COL215 Software Assignment 3

Timing Optimization in Gate Positioning

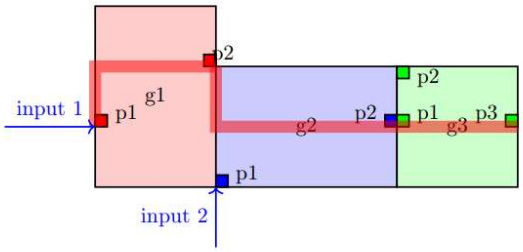
Date – 20 October,2024

S No.	Name	Entry No.	Group
1.	Sourabh Verma	2023CS50006	03
2.	Deepak Sagar	2023CS50124	03

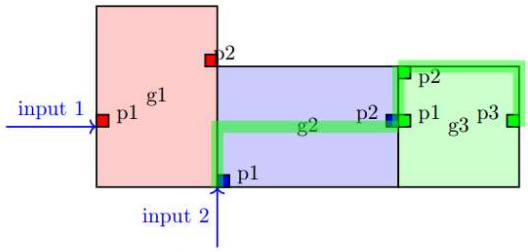
Circuit:

Gates containing different pins on the left(input) and right(output) sides connected to other gates.

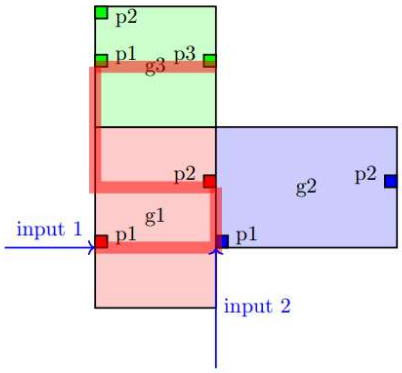
Also, there is delay for each gate and an uniform delay per wire length.

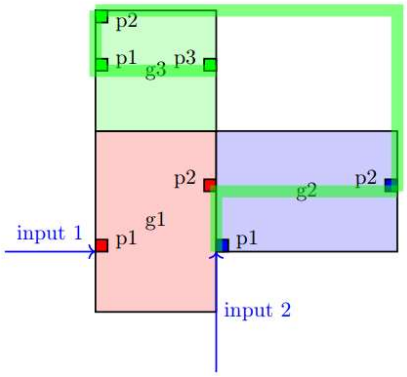


(a) Path I from input 1 to Output 1



(b) Path II from input 2 to Output 1





Aim:

minimize the critical path and critical path delay of whole circuit.

Where, Critical Path of a circuit is the longest path delay from any primary input to primary output of the circuit.

Method:

Placing the gates by using

Procedure:

We started by taking a random placement of gates in space and then using simulated annealing we found out the optimum placement of the gates which gives us the minimum critical path delay.

For finding critical path delay of a configuration of gates we used first checked which gate has a primary input and then iteratively started traversing from each primary input. While traversing we checked which next pin will lead us to the maximum path delay for which we used recursion and calculated the path delay from that pin. In this way we found out the maximum path delay and the critical path for the given placement of the gates. Then we used simulated annealing to find out optimal configuration of gates by randomly moving one of the gates from its place to any other place decided randomly. We ran this step 1 lakh time and got the best possible configuration of gates and hence the optimum critical path.

Constraints on INPUT:

- **$0 <$ Number of gates ≤ 1000**
- **$0 <$ Width of gate ≤ 100**
- **$0 <$ Height of gate ≤ 100**
- **$0 <$ Number of pins on one side of a gate $\leq$ Height of Gate**
- **$0 <$ Total Number of pins ≤ 40000**

Time and Space Complexity Analysis:

G=No. Of Gates, P = No. Of Pins, W = Wiringsq

Total Time Complexity:

- **Input parsing**: O(G+P+W)
- **Sorting gates**: O(GlogG)
- **Placing each gate:** O(G)+O(G)
- **Calculating total wire length**: O(W*G)
- **Bounding box calculation**: O(G)
- *For delay calculation and random function : O(10^4*(100+W^2))*

Overall Time Complexity = O(G^2 + G*W + 10^4*(100+W^2))

Total Space Complexity:

- Storing gates and their connections: O(G+W)
- Storing pins: O(P)
- Temporary variables for sorting and placement: O(G)

Overall Space Complexity = O(G+P+W)

Testing Strategy:

- Will check for small cases, no overlapping should happen
- Will check for medium cases, for efficient working or not
- Will run the code for large gate values too, for confirming time complexity and space complexity

- Will check in each whether cyclic components raise error or not

Test Cases:

Tc 1:

```
bounding_box 5 5
critical_path g1.p1 g1.p2 g3.p1 g3.p3
critical_path_delay 21
g1 1 0
g2 0 3
g3 3 2
```

```
g1 2 3 5
pins g1 0 1 2 2
g2 3 2 3
pins g2 0 0 3 1
g3 2 2 6
pins g3 0 1 0 2 2 1
wire_delay 4
wire g1.p2 g3.p1
wire g2.p2 g3.p2
```

(tc-1)

Motivation: test case given in the Moodle itself, helps in working for disconnected wiring.

Tc 2:

```
g1 4 5 2
pins g1 0 1 0 4 4 3 4 4
g2 4 3 3
pins g2 0 2 4 3
g3 4 2 10
pins g3 0 1 4 0
g4 5 5 4
pins g4 0 5 0 2 5 1
g5 2 3 6
pins g5 0 1 2 3
wire_delay 5
wire g1.p3 g3.p1
wire g1.p4 g2.p1
wire g2.p2 g4.p1
wire g3.p2 g4.p2
wire g4.p3 g5.p1
```

```
bounding_box 17 8
critical_path g1.p1 g1.p3 g3.p1 g3.p2 g4.p2 g4.p3 g5.p1 g5.p2
critical_path_delay 43
g1 0 0
g2 5 5
g3 5 2
g4 9 1
g5 15 1
```

(tc-2)

Motivation: test case of moodle too

Tc 3:

```
g1 4 3 2
pins g1 0 2 4 1
g2 3 2 1
pins g2 0 0 3 2
g3 4 5 6
pins g3 0 5 0 3 4 5 4 3 4 1
g4 3 3 9
pins g4 0 2 3 1
g5 3 3 6
pins g5 0 0 3 3 0 3
g6 5 2 3
pins g6 0 1 5 0
wire_delay 2
wire g1.p2 g3.p2
wire g2.p2 g3.p1
wire g3.p5 g6.p1
wire g3.p3 g4.p1
wire g3.p3 g5.p3
wire g4.p2 g5.p1
```

```
bounding_box 21 11
critical_path g1.p1 g1.p2 g3.p2 g3.p5 g6.p1 g6.p2 g3.p3 g4.p1 g4.p2 g5.p1 g5.p2 g3.p3 g5.p3 g5.p2
critical_path_delay 38
g1 0 5
g2 2 3
g3 5 2
g4 9 7
g5 12 8
g6 16 0
```

(tc-3)

Motivation: moodle test-case

Tc 4:

```
Cycle found in the graph
```

```
g1 10 10 10
pins g1 0 1 0 2 0 4 10 6 10 9
g2 10 10 10
pins g2 10 8 10 6 0 4 10 2 0 0
g3 10 10 10
pins g3 0 0 0 10 10 0 10 10 0 5 10 5
wire_delay 5
wire g1.p3 g2.p4
wire g1.p5 g3.p1
wire g3.p6 g2.p3
wire g3.p5 g1.p1
```

(tc-4)

Motivation: simple cyclicity check

Tc 5:

```
Cycle found in the graph
```

```
g1 15 10 6
pins g1 0 0 0 2 0 4 15 1 15 3 15 5
g2 10 7 4
pins g2 0 0 0 2 10 3 10 5
g3 20 12 8
pins g3 0 1 0 3 0 5 20 2 20 4 20 6 20 8
g4 12 8 5
pins g4 0 0 0 2 12 3 12 6
wire_delay 6
wire g1.p1 g2.p3
wire g3.p5 g4.p2
wire g2.p4 g1.p6
```

(tc-5)

Motivation: complex cyclcitty check randomly generated by Al

Tc 6:

```
bounding_box 23 31
critical_path g1.p1 g1.p6 g2.p5 g2.p4 g3.p2 g3.p5
critical_path_delay 183
g1 0 8
g2 7 0
g3 7 21
```

```
g1 20 12 6
pins g1 0 1 0 3 0 5 0 7 0 9 20 6
g2 14 8 5
pins g2 0 2 0 4 0 6 0 8 14 4
g3 16 10 5
pins g3 0 1 0 3 0 5 0 7 16 5
wire_delay 6
wire g1.p6 g2.p5
wire g2.p4 g3.p2
```

(tc-6)

Motivation: 1 output pin and multiple input pins

Tc 6:

```
bounding_box 7 4
critical_path g2.p1 g2.p2 g3.p2 g3.p3 g4.p2 g4.p3
critical_path_delay 26
g1 3 1
g2 0 2
g3 5 1
g4 4 0
```

```
g1 2 3 5
pins g1 0 1 2 2
g2 3 2 3
pins g2 0 0 3 1
g3 2 2 6
pins g3 0 1 0 2 2 1
g4 3 1 4
pins g4 0 0 3 1 3 0
wire_delay 3
wire g1.p2 g3.p1
wire g2.p2 g3.p2
wire g2.p2 g4.p1
wire g3.p3 g4.p2
```

(tc-7)

Motivation: multiple output pins , less input pins

==========END==========