BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CSE482 (VLSI Design Sessional), July 2018 Term

Assignment 1: Implementation of Lee and Soukup Algorithm

1 Introduction

In this assignment you have to implement two maze routing algorithms for global routing. The first one is Lee's Algorithm and the second one is Soukup's Algorithm.

2 Tasks

2.1 Implementation of Lee's Algorithm

Lee's Algorithm in given in the Figure 8.16 in [1]. The algorithm takes source s and target t in the input, and finds a shortest path with breadth first technique between s and t. That is, this algorithm is designed to route only one net in the grid graph. In our implementation, we will have multiple nets that need to be routed. To solve this, we will use the same algorithm for multiple times. In the input file, the nets to be routed are given in order, which means the first net is to be routed first. The next net is to be routed after the first one is done and similarly for others. When routing a net, the vertices that are used by the previous nets must be thought as blocked so that no nets intersect one another. Implement the algorithm for a single net, that is for one source and one target. Then take the first net from the input file and route the net through the grid graph. When the routing is done, block the vertices that are used to route the net. Then use this updated grid graph to route the next net using the same algorithm.

2.2 Implementation of Soukup's Algorithm

Soukup's Algorithm in given in the Figure 8.18 in [1]. Soukup's algorithm takes same inputs as Lee's algorithm. Implement the algorithm for single net routing. And then use it multiple times for multiple routing as mentioned for Lee's algorithm. Change the definition of $NGHBR - IN - DIR(v_i, v_j)$ as given: *NGHBR-IN-DIR(vi,vj)* returns the neighbor v_k of v_j such that $DIR(v_j, v_k)$ does not go beyond the direction of $DIR(v_i, t)$.

2.3 To Do

For a input file, run both the algorithms and generate two separate output files for two algorithms in the format as specified. You have to show the complete grid graph with blocked vertices and routed nets. If routing of certain net is not possible, mention that in the output file and try routing for the next ones. For a certain net, you have to consider the source and target vertices of all the other nets as blocked vertices. Denote each vertex as '0'; each blocked vertex as '#', source and target as 'S' and 'T' respectively for routed net, and source and target as 's' and 't' respectively for nets that cannot be routed. Use '*' to represent the path from 'S' to 'T'. For exploring neighbors, use this order: up, left, down, right.

2.4 Bonus

Find an order of the nets in which all the nets can be routed, if possible.

3 Input

- Input is to be taken from a text file.
- The first line of the input file contains an integer n, which denotes $n \times n$ grid graph.
- The next line contains an integer b, which is the number of blocked vertices.
- Each of the next b lines contain the row and column number of a blocked vertex in the form i,j.
- The next line contains an integer c that denotes the number of two terminal nets.
- Each of the next c lines contain the row and column number of two vertices for the two terminals of the net in the form i,j k,l.

See example input file for clarification.

4 Output

- Output is to be given in two different text files for two algorithms. Output format is similar for both.
- Print the entire grid as shown in the sample output file. Use the symbols for vertices, blocked vertices and net connections as shown.
- State if any net cannot be routed.

. See the example output file for clarification.

5 Rules

- **General Rules**
  - Plagiarism is strongly prohibited. Any kind of plagiarism will result into NEGATIVE marking.
  - NO submission will be granted after deadline.
  - Input must be taken from file.
  - Output must be written to file.
  - Input output format must be same as specified.
- **Language**
  You can use Java or C++ for implementation. Implementation in other languages will NOT be evaluated.
- **Submission Rules**
  Submission will be taken in moodle. Steps to be followed are given below.
  - Create a folder named as your 7-digit student ID (i.e. 1405002) in your local machine.
  - Put only the source files (.java or .cpp) in the folder created. (Do NOT copy-paste the whole project)
  - Compress the folder in zip format which must also be named as 7-digit student ID.
  - Submit the zip file in moodle before deadline. Submission Deadline

6 Submission deadline

Submission deadline is 2.00 pm on 4 February, 2019 (Monday).

7 Conclusion

Implement the algorithms so that it works for a single net. Check if it works perfectly before trying routing for multiple nets. For any query please feel free to email me at tmadnan10@gmail.com.

References

[1] Naveed A Sherwani. Algorithms for VLSI physical design automation. Springer Science & Business Media, 2012.