CS6135_HW4_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 /HW4/src directory, enter the following command:

> 💡 $ make

> 💡 It will generate the executable files "hw4" in "HW4/bin/".

- How to execute

In /HW4/src directory, enter the following command:

> 💡 Usage: ../bin/<exe> <aux file> <result file>

```
e.g.:
 $ ../bin/hw4 ../testcase/adaptec1/adaptec1.aux ../output/adaptec1.result
```

Or you can run my script:

In /HW4/src, enter the following command: (Take adaptec1 as an example)

💡 $ ./run.sh

💡 $ y

💡 $ adaptec1

- How to verify

In /HW4/src directory, enter the following command:

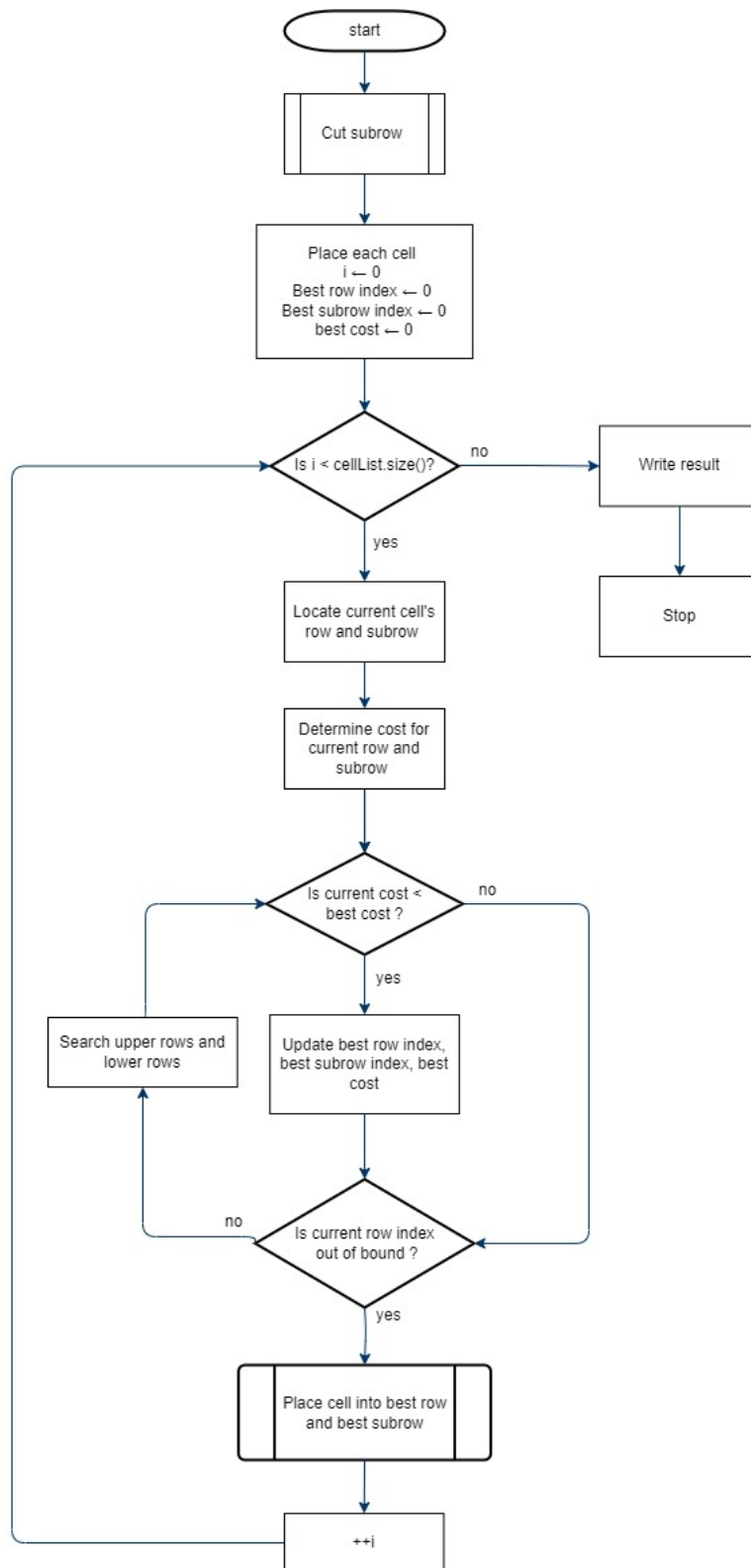💡 Usage: $ ../verifier/verify <aux file> <result file>

```
e.g.:
  $ $ ../verifier/verify ../testcase/adaptec1/adaptec1.aux ../output/adaptec1.result
```

(3) The final score and the runtime of each testcase.

```
[g110062619@ic51 ~/HW4_grading]$ ./HW4_grading.sh
|-------------------------------------------|
|                                           |
|    This script is used for PDA HW4 grading.  |
|                                           |
|-------------------------------------------|
grading on 110062619:
  testcase |  max disp. |    total disp. |   runtime | status
  adaptec1 |     286.84 |    3820287.00 |      0.82 | Maximum displacement constraint was violated for adaptec1.
  adaptec3 |     312.84 |    7370667.00 |      2.83 | Maximum displacement constraint was violated for adaptec3.
     ibm01 |    3372.01 |    5985940.50 |      0.04 | success
     ibm07 |    6163.60 |   31261980.00 |      0.14 | Maximum displacement constraint was violated for ibm07.
     ibm09 |    8961.71 |   48811596.00 |      0.17 | Maximum displacement constraint was violated for ibm09.
|-------------------------------------------|
|                                           |
|    Successfully generate grades to HW4_grade.csv  |
|                                           |
|-------------------------------------------|
```

(4)The details of your implementation. If there is anything different between
your implementation and the algorithm in the ISPD-08 paper,
please reveal
the difference(s) and explain the reasons.

```
                              ┌─────────┐
                              │  start  │
                              └─────────┘
                                   │
                           ┌───────────────┐
                           │ │ Cut subrow │ │
                           └───────────────┘
                                   │
                           ┌─────────────────────┐
                           │   Place each cell    │
                           │       i ← 0          │
                           │  Best row index ← 0  │
                           │ Best subrow index ← 0│
                           │    best cost ← 0     │
                           └─────────────────────┘
                                   │
                              ◇─────────────◇      no    ┌──────────────┐
                     ┌────────│ Is i < cellList.size()? │────────│ Write result │
                     │        ◇─────────────◇             └──────────────┘
                     │             │ yes                         │
                     │        ┌──────────────┐            ┌──────────────┐
                     │        │ Locate current cell's │    │     Stop     │
                     │        │   row and subrow      │    └──────────────┘
                     │        └──────────────┘
                     │             │
                     │        ┌──────────────┐
                     │        │ Determine cost for │
                     │        │  current row and   │
                     │        │      subrow        │
                     │        └──────────────┘
                     │             │
                     │        ◇─────────────◇   no
                     │   ┌────│ Is current cost < │────┐
                     │   │    │    best cost ?     │    │
                     │   │    ◇─────────────◇          │
                     │   │         │ yes               │
                     │ ┌──────────────┐ ┌──────────────┐│
                     │ │Search upper rows│ │Update best row │
                     │ │ and lower rows  │ │index, best sub-││
                     │ └──────────────┘ │row index, best ││
                     │   │              │     cost       ││
                     │   │              └──────────────┘ │
                     │   │                   │           │
                     │   │              ◇─────────────◇  │
                     │   └──no──────────│Is current row  │──┘
                     │                  │index out of bound?│
                     │                  ◇─────────────◇
                     │                       │ yes
                     │                  ┌──────────────┐
                     │                  ││ Place cell into ││
                     │                  ││ best row and    ││
                     │                  ││ best subrow     ││
                     │                  └──────────────┘
                     │                       │
                     │                  ┌──────────────┐
                     └──────────────────│     ++i       │
                                        └──────────────┘
```

- Cite paper:

Abacus: Fast Legalization of Standard Cell Circuits with Minimal Movement

by Peter Spindler, Ulf Schlichtmann and Frank M. Johannes

- 程式的flow

    1. 讀.aux檔，會再透過.aux去讀取.node, .pl, .scl檔

    2. 依照每個row，再看該row上有無terminals來切出這個row的所有subrow
    (即兩個for-loop，外層是on rows，內層是on terminals)

    3. 將每個cell依序執行abacusDP，即算出它當前最佳的擺放位置。

        a. 先算出它當前的row和subrow，並計算出cost(displacement)

        b. 從當前row往上一列一列試擺，必須符合沒有超出最上方row的index條件，並且往上移動的距離必須至少<當前最佳的cost，否則直接break往上的function(searchUpRow)，如果往上能有更好的cost，則更新row, subrow, cBest

        c. 同(b)只是這次改為往下一列一列試擺。

        d. 確定好當前最佳的擺放位置後，更新cell的x_final跟y_final，以及相對應的cluster，如同paper的方式。

    4. 由於有siteWidth的限制，亦即每個cell必須貼齊site擺放，所以需要額外計算每個cell離當前所在的site上，是靠左擺還是靠右擺會比較好。

    5. 將最後的結果寫入.result

- 與paper不同之處:

    💡 paper是從row0開始擺放，但我是從該cell所位在的row開始當作current row，然後再往上試擺，往下試擺，因為paper有提到each cell is first moved to the nearest row (according to the global position)的improvement，而且這樣可以省下一些根本不可能考慮的row的時間。

💡 paper在進行試擺時，會將cell先加進cluster(if exists)，即真的將cell擺進去當前的row和subrow，但是如果擺進去並沒辦法改善cBest的話，還必須還原更動過的地方，例如cluster的nodes，cell的x, y等等，所以我這邊用的是place Virtual cell，即模擬擺放進去的過程，拿出計算擺放過程所需要的weight, position, width來試算，而若真的有改善cost，再存當前的row , subrow index，等到全部該看的row都看完的時候，再真的把cell擺進去，更新對應的cluster。

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

💡 1. 透過選取subrow的方式來speed up program，如(4)所述

💡 2. 透過更改cell的weight來改善solution的quality，如下面兩張圖所示

- 圖1: Final result by initializing cell weight to be width of cell.

```
[g110062619@ic51 ~/HW4_grading]$ ./HW4_grading.sh
|-------------------------------------------|
|                                           |
|    This script is used for PDA HW4 grading. |
|                                           |
|-------------------------------------------|
grading on 110062619:
  testcase |  max disp. |     total disp. |   runtime | status
  adaptec1 |     286.84 |      3820287.00 |      0.82 | Maximum displacement constraint was violated for adaptec1.
  adaptec3 |     312.84 |      7370667.00 |      2.83 | Maximum displacement constraint was violated for adaptec3.
     ibm01 |    3372.01 |      5985940.50 |      0.04 | success
     ibm07 |    6163.60 |     31261980.00 |      0.14 | Maximum displacement constraint was violated for ibm07.
     ibm09 |    8961.71 |     48811596.00 |      0.17 | Maximum displacement constraint was violated for ibm09.
|-------------------------------------------|
|                                           |
|    Successfully generate grades to HW4_grade.csv |
|                                           |
|-------------------------------------------|
```

- 圖2: Initial result by initializing cell weight to be height * width of cell.

```
[g110062619@ic51 ~/HW4_grading]$ ./HW4_grading.sh
|-------------------------------------------|
|                                           |
|    This script is used for PDA HW4 grading. |
|                                           |
|-------------------------------------------|
grading on 110062619:
  testcase |  max disp. |    total disp. |  runtime | status
  adaptec1 |     286.84 |     3820287.00 |     0.96 | Maximum displacement constraint was violated for adaptec1.
  adaptec3 |     345.96 |     7391699.00 |     2.79 | Maximum displacement constraint was violated for adaptec3.
    ibm01  |    4757.27 |     6898485.00 |     0.04 | success
    ibm07  |   10428.03 |    41757756.00 |     0.14 | Maximum displacement constraint was violated for ibm07.
    ibm09  |   16460.71 |    93195344.00 |     0.16 | Maximum displacement constraint was violated for ibm09.
```

- 圖3: method1代表用height*width來當作cell的init weight，method2則是使用width作為cell的init weight
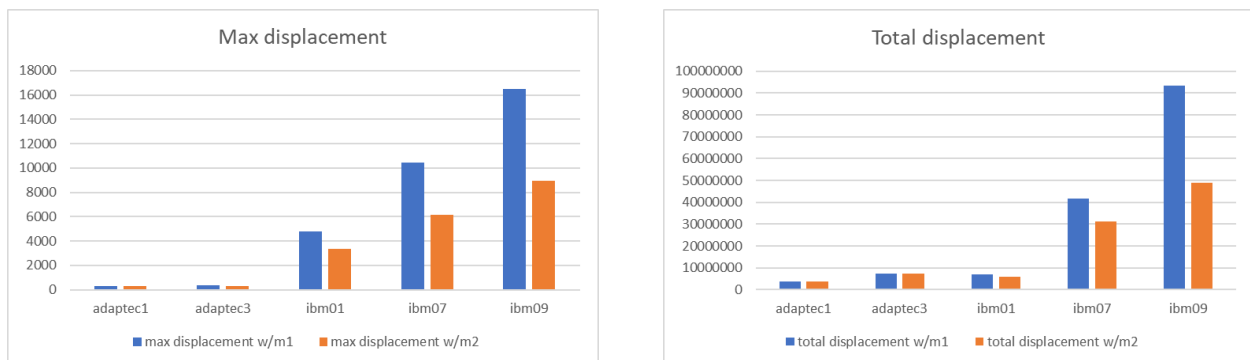
(6)Please compare your results with the previous top 5 students' results, and show your advantage either in runtime or in solution quality. Are your results better than theirs?

√ 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 5 students' results

Ranks	Total Disp. (10^6)					Runtime (sec)				
	ibm01	ibm07	ibm09	ada1	ada3	ibm01	ibm07	ibm09	ada1	ada3
1	5.5	27.86	41.96	3.07	5.08	0.06	0.23	0.35	2.35	4.17
2	6.1	30.24	47.92	3.42	5.47	0.07	0.27	0.33	1.74	3.44
3	6.24	31.63	52.21	3.32	5.21	0.05	0.17	0.2	1.07	3.26
4	6.04	31.11	50.28	3.38	5.27	0.28	1.22	1.22	12.81	58.14
5	5.57	29.9	46.5	3.69	5.48	0.1	0.42	0.48	28.21	99.99

💡 Runtime部分，我全部testcase贏過rank 1的成績。至於solution quality，我在ibm01排行rank2，ibm07排行rank3，ibm09排行rank3，adaptec1和adaptec3則都稍遜一些，可能是因為挑選subrow時和runtime的trade off，或者是我在計算position時有些地方沒寫好，導致答案的品質有點不穩定。

(7)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的部分。

(8)What have you learned from this project? What problem(s) have you encountered in this project?

💡 在cut subrow時，一開始不確定會不會有整個terminal在某個row的左邊，如果有，會導致我延伸了當前subrow的長度，所以我後來先加了一個判斷式，判斷當前的subrow的x_min是否至少會<= 當前terminal的x_max，但後來發現根本不會有這種狀況，理由是global placement本來就會避免掉這種情形。

💡 一開始在切割subrow時，不確定是否會有大的terminal整個涵蓋住小的terminal的問題，導致我在更新subrow的x座標時會有些case不好處理。但後來也發現不會發生，理由同上。

💡 在計算cell的current position該位於某個row的哪個subrow時，一開始我忽略cell可能整個卡在terminal的情況，(即cell->x_global < subrow->x_min && cell->x < subrow->x_max && curSubrowIdx > 0)導致會少考慮一些情形。

💡 改指標指向的cluster的data，同時也不斷更新指標，但最後沒有返回正確的lastCluster指標而是停留在第一個指標，原本想說用指標的指標來修改，但後來發現有些問題作罷，而改用return指標的方式。這個問題會直接導致我collapse有bug，後來有修掉。

💡 有一個小問題是我把output路徑寫死，所以一開始跑verifier都正常，但是跑助教的grading.sh就是跑不出東西，後來才發現output路徑應該要是吃user的parameter才對。

💡 無論我怎麼修正程式都無法符合spec所要求的小於max displacement的條件，僅有一個testcase能夠符合。