Shortest Path Analysis for Efficient Traversal Queries in Large Networks

Waqas Nawaz, Kifayat Ullah Khan and Young-Koo Lee

Dept. of Computer Engineering, Kyung Hee University
Seocheon-dong, Giheung-gu, Yongin-si, Gyeonggi-do, 446-701, Korea

Abstract

The shortest path problem is among the most fundamental combinatorial optimization problems to answer reachability queries. A single pair shortest path computation using Dijkstra’s algorithm requires few seconds on a very large social network. Many traversal algorithms imply precomputed information to speed up the runtime query process. However, computing effective auxiliary data in a reasonable time is still an active problem. It is hard to determine which vertices or edges are visited during shortest path traversals. In this paper, we provide an empirical analysis on how traversal algorithms behave on different real life networks. First, we compute the shortest paths between a set of vertices. Each shortest path is considered as one transaction. Second, we utilize the pattern mining approach to identify the frequency of occurrence of the vertices appear together. Further, we evaluate the results in terms of network properties, i.e. degree distribution, average shortest path, and clustering coefficient, along with the visual analysis.

Keywords: Shortest Paths, Dijkstra, Frequent Pattern Mining, Social Network, Traversal Queries, Large Graphs, Social Network

1Corresponding author


1 Introduction

The shortest path (SP) problems are among the most fundamental combinatorial optimization problems with many applications, and still remain an active area of research [1]. The shortest path play crucial role while activating the communication links in telephone routes whenever user makes a call from one location to another. The careful selection of number of lanes in each road while designing the road system also requires assuming shortest path strategy followed by passenger to estimate the traffic load. SP is used in finance for arbitrage, assembly line inspection system design, traffic simulation, image segmentation, drug target identification and message routing. Nevertheless, the graph or network analysis requires SP for identifying the graph median, community detection, social search, social networking, and message passing.

A single pair shortest path (SPSP) computation using Dijkstra’s algorithm requires few seconds on a very large graphs even with efficient graph processing systems [5] [4]. Many traversal algorithms imply precomputed information to speed up the runtime query process. The auxiliary edges are introduced during preprocessing stage to accelerate the shortest path traversal queries on very large graphs [2]. The process of adding these auxiliary edges to the original graph is effective and can be done in a reasonable time. However, it is hard to anticipate effective regions for these edges in the graph.

There is no study available in literature to analyze the occurrence of vertices or edges during the shortest path traversals. In this paper, we provide an empirical analysis on which type of vertices are traversed during shortest path computation in social networks. First, we compute the shortest paths between set of vertices. Each shortest path is considered as one transaction. Second, we utilize the pattern mining approach to identify the frequency of occurrences of the vertices in all transactions. We also provide statistical analysis in terms of network properties, e.g. degree distribution, average shortest path, and clustering coefficient. The contributions of this paper are as follows: (1) Empirically prove that a significant amount of shortest paths are overlapped (2) The behavior of the overlapped regions in diverse networks, e.g. Scale free networks (3) The impact of hub-nodes on the shortest paths, e.g. What portion of the shortest paths are pass through the hub nodes or across dense regions (4) Visual analysis on the coverage of the entire graph through shortest paths.

2 Shortest Path Analysis

In this section, we highlight the process of finding frequency of vertex occurrences in real life social graphs. The overall architecture of the system is given in Fig. 1. The source graph data is usually available in the form of set of vertices and edges. There are two main components to find the shortest path
overlapped regions. Firstly, the graph traversal algorithm is used to find all pairs of shortest paths, i.e. $P$. Each path $p_i$ consists of sequence of vertices from source to destination. The graph traversal module produces set of shortest paths between all pair of source and destination as intermediate results. All the shortest paths are computed using well-known Dijkstra algorithm. Secondly, the overlapped regions of shortest paths are identified through pattern mining approach.

2.1 Pair-wise Shortest Path Computation

There are two possibilities to compute the shortest paths between pair of vertices in the social graph, namely exhaustive and non-exhaustive approach. The brute force or straight forward approach is to compute all possible pair of shortest paths from a vertex to another. If $N$ is the number of vertices in the graph then $N^2$ shortest paths are computed. Consequently, $N$ time’s single source shortest path (SSSP) algorithm is executed. A single pair shortest path computation in relational environment costs (5M, 15), (10M, 27), (15M, 33), and (20M, 41) in the unit of ($|N|$, sec) [2]. For a huge graph, we randomly choose $K$ vertices and compute $K$ times SSSP for efficient analysis. It is computation intensive process for even moderate size (medium or large scale) graphs. The time complexity growth is quadratic to the size of the graph, i.e. number of vertices. In order to compute shortest paths in reasonable time, a greedy approach is essential and preferred. For a huge graph, we limit the number of SSSP execution up to $K$. Therefore, instead all pair shortest paths, $K \times N$ number of shortest paths are computed where $K << N$. These sample shortest paths are further utilized for identifying the overlapped regions. The social networks are usually dense, follows the power law distribution, so even small number of shortest paths can lead us to better or acceptable analysis.

2.2 CORE Detection using Pattern Mining

The term continuous overlapped region (CORE) refers to the portion of the graph which is traversed through multiple shortest paths. In other words, it is a sequence of adjacent vertices or edges. Moreover, the shortest paths contributing for CORE are labeled as seed paths.
Figure 2: Visualization of the Facebook Network (a) original graph and (b) shortest path traversals.

Definition 2.1 \textit{CORE} A shortest path $p_i$ contained in a set of seed paths, i.e. $P_{\text{seed}} = \{p_{s1}, p_{s2}, \ldots, p_{sL}\}$, is defined as continuous overlapped region where $L \geq 2$, and $P_{\text{seed}}$ set of shortest paths.

The problem of detecting the COREs from set of shortest paths can be transformed into data mining field by considering each shortest path as a transaction and vertices refer to items. All the shortest paths are reflected as the transaction database. Pattern mining is one of the data mining approaches to find the existing patterns in the data. In this context patterns refer to vertex associations, behaviors, arrangements, or forms in the shortest path. The frequent occurrence of such patterns can lead us to the desired solution. This process of pattern detection is known as frequent pattern mining. There are various approaches in literature to mine frequent patterns. The examples are Apriori, AprioriTID, FP-Growth, Relim, Eclat, and H-mine. The FP-Growth method is efficient compared to other approaches [3]. Using FP-growth, we identify the frequency of one or more vertices in all transactions.

3 Results and Discussion

We have used two real life datasets, Facebook and Email network, from Stanford repository at http://snap.stanford.edu/. All the experiments are carried out on 32bit single machine running Windows 7. The proposed method is implemented using Java. Fig. 3 shows the degree distribution of the vertices along with the frequency of occurrences of vertices during shortest path traversals. Fig. 3(a) shows the original graph vertices on horizontal axis and the degree of each vertex is along vertical axis. It seems the Facebook graph follow power law distribution. We have plotted frequency of occurrences of each vertex along the vertical axis in Fig. 3(b) where very few vertices have
Figure 3: Shortest path analysis (a) degree distribution of vertices, (b) (c) (d) frequency of occurrence for one, two, and three vertices respectively during shortest path computation.

appeared in majority shortest paths. Similarly, the occurrences of more than one vertex are shown in Fig. 3(c) and Fig. 3(d). Intuitively, when we increase the pattern length, say 2 or 3 consecutive vertices, the number of patterns should decrease. In Fig. 3(c) and 3(d) the total numbers of two and three length patterns are three times smaller than one length patterns. We have visualized the Facebook network along with the shortest path trajectories in Fig. 2.

4 Conclusion

In this paper, we empirically analyze the shortest paths to anticipate the behavior of traversal algorithm on real life networks. A set of shortest paths are evaluated using pattern mining approach. We have found that the nodes with very high degree are retained in majority shortest paths. However, nodes with average degree are not considered by the traversal algorithm. The statistical analysis also shows the similar behavior in terms of network properties, including clustering coefficient, average shortest path, and betweenness centrality, on various types of networks. The influence of edge weights and directional information on shortest path traversal is still an interesting area of research.
Acknowledgements. This research was supported by the MSIP (Ministry of Science, ICT&Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (NIPA-2014(H0301-14-1020)) supervised by the NIPA (National IT Industry Promotion Agency).

References


Received: May 1, 2014