Physical Design for Nanometer ICs Programming Assignment 2

B07901087 電機三 蔡沛洹

1. File explanation:

This program is written in C++, and there are several cpp and header file for my floorplanner.

module.h: it contains Net, Block, Node structures

module.cpp: it contains node operation(remove, insert) and Net operation(update terminal and calculate HPWL)

floorplanner.h: some parameters for SA and tree initialization are listed above, and it contains the main structure for the floorplanning algorithm, such as B* tree, and SA, it also contain some report functions

floorplanner.cpp: some important function such as FastSA and tree packing, tree initialization are in this file.

main.c: for execution.

To compile the binary, please see readme.txt

Plot:

Also see readme.txt

2. data structure:

Contour:

I implemented contour using std::map in C++, and the structure is maintain by only maintain the last point of the some contour: (since it's hard to explain it directly, I give the following example)

Assume the map contains

(0,0) – (2,1) – (4,0) – (7,2) – (8,3)

It means that the contour is:

(0,0) – (1,1) -- (2,1) – (3,0) – (4,0) – (5,2) – (6,2) – (7,2) – (8,3)

B* tree:

It is the binary tree, and I followed its original definition to implement it. However, I add a link pointing from child to parent since it will be used when delete nodes and insert nodes.

3.Algorithm:

Node

There are two main operations for Node class, which is used for op2 for B* tree.

Remove:

The most important part of remove is how to deal with removing a node which has two children. In most case, we can just pick a rightmost child to replace it and delete the node, however, I pick a leaf to replace it for robustness.

Insert:

In my insert operation, when the node being inserted has child, I randomly make the origin child to be left or right child of the inserted node. In my observation, the B* tree have better packing result(area) when the tree is skewed. By implementing insert operation in this way, I can may the tree to be more skewed.

Tree initialization:

Since in large case it may take large effort for the floorplanner to find a feasible solution if we only use complete binary tree, So I implemented a greedy tree initialization, which can make the initial tree fit into the outline as much as possible. The method I took is similar to the first fit decreasing algorithm in bin packing: 1.first rotation all block to fit H/W ratio of the outline, and sort them by area. Then we just put it into the outline from left to right, and make a constrain so that the block won't exceed the Width of the outline

FastSA:

I use modified the original SA by letting the probability to op2 and op3 larger since the op2 can make the tree to be skewed, and op3 will change the floorplanning structure, so I think this may improve the performance of my result.

4.Discussion:

There are three main interesting things I found: 1. Cost function 2. Initial solution 3. operations

1.Cost function:

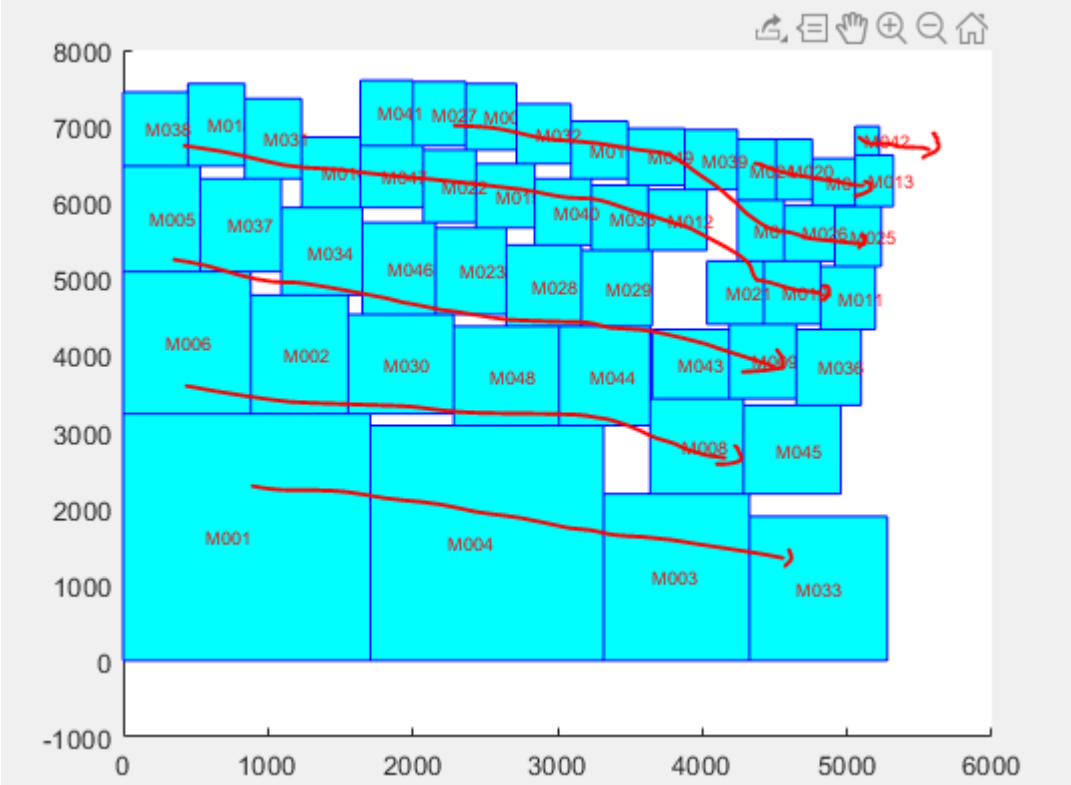
I only use area and wirelength in my cost function at the beginning, however, I found that I have a problem when dealing with larger case: my SA will run too long to fit into the outline. As a result, I add Wpenalty and Hpenalty when the outline was exceeded. (the penalties also need normalization or they will be too dominant).

Finally, I add the ratio penalty term to make the planning outline be more similar to the restricted area, which may help SA to fit into the restricted outline.

2.Initialization:

I use the first fit decreasing like algorithm to implement tree initialization.

First, I sort blocks by their area. Second, I place blocks from left to right. If the right boundary is reached, I place the next block right above the left most block, and keep growing right... Finally, When the upper bound is reached, I pick some median blocks called “submother block”, and place remaining blocks on them. However, I didn’t consider wirelength, so the quality of SA is important or I will get a bad wirelength solution.



3. operations

Finally, I had found an interesting data structure for swapping operation. In the beginning, I though that I need to manipulate pointer for swapping operation, however, I can maintain a linking between node and block, and just swap the corresponding block of two nodes!

4.precision issue

I found that there are some difference between mine answer and checker, in most

case, this can be solved by setprecision function. However there are still some cases that make difference between the checker's and mine answer…(原因應該是四捨五入的不同?我有附上計算結果)

```
alpha*bestarea: 1.7705e+07
cost? --->17836474.875
area: 23606730
wirelength: 525709.5
[pd2132@edaU6 ~/prog2/PD_PA2]$ bin/checker/checker "bin/xerox.block
ox.nets" bin/result.txt 0.75

Checking Report...
cost:       actual  17836476.000000/reported 17836474.000000
wirelength: actual  525709.500000/reported 525709.500000
area:       actual  23606730.000000/reported 23606730.000000
width:      outline 6937/actual 5565/reported 5565.000000
height:     outline 5379/actual 4242/reported 4242.000000
runtime: 0.199151
```