Student Name: LUO QIN
Student ID: 1155184128
Department: Computer Science & Engineering

CENG 5270: EDA for Physical Design of Digital Systems
Homework 3

---

1. **Principles**

   Our implementation is mainly based on the A* search maze routing. The overall flow of our global router is as follows:

   i) Read the netlist and construct the routing graph. Here we use the 2-dimensional array to store the routing graph information. The first dimension is the column id of the routing grid, where the odd column are the vertical routing edges and the even column are the horizontal routing edges.

   ii) Determine the net ordering. Here, we suppose that the nets with the smallest bounding box have the highest priority because the net with a larger routing box would have more choices for constructing the routing path.

   iii) For each single net, we would use the A* search to construct the routing path. The pseudo-code of A* search tree is as follows. Here, to incorporate both the wirelength and the resource overflow into consideration, we use the combination of the hpwl and the overflow as the cost in the priority queue and the movement from the source gridnode to its neighbors.

   $$Cost = (1 - \lambda)(WL + HPWL(cur, dst)) + \lambda OV \tag{1}$$

   Here, WL is the number of path segments from the source gridnode to the current gridnode with the smallest cost. $HPWL(cur, dst)$ is the HPWL from the current gridnode to the destined gridnode, representing an approximation towards the future cost. OV is the overflow of the segment, which is calculated as the difference between the demands and the capacity.

   λ is the coefficient to weight the wirelength term and the overflow term. At the beginning, the λ is set as 0, meaning in the early stage, we concentrate more on the wirelength minimization. Then the λ would increase by 0.1 for every 200 nets. After routing 1800 nets, the λ would increase to 1.0, fully concentrating on the overflow minimization.

   iv) After getting the routing paths for each net, we calculate the total wirelength and the total overflow.

2. **Running the Program**

   In our implementation, we use Cmake to compile the overall program.

```
$ cd src
$./scripts/build.py -o release
```

   Then the executable file *hw3* would be generated in the *bin* folder. Then we could run the executable file with the following command as:

```
$cd bin
$./hw3 <in_file_path> <out_file_path>
```

   For example, if we want to run the maze router on ibm01.modified.txt, we could run:

Algorithm 1 A*-tree

```
1: procedure AStarTree(start, goal)
2:     openSet ← empty priority queue
3:     closedSet ← empty set
4:     openSet.enqueue(start)
5:     start.cost ← 0
6:     start.heuristic ← calculateHeuristic(start, goal)
7:     while not openSet.isEmpty() do
8:         current ← openSet.dequeue()
9:         if current = goal then
10:            return constructPath(current)
11:        closedSet.add(current)
12:        for all neighbor in current.neighbors do
13:            if neighbor in closedSet then
14:                continue
15:            newCost ← current.cost + calculateCost(current, neighbor)
16:            if neighbor not in openSet or newCost < neighbor.cost then
17:                neighbor.cost ← newCost
18:                neighbor.heuristic ← calculateHeuristic(neighbor, goal)
19:                neighbor.priority ← newCost + neighbor.heuristic
20:                neighbor.parent ← current
21:                if neighbor not in openSet then
22:                    openSet.enqueue(neighbor)
23:                else
24:                    openSet.updatePriority(neighbor)
25:    return constructPath(null)
```

Table 1: Results of the maze routing based on A* search

Testcase	Basic Information		Results		
	Nets	GridSize	Total Overflow	Total Wirelength	Runtime (s)
ibm01.modified	13357	(64,64)	2227	58523	0.76
ibm02.modified	22456	(80,64)	3470	157724	1.65
ibm03.modified	21609	(80,64)	3185	143818	1.57
ibm04.modified	27781	(96,64)	4586	159162	2.19

```
$ ./hw3 ../testcase/ibm01.modified.txt ../output/ibm01.modified.out
```

3. **Experiment Results**

   We run our maze router on the four provided testcases (ibm01.modified to ibm04.modified), and the results are as Table 1.

   The results show that the total total overflow of these 4 cases concentrate between 2000 and 4000. The minimum wirelength is 2227, and the largest wirelength is 4586. The largest runtime is 2.19 seconds. We could also observe that as the number of nets and the grid size increases, the total overflow, wirelength and runtime would increase.