CS 313000 Introduction to Computer-Aided Design of Integrated Circuits

Programming Assignment 2 (Deadline: 2018/11/26 23:59)

After technology mapping, we have to implement our design on the die physically considering area, interconnect wirelength, power, etc. This is called physical design. Partitioning is an important step in physical design before placing logic elements on the die.

Since a design may contain over a hundred of thousands of standard cells totally, the problem size is usually too large to be handled efficiently by a placement tool. So, a "shrinking procedure" called partition can be applied. Multiple cells can be partitioned together to form a group. The number of groups formed is much smaller than the original number of cells. Then we may first place the groups and refine the placement thereafter.

In this assignment, you have to implement a partitioning program. Given a set of cells with its area, a netlist that describes the connection between cells of this circuit, and the maximum area constraint of a group, your program should compute a group of cells that minimize the cost function described below.

$$C(i) = (\ (\#\ \text{groups that net i spans}) - 1\ )^2$$

The cost is the sum of C(i) of each net in the netlist. That is, if net j spans 1 group (all cells belong to this net are in the same group), $C(j) = (1-1)^2 = 0^2 = 0$. If net k spans 5 groups (the cells belong to this net are in 5 groups), $C(k) = (5-1)^2 = 16$. If a circuit contains nets j and k only, the total cost will be 0+16=16.

Basic version (two-way partition)

Input Format:

Three input files, **.aux**, **.area**, and **.net**, are used to describe the area constraint of one group and filenames, the area of each cell, and the netlist of the circuit, respectively.

.aux: The first two lines indicate the filenames of the .area and .net files, respectively. The third line indicates the area constraint of one group as a positive integer.

.area: The first line indicates the total number of cells. If there are ***n*** cells, each of the following ***n*** lines indicates the unique cell number of a cell, ranging from ***0*** to ***n-1***, followed by its area in positive integer.

.net: The first line indicates the total number of nets. If there are ***m*** nets, the following ***2m*** lines describe the netlist. The ***2i+2th*** line indicates the number of cells connected by net ***i*** and the ***2i+3th*** line lists the cells connected by the net ***i*** by their unique cell numbers where ***i*** is from ***0*** to ***m-1***.

sample.aux

```
Sample.area     //.area file name
Sample.net      //.net file name
71              //maximum area constraint per group, in the basic version,
                //this will be the (total area * 0.55), e.g. 128 * 0.55 = 71
```

sample.area

```
10              //total number of cells, n =10
0 8             //cell 0 with area 8
1 15            //cell 1 with area 15, etc.
2 6
3 22
4 11
5 8
6 13
7 18
8 20
9 7
```

sample.net

```
3               //total number of nets, m = 3
5               //net 0 contains 5 cells
0 3 9 6 8       //cells 0, 3, 9, 6, 8 are connected by net 0
4               //net 1 contains 4 cells, etc.
1 2 3 7
6
4 5 6 7 8 9
```

Ouput Format:

The first line gives the cost computed using the given formula. The second line gives the number of groups, ***k***. The ***i+3th*** line gives the group index, ranging from ***0*** to ***k-1***, that cell ***i*** belongs to.

sample.aux.out

```
3          //total cost = 1+1+1
2          //total number of groups, k = 2 (fixed)
1          //cell 0 belongs to group 1
0          //cell 1 belongs to group 0
0          //cell 2 belongs to group 0
0          //cell 3 belongs to group 0
0          //cell 4 belongs to group 0, etc.
0
1
1
1
1
```

Advanced version (k-way partition)

Input Format:

sample.aux

```
Sample.area    //.area file name
Sample.net     //.net file name
50             //maximum area constraint per group
```

sample.area

```
10             //total number of cells, n =10
0 8            //cell 0 with area 8
1 15           //cell 1 with area 15, etc.
2 6
3 22
4 11
5 8
6 13
7 18
8 20
9 7
```

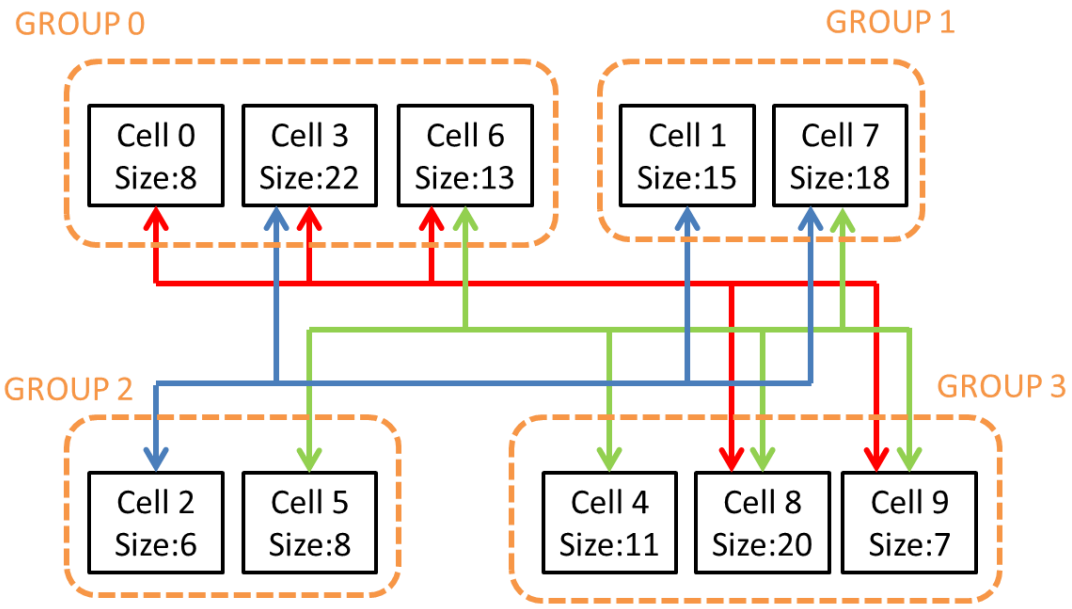
sample.net

```
3              //total number of nets, m = 3
5              //net 0 contains 5 cells
0 3 9 6 8      //cells 0, 3, 9, 6, 8 are connected by net 0
4              //net 1 contains 4 cells, etc.
1 2 3 7
6
4 5 6 7 8 9
```

Output Format:

sample.aux.out

```
14      //total cost = 1+4+9
4       //total number of groups, k = 4
0       //cell 0 belongs to group 0
1       //cell 1 belongs to group 1
2       //cell 2 belongs to group 2
0       //cell 3 belongs to group 0
3       //cell 4 belongs to group 3, etc.
2
0
1
3
3
```



The red net spans 2 groups. ➔ cost = $(2-1)^2 = 1$
The blue net spans 3 groups. ➔ cost = $(3-1)^2 = 4$
The green net spans 4 groups. ➔ cost = $(4-1)^2 = 9$

Compilation & Execution Issues:

Your program should support the following way of compilation **ONLY**. In other words, any other compiling methods would not be allowed.

account@workstation:~$ make

Your program should support the following way of execution.

account@workstation:~$ <your program> <input file name>
e.g. u9962999@NTHU:~$ ./a.out sample.aux

That is, TAs will add the input file name after the executable file name as an input. If your program is not able to support that, point deduction will be considered.

Grading Policy:

You have to write this program in C or C++. TAs will verify your program on ic18 on NTHUCAD workstation with 6 released test cases and 2 hidden cases. As the verification program will be released by TAs, you can verify your program first by yourselves. You should also write a MAKEFILE to compile your code or we may not be able to run it successfully. In addition, a REPORT is also needed, in which you should introduce the algorithm you used and how to execute your program. The page limit of the report is 10. The output file of your program should be named as *name**.out*** where *name* is the input file name, e.g., if the input file name is ***case1.aux***, the ***case1.aux.out*** is the name of the output file.

The files you need to hand in:

1. Source codes in C/C++.
2. A makefile.
3. A report.

You should use iLMS system to submit your homework. If you have difficulty in submission, please contact TAs.

Your program should be scalable for large cases. This assignment will be ranked and scored according to the quality of the experimental results and the runtime. Specifically, the percentage is as follows:

Accuracy	48%
Runtime	32%
Report	20%
Bonus	20%

Basic version:

Result	Accuracy	Runtime
Ranking top 35%	6	4
Ranking between 35%-70%	4	3
Ranking lower than 70%	2	2
Functionally not equivalent	1	1
Segmentation fault or TLE	0	0

Advanced version:

Result	Accuracy	Runtime
Ranking top 35%	1.5	1
Ranking between 35%-70%	1	0.8
Ranking lower than 70%	0.5	0.5
Functionally not equivalent	0	0
Segmentation fault or TLE	0	0

Note that if the runtime of your program for a case is longer than 2 hours, it will be treated as Failed for TLE (time limit exceeded).