VLSI Physical Design Automation – HW2

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

Part Ⅰ :

How to compile and execute my program .

- To compile the program , first go to directory " HW2/src/ " and execute the following command , then the executable file named " hw2 " will be generated in directory " HW2/bin/ " .

Command : $ make

- To execute the program , it can be executed with command 1 or 2 in directory " HW2/src/ " or " HW2/bin/ " respectively .

Command 1 : $ ../bin/hw2 < nets > < cells > < output >
Command 2 : $ ./hw2 < nets > < cells > < output >

e.g. : $ ../bin/hw2 ../testcases/p2-1.nets ../testcases/p2-1.cells ../output/p2-1.out

Part Ⅱ :

	p2-1	p2-2	p2-3	p2-4	p2-5
I/O time (s)	0.005	0.038	0.616	1.086	2.607
Computation time (s)	0.004	0.093	69.315	156.101	115.602
Runtime (s)	0.02	0.15	70.13	157.54	118.60
Initial cut size	41	376	18361	54135	141536
Final cut size	29	299	15342	50655	140266

Part Ⅲ :

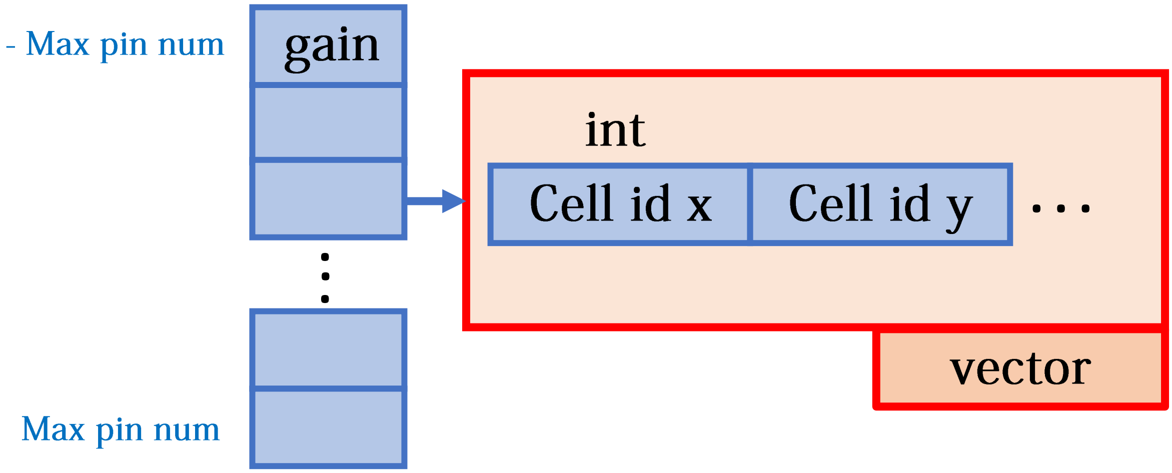
The difference between mine and FM Algorithm described in class .

- Besides the initial partition and the condition to terminate, there is no big difference between my algorithm and FM Algorithm described in class . I will explain the method of my initial partition and terminate condition in later section .

Did you implement the bucket list data structure ?

- Yes , I implement bucket list with the following code to store two gain lists , partition set A and B .

```
Code :       map < int , vector < int > > Blist_A , Blist_B ;
```



How did you find the maximum partial sum and restore the result ?

- I use integer sum to store the partial sum and max_sum to store the maximum partial sum . In the end of every iteration in FM algorithm , I will compare these two variables and check whether to update or not .

My checking algorithm

```
if      sum >= max_sum :
        if      sum > max_sum :
                restore the information
        else if       sum = max_sum :
                if weight difference is smaller , restore the information
```

- Since I use the following code to store the connection between cells and partition sets , I copy the same structure to store the information in maximum partial sum .

```
Code :       map < int , char > partition_set ;
        //   int store cell id , char store  'A'  /  'B'
```

- I use variable final_cutsize to store the cut size in maximum partial sum .

What else did to you do to enhance your solution quality or to speed up your program ?

- **In the initial partition phase :**

1. I first split the cells by nets , moving the cells in the same net until the weight in A set over half of the total weight , others move into B set . Calculate the initial gains .

2. Since I want the initial cut size more smaller , I move the cells which has positive gain from A set to B set under the balance constraint . Calculate cell gains in this condition , then do the same thing from B to A again .

These steps will make the initial cut size small , but there will be a problem that you must move the negative gain cells to reduce the cut size more , otherwise the final cut size will only decrease a little bit .

- **The condition to terminate :**

I set my program to terminate if the max cell gains in A set and B set are both less or equal to 0 , since it is too time consuming to move the cells of negative gain in my program . But the combination between this condition and the partition phase I mention above makes my final cut size undesirable .

Part Ⅳ :

Compare with the top 5 students' results from last year .

	Cut size					Runtime (s)				
Ranks	p2-1	p2-2	p2-3	p2-4	p2-5	p2-1	p2-2	p2-3	p2-4	p2-5
1	6	191	4441	43326	122101	0.01	0.07	3.05	5.01	42.06
2	6	161	1065	43748	125059	0.01	0.1	3.11	9.84	18.77
3	6	358	2186	45430	122873	0.04	0.78	21.21	115.38	59.78
4	5	302	1988	46064	124862	0.03	0.17	7.04	6.93	8.22
5	6	411	779	46356	125151	0.01	0.16	5.49	12.31	13.57
Mine	29	299	15342	50655	140266	0.02	0.15	70.13	157.54	118.60

- Comparing the runtime , there is no big difference in the first two testcases , but if it becomes more complicated like p2-3,4,5 , there will be a large gap between mine and their programs .
- Only result of testcase 2 can be compared , other results are not desirable in my program , especially testcase 3 .

Part V : (Conclusion)

- Since in the beginning , I did not know there are some useful structure in CTL like map , vector , so I encountered some problems while changing the program using linked list to CTL structure .

- I found that my program is not so efficient to run the complicated testcases in the limited time , so I am planning to get some codes from other students and see how to write code more efficiently after the deadline of this work .

- Since I do not know how to implement parallelization , I think I can learn this part from others in the future , too .