Report for Programming Assignment #3 Analytical Global Placement with Conjugate Gradient Optimization

Student Name: Yi-En Wu
Student ID: B11901188
Course: Physical Design for Nanometer ICs
Instructor: Prof. Yao-Wen Chang

May 18th, 2025

---

1 Introduction

This programming assignment focuses on implementing a global placement algorithm for large-scale circuit designs, and I choose to apply analytical approach. The objective is to minimize the total wirelength while evenly distributing cells across the chip. This is achieved through a density-based placement formulation optimized using the Conjugate Gradient method.

2 Data Structures Used

Since I chose to apply analytical approach for this assignment, the main effort is constructing the wirelength and density function. There are not many different data structures used, so I will mainly discuss what I've done for the *wirelength*, *density*, and $objective_function$ class.

Below is the objective function that I used(NTUplace3) for the analytical placement.

$$\min\ W(\mathbf{x},\mathbf{y}) + \lambda \sum_{b} \left(\hat{D}_b(\mathbf{x},\mathbf{y}) - M_b\right)^2 \tag{1}$$

2.1 Wirelength Class

For the wirelength model, I use the WA(weighted-average) wirelength model, where the wirelength is calculated as

$$W_{WA}(\mathbf{x},\mathbf{y}) = \sum_{e\in E}\left(\frac{\sum_{v_i\in e} x_i \exp(x_i/\gamma)}{\sum_{v_i\in e}\exp(x_i/\gamma)} - \frac{\sum_{v_i\in e} x_i \exp(-x_i/\gamma)}{\sum_{v_i\in e}\exp(-x_i/\gamma)} + \frac{\sum_{v_i\in e} y_i \exp(y_i/\gamma)}{\sum_{v_i\in e}\exp(y_i/\gamma)} - \frac{\sum_{v_i\in e} y_i \exp(-y_i/\gamma)}{\sum_{v_i\in e}\exp(-y_i/\gamma)}\right) \tag{2}$$

In my implementation, I use $\gamma = 500$ to approximate the wirelength. To store each net's wirelength, I used vector array to store all the wirelenth, which is efficient when accessing the data.
To calculate the gradient of the wirelength, I simply take derivatives of the WA model.

2.2 Density Class

This class models the density penalty and its gradient. To approximate the density with a differentiable function, I chose the bell-shaped function and divided the whole cell into multiple grids. After calculating each cell's density, I use a 2D Guassian filter to smooth the density and then apply level smoothing. Which is the method used in the paper provided(NTUplace3).

$$D_b(\mathbf{x}, \mathbf{y}) = \sum_{v \in V} P_x(b, v) P_y(b, v) \tag{3}$$

where P_x and P_y are the overlap functions of bin b and block v along the x- and y-directions. We adopt the bell-shaped potential function p_x to smooth P_x. p_x is defined by

$$p_x(b, v) = \begin{cases} 1 - ad_x^2, & 0 \le d_x \le \frac{w_v}{2} + w_b \\ b\left(d_x - \frac{w_v}{2} - 2w_b\right)^2, & \frac{w_v}{2} + w_b < d_x \le \frac{w_v}{2} + 2w_b \\ 0, & \frac{w_v}{2} + 2w_b < d_x \end{cases} \tag{4}$$

where

$$a = \frac{4}{(w_v + 2w_b)(w_v + 4w_b)}, \quad b = \frac{2}{w_b(w_v + 4w_b)} \tag{5}$$

The above equations show how the density is approximated by using the bell-shaped model.
The data structure I used is also a 2D vector array to store each bin's density, and to calculate the gradient, I take the derivative of the equations above.

2.3 Objective Function Class

Combines both wirelength and density objectives.

- Provides the cost function used in optimization.
- Calls forward (operator()) and backward (gradient) methods of both sub-objectives.

3 Findings

3.1 Bell-Shaped Potential Helps with Local Influence

Instead of a sharp cutoff, bell-shaped functions give smooth partial contributions to nearby bins. This prevents oscillations in gradient and improves convergence. At first, I have tried directly applying Guassian smoothing to the real density, but two problems occurred. One is that this approximation isn't differentiable; second, some part of the density is too flat which made the cells unable to spread out.

3.2 Normalization in Density

At the beginning, I've tried numerous method for the density model; however, none of them can effectively spread cells out(for I initialize all the cells in the center of the chip). Therefore, I found out in the paper, NTUplace3, it implemented normalization for the density, after I adopt this method, the density gradient became way better, be cause of this smoothing, cells are more probable to spread across.

3.3 Cells Spreading out of the Chip Boundary

This is the main problem that I still face and haven't resolved. (Thanks to the strong and powerful legalization tools provided by the TAs) Since I used analytical method, when cells experience density's gradient force, it is likely that cells will be spread out of the chip boundary. I've tried to create a virtual potential barrier along the outline, but it seems that cells can still overcame the barrier if the gradient is strong enough. Also, if a cell accidentally crossed the barrier and went out the chip outline, it may become more difficult for the cell to be pulled back into the chip for there is a barrier. Therefore, I've tried to clamp the cell's position to the boundary if it went over the chip outline, but the result became weird. Eventually, I did nothing to the cells that go out of the boundary. Instead, I controlled the spreading force to make the number of out-flowing cells as small as possible.

4 Global Placement Result

There are two test cases given, below are the global placement results, with the cells distribution and the density map shown.

4.1 Test Case 1: ibm01-cu85

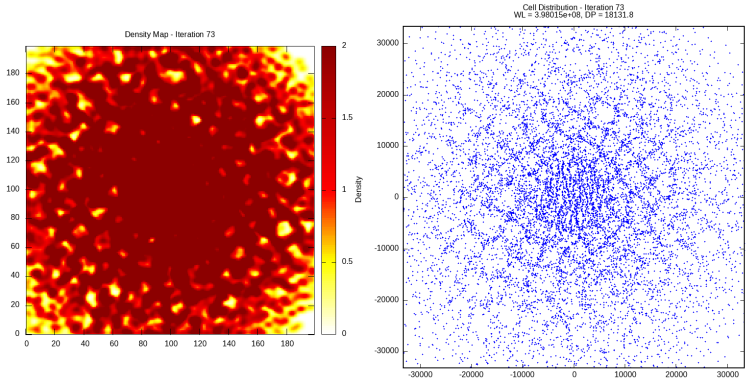

Figure 1: ibm01-cu85

Final HPWL: 185407820

4.2 Test Case 2: ibm05

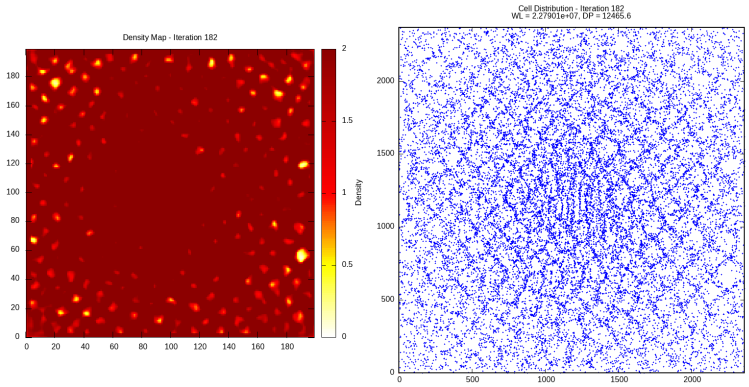

Figure 2: ibm05

Final HPWL: 17529576

5 Conclusion

Some of my classmates mentioned that the second programming assignment—B*-tree floorplanning—was the most challenging. However, for me, this assignment has been the most difficult and the one I'm least satisfied with.

When examining the density and cell distribution plots, it's clear that the cell spreading is uneven. There appear to be noticeable patterns in the distribution, but I haven't yet identified the underlying cause. I suspect the issue stems from an error in my gradient calculation for the density function, which may have led to the suboptimal performance.

This assignment has made me deeply aware of the importance of a well-designed approximation model. A good approximation can yield results that closely mirror the real-world counterpart, whereas a poor model or incorrect mathematical derivation—possibly the reason for my current outcome—can cause the system to behave unexpectedly.

I hope to eventually pinpoint the root cause of my error and improve upon it in future work.