VLSI Physical Design Automation
Final Project

Team Member's SID/Name:
109062568 黃鵬泰
110062526 小山翼

Part Ⅰ： How to compile and execute the program.

- To compile the program , go to directory **"src/ "**and execute the following command , then the executable file named **"cell_move_router"**will be generated in **"bin/ "** .

Command: $ make

- It can be executed with command 1 or 2 in directory **"src/ "** or **" bin/ "** respectively .

Command 1: $ ../bin/cell_move_router <input file> <output file>
Command 2: $ ./cell_move_router <input file> <output file>
e.g.: $ ../bin/cell_move_router ../testcases/case1.txt ../output/case1.txt

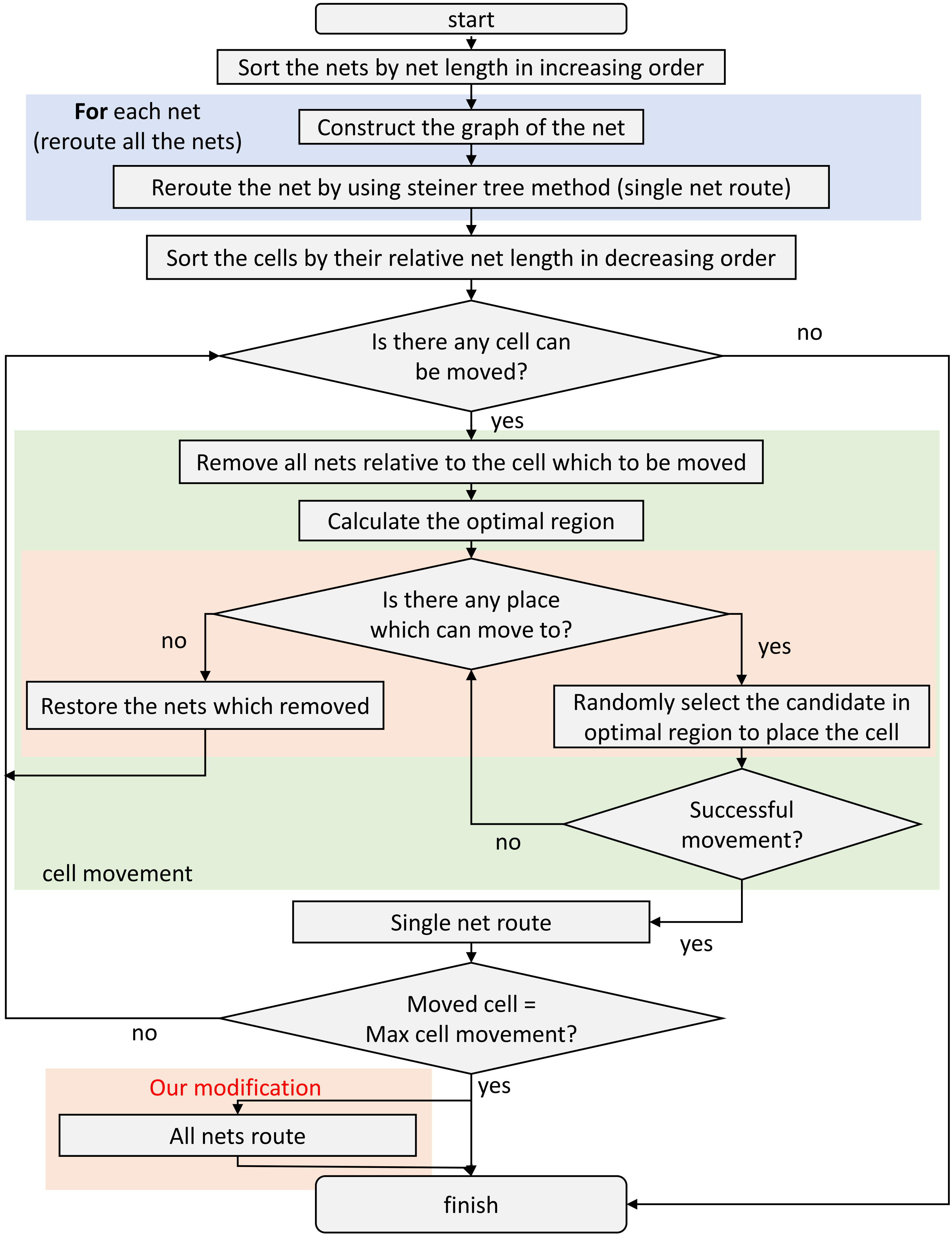
Part Ⅱ： Final score and the runtime of each testcase.

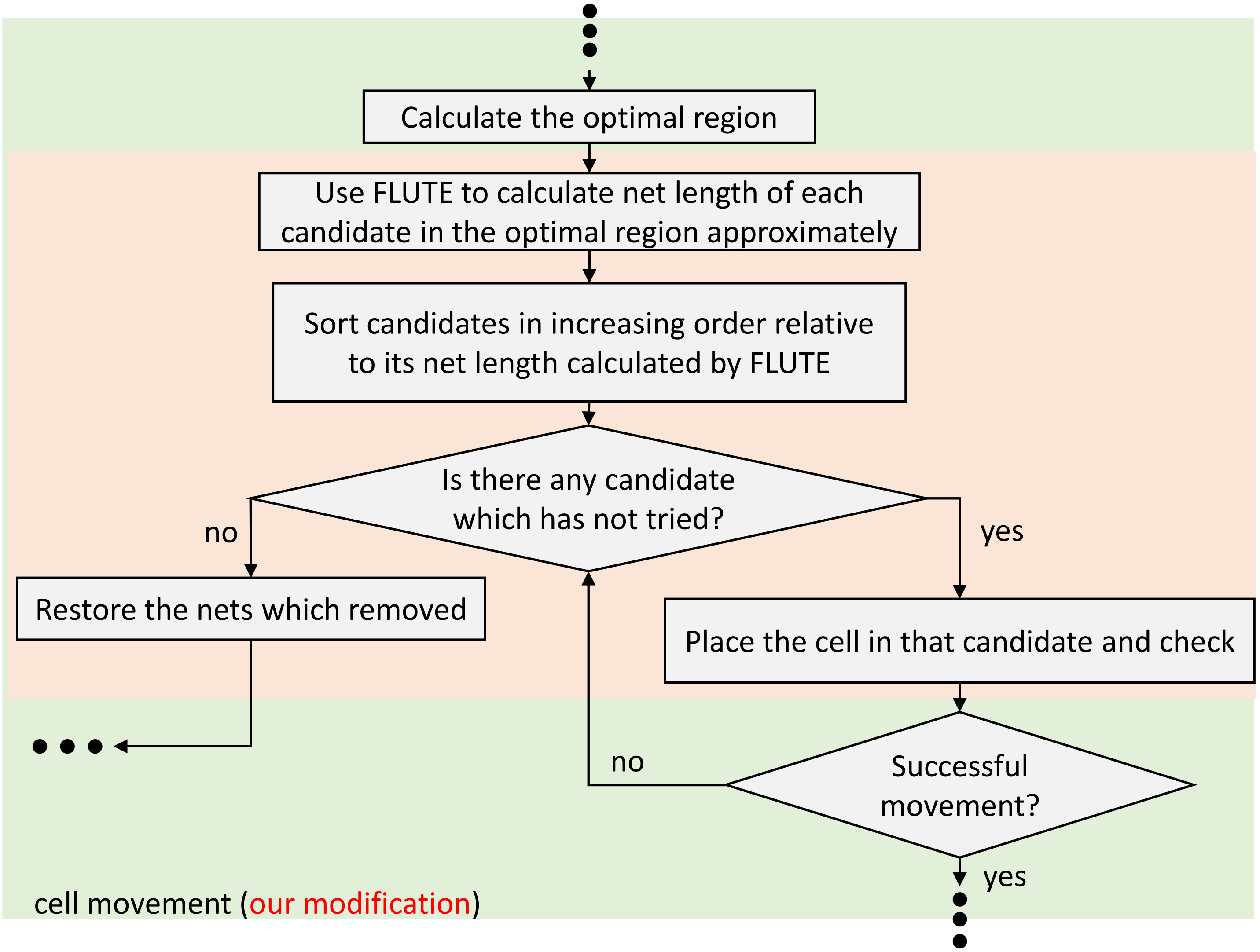
	The original code		After our modification		
testcases	score	runtime (s)	score	runtime (s)	R_{diff}
case1	39.7	0.00	43.3	1.39	0.429
case2	6.1	0.00	6.1	1.37	0.158
case3	5416.2	1.22	5981	8.38	0.201
case4	336944	156.45	356669	172.81	0.233
case5	749735	594.49	808803	810.22	0.220
case6	375407	1157.06	461415	1500.46	0.082
case3B	5855	1.24	6086.2	8.54	0.202
case4B	344578	141.45	367012	175.05	0.231
case5B	745943	621.63	809446	853.01	0.221
case6B	396825	1158.61	478361	1617.08	0.086
Avg. ratio	1	1	1.131	2.69	0.206

* The definition of R_{diff} is same as [1].

Part Ⅲ: Algorithm

- In this part, I will show the original flow chart (this page) and the red part is where we do our modification (next page). We modified the code which TA provided to us [2].





- In the beginning, the nets will be all rerouted since the initial routing may be not the optimal solution. In this step, the nets will be sorted by their net length by increasing order and reroute using steiner tree method one by one. We will reroute all nets after moving the cells, too.

- After rerouting, the next step is cell movement. First, the cells will be sorted by their relative net length in decreasing order. The reason is because the longer its relative net length is, its potential of wire reduction may be higher. Then the cells will be chosen to move in cell movement part.

- In cell movement part, the optimal region will be calculated, which is the region that the cell can be placed. After calculating the optimal region, the candidate which to place the cell has to be determined. It determine the candidate randomly in the original code, but their may be the problem that it can not ensure that it is the optimal solution, so we modify its method.

- In our method, we will first calculate the net length between other connected pins and each candidate by using FLUTE roughly. Then sort the candidates by increasing order relative to their net length which has been calculated. Then try to place the cell to each candidate until the cell movement success. By this method, we think that it can find the optimal solution of each cell movement. The cell movement will continue until reach the number of maximum cell movement.

Part Ⅳ: The tricks we do to enhance our solution quality.

We modify the method of choosing the cell to be moved in cell movement part to enhance our solution quality. The reason is because in the original implementation, it can not ensure that it is the optimal solution. We can conclude that the result after adding our idea is successful from the table in part Ⅱ. Our code spends more time since we have to calculate all net length between other connected pins and each candidate every time after choosing the cell to move. But I think it is not a big problem because we can further improve it by using multithread method. From the table in part Ⅱ, we can see that our implementation achieves 13.1% better scores than the original one. And R_{diff} shows that our implementation can achieve 20.6% total wire reduction on average.

Part Ⅴ: What have you learned from this project? What problem(s) have you encountered in this project?

In the beginning, we have to spend a lot of time understanding the code which TA provided. After that, we first read the paper[1] and discuss where can be modified. During the modification, we often encountered the segmentation fault but have no idea where the problems are. We have learned a lot about placement and routing during this project while surveying papers and modifying the code.

Part Ⅵ: References

[1]: Fangzhou Wang, Lixin Liu, Jingsong Chen, Jinwei Liu, Xinshi Zang and Martin D.F. Wong, Department of Computer Science and Engineering, The Chinese University of Hong Kong, “Starfish: An efficient P&R Co-Optimization Engine with A*-based Partial Rerouting”, 2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)
[2]: jacky860226/ICCAD-2021-B, https://github.com/jacky860226/ICCAD-2021-B