ANALYSIS OF PARALLEL IMPLEMENTATIONS OF CENTRALITY ALGORITHMS [*]

Patricia D. Soriano, Kevin H. Amrein, Sam P. Carswell, and Michael O. Lam
James Madison University
Harrisonburg, VA 22807
540-568-3335
{sorianpd, amreinkh, carswesp}@dukes.jmu.edu, lam2mo@jmu.edu

ABSTRACT

This paper explores parallel implementations of three network analysis algorithms for detecting node centrality: betweenness centrality (BC), eigenvalue centrality (EC), and degree and line importance (DIL). All solutions were written in the C programming language using OpenMP library for parallelization. We evaluated these implementations for accuracy and parallel scaling performance using five example networks. We found that the algorithms accurately reflect different notions of centrality. While DIL performs better in general because it is asymptotically faster than the other two algorithms, BC demonstrates better parallel strong scaling.

INTRODUCTION

Networks are meaningful and prevalent in all fields of study. They represent the connections and interactions among components in any complex system, ranging from social relationships to chemical compounds. Analysis of networks provide rigorous insight into community structures that may otherwise be overlooked. The algorithms here aimed to discover nodes that play a critical or "central" role in the dynamic processes within a network. Centrality varies according to the network being analyzed. Within a network that contains a single community, the node with the most connections (a "core" node) may be considered to be most central. In contrast, analysis of a network with multiple communities may conclude that the central node is the one which connects the most communities (a "bridge" node). Therefore, the notion of centrality depends on the problem domain.

Background

The first algorithm, Betweenness Centrality (BC) [1], finds critical nodes by calculating the "betweenness" of each node. The betweenness of a node is calculated by finding the shortest path from every node in a network to every other node in that network and counting how many times the shortest paths used the node in question. The node with the highest betweenness is the most critical node in the network. This algorithm excels at identifying nodes that serve as bridges between communities; although a bridge node may not have the highest number of connections overall, BC identifies it as the most critical because it is used in the shortest path from every node in one community to every node in another.

The second algorithm, Eigenvalue Centrality (EC) [2], returns the node with the most change in eigenvalue computation upon its removal from the graph. It is asymptotically slower than BC and DIL because eigenvalue calculations are O(n^3) with respect to the network size n.

EC views the graph as an adjacency matrix. An adjacency matrix translates nodes as rows and columns and encodes node interactions as binary values, where a pairwise connection is denoted as a one and its absence as a zero. This adjacency matrix can be converted to coefficients in a set of equations, after which eigenvalues are computed to determine the most important solution for those equations. For each node, an adjusted adjacency matrix is generated to exclude that node by replacing it with zeros. Whichever node produced the greatest change in eigenvalue estimation had the most impact on the set of equations, establishing said node as critical.

The equation in Fig. 1 describes this process, where c is the number of communities, γ is the eigenvalue, k is the current node index, and P_k is the approximate eigenvalue centrality:

$$P_k = -\sum_{i=1}^{c} \frac{\Delta\lambda_{\cdot i}}{\lambda_{\cdot i}},$$

Fig 1. Eigenvalue Centrality Formula

Finally, Degree and Line Importance (DIL) [3] is a ranking method that finds the most important node by considering the degree of each node and the importance of its edges. These factors allow the method to distinguish between nodes of the same degree as well as identifying a bridge node when present. The equation in Fig. 2 finds the importance of each node (L_{vi}) while the equation in Fig. 3 describes how to find the node's contribution to its edge ($W_{vi\,vj}$).

$$L_{v_i} = k_i + \sum_{v_j \in \Gamma_i} W_{v_i v_j},$$

Fig 2. DIL Node Importance Formula

$$W_{v_i v_j} = I_{eij} \cdot \frac{k_i - 1}{k_i + k_j - 2},$$

Fig 3. DIL Node Contribution Formula

The importance of each node (L_{vi}) is calculated by adding the degree of that node and the sum of the node's contribution to its edges. The node's contribution ($W_{vi\,vj}$) to edge importance is based on its degree relative to the node it is connected to. The importance of the edge (I_{ej}) is determined by the degrees of the nodes that it connects. Therefore, an edge with significant importance will connect nodes with high degrees, thereby having a higher influence on connectivity relative to other edges in the network. This algorithm finds the node that has the highest connectivity and the greatest impact on the connectivity of other nodes within the network.

Project Goals

Our objective was to implement and analyze serial and parallel solutions for the three approaches discussed above in order to evaluate and compare centrality detection accuracy as well as performance based on execution times. Accuracy is defined here as the implementation's ability to find either of the critical nodes (the "core" node or "bridge" node).

METHODOLOGY

The implementations are parallelized by adding OpenMP's parallel "for" pragma (a code annotation) on the for loop that does the calculations for each node. This pragma divides the node calculations between multiple threads. The "critical" pragma is also added to prevent race conditions when comparing the current and max values for replacement if necessary. We use the IGraph library to store and manipulate the graphs and adjacency matrices and to perform betweenness and eigenvalue calculations.

To test accuracy, we used these three datasets: Zachary's Karate Club [4], Dolphins' Social Network [5], and Les Misérables Network [6]. We used Zachary's Karate Club as the first case because it is clear which nodes are important. Fig. 4 [7] represents the graph. The two important nodes that the implementations should find are node 0 (the instructor of the karate club) and node 33 (the president). The graph represents the patterns of friendship between people in the club. The next dataset encodes frequent associations between 62 dolphins in a community located in New Zealand. There are 159 edges total in the dataset. Lastly, we used the Les Misérables Network, containing characters from Victor Hugo's novel. Each node is a character and an edge between two nodes means that those two characters appeared in the same chapter together. There are 254 edges total.

Betweenness and eigenvalue centrality have been studied extensively, including Batool's and Niazi's research of validating centrality measures [8]. We thus evaluated the accuracy of our implementations by comparing with their results. Due to the lack of ground truth for degree and line importance, we evaluated our implementation on its agreement with the other two methods as well as by manual examination of the graph.

We tested performance using the Les Misérables Network and two additional datasets: Facebook Social Network [9] and Scientific Collaboration Network [10]. The Facebook Social Network consists of 347 nodes and 5038 edges collected from Facebook data. Lastly, the Scientific Collaboration Network portrays a real-life network of scientists who collaborated together on network theory and experiment, compiled by Mark Newman with a total of 1589 nodes and 2742 edges.

RESULTS

Accuracy Tests

For Zachary's Karate Club, EC outputs node 33 (president) while BC and DIL output node 0 (karate instructor). EC identifies the president as having the highest eigenvalue, the person most connected with the other connected members in the club. Compared against Batool's and Niazi's research [8], our implementations produce the correct result for this dataset. Because a "bridge" node is explicitly present for this dataset, DIL identifies the karate instructor as the most important despite having fewer connections (lower degree value) than the president. As the "bridge" node, the instructor is the path of information between the other members.

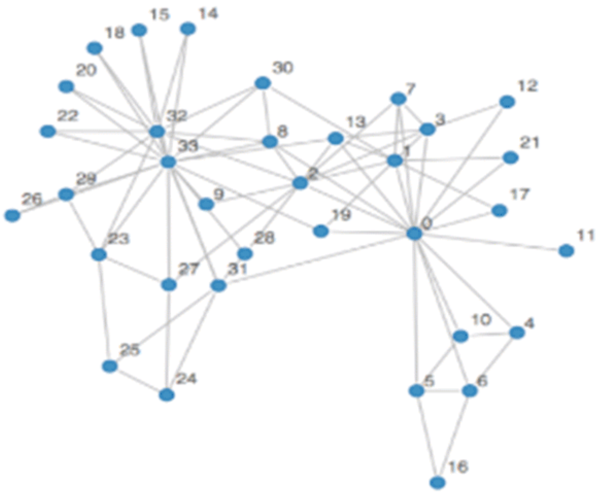

Fig.4. Graph of Karate Club Network

For the Dolphins' Social Network, EC and DIL identify node 14 (Grin) as being the most important, while BC identifies node 36 (SN100). Fig. 5 [8] shows Grin (middle node) having the highest eigenvalue while Fig. 6 [8] shows SN100 having the highest betweenness. DIL identifies node 14 (Grin) as most important because it is the node with the highest degree. Since the network is composed of only one community, the "bridge" node (which according to BC is SN100) is not found.

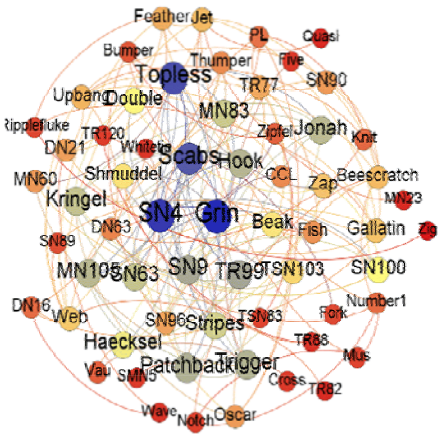

Fig 5. Eigenvalue Centrality

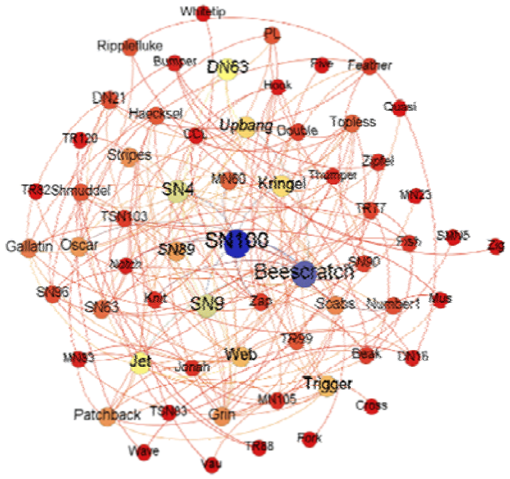

Fig 6. Betweenness Centrality

The Les Misérables network has multiple communities, as seen in Fig. 7 [11]. Each community is represented in a different color. EC identifies node 48 (Gavroche) as the most critical while BC and DIL identify node 11 (Valjean). Valjean has the most connections in the graph and Gavroche has the second highest. Although Gavroche has a lower degree value, EC identifies it as important because it has more "influential" connections in the graph as a whole. BC and DIL identify Valjean because he is the clearly defined "bridge" node. He is the one who connects other unacquainted characters in the novel. Findings from Alvarez-Soccoro's research on eigencentrality [12] further confirmed the correctness of our BC and EC implementations.

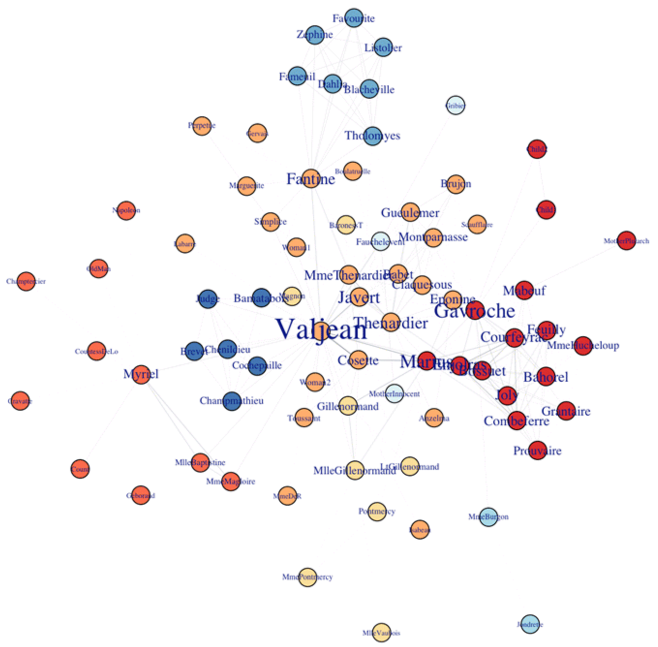

Fig 7. Graph of Les Misérables Network

Performance Tests

Figures 8-10 show execution times of Les Misérables, Facebook Social Network, and Scientific Collaboration datasets in our scaling experiments.

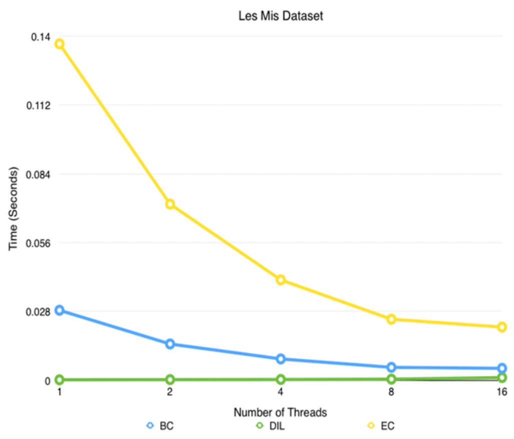

Fig 8. Les Misérables Network Execution Times Graph

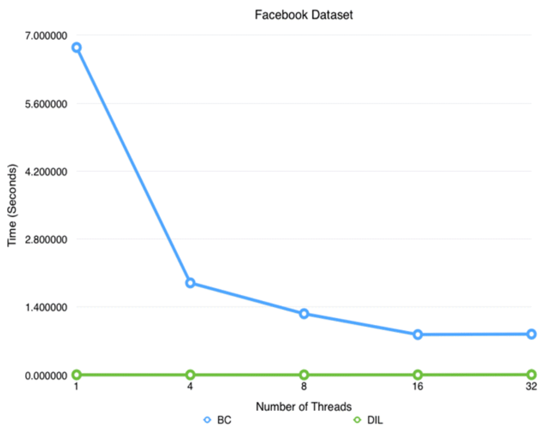

Fig 9. Facebook Social Network Times Graph

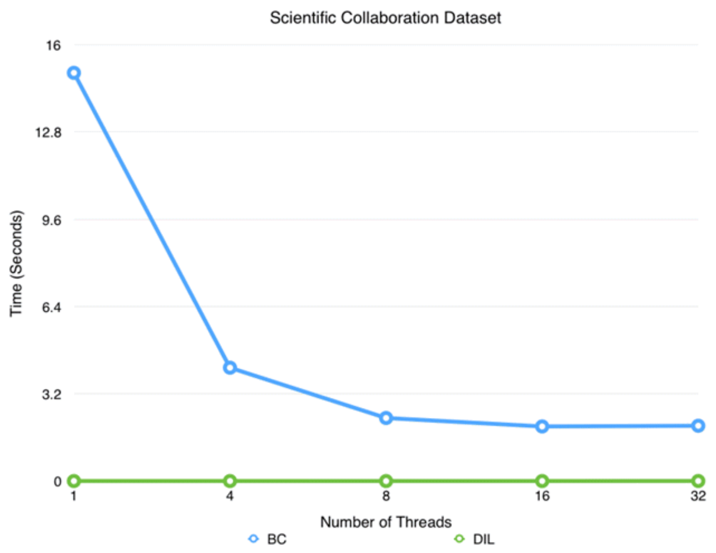

Fig 10. Scientific Collaboration Network Execution Times Graph

EC shows strong scaling when run with the Les Misérables network, which means that execution time decreases as the number of threads used to run the program increases. However, it does not exhibit the same property when run with other datasets. It is also considerably slower overall compared to BC and DIL due to the computational complexity of calculating the eigenvalues; the algorithm's overall time complexity is $O(n^4)$ where n is the number of nodes in the network. We have not included EC in the scaling graphs for the last two datasets due to its long execution times.

BC shows strong scaling consistently across all three datasets. Compared to the results from the Facebook Social Network (Fig. 9), BC runs slower with the Scientific Collaboration Network (Fig. 10) because the latter dataset has more nodes and betweenness calculations have a time complexity of $O(n*e)$, where n is the number of nodes in the graph and e is the number of edges. Because these calculations must be done for every node, the algorithm has an overall time complexity of $O(n^2*e)$. However, the algorithm parallelizes well because the betweenness calculations are independent and work can be balanced equally among the threads.

DIL runs faster than BC over all three datasets because it uses local information instead of global to calculate each node's importance. It only needs the degrees of the current node and its neighbors. Thus, DIL has an overall time complexity of $O(n*d)$, where n is the number of nodes in the graph and d is the maximum degree of any node. Unlike BC, there are no pairwise node calculations. When parallelized, DIL showed very little scaling, meaning the increased number of threads have little to no impact on execution time. Due to the simplicity of the calculations, the creation and destruction of threads negates the benefits of parallelization.

CONCLUSION

Of the three algorithms we studied, DIL provided superior performance and is able to identify critical nodes even in networks where the number of distinct communities is unknown. However, this approach is not helpful when you want to always identify a specific type of centrality ("core" or "bridge") regardless of the community structure. Comparatively, BC is more consistent in finding "bridge" nodes in multiple communities, and EC finds the "core" node. EC also demonstrated the slowest execution times and produced a non-critical node when run with the large Scientific Collaboration Network. Thus, EC should only be used for small, single-community networks.

These results demonstrate that our implementations are able to capture the different notions of centrality depending on the algorithm, and at least one of our implementations (BC) demonstrates excellent strong scaling. Future work could include further optimization as well as development of new hybrid network analysis algorithms combining the strengths of multiple existing algorithms.

REFERENCES

[1] Girvan, M. and Newman, M.E., Community structure in social and biological networks, *Proceedings of the national academy of sciences*, 99 (12), 7821-7826, 2002.

[2] Wang, Y., Di, Z. and Fan, Y., Identifying and characterizing nodes important to community structure using the spectrum of the graph, *PloS one*, 6 (11), e27418, 2011.

[3] Liu, J., Xiong, Q., Shi, W., Shi, X. and Wang, K., Evaluating the importance of nodes in complex networks, *Physica A: Statistical Mechanics and its Applications*, 452, 09-219, 2016.

[4] Zachary, W.W., An information flow model for conflict and fission in small groups, *Journal of Anthropological Research*, 33, 452-473, 1977.

[5] Lusseau D., Schneider K., et.al., *Behavioral Ecology and Sociobiology* 54, 396-405, 2003.

[6] Knuth D. E., *The Stanford GraphBase: A Platform for Combinatorial Computing*, Addison-Wesley, Reading, MA, 1993.

[7] Understanding Networks, Log 3: Importing Networking Data, 2010 https://understandingnetworks.wordpress.com, retrieved April 27, 2017.

[8] Batool, K. and Niazi, M.A., Towards a methodology for validation of centrality measures in complex networks. *PloS one*, 9 (4), e90283, 2014.

[9] McAuley J. and Leskovec J., *Learning to Discover Social Circles in Ego Networks*. NIPS, 2012

[10] Newman M. E.J., *Phys. Rev.* E 74, 036104, 2006.

[11] Ren, X., L06 Discussion: Visualization, Les Mis Network Graph, 2015, https://rstudio-pubs-static.s3.amazonaws.com/115658_4bb31c520bab4389bf0f8bdab6b27d36.html, retrieved April 27, 2017.

[12] Alvarez-Socorro A.J., Herrera-Almarza G.C., González-Díaz L.A., Eigencentrality based on dissimilarity measures reveals central nodes in complex networks. *Scientific Reports*, 5, 17095, 2015