Interconnect-Driven Floorplanning by Searching Alternative Packings

Chiu-wing Sham
Department of
Computer Science & Engineering
Chinese University of Hong Kong
Hong Kong
852-26098344
cwsham@cse.cuhk.edu.hk

Evangeline F. Y. Young
Department of
Computer Science & Engineering
Chinese University of Hong Kong
Hong Kong
852-26098401
fyyoung@cse.cuhk.edu.hk

Hai Zhou
Department of
Electrical & Computer Engineering
Northwestern University
Evanston, 60208-3118
847-4914155
haizhou@synopsys.com

Abstract— In traditional floorplanners, area minimization is an important issue. Due to the recent advances in VLSI technology, the number of transistors in a design and their switching speeds are increasing rapidly. This results in the increasing importance of interconnect delay and routability of a circuit. We should consider interconnect planning and buffer planning as soon as possible. In this paper, we propose a method to reduce interconnect cost of a floorplan by searching alternative packings. We found that if a floorplan F contains some rectangular supermodules, we can rearrange the blocks in the supermodule to obtain a new floorplan with the same area as F but possibly with a smaller interconnect cost. Experimental results show that we can always reduce the interconnect cost of a floorplan without any penalty in area and runtime by using this method.

I. Introduction

Floorplanning plays an important role in physical design of VLSI circuits. It plans the shapes and locations of the modules on a chip, and the result of which will greatly affect the overall performance of the final circuit. In the past, area minimization is the major concern in floorplan design. As technology develops rapidly, VLSI circuits continue to scale down. Sizes of transistors are getting smaller and a significant portion of circuit delay is coming from interconnects. In some advanced systems today, as much as 80% of the clock cycle is consumed by interconnects [2]. The domination of interconnect in system performance has made area minimization less important while routability and delay become the major concern in floorplanning and many other designing steps.

There are several previous works addressing the interconnect issues in floorplan design. In the paper [5, 3, 9, 2, 1], the authors formulated different congestion-related cost functions (evaluated by some simple global routing) including a hybrid length plus congestion cost function and these cost functions are then optimized by applying some heuristics such as simulated annealing and genetic algorithm. In the paper [2], wires are assumed to be routed in either L-shaped or Z-shaped in their congestion estimation. However, the paper [8] showed that the congestion-related cost functions cannot reduce congestion effectively. It is because simple global routing may have a great difference with the final detail routing. Lou et al. [4] applied probabilistic analysis to estimate congestion and routability, and they showed that their estimations correlate well with post-route congestion. Besides that, papers [7, 6] also used probabilistic analysis in congestion estimation with buffer planning.

Even though interconnect-driven floorplanners can perform congestion optimization and buffer planning effectively, there is a significant penalty on runtime. In this paper, we propose an approach to reduce interconnect cost by searching alternative packings. We have found that if there is a packing F containing some rectangular supermodules, we can rearrange the blocks in the supermodules to obtain a new packing with the same area as F but with possibly smaller interconnect cost. In general, we can apply this approach on the final floorplan solution for different purposes such as congestion reduction and buffer planning.

This paper is organized as follows. There will be an overview of our method in section II. We will discuss the method of searching alternative packings in details in section III. The implementation of a floorplanner with alternative packing searching will be discussed in section IV. Experimental results will be shown in section V. Finally, we will give a conclusion in section VI.

II. Overview of the Method

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

Definition 1 *Given a floorplan F of a set of modules, an alternative packing of F is another floorplan that has the same area as F, independent of the module dimensions.*

An example of the alternative packings is shown in figure 1. In our method, we want to find all the alternative packings of a floorplan solution. Generally, alternative

packings can be obtained by flipping some rectangular supermodules in a floorplan horizontally, vertically or both horizontally and vertically.

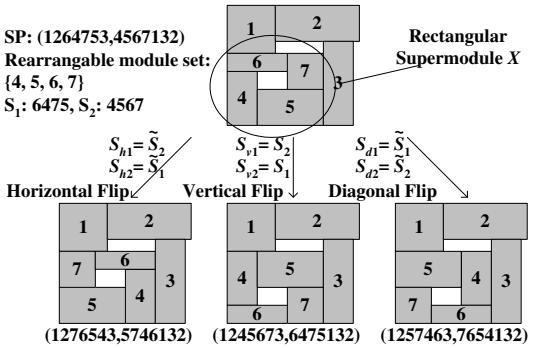

Fig. 1. Example of alternative packings

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

III. Searching Alternative Packings

In figure 1, there is a rectangular supermodule X containing four modules in the packing. We can change the packings while preserving the area by three ways: flipping X horizontally, flipping X vertically and flipping X horizontally and vertically (we call this a diagonal flip). In the method of searching alternative packings, we will use sequence pair to represent a floorplan. We can obtain alternative packings with the same area by working on the sequence pair representations only. In this section, we will discuss how we can find an alternative sequence pair from a given sequence pair (S_1, S_2) such that the packing represented by the alternative sequence pairs will have the same area as that represented by (S_1, S_2) independent of the dimensions of the modules.

A. Rectangular Supermodules in Sequence Pair

In order to construct alternative packings, we need to find all the rectangular supermodules in the 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 forms a rearrangable module set if they form contiguous sub-sequences 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 sub-sequences in both S_1 and S_2. Thus, the set of modules $\{4, 5, 6, 7\}$ is a rearrangable module set according to the above definition. In addition, the set of modules $\{4, 5, 6, 7\}$ will form a rectangular supermodule in the packing according to the following theorem.

Theorem 1 *Given a sequence pair (S_1, S_2) and its corresponding packing F, the set of modules in a rearrangable module set will form a rectangular supermodule in F independent of the dimensions of the modules.*

Proof Given a set of modules M_r in a rearrangable module set, the modules in M_r form a contiguous sub-sequence s_{r1} in S_1 and form a contiguous sub-sequence s_{r2} 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 $(...m_i...s_{r1}..., ...m_i...s_{r2}...)$. It 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 of the given floorplan should be $(...m_i...s_{r1}..., ...s_{r2}...m_i...)$. 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. As a result, the set of modules in a rearrangable module set will form a rectangular supermodule in the packing independent of the dimensions of the modules. □

B. 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 rectangular supermodules (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 $S_1 = s_{11}s_{12}...s_{1n}$ and $S_2 = s_{21}s_{22}...s_{2n}$ where n is the total number of modules. If a contiguous subsequence in S_1 contains two modules, the sub-sequence should be $(s_{1i}s_{1i+1})$ for some $1 \leq i \leq n-1$. If a contiguous sub-sequence in S_1 contains three modules, the subsequence should be $(s_{1i}s_{1i+1}s_{1i+2})$ for some $1 \leq i \leq n-2$. Similarly, if a contiguous sub-sequence in S_1 contains $n-1$ modules, the sub-sequence should be $(s_{11}...s_{1n-1})$ or $(s_{12}...s_{1n})$. Notice that the sub-sequences cannot contain more than $n-1$ modules. The total number of contiguous sub-sequences in S_1:

$$= \begin{matrix} 2 + ... + (n-2) + (n-1) \\ \frac{(n+1)(n-2)}{2} \end{matrix}$$

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 sub-sequences from S_1 or S_2 and check whether they also form a contiguous sub-sequence in the other sequence. We propose the algorithm FRMS (Find Rearrangable Module Set) to find all the rearrangable module sets. The FRMS algorithm is shown in figure 2.

FRMS Algorithm

Input: Total no. of modules, n
A sequence pair, $(s_{11}...s_{1n}, s_{21}...s_{2n})$
Output: Total no. of rearrangable module sets, *count*
An array of the rearrangable module sets, $rs[]$

$count = 0$
for each j where $1 \leq j \leq n$
  $ssp[j].low = s_{1j}.pos$
  $ssp[j].up = s_{1j}.pos$ ($s_{1j}.pos$ is the position of s_{1j} in S_2)
for $i = 2$ to n
  for $j = i - 1$ downto 1
    if $ssp[j].low > s_{1i}.pos$
      $ssp[j].low = s_{1i}.pos$
    if $ssp[j].up < s_{1i}.pos$
      $ssp[j].up = s_{1i}.pos$
    if $(ssp[j].up - ssp[j].low) = (i - j)$
      $rs[count].end = ssp[j].up$
      $rs[count].start = ssp[j].low$
      $count = count + 1$

Fig. 2. The algorithm of FRMS

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 can then find the first and the last position in S_2 for those modules in s. If the difference between the first and the last position is equal to the number of modules in s, s is a rearrangable module set. In this algorithm, we will scan the sub-sequences in following order: $s_{11}s_{12}$, $s_{12}s_{13}$, $s_{11}s_{12}s_{13}$, ..., $s_{1n-1}s_{1n}$, $s_{1n-2}s_{1n-1}s_{1n}$, ..., $s_{11}...s_{1n}$. Once we have found a rearrangable module set, we will store the starting and ending position of this set in S_2 in the array $rs[]$. An example is shown in figure 3.

Consider the example in figure 3, we have the packing represented by the sequence pair (S_1, S_2) where $S_1 = abcdefg$ and $S_2 = fcbdgae$. The x-axis i is the last position of a sub-sequence in S_1 and the y-axis j is the first position of a sub-sequence in S_1. The entry at each position (i, j) is $(ssp[j].low, ssp[j].up)$. Notice that these entries are generated column by column. Each newly generated column will overwrite the previously generated column dynamically.

At the beginning, both $ssp[j].low$ and $ssp[j].up$ are initialized to $s_{1j}.pos$ for all j where $s_{1j}.pos$ is the position of module s_{1j} in S_2. When we consider $i = 2$ and $j = 1$, this corresponds to the sub-sequence ab, the sub-sequence of S_1 from position 1 to position 2. The positions of a and b in S_2 are 3 and 6 respectively, so $ssp[1].low$ is 6 and $ssp[1].up$ is 3. Then $ssp[1].up - ssp[1].low + 1$ is equal to 4. It means that the shortest sub-sequence in S_2 containing both a and b has four modules. However, there are only two modules in the sub-sequence ab in S_1 (it can be computed by $i - j + 1$). It means that ab cannot form a contiguous sub-sequence in S_2, so it is not a rearrangable module set.

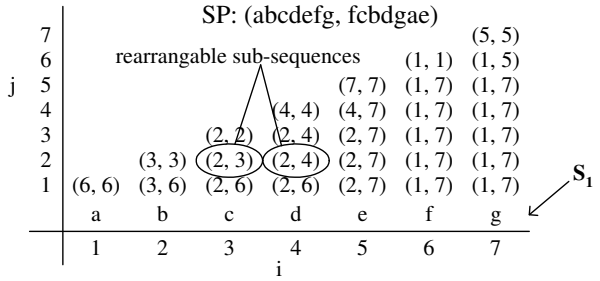

Fig. 3. An example of finding rearrangable module sets by FRMS

When we consider $i = 3$ and $j = 2$, this corresponds to the sub-sequence bc, the sub-sequence of S_1 from position 2 to position 3. The positions of b and c in S_2 are 2 and 3 respectively, so $ssp[2].low$ is 2 and $ssp[2].up$ is 3. $ssp[2].up - ssp[2].low + 1$ is equal to 2, so the shortest subsequence in S_2 containing both b and c has two modules. Since there are also two modules in the sub-sequence bc, bc forms contiguous sub-sequences 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 sub-sequence bcd. The positions of b, c and d in S_2 are 2, 3 and 4 respectively, so $ssp[2].low$ is 2 and $ssp[2].up$ is 4. $ssp[2].up - ssp[2].low + 1$ is equal to 3. It means that the shortest sub-sequence in S_2 containing b, c and d has three modules. Since there are three modules in the sub-sequence bcd, bcd forms contiguous sub-sequences in both S_1 and S_2 and it is a rearrangable module set. By this algorithm, we will scan all the possible sub-sequences in S_1 and have the following theorem.

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.*

C. 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 corresponding alternative sequence pairs which are obtained by rearranging the modules in one or more

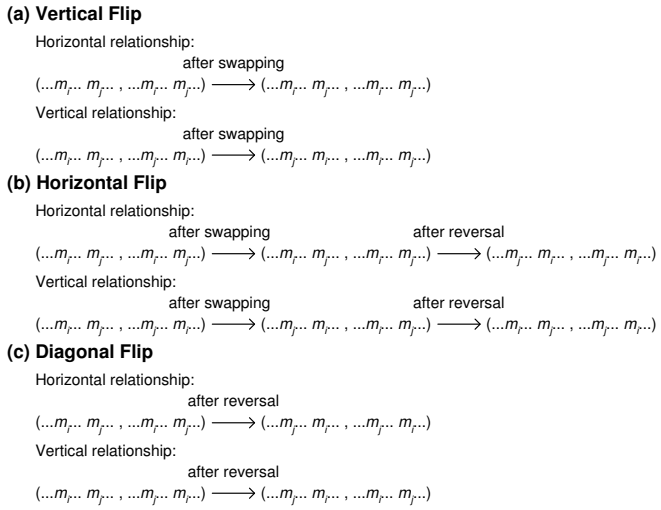

Fig. 4. Vertical flip

rearrangable module sets in (S_1, S_2). According to theorem 1, the modules in a rearrangable module set should form a rectangular supermodule in the packing. The rearrangements should be identical to performing a horizontal flip, vertical flip or diagonal flip on the supermodule.

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 horizontal and vertical flip by swapping or reversing the rearrangable subsequences.

C.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 supermodule formed by $(S_1, S_2) = (6475, 4567)$, we swap S_1 and S_2: $S_{v1} = S_2$ and $S_{v2} = S_1$. Before swapping, if a module m_i is on the left of m_j in the initial packing, m_i should be in front of m_j in both S_1 and S_2. After swapping, m_i is still in front of m_j in both S_{v1} and S_{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 S_1 and after m_j in S_2. After swapping, m_i is after m_j in S_{v1} and in front of m_j in S_{v2}. Thus, the vertical relationships between the modules will be reversed. An illustration is shown in figure 4a.

C.2 Horizontal Flip

When we perform horizontal flip, we reverse and swap the sequences: $S_{h1} = \tilde{S_2}$ and $S_{h2} = \tilde{S_1}$. 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 S_1 and S_2 before the reversal but m_i will be after m_j 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 S_1 and after m_j in S_2. After the reversal, m_i will become after m_j in $\tilde{S_1}$ and in front of m_j in $\tilde{S_2}$. 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 4b.

C.3 Diagonal Flip

Finally, we perform diagonal flip. Actually, this can be considered as performing a horizontal or vertical flip first and then followed by the other one. Thus $S_{d1} = S_{h2} = \tilde{S_1}$ and $S_{d2} = S_{h1} = \tilde{S_2}$, or $S_{d1} = \tilde{S_{v2}} = \tilde{S_1}$ and $S_{d2} = \tilde{S_{v1}} = \tilde{S_2}$. As a result, $S_{d1} = \tilde{S_1}$ and $S_{d2} = \tilde{S_2}$. An illustration is shown in figure 4c.

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

IV. Implementation

In order to evaluate the method of searching alternative packings in improving interconnect cost, we have implemented this method in an interconnect-driven floorplanner that uses half-perimeter estimation on wirelength calculation and uses sequence pair as the floorplan representation. In this floorplanner, we propose a three step process to reduce interconnect cost. First, we find all the rearrangable module sets of a give sequence pair (S_1, S_2) by FRMS. Then the alternative sequence pairs are obtained by swapping and reversing some rearrangable module sets in (S_1, S_2). Finally, we optimize the interconnect cost by selecting the alternative packing with the minimum wirelength. Since all the alternative packings have the same area, there is no area penalty on picking different alternative packings.

A. Re-calculation of Interconnect Cost

After we have found the alternative sequence pairs, we need to re-calculate the interconnect cost. It is time consuming if we need to re-construct the horizontal and vertical constraint graphs whenever an alternative sequence pair is obtained.

In our floorplanner, we can calculate the new positions of the modules in the alternative packings by the following method. First of all, a rearrangable module set will form a rectangular supermodule in the packing. We can thus obtain the co-ordinates of the upper right corner (x_{up}, y_{up}) and the co-ordinates of the lower left corner (x_{low}, y_{low}) of the rectangular supermodule. Then we can estimate the new positions of the modules in the rearrangable module

sets by the following equations according to the operations.

Horizontal flip,

$$x_{new} = x_{low} + (x_{up} - x_{old} - length) \quad (1)$$
$$y_{new} = y_{old}$$

Vertical flip,

$$x_{new} = x_{old} \quad (2)$$
$$y_{new} = y_{low} + (y_{up} - y_{old} - height)$$

Diagonal flip,

$$x_{new} = x_{low} + (x_{up} - x_{old} - length) \quad (3)$$
$$y_{new} = y_{low} + (y_{up} - y_{old} - height)$$

where (x_{old}, y_{old}) is the position of the module before flipping, (x_{new}, y_{new}) is the position of the module after flipping, and *length* and *height* are the length and height of the module respectively.

From the above equations, we can find the new positions of the modules in the alternative packings efficiently.

B. Cost Function

We use simulated annealing in our floorplanner. Simulated annealing is an iterative and non-deterministic optimization technique. We use the following cost function:

$$Cost = Area + \alpha \times Wire$$

where *Area* is the area of the floorplan, *Wire* is the total wirelength (half-perimeter estimation is used), α is a parameter. This parameter will be set at the beginning of the simulated annealing process according to the ratio of importance of the area term and the wirelength term. This can be done by performing a sequence of random walks at the beginning of the annealing process and sampling the average values of these penalty terms. The value of α can then be computed accordingly.

C. Time Complexity

According to the algorithm of FRMS, we need to scan all the possible sub-sequences. If the packing contains n modules, there will be two sub-sequences with $n-1$ modules, three sub-sequences with $n-2$ modules, ..., and n sub-sequences with one module. Therefore, we need to scan $2+3+...+n-1+n$ times which is equal to $\frac{(n+2)(n-1)}{2}$. As a result, the time complexity is $O(n^2)$.

V. Experimental Results

We have implemented two floorplanners, a floorplanner $F1$ based on simulated annealing that applies the three step process of searching alternative packings to reduce the interconnect cost of the final floorplan solution only and a floorplanner $F2$ that applies the three step process to reduce the interconnect cost of every intermediate floorplan solution in the annealing process.

In the experiments, we use half-perimeter estimation in wirelength computation. The data sets used in the experiments are *hp*, *ami*33, *ami*49, *playout* and six randomly generated data sets. Each result shown in table II and III is the average value obtained by performing an experiment eight times. The information of the data sets is shown in table I.

Cases	No. of modules	No. of Nets	Cases	No. of modules	No. of Nets
*c*30_1	30	1000	*c*50_3	50	2500
*c*30_2	30	1000	*hp*	10	83
*c*30_3	30	1000	*ami*33	33	123
*c*50_1	50	2500	*ami*49	49	408
*c*50_2	50	2500	*playout*	62	1161

TABLE I
Information of data sets

Table II shows the improvement in wirelength after searching alternative packings of the final floorplan solution. As the time penalty for searching alternative packings of the final floorplan solution is so small, i.e., less than 0.1% for all test cases, and the improvement on wirelength is guaranteed, it is desirable to apply the method. However, the average improvement on wirelength is not big because the number of alternative packings is limited and the final solution is already optimized in interconnect cost by the annealing process.

From table II, we can see that the improvement will increase when the weight of wirelength in the cost function is reduced. It is because when the weight of wirelength in the cost function is reduced, the wirelength will be less optimized in the solution of the simulated annealing process. This may result in an increase in the differences in wirelength between a packing and its alternative packings. As a result, both the average improvement and the maximum improvement increase when the weight of wirelength in the cost function is reduced.

Comparing the floorplanner $F1$ and $F2$ in table III, we can see that the result between $F1$ and $F2$ is similar. However, the runtime of $F1$ is much faster than that of $F2$. We can conclude that applying the three step process of searching alternative packings to reduce the interconnect cost of the final floorplan solution is good enough and there is no need to apply the method to all intermediate floorplan solutions in the annealing process.

VI. Conclusion

In this paper, we present a new method to reduce interconnect cost by searching alternative packings. We have

Cases	[1] $r = 2$		$r = 4$	
	Mean Improvement in Wirelength	Max. Improvement in Wirelength	Mean Improvement in Wirelength	Max. Improvement in Wirelength
*c*30_1	0.11%	0.88%	0.32%	1.05%
*c*30_2	0.06%	0.48%	0.07%	0.53%
*c*30_3	0.13%	1.04%	0.32%	1.55%
*c*50_1	0.12%	0.95%	0.23%	1.74%
*c*50_2	0.03%	0.28%	0.12%	0.93%
*c*50_3	0.08%	0.67%	0.46%	1.94%
hp	0.18%	1.28%	0.52%	4.25%
*ami*33	0.16%	0.95%	0.36%	2.74%
*ami*49	0.06%	0.39%	0.26%	2.01%
playout	0.06%	0.19%	0.07%	0.49%

[1] $\alpha \approxeq \frac{wirelength*r}{area}$

TABLE II
MAXIMUM AND AVERAGE IMPROVEMENT ON WIRELENGTH BY APPLYING THE METHOD OF SEARCHING ALTERNATIVE PACKINGS ON THE FINAL SOLUTION FLOORPLAN

Cases		Area ($10^3 \mu m^2$)	Wirelength ($10^3 \mu m$)	[1] New Wirelength ($10^3 \mu m$)	Runtime (s)
*c*30_1	$F1$	561.64	656.83	652.56	73.98
	$F2$	562.30	655.49	—	381.79
*c*30_2	$F1$	355.14	592.47	591.32	71.59
	$F2$	356.72	589.98	—	441.64
*c*30_3	$F1$	279.36	525.12	521.08	73.18
	$F2$	280.42	522.08	—	539.19
*c*50_1	$F1$	578.54	1824.25	1820.81	188.55
	$F2$	577.98	1815.46	—	708.63
*c*50_2	$F1$	509.09	1786.42	1784.25	188.06
	$F2$	511.21	1783.15	—	612.93
*c*50_3	$F1$	929.61	2392.52	2380.09	192.37
	$F2$	928.98	2382.12	—	639.12
hp	$F1$	9.881	5.399	5.389	1.69
	$F2$	9.879	5.340	—	7.56
*ami*33	$F1$	1213.47	73.55	73.25	13.49
	$F2$	1213.30	73.49	—	81.79
*ami*49	$F1$	38080	1093.67	1092.09	33.18
	$F2$	37940	1117.91	—	89.09
playout	$F1$	957.75	676.26	675.81	101.36
	$F2$	955.14	680.63	—	308.56

[1] Further optimized by searching alternative packings.

TABLE III
COMPARISONS BETWEEN $F1$ AND $F2$

found that if a packing F contains some rectangular supermodules, we can rearrange the blocks in the supermodules to obtain a new packing with the same area as F but possibly with an improved interconnect cost. In our implementation, we find all the rearrangable module sets in a sequence pair and then obtain the alternative sequence pairs by performing swapping or reversing on the rearrangable module sets. The wirelengths of the alternative packings can be calculated easily without re-construction of the horizontal and vertical constraint graphs. According to the experimental results, our floorplanner can reduce interconnect cost without any penalty in area or runtime.

REFERENCES

[1] C. C. Chang, J. Cong, D. Z. Pan, and X. Yuan. Interconnect-driven floorplanning with fast global wiring planning and optimization. In *Proc. SRC Tech. Conference*, 2000.

[2] H. M. Chen, H. Zhou, F. Y. Young, D. Wong, H. H. Yang, and N. Sherwani. Integrated floorplanning and interconnect planning. In *Proceedings of IEEE Internation Cnference on Computer-Aided Design*, pages 354–357, 1999.

[3] G. Huang, X. L. Hong, C. Qiao, and Y. Cai. A timing-driven block placer based on sequence pair model. In *Proceedings of ASP-ACM/IEEE Design Automation Conference*, pages 249–252, 1999.

[4] J. Lou, S. Krishnamoorthy, and H. S. Sheng. Estimating routing congestion using probabilistic analysis. In *Proceedings of Internation Symposium on Physical Design*, pages 112–117, 2001.

[5] S. M. Sait, H. Youssef, S. Tanvir, and M. S. T. Benten. Timing influenced general-cell genetic floorplanner. In *Proceedings of the ASP-ACM/IEEE Design Automation Conference 95/CHDL 95 /VLSI 95*, pages 135–140, 1995.

[6] C. W. Sham, W. C. Wong, and E. F. Y. Young. Congestion estimation with buffer planning in floorplan design. In *Proceedings of Design, Automation and Test in Europe Conference and Exhibition*, pages 696–701, 2002.

[7] C. W. Sham and E. F. Y. Young. Routability-driven floorplanning with buffer planning. In *Proceedings of Internation Symposium on Physical Design*, pages 50–55, 2002.

[8] M. Wang, X. Yang, and M. Sarrafzadeh. Congestion minimization during placement. In *IEEE Transactions on CAD of Integrated Circuit and System*, pages 1140–1148, October 2000.

[9] H. Youssef, S. M. Sait, and K. J. Al-Farra. Timing influenced force directed floorplanning. In *Proceedings of 32th ACM/IEEE Design Automation Conference*, pages 156–161, 1995.