CS6135_HW2 Report

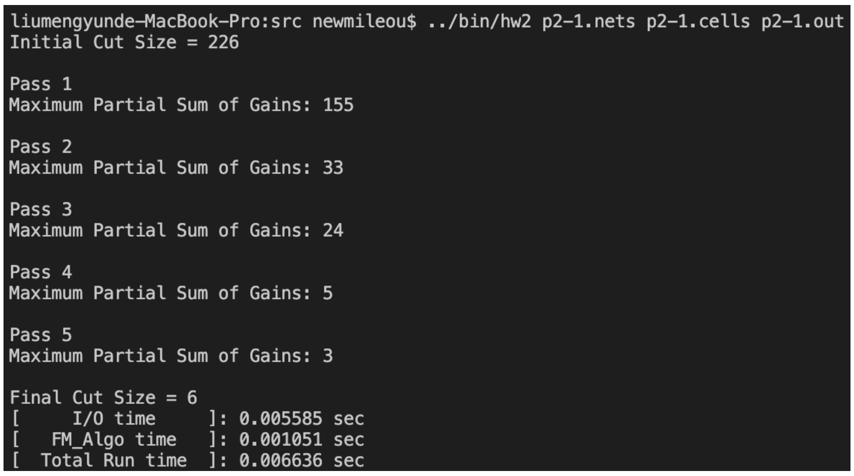
109062648 柳孟芸

1. **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
- In "src/" directory, type the command:
  ```
  $ make
  ```
  It will generate the executable file "hw2" in "bin\" directory.
- If you want to remove it please type the command:
  ```
  $ make clean
  ```

➤ How to execute
- In "src/" directory, enter the following command:
  Format: ..bin/<exe> <nets file> <cells file> <output file>
  e.g.:
  ```
  $ ../bin/hw2 p2-1.nets p2-1.cells p2-1.out
  ```
  --Note: output file will generate in "output\" directory.
- In "bin/" directory, enter the following command:
  Format: ./<exe> <nets file> <cells file> <output file>
  e.g.:
  ```
  $ ./hw2 p2-1.nets p2-1.cells p2-1.out
  ```
  --Note: output file will generate in "output\" directory.

➤ Execution example



```
liumengyunde-MacBook-Pro:src newmileou$ ../bin/hw2 p2-1.nets p2-1.cells p2-1.out
Initial Cut Size = 226

Pass 1
Maximum Partial Sum of Gains: 155

Pass 2
Maximum Partial Sum of Gains: 33

Pass 3
Maximum Partial Sum of Gains: 24

Pass 4
Maximum Partial Sum of Gains: 5

Pass 5
Maximum Partial Sum of Gains: 3

Final Cut Size = 6
[     I/O time     ]: 0.005585 sec
[   FM_Algo time   ]: 0.001051 sec
[  Total Run time  ]: 0.006636 sec
```

➤ I didn't implement parallelization.

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

	Runtime(s)			Cutsize
	I/O Time	Computation Time (FM Algorithm)	Total Runtime	
p2-1	0.006219	0.001403	0.007622	6
p2-2	0.020000	0.110000	0.130000	411
p2-3	0.870000	3.360000	4.230000	779
p2-4	1.680000	10.610000	12.290000	46356
p2-5	5.300000	7.260000	12.560000	125151

3. **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?**

我完全照著上課教的FM演算法實作，沒有什麼不同，一開始先算出initial gain，找出其中擁有最大gain值的cell，搬動base cell之後，更新那些在critical nets上的cell的gain值，最後找出maximum partial sum Gk，並且只做那k個搬動，重複以上每個步驟直到新的Gk<=0為止。

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

我用C++標準程式庫中的一個class叫做map去實作投影片的bucket list：
map <int, Node*> bucket_list[2];
其中bucket_list[0] is for set A & bucket_list[1] is for set B，
利用這兩個bucket list去存各個cell的gain值，
而Node是我自己建立的一個class，是一個double link list的結構，方便做刪除跟插入。

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

我用一個變數叫psgain去儲存partial sum：累加每一次的update_gain演算法中被搬動的base cell的gain值；
並且用一個變數叫best_psgain去儲存the maximum partial sum：在每一次的update_gain演算法最後判斷此輪的psgain是否大於上一輪的best_psgain，若是則將psgain assign 給 best_psgain，若否則不更改best_psgain值。
若best_psgain值大於0則進行下一個pass，並用restore_best()這個function去restore擁有maximum partial sum輪次的相關資訊。

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?**

- 在runtime方面，與top5相比整體之下我5個testcases的結果都還滿不錯的，可能是因為我沒有花過多的時間在initial partition上，也沒有用過多複雜的資料結構去實作此次作業，所以可以得到比較好的時間。
- 在solution quality方面，並沒有所有的結果都比top5好，可能是我就照著投影片的方法做沒有去想其他能增進solution quality的方法，往後有機會可以嘗試更改FM演算法或是更改update gain的方法，或許會得到比較好的結果。

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

在建立initial partition時，我並沒有使用老師上課建議用的排序方法去建立，我只是在讀cell file時簡單用if判斷式判斷當set A的size小於set B的size，我就將被讀取的cell加入set A中，else就加入set B，省去排序的時間。

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

經過這次作業，學習到如何用FM Algorithm去實作Partition，中間花了很多時間回去重新複習上課的影片，才完整了解FM的每個步驟，尤其是updating cell gains的部分花了很多時間搞懂，一開始也沒有把整個程式的架構想好就開始打了，導致中間有很多segmentation fault的問題也是花了不少時間解決。

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

Sorry, I didn't implement parallelization.