VLSI Physical Design Automation – HW3

Name: 小山翼 (Koyama Tsubasa)
Student ID: 110062526

Part Ⅰ : How to compile and execute my program .

- To compile the program , go to directory **“ HW3/src/ ”** and execute the following command , then the executable file named **“ hw3 ”** will be generated in directory **“ HW3/bin/ ”** .

Command : $ make

- To execute the program , it can be executed with command 1 or 2 in directory **“ HW3/src/ ”** or **“ HW3/bin/ ”** respectively .

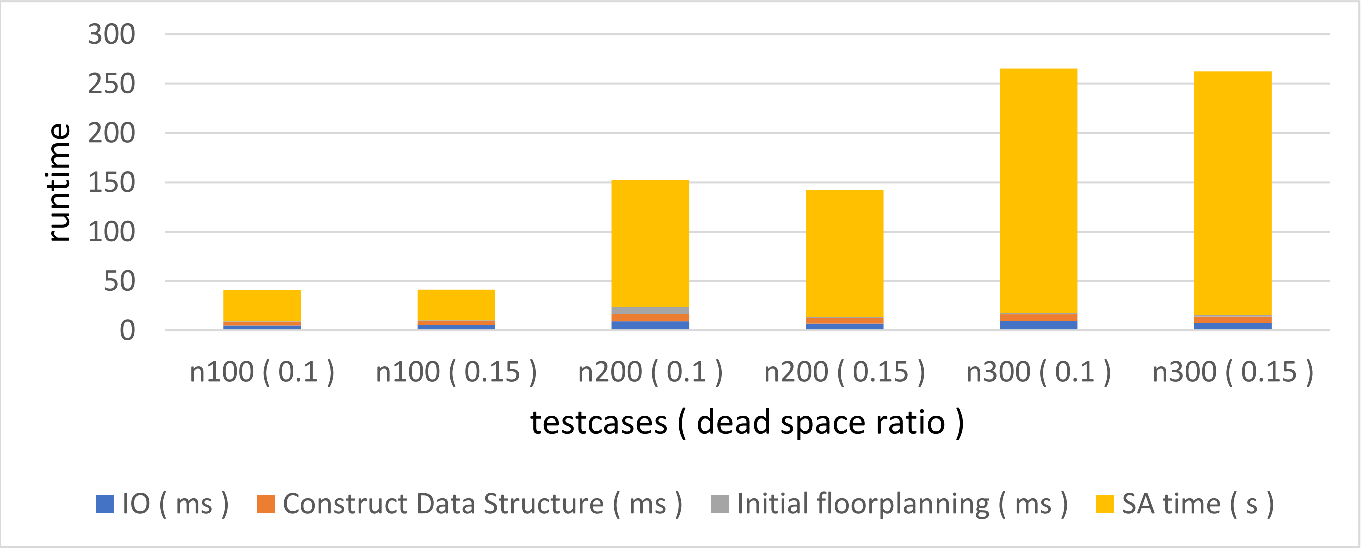
Command 1 : $ ../bin/hw3 < hardblocks > < nets > < pl > < floorplan > < dead space ratio >
Command 2 : $ ./hw3 < hardblocks > < nets > < pl > < floorplan > < dead space ratio >

e.g. :

```
$ ../bin/hw3 ../testcases/n100.hardblocks ../testcases/n100.nets ../testcases/n100.pl ../output/n100.floorplan 0.1
```

Part Ⅱ : Statistics

Dead space ratio	0.1			0.15		
testcases	**n100**	**n200**	**n300**	**n100**	**n200**	**n300**
Wirelength	217102	399081	573001	220389	389061	556168
IO (s)	0.00514	0.00914	0.00939	0.00551	0.00692	0.00766
Data Structure (s)	0.00374	0.00762	0.0068	0.00411	0.00573	0.00626
Initial Floorplanning (s)	0.00041	0.00068	0.00154	0.00073	0.00076	0.00144
SA Time (s)	31.5976	128.481	247.4	30.9585	128.643	247.103



Part Ⅲ : How small the dead space could be in my program .

testcases	n100	n200	n300
Smallest dead space ratio	0.09	0.097	0.1

Part Ⅳ : Algorithm

pseudo code

Input : files of hardblocks , nets , pl and dead space ratio
Output : floorplan file

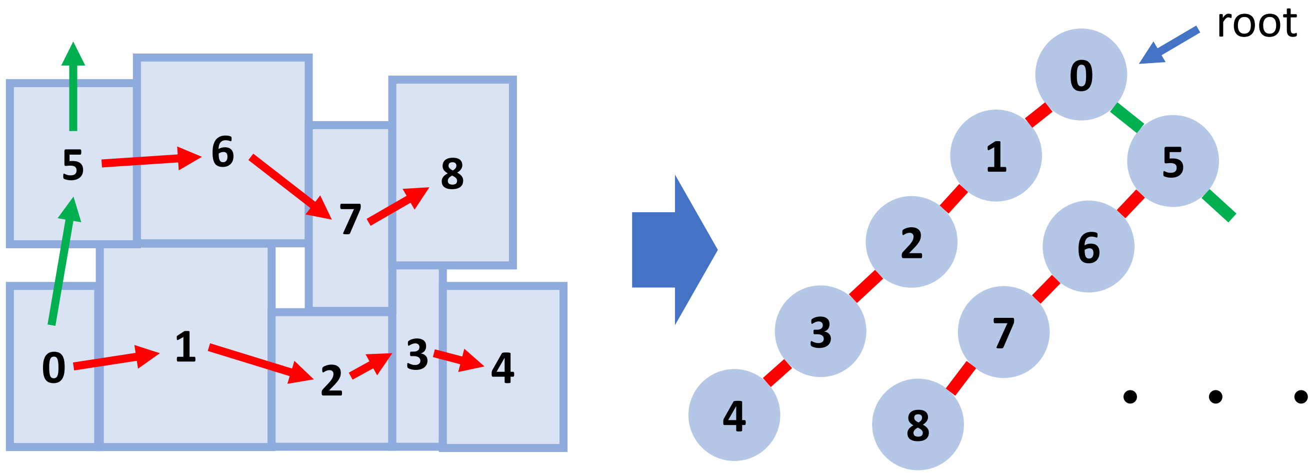
```
begin
read files
do initial floorplanning and construct B* tree
set initial temperature to 100000 , cooling rate to 0.85
calculate how many times to run in a temperature
 ( hardblock number*10 , denoted as SA_limit )
repeat
    for n=1 to SA_limit do
        tree perturbation
        calculate each blocks’ coordinate and update contour
        calculate half-parameter wirelength ( HPWL )
        calculate current cost
        if current cost < best cost then
            store current result and update best cost to current cost
    temperature = cooling*temperature
    if width and height in current contour does not change then
        if the random number from 0 to 99 < probability then
            enforce B* tree to perturb once and store the result
until temperature < 0.01
end
```

- **Initial floorplanning and construct B* tree**

In initial floorplanning , I place the hardblocks by tracing the nets since I want to minimize the initial wirelength . Moreover , I rotate some hardblocks which width is bigger than its height to guarantee that every hardblocks are vertical . I also make the constraint that there will be no hardblocks exceeding the outline of X in the initial floorplanning .The following is my placing algorithm .

1. place the hardblock as the root at (0,0)
2. place the next hardblock right adjacent and compact to left and down
3. repeat (2) if current width + hardblock's width < outline of X
4. place the next hardblock above and compact to left and down
5. do (2) ～ (4) until finish placing all the hardblocks

My B* tree implementation is a little bit differ from which implemented in the class . The following is my implementation , the hardblock numbers are the placing order .



- **Tree perturbation**

Since all of the hardblocks are vertical after the initialization , I set a low probability of rotation in order to decrease the probability exceeding the limited width . In my program , I only use the rotation and swap operations with probability of 30% and 70% respectively .

- **Calculate cost**

```
Cost  =  0.5*floorplanning area  +  0.5*wirelength
If ( width of contour > outline of X ) then
        Cost  +=  ( width of contour – outline of X )*500
If ( height of contour > outline of Y ) then
        Cost  +=  ( height of contour – outline of Y )*500
```

→ **penalty**

- **Escape from the local minimum**

In the original Simulated Annealing , there will be the chance to escape from local minimum many times in every temperature . But in my program , I will only check whether the width and height of contour updated or not after updating the temperature , if yes then there will be some probability to perturb the B* tree forcibly to hope that it can escape from the local minimum . The reason why I change this part is because I can not get the good result by using the original implementation . My probability is set to temperature/1000 .

- **Conclusion of Algorithm**

I reference the basic of SA from [1] and do some modification to it .

Part Ⅴ: The tricks I do to speed up my program or to enhance my solution quality .

The tricks I do are the following :

1. I built the initial floorplanning depend on the nets in order to minimize the initial wirelength .
2. There will be no hardblocks exceeding the outline of X in my initial floorplan.
3. Since my B*tree is easy to implement than the original one , it can be easily constructed or perturbated .
4. I add some penalties to the cost function to make the aspect ratio approaches 1 more easily .

Part Ⅵ: Compare with the top 5 students' results last year .

	Wirelength			**Runtime (s)**		
Ranks	**n100**	**n200**	**n300**	**n100**	**n200**	**n300**
1	200956	372143	516906	**24.63**	**47.29**	**65.81**
2	198593	368731	535257	200.25	308.06	226.42
3	**194369**	**354107**	**491069**	385.75	709.61	926.55
4	204001	367298	499733	330.42	576.15	793.26
5	208575	378187	567794	26.72	120.73	247.22
Mine	**220389**	**389061**	**556168**	**33.78**	**134.77**	**254.97**

I think my wirelength of each case are not so well since my program didn't find the better local minimum , but the runtime is almost as same as others or even better .

Part Ⅶ: What have you learned from this homework ? What problems have you encountered in this homework ?

I have learned the difficulties of implementing SA algorithm with good results . In the beginning , I tried to implement the Fast SA algorithm but while I was running the program , I found that it is too difficult to get a good results by this algorithm , so I decided to use the original SA with some modifications like how to do the initial floorplanning . I have met some problems while swapping the B* tree since I use the pointer to connect hardblocks , I forgot that I have to consider some special conditions like the hardblock which chosen to be swapped is the parent of another one .

Part Ⅷ: References

[1] Modern Floorplanning Based on B*-Tree and Fast Simulated Annealing , Tung-Chieh Chen , Yao-Wen Chang , IEEE TCAD , VOL.25 , NO.4 , APRIL 2006