Parallelization of Floorplanning in VLSI

Po-Shen Chen
Institute of Computer Science and Engineering
NYCU, Hsinchu, Taiwan
boson13579@gmail.com

Yu-Hsuan Tsao
Institute of Computer Science and Engineering
NYCU, Hsinchu, Taiwan
selina920120@gmail.com

Chia-Yen Hsu
Institute of Computer Science and Engineering
NYCU, Hsinchu, Taiwan
yen0224@proton.me

Abstract

With the rapid development of semiconductor processes, the design complexity of Very Large Scale Integration (VLSI) is increasing. Among them, analog circuits are highly sensitive to physical effects such as symmetry, noise, and thermal gradients. The floorplanning stage has a decisive impact on the final performance, power, and area (PPA) of the chip. A good analog circuit layout not only needs to minimize the chip area, but also needs to consider signal integrity, component matching, and overall performance, such as the **Integral Nonlinearity (INL)** index of a digital-to-analog converter (DAC).

Although commercial EDA tools in the industry have developed more complex layout technologies (such as machine learning assistance and constraint-driven optimization), the B*-Tree combined with the simulated annealing algorithm is still one of the standard methods for studying layout problems in academia. This program aims to solve the layout problem of rectangular modules. The goal is to minimize the chip area and optimize the INL value under the condition of no overlap.

However, the simulated annealing algorithm is essentially an iterative random search process, and its execution time is often very long for large-scale problems. In order to speed up the solution process and find better solutions, the **parallelization** of this floorplanner has become a research direction of great value. This project aims to explore and implement an efficient parallel floorplanner to meet the growing design challenges.

1 Introduction and Motivation

As semiconductor manufacturing processes advance, the design complexity of Very Large-Scale Integration (VLSI) circuits has escalated. Analog circuits, in particular, are highly sensitive to physical effects such as symmetry, noise, and thermal gradients. Consequently, the floorplanning stage critically impacts the final Performance, Power, and Area (PPA) of the chip. An effective analog circuit layout must not only minimize the chip's footprint but also ensure signal integrity, device matching, and optimal performance, exemplified by the **Integral Nonlinearity (INL)** metric in Digital-to-Analog Converters (DACs).

While commercial EDA tools have incorporated sophisticated layout techniques like machine learning assistance and constraint-driven optimization, the combination of B*-Tree representation and Simulated Annealing (SA) remains a foundational academic method for tackling the floorplanning problem. This project aims to address the placement of rectangular modules, with the objective of minimizing chip area and optimizing INL while adhering to non-overlapping constraints.

However, SA is an iterative and stochastic search algorithm, which can be computationally prohibitive for large-scale designs. To accelerate the search for optimal solutions, the **parallelization** of the floorplanner is a promising research avenue. This project focuses on the design and implementation of an efficient parallel floorplanner to address these mounting design challenges.

2 Problem Statement

2.1 Problem Description

This project targets the **Analog Circuit Floorplanning** problem for parallelization, specifically focusing on improving a sequential floorplanner that uses a Simulated Annealing (SA) algorithm for optimization.

The core task is defined as: given a set of 'n' rectangular modules, find a legal placement that satisfies the following conditions:

(1) **Hard Constraints**:
- All modules must be placed within the chip boundary.
- No two modules can overlap.
- Pre-defined symmetry constraints must be met.

(2) **Optimization Objectives**:
- **Area Minimization**: Minimize the area of the bounding box enclosing all modules.
- **INL Optimization**: For DAC circuits, minimize the Integral Nonlinearity to ensure conversion accuracy.
- **Cost Function**: Cost = $\alpha \times$ Area + $\beta \times$ INL, where α and β are weighting factors.

2.2 Rationale for Choosing This Problem

2.2.1 Significant Computational Bottleneck. Sequential SA faces severe performance limitations when handling practical analog circuit layouts. Parallelization is essential to overcome these bottlenecks and achieve better results in a shorter time.

2.2.2 High Parallelizability. The combination of SA and B*-Tree offers multiple opportunities for parallelization, including parallel evaluation of states, multi-start parallel search (Replica Exchange), parallel tempering, and parallel exploration of neighborhood moves.

2.2.3 Quantifiable Performance Metrics. The problem has clear metrics for evaluating the effectiveness of parallelization, such as speedup, efficiency, scalability, and improvements in solution quality (area and INL).

2.2.4 Practical and Industrial Relevance. Automated analog layout is crucial in the semiconductor industry to reduce design cycles, handle increasing complexity, and improve chip yield and performance.

3 Proposed Approach

We plan to adopt a **Multi-Start Simulated Annealing** parallelization strategy. The core idea is to run multiple SA instances independently and concurrently, each starting from a different initial state (a different randomly generated B*-Tree). The global best solution is then selected from the best solutions found by all instances.

3.1 System Architecture

The system architecture is depicted in Figure 1.

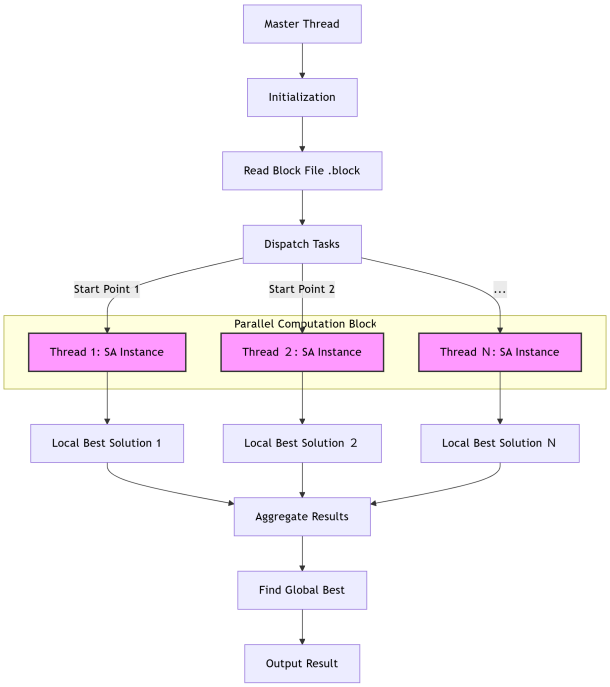

Figure 1: System Architecture Diagram.

3.1.1 Master Thread. Responsible for initialization, reading the input block file, dispatching tasks to worker threads, and aggregating results to find the global best solution.

3.1.2 Worker Threads (SA Instances). Each worker thread independently executes the complete SA algorithm, starting from a unique random B*-Tree. It performs iterative optimization and reports its best-found solution back to the master thread.

4 Language and Tool Selection

We will use **C++ with OpenMP**.

- **Performance**: C++ offers high performance and fine-grained control over hardware resources, which is critical for computationally intensive CAD tools.
- **Ease of Use**: OpenMP's pragma-based model simplifies parallel programming compared to manual thread management with Pthreads, reducing development complexity.
- **Incremental Parallelization**: OpenMP allows for targeted parallelization of the most time-consuming parts of the code without major architectural changes.
- **Portability**: OpenMP is a widely supported industry standard, ensuring code portability across various compilers and platforms.

5 Related Work

The automation of analog circuit layout has been a prominent research area in VLSI CAD. The **B*-Tree**, proposed by Murata et al., is an efficient representation for non-slicing floorplans [1]. Simulated Annealing is widely used for such NP-hard optimization problems due to its ability to escape local optima.

Several parallel SA strategies have been proposed, including Move Decomposition and Parallel Markov Chains. However, the multi-start approach adopted in this project is conceptually simple, easy to implement, and has been shown to achieve good speedup on multi-core architectures, making it suitable for our objectives.

6 Expected Results

We anticipate achieving the following outcomes:

(1) **Significant Speedup**: Achieve a speedup of over 5x on a multi-core system.
(2) **Improved Solution Quality**: Find layouts with smaller area and lower INL within the same execution time.
(3) **Good Scalability**: Demonstrate near-linear performance improvement as the number of processor cores increases.
(4) **Generalizable Framework**: Develop a parallelization framework applicable to other similar combinatorial optimization problems.

7 Project Timetable

Table 1: Project Development Timetable

Time	Work Description
Weeks 1-2	Study floorplanning and implement sequential version
Weeks 3-4	Study OpenMP and implement parallel version
Weeks 5-6	Continue development of the parallel version
Weeks 7-8	Refine granularity and optimize performance
Weeks 9-10	Write final report and documentation

References

[1] Hiroshi Murata, Kunihiro Fujiyoshi, Satoshi Nakatake, and Yoji Kajitani. 2003. Rectangle-Packing-Based Module Placement. In *The Best of ICCAD*, A. Kuehlmann (Ed.). Springer, Boston, MA. https://doi.org/10.1007/978-1-4615-0292-0_42

[2] Xiaoping Tang, Ruiqi Tian, and D. F. Wong. 2000. Fast evaluation of sequence pair in block placement by longest common subsequence computation. In *Proceedings of the conference on Design, automation and test in Europe (DATE '00)*. https://doi.org/10.1145/343647.343713

[3] Florin Balasa. 2000. Modeling non-slicing floorplans with binary trees. In *IEEE/ACM International Conference on Computer Aided Design (ICCAD-2000)*. https://doi.org/10.1109/ICCAD.2000.896443