Graph Centrality Estimation Using Pivots

Social Network Analysis for Computer Scientists — Course Project Paper

Andreas Papagiannis
LIACS, Leiden University
and.papagiannis@gmail.com

Konstantinos Gkosios
LIACS, Leiden University
kgkosios95@icloud.com

ABSTRACT

Centrality is a very important term in network analysis. As the size of the data is increasing rapidly every year the average centrality computation is impracticable, since usually we have to solve a single-source-shortest-paths (SSSP) problem for every node in a graph. This is a time-consuming process. Therefore, the average computation requires a lot of time to compute the centrality of a network especially in large graphs. We here present, a more efficient way and its strategies to estimate centrality from a limited number of SSSP computations. At the we will examine the quality of those estimates by presenting some results.

Keywords

Graph centrality estimation, Pivot selection strategies, Closeness centrality, Betweenness centrality, Single-source-shortest-path

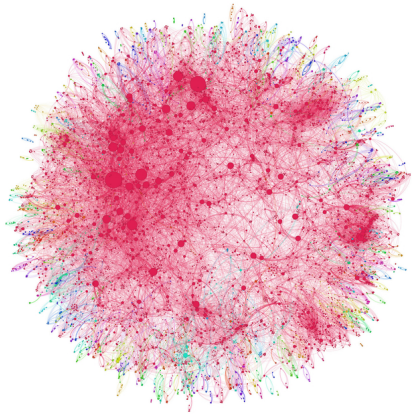

Figure 1: Computing time for closeness and betweenness centrality is n^2 where n is the number of nodes in a network. In such large networks as shown in this figure the exact computation is not affordable.

SNACS '18 Social Network Analysis for Computer Scientists, Master CS, Leiden University (liacs.leidenuniv.nl/~takesfw/SNACS).

1. INTRODUCTION

In the analysis of complex networks, centrality is an essential concept. The problem of identifying the most central nodes has been a fundamental question in many different fields, such as psychology, sociology, computer science and many more. Two of the most widely used centrality measures are closeness and betweenness centrality. Those two centrality measures are based on shortest path distances. We usually, have to solve a single-source-shortest-path problem for every node. Meaning that we have to find the shortest paths from a source node to all other nodes in the graph.

Since the size of the gathered data in electronic form is increasing rapidly every year, the need of the average computation of centrality has been increased. Although, this process is very simple and efficient for small size networks, for larger sizes it is a time-consuming process. The required time is at least quadratic in number of nodes, which is not affordable or in any way time efficient. Therefore, it is important to determine a more efficient method to accurately estimate centrality in large networks. We here present an experimental study by providing a better method and its strategies. The estimators are based on restricted number of single-shortest-paths computations from a set of selected **pivots**.

Pivots are selected according to some strategies for which we discuss more in Section 4. All strategies select k unique pivots $p_1, p_2, ..., p_k \in V$, such that the results from solving an SSSP problem from every pivot are similar to those from every node in V. In the following strategies, experiments have already been made in similar works.

In this paper, we propose and experiment in some new pivot selection strategies which are based on their page rank value. More details are provided in Section 4.

This paper is organized as follows. The problem and some basic details for the centrality computation is presented in Section 2. Related and previous work that has been made for this topic is shown in Section 3. In Section 4, the strategies and some algorithms for the approximate computation of centrality are discussed. The type of graphs and some more details about them are presented in Section 5 and in Section 6 the results from the experiments we made are discussed throughout. Finally, we conclude in Section 7 listing all of our observations.

2. PROBLEM STATEMENT

In the following paper we will consider a graph $G(V, E)$, where V is a set of nodes, and E is a set of edges. In particular, we restrict ourselves to work with simple graphs. Meaning that our graphs are undirected, connected and unweighted. Furthermore, $n = |V|$ denotes the number of nodes and $m = |E|$ the number of edges. We define **path** as an alternating sequence of nodes and edges. Also, **path length** is the number of edges in a path. In our case we do not include a paths with infinite path length formed from graph cycles.

A graph is connected if there is at least one path between every pair of nodes. Two nodes $v, u \in V$ are connected, if there is a path that starts from node v ending at node u.

The distance $d(s, t)$ between two nodes $s, t \in V$ is defined as the number of edges in a shortest path connecting them. The biggest distance between any pair of nodes in a graph is called diameter d.

In this paper, we work with two of the most popular shortest-path centrality measures, closeness and betweenness centralities for every node in undirected, connected and unweighted graphs.

Closeness Centrality

A node is central if the average number of edges needed to reach another node is small. Thus, the more central a node is considered, the closer it is to all the other nodes. Moreover, a central node should be very efficient in spreading information to all other nodes in a graph. In a connected graph, closeness centrality of a node $C_C(v)$ is defines as,

$$C_C(v) = \frac{n-1}{\sum_{u=1}^{n-1} d(v,u)}. \quad (1)$$

where n is the number of nodes that can reach u and $d(v, u)$ is the distance between nodes v and u.

Betweenness Centrality

Betweenness centrality ranks the nodes according to their participation in the shortest paths between other node pairs. Nodes with a high betweenness centrality value are considered as bridge nodes of the network. The betweenness centrality of a node $C_B(v)$ is defined as,

$$C_B(v) = \sum_{s,t \in V} \frac{\sigma(s,t|v)}{\sigma(s,t)}. \quad (2)$$

where $\sigma(s, t)$ is the number of shortest paths that connect nodes s and t and $\sigma(s, t|v)$ is the number of shortest paths connecting nodes s and t but also include node v.

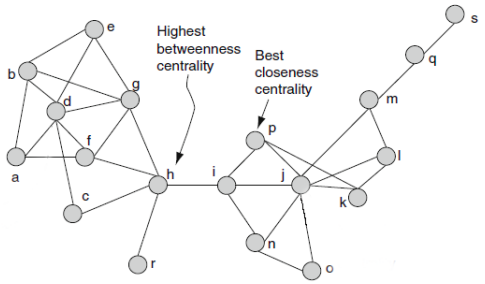

Figure 2: Illustration of the closeness and betweenness centrality measures in a small graph.

For small-world networks which we define as those for which $(L \propto logn)$, where L is the distance between two nodes and n the number of nodes. In such graphs, most nodes can be reached from every other node by a small number of steps. The closeness centrality can be well computed by solving a single-source shortest-path (SSSP) problem for every node. To compute the centrality we sum up in each iteration all the distances found from the source node. An efficient way to such computation is by using standard breadth-first search where the total running time is $O(nm)$.

Although, the computation of closeness centrality is straightforward, for betweenness centrality it is more complicated. In this case, we don't have to compute lengths but number of shortest path between pairs of nodes. However, we have again to solve an SSSP problem and in fact we use the same algorithm as we did for closeness centrality computation. Thus, the two centrality measures has the same time complexity.

For large-world graphs, the average computation of centralities as described above is very expensive in term of time and space as the running time is $\Omega(n^2)$.

The exact computation require to solve n single-source shortest-paths problems, for every node and then to sum them all up. We define as **pivots** the nodes that a single-short shortest-paths problem is solved. The main idea of this paper, is to estimate the centrality value from a small number of SSSP computations, i.e. from a small number of pivots.

3. RELATED AND PREVIOUS WORK

Fast Approximation of Centrality [2] is an article published in 2004 and written by David Eppstein and Joseph Wang. It mathematically proves that the exact centrality value of a graph can be estimated by extrapolating the contributions obtained from just a few SSSP computations. It also introduces and algorithm called `RAND` which estimates closeness and betweenness centrality values by using pivots randomly selected. However, the algorithm was not tested in practice and thus no results were presented.

Centrality Estimation in Large Networks [1] is an article published in 2006 written by Ulrik Brandes and Christian Pich and should be thought as an extension to the previous article mentioned. In their work they added more pivot

selection strategies and tested them in a number of graphs which by modern standards would not necessarily be considered as large graphs. Also, their results unfortunately lack any numerical comparison of the pivot selection strategies.

In this paper, we add some pivot selection strategies based mostly on the page rank value of each node of the graph. We test our own as well as all strategies included in *Centrality Estimation in Large Networks* [1] in larger graphs than before and lastly, we introduce a way to numerically compare all strategies in order to find which strategy converges faster to the real value that is estimated.

4. SUGGESTED APPROACH AND ALGORITHMS

Centrality values computation consists of solving n single-source shortest-paths problems (SSSP), one for each vertex, and each SSSP contributes one summand to the result. This contribution is the distance to the source for closeness, and the one-sided dependency of the source for betweenness. The vertices for which an SSSP is solved are called *pivots*. Based on an idea put forward by Eppstein and Wang [2], the exact centrality value can be estimated by extrapolating the contributions obtained from just a few SSSP computations.

Table 1: Pivot selection strategies

Strategy	Selection rule
`Random`	uniformly at random
`RanDeg`	random proportional to degree node value
`RanPgRank`	random proportional to page rank node value
`Degree`	maximize degree node value
`pgRank`	maximize page rank node value
`pgRankRev`	minimize page rank node value
`pgRankAlt`	alternate `pgRank` & `pgRankRev`
`MaxMin`	maximize minimum distance to previous pivot
`MaxSum`	maximize sum of distances to previous pivot
`MinSum`	minimize sum of distances to previous pivot
`Mixed3`	alternate `MaxSum`, `MinSum`, and `Random`

Table 1 contains all pivot selection strategies tested. Strategies `Random`, `RanDeg`, `MaxMin`, `MaxSum`, `MinSum` and `Mixed3` are strategies already tested and explained throughout in *Centrality Estimation in Large Networks* [1]. To select pivots they mainly calculate the degree value of each node as well as distances from the previously selected pivot.

Strategies `Degree`, `pgRank`, `pgRankRev`, `pgRankAlt` as well as `RanPgRank` are our own contribution. `Degree` selects pivots by maximizing each nodes degree value. In other words, if we choose to select five pivots to estimate the desired centrality measures, those pivots will be the five nodes having the highest degree of the whole graph. Similarly, `pgRank` selects pivots by maximizing each nodes page rank value. `pgRankRev` selects pivots by minimizing each nodes page rank value. Nodes with high degree or page rank value are also expected to score high in closeness and betweenness centrality values. Thus maximizing these values as done in `Degree` and `pgRank` is expected to overestimate both closeness and betweenness values when using pivots. Of course, `pgRankRev` is expected to underestimate these values for the same reason. In order to bypass this problem, we introduce `pgRankRev` which alternates between `pgRank` and `pgRankRev`. `RanPgRank` chooses pivots with a probability proportional to each nodes page rank value and works in the same fashion of `RanDeg`. In a way, one can think it as selecting pivots using stratified sampling depending on the page rank value of all nodes.

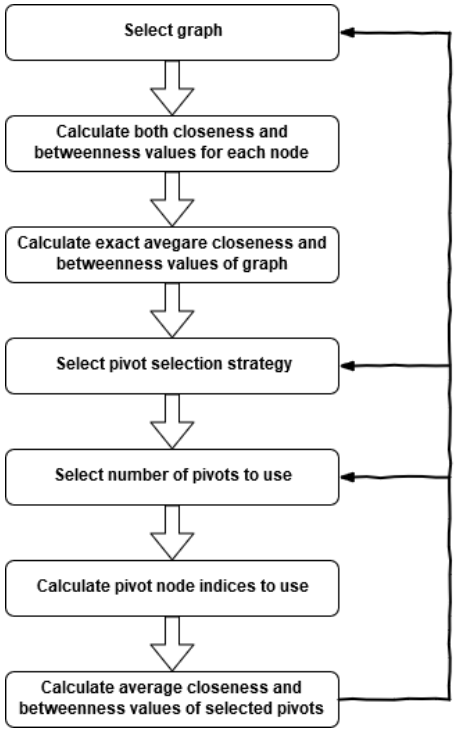

Figure 3: Experimental Pipeline

Figure 3, explains the implemented experimental pipeline. At first, we iterate through all graphs and calculate the average closeness and betweenness centralities by finding the respective values of each node in the graph. Then, we iterate through all pivot selection strategies. Having set a selection strategy, starting with one pivot, we keep adding pivots until all nodes of the graph are selected as pivots. Having set a number of pivots, we extract the node indices that were selected as pivots depending on the current selection strategy. Having selected the nodes indices, we then average their closeness and betweenness centrality values, which were calculated in the beginning. This last averaging is our estimation of average closeness and betweenness for the respective graph using the current selection strategy and number of pivots. In that way we end up with estimations for all possible number of pivots as well as for all pivot selection strategies.

The exact code used in order to output any results as well all results produced can be found in the following Github repository: https://github.com/dru93/Graph-centrality-estimation

5. DATASETS

The following table contains all graphs that have been tested so far. The two first graphs are randomly generated and the third one is a graph with information over the power grid of the US and it can be considered as a road-map network. Erdos Renyi graphs are sought to have a binomial node degree distribution. On the other hand, Barabasi Albert graphs nodes follow a power law degree distribution. Also, Barabasi Albert graphs are generated by a preferential attachment process and resemble tree graphs. All of these graphs are connected, however, we could also use unconnected graphs as long as we calculate the average closeness and betweenness values for their giant component instead of the whole graph. What these graphs also share, is that all of them are undirected and their edges have no weights as mentioned before.

Table 2: Datasets tested

Graph	# of nodes	# of edges
Erdos-Renyi	10.000	29.972
Barabasi-Albert	10.000	9.999
US power grid	4.941	6.594

All three of these graphs have a number of differences concerning their properties such as their degree distribution mentioned earlier. We examine all of them as certain pivot strategies may outperform the rest strategies in specific graph types.

Figure 4: Power grid graph visualization

6. EXPERIMENTS AND RESULTS

Framework

Listed as follows, are the programming languages and libraries used in this project.

- `python3`
- `R`
- `graph-tool` [3]
- `ggplot2` [4]

`Python3` is used in order to generate the data (all graphs discussed previously) as well as calculations such as the centrality closeness and betweenness values and the pivot selection strategies. `R` is used in order to plot and numerically compare all outputted results.

Results

Figures 5-7 show how well each pivot selection strategy performed in the graphs we tested. the straight black horizontal line always represents the real value estimated (closeness or betweenness centrality). Each different strategy is depicted with a differently colored line as shown in the legend of each figure. Y-axis is always scaled in the `log10` scale in order to focus on small number of pivots.

All pivot selection strategies including calculation of distances between nodes (namely `MaxMin`, `MaxSum` and `MinSum`) were omitted when estimating centrality measures for networks with more than 20 pivots as they are too computationally intense. Thus, these selection strategies are depicted only in Figure 5 were the tested graph has only 10 nodes.

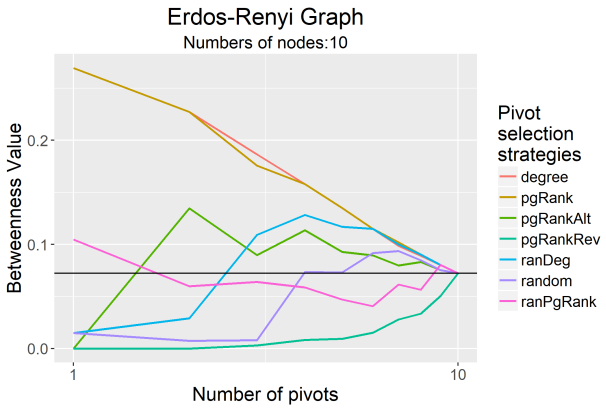

Figure 5: Erdos-Renyi Graph with 10 nodes betweenness centrality approximation using pivots

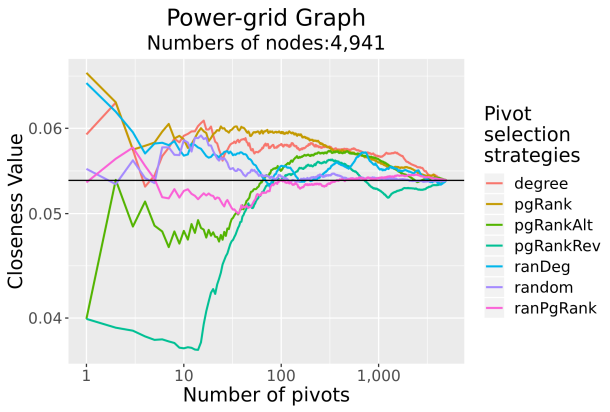

Figure 6: US Power Grid Graph with 4941 nodes closeness centrality approximation using pivots

In all three Figures 5-7 we observe that pivot strategies that select pivots by maximizing another centrality measure (namely `pgRank` and `Degree`) initially overestimate both closeness and betweenness values.

Another small observation is that only when estimating the closeness value of a Barabasi-Albert graph as shown in Figure 7, no strategies except `Random` seem to underestimate the exact closeness value of the graph.

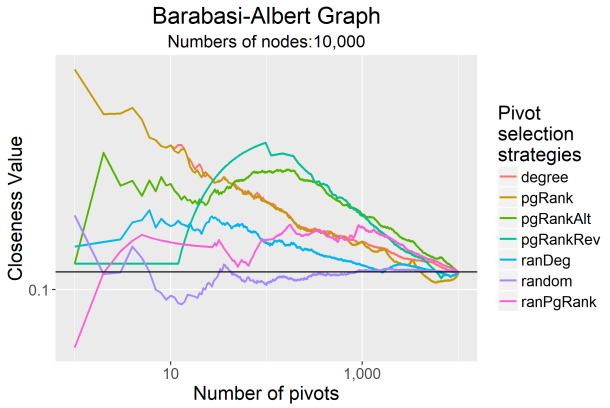

Figure 7: Barabasi-Albert Graph with 20 nodes betweenness centrality approximation using pivots

Generally speaking, all pivot selecting strategies converge fast to the true closeness or betweenness value. After performing t-tests using only 50 pivots, for nearly all pivot selection strategies, the estimation was statistically not different than the true value ($a = 0.05$).

Comparison of pivot selection strategies

In order to compare the results and specifically find out how each pivot strategy performs in estimating the true value, we used the following simple measure:

$$10^4 * \sum_{k=1}^{\#\text{ of nodes}} |\text{estimation using } k \text{ pivots} - \text{real value}| \quad (3)$$

This measure is basically the sum of the differences between the real value and all estimations using from one to all nodes. The only use of the multiplier in from of the sum is to move the decimal of the measure to the left, in order to make the numerical comparison easier.

Tables 3 and 4 show the value of the aforementioned measure for each strategy and all graphs rounded so that only the first two decimals appear. So in that way we can compare all pivot selection strategies and evaluate which perform better.

Table 3: Differences in closeness centrality

	Erdos-Renyi	Barabasi-Alb.	Power-grid
`Random`	0.91	4.67	1.17
`RanDeg`	22.87	20.97	6.94
`RanPgRank`	20.02	50.11	5.12
`Degree`	73.37	44.15	18.12
`pgRank`	69.77	43.66	13.89
`pgRankRev`	78.96	76.01	9.79
`pgRankAlt`	11.33	97.44	9.82

In both tables, `Random` pivot selection strategy is closer to the real value with a big difference than the second closer strategy in all cases. Strategies maximizing or minimizing some centrality value (namely `Degree`, `pgRank` and `pgRankRev`) perform poorly as expected since they either overestimate or underestimate the real value. Another small observation here is that while `pgRankAlt` strategy seems to perform good in general compared to the rest strategies, when it comes to estimating the values of a Barabasi-Albert graph it is one of the worse if not the worse strategy.

Table 4: Differences in betweenness centrality

	Erdos-Renyi	Barabasi-Alb.	Power-grid
`Random`	0.03	1.61	3.38
`RanDeg`	0.33	7.76	20.78
`RanPgRank`	0.67	9.09	16.71
`Degree`	3.35	37.83	45.90
`pgRank`	3.30	37.61	40.94
`pgRankRev`	2.41	7.98	26.01
`pgRankAlt`	1.04	31.45	17.99

7. CONCLUSION

Depending on the selection strategy, pivots may get as computationally intense (or even more) as calculating the closeness and betweenness values of each node. The total pivot selection strategy runtime should never be greater that the runtime of calculating the except closeness or betweenness values.

`Random` pivot selection seems to outperform all pivot strategies in performance. This is probably happening because the underlying condition under the assumption that pivots

can produce an accurate estimations, is that the pivot nodes themselves are independent from each other. This condition is fulfilled only when using the random pivot selection strategy as all other strategies calculate measures such as distance between pivots, degree or page rank values before extracting the pivots and thus the independence between pivots condition does not hold. In other words, as the authors of *Centrality Estimation in Large Networks* [1] put it, "*[...] structural imbalance present in most networks cause deterministic strategies to run into traps [...]*". This means that the only pivot selection strategy that should be used is selecting pivot at random. Also, last point to be made, is that there is no need to select a large number of pivots, as shown above, only a small subset of the nodes will accurately estimate both closeness and betweenness centrality values.

8. REFERENCES

[1] U. Brandes and C. Pich. Centrality estimation in large networks. *International Journal of Bifurcation and Chaos*, 17(7):2303–2318, 2007. http://algo.uni-konstanz.de/publications/bp-celn-06.pdf.

[2] D. Eppstein and J. Wang. Fast approximation of centrality. *Journal of Graph Algorithms and Applications*, 8(1):228–229, 2004. http://www.emis.ams.org/journals/JGAA/accepted/2004/EppsteinWang2004.8.1.pdf.

[3] T. P. Peixoto. The graph-tool python library. *figshare*, 2014. http://figshare.com/articles/graph_tool/1164194.

[4] H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2009. http://ggplot2.org.