

Software Development for Electronic Design Automation, Spring 2024

Programming Assignment #2

Soft Block Floorplanning (due May 5, 2024 (Sunday) on-line)

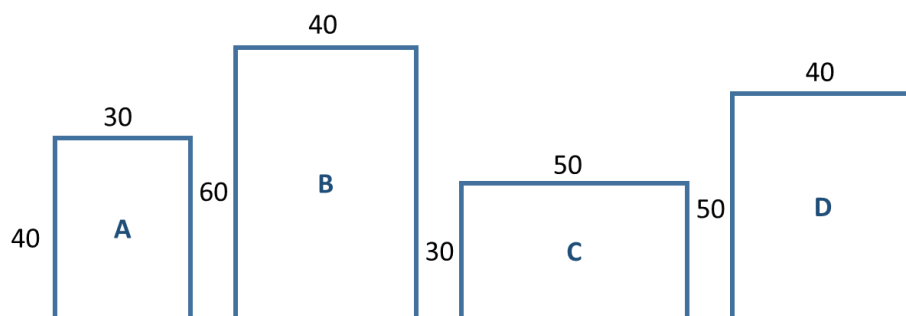
1. Problem Description

This programming assignment asks you to write a chip floorplanner that can handle soft macros. Given a set of rectangle macros, the floorplanner places all macros without any overlaps and makes the overall area as small as possible. The chip aspect ratio should be near 1 as well. We assume that the lower-left corner of this chip is the origin (0,0), and no space (channel) is needed between two macros.

2. Input

Each test case (floorplan_**) gives the number of blocks and the aspect ratio constraint. Then the initial block dimensions are listed. The file format is as follows:

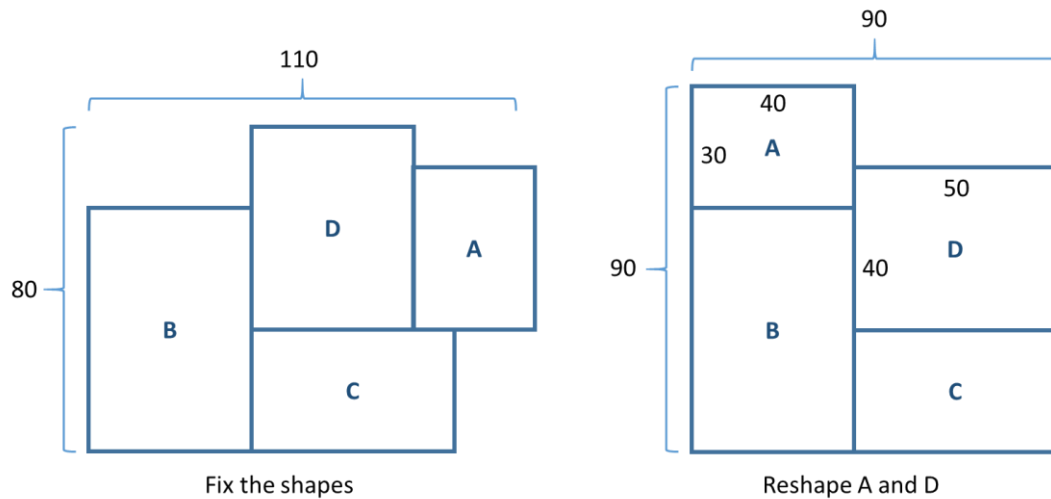
| Input Format | Sample Input |
|--|--------------------|
| NumBlocks <# of blocks> | NumBlocks 4 |
| MinAspectRatio <minimum $\frac{\text{block height}}{\text{block width}}$ > | MinAspectRatio 0.5 |
| MaxAspectRatio <maximum $\frac{\text{block height}}{\text{block width}}$ > | MaxAspectRatio 2 |
| <macro name> <macro width> <macro height> | A 30 40 |
| ... More macros | B 40 60 |
| | C 50 30 |
| | D 40 50 |



3. Output

The output file (*.out) records the problem output, which consists of three parts: (1) the chip width and height, and (2) the bounding-box coordinate for each macro (specified by the lower-left corner and upper-right corner). The output file format is shown below.

| Output Format | Sample Output |
|---|---------------------|
| <chip_width> <chip_height> | 80 90 |
| <macro_name> <macro width> <macro height> <x1> <y1> <x2> <y2> | A 40 30 0 60 40 90 |
| // (x1, y1): lower-left corner, (x2, y2): upper-right corner | B 40 60 0 0 40 60 |
| | C 50 30 40 0 80 30 |
| | D 50 40 40 30 80 70 |



The left example result above uses the initial block dimensions and the right one reshapes Blocks A and D to get a smaller chip area and a better chip aspect ratio.

4. Language/Platform

- (a) Language: C or C++.
- (b) Platform: Unix/Linux or Windows.

5. Command-line Parameter

In order to test your program, you are asked to add the following command-line parameters to your program:

`./floorplanner [input file name] [output file name]`

6. Submission

You need to submit the following materials in a compressed **[student id]-p2.tgz** file (e.g., b11007000-p2.tgz) at the course website by the deadline: (1) source codes, (2) Makefile, (3) a text readme file (readme.txt) stating how to build and conduct your program, and (4) **a report (report.docx) no more than 2 pages introducing your data structures and algorithms**. Please carefully read the following instructions:

- The compressed file [student id]-p2.tgz file contains only a single folder named **[student id]-p2** (e.g., b11007000-p2). Use only lowercase letters for the compressed file and folder names.
- Only a compressed file in the *.tgz format will be accepted.
- **Do not submit files or folders other than those specified above.**
- Please ensure that your work can be successfully executed in the Linux environment.

****If the above requirements are not met, penalties will be imposed**

7. Grading Policy

This programming assignment will be graded based on (1) the correctness, (2) readme.txt and report, (3) **running time no more than 10 minutes**, and (4) solution quality evaluated by:

$$\text{chip width} \times \text{chip height} \times \frac{\max(\text{chip width}, \text{chip height})}{\min(\text{chip width}, \text{chip height})}$$

8. Online Resources

Sample input files can be found at the course website.