FB-APSP: A new efficient algorithm for computing all-pairs shortest-paths

Dyson Pereira Junior, Emilio Carlos Gomes Wille

PII: S1084-8045(18)30244-3
DOI: 10.1016/j.jnca.2018.07.014
Reference: YJNCA 2180

To appear in: *Journal of Network and Computer Applications*

Received Date: 7 December 2017
Revised Date: 24 May 2018
Accepted Date: 23 July 2018

FB-APSP: A New Efficient Algorithm for Computing All-Pairs Shortest-Paths

Dyson Pereira Junior and Emilio Carlos Gomes Wille

***Abstract*—We describe a new forward-backward method for an all-pairs shortest-paths (APSP) algorithm. While most APSP algorithms only scan edges forward, the algorithm proposed here also scans all edges backward because it assumes that edges in the outgoing and incoming adjacency lists of the vertices appear with the same importance. The running time of the algorithm on a directed graph with n vertices, and m edges and positive real-valued edge weights in a deterministic way is $O(n\delta^2\lceil 2^{\log_\delta(n-1)-1}\rceil)$, and the space complexity is $O(n\delta\lceil 2^{\log_\delta(n-1)-1}\rceil)$, where $\delta = m/n$ is the density of the graph. Simulations on graphs with up to 10000 vertices with real, positive weights show that our FB-APSP algorithm, using additional working space less than 75% of space used by Speed-Up Floyd-Warshall algorithm, is faster on large sparse graphs, particularly planar graphs.**

***Index Terms*—Algorithms, Graph Theory, Networks, Routing, Traffic Control.**

I. Introduction

The all-pairs shortest-paths (APSP) problem is one of the most fundamental graph problems and has multiple applications, such as network routing in communications protocols, transportation routing in road networks and robotics, and VLSI design in microelectronics. The APSP problem [1] is to determine, for a weighted, directed graph $G = (V, E, c)$, the shortest path between every pair of vertices, where $|V| = n$, $|E| = m$, and $c : E \to \mathbb{R}^+$ is the edge weight.

When solving APSP problems on graphs in the real world, the running time of the algorithm and the size of its working space are the key issues. Many algorithms that solve the APSP problem efficiently use a large working space ($O(n^2)$) [1]. Division of the APSP problem into n SSSP (single-source shortest-path) problems has become one of the classic approaches used to solve the problem [1]. An SSSP algorithm is run once for each source-vertex. Dijkstra's algorithm [2] can solve SSSP problems on graphs with positive edge weights and solves the APSP problem in $O(mn \log n)$ time using a binary heap [3] and in $O(mn+n^2 \log n)$ time using a Fibonacci heap [4]. When Dijkstra's algorithm is used to solve the APSP problem, the term n-Dijkstra is normally used. The algorithm is a good choice in many real-world situations, although various proposals for solving the SSSP problem have been put foreward [5]. The information contained in the shortest paths from the other vertices is not available to Dijkstra's algorithm while an SSSP from a source vertex is being calculated. Some algorithms that use the information contained in the shortest paths from multiple source vertices are discussed below. One approach that uses a dynamic programming formulation is the Floyd-Warshall algorithm [6], [7].

A speed-up version of the Floyd-Warshall algorithm was proposed by Aini and Salehipour [8] for which the APSP problem is solved with lower computational cost. Such algorithm, hereinafter referred to as the SU-FW algorithm, reduces the amount of calculation from that required by the Floyd-Warshall algorithm substantially.

The property *every subpath of a shortest path is a shortest path* [1] was used by Demetrescu and Italiano in the algorithm proposed in [9] to reduce the number of operations in a priority queue. Priority queues are considered *bottlenecks* for many shortest path algorithms, including Dijkstra's algorithm. The algorithm proposed by Demetrescu and Italiano (hereinafter referred to as DI algorithm), requires a large working space of the order of $O(n^2)$ for finding the APSP on general directed graphs with non-negative real-valued edge weights.

In this work, we propose a new forward-backward method for an all-pairs shortest-paths algorithm. While most APSP algorithms only scan edges forward, the algorithm proposed here also scans all edges backward because it assumes that edges in the outgoing and incoming adjacency lists of the vertices appear with the same importance. Hereinafter our algorithm will be referred to as FB-APSP algorithm. In order to solve the shortest-paths problem, two extreme approaches are known: one that makes use of the shortest paths from multiple source vertices when it is executing but uses a large working space ($O(n^2)$), and another that uses a small working space when it is executing ($O(m + n)$) but does not make use of the shortest paths from other source vertices. In this paper we describe our contribution to finding a solution to the APSP problem for any graph using an algorithm that is the best combination of these two extremes.

In order to validate our proposal, we compare results derived from the FB-APSP algorithm with those from the DI algorithm [9], from the FW algorithm [1] and from the SU-FW algorithm [8] (to our knowledge the best in literature). Often, when evaluating a computational algorithm, the execution time is the most important factor. Our proposal takes into account this issue in detriment to the use of memory space. Our analysis shows that the FB-APSP algorithm is substantially faster than SU-FW at the expense of using a little more memory. The running time of the FB-APSP algorithm on a directed graph with positive real-valued edge weights in a deterministic way is $O(n\delta^2\lceil 2^{\log_\delta(n-1)-1}\rceil)$, and the space complexity is $O(n\delta\lceil 2^{\log_\delta(n-1)-1}\rceil)$, where $\delta = m/n$ is the density of the graph.

D. Pereira Junior and E. C. G. Wille are with the Department of Electronics, Federal Technological University of Paraná (UTFPR), Curitiba, PR, Brazil. e-mail: dyson@utfpr.edu.br, ewille@utfpr.edu.br.

Manuscript received April 19, 2005; revised August 26, 2015.