Accelerating pipelined Bellman-Ford Algorithm for calculating single source shortest distance on XEN10 FPGA Board
EE214 Project
RA InCharge : Ritu Sharma(213079023)

Introduction

Bellman-Ford is a shortest-path algorithm that finds the shortest paths from a single source vertex to all other vertices in a weighted graph. The Bellman-Ford algorithm works by iteratively relaxing the edges of the graph. The relaxation process involves updating the tentative distance of the vertices based on the current shortest paths found so far. The algorithm repeats this relaxation process for a certain number of iterations (equal to the number of vertices minus 1), ensuring that it gradually refines the distance estimates until the optimal shortest paths are found.

Overall, Bellman-Ford is a fundamental algorithm for finding shortest paths in graphs and is widely used in networking, routing, and other applications where the graph may have varying edge weights.

Objective

Develop a high-level architecture for the Bellman-Ford algorithm on an FPGA board, taking into account the limitations of on-chip memory and computational resources. Divide the algorithm into four stages: memory read block, sorting block, computation block, and memory write block.

It is required to do **parallel processing** in this project to increase the throughput. Hence, you are supposed to take 4 words at a time from the memory. Each word contains information about the weight of the edge, source node number, and destination number.

Once you get your input you can start finding the shortest distance according to the **Bellman-Ford algorithm** which includes sorting and computational blocks.

Also, this project should be **Pipelined** to increase the throughput further. The overall architecture is divided into 5 stages, so you need to implement 4 pipelined registers.

The 5 stages of the pipeline are Reading graph, memory read, sorting, computation, and memory write.

Weekly milestones

Pre-requisite:

- Understand the Bellman-Ford algorithm and its application in finding the shortest path between two nodes in a weighted graph.
- Read the paper " Accelerating Large-Scale Single-Source Shortest Path on FPGA" [1] for the project architecture.

Week 1

- Fix the architecture which includes
    - Max no of nodes and edges
- Implement the architecture in VHDL, starting with small modules and sub-modules
    - Memory files
    - Shorting Block
    - Computational Block
    - Pipeline Register
    - Forwarding Block

Week 2

- Once the modules and sub-modules are working correctly, integrate them into the main module using a hierarchical design.
- Validate the implementation using test benches and simulations in ModelSim Altera.
- Once the simulation results are satisfactory, program the FPGA board with the VHDL code and use JTAG-UART to communicate with the system.
- Provide the input to the Bellman-Ford algorithm using switch pins as node address inputs and output to LEDs to represent the weight of the node whose address is given as input. Collect the output from the algorithm and cross-check it with the expected output to validate the implementation on the XEN10 board [3].
- Write code for traceback of path in python/C/C++.
- Trace back the shortest distance path from the predecessor values on the FPGA itself and display using an LED matrix.

BONUS

- Transform an image of a maze into a graph representation, then apply the above algorithm to find the path from the maze's entrance to its exit. Finally, convert the resulting path back into an image format to display the solution.

References

[1] Introduction to Algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein

[2] https://www.cs.albany.edu/~cchelmis/pubs/raw15.pdf

[3]https://algorithms.discrete.ma.tum.de/graph-algorithms/spp-bellman-ford/index_en.html