Shortest Path Algorithms Verification with Idris

Eileen Feng

November 16, 2018

Shortest path problem is the problem that aims to find the path with minimum weights between two nodes in a graph. Dijkstra's and Bellman-Ford are two of the well-known algorithms for solving shortest path problem.[3][4][6] These two algorithms are widely applied in our daily life, for instance finding the shortest path in road map, or the minimum-weight route in networks. Given their significance in solving real-life issues, it is important and worthwhile to prove their correctness. In addition, it is amazing and valuable to explore how algorithm verification can be represented as a programming issue, and that verifying the correctness of an algorithm can be achieved not only through theoretical proof, but also through implementation.

Idris is a functional programming language that supports dependent types. It is built on a logically sound foundation, which indicates its capability for algorithm verification. The reason of choosing Idris over other proof assistants such as Coq, is that Idris is developed as a programming language first, and proof assistant second. As Dijkstra's and Bellman-Ford are practical algorithms that are frequently implemented in real-life applications, we hope to run their verification as programs. Existing researches on proving Dijkstra's algorithm with Coq lays a solid foundation for our verification of Dijkstra's in Idris.[1][5] Although there are few extant attempts in verifying Bellman-Ford Algorithm, existing work on functional algorithms serves as a good reference for methodologies in algorithm verification.[2]

We plan to approach this problem through the following steps. First, we have to define concepts involved in both algorithms, which include but not limited to the concept of node, edge, path, and graph. Secondly, based on our definitions, we will find a way of proving that both algorithms indeed find the shortest path within the graph. One way of approaching this might be investigating the relations between the shortest path with all the other paths in the graph. We will provide lemmas and intermediate proofs along the way if necessary. This is remarkable because it allows us to verify algorithms not only through theoretical proofs, but also through implementations.

References

[1] Coq proof assistant, 1999-2018.

[2] Andrew W. Appel. Verified functional algorithms, 2018.

[3] Richard E. Bellman. On a routing problem. *Quarterly of Applied Mathematics*, 16:87–90, 1958.

[4] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[5] Kevin Hammond Edwin Brady, Christoph Herrmann. Lightweight Invariants with Full Dependent Types. *TFP 2008*.

[6] A. Shimbel. Structure in communication nets. *Polytechnic Press of the Polytechnic Institute of Brooklyn*, page 199–203, 1955.