Approximating Betweenness Centrality using Random Sampling from k-cores

Hankyu Jang
Department of Computer Science
University of Iowa
hankyu-jang@uiowa.edu

Abstract

One of the well-studied topics in network science is finding central nodes in a graph that are computed using centrality measures. Betweenness centrality, which is based on shortest-path distances, is one of the widely used centrality measures. However, the exact computation of betweenness centrality is not feasible for large networks due to the heavy computation time. We present an approach to approximate betweenness centrality by randomly sampling pivot nodes from communities that are detected by the k-core approach.

1 Introduction

Centrality indices are used to find central nodes in a graph. A node is regarded as central if it is easily reachable from other nodes, frequently in paths between other nodes, affect the graph when deleted, or has a recursive function of actor centrality [1]. Betweenness centrality is calculated using the shortest paths of all the nodes in the graph. Nodes that frequently appear within the path of other nodes have high betweenness centrality.

Different models have been proposed to measure the betweenness centrality. The most widely used betweenness centrality measure is proposed by Freeman [2], where he assumed that information only spreads through the shortest paths. Later, Newman [3] relaxed this assumption by calculating betweenness centrality of essentially all paths between nodes by using random walks. Recently, another betweenness centrality measure [4] emerged to fill a gap between Freeman's centrality measure based on shortest paths and Newman's approach based on random walks.

Throughout the paper, we restrict betweenness centrality as Freeman's approach, only taking account of shortest paths. As the network grows, the running time of computing betweenness centrality measure increases rapidly. Freeman's algorithm takes $\theta(n^3)$ in time and $\theta(n^2)$ in space. Later, Brandes [5] improved the running time of Freeman's algorithm for sparse graphs. Brandes's algorithm requires $O(n+m)$ in space and run in $O(nm)$ and $O(nm+n^2 \log n)$ time on unweighted and weighted graphs respectively, which is the state-of-art implementation of exact betweenness centrality calculation.

Running time of computing betweenness centrality depends on the number of nodes and edges, meaning that the number of single-source shortest path computations for nodes increases rapidly as graph size becomes large. Hence researchers have focused on sampling a small set of pivot nodes when calculating shortest paths to reduce the number of single-source shortest path computations. In this paper, we propose a novel approach to use k-core approach to first detect community structures, and then use the information to efficiently sample pivot nodes.

2 Related Works

Eppstein and Wang [6] inspired the use of a small set of pivot nodes to approximate betweenness centrality. Later Brandes and Pich [7] stated that the approximation of betweenness centrality using pivot nodes that are selected uniformly at random was superior to that of more sophisticated, deterministic selection strategies. They showed that the accuracy of a random pivot selection strategy was monotonic in the number of pivot nodes whereas some deterministic strategies ran into traps at some point showing that adding more pivot nodes worsened the centrality estimation. Bader et al. [8] proposed an approximation algorithm based on adaptive sampling technique that

reduces the number of single-source shortest path computations for nodes with high betweenness centrality measure by keeping track of partial contribution of each sampled nodes.

Later, Kourtellis et al. [9] proposed an algorithm that improved accuracy in detecting high betweenness centrality nodes compared to previous approaches. [9] introduced a new metric κ-path centrality which is based on the random walk implementation of Newman [3]. Kourtellis et al. [9] made two changes in Newman's approach by assuming the length of the path to be κ, which is a parameter dependent on the network, and not letting random walk on visited nodes. Kourtellis et al. [9] presented the percentage overlap of the top-N % nodes of different approximation algorithms to the exact betweenness centrality measures. The κ-path centrality outperformed Brandes and Pich [7] and Bader et al. [8] for approximating top 1% of the nodes with high betweenness centrality.

Recently, there had been some works that used clustering technique to approximate betweenness centrality. Suppa and Zimeo [10] detected clusters of a graph using Louvain method which is based on a heuristic method for community detection. After identifying clusters in the graph, they classified nodes based on topological characteristics and selected one node for each class as pivot nodes. Later, Furno et al. [11] tested the algorithm of Suppa and Zimeo [10] on road networks and showed that the algorithm was well suited for the graph.

3 Background

3.1 Graph

Let $G = (V, E)$ represent a *graph*, where V is a set of *nodes* and E is a set of *edges* that connect nodes. Let $n = |V|$ denote the *number of nodes* in V and $m = |E|$ denote the *number of edges* in E.

A *path* is a sequence of edges that connect *nodes* $s, t \in V$. The *length* of a path is the number of edges in the path. Two nodes $s, t \in V$ are connected if there exists a path between them.

Shortest path of $s, t \in V$ is a path between s and t where the length is minimum among all possible paths between s and t. Note that there could be more than one shortest path between s and t. Let σ_{st} denote the *number of shortest paths* between s and t. For $v \in V$ let $\sigma_{st}(v)$ denote the number of shortest paths between s and t that pass through t. Note that $\sigma_{st}(v) = 0$ if $v \in \{s, t\}$.

The *k-core* [12] is the maximal subgraph of G where each node in k-core is connected to at least k nodes in the subgraph which can be disconnected. k-cores are nested, meaning that $i+1$-core is a subgraph of i-core. The *degeneracy* of G is the maximum coreness of the graph denoted as k_{max}.

In this paper, we restrict G as an unweighted, undirected graph with no self-loops and multiple edges. Also, we restrict the experiment on connected graphs. If the graph is not connected, we use the subgraph which is the largest connected component of the graph.

3.2 Betweenness Centrality

3.2.1 Exact computation

Freeman [2] defined betweenness centrality where $C_B(v)$ denote the betweenness centrality of a node v, which refers to the number of shortest paths between all pairs of vertices that pass through v.

$$C_B(v) = \sum_{s \neq v \neq t} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}} \tag{1}$$

Later, Brandes [5] reformulated (1) by introducing *dependency score*.

$$C_B(v) = \sum_{s \neq v \neq t} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}} = \sum_{s \neq v \neq t} \delta_{s,t}(v) = \sum_{s \neq v} \delta_s(v) \tag{2}$$

Where $\delta_s(v) = \sum_{t \neq v} \delta_{s,t}(v)$ is the *one-sided dependency* of s on v.

3.2.2 Approximate Computation

Brandes and Pich [7] presented an approximation algorithm for estimating betweenness centrality by randomly sampling $|S|$ pivot nodes with replacement, where S denote the *set of pivot nodes*. Note that $\delta_s(v)$ is computed by nodes in S. Let $\hat{C}_B(v)$ denote the estimated betweenness centrality of node v.

$$\hat{C}_B(v) = \frac{n}{|S|} \sum_{s \in S} \delta_s(v) \tag{3}$$

3.3 Community Detection

Community in a graph refers to a set of nodes that are highly intra-connected and sparsely inter-connected. Many methods were developed to detect communities in a graph. Seidman presented k-core [12] which is frequently used in community detection. Later, Batagelj and Zaversnik presented an efficient algorithm that runs in $O(m)$ [13] for determining the cores. Due to the fast running time of the algorithm, we used k-core to detect communities in the paper.

4 Experiments

We assess the quality of k-core based approximation algorithm on several graphs. The speedup of the algorithm is proportional to the number of pivot nodes. Hence we do not report the actual running time of the approximation algorithm. Data generation, community detection, and evaluation were implemented in Python language using Networkx [14] package and graph plotting was done in R using igraph [15] package.

4.1 Data

We experiment with three synthetic graphs and one real network in this study. **rand-er** is Erdos Renyi's [16] random graph model that generates a graph where the degree of each node follows a binomial distribution. The algorithm fixes number of nodes n and draws edges uniformly at random with probablility p where $0 < p < 1$. Expected number of edges with parameters n and p is $\binom{n}{2}p$.

rand-ba is a random graph model described by Barabasi and Albert [17], also called *preferential attachment* that generate heavy-tailed graphs. The algorithm starts with m_0 nodes and adds one node at a time until $|V| = n$. New nodes are added to the m existing nodes where $m \leq m_0$ with a probability that is proportional to the degree of existing nodes. Hence, nodes have a higher probability to be connected to nodes that already have many edges, leading the random graph to be heavy-tailed.

rand-ws is a random graph model by Watts and Strogatz [18] that produces graphs that have *small world* properties. The algorithm starts with a ring of n vertices where each vertex is connected to k nearest neighbors. Then, each edge is rewired randomly with probability p where $0 < p < 1$. Graphs generated in this approach have high local clustering.

hcw-sparse and **hcw-dense** are contact networks generated from the login records of healthcare workers at the University of Iowa Hospital and Clinics on electronic medical records [19]. 19.8 million login records are spanning over 21 months (September 1, 2006, through June 21, 2008) and each event is logged by de-identified user ID, date, login and log out time, and location. We generated **hcw-sparse** where nodes are healthcare workers who logged into the system on September 1 and edges are the number of logins that happened in the same room with an overlap between events.

For **hcw-dense**, we looked for four week period starting on September 8, and added edges if the login events happened within 30 minutes and are within five rooms apart from each other.

Label	Network	Nodes	Edges	Source
rand-er	random graph	1,000	$\approx$ 10,000	Erdos Renyi [16]
rand-ba	preferential attachment	1,000	19,600	Barabasi Albert [17]
rand-ws	small worlds	1,000	2,000 - 10,000	Watts and Strogatz [18]
hcw-sparse	contact network	1,461	3,973	Curtis et al [19]
hcw-dense	contact network	17,903	250,542	Curtis et al [19]

Table 1: Networks used in the experiments

4.2 Method

For a graph that has several communities many pairs of nodes would pass through the nodes that connect different communities. Based on this idea, we present a randomized approximation

algorithm *k-core betweenness centrality* that uses the community structure in a graph to effectively sample pivot nodes from the graph. Then we evaluate the accuracy of this algorithm with Brandes and Pich (3) centrality estimation algorithm.

4.2.1 Pivot Selection

For each graph, communities are detected using k-core approach. Let G_0 be the original graph and n denote the number of nodes in G_0. Let G_a^0, G_a^1, $\cdots$,G_a^{c-1} denote c communities in a-core. Let the number of nodes in a community G_a^c be n_a^c and let $n_a = \sum_{i=0}^{c-1} n_a^i$ where n_a denote total number of nodes in a-core. For $a = 1, 2, ...$ the algorithm computes n_a and returns G_a^0, G_a^1, $\cdots$,G_a^{c-1} when $n_a \leq \frac{1}{2}n$. Then for each community, the algorithm randomly selects $\lfloor \sqrt{n_a^c} \rfloor$ nodes as pivot nodes.

Algorithm 1: Pivot Selection - k-core betweenness centrality

```
   Input  : G_0
   Output: pivot nodes
 1 Function pivot_selection(G_0):
 2     a = 1;
 3     pivot_nodes = ∅;
 4     while True do
 5         G_a^0, G_a^1, ··· ,G_a^{c-1} = compute a-core in G_0;
 6         n_a = Σ_{i=0}^{c-1} n_a^i (where n_a^i is number of nodes in G_a^i);
 7         if n_a == 0 then
 8             break (get out of the while loop);
 9         else if n_a ≤ (1/2)n then
10             for i ← 0 to c − 1 do
11                 pivot = Randomly sample ⌊√(n_a^i)⌋ nodes from G_a^i;
12                 Append pivot to pivot_nodes;
13             end
14             break (get out of the while loop);
15         a = a + 1;
16     end
17     return pivot_nodes;
18 End Function
```

4.2.2 Approximate Computation

We carried out 100 repetitions of betweenness centrality approximation on two approximation algorithms (k-core betweenness centrality and Brandes Pich) on five different graphs in Table 1. The approximated betweenness centrality is computed in G_0 using sampled pivot nodes. Shortest paths for all possible pairs of nodes in pivot nodes are computed, and nodes that frequently appear in the shortest paths would have high approximated betweenness centrality. This approximation is computed using Equation (3), which is Brandes and Pich's formula [7].

4.3 Performance metric

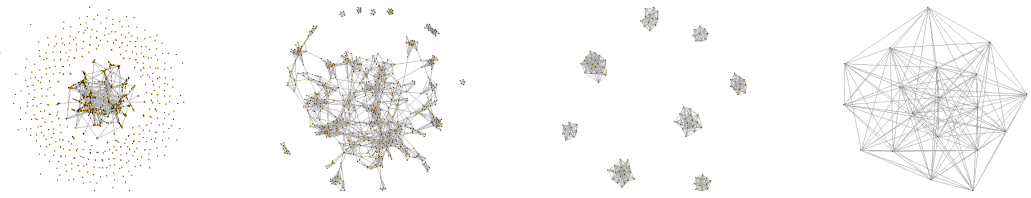
To evaluate the accuracy of approximation algorithms, we compared the *top-N%* high approximated betweenness centrality nodes with those nodes of exact computation. We measured the percentage overlap of approximation algorithm (k-core betweenness centrality and Brandes Pich [7]) with exact computation (Brandes [5]).

4.4 Results

k-core was able to capture communities in **hcw-sparse** contact network in different choices of k. **hcw-sparse** has 1,461 nodes and 3,793 edges. Algorithm 1 returned 28 pivot nodes that was sampled from 4-core in Figure 1b that has 577 nodes and 1,461 edges. Figure 2a shows the accuracy of the two approximation algorithms.

hcw-dense has 17,903 nodes and 250,542 edges. Algorithm 1 returned 272 pivot nodes that was sampled from 14-core that has 8,738 nodes and 17,903 edges. Figure 2b shows the accuracy of the two approximation algorithms.

For three random graphs, k-core could not detect communities. Hence, these graphs were not used to test approximation algorithms. To generate **rand-er**, $n = 1000$ and $p \approx 0.02$ were used that connected approximately 10,000 edges. The degeneracy of the graph was on average 13.97, with 888.93 nodes and 8,656.4 edges in k_{max}-core. For k in 1, 2, $\cdots$, k_{max}, k-cores had only



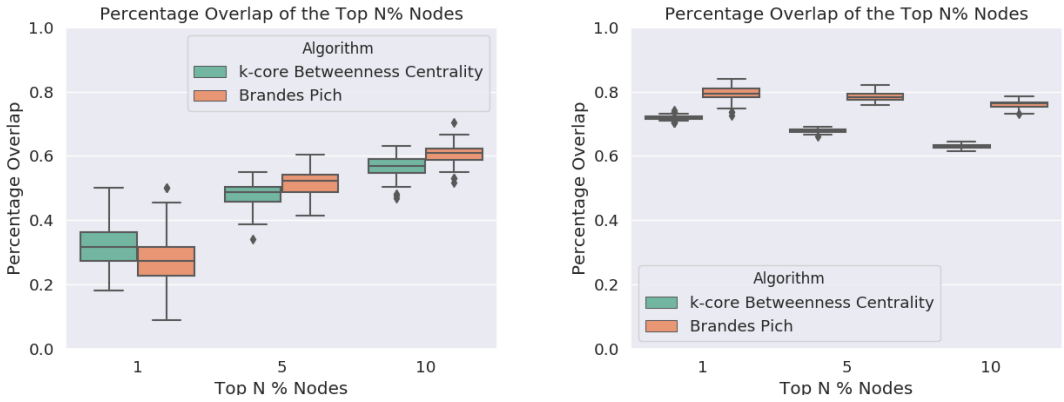
(a) k=0 (b) k=4 (c) k=8 (d) k=11

Figure 1: k-cores of **hcw-sparse** contact network

one connected component, implying that there was no community structure captured using k-core approach in **rand-er**.

Again, k-core approach failed to find communities in **rand-ba**. $n = 1000$, $m = 20$, were used to generate random graph with 19,600 edges. In the networkx package, $m = m_0$, meaning that initially 20 nodes are fully connected with each other, and new node is attached to existing 20 nodes with probability proportional to the degree of existing nodes. This setting lead random graph to have degeneracy of 20 and k_{max} was identical to the original graph.

For **rand-ws**, we set $n = 1000$ and then tried various pairs of parameters of $k \in [4, \cdots, 20]$ and $p \in [0, 0.01, 0.02, \cdots, 0.5, 1]$. For any parameter k, all the nodes are connected to exactly k neighbors when $p = 0$ which means that the degeneracy of the graph is k. For $p \neq 0$, edges are rewired with probability p, however, it did not generate a $k_{max} > k$. Also, k-cores had only one connected component.



(a) **hcw-sparse** (b) **hcw-dense**

Figure 2: Percentage overlap of top N% nodes of two approximation algorithms

5 Conclusion and Future Work

The algorithm is designed for graphs that have community structures which could be identifiable using k-core. The k-core betweenness centrality deterministically chooses k and randomly sample the square root of the number of nodes in each community. We believe there could be ways to improve the algorithm in selecting k and the way of sampling nodes from communities.

We conducted experiments to assess k-core betweenness centrality on different graphs. Results in Figure 2 show that Brandes Pich's algorithm works better than k-core betweenness centrality most of the time. We do not have a convincing explanation so far, but one possible interpretation is that high betweenness centrality nodes may not only be the ones that connect two communities. The inability to capture these nodes is the weakness of k-core betweenness centrality.

Although k-core betweenness centrality performed worse than Brandes Pich's algorithm for most of the time, Figure 2a shows that there exist a case when k-core betweenness centrality performed better than Brandes Pich's algorithm. Hence, the experiment of k-core betweenness centrality on real-world graphs with community structures is encouraged.

References

[1] D. Koschützki, K. A. Lehmann, L. Peeters, S. Richter, D. Tenfelde-Podehl, and O. Zlotowski, *Centrality Indices*, pp. 16–61. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005.

[2] L. C. Freeman, "A set of measures of centrality based on betweenness," *Sociometry*, vol. 40, no. 1, pp. 35–41, 1977.

[3] M. E. J. Newman, "A measure of betweenness centrality based on random walks," *Social Networks*, vol. 27, no. 1, pp. 39–54, 2005.

[4] I. Kivimäki, B. Lebichot, J. Saramäki, and M. Saerens, "Two betweenness centrality measures based on Randomized Shortest Paths," *Scientific Reports*, vol. 6, no. February, pp. 1–15, 2016.

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

[6] D. Eppstein and J. Wang, "Fast approximation of centrality," in *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '01, (Philadelphia, PA, USA), pp. 228–229, Society for Industrial and Applied Mathematics, 2001.

[7] U. BRANDES and C. PICH, "Centrality estimation in large networks," *International Journal of Bifurcation and Chaos*, vol. 17, no. 07, pp. 2303–2318, 2007.

[8] D. A. Bader, S. Kintali, K. Madduri, and M. Mihail, "Approximating betweenness centrality," in *Algorithms and Models for the Web-Graph* (A. Bonato and F. R. K. Chung, eds.), (Berlin, Heidelberg), pp. 124–137, Springer Berlin Heidelberg, 2007.

[9] N. Kourtellis, T. Alahakoon, R. Simha, A. Iamnitchi, and R. Tripathi, "Identifying high betweenness centrality nodes in large social networks," *Social Network Analysis and Mining*, vol. 3, pp. 899–914, Dec 2013.

[10] P. Suppa and E. Zimeo, "A clustered approach for fast computation of betweenness centrality in social networks," in *2015 IEEE International Congress on Big Data*, pp. 47–54, June 2015.

[11] A. Furno, N. E. E. Faouzi, R. Sharma, and E. Zimeo, "Two-level clustering fast betweenness centrality computation for requirement-driven approximation," in *2017 IEEE International Conference on Big Data (Big Data)*, pp. 1289–1294, Dec 2017.

[12] S. B. Seidman, "Network structure and minimum degree," *Social Networks*, vol. 5, no. 3, pp. 269 – 287, 1983.

[13] V. Batagelj and M. Zaversnik, "An o(m) algorithm for cores decomposition of networks," 2002.

[14] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using networkx," in *Proceedings of the 7th Python in Science Conference* (G. Varoquaux, T. Vaught, and J. Millman, eds.), (Pasadena, CA USA), pp. 11 – 15, 2008.

[15] G. Csardi and T. Nepusz, "The igraph software package for complex network research," *InterJournal*, vol. Complex Systems, p. 1695, 2006.

[16] P. Erdös and A. Rényi, "On random graphs i," *Publicationes Mathematicae Debrecen*, vol. 6, p. 290, 1959.

[17] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[18] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, pp. 440 EP –, Jun 1998.

[19] D. E. Curtis, C. S. Hlady, G. Kanade, S. V. Pemmaraju, P. M. Polgreen, and A. M. Segre, "Healthcare worker contact networks and the prevention of hospital-acquired infections," *PLoS ONE*, vol. 8, no. 12, p. e79906, 2013.