第 23 卷第 1 期　　半 导 体 学 报　　Vol. 23, No. 1
2002 年 1 月　　CHINESE JOURNAL OF SEMICONDUCTORS　　Jan., 2002

一种新的无网格拆线重布算法*

谢 民　蔡懿慈　洪先龙

(清华大学计算机科学与技术系, 北京　100084)

摘要: 结合无网格布线的特点, 提出一种新的无网格拆线重布算法. 该算法显式地表示并动态更新线网所属区域的拥挤程度. 在拆线重布进行待布线网的路径搜索时, 每个扩展节点中增加拆除线网周边的拥挤权重, 从而将待布线网的路径搜索过程和拆除线网的选择过程统一起来, 有效地提高了被拆除线网重新布通的可能性. 该算法利用改进的二叉区间树有效组织中间数据, 降低计算的复杂度. 实验结果表明, 该算法能有效消除布线顺序对布线结果的影响, 提高布通率, 且算法运行速度较快.

关键词: 无网格区域布线; 拆线重布; 区间树

EEACC: 7410D; 5120

中图分类号: TN402　　**文献标识码**: A　　**文章编号**: 0253-4177(2002)01-0107-06

1 引言

布线过程是 VLSI 设计中的重要步骤, 其首要目标是提高布通率. 有网格布线算法将布线区域简化为网格点, 虽然能够提高运行速度, 但因为边界扩展的不合理性降低布线的可利用面积, 从而影响布通率. 出于性能考虑, 连线的宽度随线网的不同而变化, 这种变线宽的情况是有网格布线难以处理的. 另外在多层工艺下, 线网的引脚往往落在布线区域内部, 而不仅仅在布线区域的四周, 使用传统的通道布线器[1~3]无法解决此类问题. 因此, 无网格区域布线算法日益受到重视. 布线过程通常包括初始布线和拆线重布两部分. 前者按照一定顺序逐个安排线网, 由于布线序的影响, 初始布线一般不可能达到百分之百的布通率. 这时需要调用拆线重布过程, 通过选择拆除已布线网, 调整布线资源分配, 消除布线顺序对最终结果的影响, 提高布通率.

已有的拆线重布算法大部分属于有网格布线算法. 第一类算法[4,5]基于布线冲突检测. 这类算法先不考虑资源限制, 使用迷宫或线探索算法独立地为每个线网搜索最佳布线位置, 然后再考虑布线资源重复分配引起的冲突. 由于有网格布线算法事先将布线区域划分成网格点, 因此检测冲突十分方便. 算法将按照某种策略选择并拆除一些线网, 重新安排布线. 其中文献[4]根据线网的冲突数、通孔数量及线长来评估布线结果的质量, 为了避免陷入局部最优, 使用模拟进化的方法选择待拆除的线网, 然后用迷宫算法重新搜索. 文献[5]通过 Lagrange 松弛策略, 对布线问题迭代求解, 并将拆线重布转化为整数规划问题, 使用启发式方法求解, 力求使被拆除线网的线段总数最少. 第二类算法[6~8]在搜索过程中就考虑资源限制, 生成的布线方案始终满足设计规则的要求. 文献[6]在布线过程中直接引入拆线重布, 其弱修改过程推挤已有线网以便腾出空间, 而强修改过程通过搜索最小代价的网格点以确定待拆除的线网, 同时利用历史记录来保证算法中止. 文献[7, 8]是基于图的方法, 前者将搜索拆除线网的过程转化为搜索连接图上两个端点的最短路径, 研究的重点是如何选择线网, 使需要拆除的线网数量最少. 后者搜索的是无法布通线网上两个端点间的分割集合, 这个算法大量使用迷宫搜索过程, 因此是基于多处理单元硬件的迷宫算法.

无网格拆线重布算法中, 文献[9]基于冲突检测, 但由于检测冲突的次数多, 时间复杂度高, 最坏情况下达到 $n!$; 文献[10]在拆线尝试中考虑设计规

* 高等学校骨干教师资助计划及 973 国家基础研究项目(No. G1998030403)

2001-03-24 收到, 2001-04-17 定稿

则约束,使用递归过程加禁忌搜索的方法,每个递归过程对应一个禁止拆除的线网集合,保证算法有效终止.但这种方法在比较拥挤的布线区域内进行递归时结果受到线网顺序的影响,且由于对所有线网同等对待,选择过程带有一定的盲目性,无助于提高被拆除线网重新被布通的可能性.

本文基于角勾链[11~13]的数据结构,提出一种新的无网格拆线重布算法,通过显式表示线网所属区域拥挤程度,结合网块扩展过程,将线网路径搜索和选择拆除线网统一起来.并采用改进二叉区间树加速算法的中间比较过程.实验结果表明,该算法能有效消除布线顺序对布线结果的影响,提高被拆除线网重新布通的可能性,且算法运行速度较快.以下本文是这样组织的:第2节简介角勾链数据结构及初始布线;第3节给出拆线重布算法,包括使用的二叉区间树结构和加速策略;最后是实验结果及结论.

2 角勾链数据结构和初始布线算法简介

2.1 角勾链数据结构

本算法采用改进的角勾链型[11~13]数据结构,能显式表示版图上的所有区域,包括已占用和未占用区域,并用指针在各个模块的左下角和右上角处把它们连接起来.其指针设置如图1所示.左下角有两个指针bl和lb,分别指向左边和下边的模块,右上

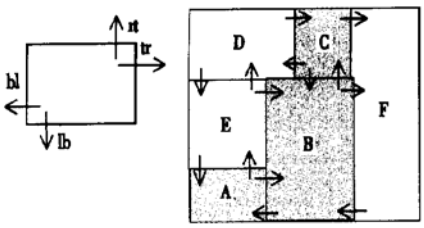

图1 角勾链结构

Fig.1 Corner stitch structure

角有两个指针tr和rt,分别指向右边和上边的模块.由于保留大量的邻域信息,角勾链的邻域操作效率较高.和布线系统相对应,水平布线层的角勾链结构沿着每个实模块的水平边向左右方向作水平分割线,直到碰到一个实模块的垂直边或者版图的垂直边界.这些水平割线将空区域分割成矩形,这种分割方法使得空矩形在水平方向最长,从而保证空区域的分割唯一.竖直层结构类似,只是将水平方向的分割线改为竖直方向.

2.2 基于块扩展的初始布线算法

初始布线采用块扩展算法[14~16],它结合了线探索和迷宫搜索的特点,路径搜索的效率很高.初始布线和拆线重布采用的扩展模式有所不同,在此作简要介绍.块扩展算法的相关定义有:

网块是满足连线规则要求,沿布线层方向伸展直到遇到障碍的矩形区域.

网块参考点是每个网块内引导搜索过程及扩展代价计算的坐标点.

连接路径是搜索过程得到的一个水平层和竖直层网块的交替序列.

块扩展有两种不同的扩展模式:

模式1:从当前网块的覆盖区域在相邻层朝两侧扩展搜索.搜索的网块只限于空块.这种模式将使用通孔,称这种模式为换层扩展.

模式2:沿当前网块的扩展方向在当前层上继续向前扩展搜索.搜索的网块包括空块以及被其他线网占用的实块.因此,生成网块可能和原有线网段有部分重叠,这就意味着要拆除已有线网才可能安排新的连线.称这种模式为同层同向扩展.

以两端线网连接为例,块扩展的算法描述如下:

```
Proc TileExpansion
    Initialize s and t
    Openlist= {s}
    WHILE Openlist≠∅ DO
        Get MinNode ∈ Openlist with minimum cost
        IF MinNode reached t THEN
            BackTrace starting from MinNode
        ELSE
            TileSet= EnumerateTile(MinNode)
            For each Tile ∈ TileSet DO
              Generate DescentNode for Tile
              Openlist= Openlist ∪ {DescentNode}
            NEXT
        ENDIF
    NEXT
```

网块代价分为两部分,分别是每个网块参考点到源点及到目标点的Manhattan距离.初始布线只调用换层扩展,在到达目标点后,从当前网块回溯到源点的网块序列就是一条连接路径.回溯过程将根

据网块的上下连接关系及周围布线情况确定实际布线位置.

由于初始布线算法的串行性, 布线顺序对布线结果有一定影响, 通常无法完全布通所有线网. 需要提交拆线重布作进一步的处理.

3　拆线重布算法

3.1　算法概述

基于冲突检测的算法在所有线网路径搜索完毕后需要相互比较是否存在冲突, 对以块表示的布线路径执行这一操作的时间复杂度很高, 且很大部分的搜索被浪费了. 因此, 本算法在为待布线网搜索路径时就考虑布线资源限制, 过滤不必要的搜索和比较. 算法流程如图 2 所示. 拆线重布也使用块扩展算法确定待布线网的路径, 但增加了同层同向扩展. 这种扩展在碰到已有连线障碍时将继续前进, 生成的网块和原有线网有一定重叠. 同层同向扩展对应的路径是换层扩展无法表示的, 因此搜索的解空间将大大超过初始布线.

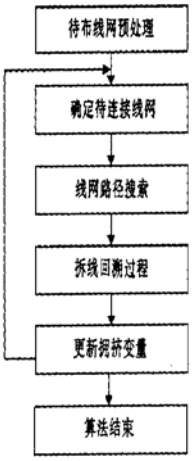

图 2　算法流程

Fig. 2　Algorithm outline

拆线重布的关键是合理选择需要拆除的线网. 选择待拆除的线网不仅要使当前线网能顺利布通, 还要使被拆除线网能较为容易地重新布通. 这就要求拆线重布过程能安排线网避开拥挤区域. 文献[10]在递归过程中人为指定禁忌拆除的线网集合, 虽然通过失败线网集合可以保留历史信息为后续递归过程利用, 但由于无法有效地区分不同拥挤度区域内的线网, 递归过程带有一定的盲目性.

由于网块扩展算法得到的连接路径包含丰富的布线区域局部信息, 可以利用它来指导待拆除线网的选择过程. 因此, 本文提出一种新的无网格拆线重布算法, 在为待布线网搜索路径时同时考虑线网所处区域的拥挤程度, 将拆除线网的选择和路径搜索统一起来. 具体做法是用变量显式表示线网所处区域的拥挤状况. 相应地, 块扩展算法中每个网块的扩展代价除了原来的线长代价外, 将新增加拥挤代价:

$$\text{cost3}=\sum_{i\in \text{OSet}}\text{RipupEffort}(\text{Net}_i)$$

$$\text{OSet}=\{i \mid \text{Net}_i,\ \text{overlaps the path from source to current node}\}$$

其中　Net_i 是和当前连接路径相重叠的已布线网; RipupEffort 根据线网所属区域的拥挤变量及重叠范围得到. 从而使拆线重布的搜索过程不仅取决于连接路径的长度, 还将受到路径上待拆线网所属区域拥挤度的影响. 得到的路径将是连线长度和拥挤程度权衡的折衷. 由于拥挤权重高的网块扩展代价将高于其他网块, 且块扩展算法每次扩展的网块都是代价最小的, 结果是促使搜索过程优先考虑拆除非拥挤区域的线网. 同样, 在回溯过程中也用线网的拥挤变量来指导连线.

为及时反映布线区域的实际情况, 拥挤变量将根据拆线重布的过程动态地更新. 每次线网布线导致拆除其他线网后, 相关的区域内拥挤变量将按比例地提升, 以反映区域内各线网彼此竞争资源的程度. 这一做法的另一结果是影响后续线网扩展过程中网块的拥挤权重, 防止后续线网立即拆除新完成的线网.

3.2　改进的二叉区间树结构

搜索过程得到的每个网块都对应一个可行矩形布线区域. 不同的连接路径, 可能到达相同区域; 特别是在拆线重布阶段引入同层同向扩展后, 部分搜索到的网块将彼此重叠. 如两个网块区域相同, 为避免重复搜索, 算法将只保留扩展代价小的网块. 即使如此, 由于判断两个区域的包含关系需要 4 次比较, 每个新产生的网块如果逐个和 Openlist 中的网块作比较, 复杂度为 $O(n)$, n 为 Openlist 的网块数. 当 n 增大时, 判断时间将显著增加. 因此, 需要有效的数据结构来组织网块, 减少不必要的判断, 降低算法的运行时间.

用于管理版图数据的结构除了角勾链外,还有Bin 结构、四叉树结构等. 角勾链适合查找区域邻接关系,但进行上述矩形包含判断时效率很低. Bin 结构依赖于划分的网格尺寸,不适当的网格大小会造成空间浪费及重复比较. 四叉树的层次式结构能有效区分位于不同区域的矩形,但由于每个四叉树节点存放的是和相应区域的垂直或水平平分线相交的图形,有可能造成重复;另外,维护该结构需要的额外操作较多. 本拆线重布算法采用改进二叉区间树分层管理网块. 和四叉树比较,该结构保留了层次式的组织方式,从而屏蔽了不必要的判断,而且由于使用统一划分方向,避免了空间重复和额外的维护. 下面介绍二叉区间树的构造方法.

构造二叉区间树时,首先沿布线方向对布线区域进行递归二划分. 图 3(a)、(b) 表示一个垂直层布线区域的划分过程. 矩形 pqrs 是垂直层的布线区域. A、B、C、D、E 是块扩展得到的网块. 线段 cd 是 $pqrs$ 的垂直二分线. 第一次划分使用 cd 将整个布线区域分为左右两部分,得到矩形子区域 $pqdc$ 和 $cdrs$,如图 3(a) 所示. 第二次划分的对象是区域 $pqdc$ 和 $cdrs$,经各自的二分线 ab 和 ef 分别划分得到四个矩形区域 $pqba$、$abdc$、$cdfe$、$efrs$,如图 3(b) 所示. 以后每次都将上次划分得到的矩形区域等分,直到子区域的 X 方向跨度小于某个阈值为止,使用的划分线始终平行于布线层方向,即垂直方向. 由于上述划分得到的所有矩形区域 Y 方向跨度都和布线区域的 Y 方向跨度相同,因此一个 X 方向区间即可表示一个区域;水平布线层区域的划分方法类似,只是划分线改为水平方向,从而所得的区间都是 Y 方向的. 伴随上述划分过程,为每个生成区间加上指针表示划分的层次关系就得到二叉区间树.

每个二叉区间树节点对应一个区间[start, end],区间方向垂直于相应布线层的方向,对图 3(a)、(b) 中的垂直布线区域,区间方向为 X 方向. 节点中存放的是完全被该区间包含,且和该区间的二分线相交的所有网块. 每个非叶子节点有左右两个子节点,分别递归对应于子区间[start, (start+end)/2],和[(start+end)/2, end]. 图 3(c) 表示和图 3(a)、(b) 划分相对应的二叉区间树. 其中,根节点 1 对应整个布线区间[a, r]、节点 2、3 分别对应区间[q, d]和[d, r]. 由于网块 A、E 被区间[q, r]完全包含,且和 cd 相交,因此,存放在二叉区间树根点 1 中. 同理,网块 D、B 被区间[q, r]和[d, r]完全包含,但只和[d, r]的二分线 ef 相交,因此存放在节点 3 中. 这样,根据新扩展的网块对应矩形的位置,需要和它作包含关系判断的实际只有二叉区间树上从根

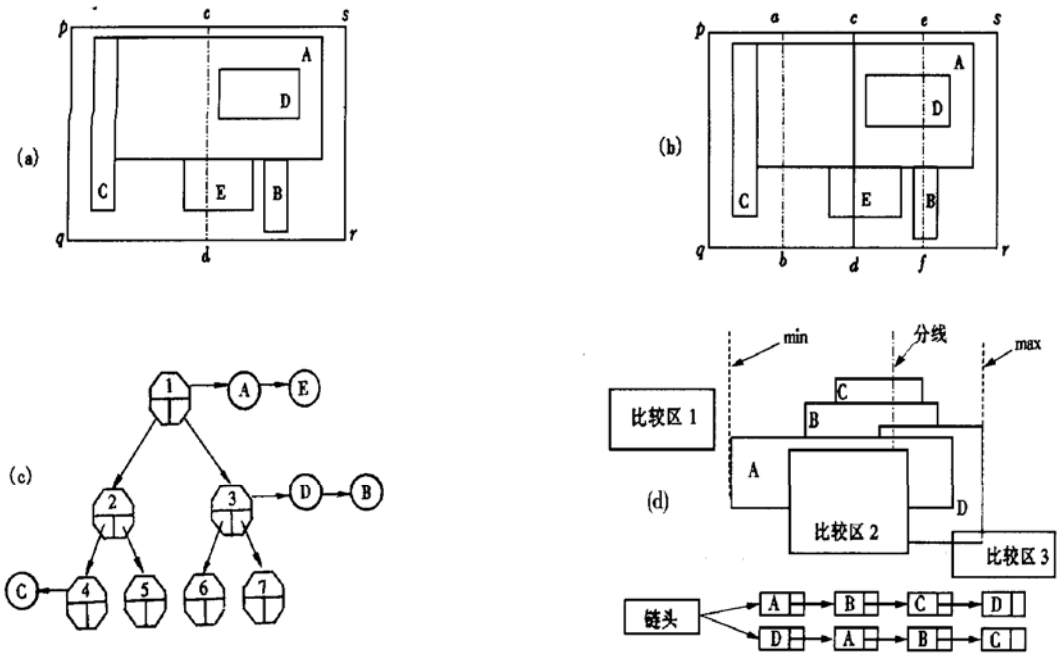

图 3 (a) 第一次划分;(b) 第二次划分;(c) 相应的区间树;(d) 节点中的链表

Fig. 3 (a) First partition; (b) Second partition; (c) Corresponding interval tree; (d) Linked list in tree node

到该网块所属树节点的路径上所有节点中包含的网块. 每个网块的比较复杂度由原来的 $O(n)$ 降低为 $O(\log(w)+s)$. 其中, s 是需要和新扩展网块作比较的网块数, w 为布线区域的宽度. 由于 $s \ll n$, 将显著提高算法运行速度.

3.3　加速策略

由于原始障碍以及布线新产生的线网段的存在, 网块扩展过程将产生许多跨在大区间二分线上的细长网块. 因此, 在布线过程中, 一个区间中所有网块的实际覆盖区间和树节点对应的区间有较大的差异. 为进一步减少不必要的比较, 本算法对原有二叉区间树作了改进, 为每个树节点显式记录中网块集合的最左边界和最右边界. 如在图 3(d) 中, 根据节点的左右边界信息, 比较区域 1、比较区域 3 不必和树节点中的四个网块做包含关系比较. 只有比较区域 2 需要逐个判断. 和二叉区间树比较, 为每个树节点加入边界信息所增加的复杂度为 $O(t)$, t 为一个树节点中的网块数目. 实验表明, 这一组织方式比通常的二叉区间树减少比较次数约 50%.

4　实验结果

本算法使用 C 语言在 SUN Enterprise e450 上实现并调试通过.

测试使用 MCNC 的测试用例 C2、C5、C7, 拆线前使用文献[13] 中的布线算法进行初始布线, 结果如表 1 所示. 同一例子的总体布线网格划分有所差异, 不同的划分总体布线网格的方法将导致初始布通率以及最终线长的不同. 其中, c2_chip 是 C2 不经过总体布线直接进行详细布线的结果, 运行时间明显增加, 但最终结果的线长要比经过划分网格后进行布线所得结果要减少约 15%. 图 4 是其中的一个布线结果图. 从实验结果看, 本拆线重布算法能够有效消除布线顺序对布线结果的影响, 使最终的布通率有明显的提高.

表 1　测试结果

Table 1　Test results

测试用例	布线区域数	线网数	运行时间/s	拆线前布通率/%	拆线后布通率/%	总线长
C2	9×11	745	197	92.8	99.9	5.424×10^5
C2_9	3×3	745	245	97.3	99.9	5.166×10^5
C5	16×18	1764	400	89.1	100	1.380×10^6
C5_30	5×6	1764	443	96.8	100	1.307×10^6
C7	16×18	2356	609	89.7	99.9	2.152×10^6
C7_30	16×18	2356	679	94.2	99.9	2.039×10^6
C2_chip	1×1	745	618	91.9	100	4.629×10^5

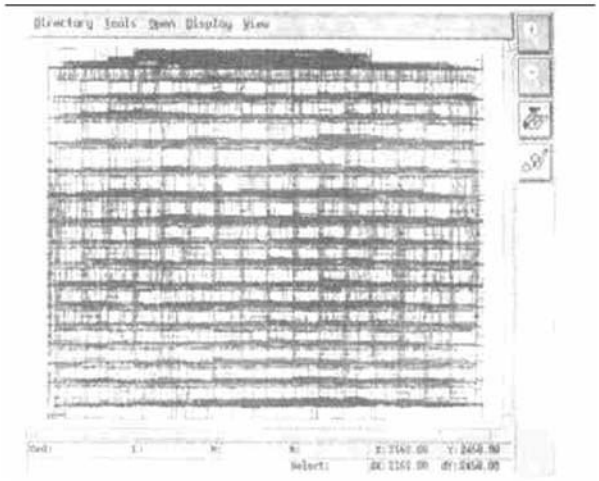

图 4　测试实例 C7

Fig. 4　Test example C7

参考文献

[1] Huang Pujiang, Hong Xianlong, Wang Erqian. A new Over-the-Cell channel router. Chinese Journal of Semiconductors, 1992, 13(8): 482[黄埔江, 洪先龙, 王尔乾. 一个新的 Over-the-Cell 通道布线算法. 半导体学报, 1992, 13(8): 482]

[2] Ying Changsheng, Hong Xianlong, Wang Erqian. DRAFT——An efficient area router based on global analysis. Chinese Journal of Semiconductors, 1988, 9(6): 596[应昌胜, 洪先龙, 王尔乾. 一个基于整体优化分析的区域布线算法 DRAFT. 半导体学报, 1988, 9(6): 596]

[3] Ying Changsheng, Hong Xianlong. Over-the-cell routing in 2-metal-layer gate array layout. Chinese Journal of Semiconductors, 1992, 13(10): 629[应昌胜, 洪先龙. 双金属层门阵列跨单元行布线问题与算法. 半导体学报, 1992, 13(10): 629]

[4] Lin Y L, Hsu Y C, Tsai F S. SILK: A simulated evolution router. IEEE Transactions on Computer-Aided Design, 1989, 8(4): 1108

[5] Rosenberg E. A new iterative supply/demand router with rip-up capability for printed circuits boards. Proc 24th Acm/IEEE Design Automation Conference, 1987, 721

[6] Shen H, Sangiovanni-Vincentelli A. Mighty: A rip-up and reroute detailed router. in Degist of Tech Papers, ICCAD, 1986, 2

[7] Raith M., Bartholomeus M. A new hypergraph based rip-up and reroute strategy. Proc 28th Acm/IEEE Design Automation Conference, 1991, 54

[8] Suzuke K, Matsunga Y, Tachibana M, et al. A hardware maze

router with application to interactive rip-up and reroute. IEEE Transactions on Computer-Aided Design, 1986, 5(4): 466

[9] Kurma N, Roger C Y, Acken John M. A gridless multilayer area router. in: Proc 4th Great Lakes Symposiun on VLSI Design Automation of High Performance VLSI System, 1994: 158

[10] Liu L, Tseng H, Sechen C. Chip-level area routing. ISPD '98 Monterey CA USA, 197

[11] Ousterhout J. Corner stitching: A data-structuring technique for VLSI layout tools. IEEE Transactions on Computer-Aided Design, 1984, 3(1): 87

[12] Zhang Yan, Wang Baohua, Cai Yici, et al. Area routing oriented hierarchical corner stitching with partial Bins. Proc Asp Dac, 2000, 2000: 105

[13] Zhang Yan. Gridless area routing datastructure management and BUS routing algorithm. Master Thesis of Tsighua University, 2000[张雁. 无网格区域布线的数据管理和 BUS 线布线算法研究. 清华大学硕士论文, 2000]

[14] Margarino A, Romano A, De Gloria A, et al. A tile-expansion router. IEEE Transactions on Computer-Aided Design, 1987, CAD-6(4): 507

[15] Tsai C, Chen S, Feng W. An H-V alternating router. IEEE Transactions on Computer-Aided Design, 1992, 11(8): 976

[16] Lu Jing. Multilayer gridless area routing algorithm. Master Thesis of Institute of Semiconductors, The Chinese Academy of Sciences, 2000, 6[卢璟. 多层无网格区域布线算法研究. 中国科学院半导体所硕士论文, 2000, 6]

A New Gridless Ripup and Reroute Algorithm*

Xie Min, Cai Yici and Hong Xianlong

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

Abstract: In accordance with the properties of gridless area routing, a new ripup and reroute algorithm is proposed. This algorithm records and dynamicly updates the congestion in the area of each net explicitly. Congestion weight along the explored path is calculated in each expansion node during the ripup and reroute process. Thus, the procedure of path exploration is unified with the selection procedure of nets to be ripped. The possibility that the ripped nets can be successfully rerouted is improved by this method. Besides, an enhanced interval tree is used to manage and organize the intermediate data so as to reduce the computation complexity. Test results indicate that this algorithm can effectively eliminate the influence of net order on routing. The complete ratio is improved after the ripup and reroute process and the operating speed of this algorithm is faster.

Key words: gridless area routing; ripup and reroute; interval tree

EEACC: 7410D; 5120

Article ID: 0253-4177(2002)01-0107-06

* Project supported by Foundation for University Key Teacher by Ministry of Education and 973 National Fundamental Funds of China(No. G1998030403)

Received 24 March 2001, revised manuscript received 17 April 2001