Linear Programming-Based Cell Placement With Symmetry Constraints for Analog IC Layout

Shinichi Koda, Chikaaki Kodama, *Member, IEEE*, and Kunihiro Fujiyoshi, *Member, IEEE*

***Abstract*—In recent high-performance analog integrated circuit design, it is often required to place some cells symmetrically to a horizontal or vertical axis. Balasa *et al.* proposed a method of obtaining the closest placement that satisfies the given symmetry constraints and the topology constraints imposed by a sequence-pair (seq-pair). However, this method has the following defects: 1) Balasa's necessary condition for existence of the cell placement that satisfies the given constraints is incorrect; 2) some cells overlap; 3) the closest placement of satisfying both the symmetry and topology constraints is not always obtained; and 4) there is no explanation of placing cells symmetrically to plural axes. In this paper, we clarify the necessary and sufficient conditions for the existence of the cell placement that satisfies the given symmetry constraints and the topology constraints imposed by a seq-pair, and we propose an efficient method of obtaining, by linear programming, the closest cell placement that satisfies the given constraints. Here, a simple constraint graph is obtained from a seq-pair in order to derive a set of linear constraint expressions. Then, to shorten the running time of linear programming, the number of linear expressions is reduced by substituting the expressions for dependent variables, and the solution is obtained. The effectiveness of the proposed method was shown by computational experiments.**

***Index Terms*—Analog circuits, linear programming, placement, sequence-pair (seq-pair), symmetry constraints.**

I. Introduction

RECENT consumer telecommunication ICs and wireless communication devices, such as cell phone systems and wireless local area network systems, require high-performance analog ICs. In the design of such analog ICs, we take into account the balance of layout-induced parasitic devices to avoid both high-offset voltage and degradation of power supply rejection ratio [1], since analog circuits are very sensitive to parasitic disturbance, crosstalk, and power supply noise. Therefore, it is often required to place cells symmetrically, and this constraint is called "symmetry constraint" [2].

So far, the layout of analog ICs has been manually designed by experts. Recently, some automatic methods of symmetric placement were proposed, where rectangle packing representations and stochastic search methods, such as simulated annealing (SA), were used.

Balasa *et al.* proposed a symmetric placement method [2], [3] with a sequence-pair (seq-pair) which is a topological representation of rectangular block placement [4]. They presented a necessary condition for the existence of the cell placement that satisfies both the given symmetry constraints and the topology constraints imposed by a seq-pair [2]. They defined a seq-pair that satisfies the aforementioned conditions as "symmetric-feasible seq-pair" and proposed a method of obtaining nearly optimum solutions by searching only a symmetric-feasible seq-pair, using SA.

However, there is a case where any seq-pairs corresponding to a certain placement that satisfies given symmetry constraints are not symmetric-feasible. In addition, when a symmetric-feasible seq-pair is decoded to a placement by Balasa's method [2], [3], there are some cases where cells overlap or the closest packing that satisfies the symmetry and the topology constraints cannot be obtained. This induces enlargement of die size and increase of wire length; both of which are undesirable in the layout design. Furthermore, how to handle more than one set of cells with symmetry constraints is not clear in [2].

A method of solving 1-D compaction problem with symmetry constraints was proposed [5]. However, when a given problem size is big, computational time becomes vast. Okuda *et al.* proposed an improved method of solving this problem in shorter time [6]. In this method, initially, a "simple constraint graph" is made. The graph consists only of vertices corresponding to cells with symmetry constraints. These constraints are converted to linear constraint expressions to obtain the coordinates of cells by linear programming. As for cells without symmetry constraints, the coordinates can be obtained by the topological constraint graph, but redundant linear expressions may be derived. This will become a cause of redundant time to solve by linear programming.

In this paper, we clarify the defects in Balasa's method and present the necessary and sufficient conditions for the existence of the cell placement that satisfies the given symmetry constraints and the topology constraints imposed by a seq-pair. We define a seq-pair that satisfies the conditions as "symmetric-real-feasible seq-pair." Then, we propose an improved method of symmetric placement [2]. In the proposed method, we obtain a simple constraint graph [6] directly from a seq-pair in $O(sn \log\log n + es)$ time and derive a set of linear constraint expressions. Here, n, s, and e are the number of cells,

Manuscript received May 3, 2006; revised September 7, 2006. This paper was presented in part at the ISPD'06, San Jose, CA, April 9–12, 2006. This paper was recommended by Guest Editor P. H. Madden.

S. Koda was with the Department of Electrical and Electronic Engineering, Tokyo University of Agriculture and Technology, Koganei 184-8588, Japan. He is now with Nintendo Company, Ltd., Kyoto 601-8116, Japan (e-mail: kouda@fjlab.ei.tuat.ac.jp).

C. Kodama was with the Department of Electronic and Information Engineering, Tokyo University of Agriculture and Technology, Koganei 184-8588, Japan. He is now with Toshiba Microelectronics Corporation, Yokohama 247-8585, Japan (e-mail: kodamada@fjlab.ei.tuat.ac.jp).

K. Fujiyoshi is with the Department of Electrical and Electronic Engineering, Tokyo University of Agriculture and Technology, Koganei 184-8588, Japan (e-mail: fujiyosi@cc.tuat.ac.jp).

Digital Object Identifier 10.1109/TCAD.2007.891365

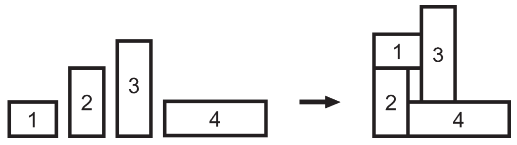

Fig. 1. Packing represented by seq-pair (1 2 3 4; 2 4 1 3).

symmetry constraints, and linear constraint expressions, respectively. Also, we propose a method of decreasing the number of both linear constraint expressions and variables by substituting the expressions for dependent variables. Thus, the closest placement that satisfies both the given symmetry constraints and the topology constraints imposed by a seq-pair can be obtained by solving linear constraint expressions, using linear programming. The effectiveness of the proposed method is confirmed by computational experiments.

II. Sequence-Pair and Symmetry Constraints

A. Sequence-Pair (seq-pair)

A seq-pair [4] is an ordered pair of Γ_+ and Γ_-, where each of Γ_+ and Γ_- is a permutation of names of given n blocks. For example, $(\Gamma_+; \Gamma_-) = (abcd; bdac)$ is a seq-pair of block set $\{a, b, c, d\}$. If block x is the ith block from the head of Γ_+, we denote $\Gamma_+^{-1}(x) = i$. Similar notation is used for Γ_-. To help intuitive understanding, we use a notation such as $(\Gamma_+; \Gamma_-) = (\cdot\cdot a \cdot\cdot b \cdot\cdot; \cdot\cdot a \cdot\cdot b \cdot\cdot)$ by which we mean $\Gamma_+^{-1}(a) < \Gamma_+^{-1}(b)$ and $\Gamma_-^{-1}(a) < \Gamma_-^{-1}(b)$. A seq-pair imposes a *"horizontal/ vertical (H/V) constraint"* on every block pair as follows: For every block pair $\{a, b\}$, a is to the left of b (equivalently, b is to the right of a) if $(\Gamma_+; \Gamma_-) = (\cdot\cdot a \cdot\cdot b \cdot\cdot; \cdot\cdot a \cdot\cdot b \cdot\cdot)$. Similarly, a is below b (equivalently, b is above a) if $(\Gamma_+; \Gamma_-) = (\cdot\cdot b \cdot\cdot a \cdot\cdot; \cdot\cdot a \cdot\cdot b \cdot\cdot)$. For example, seq-pair (1 2 3 4; 2 4 1 3) has relative position like Fig. 1.

H/V Constraint Graph: Based on the horizontal (left of) constraint imposed by a seq-pair, a directed and vertex-weighted graph $G_H(V, E)$ (V: vertex set, E: edge set) called *horizontal constraint graph* is constructed as follows [4].

1) V: source s, sink t, and vertices labeled block names.
2) E: (s, x) and (x, t) for each block x, and (x, x') if and only if x' is constrained to the right of x by the seq-pair. This can be realized by checking all pairs of elements on a seq-pair in $O(n^2)$ time, where n is the number of elements.
3) Vertex-weight: zero for s and t, width of blocks for the other vertices.

Similarly, the *vertical constraint graph* is constructed with the vertical (below) constraint and the height of each block. When the number of blocks is n, one of the optimal packings under the H/V constraint can be obtained in $O(n^2)$ time by applying the well-known longest path algorithm for directed acyclic graphs [4].

Another method of obtaining the optimal packings under the H/V constraint was proposed in [7]. In this method, a bottom left corner packing can be obtained from one seq-pair in $O(n \log \log n)$ time without making H/V constraint graphs.

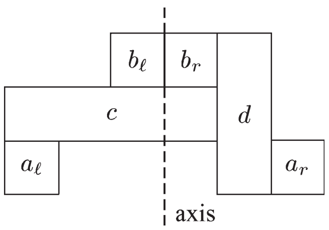

Fig. 2. Placement that satisfies symmetry constraints $\{(a_\ell, a_r), (b_\ell, b_r)\}$. Unique seq-pair $(b_\ell\, b_r\, c\, a_\ell\, d\, a_r; a_\ell\, c\, b_\ell\, b_r\, d\, a_r)$ corresponding to this placement is not symmetric-feasible.

B. Symmetry Constraints

A subset of cells is called a *symmetry group* if cells in the subset exhibit a form of symmetry and share a common symmetry axis. A *symmetry constraint* is represented by symmetry groups and gives the constraint that each pair of cells in the group must be placed symmetrically to a common vertical or horizontal axis called *symmetric axis*. The symmetry group may include *self-symmetric cells*, whose center must be placed on the symmetric axis.

In a pair of cells placed symmetrically to a vertical symmetric axis, one in the pair on the left (right) of the axis is called a *left cell* (*right cell*) and denoted as $a_\ell(a_r)$. A self-symmetric cell is denoted as a_s. A symmetry group is represented by a set of cell pairs in parentheses and self-symmetric cells like $\{(a_\ell, a_r), b_s\}$. We enumerate all symmetry groups and represent symmetry constraints.

C. Balasa's Cell Placement Method With Symmetry Constraints

1) Symmetric-Feasible Seq-Pair: If cells x and y in a symmetry group in seq-pair S satisfy

$$\Gamma_+^{-1}(x) < \Gamma_+^{-1}(y) \iff \Gamma_-^{-1}(sym(y)) < \Gamma_-^{-1}(sym(x)) \quad (1)$$

where $sym(x)$ and $sym(y)$ are the cells symmetric to cells x and y, respectively, the seq-pair S is said to be *symmetric-feasible* [2]. The aforementioned definition of symmetric-feasible seq-pair (1) is rewritten as follows when the axis of a symmetry group is vertical:

$$\Gamma_+^{-1}(x_\ell) < \Gamma_+^{-1}(y_\ell) \iff \Gamma_-^{-1}(y_r) < \Gamma_-^{-1}(x_r).$$

Balasa *et al.* mentioned in Lemma 1 of [2] that "Any placement configuration containing a symmetry group can be encoded with a symmetric-feasible sequence-pair." This is the necessary condition for any placement configuration to contain a symmetry group.

A seq-pair corresponding to the placement of Fig. 2, which satisfies symmetry constraints $\{(a_\ell, a_r), (b_\ell, b_r)\}$, is only $(b_\ell\, b_r\, c\, a_\ell\, d\, a_r; a_\ell\, c\, b_\ell\, b_r\, d\, a_r)$. Here, $\Gamma_+^{-1}(b_\ell) < \Gamma_+^{-1}(a_\ell)$ but $\Gamma_-^{-1}(a_r) \not< \Gamma_-^{-1}(b_r)$, so it is not symmetric-feasible. Therefore, the necessary condition of a symmetric-feasible seq-pair is not correct. Balasa and Lampaert [2] searched the optimum solution with SA only in symmetric-feasible seq-pair. However, if the optimum solution cannot be represented by any symmetric-feasible seq-pair, to obtain it by Balasa's method is impossible.

2) Algorithm of Determining x Coordinates of Cells: The algorithm of determining the x coordinate of each cell was proposed in [2]. It outputs x coordinates from the width of cells, symmetry constraints, and symmetric-feasible seq-pair.

First, cells are packed leftward based on a given seq-pair, and the x coordinate of each cell is obtained. Next, we focus on cells with symmetry constraints. Among x coordinates of the center of self-symmetric cells and of the center between each cell pair with symmetry constraints, the maximum value is determined as the x coordinate of the vertical symmetric axis. Then, right cells and self-symmetric cells are moved rightward one by one in order of Γ_+ of the input seq-pair if cells can be placed symmetrically. While each cell is moved rightward, the cells constrained to be on the right of it by the input seq-pair are also moved rightward by the same distance.

Finally, left cells are moved leftward one by one in reverse order of Γ_+ of the input seq-pair, and they are placed symmetrically. While each cell is moved leftward, the cells constrained to be on the left of it by the input seq-pair are also moved leftward by the same distance.

For example, when square-shaped cells of the same size $\{a_s, b_\ell, b_r, c_\ell, c_r\}$, symmetry constraints $\{a_s, (b_\ell, b_r), (c_\ell, c_r)\}$, and symmetric-feasible seq-pair $(b_\ell\, a_s\, c_\ell\, c_r\, b_r; b_\ell\, c_\ell\, c_r\, a_s\, b_r)$ are given, the x coordinate of the center between c_ℓ and c_r is determined as the x coordinate of the vertical symmetric axis after the cells were packed leftward based on the input seq-pair. The placement shown in Fig. 3(a) is obtained. Then, the self-symmetric cell a_s is moved rightward to place it symmetrically, but the cell b_r constrained to be on the right of a_s by the input seq-pair is also moved rightward by the same distance, as shown in Fig. 3(b). Since the cell b_ℓ is moved leftward to place it symmetrically, the placement shown in Fig. 3(c) is obtained.

However, Fig. 3(c) has larger width than the closest placement shown in Fig. 3(d). Therefore, the algorithm of determining x coordinates [2] has no guarantee to obtain the closest placement.

3) Algorithm of Determining y Coordinates of Cells: The algorithm of determining the y coordinate of each cell proposed in [2] outputs y coordinates from the height of cells, symmetry constraints, and a symmetric-feasible seq-pair.

These y coordinates are determined one by one in reverse order of Γ_+ of input seq-pair by moving either the left or the right cell in the pair in order to make the y coordinate of the other have the same value.

For example, when symmetry constraints $\{(a_\ell, a_r)\}$ and symmetric-feasible seq-pair $(a_\ell\, b\, c\, a_r; b\, a_\ell\, a_r\, c)$ are inputted in the algorithm, the y coordinate of a_r is determined first, and that of a_ℓ is determined accordingly. Then, cell c is placed above a_r, as shown in Fig. 4(a). After this, the y coordinate of b is determined, and a_ℓ moves above b, as shown in Fig. 4(b). Then, the y coordinate of a_r is determined to have the same coordinate as a_ℓ but a_r and c overlap, as shown in Fig. 4(c).

Therefore, in spite of the existence of a placement that satisfies all constraints as shown in Fig. 4(d), the algorithm [2] obtains different placements, where the topology constraints are not satisfied or cells overlap.

4) Handling of Plural Symmetry Groups: It was described in [2] that the algorithms of determining x and y coordinates can be easily extended to plural symmetry groups, but how to extend the algorithms was not mentioned.

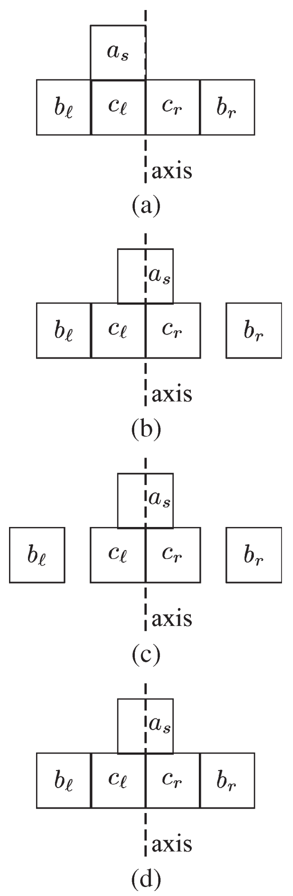

Fig. 3. Decoding process and resultant placement when symmetry constraints $\{a_s, (b_\ell, b_r), (c_\ell, c_r)\}$ and symmetric-feasible seq-pair $(b_\ell\, a_s\, c_\ell\, c_r\, b_r; b_\ell\, c_\ell\, c_r\, a_s\, b_r)$ are input in the algorithm of determining x coordinates [2]. Note that y coordinates were already calculated and satisfy the above constraints. (a) Cells are packed leftward based on a given seq-pair to determine the x coordinate of a symmetric axis. (b) a_s is placed symmetrically to the axis, and b_r moves according to the move of a_s. (c) b_ℓ moves symmetrically to b_r (the output of the algorithm of determining x coordinates). (d) Closest placement satisfies all constraints.

For example, when we try to place the cells based on symmetry constraints $\{(a_{\ell 1}, a_{r1})\}$ and $\{(b_{\ell 2}, b_{r2})\}$ and seq-pair $(a_{\ell 1}\, b_{\ell 2}\, b_{r2}\, a_{r1}; b_{\ell 2}\, a_{\ell 1}\, a_{r1}\, b_{r2})$, the seq-pair is symmetric-feasible, but the placement that satisfies all constraints never exists since cell $b_{\ell 2}$ is constrained to be below $a_{\ell 1}$, and cell b_{r2} is constrained to be above a_{r1}. Balasa and Lampaert [2] did not mention the unrealizable seq-pair like this. When a seq-pair with the given symmetry constraints is unrealizable, that should be output by the algorithm.

D. Method of Solving Compaction Problem With Symmetry Constraints

A method of solving 1-D compaction problem with symmetry constraints by linear programming was proposed [5]. If a given problem size is big, the computational time becomes vast.

Okuda *et al.* took notice of the fact that the number of symmetry constraints is smaller than the number of other constraints. They proposed an improved method of solving the

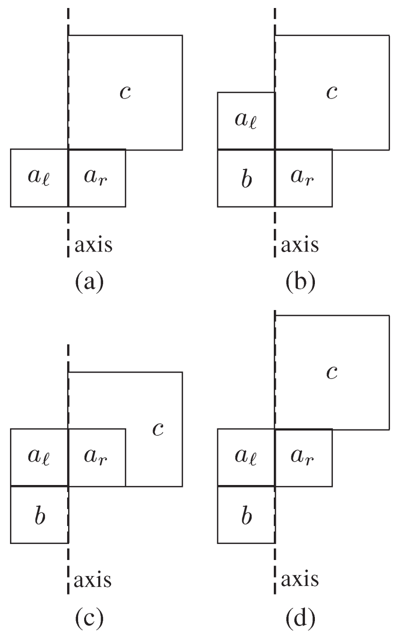

Fig. 4. Decoding process and resultant placement when symmetry constraint $\{(a_\ell, a_r)\}$ and symmetric-feasible seq-pair $(a_\ell\, b\, c\, a_r;\, b\, a_\ell\, a_r\, c)$ are input in the algorithm of determining y coordinates [2]. Note that x coordinates were already calculated and satisfy the above constraints. (a) y coordinate of a_r is determined first, and that of a_ℓ is determined to have the same coordinate as a_r. Then, c is placed above a_r. (b) y coordinate of b is determined, and a_ℓ is placed above b. (c) Cells overlap each other after changing the y coordinate of a_r to be the same as a_ℓ. (d) Placement to satisfy all constraints.

problem speedily [6]. In the whole problem, linear programming is applied only to a part of symmetry constraints, and graph algorithms are applied to the rest. The sketch of the improved method is as follows.

When symmetry constraints for a vertical symmetry axis and a horizontal constraint graph are given, from the constraint graph, pick up source, sink, and vertices corresponding to cells with symmetry constraints. Assume vertices a and b are picked up. If there is a path from a to b, set the directed edge (a, b) with the weight of the longest path value from a to b. By applying this operation to vertices with symmetry constraints, obtain a directed graph called *simple constraint graph*. A compaction problem represented by a simple constraint graph is called *core problem*. The core problem is equal to the original compaction problem.

The core problem can be obtained from a given horizontal constraint graph in $O(snm)$ time [6], since the longest path value from one vertex can be obtained in $O(nm)$ time by applying Ford's shortest path algorithm.[1] Here, s, n, and m are the number of given symmetry constraints, cells, and edges, respectively, in a given constraint graph.

Linear programming is used only for the core problem, and the longest path algorithm is applied to the rest, so the computational time relating to linear programming is cut down drastically, and the coordinates of all cells are obtained in shorter time.

[1]The given horizontal constraint graph may contain cycles.

III. Proposed Method of Cell Placement With Symmetry Constraints

As mentioned in Section II-C, the method proposed in [2] has the following defects.

1) Some cells overlap.
2) The closest cell placement that satisfies the symmetry and topology constraints is not always obtained.
3) How to handle more than one symmetry group is not clear.
4) There exists a placement that is impossible to represent by any symmetric-feasible seq-pair.

We expect that the cell placement with symmetry constraints can be solved efficiently and quickly if Okuda's approach to 1-D compaction problem with symmetry constraints [6] is applied together with a seq-pair. So, in this paper, we propose an improved method of obtaining linear constraint expressions from the given symmetry constraints and a seq-pair and solve them by linear programming. The proposed method does not cause the aforementioned defects 1), 2), and 3).

As for 4), we present the necessary and sufficient conditions for the existence of the cell placement that satisfies the given symmetry constraints and the topology constraints imposed by a seq-pair. We define a seq-pair that satisfies the conditions as "symmetric-real-feasible seq-pair" in Section III-A.

A. Symmetric-Real-Feasible Seq-Pair

Definition 1: For one symmetry group to the vertical symmetry axis, a seq-pair to satisfy the following two expressions is defined as *symmetric-real-feasible* seq-pair:

$$\left.\begin{matrix}\Gamma_+^{-1}(x) < \Gamma_+^{-1}(y)\\ \text{and}\\ \Gamma_-^{-1}(x) < \Gamma_-^{-1}(y)\end{matrix}\right) \Rightarrow \left(\begin{matrix}\Gamma_+^{-1}\left(sym(x)\right) \not< \Gamma_+^{-1}\left(sym(y)\right)\\ \text{or}\\ \Gamma_-^{-1}\left(sym(x)\right) \not< \Gamma_-^{-1}\left(sym(y)\right)\end{matrix}\right. \quad (2)$$

$$\left.\begin{matrix}\Gamma_+^{-1}(x) < \Gamma_+^{-1}(y)\\ \text{and}\\ \Gamma_-^{-1}(x) > \Gamma_-^{-1}(y)\end{matrix}\right) \Rightarrow \left(\begin{matrix}\Gamma_+^{-1}\left(sym(x)\right) \not> \Gamma_+^{-1}\left(sym(y)\right)\\ \text{or}\\ \Gamma_-^{-1}\left(sym(x)\right) \not< \Gamma_-^{-1}\left(sym(y)\right)\end{matrix}\right. . \quad (3)$$

Expression (2) represents that when left cells are in horizontal relative positions, right cells do not have the same relative positions. Expression (3) represents that when left cells are in vertical relative positions, right cells do not have the reverse relative positions.

Similarly, we can define symmetric-real-feasible seq-pair for one symmetry group to the horizontal symmetry axis. ■

Then, the following theorem is obtained.

Theorem 1: There is a placement that satisfies both H/V constraints of a given seq-pair and a symmetry constraint of one symmetry group if and only if the seq-pair is symmetric-real-feasible.

Proof: When there is a placement that satisfies both H/V constraint of a given seq-pair and a symmetry constraint of one symmetry group, it is obvious that the seq-pair is symmetric-real-feasible. Hence, in the following, we show constructively that there is a placement that satisfies both H/V constraint of symmetric-real-feasible seq-pair and a symmetry constraint of only one symmetry group to a vertical axis.

First, the vertical constraint graph is obtained from the seq-pair by the method mentioned in Section II-A. Then, the constraint graph G_V is obtained by merging vertices, each corresponding to a left cell and a right cell of a cell pair. If directed cycles do not exist in G_V, we obtain the longest path value of each vertex and determine the value as the y coordinate of the lower edge of the corresponding cell. Then, it is clear that the y coordinates satisfy both a symmetry constraint for vertical direction and the vertical constraint of a given seq-pair. However, the placement obtained is not always closest for the y direction.

If there is a directed edge from left cell a_ℓ to b_ℓ in the horizontal constraint graph obtained from the seq-pair, a directed edge is set from right cell b_r to a_r. Also, if there is a directed edge from right cell a_r to b_r, a directed edge is set from left cell b_ℓ to a_ℓ, and graph G_H is obtained. (Note that this operation of setting edges was also carried out in [11].)

If there are no directed cycles in G_H, by utilizing the graph with a slightly complicated procedure, we can obtain a placement that satisfies both a symmetry constraint for the x direction and the horizontal constraint of the seq-pair.

Accordingly, we can obtain a placement that satisfies both a symmetry constraint and H/V constraints of the seq-pair if there are no directed cycles in both G_V and G_H. Here, as shown in the following lemma, if there are vertex disjoint directed cycles in G_V and G_H, it is easily understood that the seq-pair is not symmetric-real-feasible since the minimum cycles have two edges. ■

Lemma 1: Assume that there are two seq-pairs S_1 and S_2, which have the same elements. We obtain vertical constraint graphs $G_{v1}(V_v, E_{v1})$ and $G_{v2}(V_v, E_{v2})$ from S_1 and S_2, respectively. Then, $G_v(V_v, E_{v1} \cup E_{v2})$ is obtained by merging directed edges of both $G_{v1}(V_v, E_{v1})$ and $G_{v2}(V_v, E_{v2})$. Also, we obtain horizontal constraint graphs $G_{h1}(V_h, E_{h1})$ and $G_{h2}(V_h, E_{h2})$ from S_1 and S_2, respectively. After reversing the direction of edges on $G_{h2}(V_h, E_{h2})$, $G_h(V_h, E_{h1} \cup \overline{E_{h2}})$ is obtained by merging directed edges of both $G_{h1}(V_h, E_{h1})$ and $G_{h2}(V_h, E_{h2})$. If there are vertex disjoint directed cycles in G_V and G_H, the minimum one has two edges.

Proof: All of G_{v1}, G_{v2}, G_{h1}, and G_{h2} are transitive closure since these constraint graphs are obtained from a seq-pair, so it is clear that there is no self-loop on both G_v and G_h. Therefore, assuming that directed cycles with more than two edges exist on either G_v or G_h, we can prove the above proposition by contradiction using the fact that these four graphs are transitive closure. ■

B. Linear Constraint Expressions Obtained From Symmetry Constraints and Given Seq-Pair

We discuss how to obtain the linear constraint expressions from H/V constraints of a given seq-pair. If the horizontal constraints of a seq-pair that "a is to the left of b" is given, the following inequality for the x direction can be derived [8]:

$$x(a) + w(a) \le x(b).$$

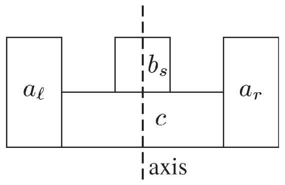

Fig. 5. Placement that satisfies symmetry constraints $\{(a_\ell, a_r), b_s\}$ and seq-pair $(a_\ell\, b_s\, c\, a_r; a_\ell\, c\, b_s\, a_r)$.

Here, $x(a)$ and $x(b)$ are x coordinates of the left edge of cells a and b, respectively, and $w(a)$ is the width of a.

We convert symmetry constraints into linear constraint expressions. When a cell pair a_ℓ and a_r is symmetrical to a vertical symmetric axis, the linear constraint expression for the x direction is

$$\text{Axis}_x - (x(a_\ell) + w(a_\ell)) = x(a_r) - \text{Axis}_x$$

where Axis_x is the x coordinate of a vertical symmetric axis of a symmetry group. The linear constraint expression for the y direction is

$$y(a_\ell) = y(a_r).$$

For self-symmetric cell a_s, the linear constraint expression for the x direction is

$$\text{Axis}_x - (x(a_s) + w(a_s)) = x(a_s) - \text{Axis}_x$$

and that for the y direction cannot be obtained.

C. Set of Linear Constraint Expressions Obtained From a Simple Constraint Graph

In order to obtain the set of linear constraint expressions from a simple constraint graph, we can easily convert H/V constraint graphs to simple constraint graphs by the method proposed in [6]. Assume the number of cells is n and that of symmetry constraints is s. It takes $O(n^2)$ time to obtain H/V constraint graphs from a given seq-pair, and the longest path value can be calculated in $O(n^2)$ time by the well-known longest path algorithm mentioned in Section II-A. Therefore, at least $O(sn^2)$ time is required to obtain a simple constraint graph.

However, if we utilize the method of obtaining the lower left corner packing from a seq-pair in $O(n \log\log n)$ time without making H/V constraint graphs [7], we can quickly obtain a simple constraint graph. So, we use this method [7] to obtain all the longest path values in $O(sn \log\log n)$ time.

In a simple constraint graph, there are both necessary and unnecessary transitive edges. For example, when we obtain a simple constraint graph from seq-pair $(a_\ell\, b_s\, c\, a_r; a_\ell\, c\, b_s\, a_r)$ and if $w(c) > w(b_s)$, not only edges (a_ℓ, b_s) and (b_s, a_r) but also transitive edge (a_ℓ, a_r) is required to avoid overlap of c and a_r. The placement corresponding to the seq-pair is shown in Fig. 5.

We will remove unnecessary transitive edges. Let one vertex pair be v_i and v_j among all vertex pairs of source, sink, and vertices corresponding to cells with symmetry constraints.

```
Algorithm SP–Core
    Input: Seq-pair ($\Gamma_+;\Gamma_-$), symmetry constraints;
    Output: Linear constraint expressions;
{ variable $len(i,j)$; /* the current longest path value
    from vertex $i$ to vertex $j$ */
  Insert "$source$" in the head of both $\Gamma_+$ and $\Gamma_-$;
  Append "$sink$" to the tail of both $\Gamma_+$ and $\Gamma_-$;
  Assume the constraint graph obtained
  from ($\Gamma_+;\Gamma_-$) is $G$;
  /* Note that the graph is not made in fact. */
  $len(i,j)$ is initialized by the negative values;
  for ($i$ =each cell with symmetry constraints
  in reverse order of $\Gamma_-$) {
    Calculate the longest path value $\ell(i,x)$ from $i$
    to all vertices $x$;
    for ($j$ =each cell with symmetry constraints
    in the right of $i$ on $\Gamma_-$)
      if (There is a path from $i$ to $j$ on $G$)
        if ($\ell(i,j) > len(i,j)$){
          $len(i,j) = \ell(i,j)$;
          Output the linear constraint expression:
          $x(i)+len(i,j) \leq x(j)$;
          for ($k$ =each cell with symmetry
          constraints in the right of $j$ on $\Gamma_-$)
            if (There is a path from $i$ to $j$ on $G$
            and $\ell(i,j)+len(j,k)>len(i,k)$)
              $len(i,k) = len(i,j) + len(j,k)$;
        }
  }
}
```

Fig. 6. Algorithm SP-Core.

In the method proposed in [6], the longest path from v_i to v_j is obtained if it exists. If there is a vertex corresponding to a cell with symmetry constraints on one of the longest paths from v_i to v_j, the method judges the edge (v_i, v_j) unnecessary. Since judgment on the necessity of the longest path requires linear time of the number of topology constraints (assume m) obtained from a seq-pair, the time complexity is $O(s^2m)$ in total [6]. If this method is applied to H/V constraint graphs of a seq-pair, it takes $O(s^2n^2)$ time because the number of topology constraints obtained from a seq-pair is $O(n^2)$.

By utilizing the method of obtaining the lower left corner packing [7], all the longest paths and their values between two vertices among source, sink, and vertices with symmetry constraints are obtained and recorded. If each recorded longest path value is equal to the longest path value via another vertex, the recorded longest path can be considered unnecessary. This method is carried out in $O(s^3)$ time. Note that $s \leq n$.

Additionally, whether transitive edges are necessary or not is judged in $O(se)$ time in the proposed method. Here, e is the number of linear constraint expressions obtained by the proposed algorithm SP-Core (Fig. 6). SP-Core calculates the longest path values and outputs linear constraint expressions if necessary. The time complexity is $O(sn \log\log n + se)$.

For example, when seq-pair $(a_\ell\, b_s\, c\, a_r; a_\ell\, c\, b_s\, a_r)$, symmetry constraints $\{(a_\ell, a_r), b_s\}$ to a vertical symmetric axis, and the width of cells $w(a_\ell) = 1$, $w(a_r) = 1$, $w(b_s) = 1$, and $w(c) = 3$

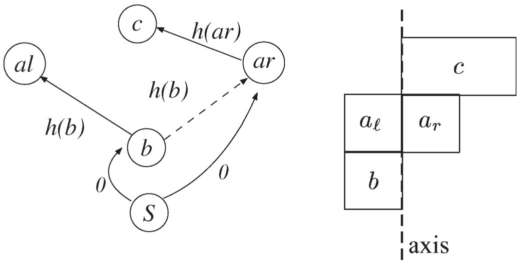

Fig. 7. Vertical constraint graph obtained from symmetry constraint $\{(a_\ell, a_r)\}$ and seq-pair $(a_\ell\, b\, c\, a_r; b\, a_\ell\, a_r\, c)$. $h(a)$ is the height of cell a, and S is the source.

are given (shown in Fig. 5), SP-Core outputs the following expressions for the x direction:

$$0 \leq x(a_\ell)$$
$$x(a_\ell) + 1 \leq x(b_s)$$
$$x(a_\ell) + 4 \leq x(a_r)$$
$$x(b_s) + 1 \leq x(a_r)$$
$$x(a_r) + 1 \leq x(\text{sink}).$$

Note that $x(\text{source}) = 0$.

The following expressions are also obtained by converting the given symmetry constraints:

$$\text{Axis}_x - x(a_\ell) = (x(a_r) + 1) - \text{Axis}_x$$
$$\text{Axis}_x - x(b_s) = (x(b_s) + 1) - \text{Axis}_x.$$

We can obtain the width of cell placement by solving these expressions for the x direction. The height can be obtained similarly but independently from the x direction by solving expressions for the y direction. Then, the area of placement is obtained from the width and the height.

D. Removal of Dependent Variables by Substitution of Expressions

Linear constraint expressions obtained from symmetry constraints are in equality, which means the distances from one cell in the pair to the symmetry axis and from the other cell to the axis are equal. Here, we focus on the variable corresponding to the right cell or self-symmetric cell and transpose it to the left side of the expression and the others to the right side of it. Then, we can decrease the number of variables and expressions by substitution.

For example, we focus on the variables $x(a_r)$ and $x(b_s)$ in the following expressions shown in Section III-C:

$$\text{Axis}_x - x(a_r) = (x(a_\ell) + 1) - \text{Axis}_x$$
$$\text{Axis}_x - x(b_s) = (x(b_s) + 1) - \text{Axis}_x.$$

The following expressions are obtained by transposing $x(a_r)$ and $x(b_s)$ to the left side of the expressions:

$$x(a_r) = 2 * \text{Axis}_x - (x(a_\ell) + 1)$$
$$x(b_s) = \text{Axis}_x - 0.5.$$

TABLE I
COMPARISON OF RUNNING TIME OF OBTAINING THE SOLUTION BY THE PROPOSED METHODS WITH SA ON PENTIUM IV 2.4 GHz. METHOD 1: A METHOD OF USING THE LINEAR CONSTRAINT EXPRESSIONS OF A SIMPLE CONSTRAINT GRAPH. METHOD 2: METHOD 1 WITH DEPENDENT VARIABLES REMOVAL BY SUBSTITUTING EXPRESSIONS. METHOD 3: METHOD 2 WITH UTILIZATION OF A VERTICAL CONSTRAINT GRAPH TO OBTAIN y COORDINATES

#Cell			#Symmetry	Time [sec]		
All	Pairs	Self	groups	Method 1	Method 2	Method 3
8	2	0	1	83.1	66.8	26.1
10	4	0	1	358.3	256.4	69.8
9	3	1	2	167.7	133.6	46.2
9	1	1	1	47.8	39.1	19.9
16	4	2	2	581.2	397.9	113.0

TABLE II
EXPERIMENTAL COMPARISONS BETWEEN THE RESULTS OF THE PROPOSED METHOD (PENTIUM IV 3.2 GHz), BALASA'S RESULTS (SUN BLADE 100), AND NONCONSTRAINED RESULTS (PENTIUM IV 3.2 GHz)

Design	#Cell	#Symmetry groups	Balasa's results [10]		Proposed method		No symmetry	
			Time [sec]	Area [%]	Time (LP part)[sec]	Area [%]	Time [sec]	Area [%]
biasynth_2p4g	65	8+12+5	780.00	115.00	205.86 (127.73)	105.65	43.98	103.86
lnamixbias_2p4g	110	16+6+6+12+4	2823.60	109.36	3026.91 (2144.98)	107.95	109.52	105.59

We substitute the aforementioned expressions for the other expressions shown in Section III-C and obtain

$$0 \le x(a_\ell)$$

$$x(a_\ell) - \text{Axis}_x \le -1.5 \quad (4)$$

$$x(a_\ell) - \text{Axis}_x \le -2.5 \quad (5)$$

$$x(a_\ell) - \text{Axis}_x \le -1.5 \quad (6)$$

$$2 * \text{Axis}_x - (x(a_\ell) + 1) + 1 \le x(\text{sink})$$

where two variables and two expressions are removed.

Furthermore, (4) and (6) are redundant since there is (5) in the aforementioned constraint expressions. We can decrease constraint expressions by removing the redundant expressions like this.

Finally, we remove redundant expressions and obtain

$$0 \le x(a_\ell)$$

$$x(a_\ell) - \text{Axis}_x \le -2.5$$

$$2 * \text{Axis}_x - (x(a_\ell) + 1) + 1 \le x(\text{sink})$$

where two expressions are removed, and we obtained the set of constraint expressions with three variables and three expressions.

E. Calculation of y Coordinates With Vertical Constraint Graph

If only symmetry groups to the vertical symmetric axes are given, we can quickly obtain y coordinates by utilizing the vertical constraint graph obtained from a given seq-pair as mentioned in Section II-A.

First, we obtain a vertical constraint graph. If a directed weighted edge from vertex x to the vertex corresponding to the left (right) cell with symmetry constraint exists, we append a directed edge with the same weight from vertex x to the vertex corresponding to the right (left) cell. y coordinates of each cell is obtained in $O(n^2)$ time by calculating the longest path value from the source to each vertex [9] using the well-known longest path algorithm for directed acyclic graphs.

In Fig. 7, the vertical constraint graph obtained from seq-pair $(a_\ell\, b\, c\, a_r; b\, a_\ell\, a_r\, c)$ is drawn in directed solid edges. Since symmetry constraint $\{(a_\ell, a_r)\}$ is given, we append edge (b, a_r) with weight $h(b)$ (drawn in directed broken edge) since edge (b, a_ℓ) with the same weight $h(b)$ exists.

IV. EXPERIMENTAL RESULTS

In order to confirm the improvement of running speed, we implemented the proposed method with SA in C language. The MOVE operation to make an adjacent solution for SA is as follows. Choose two elements randomly from a given seq-pair $(\Gamma_+; \Gamma_-)$ and exchange each other in both Γ_+ and Γ_- or in either of them. When an obtained adjacent solution is symmetric-real-infeasible, it is abandoned, and another one is obtained from the present solution.

Initial temperature and final temperature in the annealing schedule are determined properly by the preliminary run. The initial temperature is determined at the accepted ratio of more than 70% in deteriorated solutions. When the evaluation of solutions does not get worse nor better and seems to be settled, the temperature is determined as the final one. The ratio of falling temperature is determined in view of the runtime of experiments.

The following three methods are implemented to compare the running time:

Method 1:	A method of using the linear constraint expressions of a simple constraint graph.
Method 2:	Method 1 with removal of dependent variables by substitution of expressions.
Method 3:	Method 2 with utilization of the vertical constraint graph to obtain y coordinates.

We use the simplex method for linear programming problem and five kinds of cell sets, each with the different number of cells and symmetry constraints. The symmetric axis in every symmetry group is vertical only.

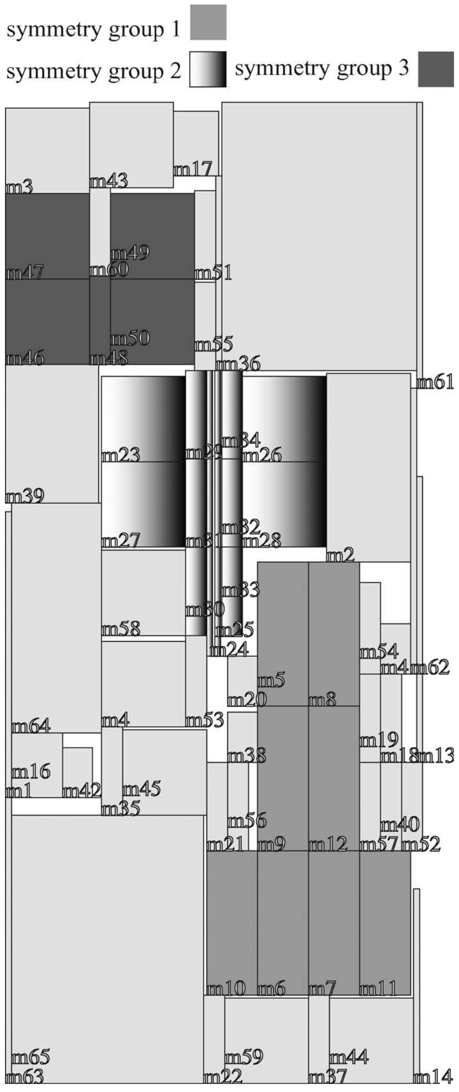

Fig. 8. Result of “biasynth_2p4g” obtained by the proposed method with SA (time: 205.86 s, packing ratio: 105.65%).

Table I shows the comparison of running time of obtaining the solution by three methods on the same annealing schedule.

Note that every method obtains the same solution but each requires different time. The speed-up by substitution and utilization of the constraint graph to obtain y coordinates is confirmed from Table I.

In order to obtain nearly optimum solution, we carried out experiments on Pentium IV 3.2 GHz by Method 3 with SA. The MOVE operation is the same as mentioned previously, and the annealing schedule is determined as above too. Two input data sets are extracted from [10] for experiments. One is named “biasynth_2p4g” and consists of 65 cells with three symmetry groups (four pairs, six pairs, and two pairs with one self-symmetric cell) extracted from [10, Fig. 9]. The other is named “lnamixbias_2p4g” and consists of 110 cells with five symmetry groups (eight pairs, three pairs, three pairs, six pairs, and two pairs) extracted from [10, Fig. 10]. Just for reference, we also carried out experiments on the aforementioned two data sets without symmetry constraints.

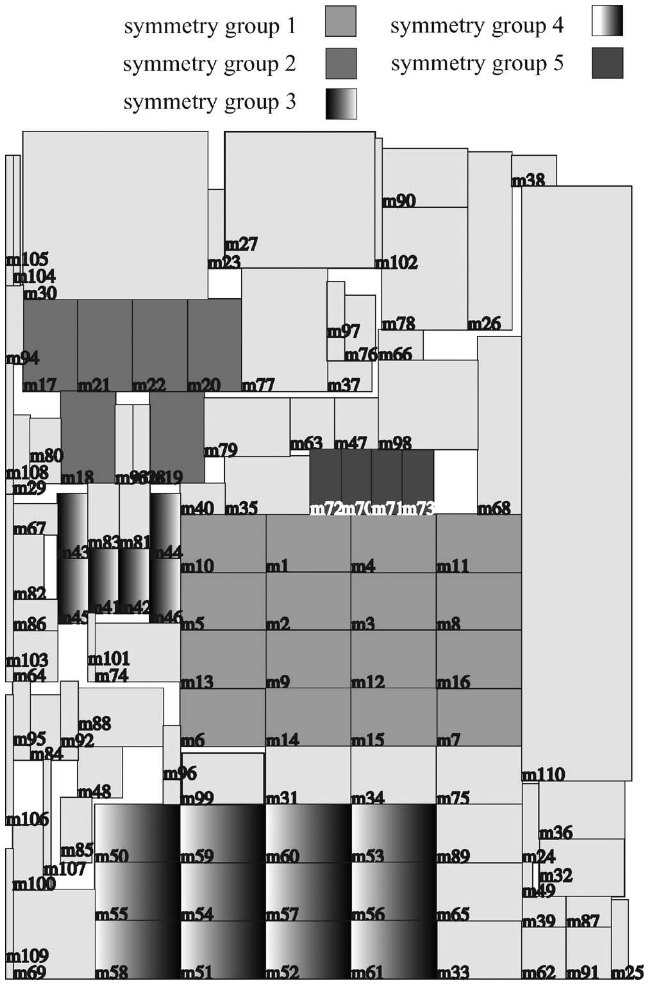

Fig. 9. Result of “lnamixbias_2p4g” obtained by the proposed method with SA (time: 3026.91 s, packing ratio: 107.95%).

Table II shows the experimental comparisons between the proposed method (Method 3), the method in [2], and the no-symmetry constraints. “Area [%]” is obtained by dividing the area of bounding box by the total area of all cells. Note that each “area [%]” in the table is manually calculated based on [10, Figs. 9 and 10]. The results in [10] were obtained by Sun Blade 100. Therefore, we cannot simply compare the results, but both in “biasynth_2p4g” and “lnamixbias_2p4g,” the proposed method obtained closer results than [10]. The packing results with symmetry constraints of “biasynth_2p4g” and “lnamixbias_2p4g” are shown in Figs. 8 and 9, respectively.

The runtime of experiments without symmetry constraints is much faster than that with symmetry constraints, since linear expressions solved by the simplex method are very few. From this matter, we find the simplex method very slow, but if we can use the state-of-the-art linear programming solvers, we can expect to obtain the results with symmetry constraints in much shorter time.

V. Conclusion

In this paper, we present the necessary and sufficient conditions for the existence of the cell placement that satisfies the given symmetry constraints and the topology constraints

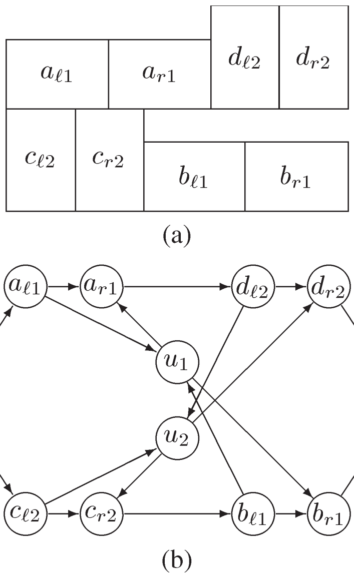

Fig. 10. (a) Placement based on seq-pair $(a_{\ell 1}\, a_{r1}\, d_{\ell 2}\, d_{r2}\, c_{\ell 2}\, c_{r2}\, b_{\ell 1}\, b_{r1};\, c_{\ell 2}\, c_{r2}\, b_{\ell 1}\, b_{r1}\, a_{\ell 1}\, a_{r1}\, d_{\ell 2}\, d_{r2})$. (b) Graph used for infeasible judgment.

imposed by a seq-pair. We defined a seq-pair that satisfies the conditions as "symmetric-real-feasible seq-pair." Then, we proposed an efficient method of obtaining the cell placement that satisfies the given constraints. In the proposed method, a simple constraint graph [6] is obtained directly from a seq-pair in $O(sn \log\log n + es)$ time, and a set of linear constraint expressions is derived from the graph. In order to shorten the time required by linear programming, the number of linear expressions is decreased by substituting the expressions for dependent variables. We used linear programming to solve the linear expressions and obtained the resultant placement. If the symmetry axis is vertical only, the placement is obtained more quickly by the vertical constraint graph based on a seq-pair. Our future problems are experiments on industrial data of analog circuits, evaluation of wirelength, more speed-up of the proposed method, and follow-up of other constraints in analog circuits.

Appendix
Efficient Judgment of Infeasible Seq-Pair

In the proposed method of using linear programming, when there is a placement to satisfy both a given seq-pair and symmetry constraints, the seq-pair is said to be *feasible* (otherwise *infeasible*), and the proposed method can necessarily obtain the closest placement based on it. However, the method requires long time since linear programming is used.

Hence, we consider to speedily remove infeasible seq-pairs as many as possible before judgment on infeasible or feasible is done and, if infeasible, to do so before determining x and y coordinates. The discussion is based on the assumption that there are no cells, which belong to plural symmetry groups, that is, each cell belongs to at most one symmetry group.

When there are plural symmetry groups, infeasible seq-pair overlooked by the condition of symmetric-real-feasible seq-pair mentioned in Section III-A exists. For example,

```
Algorithm Infeasible-check
    Input: seq-pair $S$, symmetry constraints;
    Output: A judgment of infeasible;
{ Make horizontal constraint graph $G_h(V, E_h)$
  from a given seq-pair $S$;
  foreach ($g$ = each symmetry group){
    if (there is a path from $a_\ell$ to $b_\ell$)
      put a directed edge from $b_r$ to $a_r$ in $E_h$;
    Add vertex $u$ corresponding to symmetry axis
    of $g$ to $V$;
    foreach ($c$ = self symmetric cells belong
    to symmetry group $g$){
      Merge a vertex corresponding to $c$ and $u$;
    }
    Add directed edges from all vertices corresponding
    to left cells in $g$ to $u$ in $E_h$;
    Add directed edges from $u$ to all vertices
    corresponding to right cells in $g$ in $E_h$;
  }
  if ($G_h(V, E_h)$ has cycles)
    return "seq-pair $S$ is infeasible";
  return;
}
```

Fig. 11. Algorithm Infeasible-check.

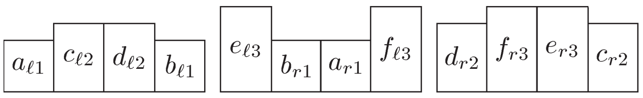

Fig. 12. Placement based on seq-pair $(a_{\ell 1}\, c_{\ell 2}\, d_{\ell 2}\, b_{\ell 1}\, e_{\ell 3}\, b_{r1}\, a_{r1}\, f_{\ell 3}\, d_{r2}\, f_{r3}\, e_{r3}\, c_{r2};\, a_{\ell 1}\, c_{\ell 2}\, d_{\ell 2}\, b_{\ell 1}\, e_{\ell 3}\, b_{r1}\, a_{r1}\, f_{\ell 3}\, d_{r2}\, f_{r3}\, e_{r3}\, c_{r2})$.

assume seq-pair $(a_{\ell 1}\, a_{r1}\, d_{\ell 2}\, d_{r2}\, c_{\ell 2}\, c_{r2}\, b_{\ell 1}\, b_{r1};\, c_{\ell 2}\, c_{r2}\, b_{\ell 1}\, b_{r1}\, a_{\ell 1}\, a_{r1}\, d_{\ell 2}\, d_{r2})$ is given under two symmetry groups $\{(a_{\ell 1}, a_{r1}), (b_{\ell 1}, b_{r1})\}$ and $\{(c_{\ell 2}, c_{r2}), (d_{\ell 2}, d_{r2})\}$. From this seq-pair, y coordinates can be easily obtained like the lower left corner placement obtained by ignoring symmetry constraints, as shown in Fig. 10(a), where there are two rows of cells. However, the relative position of two symmetric axes on the x direction obtained from two cell pairs on the upper row and that obtained from two cell pairs on the lower row contradicts. Therefore, there is no placement that satisfies all constraints.

As mentioned before, in order to judge infeasible seq-pair, where the relative position of two symmetric axes on the x direction contradicts, we can use the method that is shown in Fig. 11. For example, if we apply the aforementioned seq-pair to the method of Fig. 11, the graph, as shown in Fig. 10(b), is obtained. There is a cycle $(u_1, a_{r1}, d_{\ell 2}, u_2, c_{r2}, b_{\ell 1}, u_1)$ in this graph, so the seq-pair is judged as infeasible.

When there are plural symmetry groups, there are complicated cases where a seq-pair becomes infeasible even if there is no contradiction in the relative position between each coordinate of the symmetry axis. As an example, we consider whether the placement is possible or not when seq-pair $(a_{\ell 1}\, c_{\ell 2}\, d_{\ell 2}\, b_{\ell 1}\, e_{\ell 3}\, b_{r1}\, a_{r1}\, f_{\ell 3}\, d_{r2}\, f_{r3}\, e_{r3}\, c_{r2};\, a_{\ell 1}\, c_{\ell 2}\, d_{\ell 2}\, b_{\ell 1}\, e_{\ell 3}\, b_{r1}\, a_{r1}\, f_{\ell 3}\, d_{r2}\, f_{r3}\, e_{r3}\, c_{r2})$ and three symmetry constraints $\{(a_{\ell 1}, a_{r1}),$

$(b_{\ell 1}, b_{r1})\}$, $\{(c_{\ell 2}, c_{r2}), (d_{\ell 2}, d_{r2})\}$, and $\{(e_{\ell 3}, e_{r3}), (f_{\ell 3}, f_{r3})\}$ are given.

Here, y coordinates are easily determined to have the same value since all cells are aligned in a row like the placement obtained by ignoring symmetry constraints, as shown in Fig. 12. However, in this example, it is impossible to determine x coordinates, and the seq-pair is judged to be infeasible since "$c_{\ell 2}, d_{\ell 2}$" comes between $a_{\ell 1}$ and $b_{\ell 1}$, "e_{r3}, f_{r3}" comes between c_{r2} and d_{r2}, and "a_{r1}, b_{r1}" comes between $e_{\ell 3}$ and $f_{\ell 3}$.

Acknowledgment

The authors would like to thank Assistant Professor F. Balasa of the University of Illinois at Chicago and Associate Professor Y. Takashima of the University of Kitakyushu for their kind support.

References

[1] J. Cohn, D. Garrod, R. Rutenbar, and L. Carley, *Analog Device-Level Layout Automation*. Norwell, MA: Kluwer, 1994.

[2] F. Balasa and K. Lampaert, "Symmetry within the sequence-pair representation in the context of placement for analog design," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 19, no. 7, pp. 721–731, Jul. 2000.

[3] K. Krishnamoorthy, S. C. Maruvada, and F. Balasa, "Fast evaluation of symmetric-feasible sequence-pairs for analog topological placement," in *Proc. 5th IEEE Int. Conf. ASICON*, 2003, pp. 71–74.

[4] H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani, "Rectangle-packing-based module placement," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, 1995, pp. 472–479.

[5] L. Rijnders, P. Six, and H. J. De Man, "Design of a process-tolerant cell library for regular structures using symbolic layout and hierarchical compaction," *IEEE J. Solid-State Circuits*, vol. 23, no. 3, pp. 714–721, Jun. 1988.

[6] R. Okuda, T. Sato, H. Onodera, and K. Tamaru, "An algorithm for layout compaction problem with symmetry constraint," *IEICE Trans. Fundam.*, vol. J70-A, no. 3, pp. 536–543, 1990. (in Japanese).

[7] X. Tang, R. Tian, and D. F. Wong, "Fast evaluation of sequence pair in block placement by longest common subsequence computation," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 20, no. 12, pp. 1406–1413, 2001.

[8] J.-G. Kim and Y.-D. Kim, "A linear programming-based algorithm for floorplanning in VLSI design," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 22, no. 5, pp. 584–592, May 2003.

[9] H. Saito and K. Fujiyoshi, "The improved method of L-shaped block packing," in *Proc. 13th Workshop Circuits and Syst. Karuizawa*, 2000, pp. 245–250. (in Japanese).

[10] F. Balasa, S. C. Maruvada, and K. Krishnamoorthy, "On the exploration of the solution space in analog placement with symmetry constraints," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 23, no. 2, pp. 177–191, Feb. 2004.

[11] Y.-X. Pang, F. Balasa, K. Lampaert, and C.-K. Cheng, "Block placement with symmetry constraints based on the O-tree non-slicing representation," in *Proc. ACM/IEEE Des. Autom. Conf.*, 2000, pp. 464–467.

Shinichi Koda received the B.E. and M.E. degrees in electrical and electronic engineering from Tokyo University of Agriculture and Technology, Koganei, Japan, in 2005 and 2007, respectively.

He is currently with Nintendo Company, Ltd., Kyoto, Japan. His research interests include VLSI layout design, especially floor planning and packing for analog IC design.

Chikaaki Kodama (S'04–M'06) received the B.E., M.E., and D.E. degrees in electronic and information engineering from Tokyo University of Agriculture and Technology, Koganei, Japan, in 1999, 2001, and 2006, respectively.

He was with Fujitsu Ltd., Kawasaki, Japan, from 2001 to 2003, where he worked on custom computer-aided design (CAD) development for processor design of the SPARC architecture. He is currently with Toshiba Microelectronics Corporation, Yokohama, Japan. His research interests include VLSI layout design, especially floor planning and packing, and apparel CAD systems.

Dr. Kodama is a member of the Institute of Electronics, Information and Communication Engineers.

Kunihiro Fujiyoshi (M'96) received the B.E., M.E., and D.E. degrees in electrical and electronic engineering from Tokyo Institute of Technology, Tokyo, Japan, in 1987, 1989, and 1994, respectively.

From 1992 to 1996, he was a Research Associate at the School of Information Science, Japan Advanced Institute of Science and Technology, Ishikawa, Japan. In 1997, he joined Tokyo University of Agriculture and Technology, Koganei, Japan, as a Lecturer and has been an Associate Professor since 2000 at the Department of Electrical and Electronic Engineering, Tokyo University of Agriculture and Technology. His research interests include combinatorial algorithms and VLSI layout design.

Dr. Fujiyoshi is a member of the Institute of Electronics, Information and Communication Engineers and the Information Processing Society of Japan.