(1) 林威辰 109062531

(2) 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.
Compile:
$ make
Execute:
$ ../bin/hw2 < path/*.nets> < path/*.cells> <path/*.out>
Example:
$ ../bin/hw2 ../testcases/p2-3.nets ../testcases/p2-3.cells ../output/p2-3.out
(3)

	Case1	Case2	Case3	Case4	Case5
Cut size	30	765	16893	51809	144533
Run time	0.007s	3.2s	31.34s	45.38s	114s

(4) 用 clock()去計算 FM 的時間.cell 使用 map 來儲存，可以解決 cell 不連續的問題，也不需要使用 hash function.

	Case1	Case2	Case3	Case4	Case5
Run time	0.007s	3.2s	31.34s	45.38s	114s
I/O time	0.00s	0.04s	0.65s	1.15s	2.27s
Computation time	0.07s	3.16s	30.69s	44.23s	111.73s

(5) The details of your implementation containing explanations of the following questions:
I. Where is the difference between your algorithm and FM Algorithm described in class? Are they exactly the same?
我的設計都跟課堂上的一樣.
II. Did you implement the bucket list data structure?
If so, is it exactly the same as that described in the slide? How many are they? If not, why? You had a better data structure? Or, is bucket list useless?
我有實作 bucket list，我是使用 map<int,<list>>來存 bucket list,因為 map 可以使用負數來做 index,在設計上不用額外的 overload.
III. How did you find the maximum partial sum and restore the result?
我是是記錄下每個 partial sum，找到最大的那個 step,存下要移的 cell 和每

個 cell 原本的位置,最後用這兩個 array 還原結果.

IV. 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?

If so, please express your advantages to beat them.

If not, it's fine. If your program is too slow, then what do you reckon the bottleneck of your program is? If your solution quality is inferior, what do you think that you could do to improve the result in the future?

我的 i/o time 在第二次改進後已經相當不錯了可是瓶頸可能會在 initial gain 的時候，這部分可能會需要改整個 data structure，我還沒有想到改進的方法.
Solution quality 很差的改進方法則是可以考慮在 initial 的時候使用 random partition 先跑個數百次找到比較好的 initial cut 在進行接下來的演算法

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

我原本在 i/o 的時候想使用 buffer 一次讀進來想加快讀取的速度，但是在處理 buffer 的時候花了很多時間，也許是因為我 cells 和 nets 都是用 vector 存，所以會花很多時間，第二次補交的時候我把 cell 改用 map 存所以不需要使用 hash function，讓我的速度大大降低.

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

1. 在設計程式的時候可能要先花時間來決定要使用那些參數和大致的架構，要避免中途加入額外的參數，這會使得邏輯容易有錯誤，且難以 debug.
2. 分析輸入的檔案，我在一開始測試我的 i/o function 使用了 case 1 和 case 5，這兩個檔案的 cell 都是連續的，導致我最後尾聲測試 case 2,3,4 的時候才發現有不連續的 cell number，所以我用 map 則可以不受這個影響.

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