Analog Placement with Common Centroid and 1-D Symmetry Constraints

Linfu Xiao

Department of CSE
The Chinese University of Hong Kong
Shatin, N.T. Hong Kong
e-mail: lfxiao@cse.cuhk.edu.hk

Evangeline F.Y. Young

Department of CSE
The Chinese University of Hong Kong
Shatin, N.T. Hong Kong
e-mail fyyoung@cse.cuhk.edu.hk

Abstract— [1]In this paper, we will present a placement method for analog circuits. We consider both common centroid and 1-D symmetry constraints, which are the two most common types of placement requirements in analog designs. The approach is based on a symmetric feasible condition on the sequence pair representation that can cover completely the set of all placements satisfying the common centroid and 1-D symmetry constraints. This condition is essential for a good searching process to solve the problem effectively. Symmetric placement is an important step to achieve matchings of other electrical properties like delay and temperature variation. We have compared our results with those presented in the most updated previous works. Significant improvements can be obtained by our approach in both common centroid and 1-D symmetry placements, and we are the first who can handle both constraints simultaneously.

I. Introduction

Current trends in chip design lean towards the integration of both analog and digital circuits on a single die. Digital and analog VLSI design methodologies are very different though. The integration of high-performance analog and digital circuits leads to an increasing need of new tools compatible for both the digital and analog parts. Unfortunately, the low acceptance of CAD tools in the analog domain presents a serious bottleneck to the fast realization of mixed-signal systems. Due to a higher sensitivity of the electrical performance to layout details, analog designs are much more complicated than digital ones. Process and temperature variations can introduce severe mismatches in devices that are designed to behave identically. These undesirable effects can be alleviated by a symmetric layout. Matching and symmetry in placement and routing in analog circuits are thus of immense importance.

Placement of analog circuits is an error prone and time-consuming process. It can easily take an experienced designer weeks or months to layout even a relatively small circuit. Some devices are needed to be placed at close proximity and symmetrically with respect to an axis or to a center point. This can reduce the effect of parasitic mismatches, which will cause degradation of the circuit performance. For example, failure to adequately balance thermal coupling in a differential circuit can introduce unwanted oscillations. Circuit sensitivity to thermal gradients or process variations can also be reduced by placing the symmetric devices close to each other.

A. Previous Works

In 1-D symmetric placement, pairs of cells are required to be placed symmetrically with respect to a horizontal or a vertical axis. This problem has been studied extensively. Balasa *et al.* [8] derived a 1-D symmetric-feasible condition for the sequence pair representation. Lin *et al.* [11] presented a 1-D symmetric-feasible condition for the TCG-S representation. Kouda *et al.* [14] further looked into the symmetric-feasible condition in sequence pair and proposed a linear programming based method. Tam *et al.* [12] handled 1-D symmetry by inserting dummy nodes into constraint graphs and their approach can handle other placement constraints in addition to the 1-D symmetry constraint. Lin *et al.* [13] proposed a placement algorithm based on the B*-tree representation to handle both 1-D and 2-D symmetry. Zhang *et al.* [15] proposed a symmetry-aware placement method based on the Transitive Closure Graph (TCG) representation and derived a symmetric-feasible condition on TCG. For the common centroid constraint, Ma *et al.* [2] proposed a method based on the C-CBL representation, which can handle mosaic packings only.

B. Our Contributions

In this paper, we will present an analog placement tool that can handle both common centroid and 1-D symmetry constraints. In analog layout, common centroid placement is an essential requirement in order to reduce mismatches in devices. In some cases when the devices are small and the requirement in accuracy is not that stringent, a common centroid constraint can be relaxed to a 1-D symmetry constraint with devices placed in close proximity. Therefore, both common centroid and 1-D symmetry constraint are important to analog placement. As long as we know, this is the first work that considers both constraints simultaneously. Our approach can perform symmetric placement of an input analog circuit effectively. The approach is based on a symmetric feasible condition on the sequence pair representation that can cover completely without redundancy the set of all placements satisfying the common centroid and 1-D symmetry constraints. This condition is essential for a good searching process to solve the problem effectively, allowing it to search in a solution space without redundancy. We also introduced the concept of *group*

[1]The work described in this paper was partially supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK418106).

clustered placement to avoid placing other asymmetrical blocks within a symmetry group, so that the blocks in the same symmetry group can be placed in close proximity to reduce mismatches. We have compared our results with those presented in the most updated previous works. Significant improvements can be obtained by our approach in both common centroid and 1-D symmetry placements, and we are the first who can handle both constraints simultaneously.

The remainder of this paper is organized as follows. Section 2 presents some related works and findings. Section 3 gives the system overview and problem formulation. Section 4 presents some preliminaries about sequence pairs. The symmetric feasible condition for common centroid and 1-D symmetry constraints will be presented in Section 5. Section 6 shows the implementational details of our placement method. Experimental results will be shown in Section 7, followed by a conclusion in Section 8.

II. Related Works

Balasa *et al.* [8] proposed a 1-D symmetric-feasible condition in sequence pair. If x and y are two blocks in a 1-D symmetry group with a horizontal axis, a sequence pair (α, β) satisfying the following conditions will be feasible for a symmetric placement of the group:

$$\alpha_x^{-1} < \alpha_y^{-1} \Leftrightarrow \beta_{sym(y)}^{-1} < \beta_{sym(x)}^{-1} \quad (1)$$

where $sym(x)$ is the block symmetric to x and α_x^{-1} (β_x^{-1}) denotes the position of block x in the sequence α (β). The above feasible condition can be re-written for a symmetry group with a vertical axis as follows:

$$\alpha_x^{-1} < \alpha_y^{-1} \Leftrightarrow \beta_{sym(x)}^{-1} < \beta_{sym(y)}^{-1}$$

However, these conditions are later found to be sufficient but not necessary [14]. For instance, the sequence pair of the placements depicted in Fig. 1 does not satisfy the symmetric-feasible condition but the placement is 1-D symmetrical.

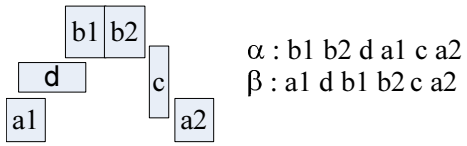


Fig. 1. Counter-example to the Symmetric Feasible Condition in [8]

Zhang *et al.* [15] proposed a more general 1-D symmetric-feasible condition using the TCG representation:

$$x \vdash y \Leftrightarrow sym(x) \vdash sym(y) \quad (2)$$

$$x \perp y \Leftrightarrow sym(y) \perp sym(x) \quad (3)$$

where $x \vdash y$ ($x \perp y$) denotes that block x is to the left of (below) block y, and $\Leftrightarrow$ denotes that the conditions on the two sides cannot occur simultaneously. The solution space of these conditions is complete, but there will be redundancy in the mapping. For example, there may be more than one symmetric-feasible TCG representations for one symmetrical placement. This redundancy will increase the size of the solution space and thus will slow down the placement process.

III. System Overview

We aim at developing a complete layout tool for analog circuit designs. The structure of the whole system can be summarized in Fig. 2. Taken as input an analog circuit and a corresponding set of matching requirements on different electrical properties, a symmetrical placement is first generated by the placer and based on that, symmetric routing is performed. Finally, testing and verification is done and the discrepancies will be fed back to the placer and the router to make appropriate changes. A verified layout of the input analog circuit will be output at the end. The scope of this paper will mainly cover the placer part.

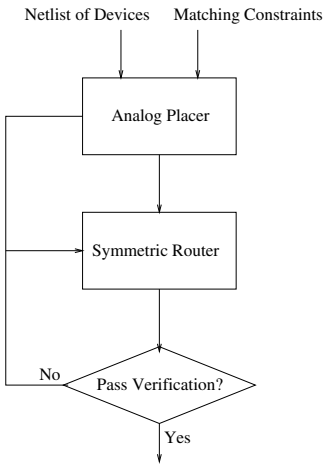


Fig. 2. System Overview

For the placement problem, we are given a netlist of devices and a set of symmetry groups. Each symmetry group contains a number of devices which are required to be placed symmetrically with respect to a single point (common centroid constraint) or to a horizontal or a vertical axis (1-D symmetry constraint). The objective is to generate a non-overlap placement of the devices such that all the symmetrical requirements are satisfied. We define the problem formally as follows:

Definition 1 Analog Placement Problem *Given a set of n blocks with dimensions $w_i \times h_i$ for $i = 1 \ldots n$, a set of m nets, and a set of k symmetry groups $g_1, g_2 \ldots g_k$, we want to obtain a placement F of the circuit satisfying the common centroid or 1-D symmetry constraint for each group, while minimizing a cost function $cost(F) = area(F) + \lambda \times wire(F)$ where $area(F)$ is the total area of F, $wire(F)$ is the total wire length of F measured by the half-perimeter estimation, and λ is a parameter specifying the relative importance between these two terms. Each symmetry group g_i is consisted of a number of self-symmetry blocks and a number of symmetrical pairs $(x, sym(x))$, which are required to be placed in a common centroid or a 1-D symmetry geometry.*

The analog placer is simulated annealing based using sequence pair as the representation. Our basic strategy is to identify a symmetric feasible condition on sequence pair that can cover completely without redundancy the set of all placements satisfying the common centroid or 1-D symmetry constraints.

In this way, the searching process can be done effectively in a complete and non-redundant feasible solution space. We will thus review the sequence pair representation in the following and discuss some of its important properties applicable in the subsequent derivations and proofs. The symmetric feasible condition for common centroid and 1-D symmetry constraints on sequence pair will be proposed and proved in the next section, followed by a detailed description of the algorithm for the placement process.

IV. Sequence Pair

A sequence-pair (SP) [3], describing a general placement of n blocks, is an ordered pair (α, β), where α and β are permutations of the block names. We use α_i to denote the block occupying the i^{th} position in sequence α and use α_A^{-1} to denote the position of block A in the α sequence. The topological relations between two blocks A and B in a sequence pair are given as follows:

- If $\alpha_A^{-1} < \alpha_B^{-1}$ and $\beta_A^{-1} < \beta_B^{-1}$, block A is to the left of block B;
- If $\alpha_A^{-1} < \alpha_B^{-1}$ and $\beta_B^{-1} < \beta_A^{-1}$, block A is above block B.

In the following, we will use the notation $A \vdash B$ ($A \perp B$) to denote that block A is to the left of (below) block B. Given a sequence pair, the topological relationship between any two blocks is unique, either horizontal or vertical, i.e., there is only one edge in either the horizontal or the vertical constraint graph between any two blocks.

A. Properties of Sequence Pair

In this section, we will prove some properties of the sequence pair representation which will be useful for the derivation of the common centroid and the 1-D symmetric feasible conditions in the subsequent sections. We define *deletable blocks* and *deletion sequence* as follows:

Definition 2 Deletable Blocks *Given a placement, a block is upper-left (lower-left) deletable if and only if all the other blocks are either below (above) or to the right of it. Similarly, a block is upper-right (lower-right) deletable if and only if all the other blocks are either below (above) or to the left of it.*

Definition 3 Deletion Sequence *Given a placement of n blocks, an upper-left (lower-left) deletion sequence is a sequence of n block names obtained by removing an upper-left (lower-left) block one after another from the placement. Similarly, an upper-right (lower-right) deletion sequence is a sequence of n block names obtained by removing an upper-right (lower-right) block one after another from the placement.*

Notice that the upper-left deletion sequence (or deletion sequence in other directions) is not unique, since there can be more than one upper-left deletable blocks in a placement. In the following, we relate the deletion sequences with the sequence pair representation.

Lemma 1 *Given a placement, a pair of sequences (a,b), where a is an upper-left deletion sequence and b is a lower-left deletion sequence, is a valid sequence pair representation of the placement.*

Proof: Given a placement P of n blocks, consider any pair of blocks x and y. Suppose that the topological relationship between x and y is horizontal and x is on the left of y. Then, in an upper-left deletion sequence a of P, x will be before y, and in a lower-left deletion sequence b of P, x will also be before y. Thus, the pair (a,b) can correctly describe the relationship between x and y when interpreted as a sequence pair representation. Similarly, if the topological relationship between x and y is vertical and x is above y. Then, in an upper-left deletion sequence, x will be before y, and in a lower-left deletion sequence, y will be before x. Thus, the pair (a,b) also describes the relationship between x and y correctly when being interpreted as a sequence pair representation of P. Therefore, we can conclude that (a,b) is a valid sequence pair representation of P. □

Lemma 2 *Given a placement, a pair of sequences (a,b), where a is a reversed lower-right deletion sequence and b is a reversed upper-right deletable sequence, is a valid sequence pair representation of the placement.*

The proof of Lemma 2 is very similar to that of Lemma 1 and it is skipped here because of the space limitation.

V. Symmetric Feasible Conditions

In this section, we will propose and prove the feasible conditions on sequence pair for common centroid and 1-D symmetrical placements. In our formulation, we will consider *group clustered* placement in which no other asymmetrical blocks will be placed *in between* a common centroid or a 1-D symmetry group. We will define it formally in the next section. In the following, we will use the notation (α_g, β_g) to denote the subsequence pair obtained from a sequence pair (α, β) by extracting just the blocks in a symmetry group g. For example, if the original sequence pair (α, β) is $(dabeca'b', cb'deaba')$ and if a symmetry group contains blocks $\{a, a', b, b', c\}$, the extracted sequence (α_g, β_g) will be $(abca'b', cb'aba')$. In the following sections, we will first discuss the condition for group clustered placements, followed by the symmetric feasible conditions on common centroid and 1-D symmetrical group clustered placements.

A. Group Clustered Placements

In analog circuits, devices in the same symmetry or common centroid group are prefered to be placed in close proximity without being interleaved with blocks not from the same group, because mismatches and parasitic effects will be much larger otherwise. We thus defined a type of placements, called group clustered placement, satisfying this constraint and consider them as candidate solutions only during the searching process.[2]

[2]Some previous works will treat each group as a rectangular super-block, which is not necessary and may result in a significant wastage in area

Definition 4 Group Clustered Placement *A placement containing a symmetry group is a group clustered placement if and only if no asymmetric blocks (blocks not belonging to that group) lie between any two blocks in the group horizontally or vertically.*

The searching space can be reduced if we only consider such kind of placements. The condition for group clustered placement[3] in sequence pair can be easily derived as

Lemma 3 *The if and only if condition on a sequence pair* (α,β) *representing a group clustered placement containing a symmetry group g is that for any block z not belonging to g, there exists no blocks, u, v, x and y in g, such that* $\alpha_u^{-1} < \alpha_z^{-1} < \alpha_v^{-1}$ *and* $\beta_x^{-1} < \beta_z^{-1} < \beta_y^{-1}$.

Proof: Consider a group clustered placement containing a symmetry group g. Consider any block z that is not in the group. This block must be either at the upper-left, upper-right, lower-left or lower-right direction of the group. W.l.o.g., lets consider the direction of upper-left, i.e., for every block x in the group, z is either on the left of x or above x. If z is on the left of x, the sequence pair (α,β) will satisfy the condition that $\alpha_z^{-1} < \alpha_x^{-1}$ and $\beta_z^{-1} < \beta_x^{-1}$. If z is above x, the sequence pair (α,β) will satisfy the condition that $\alpha_z^{-1} < \alpha_x^{-1}$ and $\beta_x^{-1} < \beta_z^{-1}$. Thus, $\alpha_z^{-1} < \alpha_x^{-1}$ for every block x in the group. Similarly, we can deduce that if block z is in the upper-right, lower-left or lower-right direction, it must be lying outside the group in sequence α or β.

The proof of the only if direction is based on the *gridding-based* packing method in [3]. Consider an $n \times n$ grid rotated by 45 degree clockwise. Each block x can be placed into the grid at position $(\alpha_x^{-1},\beta_x^{-1})$. An example is shown in Fig. 3. In this example, suppose a_1 and a_2 belong to one group, x_1 and x_2 belong to another group and r and t are blocks not belonging to any group. If a sequence pair satisfys the condition stated in the lemma, we know that the bounding box of a_1 and a_2 in the rotated grid has no other blocks lying inside and so as the bounding box of x_1 and x_2. If this is true, we can always move the gridlines in the rotated grid in such a way that the placement is group clustered, i.e., no asymmetric blocks (blocks not belonging to that group) lie between any two blocks in the group horizontally or vertically, in the original coordinate system. □

Lets look at an example sequence pair $(a_1x_1x_2a_2, x_1a_1a_2x_2)$ in which a_1 and a_2 belongs to one symmetry group and x_1 and x_2 belongs to another group. This sequence pair satisfys the condition as stated in Lemma 3 for group clustered placement and there exists a corresponding placement (Fig. 4) in which both groups are placed with no asymmetrical blocks inside.

B. Common Centroid Placement

Consider a common centroid placement of a single group g and its corresponding extracted sequence pair (α_g,β_g). When a block $x \in g$ becomes upper-left (upper-right) deletable, its symmetric counterpart $sym(x)$ should also be lower-right (lower-left) deletable. Therefore, according to Lemma 1 and 2, a pari of α_g and β_g sequences satisfying the following conditions will be feasible.

$$\alpha_g = sym(rev(\alpha_g)) \quad (4)$$

$$\beta_g = sym(rev(\beta_g)) \quad (5)$$

where the operation $rev(s)$ is reversing the string s and the operation $sym(s)$ is replacing all the blocks x in string s by their symmetric counterparts, i.e., $sym(x)$. An example is shown in Fig. 5. In this example, x_1 and x_2 are symmetric pairs for $x = \{a,b,c,d\}$ in a group. We will proof the correctness of this condition in the following lemmas and theorem.

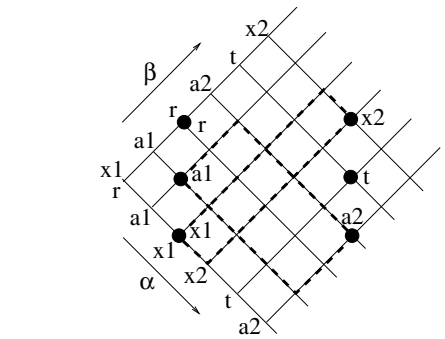


Fig. 3. An Example Illustrating the Only-if Condition of Lemma 3

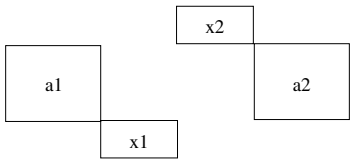
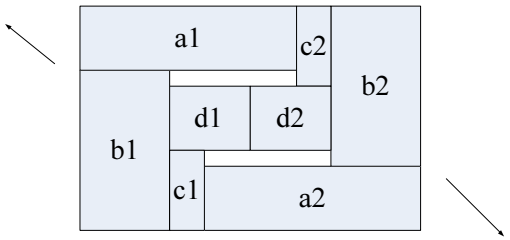


Fig. 4. An Example Illustrating the Condition of Lemma 3



upper-left deletion sequence(α): a1b1d1c1c2d2b2a2
lower-right deletion sequence(reversed α): a2b2d2c2c1d1b1a1

Fig. 5. Common Centroid Placement

Lemma 4 *A group clustered placement P of n blocks with k common centroid groups* g_i *for* $i = 1 \ldots k$ *can be represented by a sequence pair* (α,β) *such that the group clustered placement condition (as stated in Lemma 3) is true for each group* g_i *and the extracted sequence pair* $(\alpha_{g_i},\beta_{g_i})$ *for* $i = 1 \ldots k$ *satisfies the conditions (4) and (5).*

[3]This condition is equivalent to one of the three conditions in [19] for feasible recovering of rectilinear blocks from a sequence pair representation. However, that is unrelated to our current problem since we are now dealing with the condition for common centroid or 1-D symmetric placement, not the rectilinear block packing problem in [19]

Proof: First of all, the group clustered condition for each group g_i for $i = 1 \ldots k$ follows naturally according to the proof of Lemma 3. Now, consider the sequence pair (α, β) of the whole placement P and a particular group $g \in \{g_i | i = 1 \ldots k\}$. According to Lemma 1 and 2, α can be constructed as an upper-left deletion sequence or the reversed lower-right deletion sequence. Consider the upper-left deletion process. When a block $x \in g$ becomes upper-left deletable, there are two cases for its symmetric counterpart $sym(x)$: (1) $sym(x)$ is lower-right deletable at the same time, or (2) $sym(x)$ is not lower-right deletable. For the second case, among all those currently lower-right deletable blocks, none of them should be in g_i because of the common centroid requirement. We can delete them one by one until $sym(x)$ becomes lower-right deletable, then we can delete $sym(x)$. By following this procedure, we can construct the sequence α satisfying the condition that $\alpha_g = sym(rev(\alpha_g))$. We can argue similarly for the β sequence. □

Lemma 5 *Consider a sequence pair (α, β) describing a placement P of n blocks with k common centroid groups g_i for $i = 1 \ldots k$ satisfying the group clustered placement condition (as stated in Lemma 3) for each group and the conditions (4) and (5) for each extracted sequence pair $(\alpha_{g_i}, \beta_{g_i})$ for $i = 1 \ldots k$, there exists a corresponding group clustered placement with each group g_i satisfying the common centroid constraint.*

Proof: The proof is based on the *gridding-based* packing method in [3]. Consider an $n \times n$ grid rotated by 45 degree clockwise. Each block x can be placed into the grid at position $(\alpha_x^{-1}, \beta_x^{-1})$. An example is shown in Fig. 6. Note that we can always enlarge the rows and columns of the grid uniformly such that no two blocks overlap. This packing will correspond to a valid packing of the sequence pair (α, β). Consider a particular group $g \in \{g_i | i = 1 \ldots k\}$. Let a and b be the leftmost and the rightmost block of group g in the α sequence, and c and d be the leftmost and the rightmost block of group g in the β sequence. Consider the rectangle R in the grid with lower-left coordinates at $(\alpha_a^{-1}, \beta_c^{-1})$ and upper-right coordinates at $(\alpha_b^{-1}, \beta_d^{-1})$.

First of all, for each block y not in g, y must lie outside R according to the group clustered placement condition, so this gridding-based placement is a group clustered placement for g. Besides, since the extracted sequence pair (α_g, β_g) satisfys the conditions (4) and (5), their placement in the grid will be symmetrical and the common centroid constraint can be satisfied by adjusting the distances between the gridlines appropriately. Therefore, we can conclude that there exists a corresponding group clustered placement with each group g_i satisfying the common centroid constraint. □

Theorem 1 *The solution space of all possible group clustered placements with n blocks and k common centroid group g_i for $i = 1 \ldots k$ satisfying the common centroid constraint in all the groups can be covered exactly by the set of all sequence pairs satisfying the group clustered placement condition (as stated in Lemma 3) and the conditions (4) and (5) for each extracted sequence $(\alpha_{g_i}, \beta_{g_i})$ for $i = 1 \ldots k$.*

Proof: Follows from Lemma 4 and 5. □

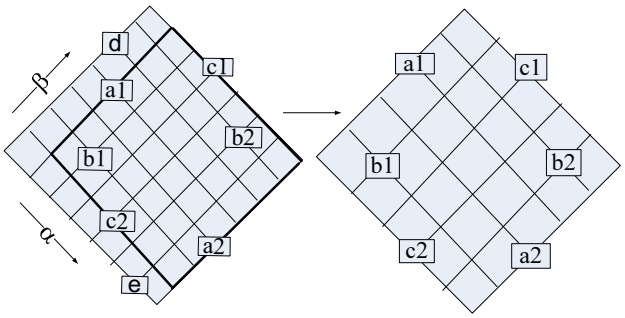
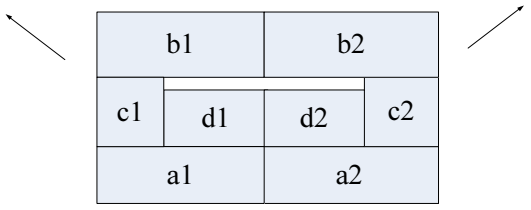


Fig. 6. A Gridding-based Packing for Sequence Pair $(\alpha, \beta) = (da_1b_1c_1c_2b_2ea_2, ec_2b_1a_2a_1db_2c_1)$



upper-left deletion sequence(α): b1b2c1d1d2c2a1a1
upper-right deletion sequence(reversed β): b2b1c2d2d1c1a2a1

Fig. 7. 1-D Symmetric Placement

C. 1-D Symmetry Placement

Using a similar approach, we can obtain the feasible condition on sequence pair for a 1-D symmetry group g in a placement as follows:

$$\alpha_g = sym(rev(\beta_g)) \tag{6}$$

An example is illustrated in Fig. 7. We can prove similarly the following theorem, but the proof is skipped here because of the lack of space.

Theorem 2 *The solution space of all possible group clustered placements with n blocks and k 1-D symmetry group g_i for $i = 1 \ldots k$ satisfying the 1-D symmetry constraint for all the groups can be covered exactly by the set of all sequence pairs satisfying the group clustered placement condition (as stated in Lemma 3) and the conditions (6) for each extracted sequence $(\alpha_{g_i}, \beta_{g_i})$ for $i = 1 \ldots k$.*

VI. Methodology

Simulated annealing is used with the sequence pair representation [3] as the basic searching engine in our approach. We will just search for those sequence pairs satisfying the group clustered placement condition and the common centroid and 1-D symmetry conditions. According to our above discussions, this solution space will cover all the required feasible solutions.

A. Floorplan Realization

In floorplan realization, we will construct a pair of constraint graphs and perform the single source longest path algorithm on

the graphs. A pair of coordinates will be obtained for each block. We then need to further adjust the coordinates of the blocks in a group in order to satisfy the required placement constraints, since these constraints are not taken into account in the lower-left compacted packing process. We have devised a scheme, as described by the pseudo-code *AdjustingHorizontalSymmetry*, to perform horizontal adjustment for common centroid groups, while the vertical adjustment can be done similarly.

For each common centroid group g, we will first compute the x-position of its center, according to the x-coordinates of each symmetric pair. We will then compute the potential displacement δ for each right-hand-side block b_2 of a symmetric pair (b_1, b_2). This displacement δ will be added to the weights of all the out-going edges $e(b_2, x)$ of b_2 where $x \notin g$ in the horizontal constraint graph so that all the blocks to the right of b_2 will be moved by δ. Finally, we will re-compute the longest paths to determine the new x positions of the affected blocks and move b_2 to the right. After this process, every symmetric pair in the same group will have the same center point, and no overlapping will be resulted because of the properties of sequence pair (a constraint edge exists between every pair of blocks horizontally or vertically). For the 1-D symmetry groups, because of the mirror symmetric requirement, the adjustment process is slightly different from that for the common centroid groups in one of the two directions. If the axis of symmetry of a 1-D symmetry group is horizontal (vertical), the adjustment done to the y-direction (x-direction) will be the same as that for the common centroid group, i.e., the *AdjustingVerticalSymmetry* (*AdjustingHorizontalSymmetry*) process, and the adjustment done in the x-direction (y-direction) will be different. In the following, we will describe a scheme, called *AdjustingHorizontalAlign*, to perform horizontal adjustment for 1-D symmetry groups, while the vertical adjustment can be done similarly. After this process, every symmetric pair in the same group will align in the x-direction, and no overlapping will be resulted because of the properties of sequence pair.

Algorithm 1 AdjustingHorizontalSymmetry

```
1:  //Horizontal adjustment for common centroid groups
2:  for each common centroid group g do
3:      find the x-position X_symAxis of the center point: X_symAxis = max{(a1.x + a2.x + a1.width)/2 | (a1, a2) ∈ g}
4:      for each right-hand-side block b2 in a symmetric pair (b1, b2) of g do
5:          calculate potential displacement δ that b2 should be shifted: δ = 2 * X_symAxis − (b2.x + b1.x + b1.width)
6:          add δ to the weights of all the out-going edges e(b2, x) of b2 where x ∉ g
7:      end for
8:  end for
9:  re-calculate longest paths in the horizontal constraint graph
10: for each common centroid group g do
11:     re-compute the x-position X_symAxis of the center point
12:     for each right-hand-side block b2 in a symmetric pair (b1, b2) of g do
13:         move b2 to satisfy the constraint: b2.x = 2 * X_symAxis − (b1.x + b1.width)
14:     end for
15: end for
```

Algorithm 2 AdjustingHorizontalAlign

```
1:  //Horizontal adjustment for 1-D symmetry groups with horizontal axis
2:  for each 1-D symmetry group g do
3:      for each symmetric pair (b1, b2) of g do
4:          suppose b1.x ≤ b2.x, compute the adjustment: δ = b2.x − b1.x
5:          add δ to the weights of all the out-going edges e(b1, x) of b1 where x ∉ g
6:      end for
7:  end for
8:  re-compute longest paths in the horizontal constraint graph
9:  for each 1-D symmetry group g do
10:     for each symmetric pair (b1, b2) of g do
11:         suppose b1.x ≤ b2.x, move b1 to align with b2: b1.x = b2.x
12:     end for
13: end for
```

B. Set of Moves

We employ the following set of moves that, starting with a sequence pair satisfying the group clustered placement condition and the conditions (4)-(6) for the extracted sequence pair of each group, can generate another candidate sequence pair satisfying the same set of conditions. It is also true that this set of moves is complete, i.e., starting from any feasible solution, we can reach any other feasible solution by using a sequence of one or more moves from this move set.

- Exchange two blocks of a group in α - Randomly choose two blocks x and y in a common centroid (1-D symmetry) group, exchange their positions in α and also exchange the positions of their symmetric counterparts, $sym(x)$ and $sym(y)$, in α (β).
- Exchange two blocks of a group in α and β - Randomly choose two blocks x and y in a group, exchange their positions in both α and β and also exchange the positions of their symmetric counterparts, $sym(x)$ and $sym(y)$, in α and β.
- Exchange two groups - Randomly choose two symmetry groups A and B. Find the shortest continuous subsequence s_1 in α (β) that contains all the blocks in A and the shortest continuous sub-sequence s_2 in α (β) that contains all the blocks in B. Exchange s_1 and s_2 in α (β).[4]
- Move an asymmetric block - Randomly pick an asymmetric block x and modify its position in α or β without violating the group clustered placement condition.
- Exchange two asymmetric blocks - Randomly choose two asymmetric blocks x and y, and exchange their positions in both α and β.
- Change the orientation of a block - Randomly pick a block x and change its orientation. If x is in a group, the orientation of $sym(x)$ is also changed

C. Annealing Schedule and Cost Function

In our annealing engine, the initial temperature is set to 10^6 and the temperature will drop at a constant rate. At each temperature, k iterations are performed, where k is proportional

[4]Note that in our experiments, two different groups will not interleave with each other in the α or β sequences for simplicity.

to the number of blocks. We use the cost function $cost(F) = area(F) + \lambda * wire(F)$ to evaluate a packing F, where $area(F)$ is the area of F and $wire(F)$ is the total wire length estimated by the half perimeter method. The parameter λ specifies the relative importance between area and wire length.

VII. Experimental Results

Our analog placer was implemented with the C programming language using the fast sequence pair evaluation algorithm [16] [17] and all the experiments were performed on an Intel(R) XEON(TM) CPU 2.20GHz linux workstation with 1 GB memory. We have done three sets of experiments. In the first two sets, we compared our approach with the two most updated previous works handling common centroid constraint, and the 1-D symmetry constraints separately. As far as we know, there is no previous works that handle both constraints simultaneously. In the last set of experiments, we tested our placers on its effectiveness in handling both constraints. In all the experiments, the annealing process is repeated ten times for each data set, and the best and average results are reported.

In the first set of experiments, we compare our algorithm with the C-CBL [2] approach, which, as far as we know, is the most updated previous work on this problem, in handling common centroid constraints. For fair comparison, we consider soft blocks in this set of experiments with each block has an upper and a lower aspect ratio bound the same as that in [2]. Table I lists the results. In this table, the second column shows the total number of blocks and the third column on "CC Groups" shows the common centroid group information. For example, for the data set c3, there are two common centroid groups, one of them has 16 blocks and the other one has 20 blocks. In [2], each common centroid group is treated as a rectangular superblock, so more dead spaces will be resulted. If the blocks in a group are very irregular in shape, the result may be unacceptable. We can see from Table I that our approach can out-perform significantly the C-CBL approach in area.

In the second set of experiments, we compare our approach with the HB*-tree approach [13], which, as far as we know, is the best previous work in terms of both quality and running time, on the 1-D symmetry placement problem, using their provided data sets (ami33, ami49 and two industrial designs, biasynth_2p4g and lnamixbias_2p4g). The results are shown in Table II. In this table, the third column on "1-D Groups" shows the 1-D symmetry group information. We can see that our placer out-performs the HB*-tree approach in running time while the area usages are similar.

In the last set of experiments, we want to demonstrate the effectiveness of our approach in handling both common centroid and 1-D symmetry constraint simultaneously. Since there is no previous works handling these two constraints simultaneously[5], only the results of our approach are shown (Table III). We use the same set of data as in [13] with some randomly generated common centroid groups and interconnections (both symmetrical and ordinary ones). We can see that our placer can handle both constraints effectively with a very scalable running time. Two resultant packings of the data set "biasynth_2p4g", "lnamixbias_2p4g" are shown in Fig. 8 and Fig. 9.

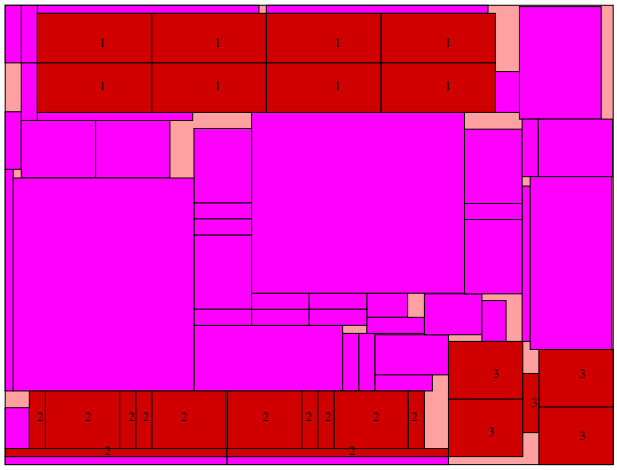


Fig. 8. Result of "biasynth_2p4g" Obtained by Our Approach Dear Space: 4.73%. Blocks labeled 1,2 belong to 1-D symmetry group, while blocks labeled 3 belong to common centroid symmetry group.

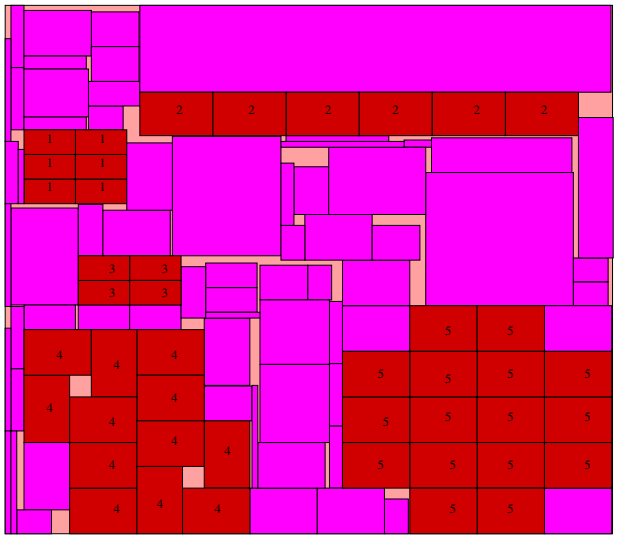


Fig. 9. Result of "lnamixbias_2p4g" Obtained by Our Approach Dear Space: 4.30%. Blocks labeled 1,2, or 5 belong to 1-D symmetry group, while blocks labeled 3 or 4 belong to common centroid symmetry group.

VIII. Conclusion

In this paper, we presented an analog placer that can handle both common centroid and 1-D symmetry constraints. This is based on a symmetric feasible condition on the sequence pair representation that can cover exactly without redundancy the set of all group clustered placement satisfying the constraints. This is the first piece work to propose and prove such feasible condition and is the first which can handle both common centroid and 1-D symmetry constraints simultaneously. We have developed an analog placer based on this condition and experimental results showed that our approach can produce symmetric layouts with short running time.

[5]The 2-D symmetry constraint in [13] is not the same as the common centroid constraint here. Instead, it is just a subset of the common centroid constraint.

TABLE I
HANDLING OF COMMON CENTROID CONSTRAINT

Data Set	Block No.	CC Groups	C-CBL		Our Approach		
			Dead Space (%)	Time (s)	Best Dead Space (%)	Average Dead Space (%)	Time (s)
c1	46	16	8.80	5.2	4.61	5.32	4.9
c2	70	20	10.2	16.3	4.35	4.89	14.5
c3	106	16,20	11.24	46.7	3.78	4.34	50.2
c4	154	16,18,20	11.74	140	3.89	4.23	152
c5	200	20,20,20,20	13.6	275	4.12	5.04	289

TABLE II
HANDLING OF 1-D SYMMETRY CONSTRAINT

Data Set	Block No.	1-D Groups	HB*-tree		Our Approach		
			Dead Space (%)	Time (s)	Best Dead Space (%)	Average Dead Space (%)	Time (s)
ami33	33	6	5.70	12	4.71	5.02	2.0
ami49	49	4	3.80	20	4.00	4.51	5.1
biasynth_2p4g	65	8,12,5	4.47	22	4.69	5.21	13.5
lnamixbias_2p4g	110	16,6,6,12,4	5.40	43	4.76	5.50	53

TABLE III
HANDLING BOTH COMMON CENTROID AND 1-D SYMMETRY CONSTRAINTS

Data Set	Block No.	CC Groups	1-D Groups	Consider Area Only		Consider (Area + WL)			
				Dead Space (%)	Time (s)	Net Number	Dead Space (%)	HPWL (10^3)	Time (s)
ami33	33	6	4	5.1	3.0	87	7.80	15.7	5.6
ami49	49	4	6	4.28	6.45	377	8.92	251.9	12.9
biasynth_2p4g	65	8,12	5	3.91	18.2	62	5.97	65.7	20.0
lnamixbias_2p4g	110	16,6,6	12,4	4.31	58.7	104	8.79	8.54	75.2

REFERENCES

[1] X. Hong, G. Huang, Y. Cai, J. Gu, S. Dong, C.-K. Cheng and J. Gu. Corner Block List: An Effective and Efficient Topological Representation of Non-Slicing Floorplan. *Proceedings of the International Conference on Computer-Aided Design*, 2000.

[2] Qiang Ma and Evangeline F.-Y. Young. Analog Placement with Common Centroid Constraints. *Proceedings of the International Conference on Computer-Aided Design*, 2007

[3] H. Murata, K. Fujiyoushi, S. Nakatake and Y. Kajitani. Rectangle-Packing-Based Module Placement. *Proceedings IEEE International Conference on Computer-Aided Design*, pages 472-479, 1995.

[4] J. Cohn, D. Garrod, R. Rutenbar and L. Carley. Analog Device-level Automation. Kluwer Acad. Publi., 1994.

[5] J. Cohn, *et al*. KOAN/ANAGRAMII: New Tools for Device-Level Analog Layout. *IEEE J. Solid-State Circuits*, 26(3):330-342, 1991.

[6] K. Lampaert, G. Gielen and W. Sansen. A Performance-driven Placement Tool for Analog Integrated Circuits. *IEEE J. Solid-State Circuits*, 30(7):773-780, 1995.

[7] E. Malavasi, E. Charbon, E. Felt and A. Sangiovanni-Vincentelli. Automation of IC Layout with Analog Constraints. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 15(8):923-942, 1996.

[8] F. Balasa and K. Lampaert. Symmetry within the Sequence-Pair Representation in the Context of Placement for Analog Design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2000.

[9] Y.-X. Pang, F. Balasa, K. Lampaert and C.-K. Cheng. Block Placement with Symmetry Constraints based on the O-tree Nonslicing Representation. *Proceedings of the 37th ACM/IEEE Design Automation Conference*, pages 464-467, 2000.

[10] F. Balasa, S.-C. Maruvada and K. Krishnamoorthy. On the Exploration of the Solution Space in Analog Placement with Symmetry Constraints. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 23(2):177-191, 2004.

[11] J.M. Lin, G.M. Wu, Y.W. Chang and J.H. Chuang. Placement with symmetry constraints for analog layout design using TCG-S. *IEEE Asia South Pacific Design Automation Conference*, pp. 1135C1138, 2005.

[12] Y.-C. Tam, Evangeline F.-Y. Young and Chris C.-N. Chu. Analog Placement with Symmetry and Other Placement Constraints. *Proceedings of the International Conference on Computer-Aided Design*, 2006.

[13] P.-H. Lin and S.-C. Lin. Analog Placement Based on Novel Symmetry-Island Formulation. *Proceedings of the 44th ACM/IEEE Design Automation Conference*, 2007.

[14] S. Kouda, C. Kodama and K. Fujiyoshi. Improved method of cell placement with symmetry constraints for analog IC layout design. *Proc. International Symposium on Physical Design*, 2006

[15] L. Zhang and C.-J. R. Shi and Y. Jiang. Symmetry-Aware Placement with Transitive Closure Graphs for Analog Layout Design. *IEEE Asia South Pacific Design Automation Conference*, 2008

[16] Xiaoping Tang, Ruiqi Tian and D.F. Wong. Fast evaluation of sequence pair in block placement by longest common subsequence computation. *Proceedings of the conference on Design, automation and test in Europe*, 2000

[17] Xiaoping Tang and D.F. Wong. FASTSP: A Fast Algorithm for Block Placement based on Sequence Pair. *IEEE Asia South Pacific Design Automation Conference*, 2001.

[18] Chris Chu. FLUTE: Fast Lookup Table Based Wirelength Estimation Technique. *Proceedings of the International Conference on Computer-Aided Design*, 2004.

[19] M.Z. Kang and W.W. Dai. Arbitrary Rectilinear Block Packing based on Sequence Pair, *Proceedings of the International Conference on Computer-Aided Design*, pages 259-266, 1998.