Course Report of Wuhan University

Summary and Consideration of Shortest-Path Distance Queries on Large Networks

Major Name:	Computer Science and technology	
Course:	Spatiotemporal management and data analysis	
Teacher:	Xiaofang Zhou, Lei Li, Pingfu Chao	
Group Members:	Wang Xu（汪旭）	2017301500205
	Long Xiaoyi（龙晓怡）	2018302100026
	Chen Boxue（陈泊学）	2017302290050

2020.7

Abstract

This paper is a summary of several methods[1][2][3] to solve the shortest path queries in large-scale networks. The ideas and experimental results of these methods are briefly described. The paper includes introductions to the shortest path problem and large-scale network problems, as well as analyze and summarize of several papers.

keywords: The shortest path methods, Algorithm, Large networks

CONTENTS

1. Shortest-Path problem introduction

The shortest path problem is a classical algorithm problem in graph theory, which aims to find the shortest path between two nodes in a graph. The specific forms of the algorithm include the following:

First, the shortest path problem of determining the starting point - that is, the problem of finding the shortest path with known starting nodes.Second, the shortest path problem to determine the end point - contrary to the problem to determine the start point, the problem is to find the shortest path with known end nodes. In the undirected graph, the problem is the same as the problem of determining the starting point. In the directed graph, the problem is the same as the problem of determining the starting point by reversing the direction of all paths.Thirdly, the shortest path problem of determining the starting point and the end point is to find the shortest path between two nodes.Finally, global shortest path problem - finding all shortest paths in a graph.

Today, the most commonly used path algorithms are: Dijkstra algorithm, Bellman Ford algorithm, Floyd algorithm, a * algorithm. A high-level service requires a complex network with wide coverage and detailed line information. If each shortest path is processed, a complete topological network must be generated, but in fact most of the points are not the nodes required, which will inevitably cause computational waste.[1]

2. Large networks problem introduction

In recent years, various types of real network data including the World Wide Web, online social networks and semantic networks have emerged in large numbers, the scale continues to grow. Considering that the scale of the system is usually extremely large, the classic shortest path algorithm generally has problems such as high computational complexity and excessive storage consumption when dealing with large-scale problems. Research on various acceleration technologies and acceleration algorithms around large-scale networks has become the focus of attention in recent years.

For solving the shortest path, traditional algorithms such as Dijkstra algorithm and Floyd algorithm are both precise algorithms with a time complexity of $O(V^3)$. They are still useful for small-scale networks. In the face of such a large network, the speed of the computer is not enough to complete the calculation in a sufficiently fast and acceptable time.

3. Prior work

The most commonly used path algorithms are: Dijkstra algorithm, Bellman Ford algorithm, Floyd algorithm, a * algorithm.By comparing the advantages and disadvantages of various algorithms, you can understand whether they are suitable for solving the shortest path query in large-scale networks problems and where improvements are needed.

Dijkstra algorithm:

Advantages: low complexity O (n * n), heap Optimization: O (n * logn)

Disadvantage: in some instances of the single source shortest path problem, there may be negative weight edges. If the graph G = (V, e) does not contain a negative weighted loop reachable from the source s, the definition of the weight of the shortest path D (s, V) is still correct for all V ∈ V, even if it is a negative value. However, if there is a negative loop reachable from s, the definition of the weight of the shortest path cannot be established. There is no shortest path from s to the node in the loop. The Dijkstra algorithm fails when there is a negative weight in the digraph.

Bellman Ford algorithm:

Advantages: After optimization, set a decision in the cycle. When a cycle is no longer relaxed, exit the cycle directly and make a negative weight ring decision.

Disadvantage: In practice, the Belman Ford algorithm often gives the solution before reaching the value of | v| - 1, which is actually the maximum value

Floyd algorithm:

Floyd algorithm is suitable for all pairs shortest paths. It is a dynamic planning algorithm. The dense map has the best effect and the edge weight can be positive or negative. This algorithm is simple and effective. Because of the compact triple loop structure, the efficiency of the algorithm is higher than that of the algorithm of | v| Dijkstra and SPFA.

Advantages: easy to understand, can calculate the shortest distance between any two nodes, simple code.

Disadvantage: high time complexity, not suitable for calculating a large number of data.

A * algorithm:

Advantages:in solving some problems, we hope to find the shortest path of state space search, that is to say, to solve the problem with the fastest method, which is what a * does.

Disadvantage: the higher the accuracy, the more time-consuming.

4. New solutions

4.1 Towards shortest path identification on large networks

4.1.1 Problem

The use of Big Data in today’s world has become a necessity due to the massive number of technologies developed recently that keeps on providing us with data such as sensors, surveillance system and even smart phones and smart wearable devices they all tend to produce a lot of information that need to be analyzed and studied in details to provide us with some insight to what these data represent. In this paper they focus on the application of the techniques of data reduction based on data nodes in large networks datasets by computing data similarity computation, maximum similarity clique and then finding the shortest path in a quick manner due to the data reduction in the graph. As the number of vertices and edges tend to increase on large networks the aim of this article is to make the reduction of the network that will cause an impact on calculating the shortest path for a faster analysis in a shortest time.

4.1.2 Solution

They have proposed in this paper a fast method and an algorithm for the approximation and reduction of a large network and a fast estimate calculation for the shortest path, in addition to previously proposed algorithms for distance estimation. They described an uncomplicated yet powerful approach to the graph data that will function on the collected similarity graph data that is computed by the reduction algorithm and achieving at query execution time that beat’s up the classical Dijkstra’s shortest path algorithm with large datasets. Thus this will help some other application that uses a large network to have a better, faster response time further experiments could be done using alternative data reduction methods that might have an effect on data reduction with greater response time. Along with the rise of Big Data and vast increase in volumes and the diversity of the data. This vast increase in the volume and also the diversity of the data as well as the speed or velocity that these data are generated with along with the veracity, uncertainty of the data due to certain inconsistency or the ambiguity generated from such data makes it a Big Data problem by covering the main four characteristics of Big Data .Another further method that could be implemented is by modifying the algorithm in finding the shortest path to try to eliminate the 0 or infinite values. Thus reducing the load on the memory and could result in a faster response. This approach has helped by reducing the number of edges in a network graph and as well as nodes in the graph by combining similar nodes based on thealgorithm.

4.1.3 Result

This paper proposes a new algorithm to deal with the shortest path problem of large-scale graphs, which is a fast estimation algorithm that can be used for approximation and reduction of large networks. The author used this method in the experiment to defeat the dijstra algorithm and large data sets when querying the shortest path, thus drawing the conclusion that this new algorithm will be helpful for obtaining faster response times in large networks.

The author shows us in the paper that the algorithm reduces the nodes while retaining the information of other nodes after the merger, and explains the principle of reducing the large network to a small network.

4.2 Fast Exact Shortest-Path Distance Queries on Large Networks by Pruned Landmark Labeling

4.2.1 Problem

A *distance query* asks the distance between two vertices in a graph. Without doubt, answering distance queries is one of the most fundamental operations on graphs, and it has wide range of applications. They can compute the distance for each query by using a breadth first search (BFS) or Dijkstra’s algorithm. However, they take more than a second for large graphs, which is too slow to use as a building block of these applications. In particular, applications such as socially-sensitive search or context-aware search should have low latency since they involve real-time interactions between users, while they need distances between a number of pairs of vertices to rank items for each search query. Therefore, distance queries should be answered much more quickly, say, microseconds. The other extreme approach is to compute distances between all pairs of vertices beforehand and store them in an index. Though they can answer distance queries instantly, this approach is also unacceptable since preprocessing time and index size are quadratic and unrealistically large. Due to the emergence of huge graph data, design of more moderate and practical methods between these two extreme approaches has been attracting strong interest in the database community .

Generally, there are two major graph classes of real-world networks: one is road networks, and the other is complex networks such as social networks, web graphs, biological networks and computer networks. For road networks, since it is easier to grasp and exploit structures of them, research has been already very successful. Now distance queries on road networks can be processed in less than one microsecond for the complete road network of the USA . In contrast, answering distance queries on complex networks is still a highly challenging problem. The methods for road networks do not perform well on these networks since structures of them are totally different. Several methods have been proposed for these networks, but they suffer from drawback of scalability. They take at least thousands of seconds or tens of thousands of seconds to index networks with millions of edges. To handle larger complex networks, apart from these exact methods, approximate methods are also studied. That is, they do not always have to answer correct distances. They are successful in terms of much better scalability and very small average relative error for random queries.

4.2.2 Solution

In this paper, they proposed a novel and efficient method for exact shortest-path distance queries on large graphs. The method is based on distance labeling to vertices, which is common to the existing exact distance querying methods, but their labeling algorithm stands on a totally new idea. The algorithm conducts breadth-first search (BFS) from all the vertices with pruning. Though the algorithm is simple, the pruning surprisingly reduce the search space and the labels, resulting in fast preprocessing time, small index size and fast query time. Moreover, they also proposed another labeling scheme exploiting bit-level parallelism, which can be easily combined with the pruned labeling method to further improve the performance. Extensive experimental results on large-scale real-world networks of various types demonstrated the efficiency and robustness of our methods. In particular, their method can handle networks with hundreds of millions of vertices, which are two orders of magnitude larger than the limits of the previous methods, with comparable index size and query time.

They plan to investigate ways to handle even larger graphs, where indices and/or graphs might not fit in main memory. The first way is to reduce the index size by reducing graphs exploiting obvious parts and symmetry and compressing labels by making dictionaries of common subtrees for shortest path trees. Another way is disk-based or distributed implementation. As they stated in Section 6, disk-based query answering is obvious and ready, and the challenges are particularly on preprocessing. However, since our preprocessing algorithm is a simple algorithm based on BFS, they can leverage the large body of existing work on BFS. In particular, since pruning can be done locally, the preprocessing algorithm would perform well on BSP-modelbased distributed graph processing platforms .

4.2.3 Result

In order to evaluate the performance of the method, the author tested in a real network including five social networks, three network diagrams and three computer networks.The performance of the method is evaluated by comparing the average query time of 1,000,000 random queries. The preprocessing time and the size of the index are also criteria for testing the robustness and efficiency of the algorithm. In addition, the author compares the two existing methods with the highest performance—hierarchical hub labeling and tree decompositions to illustrate the efficiency of the method.

The experimental results show that the efficiency of the methods in the paper on some networks has been significantly improved. The biggest improvement is the Gnutella network which is a graph created from a snapshot of the Gnutella P2P network in August 2002. The average query time of hierarchical hub labeling and tree decompositions are 11μs and 19μs respectively, while the method in the paper only uses 5.2μs .

4.3 Fast Shortest Path Distance Estimation in Large Networks

4.3.1 Problem

Recently, new motivating applications have arisen in the context of web search and social networks. In web search, the distance of a query's initiation point (the query context)to the relevant web-pages could be an important aspect in the ranking of the results . In social networks, a user may be interested in finding other users, or in finding content from users that are close to her in the social graph. This socially sensitive search model has been suggested as part of the social network search experience. Using the shortest path distance as a primitive in ranking functions for search tasks is the main motivation of our work. Although computing shortest paths is a well studied problem, exact algorithms can not be adopted for nowadays real-world massive networks, especially in online applications where the distance must be provided in the order of a few milliseconds. A full breadth-first search (BFS) traversal of a web-graph of 4M nodes and 50M edges takes roughly a minute in a standard modern desktop computer.

4.3.2 Solution

Motivated by applications such as context-aware web search and socially-sensitive search in social networks, they studied how to do fast and accurate distance estimation on real-world massive graphs using landmarks. They characterized the problem of optimal landmark selection and proved that it is NP-hard. They described several strategies for landmark selection which outperform the current approximate standard Random and the state-of-the-art exact techniques by large margins according to our extensive experimentation. In the simplest class of strategies, Centrality appears to be much more robust than Degree. Among the more elaborate strategies, the ones based on partitioning, in particular Border/P, exhibit consistent savings in computational cost across all datasets. When applied to the task of context-aware search these methods yield results of high precision.

In work they are studying methods for an effective synergy of upper and lower bounds. They also plan to investigate the issue of dynamic data structures and how to deal with frequent network updates. A key aspect is to investigate methods to provide estimates in grams for distance functions other than the shortest path distance, which could also be used as primitives in context and/or social aware search tasks.

4.3.3 Result

This paper evaluates the experimental results by designing performance indicators. The author uses p@k to indicate the size of the intersection between the first k elements returned by the approximate distance and the first k elements of X in the result set. X may be constrained by distance, so the author expands X to include elements at the same distance as the k-th element on X, because any of these elements can be regarded as the first k results of the query.

The author tried 5 across datasets, using 20 and 100 landmarks, and the methods in the paper all performed well. Among the five across datasets, the efficiency of the methods in paper except Y!IM has been improved by nearly 10%.

References

[1]Akiba T , Iwata Y , Yoshida Y . Fast Exact Shortest-Path Distance Queries on Large Networks by Pruned Landmark Labeling[J]. 2013.

[2]Selim, Haysam，Zhan, Justin. Towards shortest path identification on large networks. 2016

[3]Potamias .M,Bonchi.F,Castillo. C,
Gionis.A .Fast shortest path distance estimation in large networks.2009

[4]Wu L , Xiao X , Deng D , et al. Shortest Path and Distance Queries on Road Networks: An Experimental Evaluation[J]. Proceedings of the VLDB Endowment, 2012, 5(5):406-417.