Lab3 – Multi-chip Partition and Placement

Deadline: 2022/12/2

Lab3 Introduction

In this programming assignment, you would get a chip design and cell information with a global placement result. You need to align all standard cells to feasible rows and legalize all standard cells without overlap. You also need to minimize the movement of cells between global placement and legalized placement.

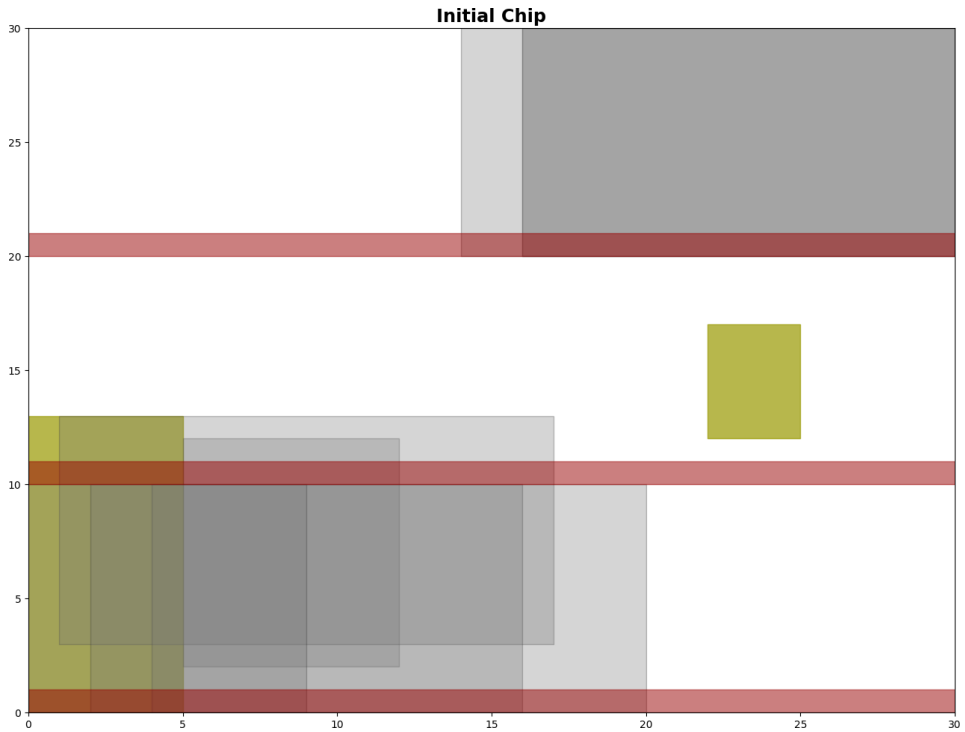
However, there are too many cells so placing them in one chip is not probable. Therefore, you would get two chips to place the cells, meaning you need to partition cells into two groups before Legalization.

Note:

- **Chip1, Chip2 have same spec (Width, Height, RowHeight, RowNumber, Terminal)**
- **Each cell can be placed on any chip, but must be placed on only one chip**
- **Cells must be aligned on the bottom of feasible rows**
- **Cells cannot overlap with each other, or overlap with Terminals**
- **Terminals cannot be moved**
- **Legal solution must be generated**

Input

Example (case1.txt)



```
DieSize 30 30   //Chip_Width, Chip_Height
DieRows 10 3  //Row_Height, Row_Number
Terminal 2      //Terminal_Number
T0 0 0 5 13    //Terminal_Name,coordinate of lower-left corner, Width, Height
T1 22 12 3 5
NumCell 8    //Celll_Number
C0 5 2 7 10  //Cell_Name,coordinate of lower-left corner, Width, Height
C1 1 3 16 10
C2 0 0 16 10
C3 16 20 14 10
C4 16 20 14 10
C5 14 20 16 10
C6 4 0 16 10
C7 2 0 7 10
```

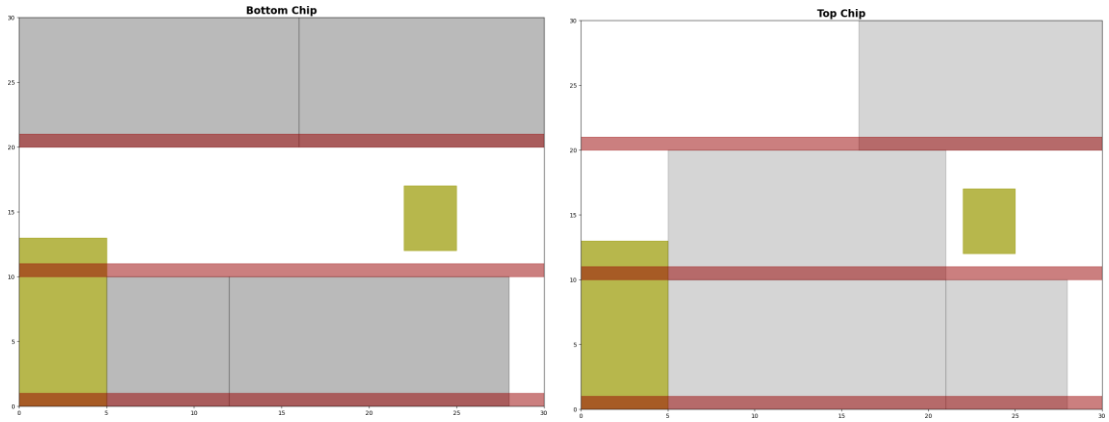
Explanation of the Input

1. The first line gives the chip size and the coordinate of the lower-left corner is (0,0).
2. The second line describes Row_Height & Row_Number
   - Row_Width = Chip_Width
   - Row_Height * Row_Number <= Chip_Height
   - Cell_Height <= Row_Height
   - The rows are stacked one after the other one from the bottom of Chip(Y=0) ex: The lower-left corner of the fifth row would be (0,4*Row_Height)
3. The third line describes Terminal_Number
4. If Terminal_Number != 0, following lines describe Terminal's information(Terminal_Name, X, Y, Width, Height)
5. Next line describes Cell_Number
6. Following lines describe Cell's information (Cell_Name, X, Y, Width, Height)

Note:

- All the values in the input file are integers.
- Terminal_Name would follow T0 -> T1 -> T2 ....
- Cell_Name would follow C0 -> C1 -> C2 ....
- Terminal may appear anywhere, but not beyond the outline of the chip
- Terminal_Number (0 ~ 100)
- Cell_Number (1 ~ 250000)
- When Cell_Number>1000, cells' initial position are roughly uniformly spread across the chip.

Output



Output Format (output.txt)

```
C0 21 0 0 //Cell_Name, coordinate of lower-left corner, Chip_Index
C1 5 10 0
C2 5 0 0
C3 16 20 0
C4 16 20 1
C5 0 20 1
C6 12 0 1
C7 5 0 1
```

Explanation of the Output

1. Describe every Cell's information(Cell_Name, X, Y, Chip_Index)

Note:

- Output all cells as input order C0 -> C1 -> C2 ....
- Chip_Index must be 0 or 1

Verifier:

A verifier will be provided to help you confirm the legality and show your final Cost. Your results will only be accepted if "Success" is displayed.
Command: ./verifier [INPUT] [OUTPUT] (-d)
(-d would show more detailed information.)

Executing Procedure

1. Compile (Make)
2. ./Lab3 [INPUT] [OUTPUT]
3. Search for [OUTPUT], if not found → break → enter 0 point
4. ./verifier [INPUT] [OUTPUT]
   if Success → take "Cost" as your score **(Smaller cost means better result)**
   else → get 0 point of that case

```
14:47 zu00895077@vda04 [~/PDA_TA/Lab3] >$ ./verifier lab3_testcase/lab3_case1.txt output.txt
----------------- Loading Input -------------------
Die Width : 30
Die Height : 30
Row_height : 10
Row_number : 3
Terminals Number: 2
Cells Number: 8
----------------- Loading Output  &  Legality Confirmation -------------------
Cells Number: 8
Chip0 Cells Number: 4
Chip1 Cells Number: 4
Chip0 Cells Area: 530
Chip1 Cells Area: 530
Chip0 Utilization: 0.588889
Chip1 Utilization: 0.588889
----------------- Initial_Overlap_Calculating -------------------
<Total_Overlap> 1112
----------------- Partition_Overlap_Calculating -------------------
<Total_Overlap> 414
----------------- Overlap_Remaining_Percent -------------------
Overlap_Remaining_Percent: 0.372302
----------------- Total_Offset_Calculating -------------------
Total_Offset: 59
----------------- Result -------------------
Success !!
Cost: 21.9658
```

Cost Function

To simplify the problem, we ignore the cost of crossing die (z axis) in this lab, we only consider the offset of the cells in Manhattan Distance on the 2D plane.

1. **Total_Offset:**

```
if placement is not legal
        exit
Total_Offset = 0
for every cells Cn:
        X1,Y1 = initial position of Cn //get from INPUT
        X2,Y2 = final position of Cn   //get from OUTPUT
        Offset += ( |X1-X2| + |Y1-Y2| )

return Total_Offset
```

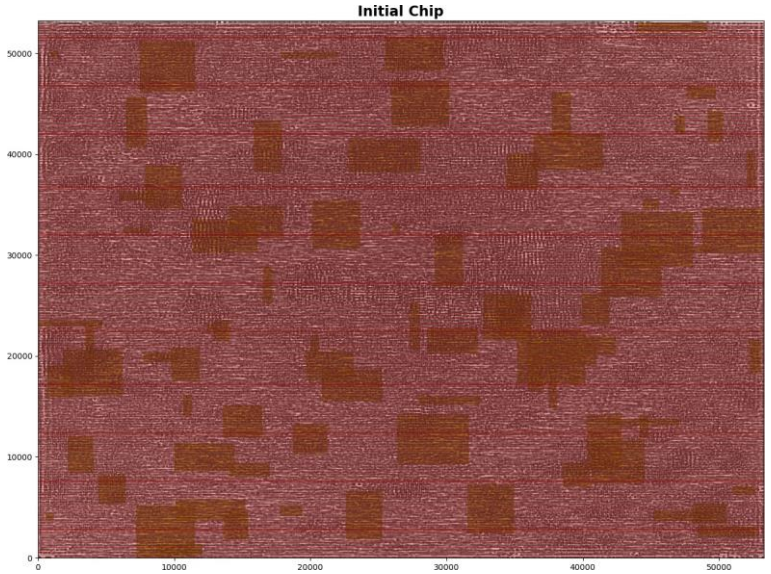
2. **Initial_Overlap:**
   Use initial position obtained from the input file to calculate the overlap area between all cells and accumulate them.

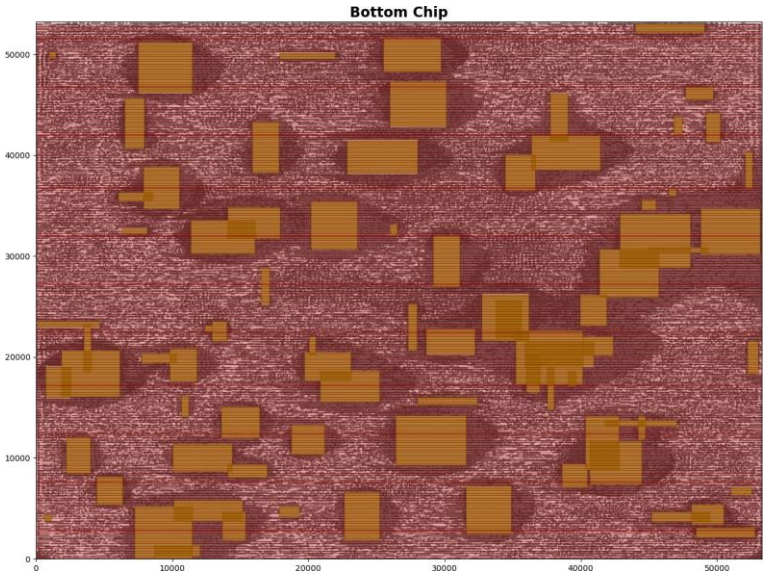
3. **Partition_Overlap:**
   Use initial position obtained from the input file to calculate the overlap area between all cells in same chip layer and accumulate them.

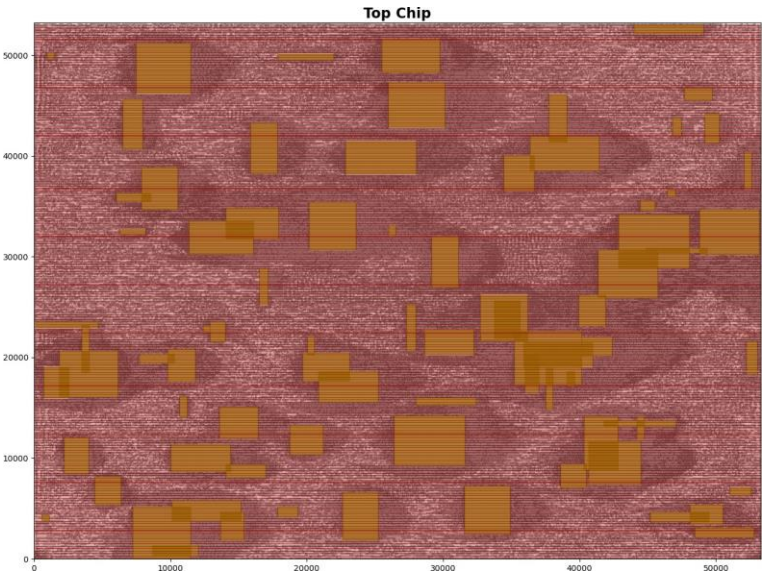
4. **Overlap_Remaining_Percent = Partition_Overlap / Initial_Overlap**

Cost = Overlap_Remaining_Percent * Total_Offset

Another Case Illustration:







Environment

1. Linux (Please make sure your code is available on our linux server(compile & execute). If it does not work, you will get zero points!!)
2. Makefile should be provided

Evaluation

1. You **MUST WRITE YOUR OWN CODE**. Copying codes may result in you **failing** this course.
2. For each case, the run time limit is set up to **5 minutes**. It will be regarded as "failed"if you use more than 5 minutes.
3. Naming rule.
    A. Name of the binary after "make" – Lab3
    B. Execution procedure: ./Lab3 [input] [output]
    C. If not following specified naming rule, you will receive zero mark
3. Late submissions will **NOT** be accepted.
4. Hidden cases will be evaluated.
5. The legality is guaranteed to have at least 75 points, the cost will be used to make you rank, the lower the cost, the higher the score.

Submission

<1>

Please put all required files in a folder, the name of the folder is your Student_ID.

```
Student_ID
    |__Source code (.cpp, .h)
    |__Makefile
```

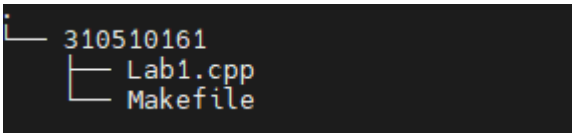
<2>

Use below command to compress the folder in the linux environment, and the compressed file name should as same as your Student_ID.

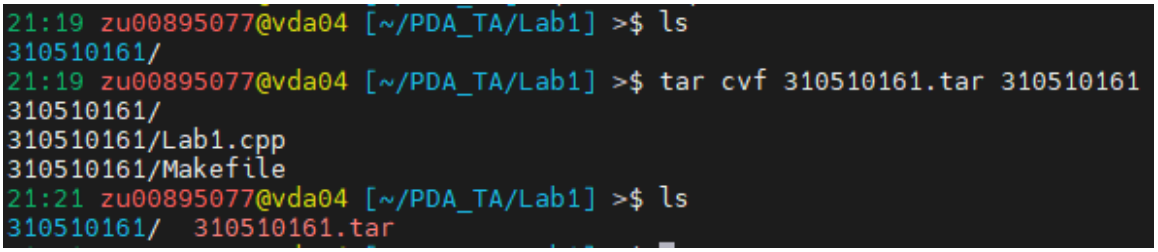
```
tar cvf Student_ID.tar Student_ID
```

Example

<1>



<2>



Upload Student_ID.tar to e3

(If your submission file is not .tar, you will get zero point!!)

Suggestion:

***** Using the following algorithms could help you on this lab.

(The earlier the stage, the greater the impact on the result)

1. (MUST DO) **Partition Algorithm:** Random、Kl、FM ……
2. (MUST DO) **Legalization Algorithm:** Tetris algorithm, Abacus .....
3. **Detailed Placement Algorithm:** Fast DP, Random, Simulated Annealing.....

* *Lab3_Supplementary.pdf* provides a flow, which takes a lot of work, but you will get good grades if you follow it.

* Pay attention to the time complexity of the algorithm in your program. In this lab, the number of cells may be as high as hundreds of thousands. If you use an algorithm >= O(n^2), it is likely to time out.