先进制程下基于多策略融合的时延优化层分配算法

刘耿耿[1,2)], 江列湫[1)], 李泽鹏[1)], 吴若昕[3)], 徐宁[4)*]

[1)] (福州大学计算机与大数据学院 福州 350116)
[2)] (中国科学院计算机体系结构国家重点实验室 北京 100190)
[3)] (福州大学梅努斯国际工程学院 福州 350116)
[4)] (武汉理工大学信息工程学院 武汉 430070)
(xuning@whut.edu.cn)

摘 要: 引入层分配算法能够有效地优化物理设计过程中的时延和通孔数等指标, 提高电路性能. 为此, 提出一种同时考虑非默认规则线和耦合效应的基于多策略融合的时延优化层分配算法. 首先针对现有工作对线网差异性考虑不细致的问题, 提出线网异化策略; 然后针对网格边拥塞情况评估不够合理的问题, 提出段分级策略; 再对非法线网进行拆线重绕时更注重考虑拥塞约束而导致时延过高的问题, 提出重绕调整策略; 最后提出多目标驱动排序策略, 对布线顺序不够合理的问题设计多种新颖的确定布线顺序的方法. 在 2.60 GHz CPU 和 64 GB 内存的 Linux 环境下, 使用 DAC12 基准电路得到的实验结果表明, 在保证不发生溢出的情况下, 所提算法能够有效地优化时延和通孔数.

关键词: 层分配; 时延; 拥塞; 通孔; 拆线重绕
中图法分类号: TP391.41 **DOI:** 10.3724/SP.J.1089.2024.19948

Delay-Driven Layer Assignment Algorithm Based on Integrating Multi-Strategy for Advanced Technology Nodes

Liu Genggeng[1,2)], Jiang Lieqiu[1)], Li Zepeng[1)], Wu Ruoxin[3)], and Xu Ning[4)*]

[1)] (*College of Computer and Data Science, Fuzhou University, Fuzhou* 350116)
[2)] (*State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100190)
[3)] (*College of Maynooth International Engineering, Fuzhou University, Fuzhou* 350116)
[4)] (*School of Information Engineering, Wuhan University of Technology, Wuhan* 430070)

Abstract: By introducing layer assignment algorithm, it can effectively optimize multiple important indicators such as delay and via counts in physical design, and then improve the chip performance. Therefore, a delay optimization layer assignment algorithm integrating multiple strategies is proposed, which considers both non-default-rule wire and coupling effect. First, aiming at the problem that the difference of nets is not carefully considered in the previous work, the differentiated net strategy is proposed. Second, in order to solve the problem of unreasonable evaluation of grid edge congestion in previous work, the segment classifying strategy is proposed. Third, aiming at the problem that the existing work pays more attention to consider congestion constraint when reassigning illegal nets, which leads to high delay of routing results, an effective reassignment and adjustment strategy is proposed. Finally, aiming at the problem of unreasonable routing order in the previous work, a variety of novel methods to determine the routing order are designed,

收稿日期: 2022-09-03; 修回日期: 2022-11-01. **基金项目**: 国家自然科学基金(62372109); 福建省杰出青年科学基金(2023J06017).
刘耿耿(1988—), 男, 博士, 教授, 博士生导师, CCF 高级会员, 主要研究方向为微流控生物芯片、VLSI 自动化设计; **江列湫**(1999—), 男, 硕士研究生, CCF 学生会员, 主要研究方向为 VLSI 自动化设计; **李泽鹏**(1998—), 男, 硕士研究生, 主要研究方向为 VLSI 自动化设计; **吴若昕**(2001—), 女, 在校学生, 主要研究方向为 VLSI 自动化设计; **徐宁**(1968—), 男, 博士, 教授, 博士生导师, CCF 高级会员, 论文通信作者, 主要研究方向为电子设计自动化、计算机软件与系统.

and then a multi-objective driven sorting strategy is proposed. The algorithm runs in the Linux environment with a 2.60 GHz CPU and 64 GB of RAM. The validation experiments use the DAC12 circuit, and the experimental results show that the proposed algorithm can effectively optimize both delay and via counts without overflow.

Key words: layer assignment; delay; congestion; via; rip-up and reassignment

随着超大规模集成电路技术的发展, 其集成度越来越高, 线网数量不断增加, 时延也随之增加. 目前, 时延优化已成为物理设计的关键问题. 为了有效地优化时延, 给出高质量电路物理设计方案, 非默认规则(non-default-rule, NDR)线这一先进制程技术引起了各界的广泛关注[1]. NDR 线分为宽线和平行线 2 种. 传统的默认规则线虽然只占用 1 个布线轨道, 但时延较高. 平行线采用并联连接方式, 占用 2 个布线轨道, 但其总电阻更小, 且时延与电阻成正比关系, 所以在同一布线层, 平行线的时延小于默认规则线的时延. 另外, 宽线线宽更大, 导线间距更大, 占用 3 个布线轨道; 但由于线宽更大, 导致宽线电阻更小, 因此在同一布线层, 宽线的时延小于默认规则线的时延.

本文将层分配布线模型设定为 9 层布线层结构, 并将其分为 3 个梯队: 第 1 梯队为 1~4 层, 第 2 梯队为 5~7 层, 第 3 梯队为 8~9 层; 本文算法的应用空间不局限于此结构, 可以解决任意结构的层分配问题. 不同梯队的金属层的默认线宽不同, 同一梯队内不同布线层的默认线宽相同, 3 个梯队中, 布线层默认线宽分别为 $1w$, $2w$, $4w$. 同时, 受制造工艺技术的限制, 第 1 梯队可使用平行线达到优化时延的目的, 第 2 梯队和第 3 梯队可使用宽线达到优化时延的目的.

当前, 布线问题吸引了越来越多研究人员的关注. 鉴于布线的高复杂性, 将布线过程分为总体布线和详细布线. 总体布线要求在一定时间内得到总体布线方案, 其追求的是效率. 近年来, 已有众多性能优越的布线器, 如 NCTU-GR2.0[2], BonnRoute[3], FGR[4], FastRoute4.0[5], Archer[6], CUGR[7]和 MiniDeviation[8]等. 高效的总体布线方案将为详细布线提供指导, 进而得到具体布线方案. 文献[9]使用 A*算法得到了高质量的详细布线方案; 文献[10]提出减少详细布线中的信号串扰的算法, 以此得到有效的详细布线方案.

因为某些布线因素在总体布线时被忽略, 所以总体布线及详细布线两者的布线方案之间存在着不匹配之处, 通过设计高效的布线方案, 能够缓解两者之间的不匹配问题. 近年来, 人们越来越关注层分配问题, 其具有高复杂性. 时序感知的层分配问题已经被证明是 NP 完全问题[11-13]. 文献[14-15]在多层布线模型下考虑了通孔和导线的电容和电阻对时延的影响, 文献[16]寻找了优化线网时延的方法; 但上述层分配工作大多未考虑 NDR 线这一先进制造工艺[11-16]. 基于此, 文献[17]提出了考虑 NDR 线的时延驱动层分配算法. 现有工作存在以下不足: (1) 对线网差异性考虑不充分, 线网权重被赋予统一的数值, 未能自适应地调整权重; (2) 未充分地考虑网格边上经过的导线具体数量, 对网格边拥塞情况评估不够合理; (3) 对非法线网进行拆线重绕时注重考虑拥塞约束, 导致时延过高; (4) 未充分地考虑各阶段不同的布线目标设计合适的布线顺序, 不能充分地发挥低时延布线层的时延优化潜力. 因此, 现有工作在各项层分配指标上仍有较大的优化空间.

本文引入 NDR 线这一先进制程, 同时考虑耦合效应这一影响时延评估准确性的因素, 提出一种高效的先进制程下基于多策略融合的时延优化层分配算法, 包括线网异化策略(differentiated net strategy, DNS)、段分级策略(segment classifying strategy, SCS)、重绕调整策略(reassignment and adjustment strategy, RAAS)和多目标驱动排序策略(multi-objective driven sorting strategy, MDSS). 本文算法融合上述各项策略, 选择更好的布线方案以及更好的布线顺序, 在保证不发生溢出的情况下, 有效地优化了线网时延以及和通孔数, 获得了高质量的布线方案.

1 问题描述

1.1 层分配概述

层分配采用如图 1 所示的多层布线模型. 图 1a 中, m_i 表示第 i 层布线层. 布线层上可能分布着若干障碍物, 障碍物与导线均占据布线资源, 相邻层布线方向是正交的, 布线层被切分成若干个网

格单元. 在总体布线时，由于同一网格单元中不会进行布线，因此可以把网格单元抽象为质点，如图 1b 所示. 网格边用于连接相邻的网格单元，相邻层网格边方向是正交的. 一条网格边上分布着多个布线轨道，网格边容量等于该网格边上分布的轨道数量. 如果条网格边上的导线超过其容量，将产生溢出，经过产生溢出的网格边的线网称为非法线网. 为了保证可布线性，需对非法线网进行拆线重绕. 随着超大规模集成电路的发展，线网数量呈现爆炸式增长，如果导线都分布在低时延布线层，无疑将产生大量非法线网，进而影响可布线性. 因此，层分配需在合理权衡布线资源和时延等各项层分配指标的情况下进行.

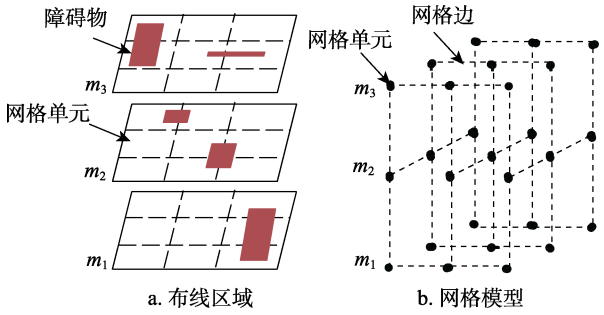


图 1　多层布线结构

如图 2 所示，在层分配中，e_2 上的导线可以分配到网格边 $e_{2,1}$ 或网格边 $e_{2,3}$. 图 2b 所示为图 2a 进行层分配后的布线方案，将 2 维布线方案中的 e_1 分配到 $e_{1,2}$，e_2 分配到 $e_{2,3}$，并产生相应的通孔连接同一线网的相邻层导线.

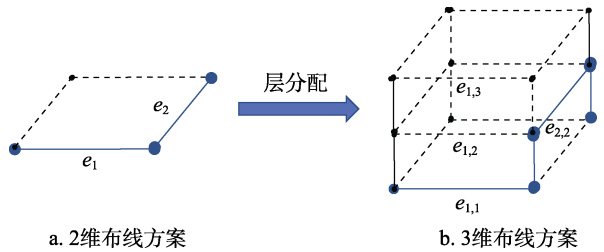


图 2　层分配示意

1.2　Elmore 时延计算模型

Elmore 时延计算模型被广泛应用于线网时延的评估[18]. 一个线网具有一个信号发射器和若干个信号接收器，从一个信号发射器到一个信号接收器上所有的导线集合称为一条路径. 导线 s 的时延计算公式为

$$d(s)=R(s)\times\left(C(s)/2+C_{\text{down}}(s)\right) \tag{1}$$

其中，$R(s),C(s),C_{\text{down}}(s)$ 分别表示导线 s 的电阻、电容和下游电容.路径由若干导线段组成，路径 p 的时延计算公式为

$$d(p)=\sum_{s\in p}d(s) \tag{2}$$

其中，$d(n)$ 表示线网 n 的时延，其计算公式为

$$d(n)=\sum_{p\in n}\lambda\times d(p) \tag{3}$$

其中，λ 表示自定义的参数. 为了使每个引脚同等重要，通常设置 $\lambda=1/S(n)$，$S(n)$ 表示线网 n 的引脚数量.

1.3　耦合效应

耦合效应的强度往往与导线之间的距离及导线密度有关，导线之间的距离越小，导线密度越大，则耦合效应的强度越大. 在超大规模集成电路中，线网数量多、布线密度大，导线之间会因电磁场相互作用产生耦合效应引起电容的变化，而电容将影响时延的计算. 因此，需在层分配过程中考虑耦合效应的影响.

导线的精确耦合电容值取决于轨道分配后的导线确切分布位置. 本文在轨道分配之前的层分配工作中使用概率估计的方法，通过生成查找表以近似评估耦合电容值，步骤如下:

Step1. 将一段确定导线类型的导线分配到某层 t.

Step2. 为了使预估布线层布线密度近似等于最终布线方案的布线密度，将导线随机分配到 $t-1,t,t+1$ 层.

Step3. 通过 fastCap[19]或者 FFTCAP[20]生成耦合电容值. 平行线参照文献[1]处理. 根据文献[17]中的方法得到平均耦合电容值，并将其存储于查找表中.

在后续布线阶段中，根据导线分配的布线层、导线类型和预估导线密度查表，得到耦合电容值.

2　本文算法

2.1　算法流程

如图 3 所示，本文算法分为预分配、修复以及精练 3 个阶段. 3 个阶段都将 2 维布线方案视为 2 维布线树，通过基于动态规划的单线网层分配算法得到 3 维布线方案.

为了同时考虑时延、通孔数和拥塞对层分配 3 维布线方案的影响，更全面地评估每个布线方案的优劣性，将目标函数设置为

$$f(n)=c_{\text{d}}\times\alpha\times d(n)+c_{\text{v}}\times\beta\times\#v_{\text{n}}+\delta\times\sum_{s\in n}c_{\text{congestion}}(s) \tag{4}$$

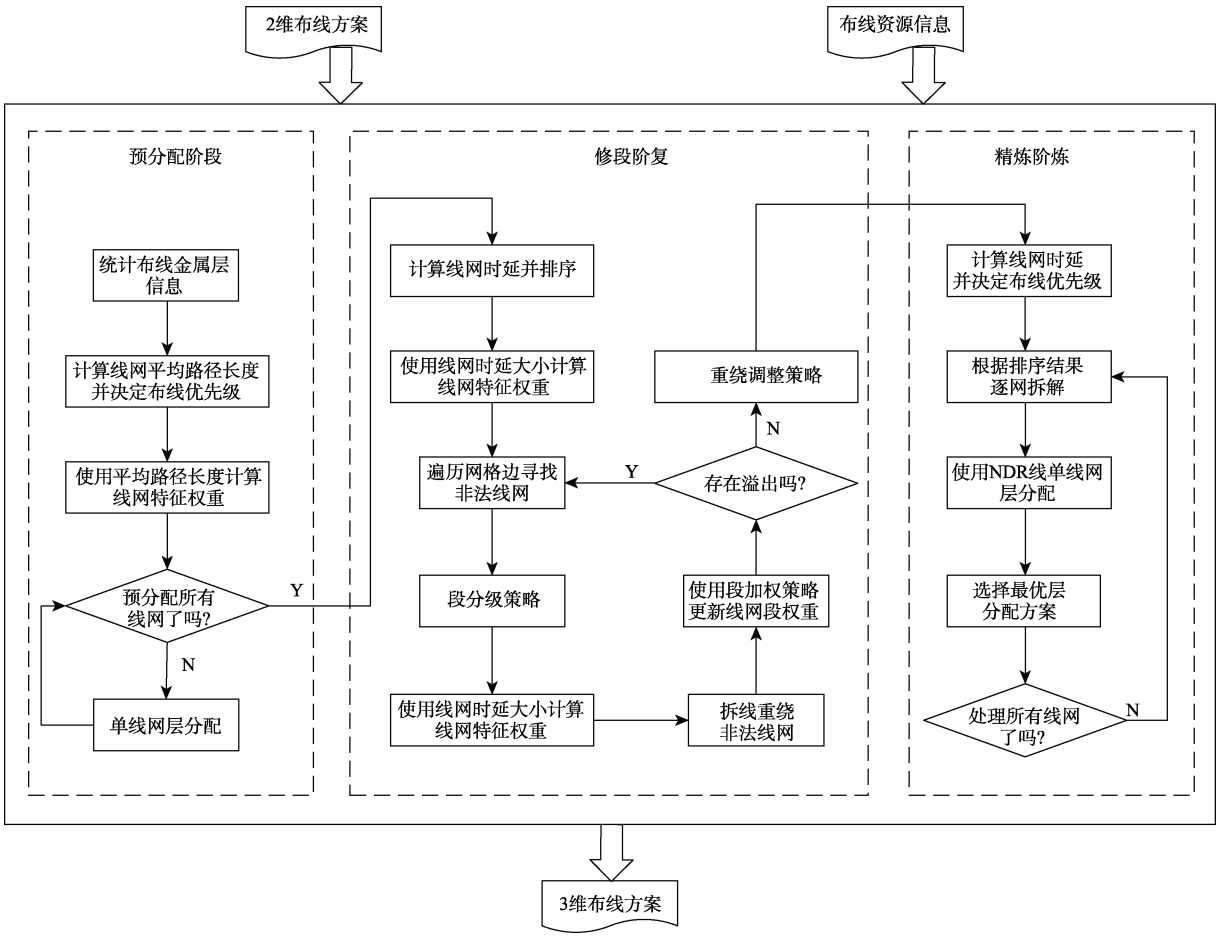


图 3 本文算法流程

其中，$f(n), d(n), \#v_{\mathrm{n}}, c_{\mathrm{congestion}}(s)$ 分别表示线网 n 的目标函数、时延、通孔数和拥塞代价；c_{d} 表示线网时延特征权重；c_{v} 表示线网通孔特征权重；α, β, δ 均表示自定义参数，分别用于调节线网时延、通孔数和拥塞代价三者在目标函数中所占权重，由于各阶段的优化目标不同，α, β, δ 值也不同.

拥塞代价的计算公式为

$$c_{\mathrm{congestion}}(e) = p(e) \times h(e) \tag{5}$$

其中，$h(e)$ 表示历史代价；

$$p(e) = \max\{0, \eta \times (u(e) - c(e))\} \tag{6}$$

表示惩罚项. 式(6)中，$c(e)$ 表示网格边 e 的容量；$u(e)$ 表示当前网格边 e 上承载的导线数量.

预分配阶段将得到初始布线方案，该方案为后续阶段提供指导. 由于标准测试电路中的线网顺序具有随机性，层分配如果采用该顺序进行布线通常不能取得良好的电路性能. 因此，本文通过对式(3)进行深入分析，提出平均路径长度. 在预分配阶段，采用平均路径长度使得布线顺序更具有合理性，为后续布线过程奠定良好的基础；同时，通过平均路径长度排序结果计算线网特征权重，充分体现线网差异性，以将低时延层的布线资源预留给时序关键线网. 为了在预分配阶段不消耗过多布线资源，预分配阶段将不使用 NDR 线. 此外，本阶段的时延权重 α 、通孔权重 β 和拥塞权重 δ 分别为 10, 1 和 0.1，保证在时延和通孔指标较为理想的情况下考虑拥塞的影响，从而减少溢出线网数量，为修复阶段减轻负担.

修复阶段将拆线重绕所有非法线网，得到无溢出的布线方案. (1) 根据预分配阶段得到的布线方案计算各线网时延并按从小到大排序，计算修复阶段的线网特征权重，使得时延权重、通孔权重和线网特征相适应，以取得更好的布线方案. (2) 通过 SCS 细致评估当前网格边的拥塞情况，合理更新历史代价；并进一步分析 2 维网格边的拥塞情况，提出线网重叠边，使得布线顺序与线网拥塞程度相匹配. 通过段加权策略[17]，保证时序关键段在多次拆线重绕后对时延优化依然具有重要影响. 在拆线重绕非法线网过程中，组合使用 SCS、重叠边排序方法以及段加权策略，保证在消除所有

非法线网后依然具有较好的时延性能. (3) 通过 RAAS 进一步优化时延. 修复阶段的时延权重 α 、通孔权重 β 以及拥塞权重 δ 分别为 10, 1 和 1, 较大的拥塞权重能加快迭代修复非法线网的速度.

精练阶段将保证在不发生溢出的情况下，进一步优化线网时延. 根据修复阶段得到的布线方案计算各线网时延并按从大到小排序，以此决定线网布线顺序；然后根据布线顺序逐网拆线重绕，并比较 2 个布线方案优劣，选择出最好的布线方案. 由于时序关键线网对电路性能的重要影响，精练阶段将使用 NDR 线进一步优化其时延. 精练阶段的时延权重 α 、通孔权重 β 以及拥塞权重 δ 分别为 10, 1.4 和 1. 较大的通孔权重保证优化线网时延时，通孔数指标仍保持较优水平.

2.2　DNS

随着超大规模集成电路制程工艺的不断发展，线网数量显著增长，而线网结构却不尽相同. 现有的层分配算法中，通常为各线网时延权重和通孔权重赋予相同数值，此方式并不能体现各线网的结构差异性，严重阻碍层分配指标优化. 针对该问题，本文根据线网结构差异性提出 DNS，使得时延权重、通孔权重和线网特征相适应，以取得更好的时延优化效果. DNS 的步骤如下.

Step1. 在层分配各阶段布线前，根据线网特征计算线网特征值，并将其从小到大排序，得到线网顺序编号. 在预分配阶段，根据平均路径长度决定线网特征值；在修复阶段和精练阶段，根据时延决定线网特征值.

Step2. 根据线网顺序编号，使用

$$c_{\mathrm{d}} = a + k \times (i/n) \tag{7}$$

$$c_{\mathrm{v}} = t \times c_{\mathrm{d}} \tag{8}$$

分别计算线网时延特征权重和线网通孔特征权重. 其中，i 表示该线网顺序编号；n 表示线网总数；a 和 k 是自定义参数，用于调节线网特征权重的大小. 本文经过多次实验，最终将两者分别设置为 11 和 100；t 是自定义参数，由于本文重点优化目标为时延，因此将其设置为 0.5.

因为时序关键线网的线网顺序 i 较大，所以式(7)(8)中时序关键线网的线网特征值较大. 通过 DNS 分辨线网的差异性，赋予时序关键线网更大的布线特权，确保时序关键线网被分配至低时延布线层，避免非时序关键线网抢夺不富裕的低时延层的布线资源. 这样做既有利于总时延优化，又促进时序关键线网时延优化.

图 4 所示为 DNS 有效性分析. m_1 和 m_5 是位于不同梯队的 2 层布线层，空心菱形表示信号发射器；实心菱形表示信号接收器. m_1 上有网格边 $e_{1,1}$ 和 $e_{1,2}$，$e_{1,1}$ 的容量为 1，$e_{1,2}$ 因为被障碍物覆盖，其容量为 0. m_5 有网格边 $e_{5,1}$ 和 $e_{5,2}$，其容量均为 1，但 m_5 上的导线段时延小于 m_1 上的导线段时延. 由于预分配阶段倾向于得到时延指标较优的布线方案，因此网格边 $e_{5,1}$ 产生溢出，线网 n_1 和线网 n_2 都是违规线网，需要在修复阶段进行拆线重绕. 未引入 DNS 时，首先拆线重绕线网 n_2，由于网格边 $e_{5,1}$ 拥塞代价较高，得到图 4c 所示的布线方案；引入 DNS 后，由于线网 n_2 的线网顺序编号较大，其线网特征权重较大，被赋予较大的布线特权，因此得到图 4d 所示的布线方案. 可以看出，图 4d 的布线方案的时延和通孔数均优于图 4c 的布线方案.

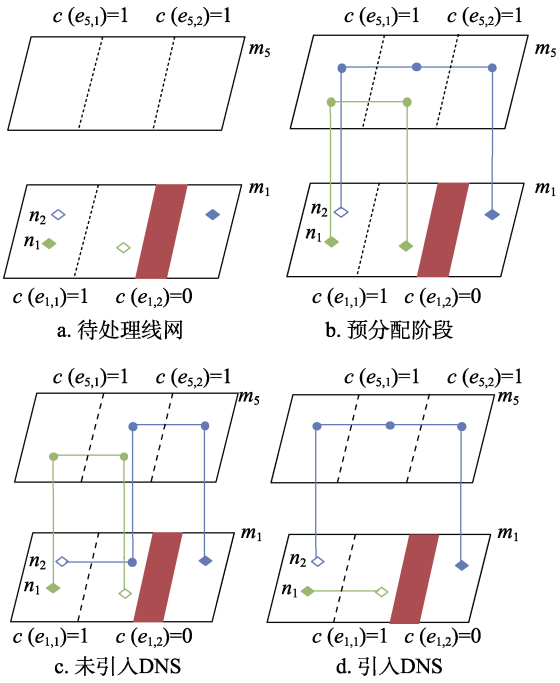


图 4　DNS 有效性分析

2.3　SCS

现有层分配算法中，通常当网格边发生溢出时才更新历史代价，其余情况下，各个网格边的拥塞情况视为一致. 但是，即使 2 个布线方案均不发生溢出，其对后续层分配的布线灵活性仍有不同影响. 针对该问题，本文算法提出 SCS，其针对当前网格边经过的具体导线数量，将网格边分为不同的拥塞等级，对网格边的拥塞情况进行更加细致的定量分析，提出历史代价更新的方法，计算公式为

$$h(e) = x \times d_{\mathrm{e}} + y \times r_{\mathrm{e}} \tag{9}$$

其中，x 和 y 都是自定义的参数，本文采用控制变量法进行多次实验，最终将两者均设置为 0.05 时获得了理想的布线方案，当 x 和 y 均为 0.05 时，拥塞代价在目标函数中与时延代价、通孔代价处于一个数量级，更全面地优化各个层分配指标；d_e 表示目前已经进行过的修复非法线网的次数；r_e 表示网格边 e 的拥塞级别，计算公式为

$$r_e = c/z \tag{10}$$

其中，c 表示当前网格边 e 上经过的导线数量；z 表示自定义参数. 若 z 的数值过大，将导致大多数网格边的拥塞级别都是一样的，不能精确地评估拥塞情况；否则，将导致网格边的拥塞级别过大，即历史代价过大，进而导致拥塞代价过大. z 的数值过大或过小将最终导致过分重视拥塞约束，不利于优化时延，所以 z 必须设置为合理的数值. 本文采用控制变量法进行多次实验后，将其设置为 10.

拆线重绕非法线网前，导线大都分布在低时延布线层，导致低时延布线层上的网格边的拥塞等级 r_e 较大. 在拆线重绕过程中，SCS 将尽量避免在拥塞级别 r_e 较大的网格边上继续布线，以优化网格边拥塞情况. 同时，在多次拆线重绕后，非法线网数量不断减少，各网格边的拥塞等级 r_e 随之降低，而拆线重绕次数 d_e 不断增加. 考虑 d_e 能够有效地防止进行多次拆线重绕后历史代价变得极小，避免拆线重绕过程不能收敛.

图 5 所示为 SCS 有效性分析. 假设线网 n_1 和线网 n_2 已完成布线，m_1 中的网格边 $e_{1,1}$ 的容量为 4，分级标准 z 为 2. 未引入 SCS 对线网 n_3 进行布线时，因为网格边 $e_{1,1}$ 并未发生溢出，$e_{1,1}$ 和 $e_{5,1}$ 拥塞代价相同，所以得到图 5a 所示的布线方案；引入 SCS 后，由于网格边 $e_{1,1}$ 的拥塞等级较高，线网 n_3 布线时会绕开网格边 $e_{1,1}$，选择在网格边 $e_{5,1}$ 布线，得到图 5b 所示的布线方案，可以看出，m_5 上导线线宽较大，时延较小，达到优化时延的目的.

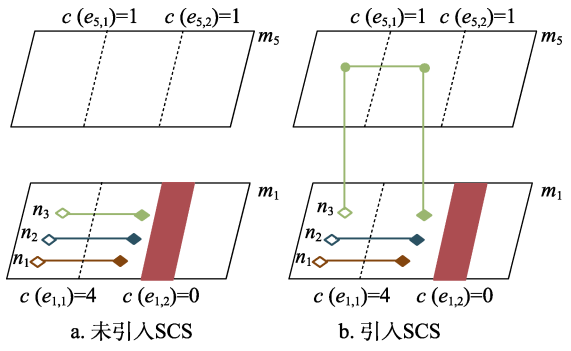


图 5 SCS 有效性分析

2.4 RAAS

在现有层分配算法中，对非法线网进行拆线重绕的过程往往更加重视如何快速地消除非法线网，导致在目标函数中拥塞代价所占的比重过大，最终布线方案中时延指标不理想. 本文在迭代修复所有非法线网后，提出 RAAS 来解决该不足之处.

算法 1. RAAS 算法.

输入. 优化前布线方案 R_1.

输出. 优化后布线方案 R_2.

```
for i = 1,2,⋯ do
    记录布线方案 R1 中线网 n_i 的时延 d_i .
end for
for i = 1,2,⋯ do
    重绕调整次数 t_reassign = 1.
    if 重绕调整次数 t_reassign 不大于阈值 t .
        线网 n_i 拆线重绕得到线网 n_i_new ;
        重绕调整次数 t_reassign = t_reassign + 1 ;
        计算线网 n_i_new 的时延 d_i_new ;
        if d_i_new > d_i
            线网 n_i_new 恢复为 n_i ;
            α = 3 × α .
        end if
    end if
end for
```

记录原线网的时延后，RAAS 对线网进行拆线重绕；然后比较新线网时延和原线网时延的大小. 如果原线网的时延更小，则线网恢复为原线网，并增大目标函数的时延权重 α. 由于增大时延权重将导致线网的时延代价在总代价中占据更大的比重，因此层分配算法倾向于选择时延指标较低的布线方案. 因为多次使用 RAAS 后无法继续明显地优化时延，所以本文算法设置了一个阈值，当优化次数达到阈值后将跳过 RAAS. 通过多次实验，本文将阈值设置为 2.

图 6 所示为 RAAS 有效性分析. 假设网格边

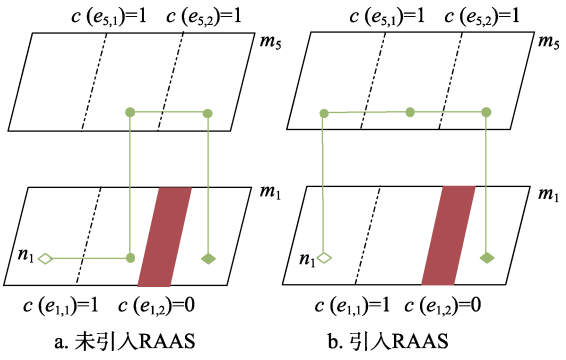


图 6 RAAS 有效性分析

$e_{5,1}$ 发生多次溢出，导致拥塞代价和时延代价失衡，即使 $e_{5,1}$ 有剩余容量也不会在该网格边上进行布线，所以得到如图 6a 所示的布线方案. 引入重绕调整后，由于在拆线重绕中不断提高时延权重，使得时延代价在目标函数中占据更大的比重，进而能够克服两者失衡问题，更充分地利用低时延层的布线资源，得到图 6b 所示的布线方案. 与图 6a 相比，图 6b 在保证通孔数不变的情况下优化时延.

2.5 MDSS

在传统层分配中，不同阶段的目标不同，但大多层分配算法采用单一的排序方法决定布线顺序，而布线顺序很大程度上决定布线性能优劣，因此不能适应多阶段的层分配算法. 针对此问题，本文算法提出 MDSS，以满足不同阶段不同的布线目标.

预分配阶段希望得到初步布线方案，为后续阶段提供指导. 由于式(3)在计算时延时，线网时延并不是路径时延之和，而是各条路径的平均时延；同时，预分配阶段前并无具体 3 维布线方案，无法准确计算线网时延. 因此，基于对 2 维布线方案线网结构以及对式(3)的分析，预分配阶段使用平均路径长度排序方法近似评估线网时延，并以此决定布线顺序. 修复阶段中，既要考虑时延优化效果，又要保证修复所有非法线网，因此使用重叠边排序方法决定其布线顺序. 精练阶段中，要求在现有布线方案的基础上进一步优化时延，因此根据时延大小决定其布线顺序.

MDSS 虽然未对是否陷入局部最优进行判断，但其设计出多种新颖的排序方法并将其应用于不同布线阶段，满足各个阶段不同的布线要求，使得每个阶段都更充分地利用低时延层的布线资源. 因此，在不增加运行时间的情况下，时延以及通孔数均得到优化.

2.5.1 平均路径长度排序方法

标准测试电路中，线网顺序具有随机性，层分配采用该顺序进行布线时通常不能取得良好的电路性能；同时，式(3)中的时延等于各路径时延平均值，并且路径长度与该路径时延呈正相关关系. 针对该特点，本文提出平均路径长度排序方法，通过计算 2 维布线方案的线网平均路径长度，可有效地预测层分配后线网时延大小，该长度决定布线顺序，为后续层分配阶段奠定良好的布线基础. 平均路径长度的计算公式为

$$l_n = \left(\sum_{p\in n}\sum_{s\in p} s\right) \Big/ S(n) \tag{11}$$

其中，s 表示线网 n 中的导线；l_n 表示线网 n 平均路径长度.

2.5.2 重叠边排序方法

现有层分配算法中，通常以线网时延大小为标准对线网进行排序，根据排序结果决定布线顺序；其弊端在于排序标准只有时延，不能综合考虑各项层分配指标的影响.

为了更全面、正确地计算线网布线顺序，本文算法提出重叠边排序方法. 考虑线网引脚数和时延一定程度上成正比关系，将线网引脚数纳入排序标准，同时，线网重叠边数量反映 2 维布线方案中线网的拥塞情况. 因此，本文算法将重叠边数量纳入排序标准.

线网重叠边的计算公式为

$$o_n = \sum_{e\in n} c_e \tag{12}$$

其中，o_n 表示线网 n 的重叠边数量；c_e 表示在 2 维布线方案中网格边 e 上所经过的导线的数量.

重叠边排序方法的计算公式为

$$p_n = \mu \times S(n) + v \times o_n \tag{13}$$

其中，μ 和 v 是自定义参数，通过调节这 2 个参数使得引脚数和重叠边数量处于同一数量级. 本文通过多次实验，设置 $\mu=1$ 和 $v=1\,000$.

3 实验及结果分析

本文算法已用 C++语言实现，所有实验均在 2.60 GHz CPU，64 GB 内存，以及 Linux 环境的 PC 机上执行. 本文算法的布局方案由 NTUplace4[21]生成，总体布线方案由 NCTU-GR2.0[22]生成. 本文测试用例源自国际设计自动化会议(design automation conference，DAC)2012 年比赛数据的测试电路[22]，其数据规模如表 1 所示.

布线方案使用本文算法，实验的重点是验证该算法中使用的各项策略的有效性，以及本文算法相对同类层分配算法的有效性. 本文中，D_{total} 表示总时延；$D_{\max}$ 表示最大时延；0.5%，1.0%，5.0%分别表示按照线网时延降序排序下，前 0.5%，1.0%和 5.0%线网的平均时延；#VC 表示通孔数；TIME 表示运行时间. 本文算法是在固定芯片面积情况下进行层分配的，同时在层分配过程中考虑拥塞的影响，最终的布线方案均不发生溢出.

本文在确定每个参数时均采用控制变量法，在保持其他参数不变的基础上，针对某个参数，根据经验给出整数范围；在该整数范围内，以等梯度增长的方式测试每个可能的参数值，最终选取实

表 1 测试用例数据规模

测试电路	网格数	线网数	布线层数
SP2	770×891	990 899	9
SP3	800×415	898 001	9
SP6	649×495	1 006 629	9
SP7	499×713	1 340 418	9
SP9	426×570	833 808	9
SP11	631×878	935 731	9
SP12	444×518	1 293 436	9
SP14	406×473	619 815	9
SP16	465×404	697 458	9
SP19	321×518	511 685	9

验结果最好的参数值作为最终参数值.

3.1 有无耦合效应的实验

耦合效应是超大规模集成电路布线工程客观存在的现象, 考虑耦合效应的层分配算法更贴近实际工程. 表 2 所示为有/无耦合效应的实验结果对比. 可以看出, 本文算法不考虑耦合效应这一影响时延评估准确性的因素时, 总时延增加 5%, 按照线网时延降序排序下, 前 0.5%, 1.0%和 5.0%线网的平均时延均增加 4%; 同时, 未考虑耦合效应的情况下, 线网时延评估较小, 导致通孔数在目标函数中占据更大权重, 最终导致通孔数减少 5%.

3.2 各项策略有效性验证

(1) DNS 根据不同线网结构特征, 赋予时序关键线网更大的线网特征权重, 更好地体现线网差异性, 更充分地利用低时延层的布线资源, 进而站在更全局的角度优化时序特性. 从图 7a~7e 可以看出, 该策略能够有效地优化时延.

表 2 本文算法有/无耦合效应实验结果对比

测试电路	有耦合							无耦合						
	D_{total} /μs	D_{max} /μs	0.5%/μs	1.0%/μs	5.0%/μs	#VC/个	TIME/s	D_{total} /μs	D_{max} /μs	0.5%/μs	1.0%/μs	5.0%/μs	#VC/个	TIME/s
SP2	21.42	467	138	113	49	6.63M	1 260	21.63	455	138	112	50	6.34M	1 153
SP3	7.07	457	83	61	21	6.13M	809	7.26	456	83	62	21	5.81M	743
SP6	5.80	203	50	38	15	5.99M	697	6.06	210	51	39	15	5.70M	632
SP7	5.38	176	37	26	9	9.20M	876	5.76	171	38	28	9	8.71M	835
SP9	3.67	152	42	30	11	5.04M	537	3.90	142	43	31	12	4.77M	513
SP11	6.90	1 470	106	64	18	5.48M	622	7.12	1 375	107	64	18	5.25M	574
SP12	4.73	570	36	25	9	8.30M	820	4.95	576	37	26	9	7.94M	784
SP14	3.55	201	49	35	13	3.89M	439	3.69	208	50	36	13	3.73M	389
SP16	4.83	143	46	36	16	3.84M	501	5.03	150	50	39	17	3.68M	449
SP19	1.68	295	19	15	7	2.99M	282	1.81	306	21	16	8	2.85M	264
比率	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.05	1.00	1.04	1.04	1.04	0.95	0.93

(2) SCS 通过统计每条网格边上承载的导线数量, 将其分成不同的拥塞等级, 更准确评估网格边拥塞情况, 优化布线方案的可布线性和时序特性. 从图 7a~7f 可以看出, 该策略对时延和通孔均有明显的优化作用.

(3) RAAS 不仅关注于如何快速消除溢出边, 还考虑时延对层分配的重要影响. 通过对线网进行数次拆线重绕并增加时延权重大小, 导致本文算法倾向于选择时延指标较低的线网, 使得在保证可布线性的前提下进一步优化时序特性. 从图 7a 及图 7c~7e 可以看出, 该策略可以优化总时延以及时序关键线网时延.

(4) MDSS 分别为预分配阶段、修复阶段和精练阶段设计不同的决定布线顺序的方法, 规范布线过程以提高层分配质量. 从图 7b 可以看出, 该策略可以明显地优化最大时延. 此外, 从图 7h 可以看出, 引入本文提出的每项策略后, 均能在可接受的运行时间内得到更理想的布线方案.

3.3 本文算法有效性验证

表 3 所示为本文算法与文献[17]算法的实验结果对比. 可以看出, 总时延降低 11%, 最大时延降低 13%, 按照线网时延降序排序下前 0.5%, 1.0%和 5.0%线网的平均时延分别降低 12%, 14%和 14%, 通孔数减少 3%; 为了避免非时序关键线网占据低时延层的布线资源, 在 DNS 中对其赋予更小的时延权重, 导致迭代次数增加, 引起运行时间增加 6%. 实验结果表明, 本文算法在可接受的时间下, 时延各项指标和通孔数指标均达到最优效果.

将本文算法分别与文献[15,23]算法进行实验对比, 进一步验证该算法的有效性, 结果如表 4 和表 5 所示. 与文献[15]算法相比, 本文算法在总时延、最大时延、按照线网时延降序排序下前 0.5%,1.0%和

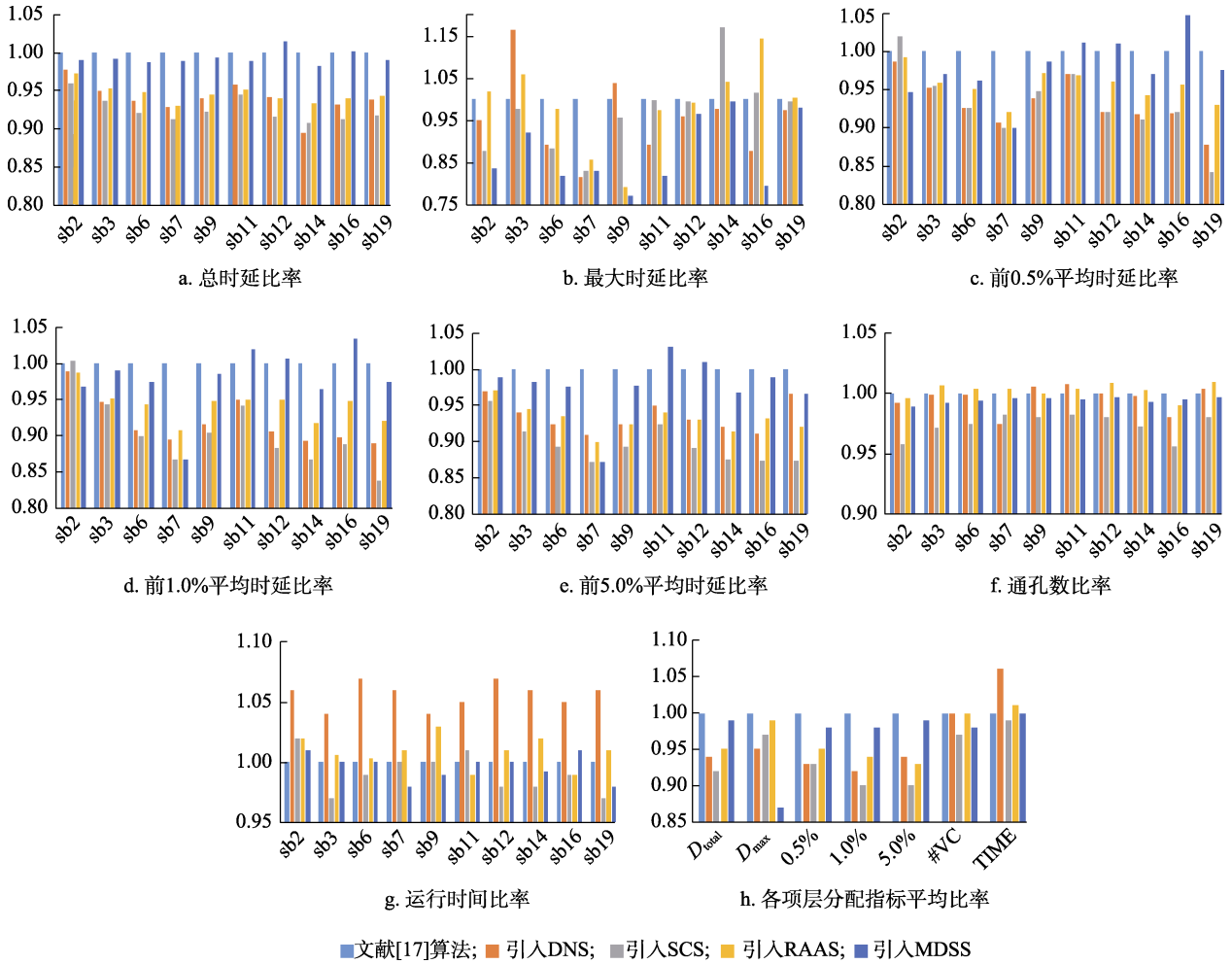


图 7　文献[17]和本文各项策略在不同电路上的有效性验证

表 3　本文与文献[17]实验结果对比

测试电路	文献[17]							本文						
	D_{total} /μs	D_{max} /μs	0.5%/μs	1.0%/μs	5.0%/μs	#VC/个	TIME/s	D_{total} /μs	D_{max} /μs	0.5%/μs	1.0%/μs	5.0%/μs	#VC/个	TIME/s
SP2	22.22	504	145	116	52	7.13M	1 198	21.42	467	138	113	49	6.63M	1 260
SP3	7.75	535	89	66	23	6.38M	781	7.07	457	83	61	21	6.13M	809
SP6	6.55	254	57	44	17	6.20M	649	5.80	203	50	37	15	5.99M	697
SP7	6.25	233	43	31	11	9.34M	826	5.38	176	37	26	9	9.20M	876
SP9	4.14	212	46	34	13	5.12M	501	3.67	151	41	30	11	5.04M	537
SP11	7.51	1 602	110	68	20	5.59M	582	6.90	1 469	106	63	18	5.48M	622
SP12	5.43	582	40	30	10	8.50M	774	4.73	570	36	25	9	8.30M	820
SP14	4.08	205	56	42	15	4.04M	396	3.55	201	48	35	13	3.89M	439
SP16	5.44	186	54	43	19	4.12M	469	4.83	143	46	36	16	3.84M	501
SP19	1.94	305	25	19	9	3.04M	268	1.68	295	19	15	7	2.99M	282
比率	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.89	0.87	0.88	0.86	0.86	0.97	1.06

5.0%线网的平均时延优化比例分别达到 10%, 9%, 9%, 11%和 12%, 通孔数优化比例达到 4%. 与文献[23]相比, 本文算法在总时延、最大时延、按照线网时延降序排序下前 0.5%, 1.0%和 5.0%线网的平均时延分别优化 7%, 13%, 7%, 8%和 9%. 实验结果表明, 本文算法时延各项指标和通孔数指标均达到最优效果.

表 4 本文与文献[15]实验结果对比

测试电路	文献[15]						本文					
	D_{total} /μs	D_{max} /μs	0.5%/μs	1.0%/μs	5.0%/μs	#VC/个	D_{total} /μs	D_{max} /μs	0.5%/μs	1.0%/μs	5.0%/μs	#VC/个
SP2	22.20	462	144	116	51	7.13M	21.42	467	138	113	49	6.63M
SP3	7.73	552	89	66	23	6.39M	7.07	457	83	61	21	6.13M
SP6	6.19	219	51	39	16	6.33M	5.80	203	50	37	15	5.99M
SP7	6.22	206	43	31	11	9.35M	5.38	176	37	26	9	9.20M
SP9	4.12	220	46	34	13	5.13M	3.67	151	42	30	11	5.04M
SP11	7.48	1 511	109	68	20	5.60M	6.90	1 469	106	63	18	5.48M
SP12	5.36	576	39	29	10	8.57M	4.73	570	36	25	9	8.30M
SP14	3.98	208	53	39	15	4.06M	3.55	201	49	35	13	3.89M
SP16	5.43	170	54	43	19	4.12M	4.83	143	46	36	16	3.84M
SP19	1.90	296	23	18	8	3.09M	1.68	295	19	15	7	2.99M
比率	1.00	1.00	1.00	1.00	1.00	1.00	0.90	0.91	0.91	0.89	0.88	0.96

表 5 本文与文献[23]实验结果对比

测试电路	文献[23]						本文					
	D_{total} /μs	D_{max} /μs	0.5%/μs	1.0%/μs	5.0%/μs	#VC/个	D_{total} /μs	D_{max} /μs	0.5%/μs	1.0%/μs	5.0%/μs	#VC/个
SP2	21.81	423	143	114	50	6.86M	21.42	467	138	113	49	6.63M
SP3	7.49	564	85	63	22	6.22M	7.07	457	83	61	21	6.13M
SP6	6.33	295	55	42	16	6.04M	5.80	203	50	37	15	5.99M
SP7	5.92	206	40	29	10	9.12M	5.38	176	37	26	9	9.20M
SP9	4.00	189	44	33	12	5.00M	3.67	151	42	30	11	5.04M
SP11	7.22	1 571	106	65	19	5.46M	6.90	1 469	106	63	18	5.48M
SP12	5.15	585	38	28	10	8.32M	4.73	570	36	25	9	8.30M
SP14	3.90	208	53	39	14	3.94M	3.55	201	49	35	13	3.89M
SP16	5.15	219	51	40	18	3.92M	4.83	143	46	36	16	3.84M
SP19	1.85	309	23	17.7	8	2.96M	1.68	295	19	15	7	2.99M
比率	1.00	1.00	1.00	1.00	1.00	1.00	0.93	0.87	0.93	0.92	0.91	0.99

4 结 语

当前, 时延已成为制约芯片性能的重要因素. 为此, 本文提出先进制程下同时考虑 NDR 线和耦合效应的基于多策略融合的时延优化层分配算法, 其包含 4 个策略: (1) DNS 通过区分线网结构, 赋予时序关键线网更大特征权重, 使其拥有更大布线特权; (2) SCS 对网格边的拥塞情况进行更加细致的定量分析, 改进历史代价的更新方式; (3) RAAS 对线网进行数次拆线重绕, 并在此过程中不断地调整目标函数中的时延权重; (4) MDSS 分别设计出平均路径排序方法和重叠边排序方法, 并根据层分配各阶段不同目标选择合适的排序方法. 实验结果表明, 与现有层分配算法相比, 本文算法在保证不发生溢出的情况下, 有效地优化了时延及通孔数.

参考文献(References):

[1] Ewetz R, Liu W H, Chao K Y, *et al*. A study on the use of parallel wiring techniques for sub-20nm designs[C] //Proceedings of the 24th Edition of the Great Lakes Symposium on VLSI. New York: ACM Press, 2014: 129-134

[2] Liu W H, Kao W C, Li Y L, *et al*. NCTU-GR 2.0: multithreaded collision-aware global routing with bounded-length maze routing[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2013, 32(5): 709-722

[3] Gester M, Müller D, Nieberg T, *et al*. BonnRoute: algorithms and data structures for fast and good VLSI routing[J]. ACM Transactions on Design Automation of Electronic Systems, 2013, 18(2): Article No.32

[4] Roy J A, Markov I L. High-performance routing at the nanometer scale[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2008, 27(6): 1066-1077

[5] Xu Y, Zhang Y H, Chu C. FastRoute 4.0: global router with ef-

ficient via minimization[C] //Proceedings of the Asia and South Pacific Design Automation Conference. Los Alamitos: IEEE Computer Society Press, 2009: 576-581

[6] Ozdal M M, Wong M D F. Archer: a history-based global routing algorithm[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2009, 28(4): 528-540

[7] Liu J W, Pui C W, Wang F Z, *et al*. CUGR: detailed-routability-driven 3D global routing with probabilistic resource model[C] //Proceedings of the 57th ACM/IEEE Design Automation Conference. Los Alamitos: IEEE Computer Society Press, 2020: 1-6

[8] Zhu W D, Zhang X H, Liu G G, *et al*. MiniDeviation: an efficient multi-stage bus-aware global router[C] //Proceedings of the International Symposium on VLSI Design, Automation and Test. Los Alamitos: IEEE Computer Society Press, 2020: 1-4

[9] Goncalves S M M, da Rosa L S, de S Marques F. An improved heuristic function for A*-based path search in detailed routing[C] //Proceedings of the International Symposium on Circuits and Systems. Los Alamitos: IEEE Computer Society Press, 2019: 1-5

[10] Feng Gang, Ma Guangsheng, Du Zhenjun. Signal-related crosstalk optimization for detailed routing[J]. Journal of Computer-Aided Design & Computer Graphics, 2005, 17(5): 1074-1078(in Chinese)
(冯刚, 马光胜, 杜振军. 信号相关的串扰优化详细布线[J]. 计算机辅助设计与图形学学报, 2005, 17(5): 1074-1078)

[11] Hu S Y, Li Z, Alpert C J. A fully polynomial-time approximation scheme for timing-constrained minimum cost layer assignment[J]. IEEE Transactions on Circuits and Systems II: Express Briefs, 2009, 56(7): 580-584

[12] Hu S, Li Z, Alpert C J. A faster approximation scheme for timing driven minimum cost layer assignment[C] //Proceedings of the ACM International Symposium on Physical Design. New York: ACM Press, 2009: 167-174.

[13] Hu S Y, Li Z, Alpert C J. A polynomial time approximation scheme for timing constrained minimum cost layer assignment[C] //Proceedings of the IEEE/ACM International Conference on Computer-Aided Design. Los Alamitos: IEEE Computer Society Press, 2008: 112-115

[14] Ao J C, Dong S Q, Chen S, *et al*. Delay-driven layer assignment in global routing under multi-tier interconnect structure[C] //Proceedings of the ACM International Symposium on Physical Design. New York: ACM Press, 2013: 101-107

[15] Dong S Q, Ao J C, Luo F Q. Delay-driven and antenna-aware layer assignment in global routing under multitier interconnect structure[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2015, 34(5): 740-752

[16] Yu B, Liu D R, Chowdhury S, *et al*. TILA: timing-driven incremental layer assignment[C] //Proceedings of the IEEE/ACM International Conference on Computer-Aided Design. Los Alamitos: IEEE Computer Society Press, 2015: 110-117

[17] Han S Y, Liu W H, Ewetz R, *et al*. Delay-driven layer assignment for advanced technology nodes[C] //Proceedings of the 22nd Asia and South Pacific Design Automation Conference. Los Alamitos: IEEE Computer Society Press, 2017: 456-462

[18] Elmore W C. The transient response of damped linear networks with particular regard to wideband amplifiers[J]. Journal of Applied Physics, 1948, 19(1): 55-63

[19] Nabors K, White J. FastCap: a multipole accelerated 3-D capacitance extraction program[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 1991, 10(11): 1447-1459

[20] Phillips J R, White J K. A precorrected-FFT method for electrostatic analysis of complicated 3-D structures[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 1997, 16(10): 1059-1072

[21] Hsu M K, Chen Y F, Huang C C, *et al*. Routability-driven placement for hierarchical mixed-size circuit designs[C] //Proceedings of the 50th Annual Design Automation Conference. New York: ACM Press, 2013: 1-6

[22] Viswanathan N, Alpert C, Sze C, *et al*. The DAC 2012 routability-driven placement contest and benchmark suite[C] //Proceedings of the DAC Design Automation Conference 2012. Los Alamitos: IEEE Computer Society Press, 2012: 774-782

[23] Chen J W, Di Z X, Chen J J, *et al*. An enhanced heuristic layer assignment method in global routing[C] //Proceedings of the 15th IEEE International Conference on Solid-State & Integrated Circuit Technology. Los Alamitos: IEEE Computer Society Press, 2020: 1-3