EDA Spring 2025

开题报告

中山大学集成电路学院 微电子科学与工程　苏依然 No.22362067 金凤琳 No.22362034

2025 年 5 月 15 日

一、项目基本信息

（一）项目名称

基于 Kernighan-Lin 算法的电路两路划分实现、分析与优化

（二）项目类别

第二类：数字 EDA 前后端中的算法实现

（三）选题来源

课程项目参考第二类选题 “(3) KL 两区划分算法”

（四）项目周期

1. 核心项目实施阶段：2025 年 5 月 19 日 - 2025 年 6 月 19 日 (约 30 天)
2. 项目优化与拓展阶段：2025 年 6 月 20 日 - 2025 年 6 月 29 日
3. 最终提交截止日期：2025 年 6 月 30 日

（五）项目成员

1. 苏依然
2. 金凤琳

二、项目内容介绍

（一）项目背景与意义

在超大规模集成电路（VLSI）设计自动化（EDA）流程中，电路划分（Circuit Partitioning)是将一个大型电路网络切割成若干个较小子电路模块的关键步骤。良好的电路划分能够有效降低后续布局布线的复杂度，优化关键路径延迟，减少功耗，并对整个芯片的性能、面积和可制造性产生深远影响。

Kernighan-Lin (KL) 算法是一种经典的启发式两路划分算法，它通过迭代地交

换节点对来最小化两个子电路之间的割边数量（Cut Size），同时力求维持分区的平衡。虽然存在许多更新的划分算法，但 KL 算法因其直观易懂、效果尚可，仍然是学习和理解电路划分问题以及启发式优化思想的重要切入点。

本项目旨在通过团队合作实践，深入理解并实现 KL 算法，掌握 EDA 领域中图算法的基本应用，并锻炼算法设计、实现、测试、分析及协作开发的能力。

（二）项目目标

1. 核心算法实现： 完整、正确地实现 Kernighan-Lin 两路划分算法，能够有效地对给定的电路网表（以图的形式表示）进行优化，以减少割边数量。

2. 网表解析与图构建： 实现自动解析自定义格式的电路网表文件，并将其转换为适用于划分算法的图数据结构，具备一定的容错性。

3. 多规模测试与验证： 在课程要求的三种不同规模（小、中、大）的网表上对实现的 KL 算法及对照算法进行全面测试和验证。

4. 性能分析与评估： 针对不同的测试用例，系统地评估 KL 算法相对于基线对照算法的划分质量（割边数、平衡性）、收敛速度和运行时间，并进行必要的比较分析。

5. 进阶指标与优化（锦上添花阶段）：

① 探索并实现至少一种 KL 算法的改进策略：暂定为对 KL 算法初始化分策略的改进以及对 Fiduccia-Mattheyses (FM) 算法核心思想借鉴及实现。

② 实现更广泛的测试结果的可视化展示，增强结果的直观性。

③ 进行更广泛的参数敏感性分析或对特定图结构（链状、星状、网格状）的性能研究。

三、预设方案

（一）核心算法：Kernighan-Lin 算法（在kl_original.py中实现;因有缺陷放弃使用)

本项目将严格遵循 KL 算法的核心思想进行实现，主要步骤包括：

1. 初始划分 (Initial Partitioning)：

① 将电路网络中的所有节点近似平均地分配到两个初始分区 A 和 B 中。② 计算初始划分状态下的割边数量。

2. D 值计算 (D-value Calculation):

① 对于每个节点 v，计算其 D 值：D(v) = E(v) - I(v)。

3. 迭代优化过程 (Iterative Improvement Pass)：

① 在每一轮（Pass）迭代中，所有节点初始为未锁定状态。

② 依次选择一对未锁定的节点 (a ∈ A, b ∈ B) 进行交换（或在 FM 类似算法中，选择单个节点进行移动），使得本次交换（或移动）带来的割边数减少量（即增益 gain）最大。

③ 被选中的节点在本轮迭代中将被锁定。

④ 记录本轮所有尝试的交换（或移动）及其对应的增益序列。

⑤ 找出使累积增益最大的交换（或移动）序列前缀，并实际执行这些交换（或移动）来更新分区。

4. 收敛条件 (Convergence Criteria)：

① 重复执行迭代优化过程，直到某一轮迭代不再产生正的累积增益。

（二）关键技术与实现细节

1. 编程语言与库：

① 主要采用 Python 语言进行开发。

② 核心图数据结构及操作将借助 NetworkX 库。

③ 数据分析与可视化可能使用 Pandas, Matplotlib 等库。

2. 网表解析：

① 定义一种简洁明了的文本文件格式作为电路网表的输入格式。

② 编写 Python 脚本解析该格式的网表文件。

3. 图的表示：

① 使用 NetworkX 的 Graph 对象来表示电路网络。

4. 数据结构：

① 使用 Python 的集合（set）或列表（list）来存储两个分区中的节点。

② 使用字典（dict）或列表存储节点的 D 值及锁定状态。

（三）测试方案

1. 测试数据生成：

① 根据课程要求，自行设计并生成三种规模的网表测试数据：小规模（10 节点，20 边）、中等规模（20 节点，40 边）、大规模（50 节点，100 边）。

②（进阶阶段）可能生成更多具有不同特性（如密度、特定结构）的测试用例。

2. 评价指标：

① 主要指标： 最终割边数、割边数减少率、分区平衡度。

② 辅助指标： 算法收敛所需的迭代轮数、算法运行时间、结果稳定性（多次运行的统计数据）。

3. 对照实验（核心基础指标）：

① 实现并测试一个简单的基线划分算法（例如，纯随机划分策略或一个简单的贪心策略）。

② 将 KL 算法的划分结果与该基线算法产生的割边数、平衡度等指标进行对比，以量化 KL 算法的优化效果。此项工作为核心功能实现阶段的必备内容。

（四）项目亮点与拓展（主要在锦上添花阶段完成）

1. KL 算法改进探索：

① 多次运行与统计分析：对每个测试用例进行多次具有不同随机初始划分的运行，统计报告最佳、最差、平均割边数及标准差。

② 改进初始划分策略：尝试几种不同的、简单的启发式初始划分方法（除了随机划分），例如基于广度优先搜索（BFS）或深度优先搜索（DFS）的某种策略，然后比较它们对 KL 最终结果的影响。

②（选）FM 算法思想借鉴： 研究 Fiduccia-Mattheyses (FM) 算法，尝试借鉴其单元移动和高效增益更新机制的思想，对基础 KL 进行改进或实现一个简化版 FM。

2. 结果可视化： 利用 NetworkX 和 Matplotlib 对划分前后的图（突出显示割边）进行可视化。

3. 参数敏感性分析： 探究初始划分策略、平衡度约束等对算法性能的影响。

四、人员分工

（一）[队员 A]

1. 主要职责：

① 负责 KL 算法核心逻辑的实现与调试（包括 D 值计算、迭代优化过程、收敛判断）。

② 参与网表解析模块和图构建模块的实现。

③ 负责基础测试用例的生成与执行（针对 KL 算法），收集核心性能指标数据。

④ 参与项目报告中算法实现、实验结果部分的撰写。

⑤（锦上添花阶段）侧重于一种 KL 算法改进策略的实现与测试。

（二）[队员 B]

1. 主要职责：

① 负责 KL 算法相关文献的深入调研，理论理解与方案设计。

② 主导网表解析模块和图构建模块的设计与实现。

③ 负责测试方案的整体设计，主导对照基线算法的实现与测试，辅助 KL 算法测试用例的生成与执行，侧重于数据分析与统计（包括 KL 与基线的对比）。

④ 负责项目报告的整体框架搭建、背景意义、预设方案、结论以及文字润色。

⑤（锦上添花阶段）侧重于结果可视化模块的实现和参数敏感性分析。

（三）共同职责

1. 共同参与项目整体方案的讨论与制定。
2. 共同负责实现和测试对照的基线划分算法（例如随机划分）。
3. 共同进行代码审查、集成测试和 Bug 修复。
4. 共同完成最终项目报告的整合与完善。
5. 共同准备项目说明文档和最终提交材料。

五、项目执行时间进度表

按顺序推进，尽量在设定截止时间前完成各阶段任务

阶段一：核心功能实现（2025 年 5 月 19 日 - 2025 年 6 月 19 日）

（一）第一周（2025 年 5 月 19 日 - 2025 年 5 月 25 日）：需求分析、方案细化与基础构建

1. 团队深入学习和讨论 KL 算法原理、实现细节及潜在挑战。
2. 确定详细的网表文件格式和图表示方案。
3. [队员 B 主导，队员 A 协助] 完成网表解析模块的详细设计与初步实现。
4. [队员 A 主导，队员 B 协助] 搭建项目框架，熟悉 NetworkX 库，完成图构建模块的初步实现。

（二）第二周 (2025 年 5 月 26 日 - 2025 年 6 月 1 日)：KL 算法核心模块实现 (1)

1. [队员 A 主导] 实现 KL 算法的初始划分模块和 D 值计算模块。

2. [队员 B 协助] 进行单元测试，并提供理论支持。

3. 共同讨论并确定迭代优化过程（Pass）的具体实现逻辑。

（三）第三周 (2025 年 6 月 2 日 - 2025 年 6 月 8 日)：KL 算法核心模块实现 (2) 与集成

1.[队员 A 主导] 完成 KL 算法的核心迭代优化过程（单轮 Pass 逻辑，包括节点选择、增益计算、节点锁定、D 值更新）。

2. [队员 A 主导] 实现算法的整体框架和收敛判断逻辑。

3. [队员 B 协助] 进行模块集成与初步的整体功能测试。

（四）第四周 (2025 年 6 月 9 日 - 2025 年 6 月 15 日)：核心功能测试、基线算法与初步数据分析

1. [队员 B 主导，队员 A 协助] 生成课程要求的小、中、大规模测试网表。

2.[共同] 实现并测试对照的基线划分算法（例如随机划分）并收集其性能数据。

3. [共同] 在所有基础测试用例上运行 KL 算法，收集核心实验数据（割边数、平衡度、轮数）。

4. [共同] 调试代码，修复 bug，确保核心功能稳定可靠。

5. [队员 B 主导] 对 KL 算法及基线算法的初步实验数据进行整理和对比分析。

（五）缓冲与核心报告撰写 (2025 年 6 月 16 日 - 2025 年 6 月 19 日)：

1. [共同] 解决遗留问题，确保核心算法及基线算法按预期工作。

2.[共同] 开始撰写项目报告的核心部分（算法原理、实现、基础实验结果与对比分析）。

3.[共同] 整理并详细注释基础部分源代码，编写清晰的项目说明文档（README）。

4. 目标：于 6 月 19 日前完成 KL 算法核心功能的稳定实现、基础测试、对照基线算法的实现与测试，并完成初步对比分析。

阶段二：锦上添花与项目完善 (2025 年 6 月 20 日 - 2025 年 6 月 29 日)

（六）第五周 (2025 年 6 月 20 日 - 2025 年 6 月 26 日)：进阶功能开发与实现

1. [队员 A] 根据分工，实现选定的 KL 算法改进策略（多次运行取优或初始划分策略改进或 FM 思想借鉴）。

2.［队员 B］根据分工，实现结果可视化模块，并开始进行参数敏感性分析或更多样化测试用例的探索。

3.［共同］定期同步进展，进行交叉测试和代码集成。

（七）第六周（2025 年 6 月 27 日 - 2025 年 6 月 29 日)：最终测试、报告完善与文档准备

1.［共同］完成所有进阶功能的测试与数据分析。

2.［队员 B 主导，队员 A 协助］整合所有实验结果，完成项目报告的最终撰写、修改与润色。

3.［共同］整理并详细注释进阶部分源代码，完善项目说明文档（README)。

4. 准备最终提交的所有材料。

阶段三：最终提交（2025 年 6 月 30 日）

（八）项目提交

于 6 月 30 日前，提交完整的项目报告、源代码及相关说明文档。