CS313000 Introduction to Computer-Aided Design of Integrated Circuits

Programming Assignment 2

Deadline: 2022/05/04 23:59

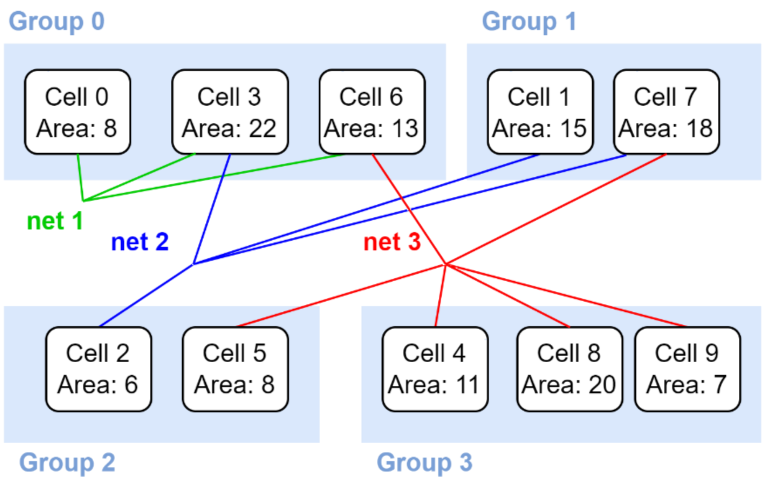
1. Introduction

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 an algorithm for **2-way (basic)** and **k-way (advanced)** partitioning. Given a set of cells with its area, a netlist that describes the connection between cells, and the maximum area constraint per group, your program should group the cells and minimize the cost function described below.

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

The total cost is the ***sum of C(i)*** of each net in the netlist. Consider a result shown in the figure below where we have 3 nets and 4 groups. Net 1 spans only one group (all cells belong to this net are in the same group), so C(1) = $(1-1)^2$ = 0. Net 2 spans 3 groups (group 0, group 1 and group 2), so C(2) = $(3-1)^2$ = 4. Net 3 spans 4 groups (group 0, group 1, group 2 and group 3), so C(3) = $(4-1)^2$ = 9. Therefore, the total cost of this case = C(1) + C(2) + C(3) = 13.



2. Input file format (*.in*)

The input file describes the constraint and the circuit. The first line indicates the area constraint per group in a positive integer. In the “basic” test cases (for 2-way partitioning), the area constraint per group = total area * 0.55. In the “advanced” test cases (for k-way partitioning), the area constraint per group < total area * 0.5, so that a k-way partitioning should be used in such test cases. Starting from the second line, two sections marked by the keywords, **.cell** and **.net**, describe the area of each cell and the netlist of the circuit, respectively.

Following the line where the keyword **.cell** is, 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 a positive integer.

Following the line where the keyword **.net** is, 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 cells number where i is from **0** to **m-1**.

case00.in

```
71         //maximum area constraint per group
.cell
10          //total number of cells, n = 10
0 8         //cell 0 with area 8
1 15        //cell 1 with area 15
2 6
3 22
4 11
5 8
6 13
7 18
8 20
9 7
.net
3            //total number of nets, m = 3
5            //net 0 contains 5 cells
0 3 9 6 8    //net 0 connects to cell 0, 3, 9, 6 and 8
4            //net 1 contains 4 cells
1 2 3 7     //net 1 connects to cell 1, 2, 3 and 7
6
4 5 6 7 8 9
```

3. Output file format (*.out*)

The output file should be the partitioning result. The first line gives the cost computed using the given formula. The second line gives the number of groups, $\mathbf{k}$. The **$\mathbf{i}$+3th** line gives the group index, ranging from **0** to **$\mathbf{k}$-1** that cell $\mathbf{i}$ belongs to.

Please note that your group index should start from 0.

case00.out (2-way)

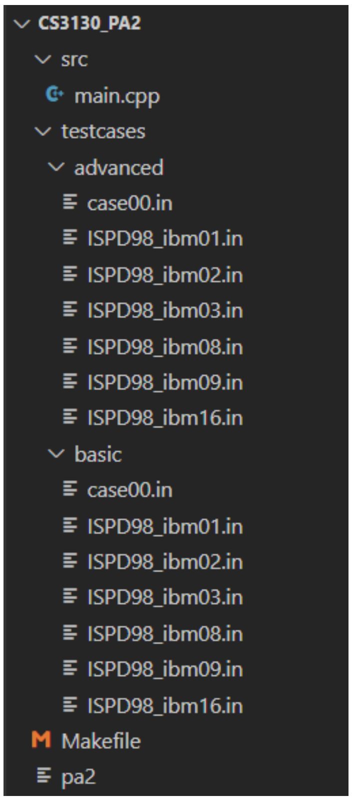
```
3     //total cost = 3
2     //total number of groups = 2
1     //cell 0 belongs to group 1
0     //cell 1 belongs to group 0
0     //cell 2 belongs to group 0
0
0
0
1
1
1
1
```

case00.out (k-way)

```
14     //total cost = 14
4      //total number of groups = 4
0      //cell 0 belongs to group 0
1      //cell 1 belongs to group 1
2      //cell 2 belongs to group 2
0
3
2
0
1
3
3
```

4. Implementation

A sample directory is provided with a file name, `CS3130_PA2.tar.gz`. Please decompress it using the command `$ tar -zxvf CS3130_PA2.tar.gz`, then a directory named `CS3130_PA2/` with a file structure like the following figure will be produced.



```
CS3130_PA2
  src
    main.cpp
  testcases
    advanced
      case00.in
      ISPD98_ibm01.in
      ISPD98_ibm02.in
      ISPD98_ibm03.in
      ISPD98_ibm08.in
      ISPD98_ibm09.in
      ISPD98_ibm16.in
    basic
      case00.in
      ISPD98_ibm01.in
      ISPD98_ibm02.in
      ISPD98_ibm03.in
      ISPD98_ibm08.in
      ISPD98_ibm09.in
      ISPD98_ibm16.in
  Makefile
  pa2
```

You have to write this program in C/C++ and a Makefile for compiling the program. TAs will compile your source code and run all the test cases (6 released cases and 2 hidden cases) by the following commands.

```
$ cd CS3130_PA2
$ make
$ ./pa2 <.in file> <.out file>
```

(e.g., `$ ./pa2 testcases/basic/case00.in case00.out`)

So, make sure that your program follows the format and accepts the corresponding arguments. Note that the <.in file> and <.out file> here are the path names of the file, so there is no need to do any processing for these arguments. A script is provided below for your reference.

```
int main(int argc, char* argv[]){
    ifstream in_file;
    in_file.open(argv[1]);    //open <.in file>

    ofstream out_file;
    out_file.open(argv[2]);    //open <.out file>
```

5. Required items

Two files should be summited to NTHU eeclass,

- `PA2_{StudentID}.tar.gz`
- `PA2_{StudentID}.pdf`

Please compress `CS3130_PA2/` into one tar file with the name `PA2_{StudentID}.tar.gz` by running the command,
`$ tar -zcvf PA2_{StudentID}.tar.gz CS3130_PA2/`. It must contain the **SOURCE CODES in C/C++** under `src/` and a **MAKEFILE**.

A **REPORT** with a file name `PA2_{StudentID}.pdf` is also needed, in which you should contain (1) your name and student ID, (2) how to execute your program, (3) introduction of the data structure and the algorithm you used and (4) other details of your implementation. The page limit of the report is 10.

6. Grading policies

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 – basic	48%	(Ranked by the cost)
Runtime – basic	32%	
Accuracy – advanced	12%	(Ranked by the cost)
Runtime – advanced	8%	
Report	20%	

For EACH test case, the accuracy and the runtime will be graded as follows,

	Basic		Advanced	
Result	Accuracy	Runtime	Accuracy	Runtime
Ranks top 35%	48	32	12	8
Ranks between 35%-70%	32	24	8	6
Ranks lower 70%	16	16	4	4
Illegal solution	8	8	0	0
Segmentation fault or **TLE**	0	0	0	0

TLE (time limit exceeded): runtime longer than 2 hours.

Other Notes

- Your program should decide to run 2-way or to run k-way partitioning by the given area constraint per group and the total area, **not by detecting the "advanced" keyword in the filename**.
- If you have any question, please post it on the Discussion board of NTHU eeclass instead of using email since there may be someone else having the same question.
- Your program should be single-threaded. Parallel computation with multiple threads or processes is not allowed.