$ to $270^\circ$). Rotation takes $O(n)$ time, which is close to constant time.

![Figure 16: A macro block $A = \{a_1, a_2, a_3\}$ is rotated from $0^\circ$ through $90^\circ$ and $180^\circ$ to $270^\circ$ in the clockwise direction. $\text{rotate}(A)$ switches the height and width for each unit block of $A$, and changes the sequence pair by switching $a_i$ with $a_{m+1-i}$ in the first or second sequence.](image)

Given a rotation is carried out on macro block $A$ represented in a feasible sequence pair $(\Gamma_1, \Gamma_2)$, only the permutation pair of $A$ is changed from H-pair to V-pair, or vice versa. Accordingly the orientation of $A$ is rotated such that the H-partition of $A$ becomes the V-partition, or vice versa. As such, the resultant sequence pair $(\Gamma'_1, \Gamma'_2)$ still satisfies condition-1.

If the resultant sequence pair does not satisfy condition-2, there exists a unit $c \in C$ between two units $b_i$, $b_j \in B$ in both $\Gamma'_1$ and $\Gamma'_2$, $C \neq B$. Three cases are possible:

1. $B \neq A$, $C \neq A$.
   The positions of $b_i$, $b_j$ and $c$ in the resultant sequence pair are exactly the same with those in the original sequence pair. Thus $c$ also interrupts $b_i$, $b_j$ in $(\Gamma_1, \Gamma_2)$, which contradicts the feasible assumption.

2. $B \neq A$, $C = A$.
   Without loss of generality, we assume $c = a_k$, the resultant sequence pair is $(b_i a_k b_j, b_i a_k b_j)$. According to $\text{rotate}(A)$, the original sequence pair should be either $(b_i a_k b_j, b_i a_{m+1-k} b_j)$ or $(b_i a_{m+1-k} b_j, b_i a_k b_j)$. If $k = m + 1 - k$, then $a_k$ interrupts $b_i$, $b_j$. Otherwise $a_k$, $a_{m+1-k}$ and $b_i$, $b_j$ will not separate each other in either sequence of $(\Gamma_1, \Gamma_2)$, then condition-3 is violated. Either case contradicts the feasible assumption of the original sequence pair.

3. $B = A$, $C \neq A$.
   Without loss of generality, we assume $b_i = a_i$, $b_j = a_j$ and the resultant sequence pair is $(a_i c a_j, a_i c a_j)$. According to $\text{rotate}(A)$, the original sequence pair should be either $(a_i c a_j, a_{m+1-i} c a_{m+1-j})$ or $(a_{m+1-i} c a_{m+1-j}, a_i c a_j)$. Since the two cases are symmetrical, we only discuss the first one. Due to the feasible assumption, unit $c$ can not appear between $a_i$ and $a_j$ in the second sequence $\Gamma_2$. Similarly, $c$ can not appear between $a_{m+1-i}$ and $a_{m+1-j}$ in the first sequence $\Gamma_1$. Therefore the original sequence pair $(\Gamma_1, \Gamma_2)$
will be one of the four possible cases shown in Fig. 17. In each of them there are two units of \( A \) interrupted by unit \( c \). A contradiction exists.

As such, we can derive that the resultant sequence pair also satisfies condition-2.

![Diagram](image)

**Figure 17:** In a feasible sequence pair \((a_i \ c \ a_j, \ a_{m+1-i} \ c \ a_{m+1-j})\), \(a_i, a_j\) will be both before or both after \( c \) in the second sequence. Similarly \( a_{m+1-i}, a_{m+1-j} \) will be both before or both after \( c \) in the first sequence. As such, the sequence pair will be one of the four possible cases: (1) \( a_j, a_{m+1-j} \) are interrupted by \( c \), (2) \( a_i, a_{m+1-i} \) are interrupted by \( c \), (3) \( a_j, a_{m+1-i} \) are interrupted by \( c \). The sequence pair cannot be feasible.

If the resultant sequence pair does not satisfy condition-3, there exist \( b_i, b_j \in B \) and \( c_k, c_l \in C, B \neq C \), \( b \)-pair does not separate \( c \)-pair in either sequence of \((\Gamma_1, \Gamma_2)\). When \( B \neq A, C \neq A \), the original sequence pair will also violate condition-3. Thus let \( B = A, b_i = a_i, b_j = a_j \).

In the original sequence pair, \( a_i, a_j \) do not separate \( c_k, c_l \) in one sequence, say \( \Gamma_1 \), while \( a_{m+1-i}, a_{m+1-j} \) do not separate \( c_k, c_l \) in the other sequence, say \( \Gamma_2 \).

Due to the feasible assumption of \((\Gamma_1, \Gamma_2)\), \( a_{m+1-i}, a_{m+1-j} \) must separate \( c_k, c_l \) in \( \Gamma_1 \). As shown in Fig. 18, we can draw a line to separate \( c_k, c_l \) from \( a_{m+1-i}, a_{m+1-j} \) in the first sequence \( \Gamma_1 \). Either \( a_i \) or \( a_j \) will be on the different side of \( a_{m+1-i}, a_{m+1-j} \); otherwise \( a_i, a_j \) will separate \( c_k, c_l \) in the first sequence. In the second sequence \( \Gamma_2 \), we can derive the symmetrical situation. Then totally four cases are possible as shown in Fig. 18, in each of them condition-3 is not satisfied, and the original sequence pair could not be feasible. As such, we can derive that the resultant sequence pair satisfies condition-3. Overall we can conclude:

**Lemma 8** When a rotation is carried out on a macro block represented in a feasible sequence pair, the resultant sequence pair remains feasible.

### 6.2 \( \Gamma_1 \)-Mutation And \( \Gamma_2 \)-Mutation

\( \Gamma_1 \)-Mutation switches two adjacent unit blocks in the first sequence: \( \cdots \ a \ b \ cdots \Rightarrow \cdots \ b \ a \ cdots \), in which \( a \in A, b \in B, A \neq B \). \( \Gamma_2 \)-Mutation is similarly carried out on the second sequence. Both operations take constant time. Since two mutations are symmetrical, we will only discuss \( \Gamma_1 \)-mutation.

17
In the following, we assume that the resultant sequence pair is the first case. If \( ? \) pair violates condition-2. Thus both sequences of \( a \) and \( b \) be either Figure 18: Given a feasible sequence pair \((\Gamma_1, \Gamma_2)\) where \( a_i, a_j \) do not separate \( c_k, c_l \) in \( \Gamma_1 \), and \( a_{m+1-i}, a_{m+1-j} \) do not separate \( c_k, c_l \) in \( \Gamma_2 \). Then \( a_{m+1-i}, a_{m+1-j} \) must separate \( c_k, c_l \) in \( \Gamma_1 \). We can draw a line to separate \( c_k, c_l \) from \( a_{m+1-i}, a_{m+1-j} \). Either \( a_i \) or \( a_j \) will be on a different side of \( a_{m+1-i}, a_{m+1-j} \), otherwise \( a_i, a_j \) will separate \( c_k, c_l \) in the first sequence. The similar case can be derived in the second sequence. Totally four cases are possible for the sequence pair : (1) \( a_i, a_{m+1-i} \) and \( c_k, c_l \), (2) \( a_i, a_{m+1-j} \) and \( c_k, c_l \), (3) \( a_j, a_{m+1-i} \) and \( c_k, c_l \), or (4) \( a_i, a_{m+1-j} \) and \( c_k, c_l \) violate condition-3. As such, the sequence pair can not be feasible.

Given \( \Gamma_1 \)-mutation is carried out on a feasible sequence pair, the permutation pair of every macro block will be preserved in the resultant sequence pair, as such condition-1 is satisfied after the operation. In the following, we assume \( A = \{ a_1 a_2 \cdots a_m \} \) and \( B = \{ b_1 b_2 \cdots b_l \} \). By relabeling the sub-blocks of \( A \) and \( B \), \( a_i \) and \( b_i \) are the \( i^{th} \) sub-block of \( A \) and \( B \) in the first sequence \( \Gamma_1 \), respectively.

**Lemma 9** When \( \Gamma_1 \)-mutation is carried out on a feasible sequence pair : \( \cdots a_i b_j \cdots \Rightarrow \cdots b_j a_i \cdots \), the resultant sequence pair may violate condition-2 only when \( a_i = a_m, b_j = b_1 \). And the violation only happens on \( a_m \) and \( b_1 \).

Given the resultant sequence pair violates condition-2, then \( d \in D \) is between \( c_u, c_v \in C \) in both sequences of \((\Gamma_1', \Gamma_2')\). According to \( \Gamma_1 \)-mutation, \( \Gamma_2 = \Gamma_2' \), then \( d \) is between \( c_u \) and \( c_v \) in \( \Gamma_2 \). On the other hand, if \( d \neq a_i, d \neq b_j, d \) is also between \( c_u \) and \( c_v \) in \( \Gamma_1 \), the original sequence pair violates condition-2. Thus \( d \) must be either \( a_i \) or \( b_j \). Similarly we can derive that one unit of \( (c_u, c_v) \) must be \( a_i \) or \( b_j \). Therefore in the resultant sequence pair \((\Gamma_1', \Gamma_2')\), two cases are possible :

1. \( a_x, a_i \in A \) is interrupted by \( b_j \). Due to \( \Gamma_1 \)-mutation, \( \Gamma_1' \) must be \( \cdots a_x \cdots b_j a_i \cdots \), then \( a_x = a_{i-k}, \) where \( k > 0 \).

2. \( b_y, b_j \in B \) is interrupted by \( a_i \). Due to \( \Gamma_1 \)-mutation, \( \Gamma_1' \) must be \( \cdots b_j a_i \cdots b_y \cdots \), then \( b_y = b_{j+l}, \) where \( l > 0 \).

In the following, we assume that the resultant sequence pair is the first case. If \( i < m \), the original first sequence \( \Gamma_1 \) will be \( \cdots a_{i-k} \cdots a_i b_j \cdots a_m \cdots \). The original second sequence \( \Gamma_2 \) will be either \( \cdots a_{i-k} \cdots b_j \cdots a_i \cdots a_m \cdots \) or \( \cdots a_m \cdots a_i \cdots b_j \cdots a_{i-k} \cdots \). Then \( b_j \) will interrupt \( a_{i-k}, a_m \) in the original sequence pair. Therefore, \( i \) must equal \( m \), i.e. \( a_i = a_m \).

If \( j > 1 \), the original first sequence \( \Gamma_1 \) will be either \( \cdots a_{i-k} \cdots b_1 \cdots a_i b_j \cdots a_{i-k} \cdots \). Due to condition-3, \( a_{i-k}, a_i \) and \( b_j \) must separate each other in the original second sequence \( \Gamma_2 \). Since \( \Gamma_2 = \Gamma_2' \), \( b_j \) can not be between \( a_{i-k}, a_i \) in \( \Gamma_2' \), which contradicts the assumption that \( b_j \) interrupts \( a_{i-k} \) and \( a_i \) in the resultant sequence pair. Therefore \( j \) must equal 1, i.e. \( b_j = b_1 \). Similarly when the resultant sequence pair is the second case, we can derive that \( a_i = a_m, b_j = b_1 \). Therefore, Lemma 9 is true.
Lemma 10 When \( \Gamma_1 \)-mutation is carried out on a feasible sequence pair \( \cdots \cdot a_i b_j \cdots \Rightarrow \cdots \cdot b_j a_i \cdots \), the resultant sequence pair may violate condition-3 only when \( a_i = a_m \), \( b_j = b_1 \). And the violation only happens on \( a_m \) and \( b_1 \).

If the resultant sequence pair does not satisfy condition-3, there exist two pairs of unit blocks which separate each other in neither \( \Gamma'_1 \) nor \( \Gamma'_2 \). Since \( \Gamma_2 = \Gamma'_2 \), the two pairs must separate each other in the first sequence \( \Gamma_1 \) due to the feasible assumption of the original sequence pair. Therefore the two pairs must include both \( a_i \) and \( b_j : a_{i-k}, a_i \in A \) and \( b_j, b_{j+l} \in B \), where \( k, l > 0 \).

If \( i < m \), the original first sequence \( \Gamma_1 \) will be either \( \cdots \cdot a_{i-k} \cdots a_i b_j \cdots a_m \cdots b_{j+l} \cdots \) or \( \cdots \cdot a_{i-k} \cdots a_i b_j \cdots b_{j+l} \cdots a_{m} \cdots \). Due to the feasible assumption of the original sequence pair, in the second sequence \( \Gamma_2 \), both \( a_{i-k} \), \( a_m \) and \( a_i, a_m \) should separate \( b_j, b_{j+l} \). Therefore \( a_{i-k} \), \( a_i \) will separate \( b_j, b_{j+l} \) in \( \Gamma_2 \). Since \( \Gamma_2 = \Gamma'_2 \), \( a_i \) and \( b_j, b_{j+l} \) satisfy condition-3 in the resultant sequence pair, which contradicts the assumption. So \( i \) must equal \( m \), i.e. \( a_i = a_m \).

In the same way, we can derive that \( b_j = b_1 \). Therefore, Lemma 10 is true.

As we can see, an infeasible solution can be generated by a mutation. During the stochastic search, the infeasible solutions can be avoided by simply cancelling the operation. However the continuity of the local search may be destroyed, the optimal solution may not be reachable. Based on above analysis, the infeasible solution can only be generated when a mutation is carried out on \( \cdots a_1 a_2 \cdots a_m b_1 b_2 \cdots b_n \cdots \), where switching \( a_m \) with \( b_1 \) may cause the violation of condition-2 or condition-3. Due to this fact, we can develop a very simple procedure called adaptation to adapt the infeasible solution into a new feasible solution. In such a way, the continuous search of the feasible solution space can be guaranteed.

### 6.3 Adaptation

Given that \( \Gamma_1 \)-mutation generates an infeasible solution, the original first sequence \( \Gamma_1 \) must be \( \cdots a_1 a_2 \cdots a_m b_1 \cdots b_2 \cdots b_n \cdots \). For example,

\[
\Gamma_1 = a_1 c a_2 d e a_3 f a_4 b_1 g b_2 h i j b_3
\]

We define squeeze-to-right (\( \Gamma_1, A \)) as follows : applying \( \Gamma_1 \)-mutation on \( a_3 \) and its right neighbor until \( a_3 \) is left adjacent to \( a_4 \), then applying \( \Gamma_1 \)-mutation on \( a_2 \) and its right neighbor until \( a_2 \) is left adjacent to \( a_3 \), and so on ...

\[
\Gamma_1 = c d e f a_3 a_2 a_3 a_4 b_1 g b_2 h i j b_3
\]

Based on the analysis of the mutations, it can be guaranteed that the intermediate sequence pairs generated during squeeze-to-right (\( \Gamma_1, A \)) are always feasible. Similarly we define squeeze-to-left (\( \Gamma_1, B \)) as follows : applying \( \Gamma_1 \)-mutation on \( b_2 \) and its left neighbor until \( b_2 \) is right adjacent to \( b_1 \), then applying \( \Gamma_1 \)-mutation on \( b_3 \) and its left neighbor until \( b_3 \) is right adjacent to \( b_2 \):

\[
\Gamma_1 = c d e f a_1 a_2 a_3 a_4 b_1 b_2 b_3 g h i j
\]

The intermediate sequence pairs generated during squeeze-to-left (\( \Gamma_1, B \)) are always feasible.

After the squeeze operations, the unit blocks of both \( A \) and \( B \) are consecutive in the first sequence. Then adaptation swaps \( A \) and \( B \) : \( \cdots a_1 a_2 \cdots a_m b_1 b_2 \cdots b_n \cdots \Rightarrow \cdots b_1 b_2 \cdots b_n a_1 a_2 \cdots a_m \cdots \). Obviously no violation of 3-conditions can occur during the swapping, and the resultant sequence pair is feasible. In the above example, the first sequence is adapted to :

\[
\Gamma_1 = c d e f b_1 b_2 b_3 a_1 a_2 a_3 a_4 g h i j
\]

In \( O(m + n) \) time, adaptation generates a new feasible sequence pair which is different from the original one.
Lemma 11 The resultant infeasible sequence pair after a mutation can always be transformed to a feasible sequence pair by the adaptation.

As such, the local moves of the stochastic search are completed. Each local move (a rotation or a mutation followed by an adaptation) takes the time proportional to the number of unit blocks in a macro block, which is close to constant time. In addition, each local move generates a feasible sequence pair. In such a way, the feasible solution space can be continuously searched by the local moves.

6.4 Rotation + Mutations + Adaptation = Exhaustive Search

Theorem 3 The optimal solution can always be reachable through a finite steps of the rotation and mutations followed by the adaptation.

Given any two feasible sequence pairs ISP = (Π₁, Π₂) and OSP = (Ο₁, Ο₂), we can always find a search path from ISP to OSP, which consists of only a finite times of the local moves. In such way, the above Theorem can be proven. The path can be constructed by following four phases:

Phase 1 for every macro block A
rotate A until the permutation pair of A in ISP equals that in OSP endfor
Phase 2 for every macro block A
squeeze-to-right(Π₁, A) squeeze-to-right(Π₂, A) endfor
Phase 3 adjust Π₁ such that Π₁ equals Ο₁
Phase 4 adjust Π₂ such that Π₂ equals Ο₂

Phase 1 first applies the rotation in ISP such that the permutation pair of every macro block equals that in OSP. Given total $M$ macro blocks, at most $3M$ times rotation are required, in which 3-conditions are always satisfied. In Phase 2, squeeze operation is applied in both sequences of ISP such that the unit blocks of each macro are consecutive in both Π₁ and Π₂. Based on the description of squeeze operations, Phase 2 includes mutations only, and the intermediate sequence pairs are always feasible. After Phase 2, for any unit blocks $a_i$, $a_j \in A$ and $b_k \in B$, $A \neq B$, $b_k$ will not appear between $a_i$, $a_j$ in either Π₁ or Π₂. Given total $N$ unit blocks in the sequence pair, Phase 2 will take $O(N^2)$ times mutations.

In Phase 3, Π₁ is adjusted to Ο₁ by applying Π₁-mutations followed by adaptations. As shown in Fig. 19 (a), starting from $i = 0$, Phase 3 compares $i^{th}$ unit of Π₁, $u_i$, with the $i^{th}$ unit of Π₁, $v_i$. If $u_i = v_i$, $i$ increases by one, i.e. Phase 3 compares the following units. Otherwise, the unit $u_i$ is located in Π₁ : $u_i = v_{i+k}$, where $k$ is a positive integer.

Let $v_{i+k-1}$ denote the left neighbor of $v_{i+k}$ in Π₁. Then in Ο₁, $v_{i+k-1}$ will be after $u_i$ which equals $v_{i+k}$, as shown in Fig. 19 (b). If $v_{i+k-1}$ and $v_{i+k}$ belong to the same macro block V, the permutation of V in Π₁ is different from that in Ο₁. Due to Phase 1, we can derive that $v_{i+k-1}$ and $v_{i+k}$ do not belong to the same macro. Therefore in Π₁, mutation can be applied on $v_{i+k-1}$ and $v_{i+k}$. If the mutation generates a feasible sequence pair, the unit $u_i$ will be moved to the left by one step in Π₁ : $u_i = v_{i+k-1}$ as shown in Fig. 19 (c). Otherwise, $v_{i+k-1} = a_m$, $v_{i+k} = b_1$.

If $a_1$ is before $v_i$ in Π₁ as shown in Fig. 19 (d), then in Ο₁, $a_1$ must be before $u_i = v_{i+k} = b_1$. Due to condition-2, $b_1$ can not be between $a_1$, $a_m$ in the second sequence of OSP. If Π₂ has not been adjusted to Ο₂ so far, $b_1$ will not be between $a_1$, $a_m$ due to Phase 2. On the other hand,
Figure 19: (a) Starting from $i = 0$, Phase 3 compares the $i^{th}$ unit of $O\Gamma_1$, $u_i$, with the $i^{th}$ unit of $\Pi_1$, $v_i$. If $u_i \neq v_i$, the unit $u_i$ must be located in $\Pi_1$ as $v_{i+k}$, where $k > 0$. (b) Let $v_{i+k-1}$ denote the left neighbor of $v_{i+k}$ in $\Pi_1$. Then in $O\Gamma_1$, $v_{i+k-1}$ will be after $u_i = v_{i+k}$. The permutation of $v_{i+k-1}$ and $v_{i+k}$ in $O\Gamma_1$ is different from that in $\Pi_1$. (c) If the mutation of $v_{i+k-1}$, $v_{i+k}$ generates a feasible sequence pair, the unit $u_i$ will be moved to $v_{i+k-1}$ in $\Pi_1$. (d) If the mutation causes an infeasible solution, $v_{i+k-1} = a_m$ and $v_{i+k} = b_1$. If $a_1$ is before $v_i$ in $\Pi_1$, then in $O\Gamma_1$, $a_1$ must be before $u_i = v_{i+k} = b_1$. Due to condition-2, $b_1$ can not be between $a_1$, $a_m$ in $O\Gamma_2$. Then $b_1$ can not be between $a_1$, $a_m$ in $\Pi_1$. The mutation of $a_m$, $b_1$ can not lead to an infeasible solution. (e) As such, in $\Pi_1$, $a_1$ must be after $v_i$. After the adaptation, unit $u_i$ will be moved to the left by $m$ steps in $\Pi_1$: $u_i = v_{i+k-m}$, where $i + k - m \geq i$. 
if $\Pi_2$ has been adjusted to $\Omega_2$, $b_1$ still can not be between $a_1$, $a_m$. As such, the mutation of $a_m$, $b_1$ can not lead to an infeasible solution. Therefore in $\Pi_1$, $a_1$ must be after $v_i$ as shown in Fig. 19 (e). After the adaptation, unit $u_i$ will be moved to the left by $m$ steps in $\Pi_1$: $u_i = v_{i+k-m}$, where $i + k - m \geq i$.

Therefore in $\Pi_1$, the unit $u_i$ is either moved to $v_{i+k-1}$ as shown in Fig. 19 (c) or $v_{i+k-m}$ as shown in Fig. 19 (e). In such way, $u_i$ can be continuously moved to the left in $\Pi_1$ until $u_i = v_i$. Then $i$ increases by one and Phase 3 visits the following unit $u_{i+1}$ in $\Omega_1$. When $i$ reaches $N$, the first sequence of ISP is adjusted to be $\Omega_1$. Phase 3 terminates. Phase 4 repeats the similar process on the second sequence of ISP. It can be easily derived that Phase 3 and Phase 4 take $O(N^2)$ times mutations and adaptations, together with the first two phases, the search path consists of at most $O(M + N^2)$ times of local moves.

7 Experimental Results and Conclusion

7.1 Packing Results

We have implemented the algorithm proposed in this paper using C language and tested it on SUN SPARC 20 workstation. The data are generated randomly. Figure 20 (a) gives the packing of ten L-shaped blocks, which takes about two minutes, the total packing area is 1.09 times of the total block area. On the other hand, Figure 20 (b) reports the packing result of 19 convex rectilinear blocks, which takes about 35 minutes, the total packing area is 1.13 times of the total block area. Due to the tight schedule, we have not enough time to refine the source code. We believe that the tradeoff of the packing quality and CPU time can be much improved, and we are working on this right now.

![Figure 20: (a) The packing of ten L-shaped blocks, which takes about two minutes, the total packing area is 1.09 times of total block area. (b) The packing result of 19 convex rectilinear blocks, which takes about 45 minutes, the total packing area is 1.13 times of total block area.](image)

7.2 Concluding Remarks

In this paper, for the first time, we solve the arbitrary shaped rectilinear block packing problem. Rectilinear macro blocks are partitioned into a set of rectangular sub-blocks, each of them is individually represented as a unit block in the sequence pair. The feasible solution space is
defined. Three conditions on the sequence pair are derived, which are necessary and sufficient for a sequence pair to be feasible. Furthermore it is proven that there always exists a feasible sequence pair corresponding to a packing of convex rectilinear blocks. Based on this fact, a stochastic search is applied on the optimization of convex block packing. Local moves are defined to search the feasible solution space both continuously and exhaustively.

References


