Programming Assignment Report

電機碩一

R06921048

李友岐

1. 演算法流程 (Algorithm Flow) (6pt)

- **Parsing → Partition → Write Output**

  其中最關鍵的當然是 **Partition** 的部份。

  主要就是參考上課投影片裡面有關 **F-M Heuristic** 之部份。

  Parsing 是順著 **Net** 順序讀下來，然後依序建好 **Cell Array** 和 **Net Array**

- **Partition Algorithm**

1. 把前面一半的 **Cell** 分入 **A** 組，剩下一半分入 **B** 組，當作初始分組。

2. 接下來算出各個 **Net** 在 **A** 組和 **B** 組連接了多少 **Cell**。

3. For each cell, initial gain=0. If F(n)==1, then gain++ ; If T(n)==0, then gain-- ;

4. Use two bucket lists to restore the gain of all cell.

5. Move the cell with max gain, but keep the balance.

6. Update the max gain in O(1) time by fast insertion and removal.

7. Keep moving cell until all the cells are locked.

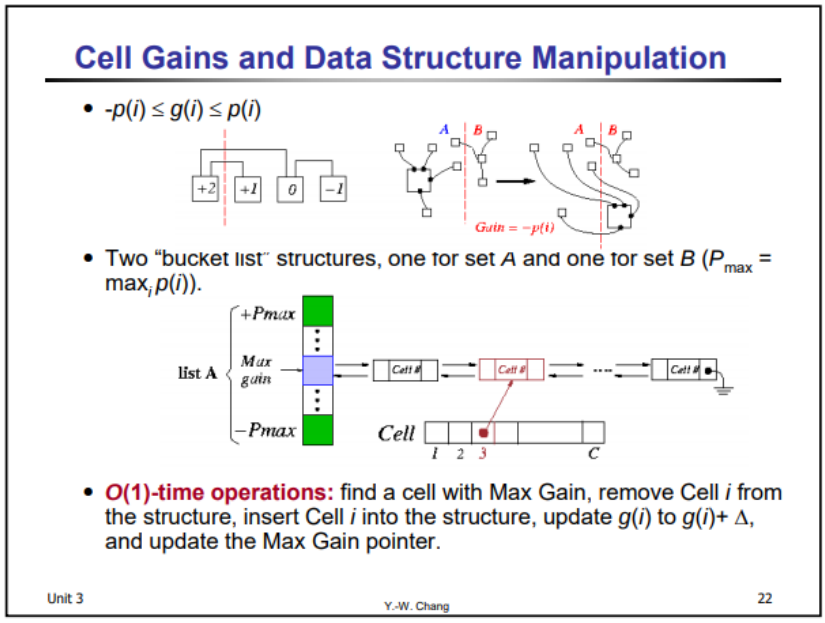
8. Doing same iteration again.

8. Do not terminate until largest partial sum<=0

- 整個 **Partition** 的時間複雜度大約為 **O(P)**，其中 **P** 是**# of total pins**。

2. 資料結構 (Data Structure) (6pt)

Cell array		Net array	
C1	Nets 1, 2	Net 1	C1, C2, C3, C4
C2	Nets 1, 3	Net 2	C1, C5
C3	Nets 1, 4	Net 3	C2, C5
C4	Nets 1, 5, 6	Net 4	C3, C6
C5	Nets 2, 3	Net 5	C4, C6
C6	Nets 4, 5, 6	Net 6	C4, C6



a. Cell array 是針對每個 cell 都建立一個 std::vector，每個 cell 所連接的 net 就存在相對應的 vector 裡。

b. Net Array 也是針對每個 Net 都建立一個 std:vector，每個 net 所連接的 cell 就存在相對應 vector 裡。

c. A 組和 B 組各建立一個 bucket list，將 gain 相同的 cell 存在一起，以 pointer 連接形成一 linked list。由於 -Pmax<=cell gain <= Pmax，因此每個 bucket list 只需要 2*Pmax+1 空間的 std::vector 即可。此資料結構的優點是可以快速的更新 Max Gain Pointer。

- **整個 Partition 的空間複雜度也約為 O(P)，其中 P 是# of total pins。**

3. 問題與討論 (8pt)

Q: 雖然說整個 partition 的時間複雜度理論上是 linear time，但實作中感覺有不少細節須特別注意?

A: 建造 cell array 和 net array 是 linear time 沒問題。用 bucket list 來 update max gain 是 constant time 也沒問題，因為 linked list 增加與刪除節點十分迅速。但如何有效率的在每個 cell 移動後和每次 iteration 結束後更新所有的 cell gain，是個蠻值得注意的細節。如果用錯方法，那光這部份就會耗費許多時間。我一開始是每移動一次就重新算所有的 cell gain，但後來發現這樣太慢。所以改成判斷有無特定 cell 的 F(n)=1 或 T(n)=0，藉此來更新 cell gain，沒影響到的 cell 就維持原本的 gain。但實作後發現這樣速度還是不夠快，也許我需要再花些時間思考，釐清自己的盲點，希望下次的作業能寫出更佳效能的程式。

4. 實驗結果

	Input 0	Input 1	Input 2	Input 3	Input 4	Input 5
Cut Size	20398	1354	2426	28693	48914	149162
Time	100 s	0.24 s	0.62 s	23 s	99 s	1305 s
Memory	65 mb	1.7 mb	3.2 mb	28 mb	60 mb	160 mb