Proposal on Parrallel Betweenness Centrality Algorithm (ParBC)

Rui Qiu (rq2170), Hao Zhou (hz2754)

Nov 2021

1 Introduction

With the popularity of personal mobile devise, social applications have boosting populations of users. Among them, there is a huge amount of interactions constructing a vast social network that is worthy of further investigation. One task is to identify how central or important a single node (user) is. To evaluate it, many metrics and algorithms are reported including the one our project focus on called Betweenness Centrality. Betweenness Centrality is formulated by Freeman [1] as shown below.

$$Betweenness(k) = \sum_{i \neq k \neq j} (\frac{\sigma_{i,j}(k)}{\sigma_{i,j}})$$

Where $\sigma_{i,j}(k)$ is the number of shortest path between i, j that pass through the node k and $\sigma_{i,j}$ is the number of shortest path between i, j. This could reflect the significance or centrality of a node in a network. For example, in the context of social network, there are many groups. A node with high betweenness centrality might be the node in the center of the whole network, in the center of some big groups or a node act like a gateway (bridge) between two major groups.

However, in reality, the network is highly likely huge is size. This could be changing to apply Betweenness Centrality algorithm on those real-world data. Therefore, the project aims to provide a parallel implementation of Betweenness Centrality Algorithm to make the it efficient to perform centrality analysis in real-world social network.

2 Project Scope and Data

The project focus on undirected graph. To be more specific, It would target on the networks listed in SNAP [2]. We would first try our implementation on a github

network [3] where there are 37,700 nodes and 289,003 edges. Each node indicate a github developer with at least 10 repositories and each edge is established if there is a mutual follower between them. Although it is unweighted in nature, we would implement a more general one so that both weighted and unweighted network could be handled by our program. If time permits, we would test our project on even larger network in this social network data collection.

3 Implementation Plan

The algorithm is divided into two steps. First step is calculating pair-wise shortest distances and the second is quantifying the times of occurrence of a node on those shortest paths. Both steps could be implemented in a parallel fashion.

To calculate the pair-wise shortest path, it is natural to turn to Floyd–Warshall algorithm [4]. However, there might be two major limitation. First, its nature does not fit for the parallel implementation since the shortest path from i to j passing through $\{1..k+1\}$ depends on the result of the shortest path from i to j passing through $\{1..k\}$. This forces the threads to be synchronised and wait in this loop. Second, its complexity is $O(n^3)$ where n is the number of nodes. However, in reality, social network is edge-sparse but the algorithm could not take a good advantage of this feature. Therefore, we turn to Dijkstra's Algorithm [5] which is more suitable for edge-sparse cases and could calculate shortest path independently from one node to any others. In addition, there might be other shortest path algorithm we might try to implement in parallel fashion. They are Johnson's algorithm [6] and Brandes' algorithm [7]

To quantify the times of occurrence of a node on those shortest paths, we could use map-and-reduce fashion to perform this calculation in a parallel manner where, node id is the key and count is the value. This could make a good use of parallelism to calculate the final betweenness centrality for all the nodes in a networks.

4 Conclusion

The project aims to provide a parallel implementation of Betweenness Centrality Algorithm that is efficient for large real-world social networks. It will perform both shortest path and betweenness calculation in a parallel fashion and also make good use of the fact that most real-world social network is edge-sparse.

References

[1] L. C. Freeman, "A set of measures of centrality based on betweenness," *Sociometry*, pp. 35–41, 1977.

[2] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, Jun. 2014.

[3] B. Rozemberczki, C. Allen, and R. Sarkar, "Multi-scale attributed node embedding," 2019.

[4] R. W. Floyd, "Algorithm 97: shortest path," *Communications of the ACM*, vol. 5, no. 6, p. 345, 1962.

[5] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[6] D. B. Johnson, "Efficient algorithms for shortest paths in sparse networks," *Journal of the ACM (JACM)*, vol. 24, no. 1, pp. 1–13, 1977.

[7] U. Brandes, "A faster algorithm for betweenness centrality," *Journal of mathematical sociology*, vol. 25, no. 2, pp. 163–177, 2001.