考虑簇减少的可变长度限制X结构Steiner最小树算法

刘耿耿[1,2,3],吕星灿[1,2,3],周茹平[1,2,3],郑 瀚[1,2,3],傅仰耿[1*]

(1. 福州大学计算机与大数据学院,福建福州 350116;2. 大数据智能教育部工程研究中心,福建福州 350116;
3. 福建省网络计算与智能信息处理重点实验室,福建福州 350116)

摘 要: 多动态电压(Multiple Dynamic Supply Voltage,MDSV)设计模式通过在芯片中划分不同电压域,使各模块在满足性能与时序要求的前提下采用差异化供电,从而降低整体功耗并提升能效。然而,现有的Steiner最小树(Steiner Minimum Tree,SMT)问题相关工作尚未考虑到MDSV设计模式带来的新约束条件与新优化目标,难以满足MDSV设计模式下的可变长度约束,同时也未考虑引入电压转换器(Level Shifter,LS)带来的额外开销。为此,本文提出了一种考虑簇减少的可变长度限制X结构SMT算法(Variable Length-Restricted X-architecture Steiner Minimum Tree algorithm considering Cluster Reduction, VLRXSMT-CR)。首先,为有效求解MDSV设计模式下SMT这一离散化多目标优化问题,提出了离散化的正向学习算子和变异算子,并引入竞争选择机制与基于拥塞度的外部存档更新机制,实现了LS数量和线长的多目标优化。其次,为避免算法过早陷入局部最优并增强迭代后期的局部搜索能力,以进一步提升解的质量,提出了多样化麻雀比例调整策略。再次,为提高算法更新阶段学习对象的多样性,将分层学习机制引入离散麻雀搜索算法(Sparrow Search Algorithm,SSA)更新过程中。最后,为满足MDSV设计约束并进一步优化布线结构,引入了调整与混合精炼策略,对候选解中违反约束条件的布线边进行调整,同时对候选解中的非最优局部拓扑结构进行优化。与现有方法相比,本文提出的算法在线长指标上取得一定优化的同时,在MDSV设计模式下的关键性能指标(LS)使用数量方面取得了38.60%~51.91%的大幅优化。

关键词: 可变长度限制Steiner最小树(SMT);电压簇;分层学习麻雀搜索;多动态电压(MDSV)设计模式;多目标优化

基金项目: 国家自然科学基金(No.62372109);福建省杰出青年科学基金(No.2023J06017)

中图分类号: TP301 **文献标识码:** A **文章编号:** 0372-2112(2026)04-1682-18

电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20260062

Variable Length-Restricted X-Architecture Steiner Minimum Tree Algorithm Considering Cluster Reduction

LIU Genggeng[1,2,3], LÜ Xingcan[1,2,3], ZHOU Ruping[1,2,3], ZHENG Han[1,2,3], FU Yanggeng[1*]

(1. *College of Computer and Data Science*, *Fuzhou University*, *Fuzhou*, *Fujian* 350116, *China*;
2. *Engineering Research Center of Big Data Intelligence*, *Ministry of Education*, *Fuzhou*, *Fujian* 350116, *China*;
3. *Fujian Key Laboratory of Network Computing and Intelligent Information Processing*, *Fuzhou*, *Fujian* 350116, *China*)

Abstract: The multiple dynamic supply voltage (MDSV) design paradigm partitions a chip into multiple voltage domains, enabling modules to operate with differentiated supply voltages while meeting performance and timing requirements, thereby reducing overall power consumption and improving energy efficiency. However, existing studies on the Steiner minimum tree (SMT) problem have not yet taken into account the new constraints and optimization objectives introduced by the MDSV design paradigm. Consequently, these methods are unable to effectively handle variable length restriction under the MDSV environment and fail to account for the additional overhead introduced by level shifters (LSs). To address these issues, this paper proposes a variable length-restricted X-architecture SMT algorithm considering cluster reduction (VLRXSMT-CR). First, in order to effectively solve the discrete multi-objective optimization problem of the SMT under the MDSV design paradigm, discrete forward learning operators and mutation operators are proposed. In addition, a competitive selection mechanism and a congestion-based external archive update mechanism are introduced to achieve multi-objective optimization of the number of LSs and routing wirelength. Second, to avoid premature convergence of the algorithm to local optima and to enhance local search capability in the later stages of iteration, thereby further improving solution quali-

收稿日期:2026-03-05;录用日期:2026-04-08;责任编委:明鑫

*通信作者:傅仰耿

ty, a diversified sparrow proportion adjustment strategy is proposed. Then, to improve the diversity of learning targets during the algorithm update stage, a hierarchical learning mechanism is introduced into the update process of the discrete sparrow search algorithm (SSA). Finally, in order to satisfy MDSV design constraints and further optimize routing structures, adjustment and hybrid refinement strategies are introduced to adjust routing edges that violate constraint conditions in candidate solutions, while optimizing non-optimal local topological structures in candidate solutions. Compared with existing methods, the proposed algorithm achieves certain improvements in routing wirelength, while obtaining a significant improvement of 38.60% to 51.91% in the number of LSs, which is a key performance metric under the MDSV design paradigm.

Keywords: variable length-restricted Steiner minimum tree (SMT); voltage cluster; hierarchical learning-based sparrow search; multiple dynamic supply voltage (MDSV) design paradigm; multi-objective optimization

Foundation Item(s): National Natural Science Foundation of China (No.62372109); Fujian Science Fund for Distinguished Young Scholars (No.2023J06017)

0 引言

随着金属氧化物半导体(Metal Oxide Semiconductor,MOS)晶体管特征尺寸越来越小,片上系统(System on Chip,SoC)设计复杂度迅速增加,更多的新架构、新功能被添加至芯片中[1],大量晶体管集成在一个小面积芯片上,这使得互连延迟所带来的性能影响越发重要。同时,导线长度与互连延迟成二次关系,因此导线长度成为超大规模集成电路(Very Large-Scale Integration,VLSI)布线问题的重要优化目标之一[2]。此外,计算性能不断提高的同时带来了高功耗的问题。随着如手机、平板电脑等便携式设备的普及,电池容量受限的电子设备,使得功耗成为芯片制造商亟待解决的关键问题之一[3]。其中,动态功耗占据芯片总功耗的大部分,是其主要来源之一[4]。

考虑到供电电压与动态功耗之间的二次关系,因此通过降低供电电压能够有效降低电路功耗,然而这会导致电路性能降低[5]。在传统电压供应模式下,芯片所有功能部件均工作在同一高电压模式下,部分本可在较低电压下正常运行的部件也因此承受了不必要的高压,从而产生额外的功率消耗。针对这一问题,研究人员提出了多动态电压(Multiple Dynamic Supply Voltage,MDSV)设计模式。MDSV设计模式允许根据需求动态调整电源域的供电电压。当将具有较低电压的驱动节点连接到具有较高电压的接收节点时,需在较高电压电源域边界路径上插入电压转换器(Level Shifter,LS),而将具有较高电压的驱动节点连接到具有较低电压的驱动节点时无需LS。然而,LS的使用会在面积、延迟和功耗方面会产生额外开销[6]。减少LS的使用有利于提高电路性能并降低功耗开销,因此,在MDSV设计模式下的布线算法需将LS数量纳入优化目标。

此外,互连线长度对时延效应与电路功耗指标均具有显著影响,因此互连线长度成为VLSI物理设计阶段中重要的优化目标之一。Steiner最小树(Steiner Minimum Tree,SMT)模型具有良好的线长优化能力,是VLSI物理设计的基础模型之一,常被用于布局和布线阶段[7]。为此,研究人员基于SMT模型开展了一系列研究工作,提出了多种SMT算法用于解决布局布线问题。传统布线算法大多仅允许在水平或垂直方向上进行布线,称为曼哈顿结构或直角结构。然而,直角结构由于其布线方向受限,无法充分利用布线资源,不利于实现线长优化。为此,研究人员进一步提出了非曼哈顿结构。已有研究表明,非曼哈顿结构通过增加布线方向能够更充分地利用布线资源,其中X结构是最具代表性的形式之一[8]。

同时,芯片的集成度增加使得芯片中的元件和模块越来越多。在实际布线过程中,为避免拥塞、重叠,芯片中的部分模块、宏单元以及已预先布通的线网均被视为障碍,在布线过程中考虑障碍具有重要意义[7]。因此,研究人员进一步提出了绕障SMT(Obstacle-Avoding X-architecture Steiner Minimum Tree,OAXSMT)算法。在现代物理设计流程中,芯片可布线区域通常包括多层,障碍物通常仅分布于器件层及部分低层金属层中,并未阻挡所有布线层,因此导线仍可布设于这些障碍物上方的区域内[9]。然而,现有的SMT算法大多适用于解决无障碍情况或是完全绕障情况。在这种情况下,完全绕障的布线算法由于需避开障碍物,往往会带来布线资源的浪费。这种允许导线在一定程度上穿过障碍物的SMT问题被称为考虑障碍内布线资源的SMT问题。由于信号在导线的传输过程中会出现衰减与失真现象,因此需插入中继器以实现信号的再生与放大。考虑到障碍区域内无法放置中继器,因此在布线时需限制导线在障碍中的连续长度。通过在布线过程中引入长度约束,可在满足信号完整性要求的前提下,允许导线穿越障碍区域,从而节省障碍区域外部的布线资源,并有效缩短总线长。

本文将LS数量优化问题建模为每条源点到汇点布线路径中电源定域覆盖关系的分析问题，当路径中相邻引脚所属电源定域不满足覆盖关系时，将对应引脚集合抽象为一个电压簇，电压簇的存在对应LS的插入需求。现有的SMT算法未考虑LS，无法解决MDSV设计模式下的布线问题。同时，中继器所能驱动的最大导线长度随着中继器供电电压增加而增加[10]。在包含多种供电电压的MDSV设计模式中需考虑中继器驱动长度的不同，基于相同长度限制的算法易产生约束违规的布线结果，难以实现有效线长优化。此外，群智能算法在解决多峰值、复杂约束的优化问题中展现出显著优势[11-12]。麻雀搜索算法（Sparrow Search Algorithm，SSA）作为典型代表，凭借其出色的全局探索能力以及良好的可扩展性，已在各类优化问题中展现出广阔的应用前景[13-17]。同时，SSA通过引入不同角色的分工机制，使其在应对布线问题中的多峰特性与复杂约束空间时更具优势，能够有效避免陷入局部最优。

综上所述，为更贴合MDSV的设计模式，本文进一步引入了LS数量优化这一关键性能指标，并将减少LS数量的优化目标建模为簇减少问题，设计了一种考虑簇减少的可变长度限制X结构SMT算法（Variable Length-Restricted X-architecture Steiner Minimum Tree algorithm considering Cluster Reduction，VLRXSMT-CR），实现了线长与LS数量的协同优化，为MDSV设计模式下的低功耗物理设计提供了新的解决思路。主要贡献如下：

（1）针对引入MDSV设计模式产生的新布线问题，设计了离散学习算子、多样化麻雀调整策略、分层学习机制以及调整和精炼策略，对SSA进行系统离散化构建，使其能够有效求解离散Steiner树布线优化问题，最终设计了VLRXSMT-CR算法。

（2）针对布线线长与LS数量协同优化需求，设计了离散正向学习算子与变异算子，并引入竞争选择机制及基于拥塞度的外部存档更新机制，使算法能够有效求解MDSV设计模式下的多目标Steiner树优化问题。

（3）提出了多样化麻雀比例调整策略，以增强算法在不同搜索阶段的自适应能力，有效避免了算法过早陷入局部最优并加快了收敛速度。同时将分层学习思想引入搜索过程，通过构建多层学习结构，使得发现者与跟随者在更新过程中能够向更多麻雀个体学习，从而提高种群搜索多样性并增强全局搜索能力。

（4）设计了调整策略，用于对候选解中违反约束条件的布线边进行调整。同时设计了混合精炼策略，通过最大化公共边长度并消除环路结构，减少冗余连线并进一步降低布线树总线长。

（5）实验结果表明，本文算法在兼顾线长优化的同时，在MDSV设计模式下的关键性能指标，即在LS使用数量方面取得了显著的优化效果，能够更好地满足MDSV设计模式下布线过程中的设计约束要求。

1 相关工作

1.1 SMT

文献[18]提出了一种快速且高质量的直角SMT（Rectilinear Steiner Minimum Tree，RSMT）算法，但该算法在运行时内存消耗较大。文献[19]提出了一种有效的算法，与快速查找表估计（Fast Lookup Table Estimation，FLUTE）相比，可显著减少关键路径的线长和延迟。文献[20]首次提出了基于FLUTE的图形处理单元（Graphics Processing Unit，GPU）加速RSMT生成算法，与在40个中央处理单元（Central Processing Unit，CPU）内核上运行的FLUTE相比，运行加速高达10.47倍，填补了现有GPU加速框架在关键组件上的缺失。文献[21]提出了一种基于神经网络的RSMT算法框架NN-Steiner，该算法框架是第一个具有有限大小的神经架构，同时该算法框架具有易于训练、泛化能力强的特点，能够取得良好的线长优化效果。文献[22]提出了一种基于深度强化学习的统一方法，仅用一套模型即可同时构造曼哈顿结构与非曼哈顿结构的SMT。文献[23]基于粒子群优化（Particle Swarm Optimizer，PSO）算法，提出了一种高效的X结构SMT（X-architecture Steiner Minimum Tree，XSMT）两阶段构造算法，该算法包括社会学习离散PSO搜索和布线线长优化两个阶段。实验结果表明，该算法在线长缩短上展现出显著的优化能力，并且具有很强的稳定性。

尽管现有SMT构造方法在算法效率和线长优化方面取得了一定的进展，但上述研究大多基于理想化布线环境，通常假设布线区域连续且不存在物理障碍。在实际VLSI设计中，大量宏单元与功能模块会形成复杂的阻挡区域，使得传统Steiner树模型在实际布线过程中可能产生非法解。因此，如何在存在障碍物的情况下构建满足布线约束的Steiner树逐渐成为重要研究方向。

1.2 OAXSMT

考虑到在实际布线过程中往往存在障碍，OAXSMT问题逐渐成为研究热点之一。文献[24]首次提出了受多头绒泡菌启发的绕障直角SMT（Obstacle-Avoding Rectilinear Steiner Minimum Tree，OARSMT）布线算法，

对于给定的一组引脚和一组片上功能模块,可自动构造连接所有引脚的OARSMT。文献[25]提出了一种基于有效可满足性的方法,用于在给定障碍物集合的约束下,为指定引脚集构建OARSMT。该算法首先在不考虑障碍物的情况下构造生成树,并将线网分解为若干两端线网,然后针对考虑障碍的每个子线网,采用基于伪布尔可满足性的技术生成相应的绕障Steiner树。实验结果表明,该算法在诸多情况下能够产生更优的布线结果。文献[26]提出了一种基于深度强化学习的框架,能够自动学习和生成启发式算法来解决OARSMT问题。文献[27]提出了一种基于局部搜索策略的OARSMT算法,在提升解质量的同时兼顾了计算效率,取得了较好的优化效果。文献[28]提出一种基于离散化PSO的OAXSMT构建算法,使用基于并查集的交叉算子与变异算子对PSO算法进行改进,实现了PSO算法的离散化,同时引入了调整策略、精炼策略两种启发式策略,进一步增强了算法的搜索能力。

尽管现有绕障Steiner树研究在考虑障碍条件下的布线可行性与解质量优化方面取得了一定的进展,但上述方法通常将障碍物视为完全不可穿越区域,忽略了障碍物内部可能存在的可用布线资源。这种保守建模方式在一定程度上限制了布线空间的利用率,易导致布线线长增加。因此,近年来研究者开始探索允许利用障碍物内部布线资源的Steiner树构建方法,以进一步提升布线灵活性与整体性能。

1.3　考虑障碍内布线资源的SMT

在实际布线过程中,障碍通常位于设备层与较低的金属层,布线路径可由障碍上方通过。然而,障碍区域内无法放置中继器。当障碍区域布线路径过长时会出现信号失真问题。允许布线路由在一定程度上有选择地穿越障碍区域,可进一步优化SMT的线长。因此,有必要构造考虑障碍内布线资源的SMT问题模型。文献[29]提出了一种基于2-近似算法的长度限制曼哈顿SMT算法,该算法在最坏情况下的运行时间为$O\left((k\log_2 k)^2\right)$。实验结果表明,该算法在线长优化能力上相较于传统绕障算法,能够取得更好的优化结果,显著缩短布线路径长度,有效平衡时延与线长的优化需求。文献[7]提出了一种基于遗传操作改进PSO的X结构布线算法,结合遗传操作来解决离散的长度限制SMT问题,设计了一种候选点选择策略确保粒子能够满足约束条件,并在精炼阶段增加公共边的长度以优化布线树的总线长。实验结果表明,该算法在优化布线长度方面具有较好的性能。

现有研究通过允许有限度地利用障碍内部布线资源,在缩短布线线长方面取得了一定的进展。然而,随着低功耗设计需求的不断增强,MDSV设计模式被广泛应用。在此背景下,在Steiner树构造过程中考虑MDSV设计模式已成为实现高质量低功耗布线的重要研究方向。

1.4　MDSV设计模式

作为功耗优化的重要方法之一,目前已有诸多相关工作应用了MDSV设计。文献[30]将MDSV设计模式应用于雾计算中的终端节点,提出了能量最小化部分任务卸载(Energy-Minimized Partial Task Offloading,EMPTO)方案,实现动态调整终端节点的计算速度。理论证明和仿真结果表明,EMPTO能够在满足时延约束的条件下实现整体功耗的降低。MDSV设计模式的提出为VLSI物理设计阶段带来了新需求,包括布图规划、布局布线等。文献[31]首次提出了多电压(Multiple Supply Voltage,MSV)模式下的功耗驱动总体布线算法。然而,上述工作未考虑到MDSV设计下的可变长度约束条件,难以适用于解决MDSV设计下的总体布线问题。文献[10]首次提出了解决MDSV设计模式下总体布线问题的相关算法,定义了电源域感知布线(Power Domain-Aware Routing,PDR)问题并提出了点到点的PDR算法,然而该工作过于简化了MDSV设计模式下的数学问题,并未从降低功耗的目标出发进行相应的实验研究。

综上所述,现有关于SMT的研究尚未系统考虑到MDSV设计模式下的可变长度限制与LS开销,这种模型层面的缺失使得传统方法难以在多动态电源域环境下准确刻画实际布线约束,虽然易导致布线结果在线长方面具有一定优势,但会引入大量不必要的电平转换器,同时无法满足多动态电源电压设计模式下的可变长度约束要求,进而影响其时序性能与可靠性,降低布线结果在实际低功耗设计中的可用性。为此,针对引入MDSV设计模式产生的新布线问题,本文提出了VLRXSMT-CR。

2　初步工作

2.1　问题模型

考虑簇减少的可变长度限制XSMT问题,要求在满足可变导线长度约束的条件下,通过引入若干Steiner点为给定的n个引脚$P=\{P_1,P_2,\cdots,P_n\}$、k个障碍$B=\{B_1,B_2,\cdots,B_k\}$以及m个电源域$D=\{D_1,D_2,\cdots,D_m\}$构建一棵XSMT。该SMT需连接线网N中位于不同电源域内所有引脚。考虑簇减少的可变长度限制XSMT问题的优化目标如下:(1)优化Steiner树的总线长;(2)最小化布线树中电压簇的数量。设目标函数$f_1(X)$为个体X的线长,$f_2(X)$为X中电压簇的数量,

则考虑簇减少的可变长度限制XSMT问题定义为

$$\begin{aligned}&\min F(X)=\left(f_1(X),f_2(X)\right)^{\mathrm{T}}\\&\text{s.t. } s(X)-L_t\leqslant 0\end{aligned}\tag{1}$$

其中，$s(X)$为个体X的障碍内连通分量长度；$L_t(D_j)$为电源域D_j内中继器最大驱动长度。同时该XSMT中只允许包含0°、45°、90°与135°的边。

2.2 相关定义

定义1 驱动长度限制。中继器的驱动长度限制指在不引起信号失真或不违反设计时序要求的情况下，中继器所能驱动的最大导线长度，该长度随着中继器供电电压增加而增加[10]。

定义2 无中继器区域。由于中继器不能置于已关闭的电源域内，导线在进入无中继器区域后，连通分量长度需要满足中继器的驱动长度限制。如图1所示，当活动线网处于闲置模式时，电源域B_0、B_1和B_2处于关闭状态，则B_0、B_1和B_2为线网N的无中继器区域。

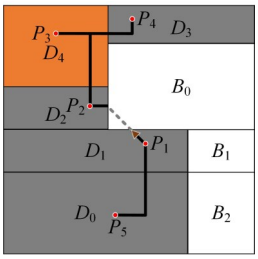


图1 MDSV布线例子

Figure 1 An example of routing under the MDSV design paradigm

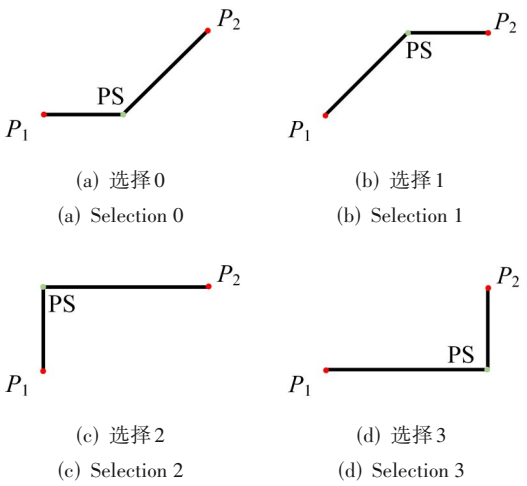
定义3 障碍。线网中存在的无中继器区域（如图1中的B_0、B_1和B_2）为布线过程中的障碍，在满足相关约束条件的前提下，布线可穿行于此类区域。本文规定障碍可为任意不重叠的矩形，障碍间允许点对点接触或边与边接触。

定义4 LS。当较低电压的驱动节点连接到较高电压的接收节点时，需在较高电压电源域边界插入LS，而由较高电压的驱动节点连接到较低电压接收节点时，则无需使用LS，如使用LS，则将在面积、时延与功耗上产生额外代价[10,32]。

定义5 引脚。引脚为线网在布线区域内且位于障碍外的待连接端点，如图1中P_1~P_5所示。

定义6 伪Steiner点。除引脚外的端点称为伪Steiner点。

定义7 X结构。采用X结构进行布线时允许在0°、45°、90°与135°四种方向上进行布线。相较于传统的直角结构，X结构包含更多可布线的方向，能够更好地利用布线资源。其中，0°与90°方向上的边称为直角边，45°与135°方向的边称为斜边。针对X结构的特点，本文规定了如定义8~定义11的四种连接选择，如图2所示，PS为伪Steiner点。



(a) 选择0 (b) 选择1

(a) Selection 0 (b) Selection 1

(c) 选择2 (d) 选择3

(c) Selection 2 (d) Selection 3

图2 连接方式

Figure 2 Connection patterns

定义8 选择0。如图2(a)所示，选择0通过直角边连接P_1与PS，再通过斜边连接PS与P_2。

定义9 选择1。如图2(b)所示，选择1通过斜边连接P_1与PS，再通过直角边连接PS与P_2。

定义10 选择2。如图2(c)所示，选择2通过垂直边连接P_1与PS，再通过水平边连接PS与P_2。

定义11 选择3。如图2(d)所示，选择3通过水平边连接P_1与PS，再通过垂直边连接PS与P_2。

定义12 Pareto支配。设X_a与X_b表示两个可行解，若满足以下关系：(1)在所有目标上X_a均不差于X_b；(2)在至少一个目标上X_a优于X_b。则X_a为Pareto占优于X_b，简称X_a支配X_b。

定义13 Pareto非支配解。解X_a被称为Pareto非支配解，当且仅当解集中不存在支配X_a的解。

2.2.1 中继长度限制约束

鉴于在障碍区域内部无法放置中继器，因此导线在障碍区域内的连续长度过长时将导致电容、转换或延迟违规[29]。本文规定当导线需进入无中继器区域时，将中继器放置于布线路径上临近电源域边界处，如图3所示。

在MDSV设计模式下，根据中继器所在的电源域电压不同，中继器最大驱动长度也随之不同，规定L_{high}为中继器位于高电压电源域内的最大驱动长度，L_{low}为中继器位于低电压电源域内的最大驱动长度。当导线在障碍区域内连续长度L大于最大驱动长度时，该布线边为非法边。

图3中边P_1P_2由电源域D_1进入障碍B_0，此时中继器放置于电源域D_1边界处，因此边P_1P_2在B_0内最大

中继长度由电源域D_1的供电电压决定。参考先前工作中的参数设置经验,本文在不同电压下中继器最大驱动长度根据布线区域较长边LBB的百分比进行设定,计算式为

$$L_{\text{high}}=\text{LBB}\times c_{\text{high}}$$
$$L_{\text{low}}=\text{LBB}\times c_{\text{low}} \tag{2}$$

其中,c_{high}为中继器位于高电压电源域内时对应的布线区域较长边LBB百分比;c_{low}为中继器位于低电压电源域内时对应的布线区域较长边LBB百分比。

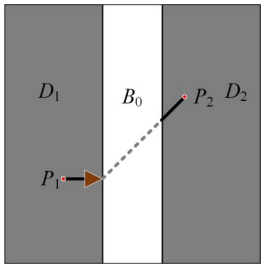
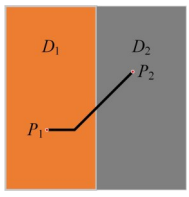


图3　中继器放置示意图

Figure 3　Illustration of repeater placement

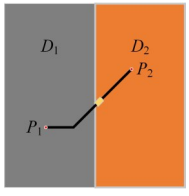
2.2.2　LS数量优化目标

由于LS在面积、延迟和功率方面会产生额外的代价,因此在布线过程中需减少由较低电压驱动节点连接到较高电压接收节点的情况。如图4所示,图4(a)中驱动节点P_1位于电压较高的电源域D_1,而接收节点P_2位于低电压电源域内D_2,此时无需引入LS;反之,图4(b)所示情况则需在目标电源域边界插入LS。



(a) 高电压域下的P_1连接到低电压域下的P_2

(a) P_1 in the high-voltage domain connected to P_2 in the low-voltage domain



(b) 低电压域下的P_1连接到高电压域下的P_2

(b) P_1 in the low-voltage domain connected to P_2 in the high-voltage domain

图4　LS放置示意图

Figure 4　Illustration of level shifter placement

2.2.3　总线长优化目标

鉴于总线长对芯片性能与功耗等关键指标具有显著影响,布线树的总线长已成为重要的优化目标之一。根据布线树中边违反约束的情况,计算总线长时采取不同的计算方式,总线长计算过程为

$$\text{Length}(X)=\sum_{e\in\text{Oseg}(X)}L_1(e)+\sum_{e\in\text{Dseg}(X)}L_2(e)+\sum_{e\in\text{Cseg}(X)}L_3(e) \tag{3}$$

其中,Length(X)为布线树X的总线长;$L_1(e)$为布线树X中边e的长度;$L_2(e)$为边e通过改变选择方式合法化后的长度;$L_3(e)$为边e通过引入障碍角点方式合法化后的长度。

3　算法设计及实现

本节基于分层学习多目标麻雀搜索方法,提出了一种考虑簇减少的可变长度限制XSMT构建算法。本节将从算法流程、优化目标、麻雀个体编码、预处理策略、多目标混合初始化策略、多目标麻雀搜索更新策略、调整策略、多目标混合精炼策略和时间复杂度分析九个方面展开介绍。

3.1　算法流程

本文所提出的VLRXSMT-CR算法主要分为以下五个阶段:

(1)预处理阶段。构造一个快速查找表,其中保存任意一对引脚通过任意选择连接时违反可变长度约束障碍物的信息,避免后续步骤中重复计算。

(2)多目标混合初始化阶段。基于最小生成树(Minimum Spanning Tree,MST)算法与随机算法构建初始种群。

(3)多目标麻雀搜索更新阶段。利用离散化更新算子对发现者、跟随者、随机选择的侦察者进行更新,应用基于拥塞度的外部存档集更新策略更新当前Pareto非支配解,同时通过竞争机制选出全局最优(gbest)作为引导搜索过程。

(4)调整阶段。通过调整边的选择方式或引入障碍角点,将违反可变长度约束的边转化为合法边。

(5)多目标混合精炼阶段。通过多种策略进一步优化得到的解,提高最终布线树的质量。

3.2　优化目标

3.2.1　布线树线长

如麻雀编码所示,每只麻雀代表一个候选Steiner树,考虑簇减少的可变长度限制XSMT问题的优化目标之一,在满足障碍区域内长度约束的前提下,最小化总线长。因此,本文在所设计的线长计算式中同时考虑了XSMT的总线长和违例边信息,设计的布线树线长适应度函数为

$$\mathrm{WL}(X_i)=\mathrm{Length}(X_i)\times\left(1+0.001\times\sum_{e_j\in X_i}\mathrm{excess}(e_j)\right) \quad (4)$$

其中，$\mathrm{Length}(X_i)$为麻雀X_i所表示的布线树的总线长；$\mathrm{excess}(e_j)$为边e_j的溢出长度，即边e_j在各个障碍内连续导线长度超过驱动长度约束的部分之和。

3.2.2 电压簇的个数

考虑到在MDSV设计模式中，由较低电压驱动节点连接到较高电压接收节点时需引入LS，因此本文设计的LS数量函数为

$$\begin{aligned}\mathrm{LS}(X_i)&=\sum_{e_j\in X_i}\sigma(p_j,q_j)\\ \sigma(p_j,q_j)&=\begin{cases}1, & \text{if } V_{p_j}<V_{q_j}\\ 0, & \text{otherwise}\end{cases}\end{aligned} \quad (5)$$

其中，$\mathrm{LS}(X_i)$为麻雀X_i所表示布线树中包含的电压簇个数，即需使用的LS数量；p与q分别为边e的驱动节点与接收节点；V_p与V_q分别为驱动节点p与接收节点q的电压。

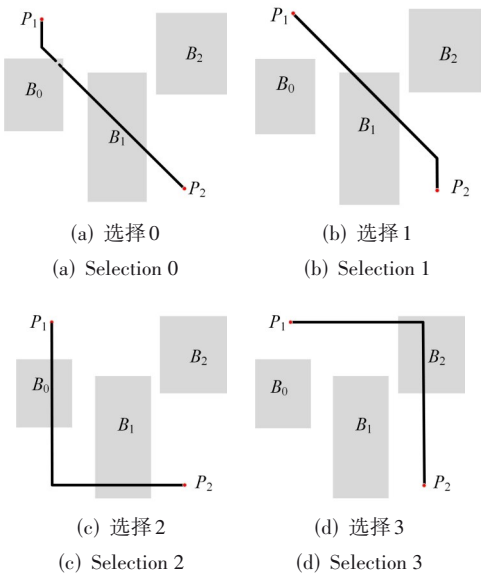
3.3 麻雀个体编码

对于一个包含n个引脚的麻雀个体，该麻雀代表的候选解由$n-1$条边构成。因此，该个体的编码需采用$3\times(n-1)$个数字表示边信息，同时麻雀编码中还需要一个适应度向量，分别反映麻雀个体的线长与LS个数，因此最终编码长度为$3\times(n-1)+2$。以图1所示麻雀为例，其编码为(1,2,1)、(2,3,2)、(3,4,3)、(1,5,2)、(27.15,2)，其中，(27.15,2)为该麻雀的适应度向量，27.15为线长，2为候选解中电压簇的个数。减少电压簇的个数即减少LS的使用，编码中前4组数表示了该麻雀中包含的边，如第一个对编码(1,2,1)表示该麻雀的第一条边由引脚P_1和引脚P_2通过连接选择1进行连接。

3.4 预处理策略

由优化目标计算式可知，当计算每只麻雀的优化目标时，不仅需考虑其代表XSMT的总线长，还需考虑其中的布线边是否合法，因此在求解不同麻雀优化目标时，往往需要反复使用各边的连接信息，如穿过障碍的情况等。若每次均需重新计算连接信息，则将产生大量的重复计算，严重影响算法的运行效率。因此，为避免重复计算带来的开销，本文设计了一个由关联数组组成的预处理查找表，通过在预处理阶段预先计算各引脚不同连接选择方式下的相关信息，以加速后续步骤中的计算过程。

以图5中引脚P_1和P_2为例，需计算二者在通过每种连接选择c相连时，所有障碍内连通分量长度超过L_i的障碍物，并将所有这些障碍信息记录在集合$\{O_k\}$中，构成最终查找表。如图5(a)所示，引脚P_1和P_2通过连接选择0穿过障碍物B_0与B_1，在障碍B_1内连通分量长度违反约束，但其在障碍物B_0内满足约束条件(以虚线表示)，因此在预处理表中记录障碍B_1。在图5(b)中，引脚P_1和P_2通过连接选择1穿过障碍物B_1，在障碍物B_1内的连通分量长度违反约束，因此在预处理表中需要记录障碍B_1。以此类推可以得到表1中的连接信息。



(a) 选择0 (b) 选择1

(a) Selection 0 (b) Selection 1

(c) 选择2 (d) 选择3

(c) Selection 2 (d) Selection 3

图5 P_1与P_2间四种连接选择示意图

Figure 5 Illustration of four connection options between P_1 and P_2

表1 包含引脚P_1与P_2的线网生成的预处理查找表

Table 1 Preprocessing lookup table for net generation containing pins P_1 and P_2

起点编号	终点编号	连接选择	违反约束个数	障碍列表
1	2	0	1	$\{B_1\}$
1	2	1	1	$\{B_1\}$
1	2	2	2	$\{B_0,B_1\}$
1	2	3	1	$\{B_2\}$

3.5 多目标混合初始化策略

高质量初始种群的构建是提升全局优化算法性能的关键基础，其多样性特征直接影响解空间的探索广度。当前现有算法依赖传统MST构造法生成初始解集，往往会导致种群多样性不足，这不仅限制了解空间的探索能力，还可能诱发搜索轨迹过早收敛于次优区域。针对这一问题，本文提出了一种混合初始化策略，采用多种策略生成初始种群，分别为基于MST算法与随机算法。设种群总数为N，初始化步骤如下所示：

步骤1：选择$0.8N$只麻雀，采用随机起点的Prim算法来构造MST，并由MST进一步构造初始解。

步骤2：选择剩余的麻雀，基于并查集随机选择

边构造MST,并进一步构造初始解。

步骤3:对N只麻雀进行非支配排序。

步骤4:更新个体最优(pbest),选择初始种群中非支配解作为初始外部存档集(Repository of External Population,REP)。

3.6　多目标麻雀搜索更新策略

3.6.1　离散化更新算子

SSA作为一种用于解决连续问题的群智能算法,最初是针对连续型问题提出的。然而,XSMT构建问题是离散化问题,因此算法引入正向交叉学习算子与变异算子对SSA更新算子进行离散化操作。

(1)正向学习算子

正向学习算子用于向更优的麻雀学习其优秀经验,以图6所示的两只麻雀X_i与X_j为例,正向学习过程如下:

步骤1:设定边的两个端点中较小的编号作为起点,较大的编号作为终点,调整X_i与X_j中的边,确保其起点编号小于终点编号并分别对各边排序。

步骤2:比较X_i与X_j,将X_i与X_j中相同的边加入集合S_1,并将剩余的边存储在集合S_2,得到的集合S_1与S_2分别如图7(a)与图7(b)所示。

步骤3:从集合S_2中随机选取边加入集合S_1,并利用并查集检查新的边引入是否会产生环,直到集合S_1构成一棵完整的新Steiner布线树,如图7(c)所示即为通过正向交叉学习产生的新布线树。

(2)变异算子

本文引入离散变异算子以代替SSA更新公式中的惯性部分,变异运算过程如下:

步骤1:从麻雀个体X_i中随机选择一条边$e_1(P,Q,c_1)$将其删除。

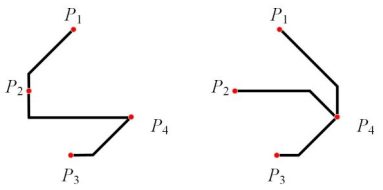
步骤2:扫描X_i中剩下的边,利用并查集将所有引脚划分为两个连通分量。

步骤3:分别从两个连通分量中随机选出两个引脚,记为P_1'与Q_1',同时通过随机方式选择新的连接方式c_1',得到新边$e_1'(P_1',Q_1',c_1')$,将e_1'加入X_i。

步骤4:接着从X_i中随机选择一条边e_2并移除这条边,同样将引脚划分为两个连通分量。

步骤5:分别从两个连通分量中随机选择引脚,记为P_2'与Q_2',并通过新的连接方式c_2'连接得到$e_2'(P_2',Q_2',c_2')$,将得到的边e_2'加入X_i。

经过上述步骤后得到的X_i即为变异得到的新布线结果。变异算子的运算过程如图8所示,算法选择边$e_1(1,2,1)$和边$e_2(2,4,2)$作为待变异的边,对e_1进行变换后得到新的边$e_1'(1,2,0)$,变换e_2后得到新$e_2'(2,3,0)$。

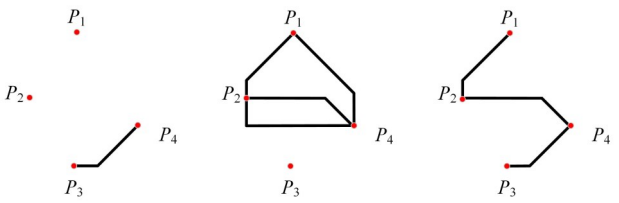


(a) 麻雀X_i　(b) 麻雀X_j

(a) Sparrow X_i　(b) Sparrow X_j

图6　麻雀X_i与X_j

Figure 6　Sparrow X_i and X_j

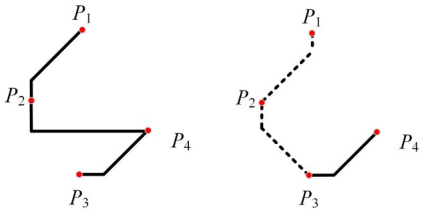


(a) 集合S_1　(b) 集合S_2　(c) 集合S_3

(a) Set S_1　(b) Set S_2　(c) Set S_3

图7　正向交叉学习示意图

Figure 7　Illustration of forward crossover learning



(a) 变异前　(b) 变异后

(a) Before mutation　(b) After mutation

图8　变异算子示意图

Figure 8　Illustration of mutation operator

3.6.2　发现者、侦查者动态调整机制

考虑到算法迭代初期需较强的全局搜索能力以规避早熟收敛风险,随着搜索进程推移,逐步增强局部求解能力来提升解的质量,本文算法设计了两种动态参数调节策略来调节种群中发现者数量和侦察者数量。

发现者个数N_{pro}的设定为

$$N_{\text{pro}}=N_{\text{pro}_{\max}}-(N_{\text{pro}_{\max}}-N_{\text{pro}_{\min}})\times\frac{t}{T} \tag{6}$$

其中,$N_{\text{pro}_{\max}}$与$N_{\text{pro}_{\min}}$分别为N_{pro}的最大值与最小值;t为当前迭代次数;T为算法的最大迭代次数。本文规定$N_{\text{pro}_{\max}}$与$N_{\text{pro}_{\min}}$分别为$0.25N$与$0.1N$。

侦察者个数N_{rec}的设定为

$$N_{\text{rec}}=\begin{cases}0.2N, & \text{if } t<\dfrac{2}{3}T\\ 0.35N, & \text{otherwise}\end{cases} \tag{7}$$

3.6.3 麻雀个体学习机制

为了提高迭代过程中麻雀种群的多样性，本文基于离散化正向学习算子与变异算子，设计了多种麻雀个体学习机制，具体如下：

(1)个体历史经验学习

麻雀X_i个体历史经验学习操作$C_1(X_i)$为

$$C_1(X_i)=C(X_i,\text{pbest}_i) \tag{8}$$

其中，pbest_i为麻雀X_i的个体历史最优；$C(X_a,X_b)$为X_a与X_b进行正向交叉学习。

(2)群体历史经验学习

群体历史经验学习操作$C_2(X_i)$为

$$C_2(X_i)=C(X_i,\text{gbest}) \tag{9}$$

(3)基于分层学习的群体历史经验学习

考虑到在麻雀搜索的过程中，发现者与跟随者在更新位置时依赖于全局最优值和局部最优值，为提高学习对象的多样性与解的多样性，本文在更新过程中对发现者群体进行排序与分层。发现者根据Pareto支配关系进行排序，而分层方式如图9所示。其中，每层中发现者个数$n_j(j=1,2,\cdots,L_{\max})$满足：

$$n_j=\begin{cases}2^{j-1}, & \text{if } j<L_{\max}\\ N_{\text{pro}}-2^{j-1}+1, & \text{if } j=L_{\max}\end{cases} \tag{10}$$

发现者个体X_i所属层$L(X_i)$满足：

$$L(X_i)=\lceil\log_2(i+1)\rceil \tag{11}$$

对应地，发现者总层数$L_{\max}=L(N_{\text{pro}})$。

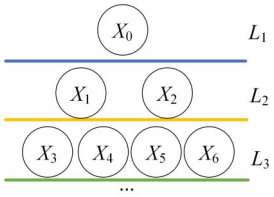


图9 发现者分层示意图

Figure 9 Illustration of discoverer hierarchy

在发现者更新过程中，第j层的发现者在第$j-1$层发现者中随机选择作为学习对象。当前发现者X_i分层学习操作$C_{\text{pro}}(X_i)$为

$$C_{\text{pro}}(X_i)=\begin{cases}C(X_i,X_t), & \text{if } i\neq 0\\ X_i, & \text{otherwise}\end{cases} \tag{12}$$

其中，学习对象X_t满足$X_t\in S\left(L(X_i)-1\right)$，$S(l)$为第$l$层中发现者构成的集合。

$S(l)$定义为

$$S(l)=\{X_j|j\in\mathbb{N}\wedge 2^{l-1}-1\leqslant j<2^l-1\} \tag{13}$$

其中，$\mathbb{N}$为自然数集合。

当个体X_i为跟随者时，分层学习操作$C_{\text{sub}}(X_i)$为

$$C_{\text{sub}}(X_i)=C(X_i,X_{\text{pro}}) \tag{14}$$

其中，X_{pro}为选择出的学习对象，选择步骤如下：

步骤1：设$L=L_{\max}$。

步骤2：从集合$\{1,2,\cdots,L\}$中随机选择l，根据式(13)计算$S(l)$，从中随机选择发现者X_{pro}。

步骤3：若X_{pro}不支配X_i且$j\neq 1$，则令$L=l$并转到步骤2，否则结束查找。

3.6.4 各角色更新机制

依据本文提出的离散化学习机制，算法为不同麻雀角色设计了相应更新方案，具体如下：

(1)发现者更新

在发现者更新阶段，算法选择适应度较高的N_{pro}只麻雀作为发现者进行更新。N_{pro}根据式(6)设置，目前发现者个体X_i^t正向交叉概率p_1与变异概率p_2根据式(15)设定。

$$p_1=p_2=\exp\left(\frac{t}{\alpha\times T}\right) \tag{15}$$

其中，α为随机浮点数且$\alpha\in(2,3)$。

根据当前警戒值R_2与随机警戒值R，由X_i^t得到新个体的过程为

$$X_i^{t+1}=\begin{cases}C_2\left(C_{\text{pro}}\left(C_1(X_i^t)\right)\right), & \text{if } R_2<\text{ST}\\ C_2\left(C_1\left(M(X_i^t)\right)\right), & \text{if } R_2\geqslant\text{ST and } R<\text{CR}_{\text{pro}}\\ C_2\left(C_{\text{pro}}\left(M(X_i^t)\right)\right), & \text{if } R_2\geqslant\text{ST and } R\geqslant\text{CR}_{\text{pro}}\end{cases} \tag{16}$$

其中，$R_2\in[0,1]$；$\text{ST}\in[0.5,1.0]$；M代表变异操作；CR_{pro}为变异系数且本文中取$\text{CR}_{\text{pro}}=0.3$。

(2)跟随者更新

在跟随者更新阶段，算法选择剩余的$N-N_{\text{pro}}$只麻雀作为跟随者进行更新。目前跟随者个体X_i^t正向交叉概率q_1与变异概率q_2基于其线长自适应调节，计算式为

$$q_1=q_2=\exp\left(-\frac{l_i-l_g}{l_N-l_g}\times\frac{1}{\alpha}\right) \tag{17}$$

其中，l_i为麻雀X_i^t的线长值；l_g为通过竞争选择机制选出全局最优解的线长值；l_N为麻雀X_N^t的线长值。

算法依据当前跟随者适应度值的不同采取不同的更新策略，计算式为

$$X_i^{t+1}=\begin{cases}C_2\left(C_{\text{sub}}\left(M(X_i^t)\right)\right), & \text{if } R<\text{CR}_{\text{sub}} \text{ or } i>\dfrac{N}{2}\\ C_2\left(M(X_i^t)\right), & \text{otherwise}\end{cases} \tag{18}$$

(3)侦察者更新

在侦察者更新阶段，算法从整个种群中随机选择N_{rec}只麻雀作为侦察者，算法根据目前跟随者适应度值的不同采取不同的更新策略，计算式为

$$X_i^{t+1}=\begin{cases}C_2\left(M(X_i^t)\right), & \text{if gbest}\succ X_i^t\\ C_2\left(C_1\left(M(X_i^t)\right)\right), & \text{otherwise}\end{cases} \tag{19}$$

3.6.5 竞争选择机制

在麻雀搜索的过程中,跟随者和意识到危险的麻雀在更新位置时会严重依赖于全局最优值和局部最优值,因此本文引入了一种竞争选择机制以提高种群多样性。在每轮迭代开始时,位置较好的麻雀间将随机进行两两竞争,并提供优胜的麻雀作为全局最优解来引导其他麻雀,具体如下:

步骤1:从外部存档集中选择N个麻雀作为候选解。

步骤2:对候选解与当前全局最优解的目标函数值进行归一化。

步骤3:对归一化后候选解的目标函数向量进行两两比较,选择与当前全局最优解夹角最小的麻雀作为胜者,胜者进入下一轮比较,直到选出最终胜者。

步骤4:将最终胜者作为新的全局最优解。

3.6.6 基于拥塞度的外部存档集更新

与单目标SSA不同,多目标优化算法的适应度为目标向量,无法进行直接比较。因此,本文将种群根据目标向量进行非支配排序,并在种群外设置一个REP,利用算法每次迭代所产生的非支配解更新REP。基于拥塞度的外部存档集更新过程如下:

步骤1:以线长为第一关键词,LS数量为第二关键词对麻雀种群进行非递减排序。

步骤2:遍历整个麻雀种群,检查种群中麻雀X_i是否被麻雀X_{i-1}支配,若是则跳过该麻雀,否则转到步骤3。

步骤3:检查外部存档是否存在解支配麻雀X_i,若存在则跳过X_i并转到步骤2,否则转到步骤4。

步骤4:检查外部存档是否存在解被麻雀X_i支配,移除被支配解并将麻雀X_i加入外部存档。

步骤5:若外部存档种群个体数达到或超过上限,则利用式(20)计算外部存档中解的拥塞度,并移除拥塞度值最小的解,直到外部存档种群个体数低于上限后转到步骤2。

$$\begin{aligned}
\mathrm{cong}_x(\mathrm{REP}_i)&=\min\left\{\frac{\left|\mathrm{WL}(\mathrm{REP}_{i-1})-\mathrm{WL}(\mathrm{REP}_i)\right|}{\left|\mathrm{WL}(\mathrm{REP}_{i+1})-\mathrm{WL}(\mathrm{REP}_i)\right|},\right.\\
&\left.\frac{\left|\mathrm{WL}(\mathrm{REP}_{i+1})-\mathrm{WL}(\mathrm{REP}_i)\right|}{\left|\mathrm{WL}(\mathrm{REP}_{i-1})-\mathrm{WL}(\mathrm{REP}_i)\right|}\right\}\\
\mathrm{cong}_y(\mathrm{REP}_i)&=\min\left\{\frac{\left|\mathrm{LS}(\mathrm{REP}_{i-1})-\mathrm{LS}(\mathrm{REP}_i)\right|}{\left|\mathrm{LS}(\mathrm{REP}_{i+1})-\mathrm{LS}(\mathrm{REP}_i)\right|},\right.\\
&\left.\frac{\left|\mathrm{LS}(\mathrm{REP}_{i+1})-\mathrm{LS}(\mathrm{REP}_i)\right|}{\left|\mathrm{LS}(\mathrm{REP}_{i-1})-\mathrm{LS}(\mathrm{REP}_i)\right|}\right\}\\
\mathrm{cong}(\mathrm{REP}_i)&=\mathrm{cong}_x(\mathrm{REP}_i)+\mathrm{cong}_y(\mathrm{REP}_i)
\end{aligned}\tag{20}$$

3.6.7 迭代更新流程

具体的迭代更新流程如下:

步骤1:判断当前迭代次数t是否小于T,若是则转到步骤3,否则结束迭代更新。

步骤2:根据竞争选择机制选出当前的gbest。

步骤3:根据式(6)设置发现者个数N_{pro},并根据式(16)更新发现者个体。

步骤4:根据式(18)更新剩余跟随者个体。

步骤5:根据式(7)设置侦察者个数,随机选择侦察者并根据式(19)更新侦察者个体。

步骤6:回到步骤1。

3.7 调整策略

经过迭代更新后,可获得一个较优的解gbest,然而在gbest中仍可能包含某些违反约束的边。如图10(a)所示,边$(P,Q,2)$在障碍B_2内连通分量违反约束。针对这种情况,本文采用调整策略对gbest进行调整,从而使gbest所代表布线树中包含的边均符合约束条件。调整过程如下所示:

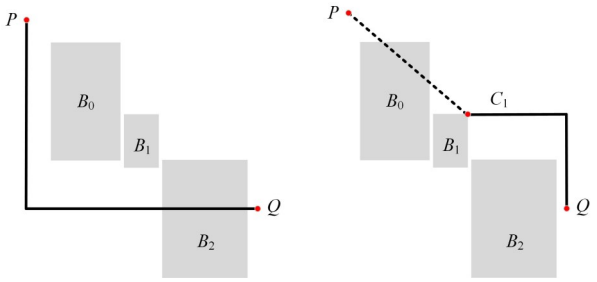
步骤1:基于查找表检查gbest中的每一条边PQ在各障碍区域内连通分量长度。若边PQ在各障碍内连通分量均满足MDSV设计模式下约束条件则重复检查下一条边,直到gbest中的每一条边均为合法边,若边PQ的障碍内长度违反约束则执行步骤2。

步骤2:从gbest中移除边PQ,根据查找表列出边PQ穿过且障碍内连续长度违反约束的相关障碍$B=\{B_1,B_2,\cdots,B_n\}$并根据起点P与各障碍中心之间的距离对集合B进行升序排序,得到新序列$B'=\{B'_1,B'_2,\cdots,B'_n\}$,并令$P$作为当前起点$S$。

步骤3:在B'_k所对应的障碍中,选择距直线SQ最近的角点并记为C_k,计算边SC_k连接关系并加入到表中,依据选择0与选择1优先级高于选择2与选择3的准则,确定SC_k的最终连接方式并加入gbest,设$S=C_k$。若当前边SQ满足约束条件或$k=n$则转到步骤4,否则$k=k+1$并返回步骤3。

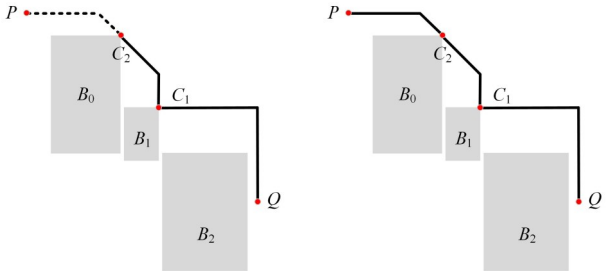
步骤4:计算SQ的连接信息并将其加入表中,优先根据选择0或选择1将SQ连接加入gbest。

本文以图10为例说明该调整策略。图10展示了当引脚P与Q通过选择2连接时的调整过程。由图10(a)可知,边$(P,Q,2)$属于违例边且其在障碍B_2内长度限制违反约束,故删除边PQ并添加边$(P,C_1,0)$与$(C_1,Q,3)$得到如图10(b)所示结果。可以发现,边$(P,C_1,0)$仍然违反约束条件,因此在后续步骤中再次对边$(P,C_1,0)$进行调整,并添加角点C_2得到如图10(c)所示结果。因此,图10(a)中的违例边经过调整后得到如图10(d)所示的最终结果。



(a) 调整前 (b) 加入角点 C_1

(a) Before adjustment (b) Adding corner point C_1



(c) 加入角点 C_2 (d) 调整结束

(c) Adding corner point C_2 (d) After adjustment

图 10 调整过程

Figure 10 Adjustment process

3.8 多目标混合精炼策略

经调整可得到一个满足障碍内可变长度约束的合法候选解。然而,该候选解通常尚未达到最优结构,与实际最优的 XSMT 存在一定差异。因此,本文设计了一种混合精炼策略对得到的全局最优解中包含的某些局部拓扑结构进行优化。

3.8.1 基于公共边的局部优化

在规模较大的布线树中,某一局部范围内通常存在多条布线路径,这些布线路径通过改变连接选择往往能够增加公共边的长度,从而带来更好的线长优化效果。以三引脚线网为例,三引脚存在 16 种选择组合,不同连接选择组合下公共边长度不同。图 11 展示了当连接方案中公共边长度达到最大值时,总线长实现最小化。基于公共边的局部优化目标是将布线树中所包含的所有非最优结构转化为最优结构,通过最大化公共边长度来减小总线长。以图 12 为例,图 12(a)中的布线树优化前线长约为 824.26,而经过优化后得到的布线树如图 12(b)所示,其总线长为 724.26,相较于优化前缩短了 12.13%。

基于公共边的局部优化步骤如下:

步骤 1:根据 gbest 中的边计算所有引脚与 Steiner 点的度与邻接边信息。

步骤 2:依次枚举各引脚及其邻接边,检查是否存在比当前 gbest 更优的连接选择组合,若不存在则继续检查其他引脚,否则转到步骤 3。

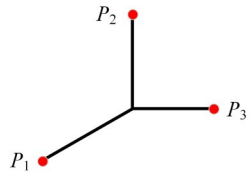
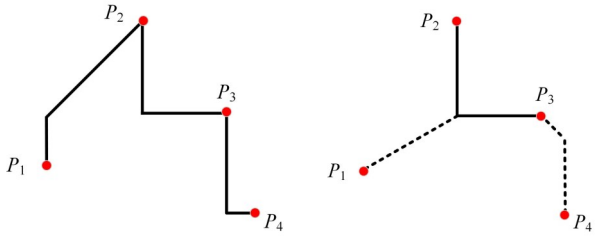


图 11 三引脚最优拓扑结构

Figure 11 Optimal topology for three pins



(a) 优化前 (b) 优化后

(a) Before optimization (b) After optimization

图 12 基于公共边的局部优化示意图

Figure 12 Illustration of local optimization based on common edges

步骤 3:基于找到的新连接选择组合方式更新 gbest,转到步骤 2。

3.8.2 边交叉检测与消除

如图 13(a)所示,可以发现,该布线树中三条布线边形成了一个交叉环路,分别是$(P_1, C_1, 2)$、$(P_2, P_3, 1)$与$(C_1, P_2, 0)$,但从树的编码角度无法发现这一问题。因此,本文提出了一种边交叉检测与消除策略,以实现消除线网中存在的导线相交情况,并进一步优化线长。

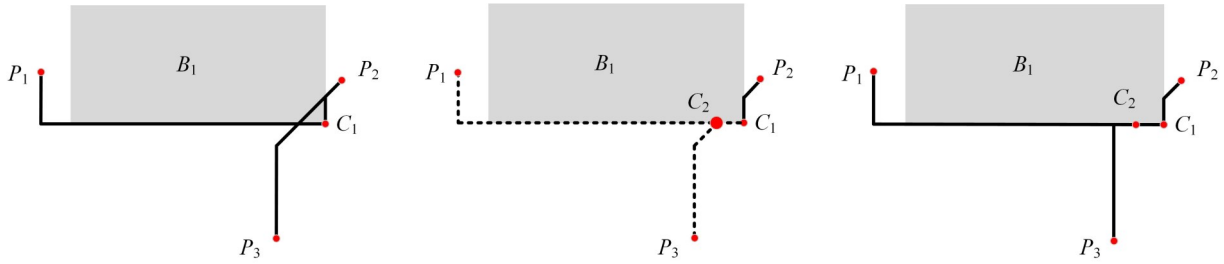
针对这种情况,可通过添加新 Steiner 点并连接相关引脚来解决。以图 13(a)为例,首先,将边$(P_1, C_1, 2)$与$(P_2, P_3, 1)$的交点作为新的引脚 C_2;其次,删除交叉的两条边$(P_1, C_1, 2)$与$(P_2, P_3, 1)$;最后,加入新的边$(P_3, C_2, 0)$、$(P_1, C_2, 2)$与$(C_2, C_1, 0)$重新构成完整树结构,得到如图 13(b)所示的新布线树。

经上述调整后,相关边的连接方式可通过基于公共边的局部优化进行进一步改进。如图 13(b)可以进一步优化为如图 13(c)所示结果。同时考虑到本文算法优化目标同时包括平均线长(Wire Length, WL)与 LS,算法将根据式(21)计算候选解评分并选择评分最小的方案实现消除相交。

$$\mathrm{Score}(X')=\frac{\mathrm{WL}_{X'}-\mathrm{WL}_{\min}}{\mathrm{WL}_{\max}-\mathrm{WL}_{\min}}+\frac{\mathrm{LS}_{X'}-\mathrm{LS}_{\min}}{\mathrm{LS}_{\max}-\mathrm{LS}_{\min}} \tag{21}$$

3.9 时间复杂度分析

定理 1 设种群大小为 N,算法迭代次数为 t,引脚个数为 n,障碍个数为 m,则 VLRXSMT-CR 算法的



(a) 发生交叉的情况　(b) 消除交叉后　(c) 进一步优化

(a) With edge crossing　(b) After removing crossing　(c) Further optimization

图13　发生边交叉的例子

Figure 13　Example of edge crossing

时间复杂度为 $O(ntN\log_2 n+tN^2+m^2n+mn^2+n^2\log_2 n)$。

证明　(1)在多目标混合初始化策略阶段,本文采用Prim算法和随机生成算法构建生成树,并采用生成树中的$(n-1)$条边初始化麻雀种群。其中,Prim算法初始化麻雀种群的时间复杂度为$O(n^2+nN)$,当随机生成算法构建生成树时基于改进并查集算法检测环,平均情况下的时间复杂度为$O(n)$。综上所述,多目标混合初始化策略阶段时间复杂度为$O(n^2+nN)$。

(2)在预处理阶段,需计算各引脚对通过不同连接选择进行连接形成的边与m个障碍的相交信息,同时考虑到边间的成对关系,故需计算的边个数为$2n(n-1)$,时间复杂度为$O(mn^2)$。

(3)在多目标麻雀搜索更新阶段,内部循环主要由交叉变异、适应度评估及种群排序构成。其中,变异操作中边的删除与新边生成时间复杂度为常数级,正向交叉对树结构的遍历时间复杂度为线性级,但在查找公共边时需对边进行排序,排序时间复杂度为$O(n\log_2 n)$,同时在交叉变异操作中需使用改进并查集算法,最好情况下的时间复杂度为$O(n)$,平均情况下的时间复杂度为$O(n\log_2 n)$。而在适应度计算时同样需进行排序,在每轮迭代中对麻雀种群的排序时间复杂度为$O(N\log_2 N)$。在外部存档集更新阶段,需要检查REP中的解与当前种群Pareto非支配解的支配关系,时间复杂度为$O(N^2)$。综上所述,在更新阶段中的时间复杂度为$O(ntN\log_2 n+tN^2)$。

(4)在基于障碍内长度的调整策略中,算法在最坏情况下,需调整$(n-1)$条边,并检测其与障碍物的相交关系,其时间复杂度为$O(m^2n)$。

(5)多目标混合精炼策略由两部分组成,即基于公共边的局部优化和边交叉检测与消除。其中,基于公共边的局部优化需枚举每个引脚的相关边并选择最优的连接选择方式组合,同时还需计算适应度值,考虑到相关边数量过多时的计算代价,算法将最大相关边数量限制为常量,因此时间复杂度为$O(n^2\log_2 n)$。在边交叉检测与消除过程中,需对所有边对进行位置关系判断,因而需执行$(n-1)(n-2)/2$次检测,故这一过程时间复杂度为$O(n^2)$。因此,混合精炼策略的时间复杂度为$O(n^2\log_2 n)$。

综上所述,VLRXSMT-CR算法的时间复杂度为$O(ntN\log_2 n+tN^2+m^2n+mn^2+n^2\log_2 n)$。证毕

4　实验结果

本文算法使用C++语言实现,采用G++11.3进行编译,测试环境为4 GHz CPU和16 GB内存的PC。本文使用多组基准作为测试用例,其中前五组测试用例(ind1~ind5)是Synopsys的工业测试用例,其余(rc01~rc10)为常用绕障问题测试用例。为了验证本文算法的有效性,实验条件的设置参考文献[9]与文献[33],将驱动长度限制设置为布线区域长边LBB的不同比例,如表2所示,其中实验条件1为完全绕障的情况。考虑到算法的随机性,以下所有实验结果均为独立运行30次后求得的平均值。

表2　测试条件设置

Table 2　Experimental settings

描述	低电压电源域长度限制L_{Low}	高电压电源域长度限制L_{High}
实验条件1	0	0
实验条件2	LBB × 1%	LBB × 5%
实验条件3	LBB × 1%	LBB × 10%
实验条件4	LBB × 5%	LBB × 10%

4.1　多目标混合初始化策略有效性验证

表3给出了在不同实验条件下多目标混合初始化策略与Prim方法在WL和LS数量上的对比结果。如表3所示,在实验条件1下,所提出策略的WL优化率介于0.18%~4.88%,平均优化率达到1.93%;LS数量优化率介于−7.84%~61.50%,平均优化率达到17.05%。在实验条件2下,WL优化率范围为0.00%~

3.00%，平均优化率为1.37%；LS数量优化率范围为-5.71%~25.00%，平均优化率为5.69%。

从整体趋势来看，所提出的多目标混合初始化策略在大多数测试电路中均能同时兼顾WL与LS数量两个优化目标。尤其在考虑障碍物的条件下，由于布线空间受限，传统Prim方法容易产生局部最优结构，而本文通过引入多目标混合初始化策略，能够在初始解阶段构建更加均衡的拓扑结构，从而为后续优化过程提供更优的搜索起点，在保证WL基本不增加的前提下，显著减少LS的使用数量。

表3 多目标混合初始化策略有效性验证

Table 3 Effectiveness verification of the multi-objective hybrid initialization strategy

测试电路	实验条件1						实验条件2					
	WL/μm			LS			WL/μm			LS		
	Prim	OURS	优化率/%	Prim	OURS	优化率/%	Prim	OURS	优化率/%	Prim	OURS	优化率/%
ind1	564	563	0.18	1.02	1.10	−7.84	563	563	0.00	1.00	1.02	−2.00
ind2	9 592	9 530	0.65	2.85	1.98	30.53	8 819	8 815	0.05	1.07	1.00	6.54
ind3	553	551	0.36	1.00	1.00	0.00	552	550	0.36	1.00	1.00	0.00
ind4	960	957	0.31	1.10	1.10	0.00	955	955	0.00	1.00	1.00	0.00
ind5	1 169	1 150	1.63	1.00	1.00	0.00	1 158	1 153	0.43	1.00	1.00	0.00
rc01	26 770	26 616	0.58	1.38	1.43	−3.62	24 097	24 086	0.05	1.00	1.00	0.00
rc02	40 797	40 352	1.09	3.42	3.12	8.77	39 832	39 371	1.16	3.55	3.30	7.04
rc03	52 958	51 686	2.40	4.67	4.42	5.35	52 853	51 268	3.00	3.50	3.70	−5.71
rc04	61 624	60 687	1.52	3.48	2.67	23.28	61 394	60 158	2.01	3.30	2.77	16.06
rc05	74 657	73 496	1.56	6.45	5.47	15.19	71 129	70 372	1.06	5.20	4.72	9.23
rc06	81 175	78 290	3.55	23.65	12.75	46.09	76 290	74 625	2.18	1.60	1.20	25.00
rc07	108 773	106 220	2.35	41.05	24.65	39.95	103 030	100 350	2.60	9.80	10.05	−2.55
rc08	119 270	113 452	4.88	48.95	43.36	11.42	106 284	103 759	2.38	1.77	1.62	8.47
rc09	116 558	111 410	4.42	30.73	11.83	61.50	107 262	104 327	2.74	2.57	2.17	15.56
rc10	161 440	155 874	3.45	16.87	12.63	25.13	157 937	154 034	2.47	12.57	11.60	7.72
平均值			1.93			17.05			1.37			5.69

4.2 多目标混合精炼策略有效性验证

表4给出了在不同实验条件下多目标混合精炼策略在WL和LS数量两个指标上的优化效果。如表4所示，在实验条件1下，所提出策略的WL优化率介于0.72%~8.72%，平均优化率达到5.06%；LS数量优化率介于0.00%~61.30%，平均优化率达到19.17%。在实验条件2下，WL优化率范围为0.15%~8.99%，平均优化率为4.50%；LS数量优化率范围为-6.35%~17.31%，平均优化率为4.54%。

在不同实验条件下，由于障碍的存在，得到候选解容易产生冗余连接和不合理的LS配置，而本文提出的多目标混合精炼策略通过对局部拓扑结构进行针对性调整，有效缩短布线WL并减少LS的使用数量。整体结果表明，所提出的多目标混合精炼策略在两个优化指标上均表现出更加稳定的优化效果，说明该混合精炼策略在复杂布线环境中能够有效挖掘解空间中的潜在优化机会。

4.3 算法有效性验证

为了验证本文所提出算法的有效性，本文选取文献[23]和文献[34]的算法作为对比方法，并从布线WL和LS数量性能指标对实验结果进行分析与讨论。

4.3.1 与文献[23]对比

表5展示了在不同实验条件下本文算法与文献[23]算法的对比情况。如表5所示，在实验条件1下，本文算法的WL优化率介于-2.84%~15.13%，平均优化率达到了3.47%；LS数量指标优化率为-43.00%~77.62%，平均优化率达到了38.60%。在实验条件2下，本文算法的WL优化率为-5.40%~3.25%，平均优化率为-0.74%；LS数量指标优化率为0.00%~90.73%，平均优化率达到了49.90%。

从整体结果来看，相较于文献[23]的方法，本文算法在大多数测试电路上能够显著降低LS数量，同时在布线WL方面保持较高水平的优化能力。尤其在实验条件1下，由于布线空间受限，传统方法容易产生冗余路径和过多的LS配置，而本文算法通过引入多目标协同优化机制，在搜索过程中综合考虑布线WL与LS数量之间的关系，从而获得更加均衡的布线拓扑结构。

表4　多目标混合精炼策略有效性验证

Table 4　Effectiveness verification of the multi-objective hybrid refinement strategy

测试电路	实验条件1						实验条件2					
	WL/μm			LS			WL/μm			LS		
	无精炼	有精炼	优化率/%	无精炼	有精炼	优化率/%	无精炼	有精炼	优化率/%	无精炼	有精炼	优化率/%
ind1	568	563	0.88	1.18	1.10	6.78	564	563	0.18	1.10	1.02	7.27
ind2	9 714	9 530	1.89	2.85	1.98	30.53	8 828	8 815	0.15	1.05	1.00	4.76
ind3	555	551	0.72	1.00	1.00	0.00	557	550	1.26	1.00	1.00	0.00
ind4	964	957	0.73	1.10	1.10	0.00	1 022	955	6.56	1.00	1.00	0.00
ind5	1 229	1 150	6.43	1.00	1.00	0.00	1 166	1 153	1.11	1.00	1.00	0.00
rc01	27 098	26 616	1.78	1.43	1.43	0.00	24 171	24 086	0.35	1.00	1.00	0.00
rc02	41 897	40 352	3.69	3.65	3.12	14.52	40 624	39 371	3.08	3.27	3.30	–0.92
rc03	55 238	51 686	6.43	5.08	4.42	12.99	53 806	51 268	4.72	3.70	3.70	0.00
rc04	65 091	60 687	6.77	3.48	2.67	23.28	64 144	60 158	6.21	3.35	2.77	17.31
rc05	78 952	73 496	6.91	6.17	5.47	11.35	74 094	70 372	5.02	5.30	4.72	10.94
rc06	83 610	78 290	6.36	24.68	12.75	48.34	80 017	74 625	6.74	1.40	1.20	14.29
rc07	114 730	106 220	7.42	41.30	24.65	40.31	108 416	100 350	7.44	9.45	10.05	–6.35
rc08	124 103	113 452	8.58	49.00	43.36	11.51	112 207	103 759	7.53	1.73	1.62	6.36
rc09	122 056	111 410	8.72	30.57	11.83	61.30	114 636	104 327	8.99	2.53	2.17	14.23
rc10	170 449	155 874	8.55	17.23	12.63	26.70	167 692	154 034	8.14	11.63	11.60	0.26
平均值			5.06			19.17			4.50			4.54

表5　本文算法与文献[23]算法对比

Table 5　Comparison between the proposed algorithm and the method in reference [23]

测试电路	实验条件1						实验条件2					
	WL/μm			LS			WL/μm			LS		
	文献[23]	OURS	优化率/%	文献[23]	OURS	优化率/%	文献[23]	OURS	优化率/%	文献[23]	OURS	优化率/%
ind1	562	563	–0.18	2.00	1.10	45.00	562	563	–0.18	1.77	1.02	42.37
ind2	9 689	9 530	1.64	2.00	1.98	1.00	8 814	8 815	–0.01	2.00	1.00	50.00
ind3	579	551	4.84	1.00	1.00	0.00	558	550	1.43	1.00	1.00	0.00
ind4	1 033	957	7.36	2.65	1.10	58.49	966	955	1.14	2.12	1.00	52.83
ind5	1 355	1 150	15.13	4.00	1.00	75.00	1 170	1 153	1.45	3.12	1.00	67.95
rc01	25 880	26 616	–2.84	1.00	1.43	–43.00	24 052	24 086	–0.14	1.00	1.00	0.00
rc02	42 035	40 352	4.00	5.60	3.12	44.29	40 694	39 371	3.25	5.58	3.30	40.86
rc03	54 003	51 686	4.29	7.28	4.42	39.29	52 109	51 268	1.61	7.00	3.70	47.14
rc04	60 462	60 687	–0.37	11.93	2.67	77.62	57 075	60 158	–5.40	10.97	2.77	74.75
rc05	72 526	73 496	–1.34	11.62	5.47	52.93	68 375	70 372	–2.92	9.28	4.72	49.14
rc06	78 706	78 290	0.53	25.12	12.75	49.24	73 019	74 625	–2.20	12.95	1.20	90.73
rc07	106 307	106 220	0.08	26.90	24.65	8.36	99 253	100 350	–1.11	16.10	10.05	37.58
rc08	—	113 368	—	—	26.50	—	102 404	103 759	–1.32	5.03	1.62	67.79
rc09	130 648	111 410	14.73	33.63	11.83	64.82	99 279	104 327	–5.08	16.07	2.17	86.50
rc10	156 944	155 874	0.68	38.70	12.63	67.36	151 491	154 034	–1.68	19.63	11.60	40.91
平均值			3.47			38.60			–0.74			49.90

4.3.2　与文献[34]对比

表6和表7展示了在不同条件下与文献[34]算法的对比情况。总体来看,本文算法在保持布线WL基本稳定的前提下,能够显著降低LS数量。具体实验结果如下:在实验条件1下,本文算法的WL优化率为–4.20%~16.12%,平均优化率达到了3.50%;LS数量指

表6 本文算法与文献[34]算法对比

Table 6 Comparison between the proposed algorithm and the method in reference [34]

测试电路	实验条件1						实验条件2					
	WL/μm			LS			WL/μm			LS		
	文献[34]	OURS	优化率/%	文献[34]	OURS	优化率/%	文献[34]	OURS	优化率/%	文献[34]	OURS	优化率/%
ind1	578	563	2.60	2.00	1.10	45.00	577	563	2.43	2.00	1.02	49.00
ind2	9 633	9 530	1.07	2.00	1.98	1.00	8 814	8 815	−0.01	2.00	1.00	50.00
ind3	592	551	6.93	1.00	1.00	0.00	571	550	3.68	1.00	1.00	0.00
ind4	1 071	957	10.64	2.85	1.10	61.40	1 001	955	4.60	2.30	1.00	56.52
ind5	1 371	1 150	16.12	4.20	1.00	76.19	1 191	1 153	3.19	3.00	1.00	66.67
rc01	25 543	26 616	−4.20	1.62	1.43	11.73	24 217	24 086	0.54	1.00	1.00	0.00
rc02	42 242	40 352	4.47	7.35	3.12	57.55	41 712	39 371	5.61	7.35	3.30	55.10
rc03	56 461	51 686	8.46	7.28	4.42	39.29	53 877	51 268	4.84	7.10	3.70	47.89
rc04	60 498	60 687	−0.31	12.00	2.67	77.75	57 189	60 158	−5.19	11.00	2.77	74.82
rc05	72 369	73 496	−1.56	12.35	5.47	55.71	68 221	70 372	−3.15	9.50	4.72	50.32
rc06	78 880	78 290	0.75	25.32	12.75	49.64	73 122	74 625	−2.06	13.00	1.20	90.77
rc07	106 255	106 220	0.03	27.75	24.65	11.17	99 209	100 350	−1.15	16.20	10.05	37.96
rc08	—	113 368	—	—	43.36	—	102 403	103 759	−1.32	5.10	1.62	68.24
rc09	—	111 410	—	—	11.83	—	99 242	104 327	−5.12	16.13	2.17	86.55
rc10	156 634	155 874	0.49	44.30	12.63	71.49	151 745	154 034	−1.51	21.03	11.60	44.84
平均值			3.50			42.92			0.36			51.91

表7 本文算法与文献[34]算法对比

Table 7 Comparison between the proposed algorithm and the method in [34]

测试电路	实验条件3						实验条件4					
	WL/μm			LS			WL/μm			LS		
	文献[34]	OURS	优化率/%	文献[34]	OURS	优化率/%	文献[34]	OURS	优化率/%	文献[34]	OURS	优化率/%
ind1	565	563	0.35	2.00	1.12	44.00	565	563	0.35	2.00	1.00	50.00
ind2	8 814	8 813	0.01	2.00	1.07	46.50	8 814	8 813	0.01	2.00	1.05	47.50
ind3	554	550	0.72	1.00	1.00	0.00	554	550	0.72	1.00	1.00	0.00
ind4	974	955	1.95	2.00	1.00	50.00	974	956	1.85	2.00	1.02	49.00
ind5	1 182	1 153	2.45	3.00	1.00	66.67	1 181	1 154	2.29	3.00	1.00	66.67
rc01	24 210	24 070	0.58	1.00	1.00	0.00	24 220	24 057	0.67	1.00	1.00	0.00
rc02	38 871	39 662	−2.03	6.65	3.00	54.89	38 921	38 789	0.34	6.75	2.92	56.74
rc03	50 970	51 483	−1.01	7.05	3.50	50.35	51 074	49 826	2.44	7.25	3.50	51.72
rc04	52 413	55 903	−6.66	11.00	2.55	76.82	52 500	52 760	−0.50	11.00	2.40	78.18
rc05	68 132	69 255	−1.65	9.18	5.00	45.53	68 136	68 878	−1.09	9.25	4.88	47.24
rc06	73 080	74 651	−2.15	13.00	0.82	93.69	73 058	72 903	0.21	13.00	2.35	81.92
rc07	99 077	100 583	−1.52	16.00	9.25	42.19	99 090	99 442	−0.36	16.00	11.03	31.06
rc08	102 353	103 706	−1.32	5.00	1.40	72.00	102 343	102 496	−0.15	5.00	2.95	41.00
rc09	99 221	104 558	−5.38	16.00	2.23	86.06	99 218	99 340	−0.12	16.00	3.70	76.88
rc10	151 373	153 896	−1.67	19.93	11.83	40.64	151 370	151 621	−0.17	19.93	16.19	18.77
平均值			−1.15			51.29			0.43			46.45

标的优化率为0.00%~77.75%，平均优化率达到了42.92%。在实验条件2下，本文算法的WL优化率为−5.19%~5.61%，平均优化率达到了0.36%；LS数量指标优化率为0.00%~90.77%，平均优化率达到了51.91%。在实验条件3下，WL优化率为−6.66%~2.45%，平均优化率为−1.15%；LS数量指标优化率为

0.00%~93.69%,平均优化率达到了51.29%。在实验条件4下,WL优化率为-1.09%~2.44%,平均优化率为0.43%;LS数量指标优化率为0.00%~81.92%,平均优化率达到了46.45%。综合实验结果表明,本文方法通过引入可控的WL代价,在满足布线质量要求的前提下显著降低LS数量,验证了所提出方法在MDSV设计中的有效性。

需要指出的是,在部分测试电路中WL指标出现了轻微的负优化现象,但整体ML变化幅度较小,而LS数量仍获得了显著优化。这一现象主要源于本文算法在MDSV设计环境下构建的多目标协同优化模型,其中布线WL与LS数量被同时纳入优化目标,在搜索过程中对二者进行动态权衡与平衡优化。在MDSV设计中,LS数量直接影响芯片的面积和功耗,相较于WL的微小波动,其优化在工程实践中具有更高的综合价值。

5　结束语

现有的SMT算法研究工作尚未考虑MDSV设计模式带来的新约束条件与引入LS所带来的额外开销,因此寻找一种能够有效解决MDSV设计模式下布线问题的可变长度限制SMT算法具有重要的意义。本文针对MDSV设计模式下可变长度约束布线问题,提出了一种VLRXSMT-CR。首先,算法通过离散化的交叉变异算子对SSA进行离散化,并通过竞争选择机制与基于拥塞度的外部存档更新机制实现多目标化,使其适用于求解离散化的MDSV设计模式下可变长度限制SMT问题。其次,为克服SSA易陷入局部最优解的问题,算法在更新阶段引入分层学习机制与参数动态调节机制,从而有效提升了算法性能。最后,算法通过基于障碍内长度的调整策略与混合精炼策略对得到的X结构Steiner最小树进行进一步优化,在缩短总线长的同时使布线结果能够满足MDSV设计约束。实验结果表明,与现有工作相比,本文算法能够在满足MDSV设计约束的同时,在WL与LS指标两个方面取得良好的优化效果。

参考文献

[1] Zheng Huajian, Ye Zhuohang, Liu Baiquan, et al. Dual-gate metal-oxide-semiconductor transistors: Nanoscale channel length scaling and performance optimization[J]. Electronics, 2025, 14(7): 1257.

[2] Razif R A M, Maharum S M M, Hasani A H, et al. Mitigation techniques for crosstalk in ICs[J]. IOP Conference Series: Materials Science and Engineering, 2019, 701(1): 012037.

[3] Chauhan P, Gupta S. Challenges and future perspectives of low-power VLSI circuits: A study[M]//Modern Electronics Devices and Communication Systems. Singapore: Springer Nature Singapore, 2023: 561-569.

[4] Varadharajan S K, Nallasamy V. Low power VLSI circuits design strategies and methodologies: A literature review[C]// 2017 Conference on Emerging Devices and Smart Systems (ICEDSS). Piscataway: IEEE, 2017: 245-251.

[5] Kao C C. Clock skew minimization in multiple dynamic supply voltage with adjustable delay buffers restriction[J]. Journal of Signal Processing Systems, 2015, 79(1): 99-104.

[6] Gundala S, Basha M M, Vijayakumar S. Level-up/level-down voltage level shifter for nano-scale applications[J]. Journal of Engineering Science and Technology, 2022, 17(1): 745-759.

[7] Liu Genggeng, Zhu Yuhan, Xu Saijuan, et al. Performance-driven X-architecture routing algorithm for artificial intelligence chip design in smart manufacturing[J]. ACM Transactions on Management Information Systems, 2022, 13(4): 3519422.

[8] 刘耿耿, 黄逸飞, 王鑫, 等. 基于混合离散粒子群优化的Slew约束下X结构Steiner最小树算法[J]. 计算机学报, 2021, 44(12): 2542-2559.
Liu Genggeng, Huang Yifei, Wang Xin, et al. Hybrid discrete particle swarm optimization algorithm for X-architecture Steiner minimal tree construction with Slew constraints[J]. Chinese Journal of Computers, 2021, 44(12): 2542-2559. (in Chinese)

[9] 汤浩, 刘耿耿, 郭文忠, 等. 考虑布线资源松弛的X结构Steiner最小树算法[J]. 模式识别与人工智能, 2020, 33(5): 401-412.
Tang Hao, Liu Genggeng, Guo Wenzhong, et al. X-architecture Steiner minimum tree algorithm considering routing resource relaxation[J]. Pattern Recognition and Artificial Intelligence, 2020, 33(5): 401-412. (in Chinese)

[10] Liu Wenhao, Li Yilang, Chao Kaiyuan. High-quality global routing for multiple dynamic supply voltage designs[C]//2011 IEEE/ACM International Conference on Computer-Aided Design. Piscataway: IEEE, 2011: 263-269.

[11] 饶凌风, 耿娜, 张勇, 等. 不确定环境下无人机任务分配的种群交互式粒子群算法[J]. 电子学报, 2025, 53(8): 2678-2690.
Rao Lingfeng, Geng Na, Zhang Yong, et al. Population interactive particle swarm optimization algorithm for UAV task allocation in uncertain environments[J]. Acta Electronica Sinica, 2025, 53(8): 2678-2690. (in Chinese)

[12] 李照希, 苏震宇, 田宇浩, 等. 基于人工智能算法的单级全差分折叠式共源共栅运算放大器的多目标设计方法[J]. 电子学报, 2025, 53(6): 1784-1791.
Li Zhaoxi, Su Zhenyu, Tian Yuhao, et al. Multi-objective design method for single-stage fully differential folded cascode operational amplifiers based on the artificial intelligence algorithm[J]. Acta Electronica Sinica, 2025, 53(6): 1784-1791. (in Chinese)

[13] Cheng Lianyu, Ling Guang, Liu Feng, et al. Application of uniform experimental design theory to multi-strategy improved sparrow search algorithm for UAV path planning[J]. Expert Systems with Applications, 2024, 255: 124849.

[14] Xue Jiankai, Shen Bo. A novel swarm intelligence optimization approach: Sparrow search algorithm[J]. Systems Science & Control Engineering, 2020, 8(1): 22-34.

[15] Lei Lei, Shao Suola, Liang Lixia. An evolutionary deep learning model based on EWKM, random forest algorithm, SSA and BiLSTM for building energy consumption prediction[J]. Energy, 2024, 288: 129795.

[16] Chang Chun, Pan Yaliang, Wang Shaojin, et al. Fast EIS acquisition method based on SSA-DNN prediction model[J]. Energy, 2024, 288: 129768.

[17] 王毅, 郑宏志, 黄欣, 等. 基于多阶段调度框架的麻雀搜索优化算法[J]. 电子学报, 2024, 52(9): 3086-3096.
Wang Yi, Zheng Hongzhi, Huang Xin, et al. Sparrow search optimization algorithm based on multi-stage scheduling framework[J]. Acta Electronica Sinica, 2024, 52(9): 3086-3096. (in Chinese)

[18] Chu C, Wong Y C. FLUTE: Fast lookup table based rectilinear Steiner minimal tree algorithm for VLSI design[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2008, 27(1): 70-83.

[19] Lin S E D, Kim D H. Construction of all rectilinear Steiner minimum trees on the Hanan grid and its applications to VLSI design[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2020, 39(6): 1165-1176.

[20] Guo Zizheng, Gu Feng, Lin Yibo. GPU-accelerated rectilinear Steiner tree generation[C]//The 41st IEEE/ACM International Conference on Computer-Aided Design. New York: ACM, 2022: 3549434.

[21] Kahng A B, Nerem R R, Wang Yusu, et al. NN-Steiner: A mixed neural-algorithmic approach for the rectilinear Steiner minimum tree problem[J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2024, 38(12): 13022-13030.

[22] Lin Zhenkun, Liu Genggeng, Huang Xing, et al. A unified deep reinforcement learning approach for constructing rectilinear and octilinear Steiner minimum tree[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2025, 44(7): 2711-2724.

[23] Liu Genggeng, Chen Xiaohua, Zhou Ruping, et al. Social learning discrete Particle Swarm Optimization based two-stage X-routing for IC design under Intelligent Edge Computing architecture[J]. Applied Soft Computing, 2021, 104: 107215.

[24] Guo Wenzhong, Huang Xing. PORA: A Physarum-inspired obstacle-avoiding routing algorithm for integrated circuit design[J]. Applied Mathematical Modelling, 2020, 78: 268-286.

[25] Kundu S, Roy S, Mukherjee S. An efficient obstacle-avoiding rectilinear Steiner tree construction method using PB-SAT[J]. IETE Journal of Research, 2023, 69(6): 3346-3356.

[26] Lin Zhenkun, Zhu Yuhan, Huang Xing, et al. Obstacle-avoiding rectilinear Steiner minimal tree algorithm based on deep reinforcement learning[C]//2023 International Conference on Artificial Intelligence of Things and Systems. Piscataway: IEEE, 2023: 149-156.

[27] Zhang Tiancheng, Lyu Zhipeng, Ding Junwen. Guiding solution based local search for obstacle-avoiding rectilinear Steiner minimal tree problem[J]. IEEE Transactions on Emerging Topics in Computational Intelligence, 2024, 8(1): 440-453.

[28] Huang Xing, Liu Genggeng, Guo Wenzhong, et al. Obstacle-avoiding algorithm in X-architecture based on discrete particle swarm optimization for VLSI design[J]. ACM Transactions on Design Automation of Electronic Systems, 2015, 20(2): 2699862.

[29] Held S, Spirkl S T. A fast algorithm for rectilinear Steiner trees with length restrictions on obstacles[C]//2014 International Symposium on Physical Design. New York: ACM, 2014: 37-44.

[30] Qin Yuancheng, Yao Yingbiao, Feng Wei, et al. Dynamic voltage scaling based energy-minimized partial task offloading in fog networks[J]. Wireless Networks, 2022, 28(8): 3337-3347.

[31] Wu T H, Davoodi A, Linderoth J T. Power-driven global routing for multisupply voltage domains[J]. VLSI Design, 2013, 2013(1): 905493.

[32] Khan Q A, Wadhwa S K, Misri K. A single supply level

shifter for multi-voltage systems[C]//The 19th International Conference on VLSI Design Held Jointly with 5th International Conference on Embedded Systems Design. Piscataway: IEEE, 2006: 24.

[33] Zhang Hao, Ye Dongyi, Guo Wenzhong. A heuristic for constructing a rectilinear Steiner tree by reusing routing resources over obstacles[J]. Integration, 2016, 55: 162-175.

[34] Zhu Yuhan, Liu Genggeng, Lu Ren, et al. Timing-driven obstacle-avoiding X-architecture Steiner minimum tree algorithm with slack constraints[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2024, 54(5): 2927-2940.

作者简介

刘耿耿　男,1988年生,福建南安人。博士,现为福州大学计算机与大数据学院教授。中国计算机学会(CCF)杰出会员。主要研究方向为EDA算法。

E-mail: liugenggeng@fzu.edu.cn

吕星灿　男,2002年生,浙江永康人。现为福州大学计算机与大数据学院硕士研究生。主要研究方向为超大规模集成电路布线。

E-mail: 13003842518@163.com

周茹平　女,1998年生,福建宁德人。现为福州大学计算机与大数据学院博士研究生。主要研究方向为集成电路设计自动化。

E-mail: zrp08200331@163.com

郑　瀚　男,2001年生,福建福州人。现为福州大学计算机与大数据学院硕士研究生。主要研究方向为超大规模集成电路布线。

E-mail: hanzheng.cn@outlook.com

傅仰耿　男,1981年生,福建南安人。博士,现为福州大学计算机与大数据学院教授。中国计算机学会(CCF)杰出会员。主要研究方向为数据挖掘与机器学习、智能决策支持系统。

E-mail: fu@fzu.edu.cn