# PDA Homework 2 Report 黃博浩

**Q1：How to compile and execute your program, and give an execution example. If you implement parallelization, please let me know how to execute it with single thread.**

--HOW TO COMPILE ：

使用make就可以直接編譯執行檔，會在目前目錄產生main執行檔，目錄中需存有Makefile、main.cpp，而執行指令 make clear就可以清除。
執行的部分，執行命令為： ./main cell_file.cells net_file.nets out_file
argv[1] = cell file, argv[2] = net file, argv[3] = output file

--HOW TO RUN：

主要有3個參數：cell, net, output_file
Usage：./main ../testcases/<.cell> ../testcases/<.nets> ../testcases/<.out>
ex： ./main ../testcases/p2-1.cells ../testcases/p2-2.nets output

**Q2：The final cut size and the runtime of each testcase.**

| p2-1 runtime | p2-2 runtime | p2-3 runtime | p2-4 runtime | p2-1 cutsize | p2-2 cutsize | p2-3 cutsize | p2-4 cutsize |
|---|---|---|---|---|---|---|---|
| 0.009 | 0.209 | 42.0741 | 588.983 | 6 | 170 | 3390 | 44475 |

**Q3：Runtime = $TIO$ + $Tcomputation$. For each case, please analyze your runtime and find out how much time you spent on I/O and how much time you spent on the computation**

| | IO time | computation time | total time |
|---|---|---|---|
| p2-1 | 0.00181222 | 0.00742984 | 0.00924206 |
| p2-2 | 0.041481 | 0.171021 | 0.209502 |
| p2-3 | 0.691784 | 41.3823 | 42.0741 |
| p2-4 | 1.45836 | 587.524 | 588.983 |

**Q4：The details of your implementation containing explanations of the following questions:**

1. **Where is the difference between your algorithm and FM Algorithm described in class? Are they exactly the same?**

在initialize的部分，課堂中的FM Algorithm是做random cut隨機分成A、B兩個group，但這樣可能會發生unbalance的情況，而為了能讓初始解更好，我先將所有cell從大到小做sorting排序好，並依序將大的cell分配給A、B group，透過這樣的方式在一開始時可以確保初始解為balance。

而在終止FM partition的部分，原本的程式是將最後一個max cell做swap後才停止，但這樣會花非常多的時間，並且在cell值很大時，也要花較多時間才能找到max partial sum，因此我的終止條件為：當max_cell gain值小於 -(max_pin_size/2) 時，則終止目前這輪的FM algorithm，並且為了避免此解卡在local optimal，我做了10輪的FM parition，希望能透過這方法找到較好的解，但是當cell size大於150000時，考慮到了執行時間，將執行輪數更改為5輪。

2. **Did you implement the bucket list data structure?**

Yes，我有實作bucket list，我是以map的方式來實作，gain值為其index，範圍為+max _pin_size ~ -max_pin_size，每個index對應一個list，list紀錄相對應的所有cell id，而每個cell struct中會存有一個指向bucket list的指標(iterator)，當需要動態做刪減時，只需要指到該位址就可以在 O(1)的時間更改bucket list。

3. **How did you find the maximum partial sum and restore the result?**

我用一個queue來記錄每個iteration下存放的max cell，並用sum 紀錄從頭到目前iteration的max cell gain總和，當max_partial_sum小於sum時，就更新現有的max_partia l_sum值為sum，並記錄目前的iteration次數，以此類推，當最後結束時，依序pop出最佳解的iteration次數，有了這些資訊後就可以將不想要移動的cell 恢復成原本的 group。

4. **Please compare your results with the top 5 students' results from last year and show your advantage either in runtime or in solution quality. Are your results better than them?**

**Top 5 of last year!!!**

| rank | 2-1 runtime | p2-2 runtime | p2-3 runtime | p2-4 runtime | p2-1 cutsize | p2-2 cutsize | p2-3 cutsize | p2-4 cutsize |
|---|---|---|---|---|---|---|---|---|
| 1 | 0.001 | 0.014 | 1.058 | 10.982 | 8 | 197 | 1275 | 46719 |
| 2 | 0.006 | 0.29 | 431.588 | 1090.998 | 6 | 255 | 2633 | 42135 |
| 3 | 0.015 | 0.423 | 29.066 | 772.805 | 6 | 511 | 1564 | 45419 |
| 4 | 0.041 | 1.307 | 679.534 | 1471.833 | 6 | 221 | 1630 | 46323 |
| 5 | 0.002 | 0.04 | 0.97 | 35.163 | 23 | 225 | 9398 | 46432 |

| p2-1 runtime | p2-2 runtime | p2-3 runtime | p2-4 runtime | p2-1 cutsize | p2-2 cutsize | p2-3 cutsize | p2-4 cutsize |
|---|---|---|---|---|---|---|---|
| 0.009 | 0.209 | 42.0741 | 588.983 | 6 | 170 | 3390 | 44475 |

以執行時間的部分來說，我比第4名還要好，並且在cutsize上，除了第4比測資外，都有很好的結果，特別是在最後一筆測資，cutsize是前5名中第2高的，但執行時間是第3名，雖然犧牲的是執行時間，我覺得真的很難兩全其美，希望未來能找到更好的implement方式達到更好的效果。

5. **What else did you do to enhance your solution quality or to speed up your program?**

   - 輪數限制10 ( cell size>150000時輪數為4 ） or  max_partial_sum為0時終止
   - 用map來實作bucket list，讓搜尋時間保持在O(1)
   - 終止條件：當max_cell gain值小於 -(max_pin_size/2) 時，則終止目前這輪的 FM algorithm
   - 每次搜尋max gain cell時，都先考慮是否balance，可以避免在執行結束後才判斷是否balance而花費過多execution time

6. **What have you learned from this homework? What problem(s) have you encountered in this homework?**

   這次作業算是目前coding生崖中少數implement的大型程式，除了更加熟悉c++ STL (ex：map、vector)，我覺得學到最多的還是學到如何Debug，從只會用cout大法到懂得用parser參數指定Debug方式，覺得學習到很多coding技巧，至於在碰到的問題上面，一開始implement bucket list的效果非常差，。

7. **If you implement parallelization, please describe the implementation details and provide some experimental results.**

   NO，我沒有使用平行化。