CS6135 HW2 Report

109062610 李至弘

- How to Compile: In /HW2/src/ enter following command:

  $ make

- How to Execute: In /HW2/src/ enter following command:

  $ ./<exe> <cell file> <net file> <output file>

Testcases	Cut size	Runtime(sec)	I/O time(sec)	FM time(sec)
P 2-1	5	0.004	0.002	0.002
P 2-2	599	0.267	0.013	0.254
P 2-3	14367	369.298	0.221	369.077
P 2-4	48329	523.756	0.428	523.327
P 2-5	127886	554.798	1.161	553.636

1. The details in my 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?

Ans.

實作上幾乎跟 FM 演算法內容差不多，一樣是建立兩個 bucket lis 來儲 存不同 set 的 cell，並利用 double linked list 實作 buck list，並利用 vector 儲存每個 cell 的 pointer，可以直接做存

取，耗費時間為 O(1)。

在更新每一個 cell 的 gain 值時一樣和 FM 判斷該 cell 所在的 net 是否為 critical，並將每一步 gain value 儲存下來，以便在結束時找到最佳解。

(2) Did you implement the bucket list data structure?

Ans.

和 FM 演算法一樣使用 bucket list 儲存每個 cell gain 值，總共兩個 bucket list，每個儲存 gain 值範圍從 +max pin value ～-max pin value。

(3) How did you find the maximum partial sum and restore the result?

Ans.

紀錄最佳的步數以及額外使用一個 vector<string> 來記錄每一個搬運過的 cell ，在最後將所有大於最佳步數的那些被搬運的 cell 做回復，僅保留小於等於最佳步數的那些 cell 結果。

(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?

Ans.

(a) Final cut size

我覺得我的 final cut size 比他們差的原因應該是因為我一開始的 initial partition 沒有分得很好，因為我是直接從按照讀取.cells 檔中 cell 的順 序開始做 partition，沒有額外做其他處理。

(b) Run time

而時間耗費比他們久的原因應該是我的 FM 終止條件設得不好，我是 從每個 bucket list 的最大 gain value 開始往下做檢查，一直到所有的 bucket list 為空才作結束，也就是我將所有的 cell 一直做搬運直到沒有 unlocked cell 滿足 balance factor，所以耗費時間過久。

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

Ans.

在 initial partition 的時候有試著將 cell 按照 size 大小排序，並從由小的 先 做搬動或是從大得先做搬動，但是沒有對 cut size 或是 time 有多大的 幫助。 Bucket list 原本是利用 vector 儲存每個 cell 的 address，不過到資料量 過大的時候耗費時間太久，所以改用 double linked list 做儲存來加速 bucket list 的更新。

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

Ans.

在這份作業中更了解在讀取資料時，該用什麼資料結構儲存資料的重要性，選擇的不合適的資料結構可能因為資料量大造成程式耗費時間過久，執行時間較沒效率，而在對記憶體進行存取時候應該要更小心謹慎，並且因為是第一次實作一個小型的演算法，算是對我來講很有意義的一份作業，讓我從中學到很多。