CS6135 VLSI Physical Design Automation
Homework 4: Global Placement

Name:賴琮翰

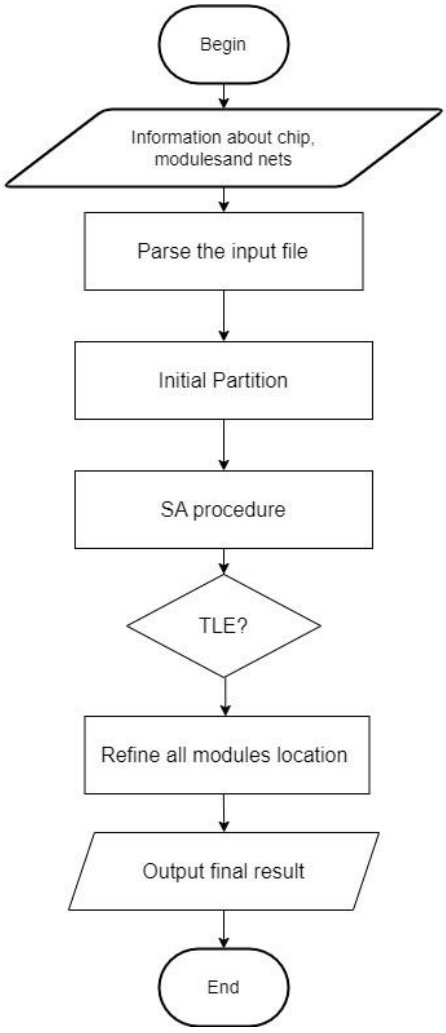
Student ID:112062542

Date:2023/12/19

1. The wirelength and the runtime of each testcase

	Wirelength	Runtime
Public1	**142023324**	**590.00**
Public2	**21057890**	**590.00**
Public3	**1406644454**	**590.00**

2. The details of your algorithm.



本次作業我使用了講義中提到的 **Placement by Simulated Annealing [1]**

[1] Sechen and Sangiovanni Vincentelli, "The TimberWolf placement and routing package," IEEE J. Solid State Circuits, Feb. 1985

方法來實作 global placement。他的 neighborhood structure 包含了兩種 perturb 方式，第一種是 move 單一 cell 的位置，第二種是選兩個 cell 交換位置。並且，他的 cost function 包含三個變數: HPWL、overlap 面積以及單個 row 擺放的長度。

在實作上，我的 cost function 僅考慮 HPWL，原因是後兩者需要花費較多時間計算，會限制 SA 搜索到更好解的時間，HPWL 也只計算擺放前後的 module HPWL 變化量。此外，經過實驗後可以發現第一種 move 雖然能有效地降低 wirelength，但其隨機擺放位置的特性可能會讓 cell overlap 情形非常嚴重，導致無法做後續的 legalize，因此我將第一種 move 的機率設為 0.001，最大化保證能做 legalization，並且保留其優化 wirelength 的可能性。

而在執行完 SA 後，所產生出來的 module 位置，需要檢查他是否超出邊界，把超出邊界的 module 貼著邊界擺放，即完成 global placement。

3. What tricks did you do to enhance your solution quality?

因為是用 **Simulated Annealing** 這種啟發式的演算法，所以參數的設定便非常重要，本次作業的優化也大部分基於參數的調整。首先，因為搜索空間極大，光是 cell 就有一萬個，且 chip area 也是非常大，所以需要將 SA 的搜尋時間拉大，以找出更多的解。因此，初始溫度和終止溫度分別設置極大和極小的數值，而 cooling rate 則設置一個接近 1 的數值，盡可能讓溫度下降過程平緩，最後讓 timer 終止程式的執行。

另外，每個溫度所嘗試的解數目也是很重要的參數，如果設的過大會導致每個溫度的執行時間過久，導致無法收斂到好的解，而若設的太小，則無法充分搜尋 solution space，錯過好的解。因此，同樣經過實驗，可以發現當每個溫度的解設置為 128 個可以在三個 public case 得到相對較好的成績。

	public1	public2	public3
9	161300230	22175221	1512436575
10	159197375	21619712	1506625054
16	155921889	21477394	1464768322
32	152457017	21192450	1412877803
64	150635019	21112729	1407381298
128	149929109	21095221	1429830346
256	150329300	21107235	1405389320
100	151090932	21154871	1404339522

最後，初始解的設置對於 SA 影響也很巨大，因為這次作業用 random placement 做 initial solution，所以 random seed 等同於 initial placement 結果。下表嘗試了不同 random seed 在 public case 上的成績，最終選擇表現最好的 seed 99999 做為程式的亂數種子。

[1] Sechen and Sangiovanni Vincentelli, "The TimberWolf placement and routing package," IEEE J. Solid State Circuits, Feb. 1985

seed			
57978	142175933	21180192	1415649086
9453	150944837	21091106	1402993485
0	149900340	21077229	1400610921
9487	153554425	21277517	1438213020
99999	142029527	21077873	1399012234
696969	150433195	21204117	1428623598

4. Please compare your results with the previous top 5 students' results and show your advantage in solution quality.

非常遺憾的是，我的成績無論在哪個 case 距離 Top5 都非常遙遠，我想原因是 Top5 大部分都是使用 analytical approach 去實作，能保證得到較好的解，而我所寫的 SA 並不能確定得到好的解，而且缺乏考慮 overlap 在後續的優化就不會特別理想，很多時候 SA 得出的成績非常好，但經過 legalize 後線長又跑回去了。我認為如果要單就 SA 繼續去做優化，可以考慮老師課堂中所教的 multilevel SA，透過將 problem size 縮小，讓 SA 表現更好，此外，縮小後的 problem size 在計算 overlap 的速度也能提升許多，就可以在 cost function 把這個條件加進去，應該就可以有不錯的成績。

另外，我在另外一堂課有學到設置一個長寬會隨著溫度冷卻而縮小 window，限制 cell 的 move 和 swap 都只能在該 window 內，讓解不會因隨機性而變動過大。不過在這次作業中，我實作該 window 後效果不是很好，我認為原因還是跟我沒有設置 overlap 條件有關。

5. Implement Parallelization

本次作業我有利用 openMP 實作平行化版本，主要平行化的部分是整個 SA 的部分，透過創立多個 thread，每個 thread 給一個 random seed，以產生不同的 initial placement 來跑 SA，如果某個 thread 結果較佳則更新最佳解，藉由這樣的平行化方式，可以在相同執行時間下搜索更多的解，假設開 16 個執行緒，那就是 16 倍的搜索空間，相較於非平行化的程式能有更高機會得出好的解。不過，在這次作業中因為實際的 HWPL 還需要經過 legalizer 和 detailer placer 才能得到，因此在 global placer 中用平行化找最小的 HPWL，並不一定在後續優化後會是最好的結果。另外，計算整個 placement HPWL 會花費大量的時間，這會壓縮到 SA 可以執行的時間。

[1] Sechen and Sangiovanni Vincentelli, "The TimberWolf placement and routing package," IEEE J. Solid State Circuits, Feb. 1985

下表為 sequential 和平行化版本的結果比較:

	Sequential	Parallel
Public1	**142023324**	**166814793**
Public2	**21057890**	**22025846**
Public3	**1406644454**	**1655123532**

- 編譯方式: 在 src 資料夾內輸入 make，即會在 bin 資料夾內生成 hw4_parallel
- 執行方式: 在 bin 資料夾內輸入

  ./hw4_parallel <.aux file> <.gp.pl file>

  Ex. $ ./hw4_parallel ../testcase/public1/public1.aux ../output/public1.gp.pl

[1] Sechen and Sangiovanni Vincentelli, "The TimberWolf placement and routing package," IEEE J. Solid State Circuits, Feb. 1985