第 25 卷第 1 期　　半 导 体 学 报　　Vol. 25, No. 1
2004 年 1 月　　CHINESE JOURNAL OF SEMICONDUCTORS　　Jan., 2004

New Incremental Placement Algorithm Based on Integer Programming for Reducing Congestion*

Li Zhuoyuan, Wu Weimin and Hong Xianlong

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

Abstract: A new incremental placement algorithm C-ECOP for standard cell layout is presented to reduce routing congestion. It first estimates the routing congestion through a new routing model. Then, it formulates an integer linear programming (ILP) problem to determine cell flow direction and to avoid the conflictions between adjacent congestion areas. Experimental results show that the algorithm can considerably reduce routing congestion and preserve the performance of the initial placement with high speed.

Key words: congestion; standard cell; incremental placement
EEACC: 2570　　**CCACC**: 7410D
CLC number: TN405.97　　**Document code**: A　　**Article ID**: 0253-4177(2004)01-0030-08

1 Introduction

As VLSI technology advances, the system complexity continues to increase and physical design is getting more and more difficult. With the advent of over-the-cell routing, the goal of every place and route methodology has been to utilize all available active area to prevent spilling of routes into channels. It is the overflow of routes that account for an increase in area. "Congestion Aware" global routers avoid this by constructing a set of routing trees for all the nets in the design to match the global supply. However, it is unlikely to achieve optimality because the quality of a routing solution is largely determined by the input placement with congestion occurring where demand exceeds supply. Further heuristic method should be applied in placement to manage local congestion to enhance and to improve the latter route ability.

Traditional placement objectives involve reducing net-cut costs or minimizing wire length[1~2]. Because of its constructive nature, min-cut based strategies minimize the number of net crossings but fail to distribute them uniformly[3]. For the same reason, traditional placement schemes which are based mainly on wirelength minimization can not adequately account for congestion. Reducing net-cut and minimizing wirelength only help reduce the routing demand globally but do not prevent causing local routing congestion. How to estimate and reduce congestion in placement is not well studied. Congestion-driven placement based on multi-partitioning is proposed in Ref. [4]. It uses the actual congestion cost calculated from precomputed Steiner trees to minimize the congestion of the chip. However, the number of partitions is limited due to the excessive computational load.

* Project supported by National Natural Science Foundation of China (No. 60121120706), National Natural Science Foundation of USA (No. CCR-0096383), Hi-Tech Research & Development Program of China (No. 2002AA1Z1460) and National Foundation Research Program of China (No. G1998030403)

Li Zhuoyuan male, was born in 1980, PhD candidate. His research interests focus on VLSI layout design.

Received 25 March 2003, revised manuscript received 21 July 2003

Wang *et al.*[8~10] study the congestion problem during placement. It first points out that minimizing wirelength is indeed equal to minimizing the average routing demand and proposes a consistent routing model defined by demand/supply relationship. By giving an example it shows that the congestion cost could be locally inconsistent with the wirelength cost. Then it focuses on finding a good objective to effectively reduce the congestion in the final placement. Experimental results show that the congestion objective is very ill behaved because it is not sensitive to placement moves. So it adapts a post processing approach after placement to reduce congestion. But the demand/supply congestion model and bounding-box routing estimation are too simple and will affect the final result. Since congestion and wirelength are globally consistent, Jason *et al.*[11,12] consider to improve local congestion with incremental placement. However, how to minimize the wirelength changes caused by congestion reducing and maintain the metrics of the initial placement is very difficult.

In this paper, an incremental placement algorithm C-ECOP for improving local congestion is proposed. It first estimates the routing congestion through a new route model. Then it constructs an integer linear programming (ILP) to move cells to reduce congestion. Finally it adjusts the positions of cells to resolve overlap.

2 Estimation of congestion in placement

2.1 Congestion cost

Intuitively speaking, congestion in a layout means too many nets are routed in local regions, the routing demand exceeds routing supply in that region. For any congestion reducing algorithm, how to estimate routing congestion in the placement stage is the key issue.

The congestion cost could be defined based on the global bin concept. We partition a given chip into several rectangular regions, each of which is called a global bin. Assume there are r rows and c columns of global bins, we label the global bin at ith row and jth column as B_{ij} or bin(i,j). Figure 1 shows an example. In Fig. 1, we have $4\times4(=16)$ global bins. The congestion is "related" to the number of crossings between routed nets and global bin edges. Each global bin has two horizontal and two vertical edges surrounding it. We will refer a horizontal global edge as e_h and a vertical global edge as e_v.

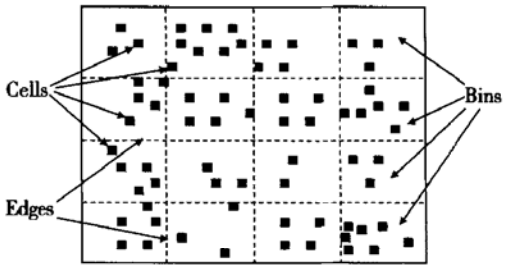

Fig. 1 Bin Structure

Given a detailed placement, all the cells and pads have fixed positions on the chip. We can use a "router" to route all the nets. The router can be a very simple global router or even a bounding box router. Obviously, the more accurate the router is, the more accurate the estimation at the placement stage will be.

For each global edge, there are routed nets crossing it. Therefore, for a global edge e, the routing demand of e, d_e, can be defined as the number of the nets crossing e. The routing supply of e, s_e, is known easily from the technology parameters. A global edge is congested if and only if routing demand d_e exceeds the routing supply s_e. The overflow is defined as follows:

$$\text{overflow}\ (e) = \begin{cases} 0, & d_e \leqslant s_e \\ d_e - s_e, & d_e > s_e \end{cases} \quad (1)$$

Then estimating the congestion of a global bin can be replaced by computing the total overflow of the global edges as Eq. (2) and the congestion cost function is defined as Eq. (3). Most of the algorithms for reducing congestion[6~10] estimate congestion like this.

$$\mathrm{con}(B_{ij}) = \sum_{k=1}^{4} \mathrm{overflow}(e_k) \qquad (2)$$

$$\mathrm{cost}_c = \sum_{i=1}^{n} \sum_{j=1}^{n} \mathrm{con}(B_{ij}) \qquad (3)$$

Table 1 gives the result of the congestion estimation on a circuit CNT100 with the method. The chip is partitioned into 4×4, 5×5, 6×6, and 7×7 global bins. It is shown that the total number of congest areas (TNC), the total demand (TD) changed due to different partition. The total routing demand increases when the partition number increases. It denotes that the demand/supply model is ill behaved. The estimation result depends on the partition scheme.

Table 1 Estimation of congestion on CNT100

Grids	Supply (H/V)[1)]	TNC (H/V)	TD (H/V)	MD (H/V)	MO[2)] (H/V)
4×4	19/12	0/1	261/198	18/13	0/1
5×5	15/10	4/4	269/192	20/14	5/4
6×6	13/8	2/8	322/226	17/11	4/3
7×7	11/7	4/6	391/251	18/8	7/1

1) H: Horizontal V: Vertical; 2) MO: Max overflow

On the other hand, when the partition number increases, the global edge becomes shorter. It seems that the routing demand crossing each global edge should decrease. From the table we can see that the max routing demand (MD) even increases when the partition number increases from 4×4 to 5×5.

From the table we can see that the total demand increases when partition amount increases. The reason is that the nets inside a larger global bin may become cross a global edge when the chip be partitioned with smaller bins. One method to deal with this is to partition the chip to global bins which are small enough. But the router used at the placement stage is somewhat "simple" compared to the real global and detailed router. It could only give out the general situation of routing. If the global bins are too small, there will be inconstancy between congestion estimation and real congested regions. The result will be improper for improving route ability.

Another method is to compute the routing demand inside a larger global bin with a proper routing model. It has a global view on congestion areas on the chip and will not lead inconstancy with following routing. So the vertical and horizontal congestion estimation of a global bin in our algorithm is defined as follows

$$\begin{aligned} \mathrm{con}_v(B_{ij}) &= \omega_1 r_v(B_{ij}) + \omega_2 \sum_{k=1}^{2} \mathrm{overflow}_v(e_k) \\ \mathrm{con}_h(B_{ij}) &= \omega_1 r_h(B_{ij}) + \omega_2 \sum_{k=1}^{2} \mathrm{overflow}_h(e_k) \end{aligned} \qquad (4)$$

where r_v and r_h are the vertical and horizontal routing demand inside B_{ij}. It can be obtained by the routing model described in 2.2. ω_1 and ω_2 are the weights.

2.2 Routing estimation model

When we are performing congestion reducing, we need to estimate congestion of placement incrementally. A global router is needed here. Obviously, the more accurate the router is, the more accurate the estimation at the placement stage will be. Routing with a real global router will provide an accurate congestion estimation. But it will be very time consuming and could not be applied in incremental placement algorithm. Routing with a simple routing model such as the bounding box model will be very fast. But the bounding box model may be far different with the characteristics of the detailed router so that it causes a bad estimation. Wang *et al*. verify the bounding box model in Ref.[8] and prove that it does not correlate with the real router and could not be applied. So it is critical that the algorithm for this application is accurate while maintaining computational efficiency.

A new star model proposed in Ref.[6] is used here. It first computes and adjusts the coordinate of the net center. The vertical and horizontal possible route paths connecting each cell to the center are on the edges of a rectangle whose two vertex locate on the cell and the center. Route possibility on each path is 0.5. Then all the route possibility on the same path is added up and could not exceed 1 as shown in Fig. 2. The experimental results show that the new star model is very close to the real routing in practice[6].

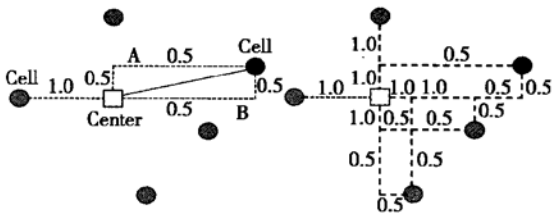

Fig. 2　Routing estimation with new star model

This model has an obvious drawback. The possible route path between a cell and the net center does not necessarily locate on the rectangle edges. It may cross any bin edges inside the rectangle as shown in Fig. 3. A probability model is proposed to correct this error.

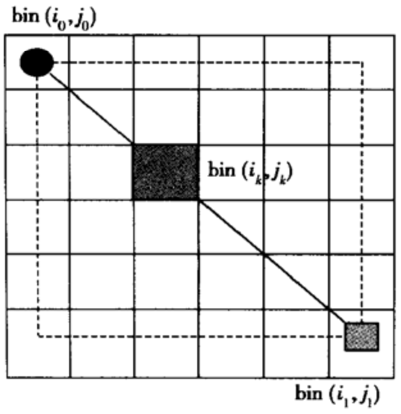

Fig. 3　Route probability

If cell C_k is in bin(i_0,j_0), the net center is in bin(i_1,j_1), all the possible routes crossing the left edge of bin(i_k,j_k) are those from bin(i_0,j_0) to bin(i_k-1,j_k), crossing the edge between bin(i_k-1,j_k) and bin(i_k,j_k), then from bin(i_k,j_k) to bin (i_1,j_1). It is in the symmetric form on the right edge of bin(i_k,j_k). So the route possibility crossing the left and right edges of bin(i_k,j_k) could be denoted as

$$p_l(k) = \begin{cases} w_{kl}, & \begin{matrix} i_0 \leqslant i_k \leqslant i_1 \\ j_0 \leqslant j_k \leqslant j_1 \end{matrix} \\ 0, & \text{otherwise} \end{cases}$$
$$p_r(k) = \begin{cases} w_{kr}, & \begin{matrix} i_0 \leqslant i_k \leqslant i_1 \\ j_0 \leqslant j_k \leqslant j_1 \end{matrix} \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

$$w_{kl} = C_{i_k-i_0-1+j_k-j_0}^{i_k-i_0-1} \times C_{i_1-i_k+j_1-j_k}^{i_1-i_k} / C_{i_1-i_0+j_1-j_0}^{i_1-i_0} \quad (6)$$

$$w_{kr} = C_{i_k-i_0+j_k-j_0}^{i_k-i_0} \times C_{i_1-i_k-1+j_1-j_k}^{i_1-i_k-1} / C_{i_1-i_0+j_1-j_0}^{i_1-i_0} \quad (7)$$

And the possibility of crossing the top and bottom edges, which are denoted as $p_t(k)$ and $p_b(k)$, are computed in the symmetric form. The vertical and horizontal routing demand inside a global bin can be easily known from this approach. The running time of congestion estimation on some circuits through this approach is listed in Table 2. It is shown that this approach is so fast that it could be used in our algorithm.

Table 2　Running time on routing estimation

Circuit	# Cell	# Net	Grid	Running time/s
Ibm01	12, 036	11, 507	30×30	0. 57
Ibm02	19, 062	18, 429	40×40	1. 95
Ibm03	21, 924	21, 621	50×50	2. 05
Ibm04	26, 346	26, 163	50×50	1. 93
Ibm05	28, 146	28, 446	60×60	4. 48

3　Reducing congestion through ILP

3. 1　Overview

Generally, minimizing congestion and minimizing wirelength conflict each other in local regions[8]. Reducing of the congestion means to sacrifice the wirelength. Incremental placement algorithm should achieve some trade-off between reducing congestion and preserving the metrics, e. g. wirelength, of the initial placement.

The design flow is shown in Fig. 4. The flow tendency of each cell is computed through a force directed technique. Then the cells could move due to their flow tendencies to reduce congestion. An integer linear programming is formulated to deal with the conflicts between multiple congested areas. After that a post process is carried out to place the moved cells and resolve overlap. The iteration of the ECO flow stops when the congestion result is acceptable.

3. 2　Computation of cell flow tendency

Based on routing estimation, we can identify the congested bins on the chip. Each global bin is

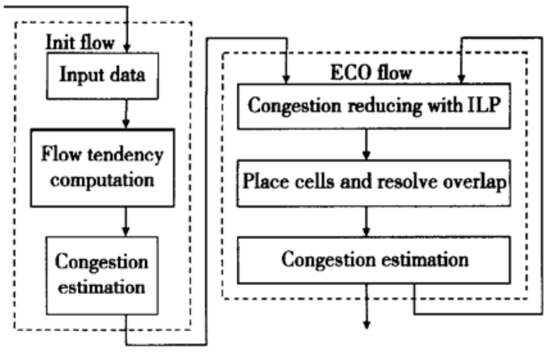
Init flow
Input data
Flow tendency computation
Congestion estimation
ECO flow
Congestion reducing with ILP
Place cells and resolve overlap
Congestion estimation

Fig. 4 C-ECOP algorithm

considered to be a congested bin when the bin congestion cost defined by Eq. (4) is greater than a certain threshold value or at least one of its global edge is overflowed. Cells in congested bins should move outside to decrease routing demand in this region and achieve more routing resource. Which cells could be moved out so that the perturbation to the initial placement could be minimized is the key issue. We compute the flow tendency of each cell to deal with this.

The horizontal flow tendency of a cell C_k in bin(i,j) is computed as follows

$$x\text{flow}_l(c_k) = \sum_{\text{net}_{k1}} p_l(k) - \sum_{\text{net}_{k2}} p_r(k) - \sum_{\text{net}_{k3}} 1 \quad (8)$$

$$x\text{flow}_r(c_k) = \sum_{\text{net}_{k2}} p_r(k) - \sum_{\text{net}_{k1}} p_l(k) - \sum_{\text{net}_{k3}} 1 \quad (9)$$

where net_{k1} are the nets whose center are in bins on the left, net_{k2} are the nets whose center are in bins on the right, net_{k3} are the nets whose center are in bin bin(i,j). $p_l(k)$ and $p_r(k)$ are defined in Eq. (5).

$x\text{flow}_l$ denotes the gain of crossing nets decreased on the left edge of bin(i,j) when moving C_k from bin(i,j) to bin$(i-1,j)$. If it is positive, moving C_k to the left bin will lead crossing nets on the edge decreased. It means the gain of overflow decrease on the edge. $x\text{flow}_r$ denotes the gain when moving cell to the right bin. And it is in the symmetric form for the vertical flow tendencies. These parameters decide the movable directions of cells.

3.3 Reducing congestion based on integer programming

For a congested bin, the cells inside should move out to reduce the nets inside and achieve more free space for routing. If bin(i,j) is vertically congested, cells in it should move along the horizontal direction as shown in Fig. 5.

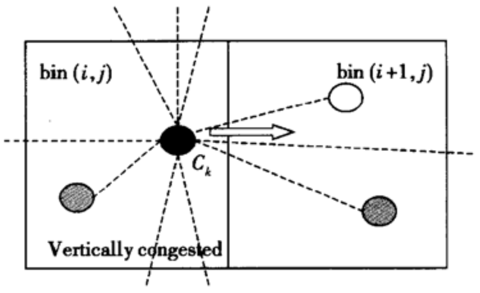

Fig. 5 Move cell to reduce congestion

When C_k is moved into bin$(i+1,j)$, the vertical route in bin(i,j) will decrease by

$$\text{gain}_v(c_k) = \sum_{\text{net}_{kt}} p_t(k) + \sum_{\text{net}_{kb}} p_b(k) \quad (10)$$

where net_{kt} are nets connected to C_k and their centers are in the top bins. net_{kb} are nets whose centers are in the bottom bins. $p_t(k)$ and $p_b(k)$ are the possibility of crossing the top and bottom edges of bin(i,j) estimated by the route model. Whether C_k should move to the left or the right bin is according to $x\text{flow}_l(C_k)$ and $x\text{flow}_r(C_k)$. It assures that the horizontal congestion will not increase when reducing vertical congestion.

Note that the vertical route demand in bin$(i+1,j)$ will increase after C_k moves into it. This may cause unexpected congested regions. Furthermore, if bin$(i+1,j)$ is already vertically congested, moving C_k will lead severe congestion in it. An arbitrative mechanism is needed to deal with the conflicts between the reducing congestion among multiple congested regions. An integer programming problem is constructed to resolve it.

As mentioned above, only the cells with one or more positive flow tendency could be moved. Experimental results show that the conditions are too strict. For some circuits there are few cells could move to reduce congestion. So we relax the conditions here. For a congested bin bin(i,j), horizontal-movable cells should satisfy the following conditions

$$x\text{flow}_l(c_k) > t_h \quad \text{or} \quad x\text{flow}_r(c_k) > t_h \qquad (11)$$

where t_h is a negative threshold. It means that we allow a little horizontal congestion increase when reducing vertical congestion. And the condition for vertical-movable cells is as follows

$$x\text{flow}_t(c_k) > t_v \quad \text{or} \quad x\text{flow}_b(c_k) > t_v \qquad (12)$$

It is obvious that only one equation in (8) and (9) could be positive. If t_h equals 0, only one of the two inequations in (11) may be satisfied, so for a movable cell C_k, its flow orientation is certain. When we relax the conditions, both of the two inequations in (11) may be satisfied. Then we randomly prescribe the moving orientation of C_k. Later we can see that this will assure high efficiency when solving the ILP problem.

The integer linear programming problem is defined as

$$\text{minimize} \qquad C_{\max}$$
$$\text{s.t.}\begin{cases} \text{con}_h(b_{ij}) - \sum\limits_{C_k} m_{kij} x_k \text{gain}_h(C_k) \leqslant C_{\max} \\ \text{con}_v(b_{ij}) - \sum\limits_{C_k} m_{kij} y_k \text{gain}_v(C_k) \leqslant C_{\max} \end{cases}$$
$$i,j = 1, 2, \cdots, N$$
$$x_k + y_k \leqslant 1$$
$$x_k, y_k \in \{0, 1\} \qquad k = 1, 2, \cdots, M$$
$$m_{kij} \in \{-1, 0, 1\}$$

where $C_{\max}$ is the maximum congestion degree over all the global bin.

x_k could equal 1 only when Eq. (11) is satisfied.

y_k could equal 1 only when Eq. (12) is satisfied.

$x_k + y_k \leqslant 1$ means that C_k could not move vertically and horizontally at the same time.

m_{kij} denote which C_k should be included in the inequations. For a movable C_k in bin(i_k, j_k), because its vertical and horizontal flow orientation is certain, the bin it could move into is only. So the parameter could be defined as

$$m_{kij} = \begin{cases} 1, & i = i_k \text{ and } j = j_k \\ -1, & \text{bin}(i,j) \text{ is the bin } C_k \text{ moves to} \\ 0, & \text{otherwise} \end{cases} \qquad (13)$$

That m_{kij} equals 1 means route demand in bin(i_k, j_k) will reduce after moving C_k out. It equals -1 means route demand in the destination bin will increase for the same amount. And the perturbation to the route probability in other bins is ignored. Experimental results show that it slightly affects the final results but considerably saves the running time.

Then for some bins, the number of its related cells is limited. Actually, it is only the movable cells inside the bin and in the four adjacent bins that may be included in the inequations. The ILP problem can be optimally solved. The problem solution determines the total number and destination bins of moved cells. A post processing is then carried out to place the cells and resolve overlap inside each bin.

3.4 Post processing

After the process of reducing congestion, some cells are redistributed among different global bins. These cells should be placed without overlap. An efficient algorithm is needed to adjust the positions of cells with the minimal perturbation to the initial placement. We use the W-ECOP algorithm[5] here to accomplish the process. For a cell to be placed, it is inserted into the adjacent row and an optimal scheme to rearrange the cells in the row is found. If free space in the row can not accept the cell, a shifting path searching process is carried on to assure cells restrict in their neighboring area so that the performance of the circuit will be preserved.

4 Experimental results

The algorithm has been implemented in C. All the experiments are done on a Sun E450 workstation with 4Gb memory. To show the effectiveness and utility of our algorithm, a part of the experimental circuits are chosen from IBM-PLACE benchmark[13] placed with a wirelength-driven placer, Dragon[14]. Other set of circuits are from industry (Synplicity Company). We compare the global routing results (overflow and wirelength) for the design before and after incremental placement.

Table 3 shows the results on the circuits from industry. As one can see, the approach of reducing congestion considerably reduces the total overflow. It has a 50 percent cut down in average. And the total wirelength increased less than 0.1 percent compared to the initial placement. Some circuits even have a decrease in wirelength after placement. This indicates that the wirelength is not sacrificed much due to the reducing of congestion. It is owed to the wirelength optimization approach in W-ECOP algorithm.

Table 4 shows the results on the IBM-PLACE benchmarks placed with Dragon. Dragon has done wirelength and routability optimization by combining powerful hypergraph partitioning package with simulated annealing technique. It could achieve better results than most of the placement tools in industry[14]. From the results we can see that the total overflow can be reduced continually through our method. And the optimization in wirelength is preserved. The algorithm is much faster on the benchmarks than on the industry circuits. The short amount of running time shows that our method can scale well for large circuits.

Table 3 Experimental results on industry circuits

Circuit	# Cell	Grid	V/H Cap1)	Overflow			Wirelength			Runtime /s
				BIP2)	AIP2)	Dec./%	W/mm	W'/mm	Inc./%	
Cnt100	760	5×5	15/15	18	12	- 33.33	46376	46961	+ 1.26	2.01
Cnt1000	8150	20×20	18/18	54	13	- 75.93	1684936	1688686	+ 0.22	19.53
M32_my	7150	20×20	22/22	252	132	- 47.62	962017	969448	+ 0.77	2.72
Gfsm300	4154	20×20	15/15	52	26	- 50.0	727760	725522	- 0.31	4.71
Sony_1	24847	40×40	57/57	442	419	- 5.2	9940513	9928071	- 0.13	193.09
Toshiba	16444	30×30	61/61	387	227	- 41.34	4264968	4259167	- 0.14	63.87

1) V/H Cap: The vertical/horizontal capacity of each grid; 2) BIP/AIP: overflow before and after incremental placement

Table 4 Experimental results compared with Dragon on IBM-PLACE benchmarks

Circuit	# Cell	Grid	V/H Cap	Overflow			Wirelength			Runtime /s
				BIP	AIP	Dec./%	W/mm	W'/mm	Inc./%	
Ibm01	12036	20×20	27/45	156	132	- 15.38	5171658	5118393	- 1.03	2.94
Ibm02	19062	25×25	54/85	597	552	- 7.53	16403624	16422055	+ 0.11	12.95
Ibm03	21924	30×30	36/49	413	363	- 12.10	14193033	14210763	+ 0.12	7.08
Ibm04	26346	35×35	38/52	337	275	- 18.40	16057004	16093988	+ 0.23	6.91
Ibm05	28146	40×40	67/110	546	415	- 23.99	42195069	42230608	+ 0.08	13.49

5 Conclusion

A new incremental placement algorithm for congestion alleviation is presented in this paper. The proposed algorithm automatically evaluates the routing congestion of a detail placement with a fast and accurate routing estimation model. Congestion areas on the chip are relieved through the cell moving. An integer linear programming (ILP) problem is formulated to resolve conflicts among multiple congested areas and avoid causing unexpected congested areas. After that an efficient algorithm for resolving overlap is used to ensure perturbing the initial placement the least. Experimental results demonstrate the effectiveness of the new approach.

References

[1] Dunlop A E, Kernighan B W. A procedure for placement of standard cell VLSI circuits. IEEE Trans Comput Aided Des, 1985, 4: 92

[2] Eisenmann H, Johannes F M. Generic global placement and floorplanning. In: Proc Design Automation Conf, 1998: 269

[3] Saab Y. A fast clustering-based Min-cut placement algorithm with simulated-annealing performance. VLSI Design: Int J

Custom-Chip Design, Simulation, Testing, 1996, 5(1): 37

[4] Meixner G, Lauther U. Congestion-driven placement using a new multi-partitioning heuristic. In: Proc Int Conf Computer-Aided Design, 1990: 332

[5] Li Zhuoyuan, Wu Weimin, Hong Xianlong, et al. Incremental placement algorithm for standard-cell layout. Chinese Journal of Semiconductors, 2002, 23(12): 1338

[6] Hou Wenting, Yu Hong, Hong Xianlong, et al. A new congestion-driven placement algorithm based on cell inflation. Chinese Journal of Semiconductors, 2001, 22(3): 275

[7] Yang Xiaojian, Kastner R, Sarrafzadeh M. Congestion reduction during placement based on integer programming. IEEE/ACM International Conference on Computer Aided Design, 2001: 573

[8] Wang Maogang, Yang Xiaojian, Sarrafzadeh M. Congestion minimization during placement. IEEE Trans Comput-Aided Des Integr Circuits Syst, 2000, 19(10): 1140

[9] Wang Maogang, Sarrafzadeh M. Modeling and minimization of routing congestion. Proceedings of the ASP-DAC 2000, 2000: 185

[10] Wang Maogang, Sarrafzadeh M. On the behavior of congestion minimization during placement. International Symposium on Physical Design, 1990: 145

[11] Coudert O, Cong J, Malik S, et al. Incremental CAD. IEEE/ACM International Conference on Computer Aided Design, 2000: 236

[12] Cong J, Sarrafzadeh M. Incremental physical design. Proc International Symposium on Physical Design, San Diego, California, 2000: 84

[13] http: // er. cs. ucla. edu/benchmarks/ibm-place/

[14] http: // er. cs. ucla. edu/Dragon/

基于整数规划的优化拥挤度的增量式布局算法*

李卓远　吴为民　洪先龙

(清华大学计算机科学与技术系, 北京　100084)

摘要: 提出了一种降低走线拥挤的标准单元增量式布局算法 C-ECOP. 首先通过一种新型的布线模型来估计芯片上的走线情况, 然后构造一个整数线性规划问题来解决可能出现的相邻拥挤区域冲突问题. 实验结果表明该算法能够有效地降低走线拥挤, 保证初始布局的质量, 并且具有很高的效率.

关键词: 拥挤度; 标准单元; 增量式布局

EEACC: 2570　　**CCACC**: 7410D

中图分类号: TN405.97　　**文献标识码**: A　　**文章编号**: 0253-4177(2004)01-0030-08

* 国家自然科学基金(批准号: 60121120706), 美国国家自然科学基金(批准号: CCR-0096383), 国家高技术研究发展计划(批准号: 2002AA1Z1460) 和国家重点基础发展规划(批准号: G1998030403) 资助项目

李卓远　男, 1980 年出生, 博士研究生, 研究方向是超大规模集成电路布图设计.

2003-03-25 收到, 2003-07-21 定稿