Optimizing Wirelength and Routability by Searching Alternative Packings in Floorplanning

CHIU-WING SHAM
The Hong Kong Polytechnic University
EVANGELINE F. Y. YOUNG
The Chinese University of Hong Kong
and
HAI ZHOU
Northwestern University

Recent advances in VLSI technology have made optimization of the interconnect delay and routability of a circuit more important. We should consider interconnect planning as early as possible. We propose a postfloorplanning step to reduce the interconnect cost of a floorplan by searching alternative packings. If a packing contains a rectangular bounding box of a group of modules, we can rearrange the blocks in the bounding box to obtain a new floorplan with the same area, but possibly with a smaller interconnect cost. Experimental results show that we can reduce the interconnect cost of a packing without any penalty in area.

Categories and Subject Descriptors: B.7.2 [**Integrated Circuits**]: Design Aids

General Terms: Performance

Additional Key Words and Phrases: Floorplanning, wirelength reduction

ACM Reference Format:

Sham, C.-W., Young, E. F. Y., and Zhou, H. 2008. Optimizing wirelength and routability by searching alternative packings in floorplanning. ACM Trans. Des. Autom. Electron. Syst., 13, 1, Article 21 (January 2008), 13 pages, DOI = 10.1145/1297666.1297687 http://doi.acm.org/10.1145/1297666.1297687

The work described in this article was partially supported by the Competitive Earmarked Research Grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. PolyU 5262/07E).

Authors' addresses: C.-W. Sham, Department of Electronic and Information Engineering, Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong; email: cwsham@alumni,cuhk.net; E. F. Y. Young, The Chinese University of Hong Kong, Shatin, NT, Hong Kong SAR, The People's Republic of China; H. Zhou, Department of Electrical Engineering and Computer Science, Northwestern University, 633 Clark Street, Evanston, IL 60208.

1. INTRODUCTION

1.1 Motivations

Interconnect optimization becomes an important issue in floorplanning. All nets should be routable in detailed routing and the design constraints on timing, power, and noise should also be satisfied. During floorplanning, the shapes and locations of the modules on a chip are planned, the results of which will greatly affect the overall performance of the final circuit. In some advanced systems today, the design densities in the deep submicron technology can be severe. This will result in a major escalation in routing demands and poor timing performance. Thus, interconnect optimization has become the major concern in floorplanning and many other designing steps.

Traditional floorplanners cannot guarantee a floorplan solution with good routability and timing performance. In some cases, the floorplanner does not even give us any interconnect information of the floorplan solution. A floorplan with poor routability and timing performance will cause a large expansion in area, or even an unroutable design failing to achieve timing closure after detailed routing, and it is too late to make any significant changes in placement by then. As a result, interconnect optimization is needed during the early stages of the design process.

1.2 Related Works

There are several previous works addressing the interconnect issues in floorplan design. In the papers Murata and Kuh [1998],Chen et al. [1999], Chang et al. [2000], Ma et al. [2003], Lou et al. [2001], Wang and Sarrafzadeh [2000], Sham and Young [2003], and Lai et al. [2003], the authors formulated different congestion-related cost functions (evaluated by some simple global routings), including a hybrid length plus congestion cost function, and these cost functions are then optimized by applying some optimization techniques like simulated annealing and genetic algorithms. In the papers Adya and Markov [2003, 2001], the fixed-outline floorplanner (parquet) has been proposed and it gives a significant improvement on the fixed-outline floorplan problem. In the paper Lai et al. [2003], the congestion model used is the average net density on the half-perimeter boundaries of different regions in a floorplan. In the papers Chen et al. [1999], Chang et al. [2000], Ma et al. [2003], a floorplan is divided into tiles and congestion is estimated at each tile, assuming that each net is routed in either L- or Z-shape. In the papers Lou et al. [2001] and Sham and Young [2003], probabilistic analyses are used to estimate congestion and routability. In the paper Wang and Sarrafzadeh [2000], the authors use a real global router to estimate congestion. This may be more accurate, but the runtime penalty is high. Even though interconnect-driven floorplanners can perform congestion estimation and buffer planning accurately, there is still a significant penalty in runtime. In the paper Jeske and Greenwood [2003], a cluster-constrainted floorplanner is proposed to solve the cluster-constrainted floorplan problem. However, the method of searching for these clusters in the final floorplan solutions to further reduce other interconnect costs has not been mentioned.

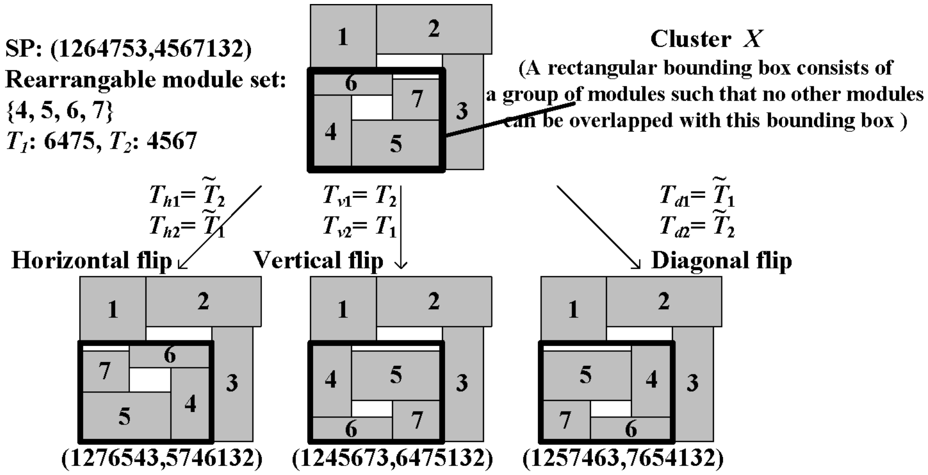


Fig. 1. Examples of alternative packings.

1.3 Our Contributions

In this article, we propose an approach to search dimension-independent area-equivalent packings (alternative packings) by looking only at the sequence pair. This approach can be used as a postfloorplanning step to reduce interconnect costs such as wirelength, congestion, and routability. We found that if a packing F contains a cluster (a rectangular bounding box consists of a group of modules such that no other modules can be overlapped with this bounding box), we can rearrange the blocks in the cluster to obtain a new packing with the same area as F, but possibly with a smaller interconnect cost. Experimental results show that we can always reduce the total wirelength and improve the routability of a floorplan without any penalty in area and the additional runtime is small.

This work is organized as follows. There will be an overview of our method in Section 2. We will discuss the method of searching alternative packings in detail in Section 3. Experimental results will be shown in Section 4. Finally, we will give a conclusion in Section 5.

2. OVERVIEW OF THE METHOD

In this section, we will introduce the method of searching alternative packings. We define an alternative packing of a floorplan as follows.

Definition 1. Given a floorplan topology F of a set S of modules without overlapping, an alternative packing of F is another floorplan topology of the modules in S that has the same area as F.

Examples of alternative packings are shown in Figure 1. In our method, we want to find all the alternative packings of a floorplan solution. Generally, the alternative packings can be obtained by flipping some clusters in a floorplan horizontally, vertically, or both horizontally and vertically. Note that a cluster in a floorplan is defined as a set of modules inside a bounding box such that no other modules can be overlapped with this bounding box.

Since all alternative packings have the same area, their areas are not required to be calculated again. However, the interconnect costs may be different because the locations of some modules have been changed. We can then find among all the alternative packings the one with the minimum interconnect cost. This means that the interconnect cost can be further optimized while keeping the area of the packing unchanged. After floorplanning, we can apply this method to further optimize the interconnect cost of the final floorplan solution.

3. SEARCHING ALTERNATIVE PACKINGS

In Figure 1, there is a cluster X containing four modules in the packing. We can change the packings while preserving the area in three ways: flipping X horizontally, flipping X vertically, or flipping X horizontally and vertically (we call this a diagonal flip). In the method of searching alternative packings, we will use a sequence pair [Murata et al. 1996, 1995] to represent a floorplan. We can obtain alternative packings with the same area by only working on the sequence-pair representations. In this section, we will discuss how we can find all alternative sequence-pairs from a given sequence-pair (S_1, S_2). The packing represented by all alternative sequence-pairs will have the same area as that represented by (S_1, S_2).

3.1 Identifying Clusters in a Sequence Pair

In order to construct alternative packings, we need to find the clusters in a given floorplan solution. We define a *rearrangable module set* in a sequence pair as follows.

Definition 2. Given a sequence pair (S_1, S_2), a set of two or more modules is a rearrangable module set if and only if they form contiguous subsequences in both S_1 and S_2.

An example is shown in Figure 1. In this example, the packing is represented by the sequence pair (1264753, 4567132). The set of modules $\{4, 5, 6, 7\}$ has formed contiguous subsequences in both S_1 and S_2. Thus, the set of modules $\{4, 5, 6, 7\}$ is a rearrangable module set according to the aforesaid definition. In addition, the set of modules $\{4, 5, 6, 7\}$ will form a cluster in the packing according to the following theorem.

THEOREM 1. *Given a sequence pair (S_1, S_2) and its corresponding packing F, a set of modules is a rearrangable module set if and only if these modules form a cluster in F independent of the dimensions of the modules.*

PROOF. Given a rearrangable module set M_r, the modules in M_r form a contiguous subsequence T_1 in S_1 and form a contiguous subsequence T_2 in S_2. Consider the relationships between the modules in M_r and the other modules not in M_r. If a module $m_i \notin M_r$ is on the left of a module in M_r, the sequence pair of the given floorplan should be of the form $(\dots m_i \dots T_1 \dots, \dots m_i \dots T_2 \dots)$. This means that m_i is on the left of all the modules in M_r. The same argument follows for m_i lying on the right of the modules in M_r. Similarly, if a module $m_i \notin M_r$ is above a module in M_r, the sequence pair should be of the form

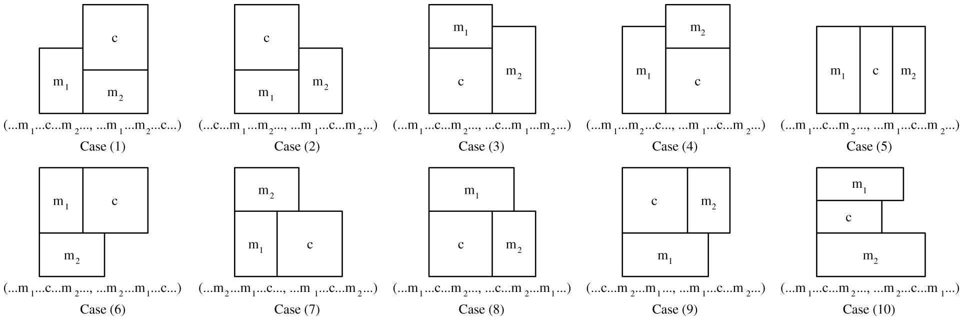

c
m1
m2
(...m1...c...m2..., ...m1...m2...c...)
Case (1)
c
m1
m2
(...c...m1...m2..., ...m1...c...m2...)
Case (2)
m1
c
m2
(...m1...c...m2..., ...c...m1...m2...)
Case (3)
m2
m1
c
(...m1...m2...c..., ...m1...c...m2...)
Case (4)
m1
c
m2
(...m1...c...m2..., ...m1...c...m2...)
Case (5)
m1
c
m2
(...m1...c...m2..., ...m2...m1...c...)
Case (6)
m2
m1
c
(...m2...m1...c..., ...m1...c...m2...)
Case (7)
m1
c
m2
(...m1...c...m2..., ...c...m2...m1...)
Case (8)
c
m2
m1
(...c...m2...m1..., ...m1...c...m2...)
Case (9)
m1
c
m2
(...m1...c...m2..., ...m2...c...m1...)
Case (10)

Fig. 2. Cases for the if-proof of Theorem 1.

$(\ldots m_i \ldots T_1 \ldots, \ldots T_2 \ldots m_i \ldots)$. It means that m_i is above all the modules in M_r. The same argument follows for m_i lying below the modules in M_r. Thus, the horizontal and vertical relationships between the modules in M_r and all the other modules not in M_r are identical, and the modules in M_r will form a cluster in the packing independent of the module dimensions.

$\Leftarrow$ Given a sequence pair (S_1, S_2) containing a set of modules $M_r = \{m_1, m_2, \ldots\}$. Suppose the modules m_1 and m_2 in M_r are separated by a module c in either S_1 or S_2, or in both S_1 and S_2, as shown in Figure 2. Consider case 1 in which the modules m_1 and m_2 are separated by c in S_1. Suppose the areas of all the modules except m_1, m_2, and c are zero, we can find some module dimensions for m_1, m_2, and c such that the modules in M_r do not form a cluster. A similar argument follows for cases 2–10. This means that the modules in M_r may not form a cluster in the packing depending on the module dimensions if M_r is not a rearrangable module set (i.e., not forming contiguous subsequences in both S_1 and S_2).

As a result, a set of modules is a rearrangable module set if and only if the modules form a cluster in F independent of the module dimensions. □

3.2 Finding Rearrangable Module Sets

In order to construct all the alternative packings (alternative sequence-pairs) of a given floorplan F, we need to find all the clusters (rearrangable module sets) in F. In this section, we will discuss how we can find all the rearrangable module sets from a given sequence pair effectively. Consider a sequence pair (S_1, S_2), where $S_1 = a_1a_2 \ldots a_n$ and $S_2 = b_1b_2 \ldots b_n$, and where n is the total number of modules. If a contiguous subsequence in S_1 contains two modules, the subsequence should be (a_ia_{i+1}) for some $1 \leq i \leq n-1$. If a contiguous subsequence in S_1 contains three modules, the subsequence should be $(a_ia_{i+1}a_{i+2})$ for some $1 \leq i \leq n-2$. Similarly, if a contiguous subsequence in S_1 contains $n-1$ modules, the subsequence should be $(a_1 \ldots a_{n-1})$ or $(a_2 \ldots a_n)$. Notice that the subsequences cannot contain more than $n-1$ modules. The total number m of contiguous subsequences in S_1 is computed as

$$\begin{aligned} m &= 2 + \cdots + (n-2) + (n-1) \\ &= \frac{(n+1)(n-2)}{2}. \end{aligned}$$

```
FRMS Algorithm
Input:    Total no. of modules, $n$
          A sequence pair, $(a_1...a_n, b_1...b_n)$
Output:   Total no. of rearrangable module sets, $cnt$
          The starting positions of the rearrangable module sets, $rs\_s_i$
          The ending positions of the rearrangable module sets, $rs\_t_i$
$cnt = 0$
for each $j$ where $1 \leq j \leq n$
    $subSP\_low_j = pos(a_j)$
    $subSP\_up_j = pos(a_j)$ ($pos(a_j)$ returns the position of $a_j$ in $S_2$)
for $i = 2$ to $n$
    for $j = i - 1$ downto 1
        if $subSP\_low_j > pos(a_i)$
            $subSP\_low_j = pos(a_i)$
        if $subSP\_up_j < pos(a_i)$
            $subSP\_up_j = pos(a_i)$
        if $(subSP\_up_j - subSP\_low_j) = (i - j)$
            $rs\_t_{cnt} = subSP\_up_j$
            $rs\_s_{cnt} = subSP\_low_j$
            $cnt = cnt + 1$
```

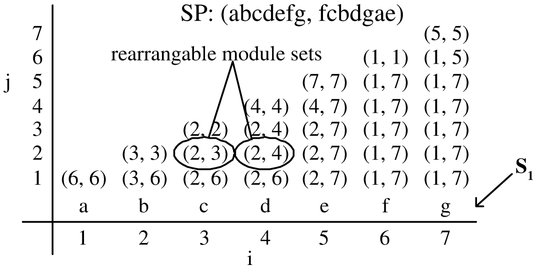
Fig. 3. The algorithm of FRMS.

According to Definition 2, a rearrangable module set in the sequence pair (S_1, S_2) should form contiguous subsequences in both S_1 and S_2. Thus, the maximum number of rearrangable module sets obtained by looking at S_2 is also equal to $\frac{(n+1)(n-2)}{2}$. In order to find all the rearrangable module sets, we can scan all the possible contiguous subsequences from S_1 and check whether they also form a contiguous subsequence in S_2. We propose the algorithm FRMS (Find Rearrangable Module Set) to find all the rearrangable module sets in a sequence pair (S_1, S_2). The FRMS algorithm is shown in Figure 3.

In the algorithm, we try to find all the rearrangable module sets by sequential search. First, we obtain a subsequence s in S_1. We will then find the first and the last positions in S_2 for those modules in s. If the difference between the first and the last positions is equal to the number of modules in s minus 1, then s is a rearrangable module set. In this algorithm, we will scan the subsequences in the following order: a_1a_2, a_2a_3, $a_1a_2a_3$, ... , $a_{n-1}a_n$, $a_{n-2}a_{n-1}a_n$, ... , $a_1 \dots a_n$. Once we find a rearrangable module set, we will store the starting and ending positions of this set in S_2. An example is shown in Figure 4 and the scanning order of this example is ab, bc, abc, cd, bcd, $abcd$, ... , $abcdefg$.

Consider the example in Figure 4, we have the packing represented by the sequence pair $(S_1 : a_1 \dots a_n, S_2 : b_1 \dots b_n)$, where $S_1 = abcdefg$ and $S_2 = fcbdgae$. The x-axis i and the y-axis j are the last and first positions of a subsequence that we consider in S_1, respectively. The entry at each position (i, j) is $(subSP\_low_j, subSP\_up_j)$. Notice that these entries are generated column by column. Each newly generated column will overwrite the previous column dynamically.

At the beginning, both $subSP\_low_j$ and $subSP\_up_j$ are initialized to $pos(a_j)$ for all j, where $pos(a_j)$ is the position of module a_j in S_2. When we consider $i = 2$ and $j = 1$, this corresponds to the subsequence ab, namely the subsequence of



j							
7							(5, 5)
6						(1, 1)	(1, 5)
5					(7, 7)	(1, 7)	(1, 7)
4				(4, 4)	(4, 7)	(1, 7)	(1, 7)
3			(2, 2)	(2, 4)	(2, 7)	(1, 7)	(1, 7)
2		(3, 3)	(2, 3)	(2, 4)	(2, 7)	(1, 7)	(1, 7)
1	(6, 6)	(3, 6)	(2, 6)	(2, 6)	(2, 7)	(1, 7)	(1, 7)
	a	b	c	d	e	f	g
	1	2	3	4	5	6	7

Fig. 4. An example of finding rearrangable module sets by FRMS.

S_1 from position 1 to position 2. The positions of a and b in S_2 are 3 and 6, respectively, so $subSP\_low_1$ is 3 and $subSP\_up_1$ is 6. Then $subSP\_up_1 - subSP\_low_1 + 1$ is equal to 4. This means that the shortest subsequence in S_2 containing both a and b has four modules. However, there are only two modules in the subsequence ab in S_1 (it can be computed by $i - j + 1$). This means that ab does not form a contiguous subsequence in S_2, so $\{a, b\}$ is not a rearrangable module set.

When we consider $i = 3$ and $j = 2$, this corresponds to the subsequence bc, the subsequence of S_1 from position 2 to position 3. The positions of b and c in S_2 are 2 and 3, respectively, so $subSP\_low_2$ is 2 and $subSP\_up_2$ is 3. Hence, $subSP\_up_2 - subSP\_low_2 + 1$ is equal to 2, so the shortest subsequence in S_2 containing both b and c has two modules, which is equal to the number of modules in the subsequence bc. Thus, $\{b, c\}$ forms contiguous subsequences in both S_1 and S_2 and it is a rearrangable module set.

Similarly, when we consider $i = 4$ and $j = 2$, this corresponds to the subsequence bcd. The positions of b, c, and d in S_2 are 2, 3, and 4, respectively, so $subSP\_low_2$ is 2 and $subSP\_up_2$ is 4. Here $subSP\_up_2 - subSP\_low_2 + 1$ is equal to 3. It means that the shortest subsequence in S_2 containing b, c, and d has three modules, which is equal to the number of modules in the subsequence bcd. Thus, $\{b, c, d\}$ forms contiguous subsequences in both S_1 and S_2 and is a rearrangable module set. In this algorithm, we will scan all the possible subsequences in S_1 and the following theorem states the correctness of the algorithm.

THEOREM 2. *Given a sequence pair* (S_1, S_2) *of a packing F, all the rearrangable module sets in* (S_1, S_2) *can be found by the algorithm FRMS.*

3.3 Alternative Sequence-Pairs

In order to construct alternative packings of a given floorplan F described by a sequence pair (S_1, S_2), we need to find the alternative sequence-pairs, which are obtained by rearranging the modules in one or more rearrangable module sets in (S_1, S_2). According to Theorem 1, the modules in a rearrangable module set form a cluster in the packing. The rearrangements should correspond to performing a horizontal, vertical, or diagonal flip to the cluster.

Given a sequence s, we use $\tilde{s}$ to denote the sequence obtained by writing s in the reversed order. For example, if $s = 123$, $\tilde{s} = 321$. We can perform a horizontal, vertical, or diagonal flip by swapping or reversing the rearrangable module sets.

(a) Vertical Flip

Horizontal relationship:

$$(\dots m_i \dots\ m_j \dots,\ \dots m_i \dots\ m_j \dots) \xrightarrow{\text{after swapping}} (\dots m_i \dots\ m_j \dots,\ \dots m_i \dots\ m_j \dots)$$

Vertical relationship:

$$(\dots m_i \dots\ m_j \dots,\ \dots m_j \dots\ m_i \dots) \xrightarrow{\text{after swapping}} (\dots m_j \dots\ m_i \dots,\ \dots m_i \dots\ m_j \dots)$$

(b) Horizontal Flip

Horizontal relationship:

$$(\dots m_i \dots\ m_j \dots,\ \dots m_i \dots\ m_j \dots) \xrightarrow{\text{after reversal}} (\dots m_j \dots\ m_i \dots,\ \dots m_j \dots\ m_i \dots) \xrightarrow{\text{after swapping}} (\dots m_j \dots\ m_i \dots,\ \dots m_j \dots\ m_i \dots)$$

Vertical relationship:

$$(\dots m_i \dots\ m_j \dots,\ \dots m_j \dots\ m_i \dots) \xrightarrow{\text{after reversal}} (\dots m_j \dots\ m_i \dots,\ \dots m_i \dots\ m_j \dots) \xrightarrow{\text{after swapping}} (\dots m_i \dots\ m_j \dots,\ \dots m_j \dots\ m_i \dots)$$

(c) Diagonal Flip

Horizontal relationship:

$$(\dots m_i \dots\ m_j \dots,\ \dots m_i \dots\ m_j \dots) \xrightarrow{\text{after reversal}} (\dots m_j \dots\ m_i \dots,\ \dots m_j \dots\ m_i \dots)$$

Vertical relationship:

$$(\dots m_i \dots\ m_j \dots,\ \dots m_j \dots\ m_i \dots) \xrightarrow{\text{after reversal}} (\dots m_j \dots\ m_i \dots,\ \dots m_i \dots\ m_j \dots)$$

Fig. 5. Operations to obtain alternative sequence-pairs.

3.3.1 *Vertical Flip.* Figure 1 shows a packing corresponding to the sequence pair (1264753, 4567132) with a rearrangable module set {4, 5, 6, 7}. To perform a vertical flip on the cluster formed by $(T_1, T_2) = (6475, 4567)$, we swap T_1 and T_2: $T_{v1} = T_2$ and $T_{v2} = T_1$. Before swapping, if a module m_i is on the left of m_j in the original packing, m_i should be in front of m_j in both T_1 and T_2. After swapping, m_i is still in front of m_j in both T_{v1} and T_{v2}. Thus, the horizontal relationships between the modules are preserved. On the other hand, if a module m_i is above m_j before swapping, m_i should be in front of m_j in T_1 and after m_j in T_2. After swapping, m_i is after m_j in T_{v1} and in front of m_j in T_{v2}. Thus, the vertical relationships between the modules will be reversed. An illustration is shown in Figure 5(a).

3.3.2 *Horizontal Flip.* When we perform horizontal flip, we reverse and swap the sequences: $T_{h1} = \tilde{T}_2$ and $T_{h2} = \tilde{T}_1$. If a module m_i is on the left of m_j in the original packing, m_i should be in front of m_j in both T_1 and T_2 before the reversal, but m_i will be after m_j in both sequences after the reversal. Then we perform swapping, which has no effect on the horizontal relationships between the modules. As a result, the horizontal relationships between the modules will be reversed. On the other hand, if a module m_i is above m_j in the original packing, m_i should be in front of m_j in T_1 and after m_j in T_2. After the reversal, m_i will come after m_j in $\tilde{T}_1$ and in front of m_j in $\tilde{T}_2$. We then perform swapping by which the vertical relationships will be reversed once again. As a result, the vertical relationships between the modules are unchanged. An illustration is shown in Figure 5(b).

3.3.3 *Diagonal Flip.* Finally, we perform a diagonal flip. Actually, this can be considered as performing either of a horizontal or vertical flip first, which is

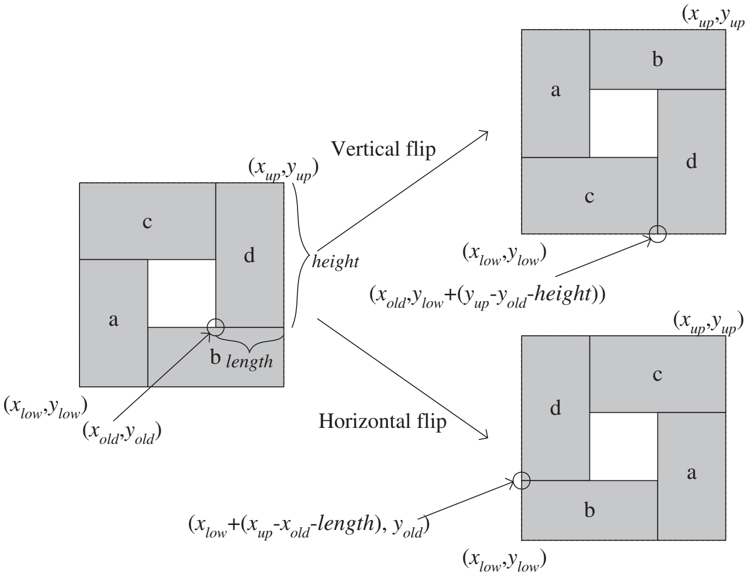


Fig. 6. Recalculation of module positions.

then followed by the other one. Thus $T_{d1} = T_{h2} = \tilde{T}_1$ and $T_{d2} = T_{h1} = \tilde{T}_2$, or $T_{d1} = \tilde{T}_{v2} = \tilde{T}_1$ and $T_{d2} = \tilde{T}_{v1} = \tilde{T}_2$. As a result, $T_{d1} = \tilde{T}_1$ and $T_{d2} = \tilde{T}_2$. An illustration is shown in Figure 5(c).

After finding all rearrangable module sets, we can obtain the alternative packings by applying vertical flips, horizontal flips, or diagonal flips on those subsequences.

3.4 Recalculation of Module Positions

After we have found an alternative sequence-pair, we need to recalculate the module positions in order to obtain the new interconnect cost. It is time consuming if we need to reconstruct the horizontal and vertical constraint graphs for each alternative sequence-pair.

In our floorplanner, we can calculate the new positions of the modules in the alternative packings by the following method. An example is shown in Figure 6. First of all, a rearrangable module set will form a cluster in the packing. We can thus obtain the coordinates of the upper right corner (x_{up}, y_{up}) and the coordinates of the lower left corner (x_{low}, y_{low}) of this cluster. Then we can compute the new positions of the modules in the rearrangable module sets by the following equations, according to the operations.

Horizontal flip:

$$\begin{aligned} x_{new} &= x_{low} + (x_{up} - x_{old} - width) \\ y_{new} &= y_{old} \end{aligned} \tag{1}$$

Table I. Information on the Testing Circuits

Test Cases	No. of Modules	No. of Nets	No. of Pins	No. of Pads
apte	9	97	264	73
xerox	10	203	696	2
hp	11	83	214	45
*ami*33	33	123	480	42
*ami*49	49	408	931	22

Vertical flip:

$$\begin{aligned} x_{new} &= x_{old} \\ y_{new} &= y_{low} + (y_{up} - y_{old} - height) \end{aligned} \tag{2}$$

Diagonal flip:

$$\begin{aligned} x_{new} &= x_{low} + (x_{up} - x_{old} - width) \\ y_{new} &= y_{low} + (y_{up} - y_{old} - height) \end{aligned} \tag{3}$$

Here (x_{old}, y_{old}) consists of the coordinates of the lower left corner of the module before flipping, (x_{new}, y_{new}) of the co-ordinates of the lower left corner of the module after flipping, and *width* and *height* are the width and height of the module, respectively.

From the preceding equations, we can find the new positions of the modules in an alternative packing very quickly.

3.5 Time Complexity

According to the algorithm of FRMS, we need to scan all possible subsequences. If the packing contains n modules, there will be two sub-sequences with $n-1$ modules, three with $n-2$ modules, ... , and $n-1$ subsequences with two modules. Therefore, we need to scan $2+3+\cdots+n-1$ times, which is equal to $\frac{(n-2)(n+1)}{2}$. As a result, the time complexity is $O(n^2)$. The running time for flipping depends on the number of clusters (L) only and the time complexity for flipping is $O(L)$.

4. EXPERIMENTAL RESULTS

In the experiments, the test cases used are the Microelectronics Center of North Carolina (MCNC). Detailed information from the testing circuits is shown in Table I. The preplaced floorplans are obtained from the floorplanner Parquet 4.5 [Adya and Markov 2003, 2002, 2001] with "*minWL*" turned on ("*areaWeight*" = 0.1 and "*wireWeight*" = 0.9). The initial floorplans are already well optimized in terms of wirelength by this floorplanner. We obtained the sequence pair from the floorplan results and our approaches have been applied (searching alternative packings) to improve the total wirelength of the floorplans.

The wirelength and area of the floorplans are shown in Table II. Results show that applying our approaches can maintain the area of the floorplans and can also improve the total wirelength with very small additional runtime. The additional runtime is small because the number of alternative packings is not large.

Table II. Comparisons With Floorplanner Parquet

Cases	Wirelength with Our Approach ($10^3\mu m$)	Wirelength ($10^3\mu m$)	Area ($10^3\mu m^2$)	Runtime of Our Approach (s)
apte	483330.62	502614.56	48478.2	0.01
xerox	466038.38	467066.94	21221.5	0.01
hp	186234.67	187214.34	10546.4	0.01
*ami*33	51794.37	52060.37	1301.1	0.06
*ami*49	786826.69	791153.56	43842.5	0.18

Table III. Maximum and Average Improvement on Wirelength and Routability

	Improvement on Total Wirelength		Improvement on no. of Overflow Tiles	
Cases	Average	Maximum	Average	Maximum
apte	0.88%	1.53%	1.05%	1.34%
xerox	0.28%	1.03%	0.15%	0.51%
hp	0.18%	0.53%	0.12%	0.14%
*ami*33	0.57%	1.26%	0.25%	0.57%
*ami*49	0.76%	1.25%	0.47%	0.74%

The total wirelength can be reduced because the simulated-annealing-based floorplanner can give floorplans with short but not minimum total wirelength.

In addition, the wirelength and routability are also tested by global routing. It is attempted to route all nets by the maze router [Kastner et al. 2002]. We have also counted the number of overflow tiles. Table III shows the improvement in wirelength and routability after searching alternative packings of the final floorplan solution. We can see that the promised average improvement in both wirelength and routability by finding alternative packings are fulfilled. In addition, the maximum improvement is significant. As the time penalty for searching alternative packings of the final floorplan solution is small, the improvement on wirelength is guaranteed. It is desirable to apply this technique.

From the experiments, we have also observed that the average number of clusters in a nonslicing packing is very small. Consider the case with the number of modules from 2–30, we generate 1000 sequence pairs from the wirelength-driven floorplanner in each case. The average number of clusters is between 1 and 3 when the number of modules in the packing is below 30. Figure 7(a) is obtained by finding the average number of clusters in all possible sequence-pairs with the number of modules ranging from 3–30. We believe that the result will be similar when the number of modules in a packing becomes larger. In order to obtain the best packing, we should find all the alternative packings by trying all possible flippings of the clusters. As the average number of clusters in a nonslicing packing is only about 3, we need to check on average $4^3 = 64$ alternative packings.

The results show that the probability of a sequence pair having at least one cluster is larger than 80% (Figure 7(b)). This means that the probability that we can improve the interconnect cost of a packing by applying this method to find the best alternative packing is also large.

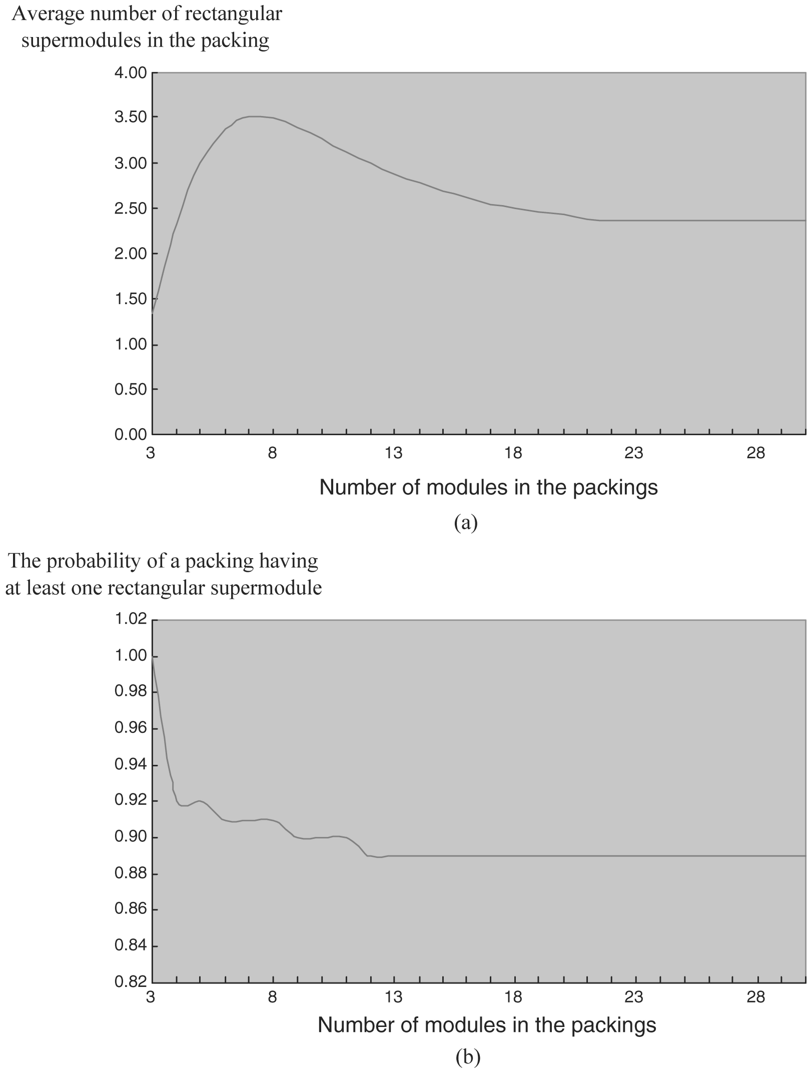


Fig. 7. Number of clusters in nonslicing floorplans.

5. CONCLUSION

In this article, we present a postfloorplanning step to reduce interconnect cost by searching alternative packings. We observed that if a packing F contains some clusters, we can rearrange the blocks in the clusters to obtain a new packing with the same area as F, but with a possibly improved interconnect cost. In our implementation, we need to identify the clusters in a sequence pair and find all the rearrangable module sets. We hence obtain the alternative sequence-pairs by performing swapping or reversing on the rearrangable module sets. The positions of the alternative packings can be calculated easily without reconstruction of the horizontal- and vertical-constraint graphs. According to the experimental results, floorplanners after applying our proposed postfloorplanning step can reduce interconnect cost without any penalty in area, and the additional runtime is small.

REFERENCES

Adya, S. N. and Markov, I. L. 2001. Fixed-outline floorplanning through better local search. *IEEE Trans. VLSI Syst.*, 328–333.

Adya, S. N. and Markov, I. L. 2002. Consistent placement of macro-blocks using floorplanning and standard-cell placement. In *Proceedings of the International Symposium on Physical Design*, 12–17.

Adya, S. N. and Markov, I. L. 2003. Fixed-Outline floorplanning: Enabling hierarchical design. In *Proceedings of the International Conference on Computer Design*, 1120–1135.

Chang, C. C., Cong, J., Pan, D. Z., and Yuan, X. 2000. Interconnect-Driven floorplanning with fast global wiring planning and optimization. In *Proceedings of the SRC Techcon Conference*, 21.

Chen, H. M., Zhou, H., Young, F. Y., Wong, D., Yang, H. H., and Sherwani, N. 1999. Integrated floorplanning and interconnect planning. In *Proceedings of the IEEE International Conference on Computer-Aided Design*, 354–357.

Kastner, R., Bozorgzadeh, E., and Sarrafzadeh, M. 2002. Pattern routing: Use and theory for increasing predictability and avoiding coupling. *IEEE Trans. Comput. Aided Des. Integr. Circ. Syst.*, 777–780.

Lai, S. T. W., Young, E. F. Y., and Chu, C. C. N. 2003. A new and efficient congestion evaluation model in floorplanning: Wire density control with twin binary trees. In *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition*.

Lou, J., Krishnamoorthy, S., and Sheng, H. S. 2001. Estimating routing congestion using probabilistic analysis. In *Proceedings of the International Symposium on Physical Design*, 112–117.

M. C. Jeske, B. Y. W. and Greenwood, G. 2003. Floorplanning with performance-based clustering. In *Proceedings of the IEEE International Symposium on Circuits and Systems*, 724–727.

Ma, Y. C., Hong, X. L., Dong, S. Q., Chen, S., Cai, Y. C., Cheng, C. K., and Gu, J. 2003. Dynamic global buffer planning optimization based on detail block locating and congestion analysis. In *Proceedings of the ACM/IEEE Design Automation Conference*, 806–811.

Murata, H., Fujiyoshi, K., Nakatake, S., and Kajitani, Y. 1995. Rectangle-Packing-Based module placement. In *Proceedings of the IEEE International Conference on Computer-Aided Design*, 472–479.

Murata, H., Fujiyoshi, K., Nakatake, S., and Kajitani, Y. 1996. VLSI module placement based on rectangle-packing by the sequence-pair. *IEEE Trans. Comput. Aided Des. Integr. Circ. Syst.*, 1518–1524.

Murata, H. and Kuh, E. S. 1998. Sequence-Pair based placement methods for hard/soft/pre-placed modules. In *Proceedings of the International Symposium on Physical Design*, 167–172.

Sham, C. W. and Young, E. F. Y. 2003. Routability-Driven floorplanning with buffer planning. *IEEE Trans. Comput. Aided Des. Integ. Circ. Syst.*, 470–480.

Wang, M. and Sarrafzadeh, M. 2000. Modeling and minimization of routing congestion. In *Proceedings of the ASP-ACM/IEEE Design Automation Conference*, 185–190.

Received June 2006; revised June 2007; accepted July 2007