# PD programming assignment #1 Rport

**B07901087 電機三 蔡沛洹**

## 1. Files

I wrote this program with C++ and used g++ std11 to compile it. My program is based on the .cpp and .h files that TA provided.

I have added some function into the file, here I will illustrate some of it.

**(1)void update_gain(Cell* base_cell)**

This function run the incremental update algorithm taught in class

**(2)bool check_balance(Cell* moving_cell)**

This function simply checks whether I can move the moving cell

**(3)Cell* find_next_cell(Cell* base_cell);**

Get the base_cell's neighbor cell with gain as large as possible

**(4)Cell* pick_base_cell()**

Pick base cell for this iteration

**(5)void partition_init()**

Do initial partition

**(6)void update_cut_size()**

Calculate cut size

**(7)void gain_init1()**

Gain initialization at the beginning of each iteration

**(8)void move_node_in_bucket(Node* target_node, int src, int des)**

move node in bucket list

**(9)Cell* get_cell_from_node(Node* some_node)**

Convert node to cell

**(10)void reset_net_part()**

Net and cell data initialization

## 2. algorithm

I used FM algorithm taught in class, and applied early stop to reduce the run time.

## 3. Data structure

I only use one bucket list for my program. I also maintain a max_cell pointer to keep track of the cell with largest gain in my bucket list so that

I can choose the base cell more quickly. Other data structure just store some data such as net or cell.

## 4. discussion

I used one bucket list because I thought it will be easier to deal with it. However, since nodes with different parts are mixed together, it took me some effort to run some case that might not happened when using two bucketList. And when the net is unbalanced, it might also took much more time that two bucketList because it needs to traverse all nodes.

## 5. conclusion

Since I paid too much time at debugging, I didn’t have time to try some different method such as using different initial partition way to split cells, comparing performance between using one bucketList and two bucketLists.