CS6135 VLSI Physical Design Automation
Homework 4: Placement Legalization

111062625 蔡哲平

1. How to compile and execute my program.

- Compile: Enter ***src/*** and make, it'll generate the executable file to ***bin/***.

  ```
  $ cd src
  $ make
  ```

- Execute

  ```
  $ ./hw4 [*.aux] [*.result]
  ```

  e.g.

  ```
  $ ./hw4 ../testcase/adaptec1/adaptec1.aux ../output/adaptec1.result
  ```

2. The total displacement, the maximum displacement, and the runtime of each testcase.

	ada1	ada3	ibm1	ibm7	ibm9
Max disp.	216.21	150.76	5151.09	10428.03	13313.83
Total disp.	3292986	5182145	6165040	30817144	50769684

	ada1	ada3	ibm1	ibm7	ibm9
IO time	0.63	1.28	0.04	0.15	0.21
Abacus time	0.54	1.64	0.02	0.09	0.07
Total time	1.17	2.92	0.06	0.24	0.28

```
grading on 111062625:
  testcase |  max disp. |     total disp. |    runtime | status
  adaptec1 |     216.21 |      3292986.25 |       1.17 | Maximum displacement constraint was violated for adaptec1.
  adaptec3 |     150.76 |      5182145.00 |       2.92 | Maximum displacement constraint was violated for adaptec3.
     ibm01 |    5151.09 |      6165040.50 |       0.06 | success
     ibm07 |   10428.03 |     30817144.00 |       0.24 | Maximum displacement constraint was violated for ibm07.
     ibm09 |   13313.83 |     50769684.00 |       0.28 | Maximum displacement constraint was violated for ibm09.
```

3. The details of my implementation:

My program is very similar to the ISPD-08 paper but with the below differences.

I. Slice rows

Since the paper didn't consider fixed cells, which are blockages in this assignment, a row needs to be sliced into sub-rows. For each row, check whether blockages are occupying it. If so, slice the row into multiple sub-rows. Keep doing this until all rows are checked.

II. Finding appropriate sub-row

Since the paper didn't consider sub-rows, I came up with a solution. Instead

of choosing the minimum cost among placing the cell into all sub-rows, which leads to a longer runtime, we can just place it into the nearest sub-row and keep choosing the next sub-row if the current sub-row doesn't have enough space. This way, we can significantly reduce the runtime and still maintain a good legalization result (small displacement).

4. What tricks did you do to speed up your program or enhance your solution quality?

As mentioned above, the trick I used to reduce the runtime is to choose the nearest sub-row instead of exhausting searching all sub-rows. This way, the runtime will be shorter. Below is a comparison table between them. You can see that although using an exhausting search has a better legalization result, the runtime is very long compared to choosing the nearest one. However, there is only a little difference between their legalization result.

(Test case: ada3)

	Exhausting search	Choosing nearest
Max disp.	150.76	150.76
Total disp.	5019211	5182145

	Exhausting search	Choosing nearest
IO time	1.26	1.28
Abacus time	29.72	1.64
Total time	30.97	2.92

5. 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.

Total disp. (10^6)					
Ranks	ada1	ada3	ibm1	ibm7	ibm9
1	**3.07**	**5.08**	**5.5**	**27.86**	**41.96**
2	3.42	5.47	6.1	30.24	47.92
3	3.32	5.21	6.24	31.63	52.21
4	3.38	5.27	6.04	31.11	50.28
5	3.69	5.48	5.57	29.9	46.5
Mine	3.29	5.18	6.16	30.81	50.76

Considering only the total displacement, you can see that my total displacement is a little better than the ranked 3rd student in all test cases.

Runtime (s)					
Ranks	ada1	ada3	ibm1	ibm7	ibm9
1	2.35	4.17	0.06	0.23	0.35
2	1.74	3.44	0.07	0.27	0.33
3	**1.07**	3.26	**0.05**	**0.17**	**0.2**
4	12.81	58.14	0.28	1.22	1.22
5	28.21	99.99	0.1	0.42	0.48
Mine	1.17	**2.92**	0.06	0.24	0.28

Considering only the runtime, you can see that my runtime is a little bit longer than the ranked 3rd student in almost all test cases.

Overall, I think my result may be ranked third compared with the previous top 5 students' results since my total displacement is shorter than the 3rd student's in all test cases, and the runtime is only a little longer than the 3rd student's.

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

   Sorry, I didn't implement parallelization.

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

   I found this assignment more difficult than the previous ones. Although I spent lots of time thinking about how to meet the maximum displacement constraint, I couldn't come up with a good solution to solve it. My program exceeds maximum displacement in almost all test cases, which is quite frustrating.