1000-9825/2001/12(12)1752-08 ©2001 Journal of Software 软件学报 Vol.12, No.12

A Hierarchical Mixed Mode Placement Algorithm*

WU Wei-min, HONG Xian-long, CAI Yi-ci

(*Department of Computer Science and Technology, Tsinghua University, Beijing* 100084, *China*)

E mail: wuwm@mail.tsinghua.edu.cn

http://www.tsinghua.edu.cn

Received October 12, 2000; accepted June 15, 2001

Abstract: A hierarchical automatic placement algorithm for mixed mode placement problem is presented. The so called mixed mode is a combination of standard cell and macro block. The presented algorithm completes the placement in both block level and cell level. In block level, the random cells are firstly partitioned into soft blocks, then SP (sequence pair) based method is used to do block placement. In cell level, firstly, quadratic placement method is used to do inner placement within each soft block, then a placement improvement routine is done to the whole chip, and at last, a combined min-cut and enumeration based detailed placement procedure completes the final placement. The algorithm is tested on a set of circuits with different number of standard cells and macro blocks, and obtains satisfactory results.

Key words: mixed mode; placement; partitioning; sequence pair; quadratic placement

Mixed Mode is a combination of standard cell style and BBL (Building Block Layout) style, which has the advantages of both. In fact, it is an often-occurred situation to introduce directly well designed and tested, different shape and size function blocks to an existing standard cell circuit. However, current algorithms deal with macro blocks in very simple ways, i.e., either move them manually or use highly heuristic method. So far, neither enough research has been done, nor are the results satisfactory in automatic placement for mixed mode. The fundamental reason is that it is very hard to find an approach that can achieve good results in tolerable run time.

The earliest work about mixed mode placement can be found in 1978[1]. Feller et al. set some macro blocks, such as RAM, ROM to the corner of the chip. Of course, no good placement quality can be obtained by this simple method. Reingold's algorithm has more generality[2], which completes placement through three steps: (1) A one-dimensional placement is generated by min-cut method. (2) All placement components are classified into categories according to their height. (3) From lowest height category (i.e. standard cell), repeatedly merge the components to form higher category components. Though this highly heuristic method runs very fast, no good result can be obtained.

* Supported by the National Natural Science Foundation of China under Grant No. 69776027 (国家自然科学基金); the National Grand Fundamental Research 973 Program of China under Grant No. G1998030413 (国家重点基础研究发展规划 973 资助项目); the Skeleton Person Support Project of Tsinghua University of China (清华大学骨干人才支持计划) and is also a cooperative project with ACADIA Design Inc. of America

WU Wei-min was born in 1966. He received his Ph.D. degree from the Harbin Institute of Technology in 1995, and is now an associate professor of Computer Science and technology, Tsinghua University. His current research interests are all aspects of design automation for VLSI. **HONG Xian-long** was born in 1940. He is a professor of the Department of Computer Science and Technology, Tsinghua University. His research interests are VLSI layout algorithms and DA systems. **CAI Yi-ci** was born in 1960. She is an associate professor of the Department of Computer Science and Technology, Tsinghua University. Her main research area is VLSI layout design.

Upton *et al.* approach the problem in a floorplanning perspective[3]. They firstly partition the standard cells into a group of soft blocks, then a simulated annealing floorplaning is done to both soft blocks and original hard blocks, and at last, another simulated annealing placement is done within each soft block. Because of the high complexity of floorplaning, the relatively simple slicing structure has to be adopted. Back *et al.* present a combined bipartitioning and slicing method[4], whose feature is that each time the bipartitioning is done along the edge of current biggest block, until all blocks are involved. The algorithm runs very fast, but the placement quality is limited, because every macro block is put along the boundary of the corresponding sub-region created by bipartitioning.

Vygen proposes a flat placement algorithm[5], which views both standard cells and macro blocks as the same placement components. In order to resolve the overlaps between macro blocks, a linear programming problem is formulated with the moving quantity as the objective, which is solved by branch and bound method. Recently, Yu *et al.* also present a mixed mode placement algorithm, named MMP[6], which applies a quadratic placement procedure to both standard cells and macro blocks. During placement procedure, the overlaps between macro blocks are eliminated by a successive slicing method. Despite the high running speed, MMP's placement result may be severely impacted in the course of resolving the very large overlaps between macro blocks. In addition, MMP is inapplicable to non-slicing structures.

Generally, the algorithms for solving the mixed mode placement problem fall into two categories, flat[5,6] and hierarchical[3,4]. The strategy of flat algorithm is to view both standard cells and macro blocks as the same placement components, whose advantage is that low complexity placement algorithm, such as quadratic-based algorithm, can be utilized to do the placement with very high speed. However, large overlaps may occur for macro blocks because they connect too many nets. So the placement quality will deteriorate while resolving the overlaps. The strategy of hierarchical algorithm is to do the placement through block level and cell level, and the overlaps involving macro blocks are eliminated in block level. In both levels, the number of placement components reduces considerably. However, block level placement generally has higher complexity. If there is not a high-efficiency placement representation method, the block placement algorithm can't achieve an acceptable result within tolerable run time. Therefore, low complexity slicing structure is adopted so far, whose limitation is evident.

Since late 1990s, there have appear some important achievements in block placement research, some very effective non slicing representations such as SP (Sequence Pair)[7], BSG (Bounded Slicing Grid)[8] were presented, which we expect could serve to solve the mixed mode placement. Here we present a novel algorithm, named HMMP (Hierarchical Mixed Mode Placer), whose idea is: a SP-based simulated annealing optimization procedure is adopted to do block placement, then quadratic placement is done within blocks, and at last placement improvement and detailed placement are done to complete the whole placement.

1 Outline of the Algorithm

1.1 Problem definition

Given a set of placement components, including standard cells, PADs and macro blocks. The standard cells are little placement components with the same height but variable width. The PADs are fixed along the boundaries of the chip, which are in charge of transmitting signals from or to the outside world. Macro blocks are also built up by standard cells, but have fixed width and height, so they are also called hard blocks. A macro block can be moved as an entity, but the relative positions of its contained cells should not be changed. Throughout the paper, we name those standard cells that are outside of any macro blocks as random cells.

Under above conditions, the mixed mode placement problem is defined as: Put the standard cells and macro cells on the fixed-sized chip with no overlap, the standard cells are guaranteed to be on the legal rows and positions, while making the total length of all nets minimized.

1.2 Placement procedure

HMMP completes the placement through block level and cell level. Block level placement is done through random cell partitioning and block placement, and cell level placement is done through inner-block global placement, placement improvement and detailed placement.

In random cell partitioning step, all the random cells are partitioned to a set of groups, each forming a soft block. In block placement step, block placement is done to both hard blocks and soft blocks. In inner-block global placement step, global placement is done within each soft block, thus the approximate cell positions are determined. The hard blocks are fixed after block placement, so are their contained cells. In placement improvement step, a local optimization method is done to improve the placement quality further to the whole chip. In detailed placement step, all cells are assigned to legal positions of legal rows.

2 Block Level Placement

2.1 Random cell partitioning

There exist many approaches to circuit partitioning problem, such as clustering[9], eigenvector decomposition[10], network flow[11], group swapping[12,13], etc. In practice, KL[12] and FM[13], which have linear complexity, are the most commonly adopted bipartitioning algorithm. KL-FM based algorithms use the number of cut nets as the metric to evaluate partitioning quality. However, the two partitions generated by KL-FM based algorithms have comparable size, which may not match with real situation of the circuit. For a hierarchical partitioning, the violation between predefined partition sizes and the reasonable ones will accumulate and finally lead to a poor partitioning result. To solve the problem, Yen-Chuen Wei proposed a concept named ratio as the new metric to locate natural clusters in the circuit[14]. We adopted ratio-cut partitioning as the basic routine in the hierarchical partitioning.

Given a network $C=(V,N)$, where V is the set of random cells and N is the set of nets. If $A\cap A'=\varnothing$ and $A\cup A'=V$, then (A,A') is a partition to C. Let $N_{A,A'}$ represent the number of cut net of (A,A'), then the ratio of this cut is defined as $R_{A,A'}=N_{A,A'}/(|A|+|A'|)$. Ratio-cut is the cut that generates the minimum ratio among all cuts to the network.

The partitioning to all random cells is a top-down and hierarchical procedure. Suppose the network formed by all random cells is $C_r=(V_r,N_r)$, the upper bound of cell number of a cluster is Num, then the algorithm is shown in Fig. 1.

```
1. Build a binary tree named TREE, with $C_r$ as the root.
2. If each leaf node of TREE contains cells less than $Num$, STOP.
3. Randomly select a leaf node $C$ which have more cells than $Num$, bi-
   partition $C$ using ratio-cut algorithm, generate two sub-circuits, $C_1=$
   $(V_1,N_1)$, $C_2=(V_2,N_2)$, as the left and right sub-tree of $C$. Go to
   setp 2.
```

Fig. 1 Hierarchical partitioning algorithm

2.2 Block placement

SP is a very effective data representation method for non-slicing structure, which makes it possible to do block placement by random search based optimization algorithms like simulated annealing. SP is an ordered pair of Γ_+ and Γ_-, each of which is a sequence of names of given modules. When a placement is given, a corresponding SP can be obtained by so called positive loci and negative loci. Conversely, if a sequence pair is given, the horizontal and vertical relations between two modules are uniquely determined, thus a placement is obtained. The set of all SPs form the solution space, where the block placement algorithm searches for the optimal/suboptimal solution. For further description about SP, please refer to Ref. [7].

Simulated annealing search method is used. We define the objective function as

$$\Phi = A + W_L * Len + W_R * |AR - ChipX/ChipY|, \tag{1}$$

where A is the rectilinear area a placement occupies, Len is the total net length, AR is the desired aspect ratio, $ChipX$ and $ChipY$ are width and height of the chip, W_L and W_R are weighting factors.

At any state in the search procedure, we look for the next state in three ways:

(1) Randomly select two blocks in Γ_+ and interchange them.

(2) Randomly select two blocks in Γ_- and interchange them.

(3) Randomly select a soft block and change its aspect ratio.

Because a SP determined placement put the blocks from left-bottom corner of the chip, empty space may occur at right(top) if the width(height) of the occupied region of a placement is less than the width(height) of the chip. In order to fill the empty space, we magnify the coordinates of all the random cells in corresponding directions. For example, if the right boundary of a placement is X_R, then for a cell with x-coordinate x, the new coordinate is computed by

$$x_{new} = x \cdot ChipX/X_R. \tag{2}$$

After this step, all the soft blocks are pulled longer in horizontal and/or vertical directions. For a hard block, distortion is not allowed, so just compute the new coordinate of its center point, and move to this new point.

Though a simple bounding-box model is used to estimate net length, the placement speed is severely impacted for reason of the large number of nets. To alleviate the impact, we identify those nets whose net lengths are equal or approximately equal. Two nets have equal or approximately equal lengths if the following two conditions are satisfied:

(1) The two nets connected the same set of soft blocks.

(2) The two nets connected the same set of hard blocks. For each hard block they connected, the minimum and maximum terminal coordinates are approximately equal.

3 Cell Level Placement

3.1 Inner-block global placement

After block level placement, the boundaries of all soft blocks are determined. For an arbitrary soft block B_i, we should find its corresponding local circuit as follows:

(1) For a cell c_j, if it does not belong to B_i, but it drives some cells that belong to B_i, then c_j is a PI of the soft block.

(2) For a cell c_j, if it does not belong to B_i, but it is driven by a cell that belongs to B_i, then c_j is a PO of the soft block.

So, all found PIs, POs, together with standard cells within B_i form a local circuit. Global placement is done to the local circuit of each soft block.

3.1.1 Problem description

The quadratic placement is described as

$$\text{minimize } L(x,y) = \sum_{n\in N} L_n w_n = \sum_{n\in N} \sum_{i,j\in n} ((x_i - x_j)^2 + (y_i - y_j)^2), \tag{3}$$

$$\text{subject to } \quad \frac{1}{|S_j|}\sum_{i\in S_j} x_i = r_j^x, \quad \frac{1}{|S_j|}\sum_{i\in S_j} y_i = r_j^y \quad j = 1,\dots,m, \tag{4}$$

where (3) is the objective function, L_n is the estimated wire length of net n, w_n is the net weight. (4) are spread constraints which enable the cells to spread evenly on the chip. At ith level, the chip area is divided into 2^{2i} regions, while all the random cells are partitioned into 2^{2i} sets according their coordinates, thus each region can be assigned a set of cells. S_j is a region with $|S_j|$ as its contained number of cells and (r_j^x, r_j^y) as its center location.

3.1.2 Solution method

Eq. (3) can be written compactly in matrix notation as

$$L(x,y)=\frac{1}{2}(x^TQx+y^TQy)+c^Tx+d^Ty, \tag{5}$$

where vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ denote the coordinates of cells, c and d are constant vectors which are determined by PADs, Q is a positive definite matrix.

The quadratic problem can be solved by Largrangian relaxation method. The Largrangian equation can be written as

$$\max_{\lambda_x,\lambda_y\geq 0}\min_{x,y}\frac{1}{2}x^TQx+\frac{1}{2}y^TQy+c^Tx+b^Ty+\lambda_x^T(e^Tx-r_x)+\lambda_y^T(e^Ty-r_y), \tag{6}$$

where λ_x and λ_y are largrangian multipliers.

With fixed λ_x and λ_y, let $\partial L/\partial x=0, \partial L/\partial y=0$, the solution can be obtained as

$$x^{(k+1)}=-Q^{-1}[\lambda_x^{(k)}e^T+c],\ y^{(k+1)}=-Q^{-1}[\lambda_y^{(k)}e^T+d]. \tag{7}$$

Between iterations, λ_x and λ_y are modified by

$$\lambda_x^{(k+1)}=\max\{0,\lambda_x^{(k)}+t_x^{(k)}((e^Tx^{(k)}-r_x))\},\ \lambda_y^{(k+1)}=\max\{0,\lambda_y^{(k)}+t_y^{(k)}((e^Tx^{(k)}-r_y))\} \tag{8}$$

Therefore, the global placement procedure is an iterative interaction between two sub-routines (see Fig. 2):

(1) Global optimization: Solve Eq. (6) and modify λ_x and λ_y, making the solution to the optimum gradually.

(2) Cell dispersion: Add spread constraints in successively levels, making the cells spread more and more uniform.

```
1. Set initial partition level $H=0$. Set $\lambda_x=0, \lambda_y=0$.
2. Exert spread constraints to level $H$. Set iterative number $I=0$;
3. Solve Eq. (7) by conjugate-gradient method.
4. Modify $\lambda_x$ and $\lambda_y$.
5. Set $I=I+1$. if $I<I_{\max}$, go to step 3.
6. If $H<H_{\max}$, set $H=H+1$, go to Setp 2; else STOP.
   Where $I_{\max}$ is maximum iteration number, $H_{\max}$ is maximum partition level.
```

Fig. 2 Global placement procedure

3.2 Placement improvement

After global placement, every random cell is restricted in a soft block, which makes the cell positions partitioning dependent. Further more, there may be many cells overlapping on the boundaries between blocks. To reduce the overlaps and improve placement quality further, we ignore the boundary limit and move each random cell to its optimal position.

For a random cell, say c_i, if the set of nets it connects to is N_i, then the total net length is

$$L_{N_i}=\sum_{n\in N_i,i\neq j}w_n\cdot[(x_i-x_j)^2+(y_i-y_j)^2]. \tag{9}$$

Fix the set of cells c_i connects to, and let

$$\partial L_{N_i}/\partial x_i=0,\ \partial L_{N_i}/\partial y_i=0. \tag{10}$$

Then we get the optimal position of c_i

$$x_i=\Big(\sum_{n\in N_i,i\neq j}w_n\cdot x_j\Big)\Big/\sum_{n\in N_i,i\neq j}w_n\cdot(n-1),\ y_i=\Big(\sum_{n\in N_i,i\neq j}w_n\cdot y_j\Big)\Big/\sum_{n\in N_i,i\neq j}w_n\cdot(n-1). \tag{11}$$

However, the computed optimal position may overlap with hard blocks, so c_i should be repositioned again.

This can be done by firstly connect the current and optimal position with a line segment and divide the line segment evenly by some points, then from the optimal position to the original position, search the points one by one until a free point is found.

Placement improvement should be done to every random cell many times to get a better placement.

3.3 Detailed placement

Detailed placement is done through three steps: (1) An initial detailed placement is obtained by inheriting cell positions determined by placement improving procedure. (2) Row assignment refining and row evening are interlaced to reduce the wire length in y and x directions respectively, while rows evening and overlap removing are also done within interlaces. (3) The wire length is reduced further by cell permutation within rows.

Before detailed placement, areas occupied by hard blocks should be determined and filled with dummy cells to prevent other real cells from entering into the areas. Dummy cells are not allowed to move between rows, but the gaps between dummy cells can be used as placement area. For further information about detailed placement, please refer to Ref. [15].

4 Complexity Analysis

The complexity of ratio cut algorithm has proven to be $O(P)$, where P is total number of pins of all random cells. The complexity of SP-based block-level placement algorithm is $O(N^2)$, where N is the number of blocks.

For global placement, the time spent on each iterative step is proportional to $m+n+p$, where n, m, p represent the number of cells, nets and PADs respectively. In large circuit, p and m increase linearly with n, so the time complexity of each iterative step is $O(n)$. The iterative number is determined by solution accuracy, and we can give it an upper bound $n^{0.5}$. A sorting operation need to be done in cell dispersion procedure to the cells, corresponding time complexity is $(n \times \log n)$. If we set the maximum partition level to log n, total time complexity is $O(n^{2.5} \cdot \log^2 n)$.

The complexity of placement improvement is $O(n)$ and detailed placement is $O(n^2)$.

5 Experimental Results

HMMP is implemented in C and works on a SUN Enterprise450 workstation. The test circuits are provided by ARCADIA design systems Inc., whose characteristics are shown in Table 1. The experimental results about run time are shown in Table 2.

Table 1 Characteristics of test circuits

Circuits	#cells	#macro blocks	#nets	Block area /chip area(%)
Block 2	7094	2	10049	37
Block 4	6330	4	10049	42
Block 6	5996	6	10049	47
Block 8	5662	8	10049	50
Block 9	5895	9	10049	53
Block 10	5151	10	10049	57

Table 2 Experimental results (run times)

Circuits	Block level placement time (s)	Cell level placement time (s)	Total time (s)
Block 2	240.8	117.3	358.1
Block 4	257.6	105.4	363
Block 6	271.1	100.9	382
Block 8	307.1	98.6	405.7
Block 9	292.3	96.5	388.8

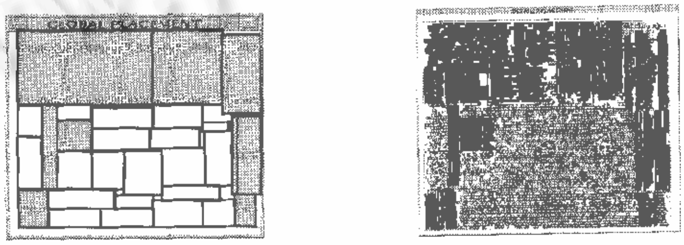
We can see in Table 2 that, although a simulated annealing method is used to do block placement, the time consumption is reasonable. The reason for that is we have adopted the very effective SP model as well as some means such as identifying approximate length nets, limiting the number of blocks for block placement and so on. Further more, it can be expected that the run time increase is limited for bigger circuits because we have effectively controlled the number of partitioned soft blocks. HMMP proves its feasibility and practicability by running very stable on all the test circuits.

In fact, in addition to reasonable run time, the main contribution of HMMP lies in the great improvement in placement quality. The comparisons with MMP are shown in Table 3. We can see that HMMP outperforms MMP on all test circuits. The improvement ratio ranges between 9.5% and 29.4%. HMMP's advantage in placement quality is obvious.

Table 3 Test results of HMMP and comparison to that of MMP (wire length)

Circuits	Wire length of MMP (μm)	Wire length of HMMP (μm)	Improve ratio(%)
Block 2	2 396 547	1 692 002	29.4
Block 4	2 253 763	1 934 645	14.2
Block 6	2 241 684	2 029 252	9.5
Block 8	2 814 446	2 022 614	28.1
Block 9	3 068 610	2 345 725	23.6
Block 10	2 484 459	2 130 945	14.2

At last, we give an illustrative show of the placement of block 9 in Fig. 3.



(a) After block level placement (b) After cell level placement

Fig. 3 Placement plot for block 9

6 Conclusion

Experimental results show that our hierarchical mixed mode placement algorithm is feasible, and can achieve very good placement quality. The most important advantage is that it is competent for non-slicing structure. Thus even if there may be many different-sized macro blocks, the placer can still give a non-overlap placement in block level. In the cell level, because quadratic placement method is adopted, the placement speed is very fast. In the future, we expect to extend it to solve timing-driven mixed mode placement problem.

References:

[1] Feller, A., Noto, A. A speed-oriented, fully-automatic layout program for random logic VLSI devices. In: Proceedings of the National Computer Conference. Anaheim, CA, 1978. 303~311.

[2] Reingold, E. M., Supowit, K. J. A hierarchy-driven amalgamation of standard and macro cells. IEEE Transactions on CAD, 1984,3(1):3~11.

[3] Upton, M., Samii, K., Sugiyama, S. Integrated Placement for mixed macro cell and standard cell designs. In: Proceedings of the 27th DAC. Orlando, FL, 1990. 32~35.

[4] Baek, Y. S., Cheon, B. Y., Kim, K. S., *et al.* Cell designer: an automatic placement and routing tool for the mixed design of macro and standard cells. IEICE Transactions on Fundamentals, 1992,75(2):224~231.

[5] Vygen, J. Algorithms for large-scale flat placement. In: Proceedings of the 34th DAC. Anaheim, CA, 1997. 746~751.

[6] Yu, Hong, Hong, Xian-long, Cai, Yi-ci. MMP: a novel placement algorithm for combined macro blocks and standard cells layout design. In: Proceedings of the IEEE ASP-DAC2000. 2000. 271~276.

[7] Murata, H., Fujiyoshi, K., Nakatake, S., *et al*. VLSI module placement based on rectangle-packing by the sequence pair. IEEE Transcations on Computer Aided Design, 1996,15(12):1518~1524.

[8] Nakatake, S., *et al*. Module Placement on BSG-Structure and IC Layout Application. In: Proceedings of the ICCAD'96. San Jose, CA: IEEE Computer Society Press, 1996. 484~49.

[9] Schuler, D.M., Ulrich, E.G. Clustering and linear placement. In: Proceedings of the 9th Design Automation Workshop. 1972. 50~56.

[10] Frankle, J., Karp, R.M. Circuit placement and cost bounds by eigenvector. In: Proceedings of the International Conference on Computer Aided Design. San Jose, CA: IEEE Computer Society Press, 1986. 414~417.

[11] Ford, L.R., Fulkerson, D.R. Flows in networks. Princeton, NJ: Princeton University Press, 1962.

[12] Kernighan, B.W., Lin, S. An efficient heuristic procedure for partitioning graphs. Bell System Technology Journal, 1970,49(2):291~307.

[13] Fiduccia, C.M., Mattheyses, R.M. A linear time heuristic for improving network partitions. In: Proceedings of the 19th Design Automation Conference. 1982. 175~181.

[14] Wei, Y.C., Cheng, C.K. Ratio cut partitioning for hierarchical designs. IEEE Transactions on CAD, 1991,10(7):911~921.

[15] Bo, Yao, *et al*. FAME: a fast detailed placement algorithm for standard-cell layout based on mixed mincut and enumeration. Chinese Journal of Semiconductors(English Edition), 2000,21(8):744~753.

分级的混合模式布局算法

吴为民, 洪先龙, 蔡懿慈

(清华大学 计算机科学与技术系,北京 100084)

摘要: 针对混合模式的布局问题提出一种分级的自动布局算法.所谓混合模式就是标准单元和宏模块相结合的布局模式.该算法在模块级和单元级两个层次上完成布局.在模块级上,首先将所有随机单元划分成若干软模块,然后采用基于序列对(sequence pair,简称SP)的方法完成模块布局;在单元级上,首先对每个软模块内部采用二次规划的布局算法进行布局,然后在全芯片范围内对布局进行改善,最后采用一种基于最小割(min-cut)和枚举相结合的快速详细布局算法完成最终布局.在一组标准单元数和宏模块数不同的电路上对该算法进行了验证,效果是令人满意的.

关键词: 混合模式;布局;划分;序列对;二次规划布局

中图法分类号: TP302 **文献标识码**: A