EDA HW3

江若綾 113062640

(1) How to compile and execute program

Compile program:

Step 1: Enter "HW3/src/"

Step 2: Enter command: $ make

Remove compile program:

Step 1: Enter "HW3/src/"

Step 2: Enter command: $ make clean

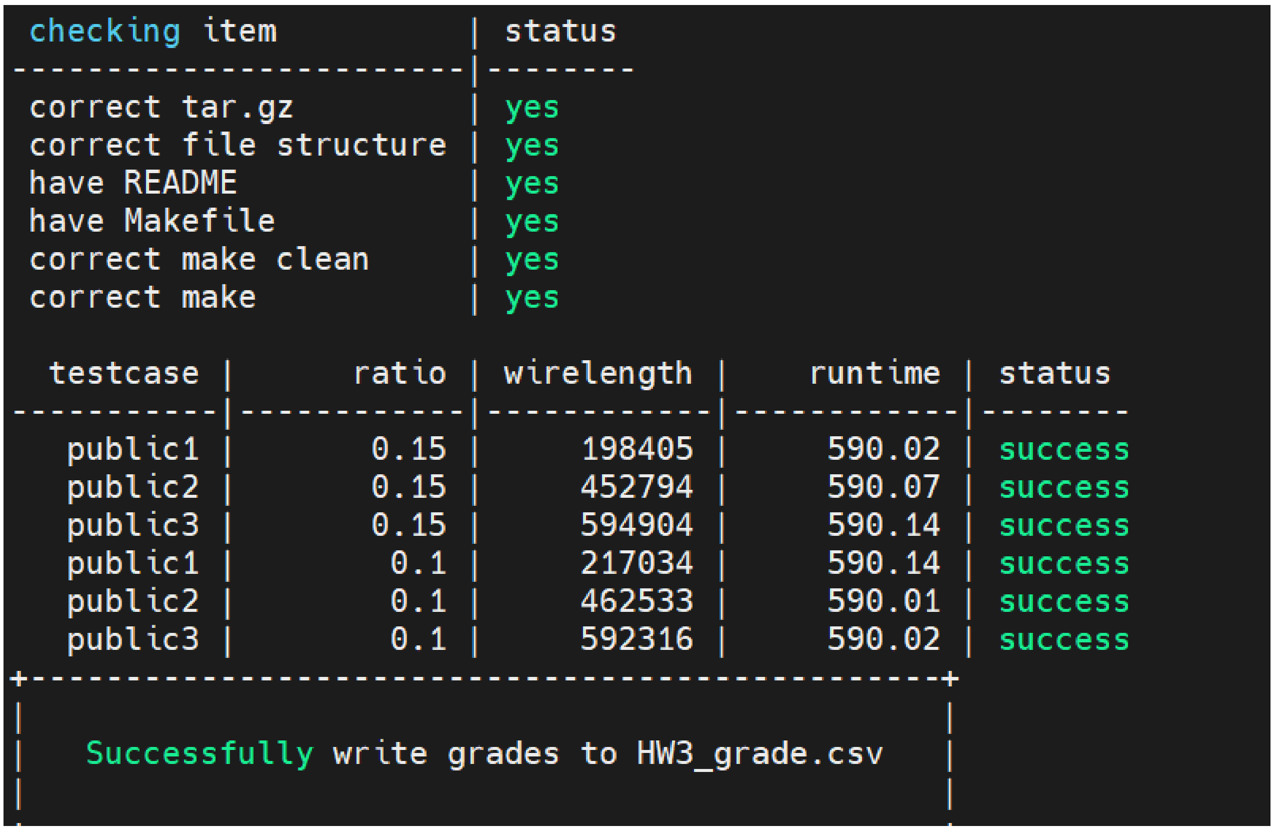
Execute program:

Step 1: Enter "HW3/bin/"

Step 2: Enter command: $ ./hw3 <.txt file> <.out file> dead _space_ratio

Execution example: $ ./hw3 ../testcase/public1.txt ../output/public1.out 0.1

(2) Wirelength and runtime of each testcase with dead space ratios 0.15 and 0.1

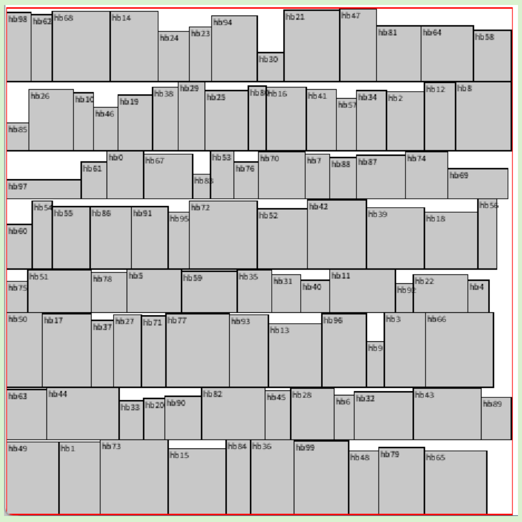
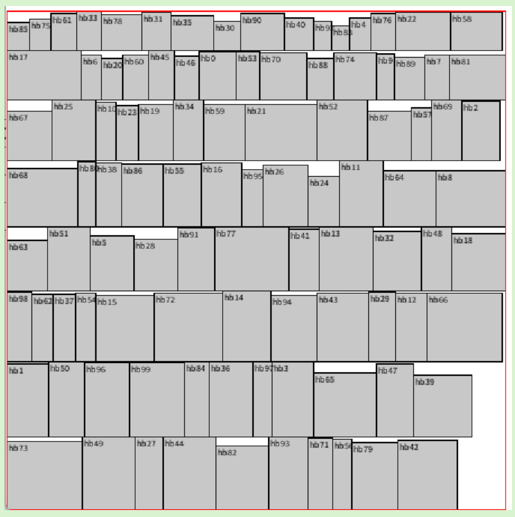
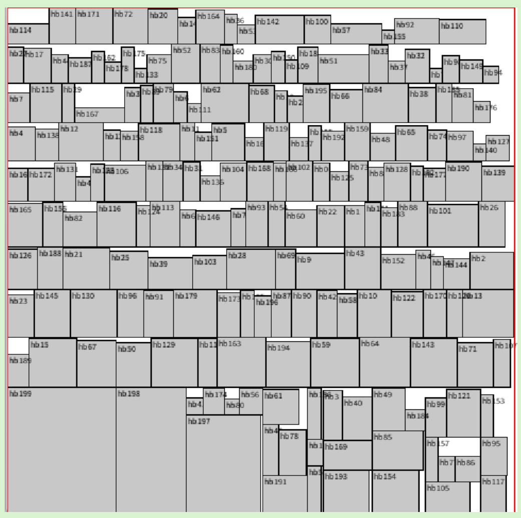
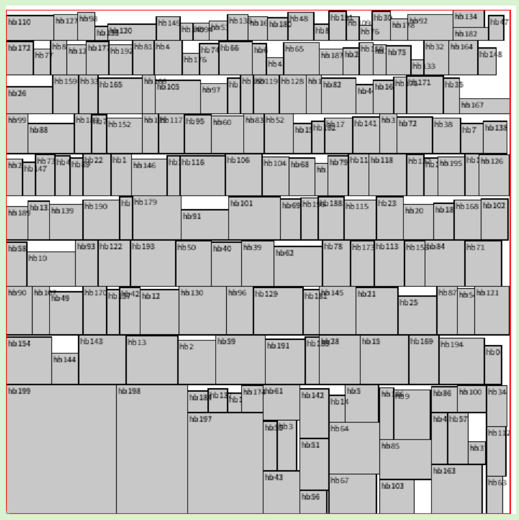
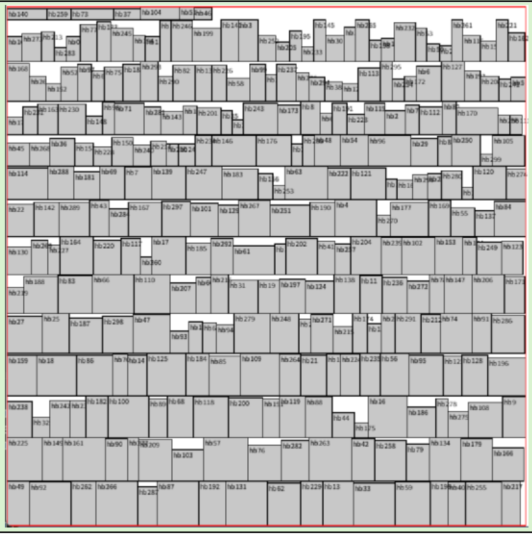
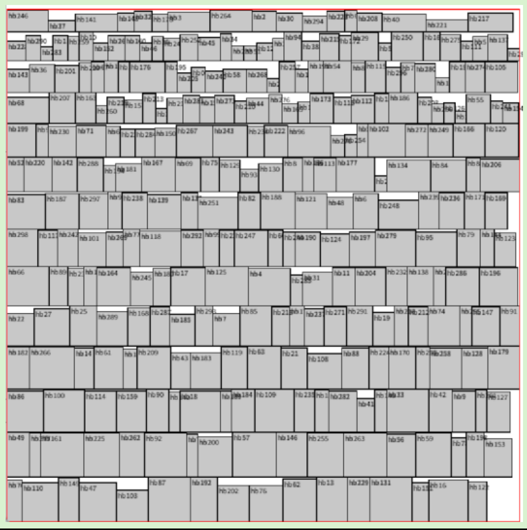


```
 checking item            | status
------------------------|--------
 correct tar.gz           | yes
 correct file structure   | yes
 have README              | yes
 have Makefile            | yes
 correct make clean       | yes
 correct make             | yes

  testcase |       ratio | wirelength |     runtime | status
-----------|-------------|------------|-------------|--------
   public1 |        0.15 |     198405 |      590.02 | success
   public2 |        0.15 |     452794 |      590.07 | success
   public3 |        0.15 |     594904 |      590.14 | success
   public1 |         0.1 |     217034 |      590.14 | success
   public2 |         0.1 |     462533 |      590.01 | success
   public3 |         0.1 |     592316 |      590.02 | success
+-----------------------------------------------+
|                                               |
|   Successfully write grades to HW3_grade.csv  |
|                                               |
```

Testcase /dead space ratio	public1/ 0.15	public2/ 0.15	public3/ 0.15	public1/ 0.1	public2/ 0.1	public3/ 0.1
Wirelength	198405	452794	594904	217034	462533	592316
Run time	590.02	590.07	590.14	590.14	590.01	590.02

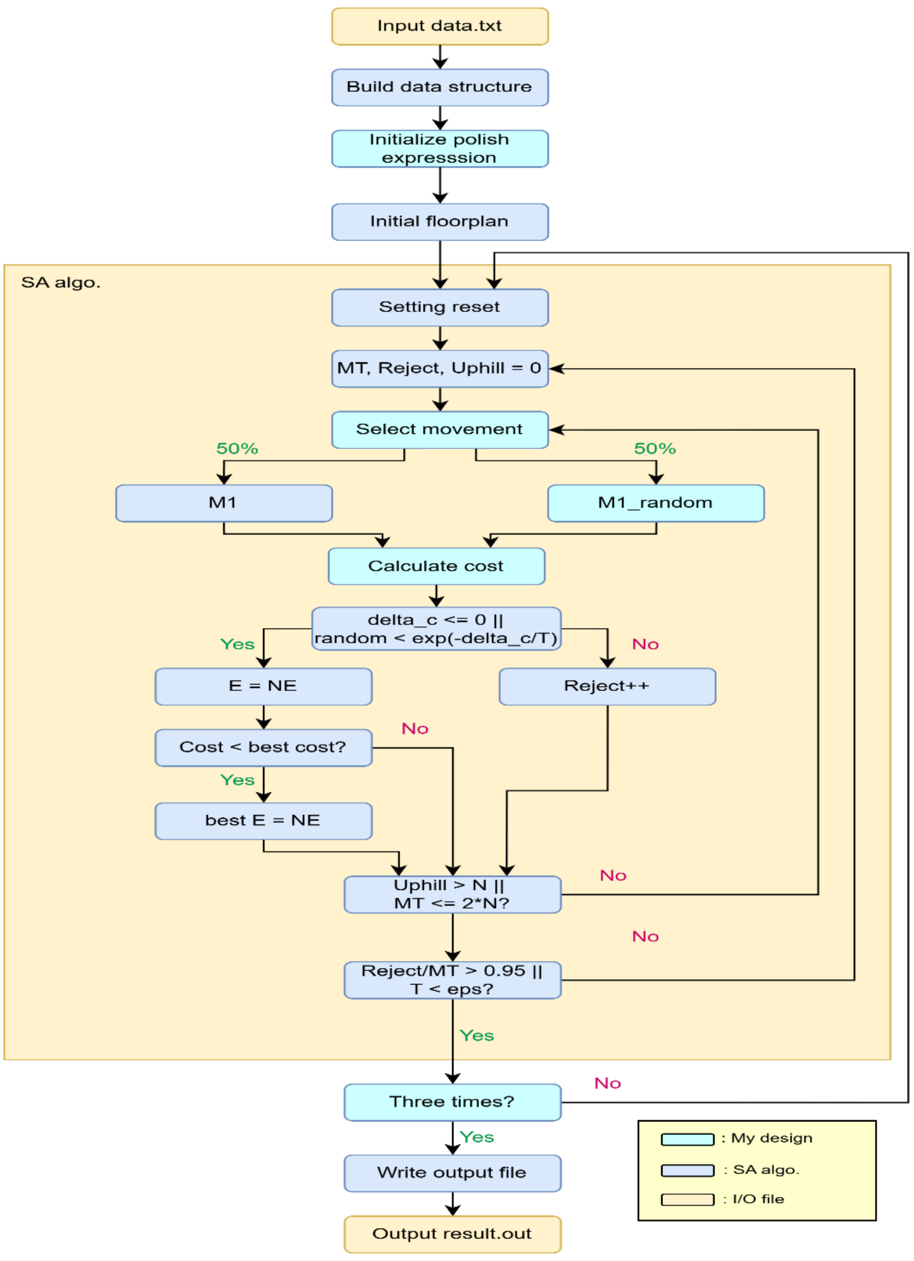
[Floor plan result]

Dead space ratio	0.15	0.1
Public1		
Public2		
Public3		

(3) Smallest dead space ratio within 10 minutes to produce legal result in 10 minutes

Testcase	public1	public2	public3
Dead space ratio	0.08	0.08	0.08
Wirelength	220995	450288	611274

(4) Algorithm Implementation



My design is similar to **Simulated_Annealing_floorplanning (SA) algorithm**[1] but with some changes. I modified some parts of the algorithm to speed up the execution time and make the final wirelength become lower.

[1] D. F. Wong and C. L. Liu, "A New Algorithm for Floorplan Design," *23rd ACM/IEEE Design Automation Conference*, Las Vegas, NV, USA, 1986, pp. 101-107, doi: 10.1109/DAC.1986.1586075.

Modification to minimize the wirelength:

The final wirelength is heavily dependent on the initial floorplan. Since there is dead space ratio constraint, it can be very hard to fit the block inside the floorplan region if the initial floorplan didn't put all the block inside the limited height and width in the first place. Thus, I try three different ways to initial my floorplan. And I use the last one as my final initialization method. In which the detail will be elaborate in the **floorplan initialization** section.

(1) Original floorplan initialization

I follow the original floorplan initialization method proposed from the paper. In which is 12V3V4....nV.

(2) Based on height & width

In which I sort the blocks based on their height and width and place the block from bigger height to lower height next to each other. This method successfully places public1 and public3 test case inside the region in the floorplan initialization.

(3) Based on height & width and try fit in blocks inside the space

Based on method (2) but fill in the blank hole that method (2) can't fill.

Modification to speed up the execution time:

(1) All pass by reference

At first all my functions that include the polish expression is pass by value. Not until I record the time that I find out how much time I've spent on only calculating the cost. Just a quick change with pass by value to pass by reference that the execution speed becomes faster.

(2) Parallel wirelength calculation

Since, I calculate the minimum shape based on the method proposed by Stockmeyer which is hard to parallelize. But I've noticed that the calculation of wirelength can be calculated separately. Thus, I use parallel wirelength calculations to speed up the execution time.

As the results, these two methods all successfully speed up the execution time but all pass by reference save more time compared to parallel wirelength calculations.

\

(5) Floorplan initialization

The initial floorplan initialization can deeply affect the validation of the final results. I implemented three methods in which to make my solution quality better and valid.

(1) Original floorplan initialization

I follow the original floorplan initialization method proposed from the paper. In which is 12V3V4....nV. This method is not going to fit in the region constraint and too far away from the optimal solution, this means that it will take longer to try to minimize the dead space to fit inside the region and minimize the wirelength, which is not good enough if we need it to be terminated within 10 minutes.

(2) Based on height & width

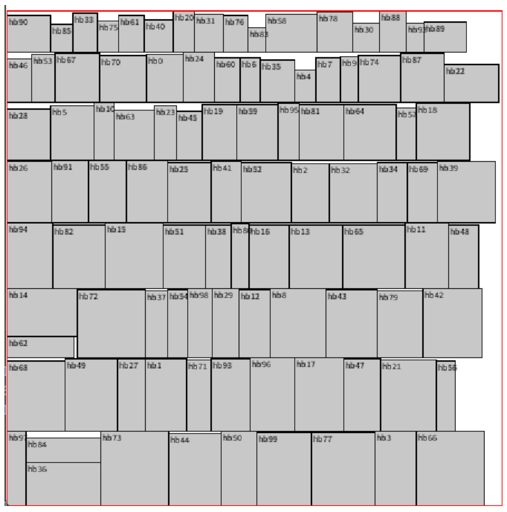
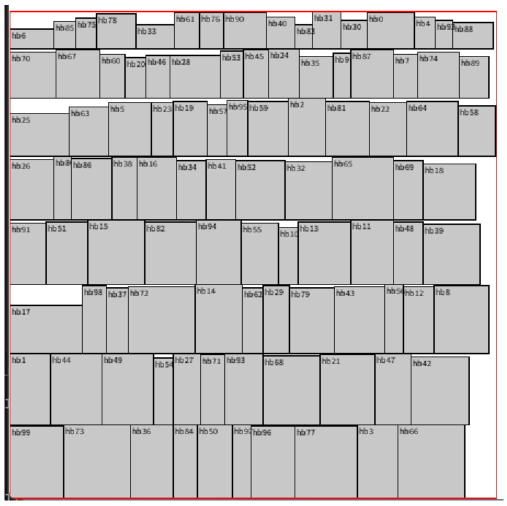
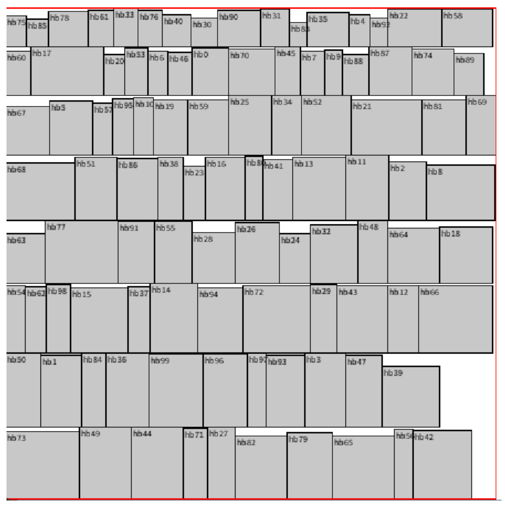
In which I sort the blocks based on their height and width and place the block from bigger height to lower height next to each other. And if the width is not enough for this row, I will change the row and place the new block on top of the previously placed blocks. And continue. This method successfully places public1 and public3 in the required region at the first place but can't work with public2 since it contains bigger blocks which can leave large holes without filling new block inside it.

(3) Based on height & width and try fit in blocks inside the space

Based on method (2) but fill in the blank hole that method (2) can't fill. In which I fill the hole the same way I put block in method (2). The problem with method (2) is that it ignores the blank that can be created during the placement. To simply put, I try to fill the blank above the placed block since there can be big blocks.

Method (3) successfully placed three public test cases in the initialization and can produced the lowest wirelength.

(6) Trick to speed up my program or enhance solution quality

Method	Floorplan	Wirelength
[Original]		233614
[Trick 1]		232096
[Trick 2] **[Final result]**		**211004**

At first, I follow the original SA algorithm's method and implement three movements into my algorithm.

[Original] Use M1/M2/M3 movements to modify the polish expression

I soon find out that M2/M3 can be destructive towards the floorplan due to the drastic changes to the polish expression. Thus, I try to only use original M1 to change the polish expression. Which is my first trick.

[Trick 1] Only use M1 movement to modify the polish expression

As the above result shows, we can see that the M1 movement is not quite enough to efficiently lower the wirelength. The reason might be the constraint inside the M1 movement. Original M1 movement only allow adjacent operands to swap. In which can be too insignificant and slow. To enhance the solution quality, I try doing random swaps between operands in which is random M1. Which is my second trick.

[Trick 2] Use M1 & random M1 movement to modify the polish expression

As the above result shows, we can see that after the adjustment, the wirelength becomes lower.

(7) Learned & encountered problems

Learned:

The SA algorithm is very heuristic and highly dependent on the initial floorplan and luck. Different random seeds can drive to very different outcomes. When dealing with this kind of algorithm, it's better to try multiple different settings and run multiple times. If the floorplan initialization is good, then the validation of the floorplan(means it can be placed inside the region) is certainly. The only problem is the minimization of the wirelength. In which it might need a little luck to meet the optimal point.

Encounter problems:

The process of thinking a new floorplan initialization method is very hard. I've noticed that if the floorplan isn't valid in the first place, it is very hard for the algorithm to find a valid solution in the process. So, it's better to find a valid solution in the first place and try to lower the wirelength in the process. Not only that, the process of building the cost function also took me a long time. This algorithm has a lot of operations that I can design on my own in which can be tough and fun at the same time.