COL215 Software Assignment 1: Gate Packing

Deadline: 25 August 2024

1 Introduction

In this assignment, we will attempt to automatically generate a compact physical layout of a gate-level circuit. Assume that all gates are rectangular, and the final layout is also rectangular. In a compact layout, blank space is minimised. See Figure 1 below.

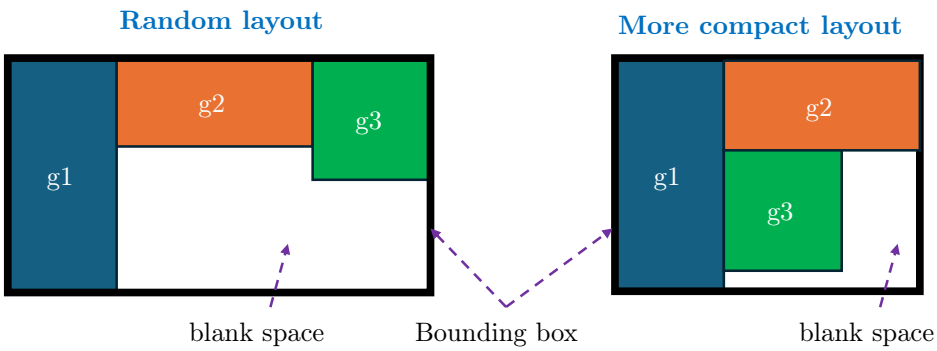

Figure 1: Gate packing example

2 Problem Statement

Given:

- a set of rectangular logic gates $g_1, g_2 ... g_n$
- ***width*** and ***height*** of each gate g_i

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

- no two gates are overlapping
- the ***bounding box*** of the entire circuit (smallest rectangle that encloses all gates) has minimum area.

2.1 Notes

- Assume that the gates cannot be re-oriented (rotated, etc.) in any way.
- In this simplified problem statement, we ignore connections between gates.

3 Formats

3.1 Input file format

Each line of the input file contains the width and height of each gate in the following format (see Figure 2) :

```
<name of gate> <width> <height>
```

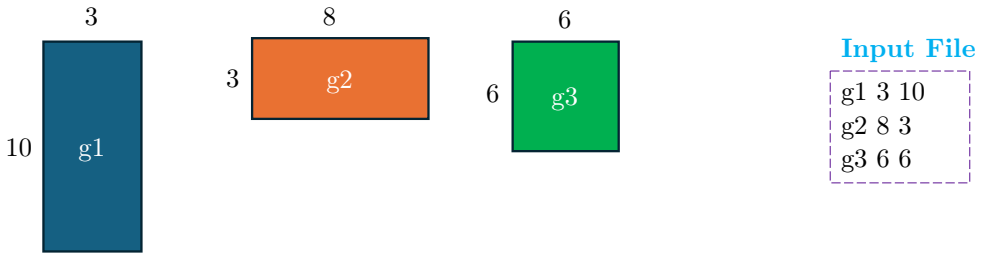

Figure 2: Input file format

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 (see Figure 3):

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

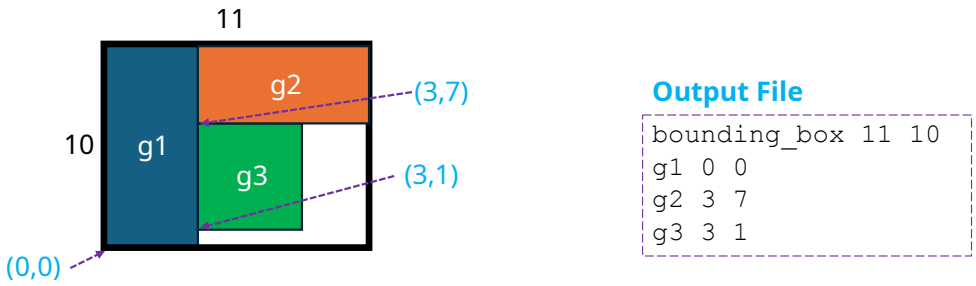

Figure 3: Output file format

4 Visualization of the circuit

Download the following files from moodle:

1. visualize_gates.py
2. Sample input.txt and output.txt

4.1 Prerequisites

Before running the script, make sure you have Python installed on your system. You also need to install the following Python packages:

```
pip3 install tk
pip3 install pillow
```

Script Usage:

1. Script Name: visualize_gates.py
2. Required files:
   - Output File: This file contains the coordinates of the gates in the format

     ```
     bounding_box <width> <height>
     <gate_name> <x_coordinate> <y_coordinate>
     ```

   - Input File: This file contains the dimensions of the gates in the format:

     ```
     <gate_name> <width> <height>
     ```

   - Example:
     - output.txt:

       ```
       bounding_box 11 10
       g1 0 0
       g2 3 7
       g3 3 1
       ```

     - input.txt:

       ```
       bounding_box 11 10
       g1 3 10
       g2 8 3
       g3 6 6
       ```

3. Command-Line Arguments:
   The script requires three arguments:
   - Coordinates File: The file containing the coordinates of the gates.
   - Dimensions File: The file containing the dimensions of the gates.
   - Grid Dimensions: The size of the grid as two integers: number of rows and columns.

   Example command

   ```
   python3 visualize_gates.py output.txt input.txt 50 50
   ```

   Here, 50 50 represents a grid of 50 rows and 50 columns.

4. Running the Script:

   (a) Place your coordinates.txt and dimensions.txt files in the same directory as visualize_gates.py.

   (b) Open a terminal (or command prompt) and navigate to the directory containing the script.

   (c) Run the script using the command provided in the example point 3.

   (d) The script will open a window displaying the grid with gates colored randomly. The gates are labeled with their names, and you can see the grid cells they cover.

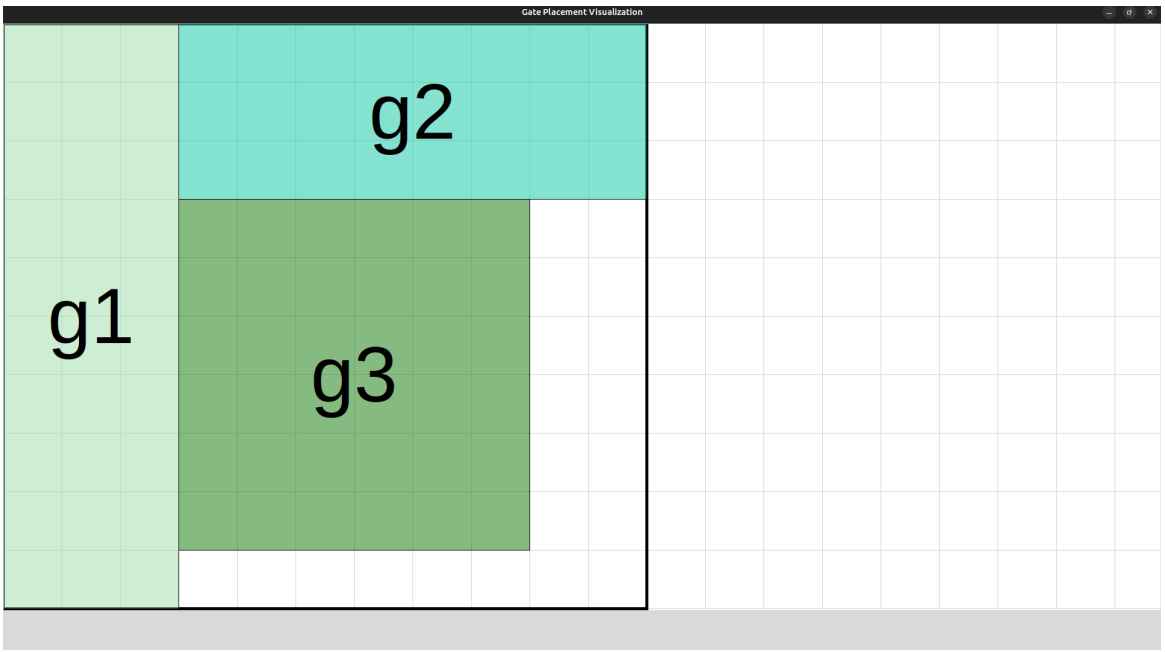

Figure 4: Gate Visualization tool example

5. Using the *draw_gate_packing* function in your implementation:

```
from visualize_gates import draw_gate_packing
.....
.....
"""
Create dictionary for the gate dimensions (gate_dimensions) and
gates coordinates (gates)
"""
#Invoke the GUI for visualization
root = draw_gate_packing(gate_dimensions, gates, (20,20))
root.mainloop()
```

5 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**

6 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 1.
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