COL215: Digital Logic and System Design

Software Assignment 2

Wiring-aware Gate Positioning

This report details our implementation of a circuit layout optimization algorithm using simulated annealing. The goal was to minimize wire length while ensuring no gate overlaps in the final layout.

Algorithm Used: Simulated Annealing

The algorithm uses simulated annealing to optimize the placement of gates while minimizing the total wire length. It iteratively perturbs the layout, accepting worse solutions with a probability that decreases over time, allowing it to escape local optima.

Methods and Attributes Used

Classes:

- Gate: Represents a gate with attributes like name, width, height, and pins.

Methods:

- simulated_annealing(gates, wire_connections, initial_temp, cooling_rate, width, height, iterations): Implements the simulated annealing algorithm for gate placement optimization.
- totalCost(gate_positions, wire_connections, gate_dict): Calculates the total wire length using semi-perimeter wire length (HPWL) method.
- isLayoutValid(gate_positions, gate_dict): Checks if the current layout has any overlapping gates.
- areGatesOverlapping(gate1_name, gate2_name, gate_positions, gate_dict): Determines if two specific gates overlap.
- group_connected_wires(wires): Groups connected wires using a breadth-first search approach.
- read_input_file(filename): Reads the input file and parses gate and wire information.
- writeOutput(gate_positions, wire_connections, total_wire_length, gates, output_file): Writes the optimized layout to an output file.

Key Variables:

- gate_positions: Dictionary storing the current positions of all gates.
- wire_connections: List of wire connections between gates.
- initial_temp: Initial temperature for simulated annealing (set to 1000).
- cooling_rate: Cooling rate for temperature reduction (set to 0.99).

- iterations: Number of iterations for the simulated annealing process (set to 10000).
-

Time Complexity Analysis

Simulated Annealing Loop: O(I * (G^2 + W))

- I: Number of iterations (10000)
- G: Number of gates
- W: Number of wire connections

Key Operations:

1. Layout validity check (isLayoutValid): O(G^2)
2. Cost calculation (totalCost): O(W * P), where P is the average number of pins per connection
3. Gate movement: O(1)

Overall Complexity: O(I * (G^2 + W * P))

The algorithm's performance in practice may be better than this worst-case analysis, especially for well-structured problems where good solutions are found quickly.

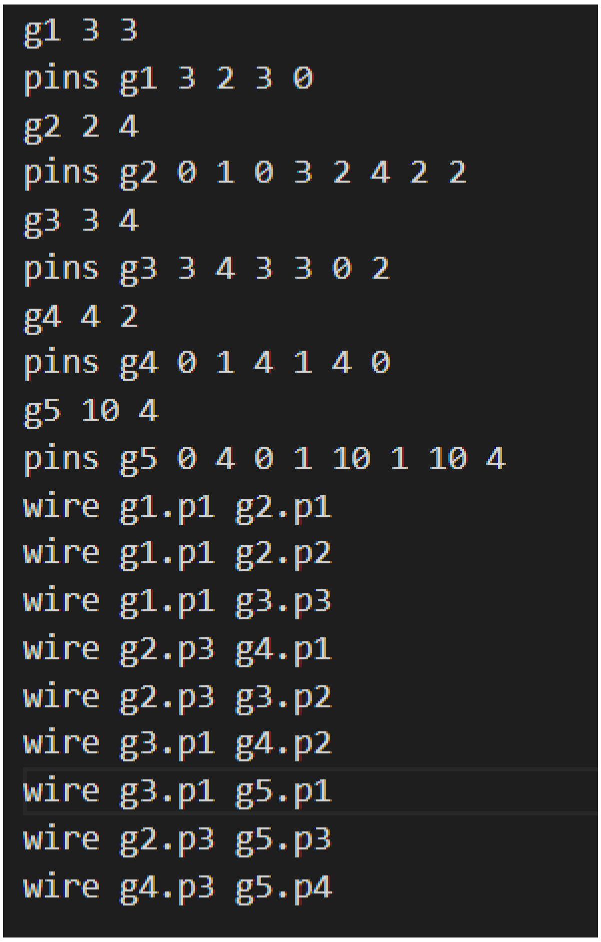
Design Decisions

1. Semi-Perimeter Wire Length (HPWL): Used as a quick approximation for wire length calculation.
2. Random Gate Movement: In each iteration, a random gate is selected and moved to a new random position.
3. Temperature Cooling: A simple geometric cooling schedule is used

   (temperature *= cooling_rate).
4. Initial Layout: Randomly placed gates are used as the starting point, ensuring no overlaps.
5. Wire Grouping: Connected wires are grouped to optimize the cost calculation process.

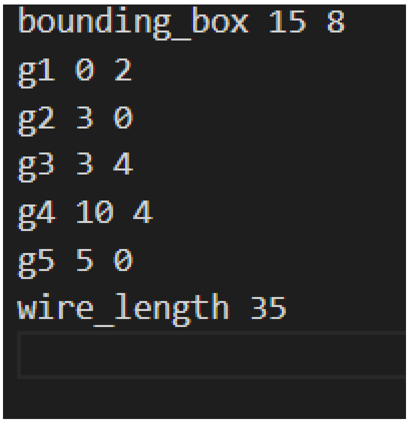
Test Cases:

1)Input

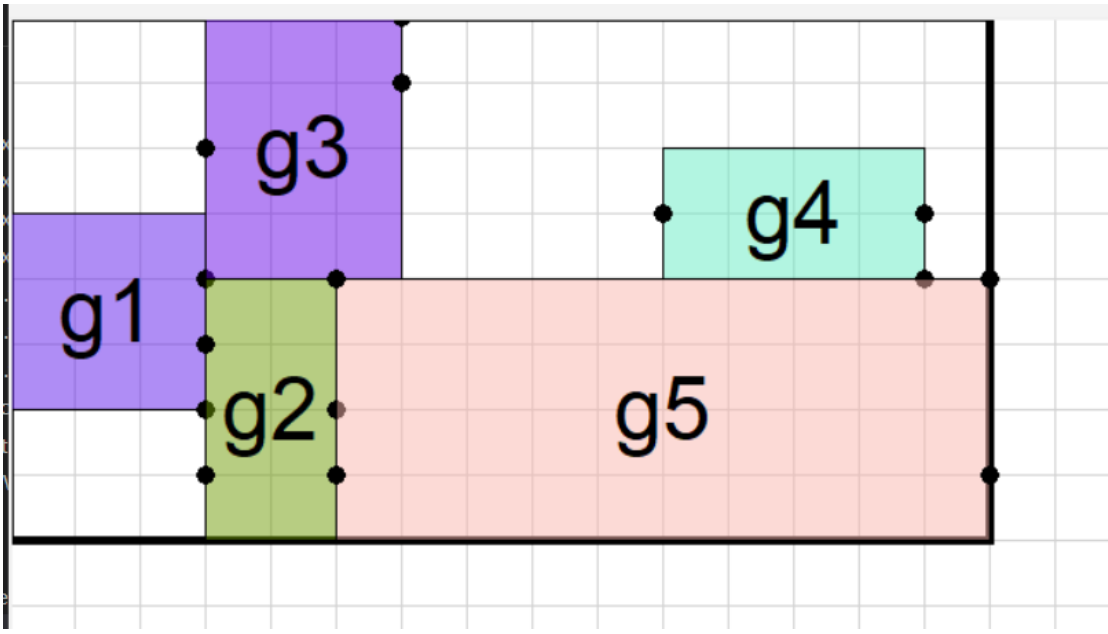


```
g1 3 3
pins g1 3 2 3 0
g2 2 4
pins g2 0 1 0 3 2 4 2 2
g3 3 4
pins g3 3 4 3 3 0 2
g4 4 2
pins g4 0 1 4 1 4 0
g5 10 4
pins g5 0 4 0 1 10 1 10 4
wire g1.p1 g2.p1
wire g1.p1 g2.p2
wire g1.p1 g3.p3
wire g2.p3 g4.p1
wire g2.p3 g3.p2
wire g3.p1 g4.p2
wire g3.p1 g5.p1
wire g2.p3 g5.p3
wire g4.p3 g5.p4
```

Output



```
bounding_box 15 8
g1 0 2
g2 3 0
g3 3 4
g4 10 4
g5 5 0
wire_length 35
```



2)Input

g1 3 3

pins g1 0 1 3 2

g2 4 4

pins g2 0 0 4 3

g3 2 4

pins g3 0 1 2 3

g4 3 3

pins g4 0 1 3 2

g5 4 3

pins g5 0 1 4 2 0 2

g6 2 3

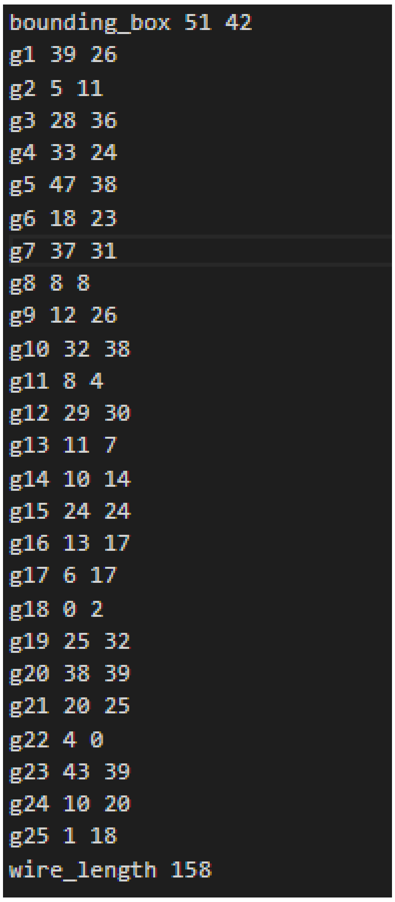
pins g6 0 0 2 2

```
g7 3 5

pins g7 0 1 3 4

g8 3 3

pins g8 0 1 3 2

g9 4 4

pins g9 0 0 4 3 4 1

g10 2 3

pins g10 0 0 2 2

g11 3 4

pins g11 0 1 3 3

g12 3 5

pins g12 0 2 3 4 3 1

g13 2 4

pins g13 0 1 2 3 0 2

g14 4 3

pins g14 0 0 4 2

g15 3 3

pins g15 0 1 3 2 3 3

g16 2 5

pins g16 0 1 2 4

g17 3 4

pins g17 0 2 3 3 3 4

g18 3 3

pins g18 0 0 3 1 3 2

g19 4 4

pins g19 0 0 4 3

g20 2 3

pins g20 0 0 2 2

g21 3 4

pins g21 0 2 3 3 3 0

g22 3 5
```

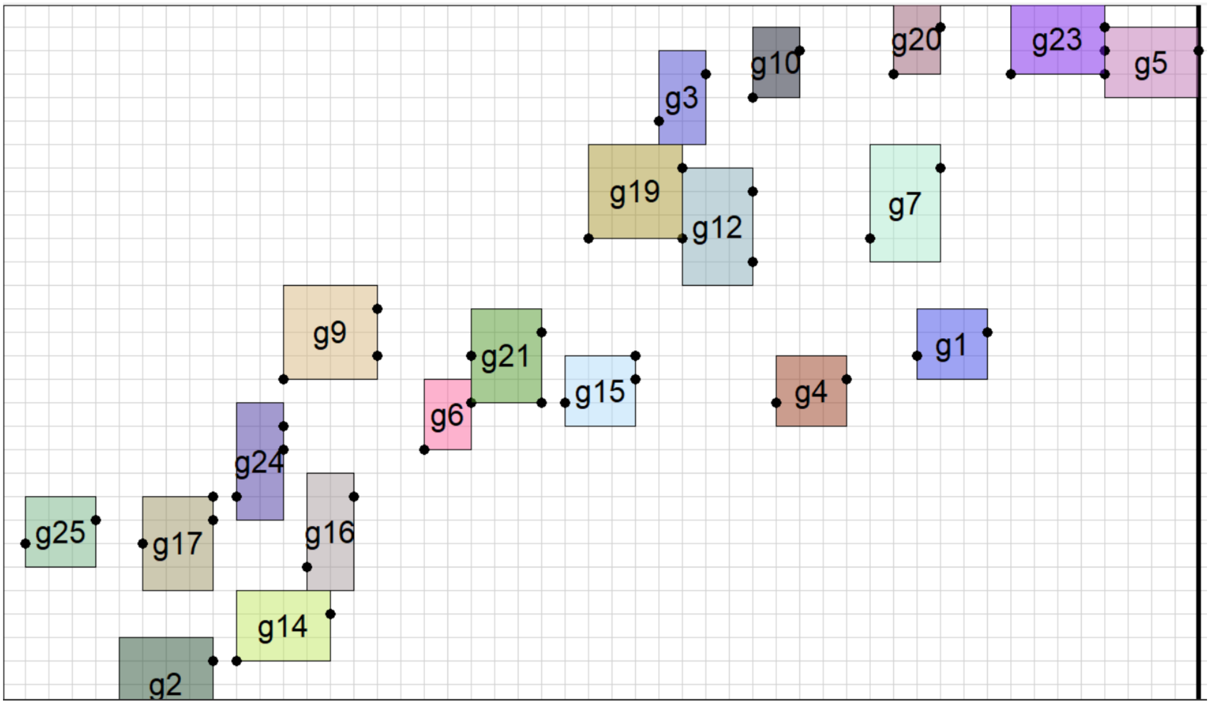
```
pins g22 0 1 3 4
g23 4 3
pins g23 0 0 4 2
g24 2 5
pins g24 0 1 2 4 2 3
g25 3 3
pins g25 0 1 3 2
wire g1.p1 g4.p2
wire g2.p1 g8.p1
wire g3.p2 g10.p1
wire g4.p1 g15.p2
wire g5.p2 g20.p1
wire g6.p2 g9.p1
wire g7.p1 g12.p3
wire g8.p2 g13.p2
wire g9.p3 g16.p1
wire g10.p2 g23.p1
wire g11.p1 g18.p2
wire g12.p1 g19.p2
wire g13.p3 g14.p1
wire g14.p2 g24.p1
wire g15.p3 g21.p2
wire g16.p2 g17.p3
wire g17.p1 g25.p2
wire g18.p3 g22.p1
wire g19.p1 g3.p1
wire g20.p2 g7.p2
wire g21.p3 g6.p2
wire g22.p2 g11.p2
wire g23.p2 g5.p3
wire g24.p3 g9.p2
```

wire g25.p2 g2.p2

Output



```
bounding_box 51 42
g1 39 26
g2 5 11
g3 28 36
g4 33 24
g5 47 38
g6 18 23
g7 37 31
g8 8 8
g9 12 26
g10 32 38
g11 8 4
g12 29 30
g13 11 7
g14 10 14
g15 24 24
g16 13 17
g17 6 17
g18 0 2
g19 25 32
g20 38 39
g21 20 25
g22 4 0
g23 43 39
g24 10 20
g25 1 18
wire_length 158
```



3)Input

g1 3 4

pins g1 0 3 0 1 3 2 3 1

g2 4 4

pins g2 0 2 4 3 4 1

g3 2 4

pins g3 0 1 0 2 0 3 2 2

g4 2 3

pins g4 0 1 2 2 2 1

wire g1.p1 g3.p2

wire g1.p3 g4.p2

wire g1.p4 g2.p1

wire g4.p1 g3.p3

wire g4.p3 g3.p1

wire g2.p2 g3.p4

wire g2.p3 g1.p2

Output

bounding_box 11 5

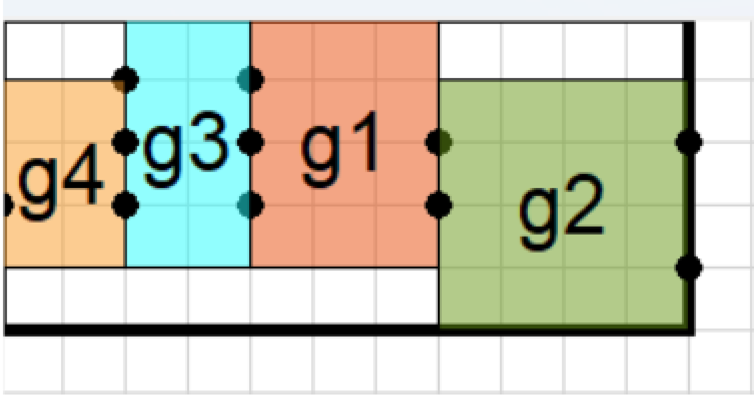
g1 4 1

g2 7 0

g3 2 1

g4 0 1

wire_length 27



4)Input

g1 3 3

pins g1 0 1 0 2 3 1 3 2

```
g2 4 1
pins g2 0 1 4 0
g3 4 1
pins g3 4 1 4 0 0 1 0 0
g4 4 3
pins g4 0 3 4 1 4 2 0 1
g5 4 4
pins g5 0 1 0 2 0 3 4 1 4 2 4 3
g6 3 3
pins g6 0 1 0 2 3 1 3 2
g7 3 4
pins g7 0 0 0 2 0 3 3 1
g8 4 4
pins g8 0 1 0 2 0 3 4 1
wire g3.p2 g7.p1
wire g7.p2 g3.p1
wire g5.p5 g3.p1
wire g5.p1 g8.p4
wire g2.p2 g6.p4
wire g2.p1 g6.p1
wire g6.p2 g4.p1
wire g1.p4 g6.p1
wire g4.p3 g8.p1
wire g1.p1 g8.p3
wire g3.p3 g4.p4
```

Output

```
bounding_box 15 14
g1 4 7
g2 5 10
g3 0 2
g4 3 4
```

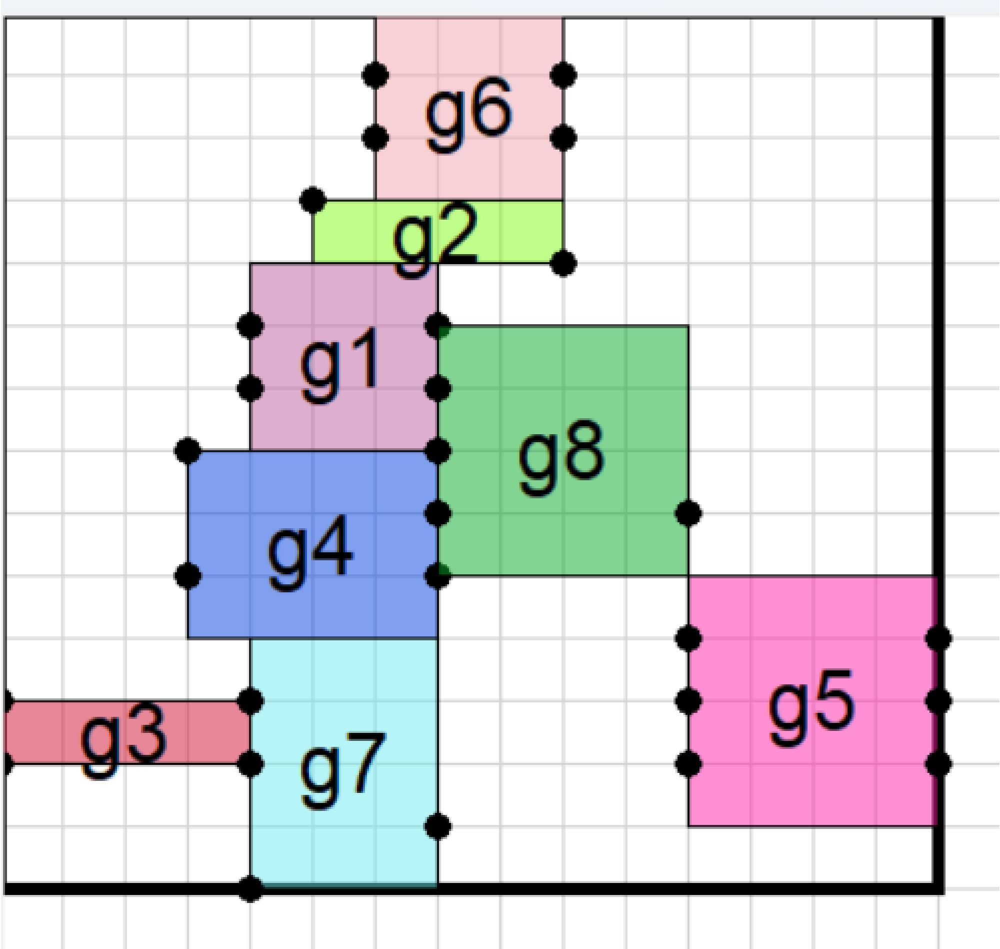
g5 11 1

g6 6 11

g7 4 0

g8 7 5

wire_length 43



5)Input

g1 1 1

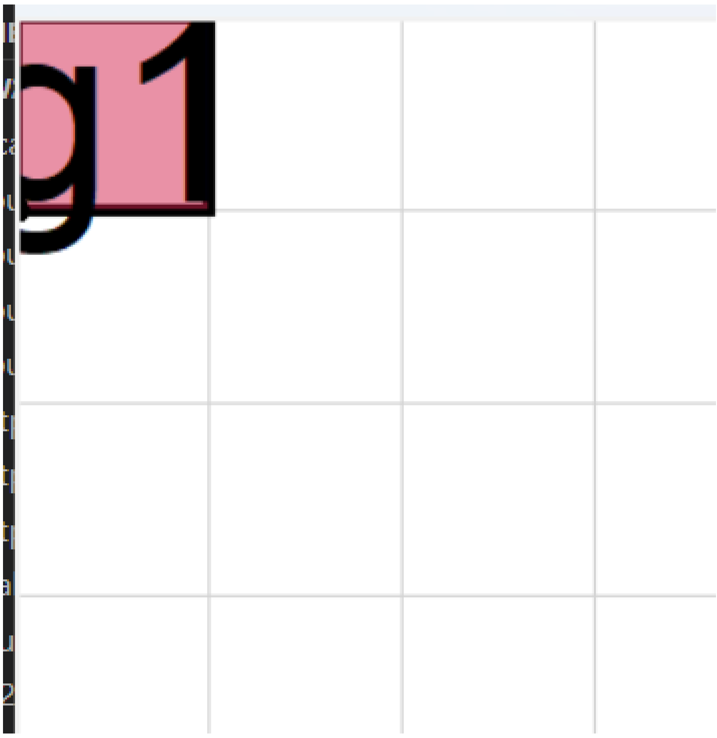
pins g1 0 0

Output

bounding_box 1 1

g1 0 0

wire_length 0



6)Input

g1 2 2

pins g1 0 0 0 2 2 0 2 2

g2 2 2

pins g2 0 0 0 2 2 0 2 2

g3 2 2

pins g3 0 0 0 2 2 0 2 2

g4 2 2

pins g4 0 0 0 2 2 0 2 2

wire g1.p1 g2.p2

wire g1.p3 g3.p1

wire g2.p4 g4.p3

wire g3.p2 g4.p4

Output

bounding_box 6 4

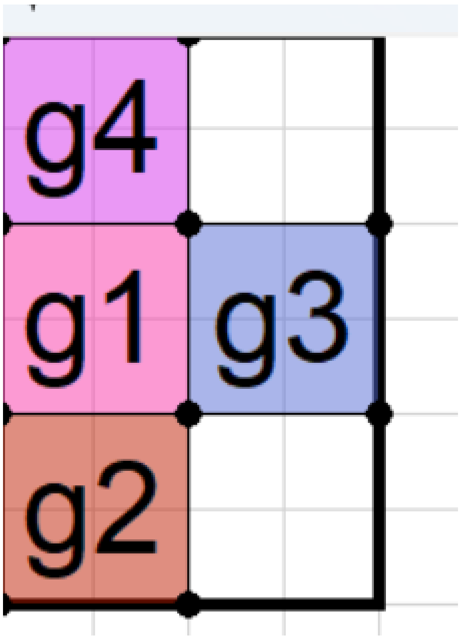
g1 2 2

g2 0 0

g3 4 2

g4 0 2

wire_length 4



Submitted By:

Khushi Mishra(2022TT12159)

Lakshya Batra(2022TT12163)