# Lab2 – Fixed Outline Floorplanning

# Deadline: 2022/10/31 23:59

## Lab2 Introduction

This programming assignment asks you to write a **fixed-outline floorplanner** that can handle hard macros. Given a set of rectangular macros and a set of nets, the floorplanner places all macros within a rectangular chip without any overlaps. We assume that the lower-left corner of this chip is the origin (0, 0), and no space (channel) is needed between two different macros. The objective is to minimize the area of the chip bounding box and the total net wirelength. The total wirelength W of a set N can be computed by

$$W = \sum_{n_i \in N} HPWL(n_i)$$

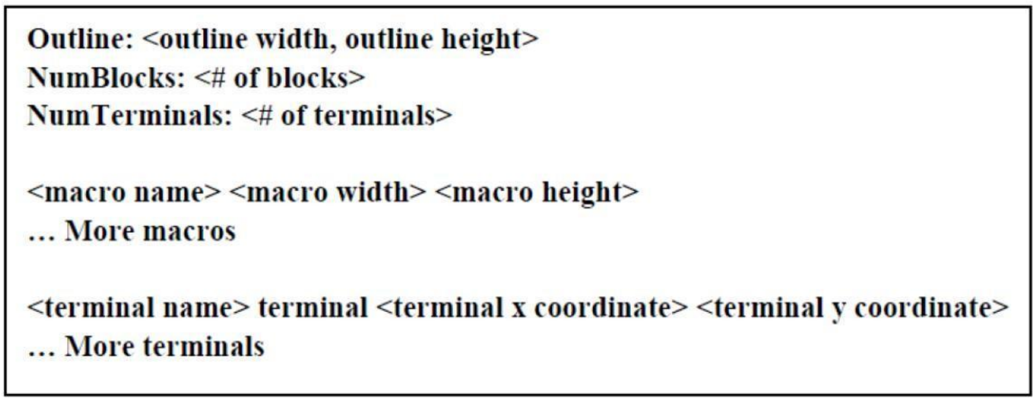
where $n_i$ denotes a net in $N$, and $HPWL(n_i)$ denotes the half-perimeter wire length of $n_i$.
The objective for this problem is to minimize

$$Cost = \alpha A + (1-\alpha)W \text{ (Note: Cost is integer.)}$$

where $A$ denotes the bounding-box area of the floorplan, and $\alpha$, $0 \leqq \alpha \leqq 1$, is a user defined ratio to balance the final area and wirelength. Note that a floorplan which cannot fit into the given outline is not accepted.

## Input

Each test case has two input files, *input.block* and *input.nets*. The first file (*input.block*) gives the **outline size**, the **number of blocks**, and the **number of terminals** defined in this file. Then the **block dimensions** are listed, followed by the terminal locations. The file format is as follows:



```
Outline: <outline width, outline height>
NumBlocks: <# of blocks>
NumTerminals: <# of terminals>

<macro name> <macro width> <macro height>
... More macros

<terminal name> terminal <terminal x coordinate> <terminal y coordinate>
... More terminals
```

Figure 1 Input file format ( *input.block* ).

The second file (input.net) gives the **number of nets** in the floorplan, followed by the **terminal information for each net**. The file format is as follows:

```
NumNets:<# of nets>
NetDegree:<# of blocks in this net>
<block name>
... More block names

...More "NetDegree" and "block name"
```

Figure 2 Input file format ( *input.nets* )

The user-defined ratio $\alpha$ is given through the command-line argument. It ranges between 0 and 1.

## Output

The output file (*output.rpt*) records the problem output. This report consists of six parts: (1) the final cost, (2) the total wirelength, (3) the chip area, (4) the chip width and height, (5) the runtime in seconds, and (6) the bounding-box coordinate for each macro (specified by the lower-left corner and upper-right corner). The report file format is shown below. )

```
<final cost>
// Cost = αA + (1-α)W
<total wirelength>
//W = Σ_{n_i∈N} HPWL(n_i)
<chip_area>
// area = (chip_width) * (chip_height)
<chip_width> <chip_height>
//resulting chip width and height
<program_runtime>
//report the runtime in seconds
<macro_name>    <x1>    <y1>    <x2>    <y2>
<macro_name>    <x1>    <y1>    <x2>    <y2>
// (x1, y1): lower-left corner, (x2, y2): upper-right corner
```

Figure 3 Output file format ( *output.rpt* )

*Note: All the x&y coordinates should be integer.*

**Command-Line Parameters**

In order to test your program, you are asked to add the following command-line parameters to your program (Please name your execution file name as “Lab2” ):

**./Lab2 [α value] [input.block name] [input.net name] [output name]**

For example, “**./Lab2 0.5 input.block input.nets output.rpt**”.

**Example**

Figure 4 illustrates an example of the IO files (assume α = 0.5):

**Input files (input.block):**

```
Outline: 120 120
NumBlocks: 4
NumTerminals: 0
A    40    50
B    60    50
C    60    50
D    40    50
```

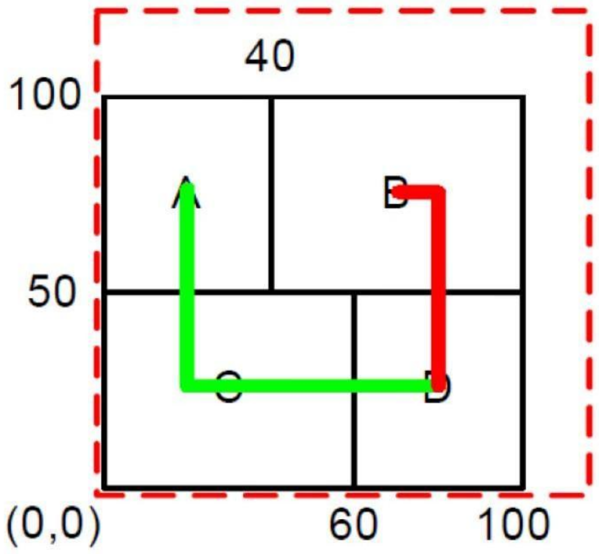


Figure 4 A floorplanning problem and its solution

**(input .nets)**

```
NumNets: 2
NetDegree: 3
A
C
D
NetDegree: 2
B
D
```

**Output files (output.rpt)**

```
5085
170
10000
100  100
0.24
A    0    50    40    100
B    40   50    100   100
C    0    0     60    50
D    60   0     100   50
```

**Output files (output.rpt)**

```
5085                              Cost
170                               HPWL
10000                             Area
100  100                Width & Height
0.24                           Runtime
A    0    50    40    100
B    40   50    100   100
C    0    0     60    50
D    60   0     100   50
```

**HPWL (Half-Perimeter Wire Length)**: The smallest square such that all net pins are inside. Use the **center point** of the Macro as its pin (round down to integer) as shown in Figure 5.

**Note: You need to round down HPWL of each net.**

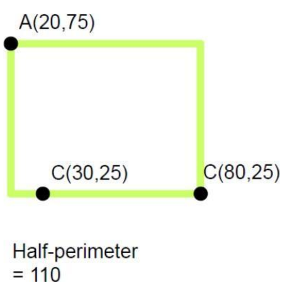


Figure 5 Example of HPWL

**Resources**

Sample input files (ami33/ami49/vda317) and a verifier are provided for you.

**Evaluation**

1. You **MUST WRITE YOUR OWN CODE**. Copying codes may result you to **FAIL** this course.
2. Naming rule.
   A. Name of the binary after "make" – Lab2
   B. Execution procedure:
   **./Lab2 [α value] [input.block name] [input.net name] [output name]**
   **./verifier [α value] [input.block name] [input.net name] [output name]**
3. Hidden cases will be evaluated

**Grading**

This programming assignment will be graded based on:

(1) Correctness of the program (no error, all blocks are in the fixed outline)

(2) Solution quality (minimize the cost). The lower the cost, the higher the score you will get.

(3) For each case, the run time limit is up to 300 seconds. It will be regarded as "failed" if you use more than 300 seconds.

(4) Please make sure your code is available on our linux server. If it cannot be executed, you will get zero point.

(5) Accept four days late submission, 10% deduction per day. That is, if you hand in on 11/1, the score will be *0.9; if you hand in on 11/4, the score will be *0.6, and submission will not be accepted after 11/4.

Also, a script on checking similarity for codes will be performed. If two codes exist high similarity. Both codes receive zero mark after confirmation.

The primary objective is to **place all blocks into the fixed outline successfully** and to minimize the **cost**.

**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
```

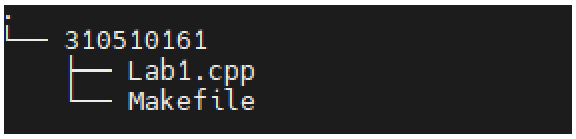
**<2>**

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

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

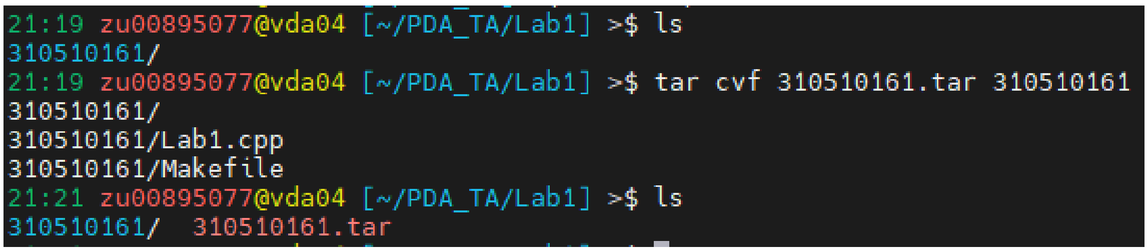
**Example**

**<1>**



```
.
└── 310510161
    ├── Lab1.cpp
    └── Makefile
```

**<2>**



```
21:19 zu00895077@vda04 [~/PDA_TA/Lab1] >$ ls
310510161/
21:19 zu00895077@vda04 [~/PDA_TA/Lab1] >$ tar cvf 310510161.tar 310510161
310510161/
310510161/Lab1.cpp
310510161/Makefile
21:21 zu00895077@vda04 [~/PDA_TA/Lab1] >$ ls
310510161/  310510161.tar
```

**Upload Student_ID.tar to e3**

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

**Supplementary Instructions :**

- The cost of your output should be integer.
- HPWL is calculated as: ( (int) x_max - (int) x_min + (int) y_max - (int) y_min )
- The display method of verifier is scientific notation, and it will only be displayed to 4 decimal places. If the result calculated by verifier is the same as your result, but the COST comparison shows FAIL, this may represent a different value after 4 decimal places in scientific representation. So please double check that you calculate the HPWL in the same way as the TA described above .
- The alpha is determined by the TA during the demo, you only need to make sure that the alpha works between 0 and 1 (including 0 and 1)
- Run time limit is 300 seconds. TA will test it by himself. The runtime in the output file will not be used for judgment.
- The terminal will only be outside the outline or on the outline.