HW3 Fixed-outline Floorplanning

資工19 104062302 洪旻夆

Compile & Execute

Compile:

```
$ make
```

Remove executable file:

```
$ make clean
```

Execute:

```
$ ./hw3 <.hardblocks> <.nets> <.pl> <.floorplan> white_space_ratio
```

e.g.

```
$ ./hw3 ../testcase/n100.hardblocks ../testcase/n100.nets ../testcase/n100.pl ../output/n100.floorplan 0.1
```

Experiments

Results of three testcases with 0.1 and 0.15 white space ratio

	n100 0.1	**n100 0.15**	**n200 0.1**	**n200 0.15**	**n300 0.1**	**n300 0.15**
Wirelength	223071	218532	407057	402445	567891	549026
Runtime	26s	26s	101s	101s	226s	226s

Minimum white space ratio able to produce feasible solution

	n100	**n200**	**n300**
white space ratio	0.65 (width=437)	0.07 (width=433)	0.075 (width=543)

According to the table, the minimum white space ratio that my program is able to produce a feasible solution in 20 minutes increases as the # hardblocks. But I believe that if there is more time for n200 and n300, my program will be able to produce feasible solutions.

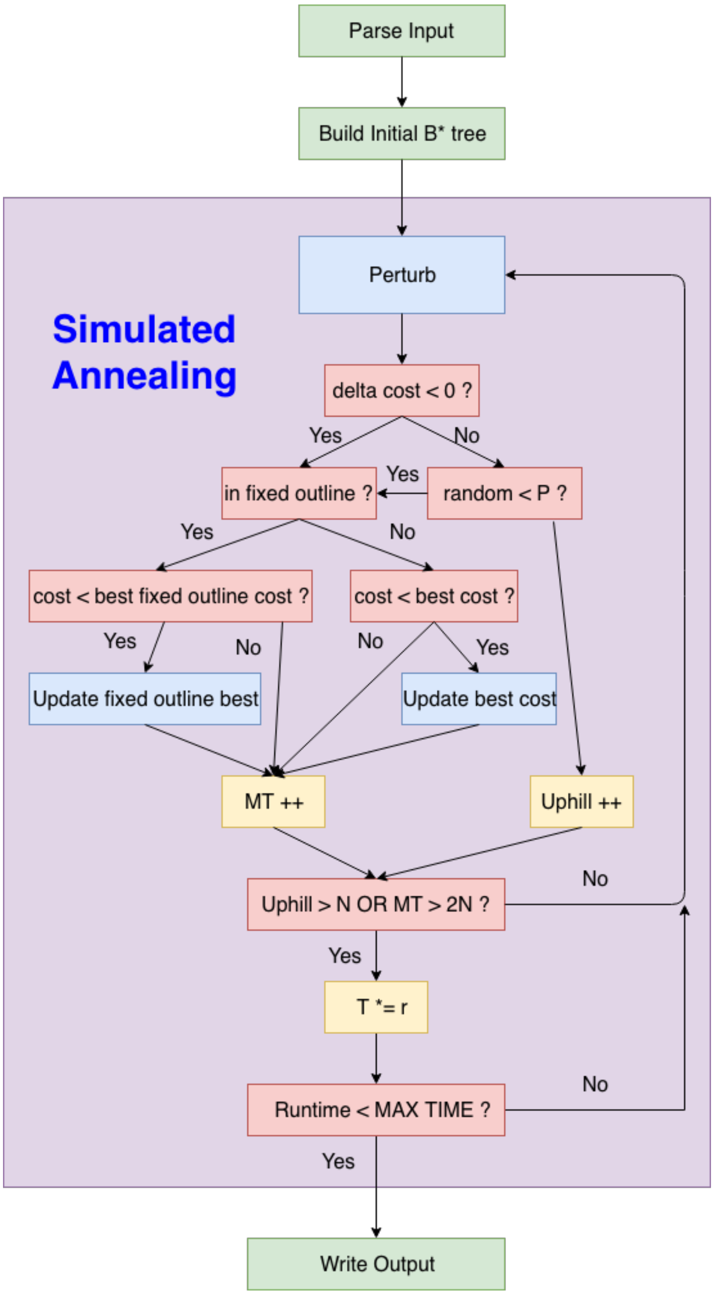
Implementations

I've implemented in **simulated annealing** algorithm using **B*-tree** data structure. The B*-tree part is relatively simple only considering the x-coordinate of left child is equal to its parent's x-coordinate plus its parent's width while the x-coordinate of right child is equal to its parent's x-coordinate and the y-coordinate of both left and right children should consider the horizontal contour.

The initial B*-tree is built randomly to construct a full binary tree. Although I've tried some other methods to initialize the B*-tree, it seems that the initialization doesn't influence the SA result. Different from original paper, I use an array to store the highest y-coordinate of each x-coordinate instead of maintaining a doubly-linked list. After some observations on the maximum width on x-coordinate, I set the maximum width of the contour array to 5 times of the target fixed-outline.

An important part of SA is the cost function. At first, I use area, wirelength and square of floorplan ratio to form the cost function. But there is a problem that the value of wirelength is usually much greater than of area. After reading some reference, I change the cost function by normalizing each term with the initial area and wirelength while also adding penalties to width and height if they aren't in the fixed-outline.

Normally, SA has two parts, the cooling of temperature and the perturbing between the cooling. At first, I've followed the normal steps by stopping SA while temperature is lower than a certain value. But in that way, my program cannot guarantee finding a feasible solution with its floorplan inside the fixed-outline. Currently, I set the stopping criteria to let the program run a specific time corresponding to the # hardblocks. This method guarantees finding a feasible solution with white space ratio of 0.15 but not 0.1 and I'm not able to have better scheduling of SA. The image on the right side shows the flow chart of my algorithm.



Results Comparison

Comparing to the top 5 results of last year, my results are in between the first place and second to forth place. It's obvious that there is a trade of in wirelength and runtime. The result of Top 1 is the fastest but has a larger wirelength than the others while in the opposite, the results of top 2 and top 3 have shorter wirelength but longer runtime. My result has a longer runtime than top 1 but a smaller wirelength and shorter runtime than top 2 and top 3 but larger wirelength. If I can further improve my SA scheduling, I believe I could have a better runtime and wirelength in a fixed amount of time.

What I've learned

In class, the professor has continually said that he isn't in favor of simulated annealing although it's truly powerful because there are lots of hyper-parameter to be tuned. After the HW3, I finally realize the reason why the professor doesn't like SA. Even though there is a rough scheduling of SA, there are still lots of parameters may be different in case to case, so it's always necessary to figure out a set of the best parameters for each problem.