CS6135_HW3_110062619_report

(1)Your name and student ID

Name: 楊淨富

Student ID: 110062619

(2)How to compile and execute your program, and give an execution example.

- How to Compile

In /HW3/src directory, enter the following command:

💡 $ make

💡 It will generate the executable files "hw3" in "HW3/bin/".

- How to execute

In /HW3/src directory, enter the following command:

💡 Usage: ../bin/<exe> <hardblock file> <net file> <pl file> <floorplan file> <deadspace ratio>

```
e.g.:
 $ ../bin/hw3 ../testcase/n100.hardblocks ../testcase/n100.nets ../testcase/n100.pl ../output/n100.floorplan 0.1
```

Or you can run my script:

In /HW3/src, enter the following command: (Take n100 with dead space ratio 0.1 as an example)

💡 $ ./run.sh

💡 $ y

💡 $ n100

💡 $ 0.1

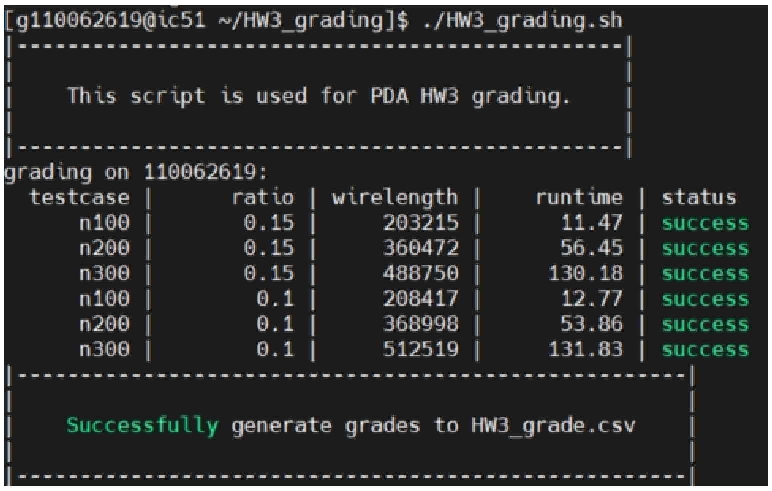
- How to verify

In /HW3/src directory, enter the following command:

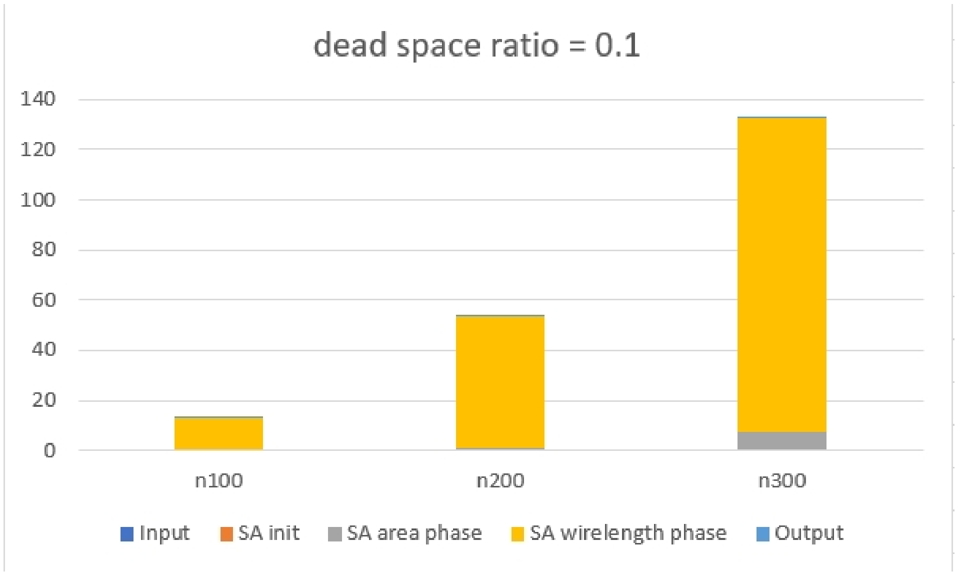
💡 Usage: ../verifier/Verify <hardblock file> <net file> <pl file> <floorplan file> <deadspace ratio>

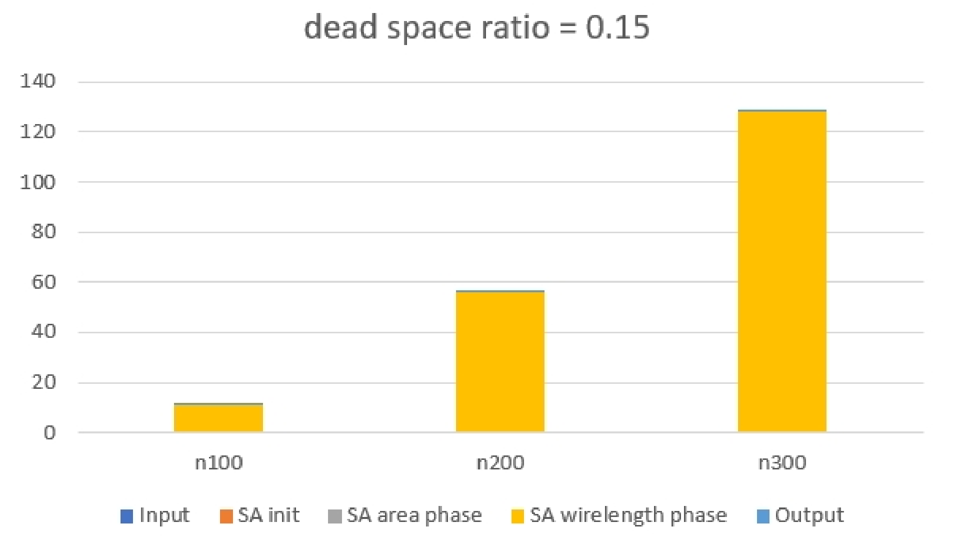
```
e.g.:
  $ ../verifier ../testcases/n100.hardblocks ../testcases/n100.nets ../testcases/n100.pl ../output/n100.floorplan 0.1
```

(3)The wirelength and the runtime of each testcase with the dead space ratios 0.15 and 0.1, respectively. Notice that the runtime contains I/O, constructing data structures, initial floorplanning, computing (perturbation) parts, etc. The more details your experiments have, the more clearly you will know where the runtime bottlenecks are. You can plot your results like the one shown below.



```
[g110062619@ic51 ~/HW3_grading]$ ./HW3_grading.sh
|-----------------------------------------------|
|                                               |
|    This script is used for PDA HW3 grading.   |
|                                               |
|-----------------------------------------------|
grading on 110062619:
  testcase |        ratio | wirelength |     runtime | status
      n100 |         0.15 |     203215 |       11.47 | success
      n200 |         0.15 |     360472 |       56.45 | success
      n300 |         0.15 |     488750 |      130.18 | success
      n100 |          0.1 |     208417 |       12.77 | success
      n200 |          0.1 |     368998 |       53.86 | success
      n300 |          0.1 |     512519 |      131.83 | success
|----------------------------------------------------|
|                                                    |
|    Successfully generate grades to HW3_grade.csv   |
|                                                    |
|----------------------------------------------------|
```

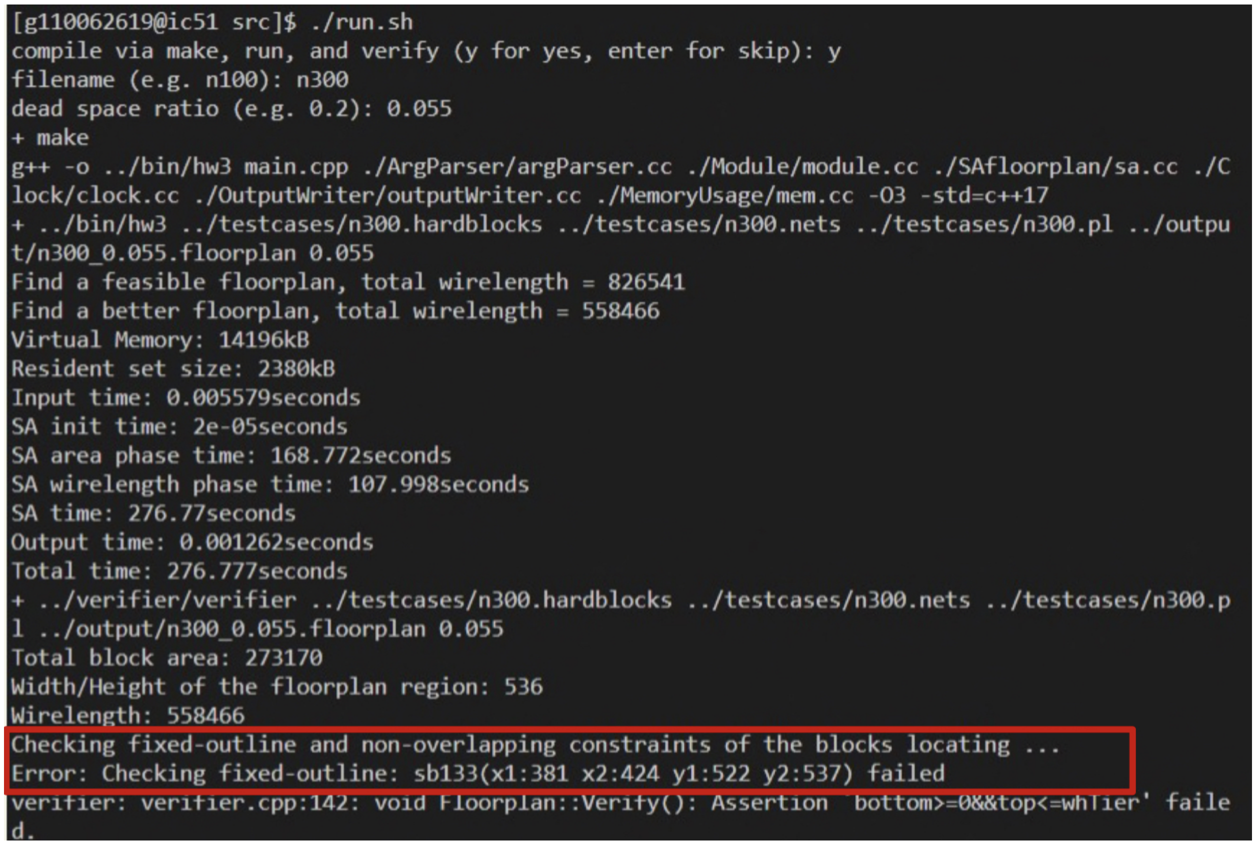




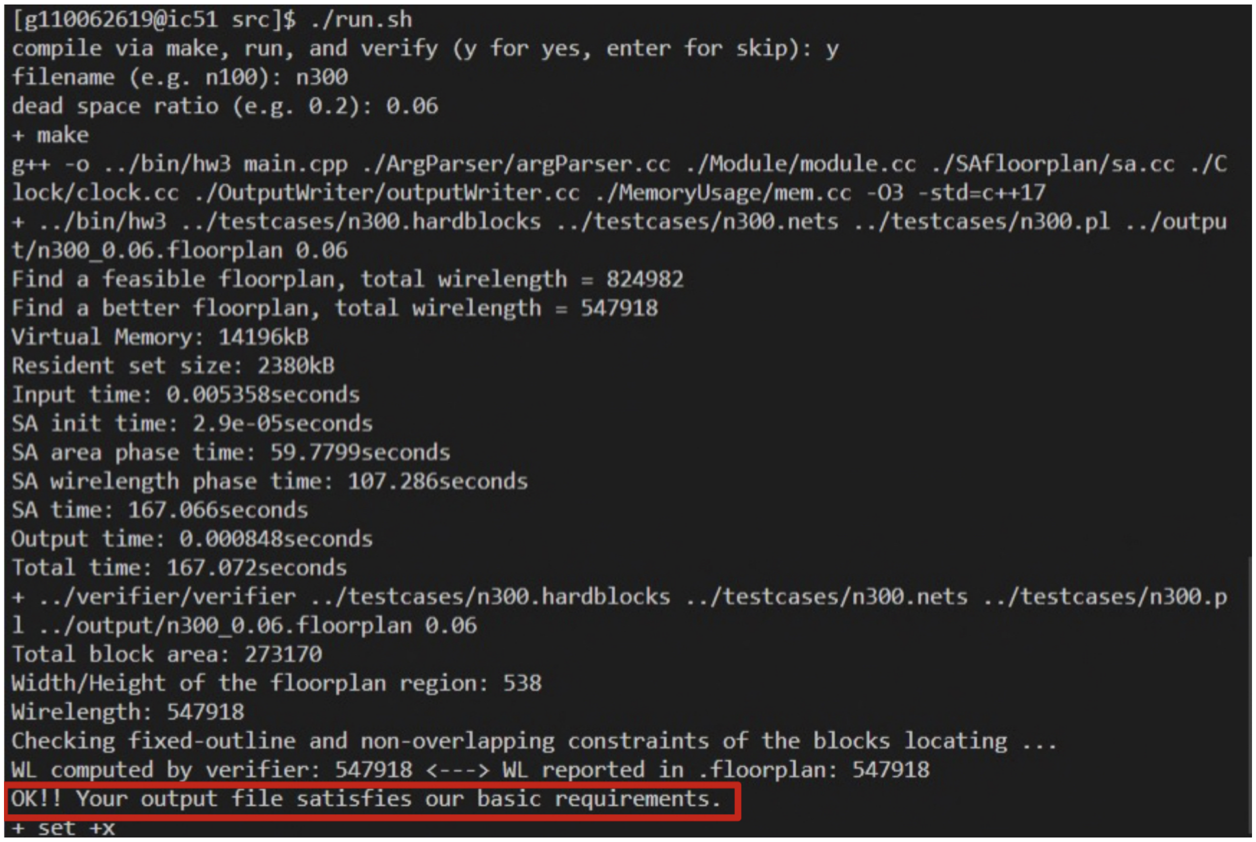
- 由以上兩張圖可以發現大部分時間(單位:秒)都是花在SA的第二個wirelength phase，理由是我的參數主要會讓第二個SA(wirelength phase)的溫度下降比較慢，而且每個溫度也跑比較多case，藉此希望能夠得到比較好的wirelength結果。

(4)Please show that how small the dead space ratio could be for your program to produce a legal result in 10 minutes.

- 我用n300來測試我所能到達的最低dead space ratio，最後的結果是0.06，因為0.055時就已經擺不下去了。(可以參考以下兩張圖片)



```
[g110062619@ic51 src]$ ./run.sh
compile via make, run, and verify (y for yes, enter for skip): y
filename (e.g. n100): n300
dead space ratio (e.g. 0.2): 0.055
+ make
g++ -o ../bin/hw3 main.cpp ./ArgParser/argParser.cc ./Module/module.cc ./SAfloorplan/sa.cc ./C
lock/clock.cc ./OutputWriter/outputWriter.cc ./MemoryUsage/mem.cc -O3 -std=c++17
+ ../bin/hw3 ../testcases/n300.hardblocks ../testcases/n300.nets ../testcases/n300.pl ../outpu
t/n300_0.055.floorplan 0.055
Find a feasible floorplan, total wirelength = 826541
Find a better floorplan, total wirelength = 558466
Virtual Memory: 14196kB
Resident set size: 2380kB
Input time: 0.005579seconds
SA init time: 2e-05seconds
SA area phase time: 168.772seconds
SA wirelength phase time: 107.998seconds
SA time: 276.77seconds
Output time: 0.001262seconds
Total time: 276.777seconds
+ ../verifier/verifier ../testcases/n300.hardblocks ../testcases/n300.nets ../testcases/n300.p
l ../output/n300_0.055.floorplan 0.055
Total block area: 273170
Width/Height of the floorplan region: 536
Wirelength: 558466
Checking fixed-outline and non-overlapping constraints of the blocks locating ...
Error: Checking fixed-outline: sb133(x1:381 x2:424 y1:522 y2:537) failed
verifier: verifier.cpp:142: void Floorplan::Verify(): Assertion `bottom>=0&&top<=whIier' faile
d.
```



```
[g110062619@ic51 src]$ ./run.sh
compile via make, run, and verify (y for yes, enter for skip): y
filename (e.g. n100): n300
dead space ratio (e.g. 0.2): 0.06
+ make
g++ -o ../bin/hw3 main.cpp ./ArgParser/argParser.cc ./Module/module.cc ./SAfloorplan/sa.cc ./C
lock/clock.cc ./OutputWriter/outputWriter.cc ./MemoryUsage/mem.cc -O3 -std=c++17
+ ../bin/hw3 ../testcases/n300.hardblocks ../testcases/n300.nets ../testcases/n300.pl ../outpu
t/n300_0.06.floorplan 0.06
Find a feasible floorplan, total wirelength = 824982
Find a better floorplan, total wirelength = 547918
Virtual Memory: 14196kB
Resident set size: 2380kB
Input time: 0.005358seconds
SA init time: 2.9e-05seconds
SA area phase time: 59.7799seconds
SA wirelength phase time: 107.286seconds
SA time: 167.066seconds
Output time: 0.000848seconds
Total time: 167.072seconds
+ ../verifier/verifier ../testcases/n300.hardblocks ../testcases/n300.nets ../testcases/n300.p
l ../output/n300_0.06.floorplan 0.06
Total block area: 273170
Width/Height of the floorplan region: 538
Wirelength: 547918
Checking fixed-outline and non-overlapping constraints of the blocks locating ...
WL computed by verifier: 547918 <---> WL reported in .floorplan: 547918
OK!! Your output file satisfies our basic requirements.
+ set +x
```

(5)The details of your implementation. If there is anything different between your implementation and the algorithm in the DAC-86 paper, please reveal the difference(s) and explain the reasons.

💡 我的initial normalized polish expression與講義有區別，主要是我並非12V34V56V這樣擺，這樣會變成要花很多時間才能夠擺進fixed outline，我的擺法是，第一列擺到第一個會超出範圍的block時，不讓他擺出範圍外，而是直接擺到第二列，這樣做的好處是，我們可以拿到一個相對較佳的initNPE，此舉能大幅縮短我第一階段的SA，即全力找出能放進範圍的NPE。

💡 我的第一種Move與講義有所不同，講義上是將相鄰兩個block進行交換，而我一開始實作這個方法時，由於會遇到超出範圍的bug，所以我直接改成隨機挑兩個hardblock進行交換而非相鄰的，我認為這樣做會增加擾動的程度，但應該是有利有弊。

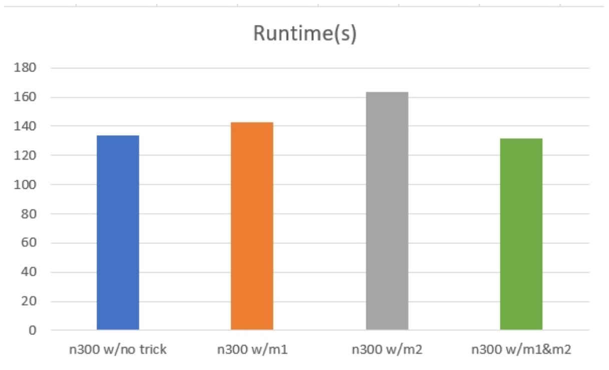
💡 講義上的第三種Move是只考慮operator在前，operand在後的情況，但我會多考慮operand在前，operator在後的狀況，但就需要多check是否違反ballot。

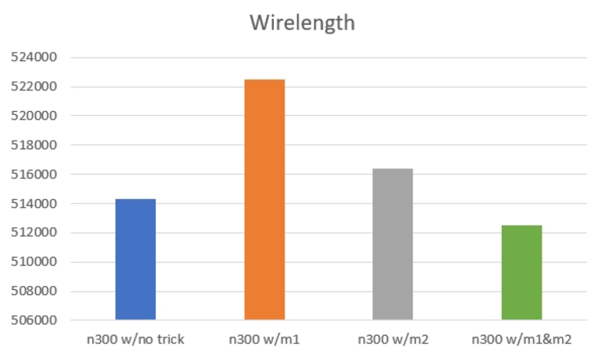
💡 講義上的T0是-delta(avg)/ln(P)，但我直接設成5000。

💡 講義上的SA是一個階段而已，而我的SA有兩個階段，第一階段稱為Area phase，主要是cost function只計算超出fixed outline的面積，而不考慮WL，理由很簡單，因為連region都放不進去就不必每次都重算total wirelength；第二階段稱為Wirelength phase，這個階段我會把SA溫度下降的速率調慢，目的是產生更好品質的結果。

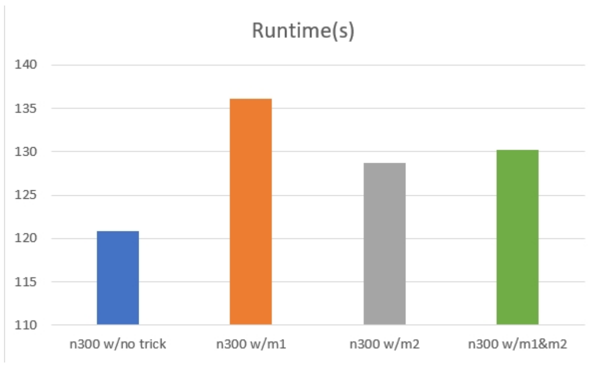
(6)What tricks did you do to speed up your program or to enhance your solution quality? Also plot the effects of those different settings like the ones shown below.

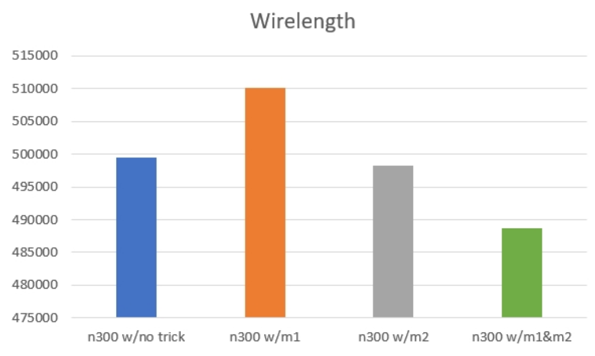
- Method 1: 用.sh跑seed，挑一個結果最好的seed當作我rand的generator。
- Method 2: 在進行selectMove時，如果當前是Area phase，我會只選擇move 1，而不考慮move 2& move3，理由是為了降低在Area phase的時間。





- 上圖是以用n300的test case，在dead space ratio 0.1的結果，可以發現method 1(seed)對wirelength的影響很大，利用method 1& method 2後，runtime有稍微降低且wirelength有大幅降低。

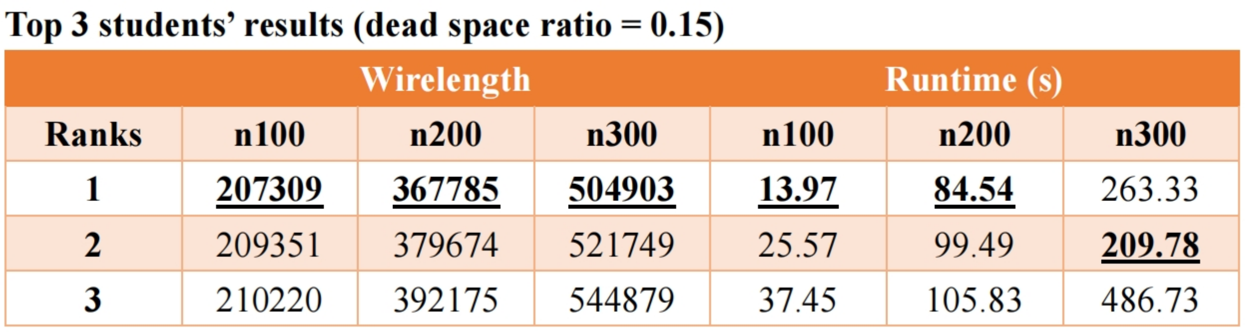




- 上圖是以用n300的test case，在dead space ratio 0.15的結果，比較神奇的是不使用method 1&method 2反而runtime最低，但是wirelength仍然是獲得不小的改善。

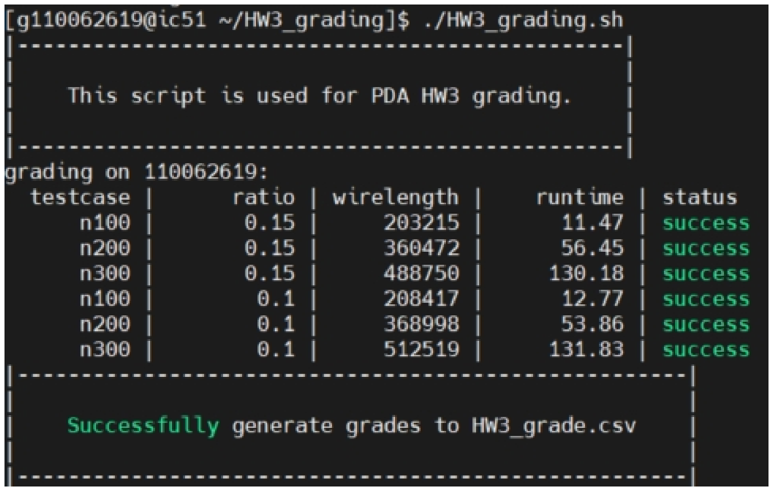
(7)Please compare your results with the previous top 3 students' results for the case where the dead space ratio is set to 0.15, and show your advantage either in runtime or in solution quality. Are your results better than theirs?

✓ If so, please express your advantages to beat them.
✓ If not, it's fine. If your program is too slow, then what could be the bottleneck of your program? If your solution quality is inferior, what do you think that you could do to improve the result in the future?



Top 3 students' results (dead space ratio = 0.15)

	Wirelength			Runtime (s)		
Ranks	n100	n200	n300	n100	n200	n300
1	**207309**	**367785**	**504903**	**13.97**	**84.54**	263.33
2	209351	379674	521749	25.57	99.49	**209.78**
3	210220	392175	544879	37.45	105.83	486.73



```
[g110062619@ic51 ~/HW3_grading]$ ./HW3_grading.sh
|------------------------------------------------|
|                                                |
|     This script is used for PDA HW3 grading.   |
|                                                |
|------------------------------------------------|
grading on 110062619:
  testcase |      ratio | wirelength |    runtime | status
      n100 |       0.15 |     203215 |      11.47 | success
      n200 |       0.15 |     360472 |      56.45 | success
      n300 |       0.15 |     488750 |     130.18 | success
      n100 |        0.1 |     208417 |      12.77 | success
      n200 |        0.1 |     368998 |      53.86 | success
      n300 |        0.1 |     512519 |     131.83 | success
|-----------------------------------------------------|
|                                                     |
|     Successfully generate grades to HW3_grade.csv   |
|                                                     |
|-----------------------------------------------------|
```

我的結果在runtime和wirelength上都比去年rank1的結果好一點點，理由是我把SA拆分成兩個階段，我盡量不要讓第一階段花太多時間，目的是盡快進入第二階段的調整，所以第一階段的溫度下降速度較快以及每個溫度跑的結果較少，因此，能夠有更多時間能夠找更好的解。

(8)If you implement parallelization (for algorithm itself), please describe the implementation details and provide some experimental results.

我並沒有實作平行化。

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

這次作業我學習到設計好的data structure會對後續一些重複操作的function有很大的影響，而且在debug上也比較容易。

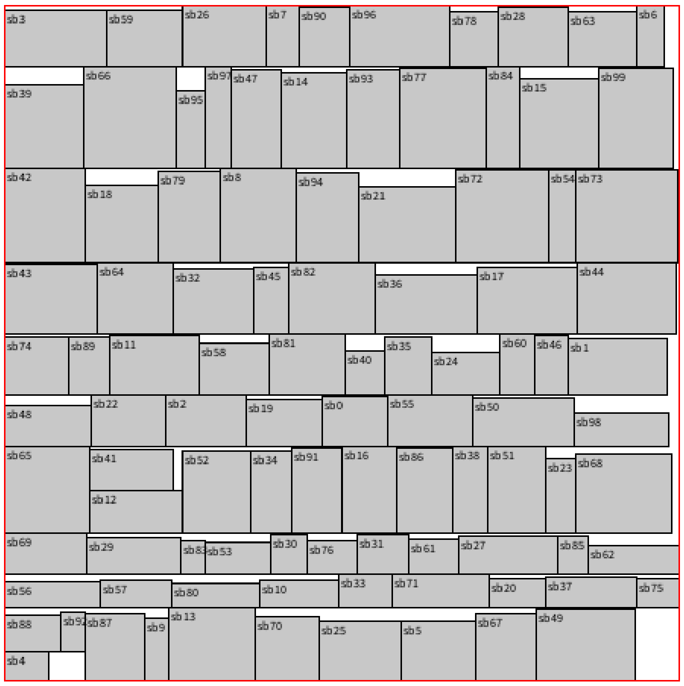
問題1: 我遇到一個bug是會在溫度下降的次數到達一定次數時，整隻程式被kill，我後來在local上跑了一下發現memory會一直上升，所以很顯然某些地方有memory leak，所以透過看MemoryUsage的function來看到底是吃多少memory，也因此，發現我在construct tree時，會一直建tree，也就是一直瘋狂拿資源，但都沒釋放掉，後來我改成宣告兩個vector<TreeNode*>來存放Tree上hardblock和cut的指標並重複利用就可以使得memory不會爆炸。

問題2: 因為我傳入initNPE和一個空的vector(bestNPE)給void SA，希望它能從initNPE開始不斷擾動來拿到一個bestNPE，但我發現因為實作關係，如果我一開始沒有把bestNPE設為initNPE的話，且dead space ratio夠大，可能會直接進入第二階段，這樣的話我bestNPE是空的vector，根本無法計算cost，直接就segmentation fault了。

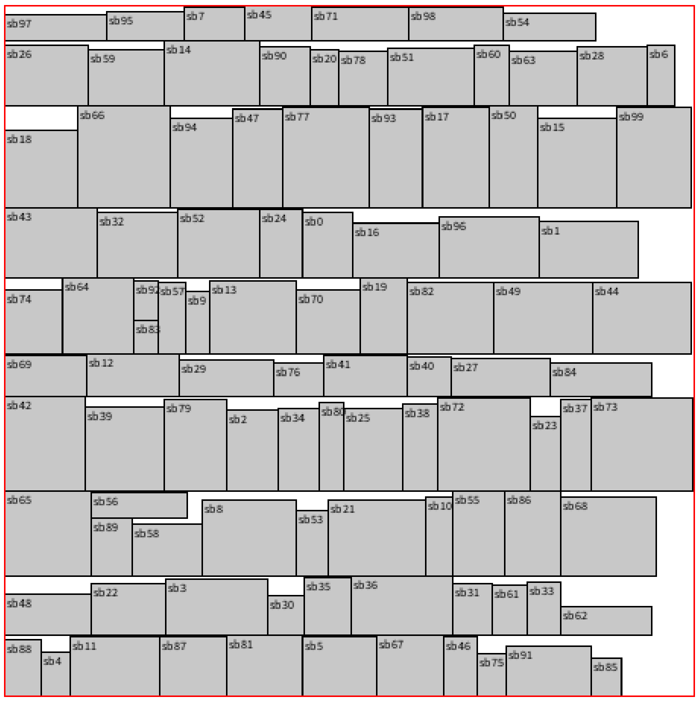
問題3: 我一開始在寫perturb時，想說傳入NPE的reference可以少一些copy的成本，但是後來發現會導致一些floorplan錯誤的情形，理由是perturb完如果delta cost > 0，我不一定會接受這個答案，不能直接就把當前的tryNPE改掉，所以並不能傳reference給perturb。而是應該pass by value，先用複製的NPE去試算，如果接受這個答案才改掉當前的NPE。

(10) Floorplan results visualization

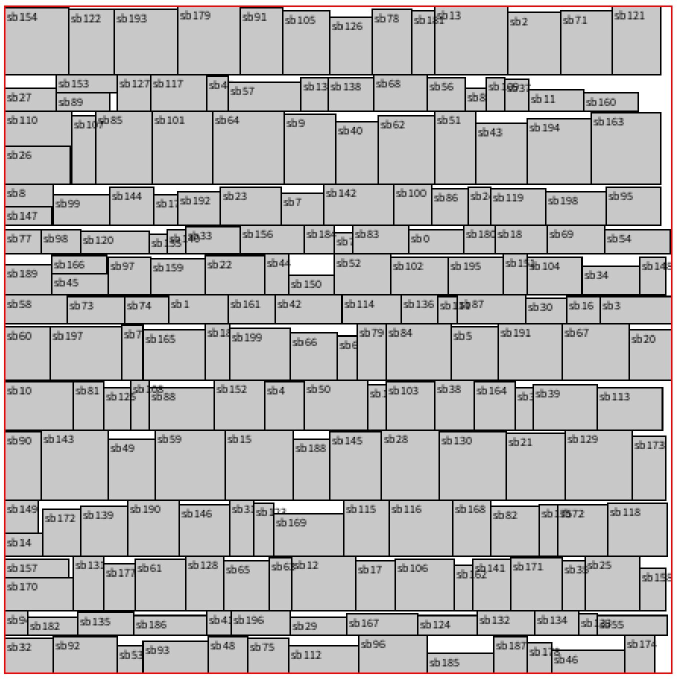
- n100 dead space ratio = 0.1



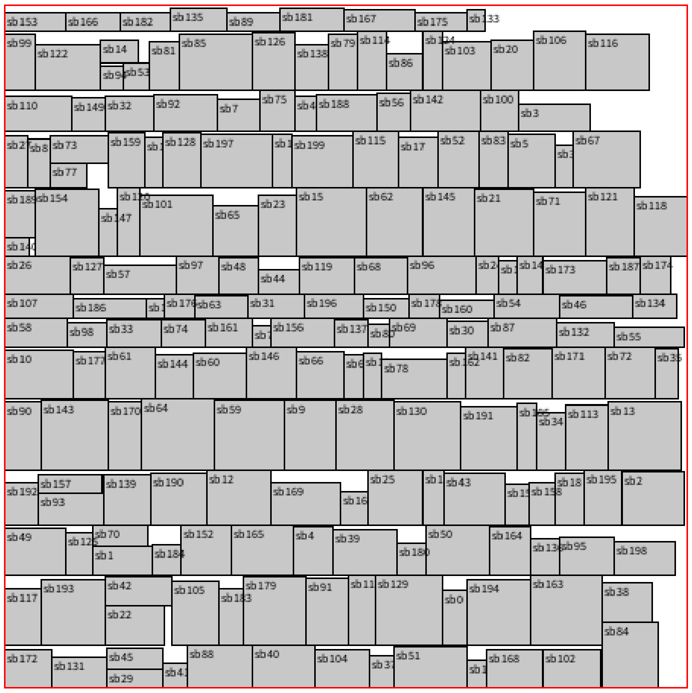
- n100 dead space ratio = 0.15



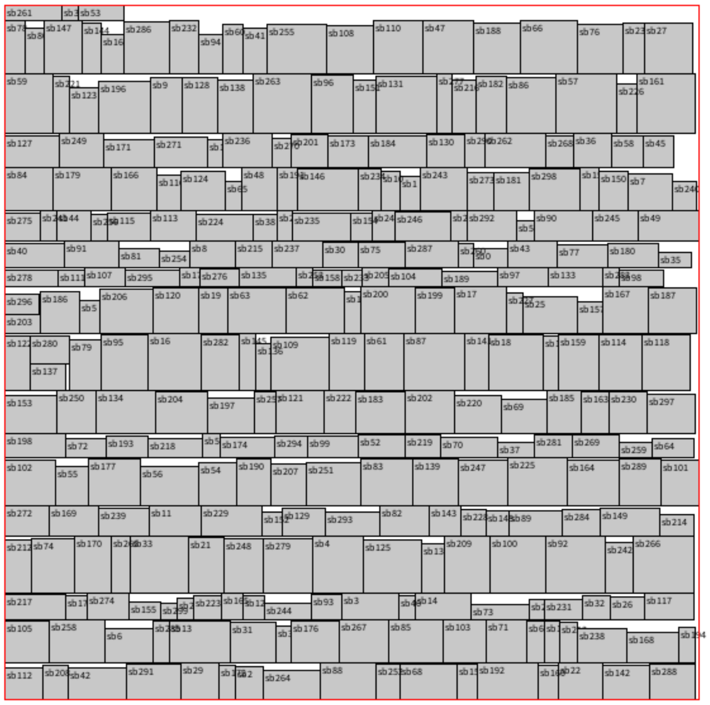
- n200 dead space ratio = 0.1



- n200 dead space ratio = 0.15



- n300 dead space ratio = 0.1



- n300 dead space ratio = 0.15