CS6135 VLSI Physical Design Automation

Homework 3: Fixed-outline Slicing Floorplan Design

Due: 23:59, April 22, 2025

---

1. Introduction

In this homework, you are asked to adapt and implement an existing algorithm, published in the DAC-86 paper entitled "A New Algorithm for Floorplan Design" by Wong and Liu, for solving the fixed-outline slicing floorplan design problem with a set of hard blocks.

2. Problem Description

(1) Input:

- A set B of rectangular hard blocks, where each block b_i in B has its width and height.
- A netlist N
- The dead space ratio, which is predefined and passed by the argument. The aspect ratio of the floorplan region is 1, so you can calculate the width w_{fl} and height h_{fl} of the floorplan region as follows:

$$w_{fl} = h_{fl} = \left\lfloor \sqrt{(total\ block\ area) \times \left(1 + (dead\ space\ ratio)\right)} \right\rfloor$$

The width w_{fl} and height h_{fl} will be rounded down from the square root of the total block area times (1 + dead space ratio). For example, if the total block area is 1000000 and the dead space ratio is 0.1, the width w_{fl} and height h_{fl} of the floorplan region are as follows.

$$w_{fl} = h_{fl} = \lfloor \sqrt{1000000 \times 1.1} \rfloor = 1048$$

Then, the coordinates of the lower-left corner and upper-right corner of the floorplan region are (0, 0) and $(w_{fl},\ h_{fl})$, respectively.

(2) Output:

- The total wirelength of all nets, where the wirelength for each net is defined as the half-perimeter wirelength (HPWL) of the minimum bounding box of pins of the net. Each pin of block b_i is located at the center of b_i. Note that the x- or y- coordinate, say c, of each block center is rounded down to an integer k such that $k \leq c \leq k + 1$.
- The coordinates $(x_i,\ y_i)$ of the lower-left corner of each block b_i, as well as the rotation status (1 for rotated, and 0 for unrotated).

(3) Objective:

By assuming each block can be rotated by 90 degrees, the total wirelength of the floorplanning result is minimized subject to the following constraints.

1. Fixed-outline constraint: Each block must be entirely inside the floorplan region.
2. Non-overlapping constraint: No two blocks overlap with each other.

3. Input File

(1) The *.txt* file:

The .txt file specifies the name and other information about the floorplan. Note that block name/pad name/net name is a case-sensitive arbitrary-length alpha-numeric string.

```
NumHardBlocks 100
// NumHardBlocks number of hard blocks
HardBlock hb0 43 33
// HardBlock block name width height
      ⋮

NumPads 334
// NumPads number of pads
Pad p0 0 0
// Pad pad name x-coordinate y-coordinate
      ⋮

NumNets 885
// NumNets number of nets
Net n0 2
// Net net name number of pins on net
Pin p0
Pin hb26
// Pin block name/pad name
      ⋮
```

4. Output File

(1) The *.out* file:

The .out file specifies the floorplanning result including the total wirelength of all nets, the total number of the hard blocks, and the coordinates of the lower-left corner of each block with/without rotation.

```
Wirelength 192561

NumHardBlocks 100
hb96 145 181 1
// block name lower-left corner coordinates (x,y) rotated
hb77 124 266 0
// block name lower-left corner coordinates (x,y) unrotated
      ⋮
```

5. Language/Platform

(1) Language: C/C++

(2) Platform: Unix/Linux

6. Report

Your report must contain the following contents, and you can add more as you wish.

(1) Your name and student ID

(2) How to compile and execute your program and give an execution example.

(3) The wirelength and the runtime of each testcase with the dead space ratios 0.15 and 0.1, respectively. Paste the screenshot of the result of running the **HW3_grading.sh**.

(4) Please show that how small the dead space ratio could be for your program to produce a legal result in 10 minutes.

(5) The details of your implementation. If there is anything different between your implementation and the algorithm in the DAC-86 paper, please reveal the difference(s) and explain the reasons.

(6) Please describe your method of your initial floorplan.

(7) What tricks did you do to speed up your program or to enhance your solution quality? Please use **HW3_printer** to generate the image to compare different stages (e.g., initial floorplan → trick 1 → trick 2 → final result) of your floorplanning results.

(8) What have you learned from this homework? What problem(s) have you encountered in this homework?

7. Required Items

Please compress `HW3/` (using tar) into one with the name `CS6135_HW3_${StudentID}.tar.gz` before uploading it to eeclass.

(1) `src/` contains all your source code, your `Makefile` and `README`.

- `README` must contain how to compile and execute your program. An example is like the one shown in HW2.

(2) `output/` contains all your outputs of testcases for TAs to verify.

(3) `bin/` contains your executable file.

(4) `CS6135_HW3_${STUDENT_ID}_report.pdf` contains your report.

You can use the following command to compress your directory on a workstation:

```
$ tar -zcvf CS6135_HW3_{StudentID}.tar.gz <directory>
```

For example:

```
$ tar -zcvf CS6135_HW3_113000000.tar.gz HW3/
```

8. Grading

- ✓ 50%: Outperform the baseline in public testcases. The wirelength of baseline for each public testcase is listed below. For hidden testcases, you only need to generate a valid result.

Case name	Wirelength	
	dead space ratio = 0.15	dead space ratio = 0.1
public1	247970	248807
public2	479218	495065
public3	693268	696662

- ✓ 30%: The wirelength of each testcase, hidden testcases included.
- ✓ 20%: The completeness of your report

Notes:

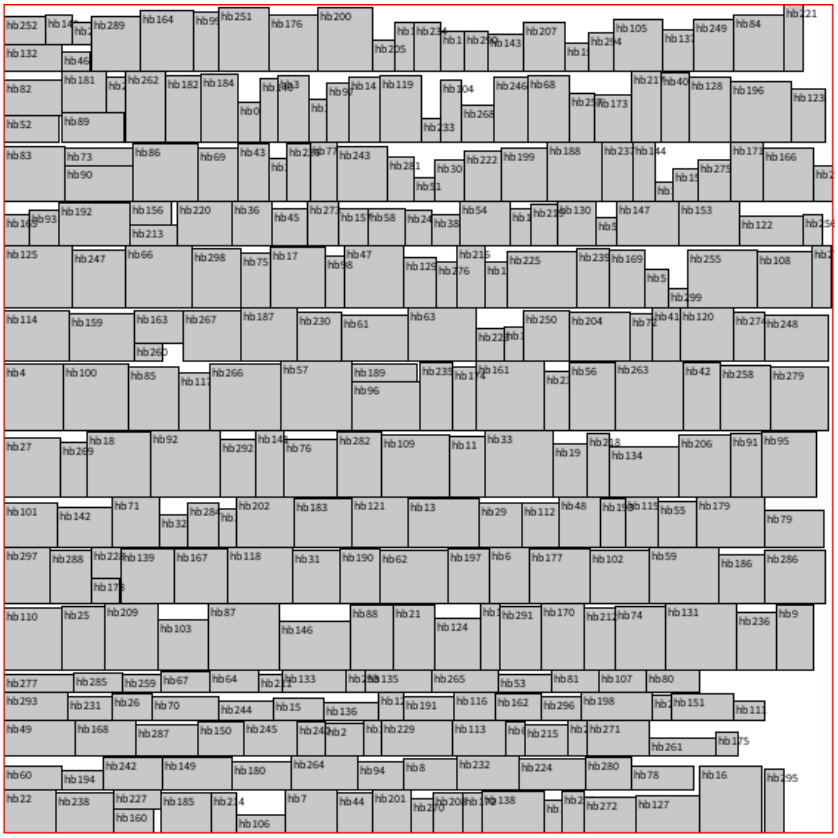
- Make sure the following commands can be executed.
  - Go into directory "`src/`", enter "`make`" to compile your program and generate the executable file, called "`hw3`", which will be in directory "`bin/`".
  - Go into directory "`src/`", enter "`make clean`" to delete your executable file.
- Please use the following command format to run your program.

  ```
  $ ./hw3 *.txt *.out dead_space_ratio
  ```

  E.g.:

  ```
  $ ./hw3 ../testcase/public1.txt ../output/public1.out 0.1
  ```
- Use arguments to read the file path. Do not write the file path in your code.
- Program must be terminated within 10 minutes for each testcase.

- Please use ic21, ic22, ic51, or ic55 to test your program.
- We will test your program by a shell script with GCC 9.3.0 on the servers mentioned above. Please make sure your program can be executed by **HW3_grading.sh**. If we cannot compile or execute your program by the script, you will get 0 points on your programming score.
- For each testcase, you should use **HW3_printer** to draw your result like the following figure and paste the figure on your report.



- Note that any form of plagiarism is strictly prohibited, including the code found on GitHub and the code from any student who took this course before. If you have any problem, please contact TA.