A Stable Fixed-Outline Floorplanning Method

Song Chen
Graduate School of IPS
Waseda University
Kitakyushu, 808-0135, Japan
chensong@aoni.waseda.jp

Takeshi Yosihmura
Graduate School of IPS
Waseda University
Kitakyushu, 808-0135, Japan
t-yoshimura@waseda.jp

ABSTRACT

In this paper, we propose a stable fixed-outline floorplanning method(IARFP). An elaborated method for perturbing solutions, Insertion after Remove(IAR), is devised for the simulated annealing. The IAR operation uses the technique of enumerating positions in Sequence Pair and greatly accelerates the searching. Moreover, based on the analysis of diverse objective functions used in the existing researches, we suggest a new objective function, which is still effective when combined with other objectives, for the fixed-outline floorplanning. Compared with the previous fixed-outline floorplanners, the proposed method is effective and efficient. Experiments showed that the proposed fixed-outline floorplanner achieved 100% success rate efficiently when optimizing area and wire-length simultaneously, while getting much smaller wirelength. On the other hand, we validated once more by experiments that aspect ratio close to one is beneficial to wire-length.

Categories and Subject Descriptors

B.7.2 [**Integrated Circuits**]: Design Aids

General Terms

Algorithms, design

Keywords

floorplanning, fixed-outline, sequence pair

1. INTRODUCTION

As the advance of IC technology, integrated density keeps increasing. A single chip can integrate more and more functions and often includes millions of transistors. Designs with billions of transistors are also in production. To cope with the increasing design complexity, multilevel hierarchical design is an essential method. Fixed-outline floorplanning (FOFP) enables multilevel hierarchical design[3]. In such a case, aspect ratio and area of floorplans may be imposed by higher level floorplanning, and, hence, a floorplan violating fixed-outline constraint is completely useless. Kahng [10] also pointed out that modern VLSI design is based on a fixed-die (fixed-outline) floorplan rather than a variable-die one.

The FOFP has been shown to be much more difficult than outline-free floorplanning[3]. In [2][3], based on the Sequence Pair(SP) representation[9], the author presented new objective functions to drive simulated annealing and better local search with new types of moves for fixed-outline floorplanning. Liu and et al[12] proposed an FOFP method based on instance augmentation, which started with a subset of blocks and progressively worked on larger subsets until all of the blocks are involved. However, there is no consideration of interconnect. Lin and et al[11] proposed evolutionary search based robust FOFP method, which only took area into account. Chen et al [5][6] presented an adaptive Fast-SA scheme that dynamically changed the weights in the cost function to optimize the wire-length under the outline constraint.

We analyzed the objective functions used in the existing FOFP methods. They all have limitations (Section 3.1) when combined with other objectives. The existing fixed-outline floorplanners can get very high success rate for fixed-outline floorplanning when area is the only objective. However, the success rate is degraded greatly when wire-length is taken into account.

In this paper, based on the analysis of diverse objective functions used in the existing FOFP methods, we suggest a new objective function, which is still effective when combined with other objectives, for the floorplanning with fixed-outline constraint. Moreover, a stable FOFP method IARFP is also proposed, in which an elaborated method for perturbing solutions, Insertion after Remove(IAR), is devised for the simulated annealing. The perturbation method is based on the technique of enumerating positions in Sequence Pair and is able to accelerate the searching greatly. Compared with previous fixed-outline floorplanners, the proposed method is very effective and efficient. Experiments show that the proposed fixed-outline floorplanner achieve 100% success rate efficiently when optimizing area and wire-length simultaneously. Compared with Parquet4.5 and B*-tree based fixed-outline floorplanners, the proposed method achieves, respectively, 13% and 7% improvements of wire-length, while achieving 100% success rate.

Additionally, it is shown in existing researches that results of the FOFP strongly depend on the ratio of the chip

ISPD'07, March 18–21, 2007, Austin, Texas, USA.

width to the chip height (aspect ratio)[3]. However, there is no proper interpretation for this. In this study, by dividing the HPWL(half-perimeter wire-length) into the wire-length along two directions (x-axes and y-axes), we interpret it intuitively and show by experiments that aspect ratio close to one is beneficial to wire-length.

The organization of the paper is as follows. Section 2 gives the problem definition and briefly reviews the SP. Section 3 shows the analysis of objective functions and the FOFP flow. Section 4 elaborates the technique of enumerating insertion points in SP. Section 5 compares the proposed FOFP method with some latest fixed-outline floorplanners and analyzes the relation between aspect ratio and wire-length. And section 6 draws some conclusions.

2. PROBLEM FORMULATION

The formulation of the fixed-outline floorplanning problem is as follows.

Let $S = \{b_i | 1 \leq i \leq n\}$ be a set of rectangular blocks among which connections(nets) exist, and each block b_i has specified width and height, w_i and h_i. The fixed-outline floorplanning is an assignment of tuples (x_i, y_i) to each block such that there is no overlapping between any two blocks, all the blocks are placed inside the specified region (fixed-outline) and some objectives, such as wire-length and etc., are optimal.

For a given collection of blocks with total area A and given maximum whitespace fraction γ, we construct a fixed-outline with aspect ratio $\lambda \geq 1$[3] as follows.

$$W_0 = \sqrt{(1+\gamma)A\lambda}, \quad H_0 = \sqrt{(1+\gamma)A/\lambda}$$

Without special declaration, aspect ratio λ is defined as $W_0/H_0 (> 1)$.

In this paper, the well-studied Sequence Pair[9] is used as floorplan/placement representation.

An SP is a pair of sequences of n elements representing a list of n blocks. It imposes the relationship between each pair of blocks as follows.

$$(< \cdots b_i \cdots b_j \cdots >, < \cdots b_i \cdots b_j \cdots >) \Rightarrow b_i \text{ is left of } b_j$$

$$(< \cdots b_j \cdots b_i \cdots >, < \cdots b_i \cdots b_j \cdots >) \Rightarrow b_i \text{ is above } b_j$$

The original paper that proposed SP presented an $O(n^2)$ algorithm to transform an SP into a floorplan. Tang et al[13] speeded up the evaluation algorithm to $O(nloglogn)$.

3. FIXED-OUTLINE FLOORPLANNING

Fig.1 shows the flow of the proposed fixed-outline floorplanner (IARFP). The initial solution will be generated randomly and then the simulated annealing engine is invoked. Instead of the traditional methods of perturbing solutions, for example, swapping blocks, moving blocks, etc., we devise a complicated perturbation method, which is based on the strategy of Insertion after Remove. O-tree based floorplanning method[8] used a simple remove and insertion perturbation strategy. In this study, we propose an elaborate insertion procedure, which can accelerate the searching greatly.

In fig1, the step of ***'Find an insertion point for the removed block'*** does not just select a random position for the removed block. Instead, candidate insertion points (CIPs) are restricted within a small range. An algorithm is devised to enumerate insertion points in SP(Section 4). During the enumerating, a small number of insertion points are selected as the CIPs for the removed block based on a cost function. The flow of the selection is as follows.

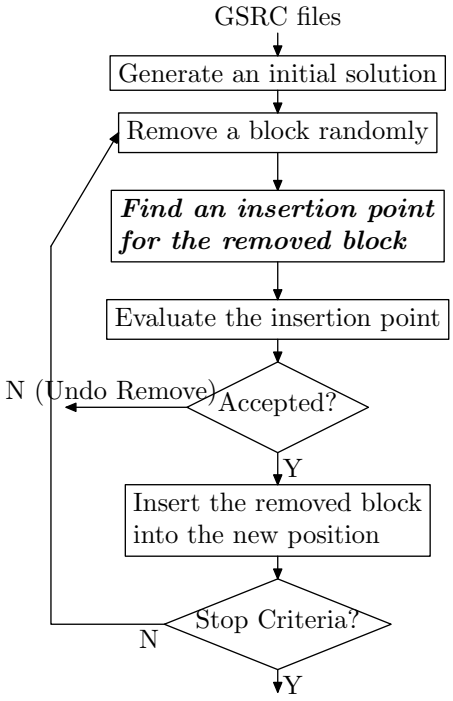


Figure 1: Optimization Flow

a. Compute the floorplan of blocks except the removed one.

b. Enumerate possible insertion points based on the floorplan information obtained in step a and select fixed number of CIPs for the removed block by rough evaluations (The evaluation is accurate if the objective is only area.).

c. Randomly choose for the removed block one of the candidate insertion points selected in step b.

In step b, the insertion points are evaluated by the linear combination of the area cost and the wire-length of nets related to the removed block, whereas the selected insertion point will be evaluated accurately by the objective function. In both of these cases, area cost is computed by the formula (7) defined in the next subsection. The enumeration of insertion points will be discussed in section 4.

3.1 Objective functions

This subsection analyzes the limitations of objective functions used in the existing FOFP methods. Additionally, we proposed a new objective function for FOFP problem. The proposed function works well when used in the case of optimizing multiple objectives.

In [3], the author suggested the following objective functions.

$$max\{W - W_0, 0\} + max\{H - H_0, 0\} \quad (1)$$

$$max\{W - W_0, H - H_0\} \quad (2)$$

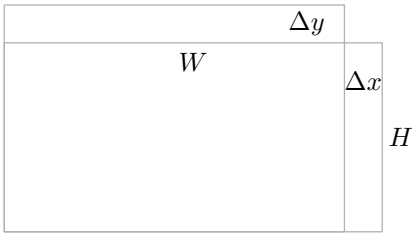


Figure 2: Area and aspect ratio

$$W \cdot H \tag{3}$$

The authors thought that a classic annealer-based floorplanner was practically unable to satisfy the fixed-outline constraints under these objective functions, thus devised slack-based moves for FOFP problem. When taking into account wire-length and other objectives, such functions are certainly ineffective, especially (1) and (2). Because the function values hardly reach zero when the competitions from other objectives exist.

The following functions are also used[12][5].

$$max(H - H_0, 0)/H_0 + max(W - W_0, 0)/W_0 \tag{4}$$

$$W \cdot H + (W/H - W_0/H_0)^2 \tag{5}$$

The formula (4) has the same problem as (1) and (2). The function (5) combined the area and the aspect ratio. Both the objectives perhaps conflict with each other. Therefore, this function potentially causes a tradeoff between the area and the aspect ratio. Fig.2 gives an illustration. Assume W decrease by Δx and H increase by Δy. The new area will be

$$\begin{aligned}&(W - \Delta x) \cdot (H + \Delta y)\\ &= W \cdot H - (H \cdot \Delta x - W \cdot \Delta y + \Delta x \Delta y).\end{aligned} \tag{6}$$

In such a case, the aspect ratio (W/H) is reduced. However, the area may either decrease or increase, which depends on the value of Δx and Δy. The two values are difficult to be predicted based on the solution perturbations. The area will be reduced when $\Delta x/\Delta y \geq W/H$. Otherwise, the change direction of area is difficult to predict.

In this paper, we suggest the following objective function for calculating the area cost in fixed-outline floorplanning.

$$E_W + E_H \cdot \lambda + C_1 \cdot max(E_W, E_H \cdot \lambda) + C_2 \cdot max(W, H \cdot \lambda) \tag{7}$$

where $E_W = max\{W - W_0, 0\}$ and $E_H = max\{H - H_0, 0\}$ are respectively the excessive width and length of the floorplan and C_1 and C_2 are some constants(2 and 1/16 in our experiments). The values related to the height of the chip are scaled by the aspect ratio λ. Without scaling H, experiments show that the smaller one of the width and the height of the fixed-outline will be violated with very large probability when aspect ratio is far from 1.

This objective function keeps effective when combined with other objectives since the width and the height can be reduced further into fixed-outline even though the excessive width and height are close to zero.

On the other hand, as the aspect ratio grows, the last item in equation (7) also increases, which ensures the competitiveness of fixed-outline constraint when wire-length is taken into account. This is because the larger the aspect ratio is $(\lambda \geq 1)$, the longer the wire-length is.(A qualitative explanation is given in the next sub-section)

3.2 Aspect ratio and wire-length

In the HPWL model, the wire-length is estimated by

$$\sum_{nets} (x_{max} - x_{min}) + (y_{max} - y_{min}),$$

where x_{max} is the maximum of x-coordinates of all the net pins(x_{min},y_{max} have similar meaning). We denote the wire-length in x-direction by $wire_x$ $(= \sum_{nets} (x_{max} - x_{min}))$ and in y-direction by $wire_y$ $(= \sum_{nets} (y_{max} - y_{min}))$.

There is a well-known simple geometric observation that the square has the smallest perimeter among all the rectangles with the same area, and the larger the aspect ratio(≥ 1) is, the longer the perimeter is. In the fixed-outline floorplanning, the chip area is fixed. Therefore, the bigger the aspect ratio is, the larger the value of $W_0 + H_0$ is. This can cause different distributions of HPWL wire-length in x-direction (corresponding to W) and y-direction (corresponding to H) from different aspect ratios. Intuitively, larger aspect ratio means larger $wire_x$ and smaller $wire_y$ since many nets perhaps have larger bounding box aspect ratios under larger chip aspect ratios. If the area covered by bounding box are similar, the value of $wire_x + wire_y$ will be larger.

In section 5.2, the experiments are devised to demonstrate this. Area cost are calculated using the formula (7) and the following formula modified from (7), in which the width and height are scaled to the size of the aspect ratio one $(W = H)$ to keep similar area cost for all the aspect ratios.

$$\begin{aligned}E_W/\sqrt{\lambda} + E_H \cdot \sqrt{\lambda} + C_1 \cdot max(E_W/\sqrt{\lambda}, E_H \cdot \sqrt{\lambda})\\ + C_2 \cdot max(W/\sqrt{\lambda}, H \cdot \sqrt{\lambda})\end{aligned} \tag{8}$$

In the experiments, beside computing the HPWL wire-length, we also calculate respectively $wire_x$ and $wire_y$. We found that the wire-length increases as the chip aspect ratio increases. The ratio of the $wire_x$ to $wire_y$ becomes larger and larger as the aspect ratio increases, whereas the geometric mean($\sqrt{x \cdot y}$) of them varies in a much smaller range compared with the change of wire-length. These are consistent with the above simple geometric observation.

4. ENUMERATING INSERTION POINTS

To evaluate an insertion point, we have to calculate the area cost(width and height of the chip) and wire-length. In the rest of this section, we illustrate the key ideas of selecting candidate insertion points for a block by enumerating insertion points based on SP.

There are $(n+1)^2$ positions to insert a block in an SP since in each sequence there are $(n+1)$ candidate positions for insertion. Some of them will be equivalent because of having the same distances to the chip boundaries. In some sense, the redundancy of sequence pair results in this.

4.1 Enumerating single-dimension coordinates

Given a sequence P, let p_i be the i-th block in P, $L^p[b]$ the position of block b in P, p_i^+ and p_i^- the insertion position after the block p_i (before p_{i+1}) and before p_i respectively, and P_i the prefix sequence of P with length i including blocks $p_j, 1 \leq j \leq i$. Given an SP (P, M), We use (p_i^+, m_j^+) to represent a candidate insertion position(CIP).

Let $LCS(P_i, M_j)$ be the weighted longest common subsequence [13] of sequence P_i and M_j.

4.1.1 Two observations on Sequence Pair

First, taking x-coordinate as an example, two observations on Sequence Pair, which are important for enumerating coordinates, are discussed . It is assumed that the origin locate at the left-bottom corner of the chip, and the coordinates of the upper-right corner of a block be the block's coordinates.

Theorem 1. *Assume an SP:* $(p_1, p_2, \cdots, p_i, \cdots, p_n;\ m_1, m_2, \cdots, m_j, \cdots, m_k, \cdots, m_n)$ *where* $p_i = m_j$ *and* $L^p[m_l]_{\{j<l<k\}} > i$. *If we insert a block into the positions* $(p_i^+, m_l^+), j \le l < k$, *the x-coordinate of the block will be* $LCS(P_i, M_j)$.

Proof. If the block is inserted into (p_i^+, m_j^+), then the x-coordinate of the block will be determined by $LCS(P_i, M_j)$. It is obvious $LCS(P_i, M_j) = LCS(P_i, M_l), j < l < k$, since $L^p[m_l]_{\{j<l<k\}} > i$, i.e., $m_l, j < l < k$ are not in sequence P_i. Consequently, the theorem is concluded.

According to this theorem, it is obvious that the x-coordinate of a block b_i is determined only by the block set $S_{b_i} = \{b_j | L^p[b_j] < L^p[b_i]\}$, i.e., the blocks that are left to b_i in P. Therefore, we need not consider blocks that are right to the block b_i in P when scanning (P, M) to calculate the x-coordinate of CIPs (b_i^+, m_l^+), $1 \le l \le n$.

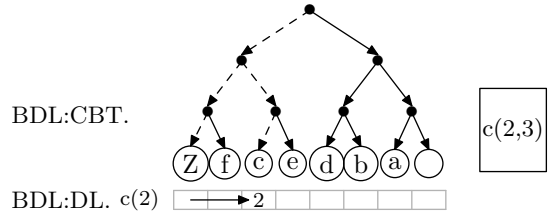
Theorem 2. *Assume an SP:* $(p_1, p_2, \cdots, p_i, \cdots, p_n;\ m_1, m_2, \cdots, m_j, \cdots, m_n)$ *where* $p_i = m_j$. *A block's x-coordinate is not less than* $LCS(P_i, M_j)$ *if we insert the block into the positions* (p_i^+, m_l^+), *where* $l \ge j$.

Proof. If a block is inserted into (p_i^+, m_j^+), then the x-coordinate of the block will be equal to $LCS(P_i, M_j)$. It is obvious that $LCS(P_i, M_j) \le LCS(P_i, M_l), j < l$. Consequently, the theorem is concluded.

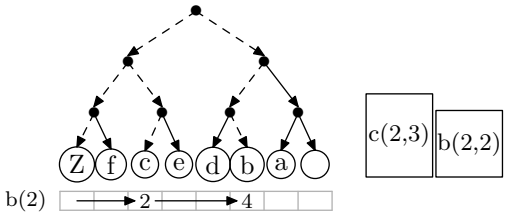
4.1.2 Enumerating x-coordinate

To enumerate insertion points in SP, we make use of the structure that consists of a complete binary tree(CBT) and a dynamical double list(DL), denoted as BDL structure, which is similar to the structure of the priority queue used in [13]. In this structure, the blocks are mapped to leaf nodes(except the most left one, which corresponds to the header of the dynamic list) of the binary tree from left to right in the order of sequence P. Therefore, the height of CBT is $log_2(n+1)$ if the number of blocks is n. If a block is kept in DL, all of the edges on the path from the root node to the corresponding leaf node are dashed. The header of the DL, whose value is zero, is always kept in the DL.

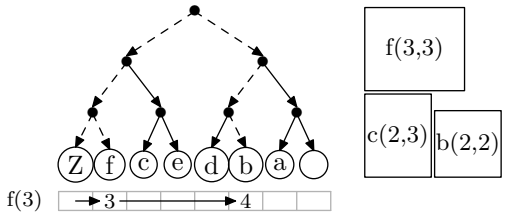
It takes $log(n+1)$ time to insert/delete a block into/from the DL with the help of CBT. In order to insert a block, starting from the corresponding leaf node, we go upward on the binary tree as far as a node with at least one dashed edge for child connection, and, then, go downward on the CBT along dashed edges (right dashed edges have priorities.) to a leaf node, whose value determines the x-coordinate of the inserted block. To update the tree structure, during the going upward stage, real line edges are dashed. When a block is deleted from the DL, we go upward on the CBT up to a node with two dashed edges. The edges visited, which are only on the path from the root to the deleted node, are changed to real lines.



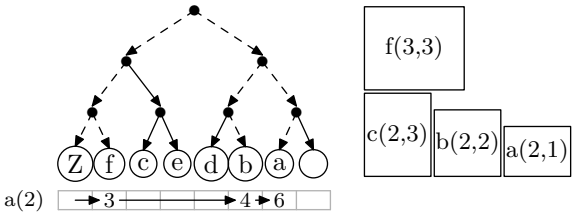
(a) The x-coordinates of CIPs (p, c^+). p is before c: 0. p is after c: 2



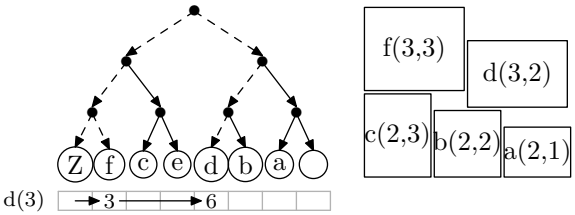
(b) The x-coordinates of CIPs (p, b^+). p is before c: 0, p is between b and c: 2 and p is after b: 4.



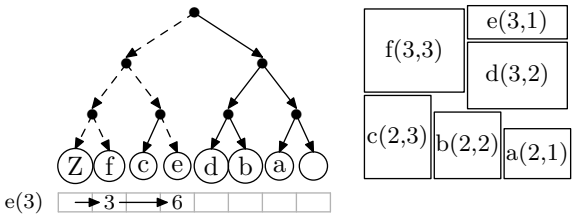
(c) The x-coordinates of CIPs (p, f^+). p is before f: 0, p is between f and b: 3 and p is after b: 4.



(d) The x-coordinates of CIPs (p, a^+). p is before f: 0, p is between f and b: 3, p is between b and a: 4, and p is after a: 6.



(e) The x-coordinates of CIPs (p, d^+). p is before f: 0, p is between f and d: 3 and p is after d: 6.



(f) The x-coordinates of CIPs (p, e^+). p is before f: 0, p is between f and e: 3 and p is after e: 6.

Figure 3: Enumeration of x-coordinates

Based on the DL, it is easy to enumerate x-coordinate of insertion points in the order of the sequence M. When visiting the i-th block m_i in M, we can know the x-coordinates of all the insertion points (p, m_i^+), where p is one position in sequence P. If p is between two consecutive elements, b_i and b_j, from DL, then the x-coordinate of (p, m_i^+) will be determined by b_i's x-coordinate. According to theorem 1, if we insert a block just after $m_i(m_i^+)$, the x-coordinate of the block has nothing to do with the blocks m_j, $j > i$. On the other hand, all or some of the blocks m_j,$j < i$, are stored in the DL in the order of the sequence P. The coordinates of the stored blocks are in ascending sort, because the blocks that cause a non-increasing order of the coordinates in the DL are deleted from the DL according to theorem 2.

Fig.3 shows an example. For convenience, we only show the forward list in DL. The sequence pair is $(P, M) = (fcedba, cbfade)$. For example, in Fig.3(b), we can easily compute the coordinates of f that is next to b in M by finding the block in DL that is before f and closest to f. The operation is achieved in $log(n+1)$ time through the complete binary tree. Here the x-coordinates of f is zero. The block f is inserted into DL. Because f's x-coordinate (3) is larger than its successor c's x-coordinate (2), and, hence, the block c is deleted from the DL. Fig.3(c) shows the updated DL and complete binary tree.

Fig.3(c) shows the status of DL after computing the x-coordinates of block f with width of 3. Based on the list DL, the x-coordinates of insertion points (p, f^+), where f^+ is the fixed position in sequence M and p is one position in P, can be enumerated as follows: The x-coordinates of the CIP (f^-, f^+) is 0, the x-coordinates of (f^+, f^+), (c^+, f^+), (e^+, f^+) and (d^+, f^+) is 3 and x-coordinates of (b^+, f^+) and (a^+, f^+) is 4. In Fig3, the x-coordinates of insertion points are shown in the caption of each sub-figure.

4.2 Enumerating insertion points

To enumerate the insertion points, we have to calculate the distance of each insertion point to four chip boundaries. Let S^r be the reverse of a sequence S. The sequence pairs (P^r, M),(P^r, M^r) and (P, M^r) are used, respectively, for calculating y-coordinate (Distance to the bottom), and x_r,y_r-coordinate (Distance to the right and the top) with the origin located at the chip's upper-right corner. The enumeration can be achieved by traversing blocks in the order of sequence M by using four dynamical lists.

Because DLs generated by (P^r, M^r) and (P, M^r) have to be visited in the reverse order of their occurrence, the DLs related to both these pairs of sequences are computed before the enumeration and are backuped. The backup can be achieved by storing the final list, the predecessor and successor of each block just after visiting the block. The DLs can be restored one by one from the final list.

Given an SP $(P, M) = (fcedba, cbfade)$, Fig.4 shows an example of the changing of DLs for enumerating insertion points. When visiting a block m_i in sequence M, we have four DLs. They are scanned simultaneously to enumerate insertion points with position m_i^+ in sequence M. For example, when we visit block c, four DLs are scanned simultaneously as shown in Fig.4(e), the insertion points (p,c^+), where p is between b^+ and c^-, have the same distance to all the boundaries. One of the 3 equivalent positions will be chosen randomly as a candidate insertion point of the block to be inserted.

Given the physical dimension of a block and an insertion point, it is easy to compute the chip width and chip height after inserting the block into the insertion point. We calculate the sum of the distances of the insertion point to the left boundary and the right boundary and the width of the inserted block. If the chip width is less than the sum, it will be updated to the sum. Otherwise, the chip width keeps unchanged. The chip height is calculated similarly. Therefore, during the enumerating of the insertion points, the formula (7) can be used to calculate the area cost of inserting the block into an insertion point.

4.3 Complexity Analysis

In the structure BDL, deleting from and adding to DL a node takes $O(log(n+1))$ time, where n is the number of blocks. The total number of deleting or adding operation is no more than $2n$. Therefore, the total complexity of updating DL should be $O(nlog(n+1))$. On the other hand, we have to traverse the DL to enumerate coordinates for each insertion point. In the worst case, the time complexity is $O(n^2)$. Backup and restoring of the DLs related to x_r,y_r-coordinate calculations take $O(n)$ time. Therefore, the time complexity of the enumeration is $O(n^2)$.

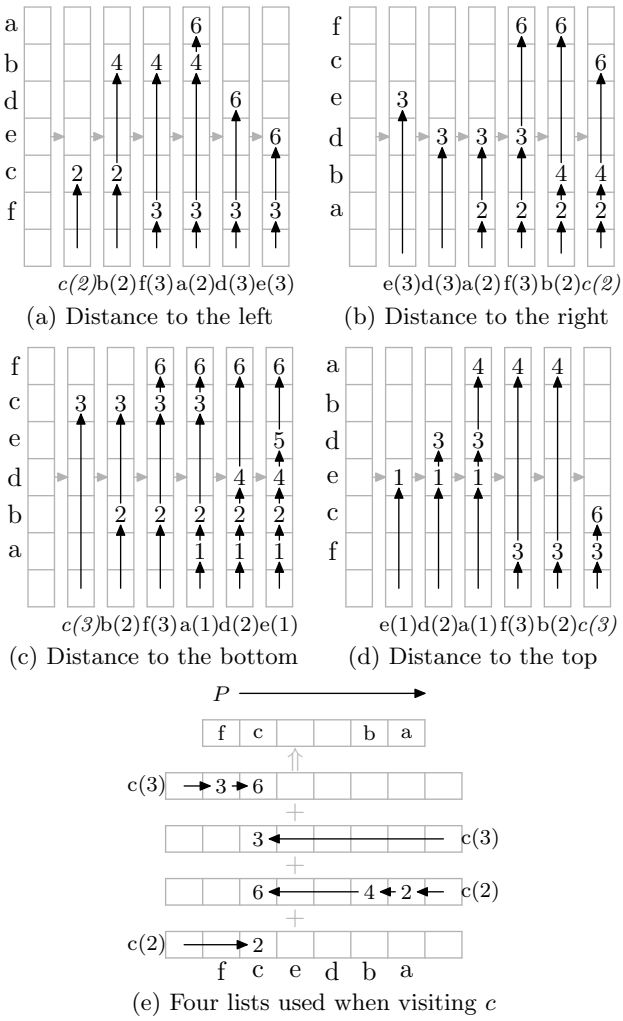


Figure 4: Enumeration of Insertion Points

5. EXPERIMENTAL RESULTS

The proposed method has been implemented in C-language on an IBM workstation (3.2GHz,3GB RAM) with Linux OS. The MCNC and GSRC benchmarks are used for the experiments. The wire-length is estimated using half-perimeter wire-length (HPWL).

5.1 Fixed-Outline Floorplanning

We compared the proposed fixed-outlined floorplanner with the latest version of Parquet[4][1] and B*-tree based fixed-outline floorplanner[5][6][7](NTU-FOFP) on the same platform. If the outline of a floorplan is smaller than H_0 and W_0, its aspect ratio can be different from the aspect ratio of the fixed outline. Otherwise, the floorplan fails to meet the outline constraint. We have tested Parquet 4.5 using SP and B*-tree, NTU-FOFP[6] and the proposed fixed-outline floorplanner(IARFP) under two situations: with/without HPWL optimization. The maximum percentage of white space is set to 10%.

First, without consideration of HPWL, we compared the success rate and runtime of the IARFP, Parquet 4.5[1] and NTU-FOFP. The expected aspect ratios of the floorplans are chosen from the range [1,3] with an interval of 0.5. The results were averaged over 50 runs for each aspect ratio. Table 1 lists the average success rate of Parquet using SP, Parquet using B*-tree, NTU-FOFP and IARFP. The success rate of Parquet using SP is 71.2% for n100, 57.6% for n200 and 50.8% for n300. NTU-FOFP obtained success rate 100% for n100 and n200, and 81.2% for n300. IARFP and Parquet using B*-tree all obtained 100% success rate. The runtime of IARFP is one tenth of those of other floorplanners, which shows the efficiecy of IAR operations proposed in this study. The total iteration (Fig.1) number in each run of test cases n100, n200 and n300 are respectively around 1500, 3000 and 4500.

Second, we compared the success rate, HPWL and runtime taking linear combination of formula (7) and HPWL as the objective function. Table 2 lists the experimental results of test cases n100 (with 885 nets and 334 I/O pins), n200 (with 1585 nets and 564 I/O pins) and n300 (with 1893 nets and 569 I/O pins). In the Parquet and NTU-FOFP, the I/O pads for all circuits are fixed at the given coordinates in the benchmark. For fair comparison, we deal with I/O pins in the same way. For n100, the expected ratios of the floorplans are chosen from the range [1,3] with an interval of 0.5 and the results were averaged on 50 runs for each aspect ratio. For n200 and n300, the results are the average of 10 runs. To run NTU-FOFP, we set '-alpha 0.9'(Other parameters default) to optimize wirelength. For Parquet(SP), the default parameters are used. It has been shown in [4] that Parquet using SP did better than Parquet using B*-tree when considering wire optimization. We also made experiments and found that Parquet using B*-tree took shorter time, but much lower success rate and larger HPWL. Consequently, we compare with Parquet using SP representation.

From the experiments on n100, It is observed that the IARFP reached 100% success rate for all aspect ratios, whereas NTU-FOFP and Parquet(SP) got respectively 45% and 34% on an average. IARFP respectively achieved 12% and 7% improvement in wire compared with Parquet(SP) and NTU-FOFP. Experiments on n200 and n300 showed that IARFP achieved 100% success rate and got 15% and 7% improvement in wire. The IARFP is also much faster than other floorplanners. The total iteration numbers in each run are respectively n100: 4700, n200: 7000 and n300: 9400.

All of the above results demonstrated the effectiveness and efficiency of the proposed fixed-outline floorplanning method.

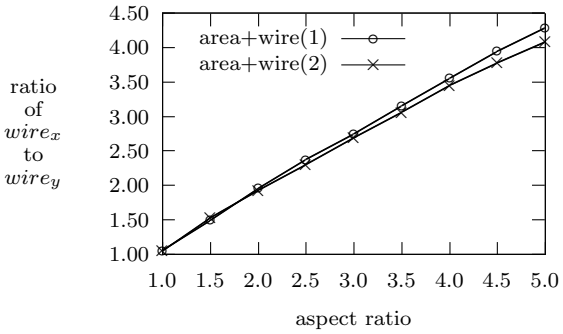


Figure 5: Aspect ratio vs. the ratio of $wire_x$ to $wire_y$.

5.2 Aspect ratio and wire-length

In this part, in order to get an accurate wire-length estimation, the I/O pins are scaled on the chip boundaries, which are different from the I/O pin positions specified in the test cases.

We devised three groups of experiments on test cases n100a and ami49 (They show the same regularity. This paper only lists the results of n100a because of page limit.) using three different objective functions.

1. Linear Combination of the formula (7) and wire-length. Formula (7) combines the excessive width and height with the maximum of width and height.
2. Linear combination of formula (8) and wire-length. Formula (8) is modified from (7) by normalizing the excessive width and height, width and height to those in the case that the aspect ratio is equal to 1.
3. formula (7)

In the figures and the following analysis, we use, respectively, series 1 (area+wire(1)), series 2 (area+wire(2)) and series 3 (area only(3)) to refer to the experimental results using the above three objective functions. In series 1 and 2, $\gamma = 10\%$. All the data points are based on 50 independent runs. The runtime are not shown since we use the same simulated annealing parameters for all experiments and the runtime of each experiment is close to that shown in Table 2.

Fig.5 shows the ratio of $wire_x$ to $wire_y$. As the aspect ratios grows, the ratio also increases. Fig.6 shows the geometric means of $wire_x$ and $wire_y$(section 3.2), which shows the change of the product of $wire_x$ and $wire_y$. In series 1 and 2, the products vary slightly(Fig.7 shows these values with smaller scale.) in various aspect ratios compared with the change of $wire_x$, $wire_y$ and the total wire length (summing $wire_x$ and $wire_y$). The sum of $wire_x$ and $wire_y$(HPWL) increases with the increment of aspect ratio. These are consistent with the simple geometric observation in section 3.2. Larger aspect ratio causes larger wire-length.

In Fig.5, we can see that the result of series 2 has a little smaller "slope". The ratios in series 2 are smaller and so is

Table 1: Results of Fixed-Outline Floorplanning.$\gamma = 10\%$ and all blocks are hard. The success rate is the ratio of the number of runs that failed to meet fixed-outline constraint to the number of runs. The aspect ratios are chosen from the range [1,3] with interval 0.5.

Cicuits	Parquet4.5(SP)[1]		Parquet4.5(BTree)[1]		NTU-FOFP[6][7]		IARFP(Ours)	
	#succ	time(s)	#succ	time(s)	#succ	time(s)	#succ	time(s)
n100	71.2%	5.95	100%	3.88	100%	2.08	100%	0.41
n200	57.6%	26.57	100%	18.50	100%	16.12	100%	1.82
n300	50.8%	66.53	100%	45.31	81.2%	97.30	100%	4.06
	0.60	15.75	1.00	10.76	0.94	18.36	1.00	1.00

Table 2: Results of Fixed-Outline Floorplanning. $\gamma = 10\%$ and all blocks are hard. All data of n100 are averaged over 50 independent runs and data of n200 and n300 are averaged over 10 independent runs. The success rate is the ratio of the number of runs failed to meet fixed-outline constraint to the total number of runs.

circuit	aspect ratios	Parquet 4.5(BTree)[1]			NTU-FOFP[6][7]			IARFP(Ours)		
		#succ	HPWL	time(s)	#succ	HPWL	time(s)	#succ	HPWL	time(s)
n100	1.0	64%	343496	20.56	100%	323053	37.44	100%	312400	7.07
	1.5	42%	337230	20.59	44 %	332040	36.94	100%	313305	7.34
	2.0	40%	346051	20.38	40 %	336690	35.62	100%	310485	7.48
	2.5	20%	353606	20.61	22 %	340408	36.70	100%	312947	7.70
	3.0	4%	372009	20.56	20 %	346936	37.45	100%	317486	7.95
		0.34	1.12	2.74	0.45	1.07	4.91	1.00	1.00	1.00
n200	1.0	40%	635682	90.52	10%	576977	158.49	100%	559099	19.50
n300	1.0	70%	761763	226.86	40%	720234	310.00	100%	652869	34.21
		0.55	1.15	5.90	0.25	1.07	8.72	1.00	1.00	1.00

the wire-length(Fig.6). The success rate in series 2, however, degraded greatly with the increase of aspect ratio(Fig.8). This is because the fixed-outline objectives is normalized (formula 8) in series 2 ,and, hence, in all the aspect ratio, the area cost will keep similar, whereas the wire-length will increase as the increasing of the aspect ratio. The wire-length gets more competitive as the increase of aspect ratio in series 2. Hence, we have to enlarge the weight of area cost for larger aspect ratios to conquer the outline constraint as formular 7 does. The formula (7) is much more effective than formula (8)under fixed-outline constraint when aspect ratios are far from one.

According to our counting, 99.4% of the experiments that failed to meet fixed-outline constraint (In series 2, totally 450 runs, 150 runs fail to meet fixed-outline.) violated the ***height*** constraint. In some sense, it also demonstrates that the wire-length keeps increasing as the aspect ratio increase. To get a smaller wire-length, the blocks have to be packed into an outline with a smaller aspect ratio, and thus the smaller one of height and width is violated more frequently.

5.3 Aspect ratio and success rate

Fig.9 shows the change of success rate on aspect ratio. In series 3, the objective is only area. The success rate seems not to decrease with increased aspect ratio. However, according to counting, 95% of the runs that failed to meet fixed-outline have violation in y-direction (of totally 450 runs, 21 failed to meet the outline constraint, 5% violated the width constraint) when $\gamma = 8\%$ and this percent is 84% when $\gamma = 6\%$(of totally 450 runs, 339 failed to meet the outline constraint and 42% violated the width constraint).

6. CONCLUSION

In this paper, we proposed a stable fixed-outline floorplanning method(IARFP). An elaborated method for perturbing solutions, Insertion after Remove(IAR), was devised for the simulated annealing. The IAR operation was devised based on the technique of enumerating block positions in Sequence Pair and accelerated the searching greatly. Moreover, based on the analysis of diverse objective functions used in the existing publications, we suggested a new objective function for the floorplanning with fixed-outline constraint. Compared with previous fixed-outline floorplanners, the proposed method is very effective and efficient. Experiments showed that the proposed fixed-outline floorplanner could achieve 100% success rate efficiently when optimizing area and wire-length simultaneously. Compared with Parquet 4.5 and B*-tree based fixed-outline flooprlanner, the proposed method got, respectively, improvement of 13% and 7% in wire-length, while achieving 100% success rate. On the other hand, we validate once more by experiments that aspect ratios close to one are good for wire-length.

7. ACKNOWLEDGMENTS

This work was supported in parts by funds from the Japanese Ministry of ECSST via Kitakyushu knowledge-based cluster projects.

8. REFERENCES

[1] S. Adya, H. H. Chan, and I. Markov. Parquet: Fixed-outline floorplanner. online:http://vlsicad.eecs.umich.edu/BK/parquet/.

[2] S. Adya and I. Markov. Fixed-outline floorplanning through better local search. In *Proc. ICCD*, pages 328–334, 2001.

[3] S. N. Adya and I. L. Markov. Fixed-outline floorplanning: Enaling hierarchical design. *Very Large Scale Integration (VLSI) systems, IEEE Trans. On.*, 11(6):1120–1135, 2003.

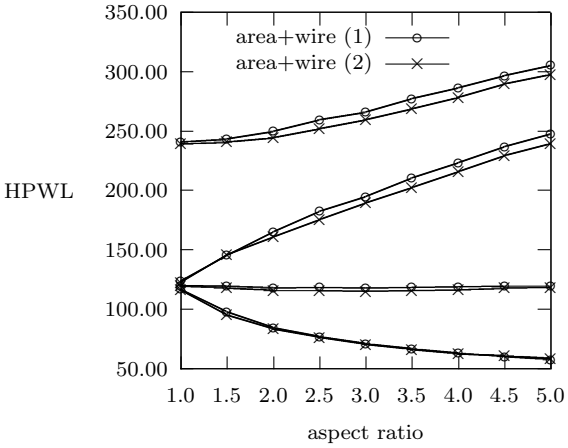


Figure 6: Aspect ratio(λ) and wire-length. Along the direction of wire HPWL increasing, the four pair of lines are,respectively, $wire_y$, geometric mean of $wire_x$ and $wire_y$, $wire_x$, total wire-length

.

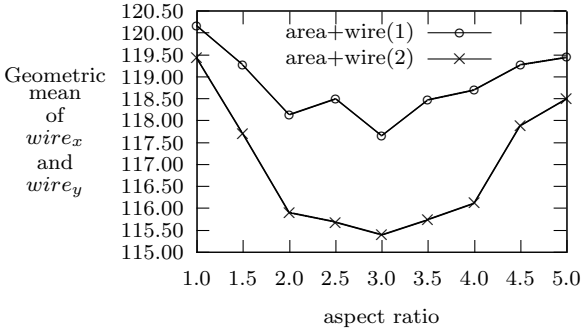


Figure 7: Aspect ratio(λ) vs. the geometric mean of $wire_x$ and $wire_y$.

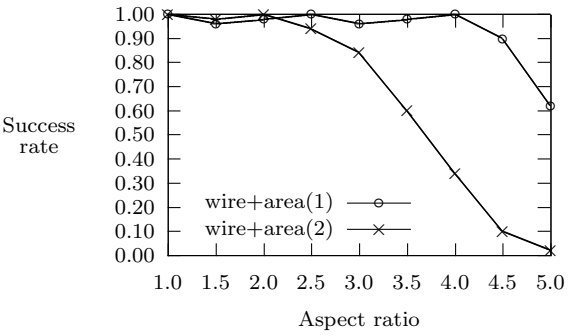


Figure 8: Success rate vs. aspect ratio(with HPWL optimization.)

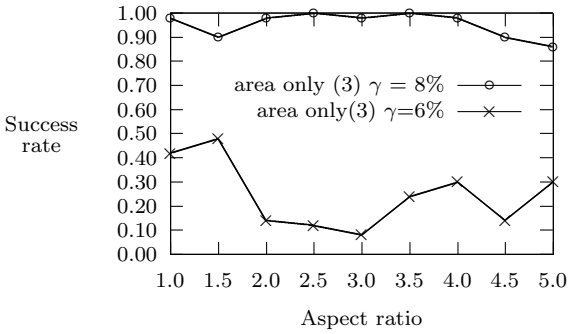


Figure 9: Success rate vs. aspect ratio.

[4] H. Chan, S. N. Adya, and I. L. markov. Are floorplan representations important in digital design. In *Proc. of ACM ISPD*, pages 129–136, 2005.

[5] T. Chen and Y. W. Chang. Modern floorplanning based on fast simulated annealing. In *Proc. ACM ISPD*, pages 104–112, 2005.

[6] T. Chen and Y. W. Chang. Modern floorplanning based on b*-tree and fast simulated annealing. *CAD of integrated circuits and systems IEEE Trans. on*, 25(4):637–650, April 2006.

[7] T. C. Chen and Y. W. Chang. B*-tree based floorplanner. Online:http://eda.ee.ntu.edu.tw/research.htm.

[8] P. Guo, C. Cheng, and T. Yoshimura. An o-tree representation of nonslicing floorplan and its applications. In *Proc. ACM/IEEE DAC*, pages 268–273, 1999.

[9] H.Murata, K.Fujiyoshi, S.Nakatake, and et al. VLSI module placement based on rectangle-packing by sequence-pair. *CAD of Integrated Circuits and Systems, IEEE Trans. on*, 15(12):1518–1524, 1996.

[10] A. Kahng. Classical floorplanning harmful. In *Proc. ACM ISPD*, pages 207–213, 2000.

[11] C. Lin, D. Chen, and Y. Wang. Robust fixed-outline floorplanning through evolutionary search. In *Proc. IEEE/ACM ASP-DAC*, pages 42–44, 2004.

[12] R. Liu, S. Dong, X. Hong, and Y. Kajitani. Fixed-outline floorplanning with constraints through instance augmentation. In *Proc. IEEE ISCAS*, pages 1883–1886, 2005.

[13] X. Tang, R. Tian, and D. F. Wong. Fast evaluation of sequence pair in block placement by longest common subsequence computation. *CAD of integrated circuits and systems,IEEE Trans. on*, 20(12):1406–1413, 2001.