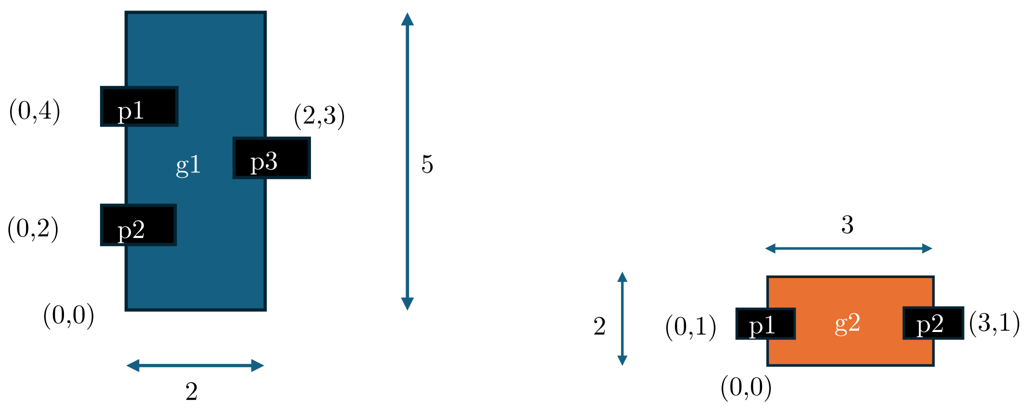
COL215 Software Assignment 2: Wiring-aware Gate Positioning

Deadline: 1st October 2024

1 Introduction

Extending on the gate packing assignment, the gates will contain pins locations as shown in Figure 1. Objective is to minimize the total wire length of the circuit.



(a) Gate 1 (b) Gate 2

Figure 1: Gate pin specification example

2 Problem Statement

Given:

- a set of rectangular logic gates $g_1, g_2 ... g_n$
- ***width*** and ***height*** of each gate g_i
- the input and output pin locations (x and y co-ordinates) on the boundary of each gate $g_i.p_1, g_i.p_2, ..., g_i.p_m$ (where gate g_i has m pins)
- the pin-level connections between the gates

write a program to assign locations to all gates in a plane so that:

- no two gates are overlapping
- the ***sum of estimated wire lengths*** for all wires in the whole circuit is ***minimised***.

2.1 Notes

- Assume that the gates cannot be re-oriented (rotated, etc.) in any way.
- Assuming that all wiring is horizontal and vertical
- Possible estimate for the wire length for a set of connected pins uses the ***semi-perimeter method***: form a rectangular bounding box of all the pin locations; the estimated wire length is half the perimeter of this rectangle.

3 Formats

3.1 Input file format

For each gate, the input file contains the width and height and the corresponding pins co-ordinates. Referring to Figure 1, **g1.p1 is at (0,2), g1.p2 is at (0,4), g1.p3 is at (2,3) with respect to the bottom left corner of g1** and **g2.p1 is at (0,1), g2.p2 is at (3, 1) with respect to the bottom left corner of g2**. Then each wire connections in the following format (see Figure 2) :

```
<name of gate> <width> <height>
<pins> <name of gate> <x_1, y_1> ... <x_m, y_m>
<wire> <g_x.p_x> <g_y.p_y>
```

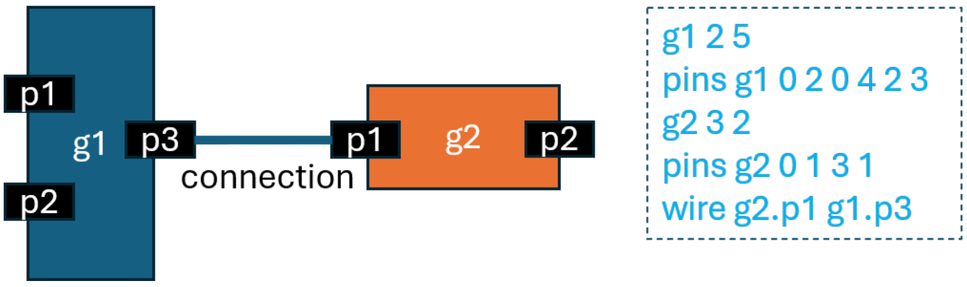


Figure 2: Input file format

3.2 Output file format

The output file begins with a specification of the bounding box, of the form:

```
bounding_box <width> <height>
```

Following the bounding box line, each line of the output file from your program should have the location of the gate, specified as the x- and y-co-ordinates of the bottom left corner as follows:

```
<name of gate> <x co-ordinate> <y co-ordinate>
```

4 Testing instructions

Initial sample test cases will be uploaded on moodle. Additionally, you are required to generate your own test cases to verify the implementation. You need to provide justification (in report) for the generated test cases.

Following are the input constraints:

- **0 < Number of gates <= 1000**
- **0 < Width of gate <= 100**
- **0 < Height of gate <= 100**

5 Assignment Submission Instructions

General assignment instructions that need to be followed for all assignments: only one partner needs to submit. Mention all team member names and entry IDs during the submission.

1. Name the submission file as entryNumber1_entryNumber2.zip or entryNumber1.zip
2. Go to Gradescope via moodle and upload the file under Software Assignment 2.
3. Only one submission per group is required. Gradescope will allow you to select the group partner.
4. The following files should be part of the zip folder:
    - Source files
    - Report as a .pdf file (handwritten report will be rejected). The report needs to state:
        - your design decisions
        - time complexity analysis
        - test cases