A New Implementation of Multilevel Framework for Interconnect-Driven Floorplanning

Zheng Xu [1,2], Song Chen [1], Takeshi Yoshimura [1] and Yong Fang [2]
[1] Graduate School of Information, Production and Systems, Waseda University, Japan
Hibikino2-6-317, Wakamatsu, Kitakyushu, Fukuoka 808-0135, Japan
[2] School of Communication and Information Engineering, Shanghai University, China
Yanchang Road 149, Shanghai 200072, China
E-mail: ryan_sec@fuji.waseda.jp

Abstract: In this paper, we propose a multilevel fixed-outline floorplanning method, called multilevel IARFP, to deal with floorplanning problem for the large-scale integrated circuit designs. We combine the IARFP [1] into the V-shaped multilevel framework [2], engaging in getting a better result with minimized wirelength. We recursively partition the circuits by using hMetis [3], to get min-cut cost. After the partition stage, we do floorplanning from top to down building sequences for merging and refinement stage. Then we bottom-up merge the sub-regions into big regions until attain the final floorplan. The IARFP is based on Sequence Pair [7] representation; for multilevel case, we present multilevel Sequence Pair representation to handle the floorplanning. We can get about 11% reductions in wirelength within about 55% run time comparing with flat IARFP algorithm.

Keywords: V-shaped Multilevel Framework, Fixed-outline Floorplanning, VLSI Design, Multilevel Sequence Pair

1. Introduction

As the nanometer technology advances, IC integrated density keeps growing at a dramatic speed. A single chip can integrate more and more functions and often includes millions of transistors, even designs with billions transistors are in production.

Traditional flat framework design technology does not scale well as chip design scalability grows. Hierarchical design is an essential method to cope with the increasing design complexity. But lacking of interconnection information makes it under satisfied. To remedy the deficiency, the multilevel framework is proposed to solve the floorplanning problem as well as graph/circuit partitioning, placement, and routing. The Λ-shaped frameworks adopt a two-stage technique, bottom-up clustering followed by top-down declustering. However, this kind of algorithms does not reach low wirelength cost since they do not have the view of the global configuration at the earlier stages. Further, most of the multilevel floorplanning algorithms consider only variable-die floorplanning.

So, we need efficient and effective methodologies and tools to deal with large-scale design complexity. Chang [2] proposed a "V-shaped" multilevel framework, shown as Figure 1. This kind of framework adopts a two-stage technology consisting of top-down partitioning and bottom-up merging. The V-shaped multilevel framework remedies the drawback of traditional Λ-shaped one since it considers the global configuration at the earlier stage.

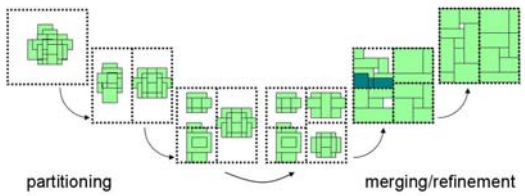

Figure 1: V-shaped framework

On the other hand, Kahng [5] pointed out that modern VLSI design is based on fixed-outline floorplan rather than variable-outline one. Otherwise, fixed-outline floorplanning (FOFP) enables multilevel hierarchical design [6], where aspect ratios and area of floorplans are usually imposed by higher level floorplanning and must be satisfied. Hence, a floorplan violating fixed-outline constraint is useless. It is easy for the existing flat fixed-outline floorplanners to get feasible solutions if area is the only objective. If wirelength is also considered, however, finding feasible solutions becomes difficult for them. IARFP [1] proposed a fixed-outline floorplanning method, which presented an elaborated new perturbation method for Simulated Annealing, named insertion after remove. In IAR perturbation method, removing a block is followed by finding a "good" position for the block instead of a random position. And the "good" position is found based on enumerating block position. And a new cost function for calculating the area cost is also proposed.

In this paper, we propose a new multilevel fixed-outline floorplanning algorithm for large-scale integrated circuit design by combining the IARFP algorithm within V-shaped multilevel framework, engaging in getting a better result with minimized wirelength, we called it "Multilevel IARFP".

Since the IARFP is based on Sequence Pair representation. Expanding to multilevel case, we propose a new representation, multilevel Sequence Pair, on which the Multilevel IARFP is based.

The Multilevel IARFP includes a three-stage structure: top-down partitioning, top-down floorplanning followed by bottom-up merging and refining. In the first stage, we use the min-cut partitioner hMetis [3] for top-down partitioning. In the second stage, we do the floorplanning in different levels from top to down by IARFP [1]. As long as building the sequence pairs, we prepare the data for next stage. In the third stage, we merge the sub-regions into bigger regions recursively until the top level. Simultaneously, we refine the merged floorplans with IAR perturbations [1] with different settings in Simulated Annealing. It will be discussed in detail later.

We propose this method aiming to deal with the large-scale integrated circuit floorplanning problem by pursuing minimized wirelength in short time consuming and no overlapping between blocks in fixed-outline chip dimension.

The organization of the paper is as follows. Section 2 gives the multilevel Sequence Pair representation we proposed. Section 3 shows the algorithm flow of Multilevel IARFP and detailed explaining. Section 4 gives the experimental results and compares the proposed Multilevel IARP with flat one. Section 5 draws some conclusions.

2. Multilevel Sequence Pair Representation

As we all know, the well-studied representation Sequence Pair [7] is used to handle floorplanning; and the IARFP is also based on it. Expanding to multilevel case, we propose a new representation method called multilevel Sequence Pair representation.

A Sequence Pair is a pair of sequences of n elements representing a list of n blocks. It imposes the relationship between each pair of blocks as follows.

$(<\cdots b_i \cdots b_j \cdots>,<\cdots b_i \cdots b_j \cdots>)$) b_i is left to b_j

$(<\cdots b_j \cdots b_i \cdots>,<\cdots b_i \cdots b_j \cdots>)$) b_i is above b_j

In multilevel case, we partition the blocks into small sub-regions, and then we do floorplannig to decide the region position in the chip level by level. So in multilevel Sequence Pair representation, the sub-regions keep order relationship in sequences. Corresponding to the floorplanning, we have different Sequence Pair in composing for different level. For the bottom level sequences, the elements express the blocks. In other levels, the elements express the regions. The blocks always keep the global position information.

The difference between bipartition and n-partition is: After bipartition, the position relationship of the regions can be directly known as there are only two sub-regions so that there is no need to do more floorplanning as for n-partition. So the bipartition equals to the n-partition followed by floorplannig. They have the same structure.

For the simple case, we explain our representation in the bipartition case. The number of regions will increase at a speed of double. And the number of blocks in each sub-region will decrease at a speed of half.

As shown in Figure 2, the chip is separated into different number of sub-regions in different level.

For the original level, the whole chip is a big region:

$(\Gamma_+, \Gamma_-)=(<r>,<r>)$

For the first level, we separate into two regions:

$(\Gamma_+, \Gamma_-)=(<r_1r_2>,<r_1r_2>)$

For the second level, chip is separated into four regions:

$(\Gamma_+, \Gamma_-)=(<r_{12}r_{11}r_{22}r_{21}>,<r_{11}r_{12}r_{21}r_{22}>)$

For the third level, chip is separated into eight regions:

$(\Gamma_+, \Gamma_-)=(<r_{121}r_{122}r_{112}r_{111}r_{222}r_{221}r_{211}r_{212}>,<r_{111}r_{112}r_{121}r_{122}r_{211}r_{212}r_{221}r_{222}>)$

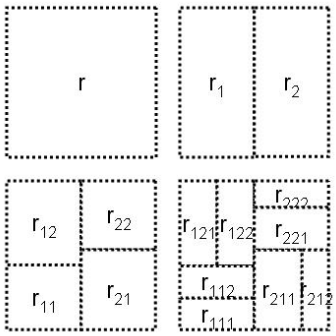

Figure 2: sub-regions in different level

We can recursively partition the circuit until the block number in each sub-region reaches which we want. The letters showed in the sequences are not blocks; they are also sequences expressing the block groups in the regions.

So far, the representation is for the simplest case, bipartition. For more common considering, we can define the number of the partition by setting the parameter of the partitioner. In other word, we can decide the number of sub-regions after every partition.

If we use 'n-partition', the region will be separated into n sub-regions every time. The number of total sub-regions will be increased at a speed of square of n. We take the advantage of this character to deal with the large-scale design by separate the big problem into small ones by pursuing the short time consuming.

3. Multilevel Expanding of IARFP

In Multilevel IARFP, we have a three-stage structure. The algorithm flow is shown like Figure 3. There are three main stages involved in our algorithm.

In the first stage, we finish the partition job. We use the min-cut partitioner hMetis [3] for top-down partitioning. The blocks are recursively separated into

sub-regions. The level number is decided depending on the scale of the floorplan problem. We do the partition level by level from top to down by setting the proper parameters for hMetis [3].

In the second stage, we use IARFP algorithm to get the floorplans for the sub-regions in different level from top to down. As already partitioned, the block number in regions is decreased at a speed of exponent. It is much faster than we directly do the floorplannig with all the blocks in the flat case.

In the third stage, we recursively merge the sub-regions into bigger regions until the top level. Simultaneously, we optimize the floorplanning with IAR perturbation [1] in already merged regions with different settings in Simulated Annealing.

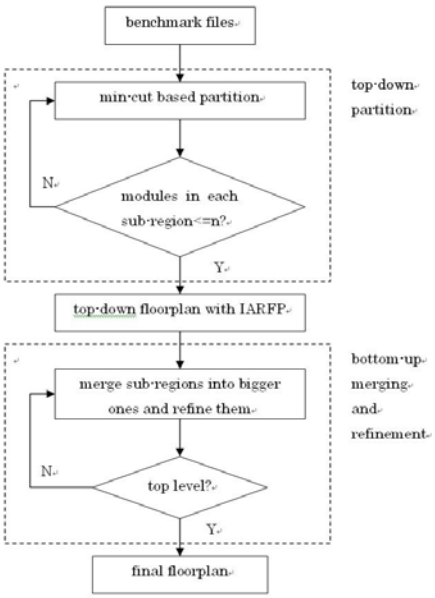

Figure 3: algorithm flow

As in flat case, the initial floorplan is built from a random sequence pair solution. Based on that, the perturbations were used to refine the floorplan. And if the scale is big, it will be much time consumed to optimize the result. But in our Multilevel IARFP algorithm, we do the floorplanning in little scale of blocks and prepare the initial floorplan for the refinement not in random solution but in merged sequences. The difficulty for refinement reduces greatly with a better initial solution.

A. top-down partitioning

We use the multilevel min-cut based partitioner hMetis [3] for the partitioning stage.

We do the partition recursively to assign the blocks into sub-regions. The partitioning goes on until the block number in sub-regions is smaller than a threshold in the bottom-level. In the bottom level the blocks buildup the regions, and in other levels the region is formed by small regions (with more blocks).

The partition is depending on their interconnect relationship. Until bottom, we just group the blocks together. The position information between regions is not built yet.

B. top-down floorplanning

In this stage, we do floorplanning not only in the bottom level but also in other levels to build the Sequence Pairs for the regions. We do the floorplanning with IARFP algorithm in the order from top to down. In this order, we take the advantage of the V-shaped multilevel framework, considering the global configuration at the earlier stage. It will be effective to the result in reducing the wirelength as we pursuing.

We do this stage as preparing the data for next merging and refinement level.

If the sub-regions come from the same father-region, the blocks must be adjacent to each other as a region in floorplanning. The corresponding elements must occur continuously in sequences. There are not intercrossing between regions at the same level. This point ensures the exactness of the position for next level.

The character ubiquitously exists not only in bipartition case but also in n-partition case.

Same as the explaining in multilevel Sequence Pair Representation, we show it in bipartition case.

For instance, sub-regions $r_{111}$and r_{112} in Figure 2 must be adjacent because they come from the same father region r_{11}. So do other adjacent regions in other levels. It ensures the favoring processing of merging and refining job.

The blocks always keep the correct global position information in the Sequence Pair though they are located in different sub-regions. For example, there are block a and b in the chip. In the first level, *a* and *b* are assigned into r_1 and r_2 respectively, and is r_1 left to r_2, so block *a* is left to block *b* in position; in the second level, r_1 is separated into $r_{11}r_{12}$ and r_2 is separated into $r_{21}r_{22}$, at the same time, *a* is assigned into r_{11} and *b* is assigned into r_{22}. Because r_{11} comes from r_1 and r_{22} comes from r_2 and r_1 is left to r_2. So r_{11} is left to r_{22}. Then *a* is still on the left side of *b* in the whole floorplan. And they also keep special order in SP.

C. bottom-up merging and refinement

We merge the small sub-regions together into bigger regions by link the sequences together (outspread level by level) from bottom to top and simultaneously refine the floorplan with IAR perturbations [1].

As the former bipartition example, we discuss the merging and refinement stage in simplest case.

We may first have the Sequence Pair like this at the beginning of this stage:

$(\Gamma_+,\Gamma_-)=(<<<r_{111}r_{112}><r_{121}r_{122}>><<r_{212}r_{211}><r_{221}r_{222}>>>,$
$<<<r_{121}r_{122}><r_{111}r_{112}>><<r_{222}r_{221}><r_{211}r_{212}>>>)$

The sequences are shown in omitting representation in the third level. The bottom-level elements like r_{111} are sub-regions with n blocks in them. The r_{111} in Γ_+ may be like <abc...>, the one in Γ_- may be like <acb...>. The r_{112} in Γ_+ may be like <def...>, the one in Γ_- may be like <fed...>.

At the same level, the parameters for IAR are set as same to refine the sub-regions. We merge the r_{111} and r_{112} together, back to be r_{11}. Then we refine it to r_{11}'.

In this processing, we can perturb the blocks both in r_{111} and r_{112} commutatively, namely "abcdef...".

After finish refining in this level, we get the SP:

$(\Gamma_+,\Gamma_-)=(<<r_{11}'r_{12}'><r_{21}'r_{22}'>>,<<r_{12}'r_{11}'><r_{22}'r_{21}'>>)$

Then we recursively do the merging and refinement with different setting. We get:

$(\Gamma_+,\Gamma_-)=(<r_1'r_2'>,<r_1'r_2'>)$

Finally,

$(\Gamma_+,\Gamma_-)=(<r'>,<r'>)$

We finish the merging and refinement stage level by level when there is only one region in the chip, and this is the final floorplan.

In the perturbations, we keep the region position; just refine the blocks position in the merged regions in the same level.

The refinement differs from that in other levels with different settings for Simulated Annealing such as iterative times and initial temperature. There are more blocks in one region in the higher level. So we need to decrease the perturbation complexity, for example we can limit iterative times, or decrease the initial temperature and so on.

4. Experimental Results

The proposed method has been implemented in C-language and run on an IBM workstation (3.2GHz, 3GB RAM) with Linux OS. The circuit sample prmy1, cb37 and GSRC benchmark n300 are used in the experiments. The wirelength is estimated using the half perimeter wirelength (HPWL) model.

$$Cost = \alpha(Costarea) + \beta(Costwirelength) \cdots (1)$$

$$Costarea = E_W + E_H \cdot \lambda + C_1 \cdot \max(E_W, E_H \cdot \lambda) + C_2 \cdot \max(W, H \cdot \lambda) \cdots\cdots\cdots\cdots\cdots\cdots (2)$$

$$Costwirelength = \sum_{nets} (x_{\max} - x_{\min}) + (y_{\max} - y_{\min}) \cdots\cdots\cdots\cdots\cdots (3)$$

The cost function was like formula (1), α and β are the weight of area and wirelength cost respectively. The cost is decided by area and wirelength. (Costarea and Costwirelength)

The E_w and E_h in formula (2) are respectively the excessive width and height of the floorplan, C_1 and C_2 are some user-defined constants. The values related to height of the chip are scaled by aspect ratio λ. The penalties are added into (1), too.

In formula (3), x_{max} is the maximum x-coordinates of all the pins involved in a net, and x_{min}, y_{max} and y_{min} have similar meaning.

We compare our Multilevel IARFP with the flat IARFP. As they are both fixed-outline, we do the experiments with the same parameter in white space, aspect ratio. We compare the wirelength cost and the corresponding time consuming.

The result from Table 1 shows we can get about 11% reductions in wirelength within about 55% run time.

Table 1: THE RESULTS OF FIXED-OUTLINE FLOORPLANNING. ALL BLOCKS ARE HARD. AVERAGE RESULTS ARE FOR 50 RUNS.

Circuits	IARFP		Multilevel IARFP	
	HPWL	Time(s)	HPWL	Time(s)
n300a	663504	31.628	555025	19.024
prmy1	124051	75.572	113025	34.901
cb37	4637623	371.491	4461068	234.066
	1.11	1.80	1.00	1.00

5. Conclusion

We proposed in this paper a multilevel fixed-outline floorplanning method to deal with floorplanning problem for the large-scale integrated circuit designs with a three-stage structure. Top-down partition stage separates the circuits into regions level by level. Top-down floorplannnig builds the sequence pair and gives initial result for next stage. The last stage merges the regions and refines the results from bottom to top.

References

[1] Song Chen and Takeshi Yoshimura, "Fixed-Outline Floorplanning: Block Position Enumeration and a New Method for Calculating Area Costs", CAD of Integr. Circuits and Syst., IEEE Trans. On, Accepted

[2] T.-C.Chen, Y.-W.Chang and S.-C.Lin; "A New Multilevel Framework for Large-Scale Interconect-Driven Floorplan- ning," CAD of Integr. Circuits and Syst., IEEE Trans. On, Accepted.

[3] hMetis, http://www-users.cs.umn.edu/~karypis /metis/hmetis/

[4] Parquet 4.5, http://vlsicad.eecs.umich.edu/BK /parquet

[5] A. B. Kahng, "Classical floorplanning harmful," in Proc. ACM ISPD, 2000, pp. 207–213.

[6] S. N. Adya and I. L. Markov, "Fixed-outline floorplanning: Enabling hierarchical design," Very Large Scale Integration (VLSI) systems, IEEE Trans. On., vol. 11, no. 6, pp. 1120–1135, 2003.

[7] H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani, "VLSI block placement based on rectangle-packing by sequence-pair," CAD of Integrated Circuits and Systems, IEEE Trans. on, vol. 15, no. 12, pp. 1518–1524, 1996.