Fast Centrality Approximation in Modular Networks

Shu Yan Chan
Computer Laboratory
University of Cambridge, UK
syc22@cl.cam.ac.uk

Ian X. Y. Leung*
Computer Laboratory
University of Cambridge, UK
ixyl2@cl.cam.ac.uk

Pietro Liò
Computer Laboratory
University of Cambridge, UK
pl219@cl.cam.ac.uk

ABSTRACT

Measuring the centrality of nodes in real-world networks has remained an important task in the technological, social, and biological network paradigms carrying implications on their analysis and applications. Exact inference of centrality values is infeasible in large networks due to the need to solve the all-pairs shortest path problem. We introduce a framework to approximate node centralities in real-world networks that are known to exhibit modularity, i.e., the presence of dense subgraphs or communities, which are themselves sparsely connected. We also propose a novel centrality measure known as Community Inbetweenness that ranks nodes based solely on community information. In a modular network of size n with $\sqrt{n}$ evenly sized communities and m edges, our framework requires linear time $O(m)$ and $O(\sqrt{n}m)$ time for the approximation of closeness and betweenness respectively. Utilizing a recently proposed linear time method in community detection, our approximation techniques are faster than traditional sampling algorithms, applicable in real-time distributed environments, and offer highly comparable results.

Categories and Subject Descriptors

H.3.1 [**Content Analysis and Indexing**]: Indexing Methods

General Terms

Algorithms, Experimentation, Measurement

Keywords

Centrality, community detection, complex network, modularity

*Corresponding author.

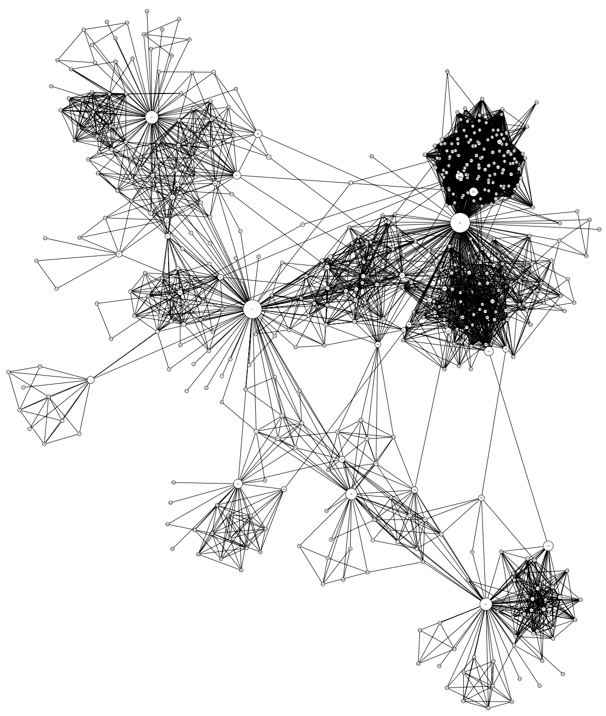

Figure 1: The snapshot of a 500-node subgraph of the Orkut OSN reveals clear modular structure. The size of the nodes corresponds to their betweenness centrality.

1. INTRODUCTION

The calculation of the importance or centrality of nodes in a network has been a vital tool in the analysis of network structure with applications such as vulnerability analysis in communication networks [11], node indexing in social networks, and even identifying key components in biological networks [3]. The exact inference of such measures, however, has long known to be infeasible in large networks due to the necessity for solving the all-pairs shortest paths (APSP) problem in the network. The associated running time can be given by $O(nm)$ where n is the size of the network and m is the number of edges. Notably, the proliferation of online social networks (OSNs) with number of nodes easily exceeding tens of millions has opened up immense research opportunities while necessitating the use of efficient and decentralized techniques for their analysis.

Network	Size	Links	Diameter[1]	Modularity[2]
Amazon (Mar'03)	409,687	4,929,260	6	0.72
Orkutz OSN [17]	3,072,411	223,534,301	9	0.67

Table 1: Characteristics of two large-scale networks. In general, they are modular, have a small diameter, and have a power-law degree and community size distribution.

Apart from their sheer size, networks found in real-world are known to exhibit some interesting properties. One such example is known as the small-world property [20], contributing to the famous notion of "six-degree of separation" [16]. Importantly, real-world networks are also known to exhibit modularity, i.e., nodes are often found in densely connected subgraphs or communities, which are themselves sparsely connected (c.f. Figure 1). The partitioning or clustering of such closely connected nodes in a network, often known as community detection, has been a subject of investigation in recent years [19]. While in certain networks such communities are "self-reported", such as groups or affiliations in an OSN, recent work has demonstrated the possibility of accurate real-time community detection with a linear-time ($O(m)$) and fully distributed algorithm [14]. Our work thus aims to exploit known community structure to approximate centrality indices. Table 1 summarizes some of the characteristics of two well known large-scale networks. We apply our methods to subgraphs of these networks as will be discussed in Section 4. Interested readers are referred to [1] for a detailed overview of the different characteristics of complex networks.

Defining the "importance" of a node in a network is not a trivial task. In the simplest case, one can regard nodes with higher number of neighbors (node degree) as important. Another notable centrality index, known as closeness, measures the average distance of the node from all other nodes in the network. Perhaps the most investigated centrality measure of all is the betweenness centrality as supported by its many variations [6]. The importance is usually defined by the fraction of shortest paths between all pairs of nodes in the network through the node concerned. Our proposed framework focuses on both the approximation of node closeness and shortest-path betweenness and can be extended to facilitate other measures.

Different measures and related methods will be further discussed in the next section. In Section 3, we present a novel centrality measure known as Community Inbetweenness as well as a framework for the estimation of the closeness and betweenness centralities. We analyze our results and comparisons in Section 4 and finally conclude with future research directions in Section 5.

2. RELATED WORK

Centrality provides the standard means to compare between nodes in networks. Consider a graph $G = (V, E)$ where V represents a non-empty set of nodes, E as a set of edges. We define the neighborhood set of node $v \in V$ reachable in n hops, $\mathcal{N}_v^n = \{v' \in V | v' \neq v \wedge d_G(v, v') \leq n\}$, where d_G is the geodesic distance. The simplest of all centrality measure then, intuitively, is the degree centrality:

$$C_D(v) = \frac{|\mathcal{N}_v^1|}{|V| - 1}, \tag{1}$$

which measures, effectively, the connectivity of a certain node. In a large network, however, the degree centrality of a node may not be representative of its influence on the whole network. A more involved measure known as closeness is defined by the average distance of all nodes in the network from v,

$$C_C(v) = \frac{\sum_{t \in V} d_G(v, t)}{|V| - 1}. \tag{2}$$

It is also common to take the reciprocal of the above to justify the term "closeness" since the above is really describing "farness".

The betweenness centrality proposed by Freeman [9] is perhaps the most popular measure, and hence is often assumed when centrality is spoken of. It measures the proportion of shortest paths in the network which go through node v and can be defined as $C_B^\infty(v)$ where:

$$C_B^n(v) = \sum_{u,t \in \mathcal{N}_v^n} \frac{\sigma_{ut}(v)}{\sigma_{ut}}, \tag{3}$$

where σ_{ut} is the number of (equal distance) shortest paths between nodes u and t and $\sigma_{ut}(v)$ is the number of those through node v. Freeman later proposed a variation known as the flow betweenness [10], which considers all possible paths through each node, weighted according to the path capacities. Another important variation proposed by Newman [18] is based on the scenarios when information are neither traveling through the shortest paths nor utilizing all possible paths in a network. Node importance is therefore based on the number of net traverses by a random walker through the node concerned between all possible pairs. This measure is conveniently named the random-walk betweenness.

The exact calculation of the closeness requires solving the APSP problem and hence require $O(nm)$ in an unweighed network which is prohibitive in large networks. An intuitive speed up is by instead solving the single source shortest path (SSSP) problem from a subset of nodes, often called the pivots [7]. While choosing pivots following certain heuristics has been shown to be of no particular benefits over choosing them randomly [7], Eppstein [8] deduced the Hoeffding's bound for these random sampling techniques and concluded that with $\Theta(\log n)$ samples, the error is within a reasonable constant bound with very high probability.

The seminal work by Brandes [5] enables the exact computation of betweenness by aggregating path counts from dif-

[1]The diameter of a network is the length of the longest path amongst the set of shortest paths between any two nodes in the network.

[2]Newman's modularity [19] is a well known measure which compares the community partitioning of a given network to a random network of identical degree sequences. A high value provides a strong evidence for the presence of communities or modules [19].

ferent sources in the network (effectively the same as solving the APSP) with time cost $O(nm)$. It improves on the Floyd-Warshall algorithm which deduces all shortest paths from all possible pairs of nodes and runs in time $O(n^3)$. Although sampling methods similar to its closeness counterpart has been investigated [7], the error bound is much greater and the number of samples required is not well understood and highly network-dependent. Since Brandes' algorithm works by path count aggregation, the whole APSP problem has to be solved even if only one node's betweenness is of concern. [2] investigates the possibility to speed up the calculation of betweenness for one given node by an adaptive sampling approximation. Another speed up, known as the egocentric centrality [15], focuses on smaller subgraphs around the node concerned and essentially evaluates the betweenness of a node in its immediate neighborhood, i.e. C^1_B, with obvious merits and drawbacks.

See [6] for other variations of betweenness and [4, 13] for some other related measures. Note also that most measures can be intuitively applied on edges as well as nodes.

3. METHODOLOGY

The key intuition of our approximation is that nodes within a community are tightly connected (i.e. communities have small diameters), the implication is that these nodes share certain characteristics that can be abstracted in devising different centrality indices. For instance, the shortest path between a node in one community to a node in another community is likely to overlap with most shortest paths between the nodes of these two community (betweenness) or have similar hops counts (closeness). We introduce notations of abstract graphs which capture these similarity with a significant reduction in the topological complexity.

We assume that community information is either known beforehand or that it is uncovered by the aforementioned label propagation community detection algorithm with running time $O(m)$, briefly summarized as follows. Each node is initially given a unique label. In each iteration, each node's label is replaced by the label which most of its neighbors (including its own) have or randomly if there are more than one choice. Experiments have shown that on average a constant number of iterations is sufficient to uncover accurately the underlying community structure in a large network [14]. The communities detected by the algorithm are disjoint and members of each detected community induce a connected subgraph - these form two further preconditions of our approximation.

3.1 Closeness

Consider a graph $G = (V, E)$ and a set $\mathcal{C} \subseteq \mathcal{P}(V)$ of non-overlapping communities, we first define an Abstract Graph $\widetilde{G}(\mathcal{C}) = (\widetilde{V}, \widetilde{E})$ as follows:

1. For each community $c_i \in \mathcal{C}$, we create a community node $\widetilde{v}_i \in \widetilde{V}$ with an associated weight $\widetilde{w}_i = |c_i|$.

2. We create an edge $(\widetilde{v}_i,\ \widetilde{v}_j)$ if there exists an edge between any members of community c_i and c_j.

We now give an overview of the Closeness Centrality approximation algorithm:

Input: Graph G and Communities $\mathcal{C}$
Output: Closeness approximate, $\widetilde{C_C}(v)$, $v \in V$

1. Construct Abstract Graph $\widetilde{G}(\mathcal{C})$

2. Deduce the weighted closeness, $\widehat{C_C}$, for each node $\widetilde{v}_i$ by:
$$\widehat{C_C}(\widetilde{v}_i) = \frac{\sum_{\widetilde{v}_j, i \neq j} \widetilde{w}_j \cdot d_G(\widetilde{v}_i, \widetilde{v}_j)}{|V| - \widetilde{w}_i};$$

3. For each node $v \in c_i$,
$$\widetilde{C_C}(v) = \frac{\widehat{C_C}(\widetilde{v}_i)}{(\mathrm{Deg}(v))^{\omega}}, \omega \geq 0,$$

where $\mathrm{Deg}(v)$ is the degree of node v. $\widehat{C_C}$ hence approximates the closeness of nodes by assuming a uniform distribution of closeness within each community. $\widetilde{C_C}$ uses extra information on the degree of the node in an attempt to differentiate nodes within the community. The assumption here is that node with high degree should have higher chance to be reached by other nodes and therefore a smaller closeness value. The parameter ω is introduced to adjust the extent to which we take degree information into consideration.

3.2 Betweenness

We propose in this section two efficient approximation techniques utilizing the community structure of the network.

3.2.1 Community Inbetweenness

Recall the seminal community detection algorithm proposed by Girvan and Newman [19] which reveals densely connected substructures of a network by gradually removing edges of high betweenness. Can we somehow reverse the idea to deduce the importance of the nodes from prior information of the community structure? As a first step, we propose a simple entropy-based measure, Community Inbetweenness (C_{CI}), which evaluates node importance based solely on its surrounding link and community information. Let $p_{v \to c}$ denote the proportion of links in node v that connects v to nodes that belong to community c, $C_{CI}(v)$ is given by:

$$C_{CI}(v) = \mathrm{Deg}(v) \cdot \sum_{c \in \mathcal{C}} p_{v \to c} \log \frac{1}{p_{v \to c}}. \qquad (4)$$

The above measure weighs nodes with higher degree and more connections to different communities over those with less degree and less disparate communications. An obvious drawback of the measure is that it would fail to recognize nodes that are contained inside their communities. These nodes may still play an important role if they lie on any shortest paths through the community, and therefore C_{CI} may have a weak correlation to such nodes with lower betweenness. We believe, however, that the above measure would be able to identify highly important nodes in a modular network (see Figure 1 where high betweenness nodes are commonly found in between densely connected subgraphs). Further investigation is given in Section 4.

3.2.2 *Shortest-path Betweenness*

We continue here to investigate how the graph abstraction framework can be utilized to speed up the derivation of shortest-path betweenness.

Average Traversal Cost.

A vital component in our estimation is to understand the cost to traverse through any community in the Abstract Graph. Paths can only enter and leave a community through a subset of nodes which have connections to the rest of the network, referred to as border nodes. A basic estimation of the Average Traversal Cost (ATC) can therefore be the mean of all the shortest paths between all pairwise combination of the border nodes. This assumes that any shortest path through the community would be equally likely to enter and leave from any one of its border nodes. The estimation can be improved by considering more and more information from neighboring communities, and eventually the entire network, so that the exact ratios of the shortest paths entering and leaving each border node pair are known.

The estimation we adopt seeks to improve on the aspects of network structure which the basic estimation omits, as illustrated in Fig. 2 below:

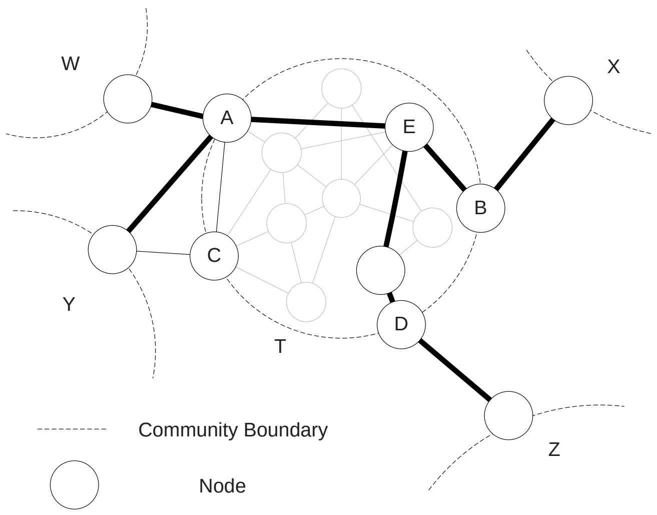

Figure 2: Schematic view of a community. Border nodes A, B, and D lie on the global shortest paths between communities.

1. There are inter-community paths which enter and leave the community via the same border node (e.g. the path $W \rightarrow A \rightarrow Y$).
2. Some border nodes (e.g. C) may lie on the suboptimal paths between nodes and thus never contribute to the global shortest paths between nodes.

To improve on the above drawbacks of the aforementioned basic estimation, the traverse cost from a community through a specific neighbor community should be computed individually.

Given an Abstract Graph, we consider a $|\mathcal{C}| \times |\mathcal{C}|$ square matrix $M^{(v)}$ for each node v, where $M^{(v)}_{i,j}$ denotes the shortest distances from neighbor community node i through v to j. Now, if the network concerned is undirected, $M^{(v)}_{i,j} = M^{(v)}_{j,i}$ and in a modular network, the number of non-zero columns or rows is less than $c \ll n$. Refer to Figure 2 where the community in focus is T. The ATC from neighbor community W through T, $ATC^T(W)$, is approximated by the mean of all the lengths of shortest paths from community W that traverse through T to reach all other communities:

$$ATC^T(W) = \frac{\sum_{j \neq W} M^{(T)}_{W,j}}{\sum_{j \neq W} \delta(M^{(T)}_{W,j})}, \tag{5}$$

where $\delta(x) = 1$ if $x > 0$, 0 otherwise.

Focused Abstract Graph.

Before finalizing the algorithm, we first introduce the Focused Abstract Graph, which in effect zoom in the Abstract Graph onto a particular community c:

Given a graph $G = (V, E)$ and a set $\mathcal{C} \subseteq \mathcal{P}(V)$ of non-overlapping communities, the Focused Abstract Graph for a community $c \in \mathcal{C}$, $\breve{G}(c) = (\breve{V}, \breve{E})$, is defined as follows:

1. The nodes in c and the intra-community edges in c are copied from $G = (V, E)$ to $\breve{G}(c) = (\breve{V}, \breve{E})$.
2. For each community $c_i \neq c \in V$, create community node $\breve{v}^c_i \in \breve{V}$.
3. An edge $(\breve{v}^c_i, \breve{v}^c_j)$ between two community nodes is created if there exists an edge between any members of community c_i and c_j.
4. An edge $(\breve{v}^c_i, \breve{v})$ between a community node $\breve{v}^c_i$ and a border node $\breve{v}$ of c is created if there exists an edge between $\breve{v}$ and any node in community c_i.
5. Assign a weight of 1 to each intra-community edges in c. Weights are assigned to the rest of the edges in the graph as follows:

$$weight(\breve{v}^c_i,\ \breve{v}^c_j) \ = \ \frac{ATC^{v^c_j}(\breve{v}^c_i) + ATC^{v^c_i}(\breve{v}^c_j)}{2} - 1;$$

$$weight(\breve{v}^c_i,\ \breve{v}) \ = \ \frac{ATC^{v^c_i}(\breve{v})}{2}.$$

ATCs are directional, e.g. $ATC^A(B)$ is the ATC of A through community node B to other nodes. This naturally leads to abstracted graph with directed edges. However, since the original graph is undirected, the edge weight between two nodes is assigned to be the average of the two corresponding directed ATC between them.

Back propagation of shortest path count.

This section explains how Brandes' algorithm [5] is adapted to the Focused Abstract Graph. The original algorithm considers the shortest paths from each source, s, in turn to all other nodes in the network. Once all shortest paths from s are derived (by Dijkstra's algorithm), the first observation is that the number of shortest paths from s to any node w, denoted by $\sigma_s[w]$, is in fact the sum of the number of shortest paths from s to w's predecessors, denoted by the set $P_s(w)$ (c.f. line **c** in Algorithm 1). Consider first the simplest case where w has only one immediate predecessor,

v, then the weighted sum of shortest paths through v, $\delta_s[v]$, is given simply by $\delta_s[w]+1$ (to include one new path from s to w). Now a vital observation comes in where w has multiple predecessors, the actual contribution of w's shortest path through counts to each $v \in P_s(w)$ is in fact weighted by $\frac{\sigma[v]}{\sigma[w]}$. The essence of Brandes' algorithm is thus based on the following theorem [5]:

$$\delta_s[v] \;=\; \sum_{w:v\in P_s(w)} (\delta_s[w]+1)\cdot\frac{\sigma[v]}{\sigma[w]} \qquad (6)$$

Informally, the algorithm looks at nodes furthest away from the source node, then considers all their immediately predecessors, and accumulates the count of shortest paths that pass through those nodes according to the above theorem. This process is completed when the immediately predecessor is the source node itself, and repeated for all nodes as different sources.

Consider now the following single shortest path between two community nodes A and Z in the Focused Abstract Graph, where A is the source.

$$\begin{tikzcd} A \arrow[r] & B \arrow[r] & C \arrow[r, squiggly] & X \arrow[r] & Y \arrow[r] & Z \end{tikzcd}$$

A single abstracted shortest path between two community nodes represents an approximation of all the shortest paths between all pairwise combination of nodes within the two communities. Therefore, using the notation $size(A)$ for the size of community A, the approximation of the real shortest paths through count $\delta_A[Y]$ (the real number of shortest paths starting from community A passing through community Y) is given intuitively by $size(A)\cdot size(Z)$. These paths must also pass through the immediate preceding community in the abstracted shortest path, X. Note also that there are extra paths passing through X on top of the $\delta_A[Y]$ paths, i.e. the shortest paths between all pairwise combination of nodes within communities A and Y, altogether $(size(A)\cdot size(Y))$ of them. Therefore, $\delta_A[X]$ is propagated from Y as $\delta_A[Y]$ + $(size(A)\cdot size(Y))$.

The full adaptation of Brandes' algorithm to the Focused Abstract Graph is therefore:

$$\delta_A[X] \;=\; \sum_{Y:X\in P_s(Y)} (\delta_A[Y]+size(A)\cdot size(Y))\cdot\frac{\sigma[X]}{\sigma[Y]} \quad (7)$$

If we take the community size represented by the nodes in the Focused Abstract Graph to be 1 (i.e. they all just represent normal nodes in the original network), then our formulation reduces back to Brandes'. This formulation is utilized in line **d** in Algorithm 1.

Overview of Algorithm.

The modified algorithm runs on the Focused Abstract Graph to estimate the number of shortest paths passing through each node in the focused community:

Input: Graph G and Communities $\mathcal{C}$
Output: Betweenness approximation, $\widetilde{C_B}(v)$, $v \in V$

The node betweenness values are computed one community at a time, for each community c:

1. Create the Focused Abstract Graph $\breve{G}(c) = (\breve{V}, \breve{E})$
2. For each node $\breve{v}$ in the Focused Abstract Graph:
   (a) Using Dijkstra's algorithm, compute the shortest paths from $\breve{v}$ to all nodes in the network. Accumulate the count of the number of (equal weight) shortest paths reaching each node as the algorithm traverse the entire network.
   (b) The above step produces a stack of nodes that can be popped in descending shortest distance from the source $\breve{v}$. Back propagate in this order, according to our description in the previous section, allows us to efficiently derive the number of shortest paths from $\breve{v}$ through each node in the Focused Abstract Graph.

   The approximation to node betweenness values of the community are derived after iterating through all the nodes in the Focused Abstract Graph, summing up the contribution of shortest paths from each source node/community passing through each nodes.

The detailed implementation of the above algorithm is given in Algorithm 1.

4. DISCUSSION

In this section we compare the performance of our algorithms with existing sampling techniques on two real-world graphs mentioned earlier as well as an artificial benchmark graph of modular network [12]. We also discuss the running time and some of the limitations of our algorithms.

4.1 Experimental Results

The real-world graphs used in our experiments are as described in Table 1. Due to time constraints for evaluating the actual centralities for comparisons, we have restricted the size of the real-world networks to 10,000. This is done by snowball-crawling on the largest component of each of the network up until the desired number of nodes is reached. While a lot of link information is inevitably lost, those sampled subgraphs retain a highly modular structure and are therefore suitable for our purpose. The artificial graph allows arbitrary average degree, mixing parameter dictating the noise between communities, as well as arbitrary degree and community size distribution exponents. The artificial benchmark graphs are created up to sizes 10,000 and 20,000 in each repetition, with average degree set at 20, mixing parameter of 0.2, degree power-law decay exponent at 3 and community-size exponent at 2.

4.1.1 Closeness Approximation

We compare our approximation $\widetilde{C_C}$ (with ω set at 0.5), with $\widehat{C_C}$, degree, and approximation by carrying out $\log(n)$ random SSSP samples. The results are summarized in Figure 3.

$\log(n)$-sampling performs consistently well over all graphs especially on the two real-world networks but worse on benchmark graphs. The benchmark graphs produce communities of highly heterogeneous sizes. The random and unbiased rewiring between communities contribute to an overall network with a very low diameter. Subsequently, the few random pivots that land on the network are insufficient to distinguish the closeness of highly similar magnitude between the nodes. This case is not true in the case of an OSN, as

Algorithm 1: Betweeness centrality via Focused Abstract Graph

```
Input   : $\breve{G}(c) = (\breve{V}, \breve{E})$, $\epsilon \in [0,1]$ (error margin)
Output: $C_B[\,]$ (Betweenness approximations)
begin
    $C_B[v] \leftarrow 0, v \in \tilde{V}$;
    foreach $s \in \tilde{V}$ do
        $S \leftarrow$ empty stack;
        /* lists of shortest path predecessors */
        $P[w] \leftarrow$ empty list, $w \in \tilde{V}$;
        /* counters for # of shortest paths */
        $\sigma[t] \leftarrow 0,\ t \in \tilde{V}$; $\sigma[s] \leftarrow 1$;
        /* distances from source */
        $d[t] \leftarrow -1,\ t \in \tilde{V}$; $d[s] \leftarrow 0$;
        /* Priority queue of nodes ordered by
        increasing d[node] */
        $Q \leftarrow$ empty queue; enqueue $s \rightarrow Q$;
        /* Dijkstra's algorithm, also counting
        the number of equal distance shortest
        paths to reach each node */
        while $Q$ not empty do
            dequeue $Q \rightarrow v$;
            push $v \rightarrow S$;
            foreach neighbor $w$ of $v$ do
                /* w found for the first time?  */
                if $d[w] < 0$ then
                    $d[w] \leftarrow d[v] + weight(v,w)$;
                    enqueue $w \rightarrow Q$;
                /* shorter path to w via v?  */
a               if $d[w] \cdot (1-\epsilon) > d[v] + weight(v,w)$
                then
                    $d[w] \leftarrow d[v] + weight(v,w)$;
                    $\sigma[w] \leftarrow \sigma[v]$;
                    $P[w] \leftarrow new\ list()$;
                    append $v \rightarrow P[w]$;
                    /* reorder w in Q with the value
                    of d[w] */
                    $Q.decreaseKey(w)$;
b               else if $d[w] \cdot (1+\epsilon) \geq$
                $d[v] + weight(v,w)$ then
                    /* one of the shortest paths to
                    w via v */
c                   $\sigma[w] \leftarrow \sigma[w] + \sigma[v]$;
                    append $v \rightarrow P[w]$;
        /* counters for number of shortest paths
        from s passing through each node*/
        $\delta[v] \leftarrow 0,\ v \in V$;
        /* S returns vertices in decreasing order
        of distance from s */
        while $S$ not empty do
            pop $S \rightarrow w$;
            foreach $v \in P[w]$ do
d               $\delta[v] \leftarrow$
                $\delta[v] + \frac{\sigma[v]}{\sigma[w]} \cdot (C(s).size \cdot C(w).size + \delta[w])$;
                /* C(v) denotes the community
                represented by v */
                if $w \neq s$ then
                    $C_B[w] \leftarrow C_B[w] + \delta[w]$;
end
```

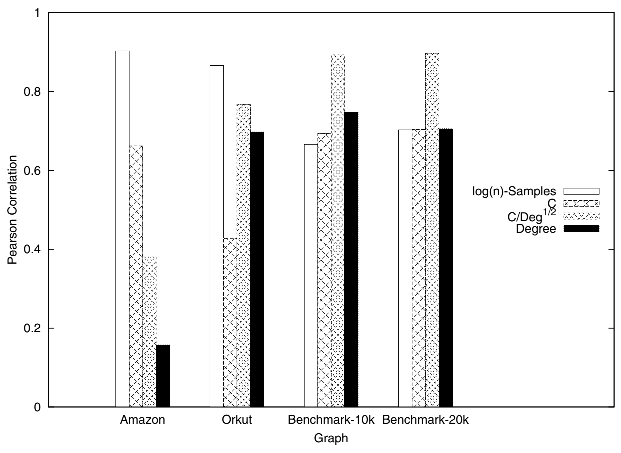

Figure 3: The Pearson correlation of four closeness approximation algorithms against the actual closeness on four networks. $\widehat{C_C}$ is denoted by C and $\widetilde{C_C}$ by $C/Deg^{1/2}$. Results are averaged over 10 realizations.

shown in Figure 1, where the network diameter is higher. We also observe that where high degree-closeness correlation is exhibited, $\widetilde{C_C}$, which takes both approximate closeness and degree into account, would positively adjust the unweighed measure $\widehat{C_C}$, contributing to a higher overall correlation to actual closeness. This is understandable given that degree ignores the global position of the node, combining also the approximated closeness is highly beneficial. Although as in the Amazon network, the lack of degree-closeness correlation greatly affects the weighted approximation. The usage of ω is hence subject to further investigation.

4.1.2 Betweenness Approximation

We first look at the performance of the proposed Community Inbetweenness(C_{CI}) measure against actual betweenness on a 5000-node subgraph of the Orkut OSN depicted in Figure 4.

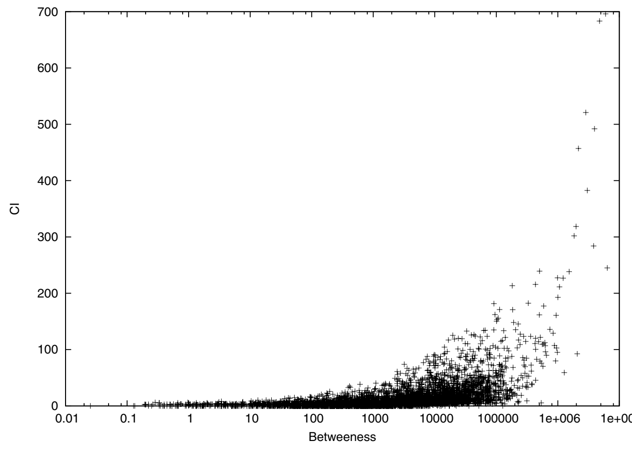

Figure 4: The linear-log plot of Community Inbetweenness (CI) against the actual betweenness on a 5000-node subgraph of the Orkut OSN.

The scatter plot clearly reveals a positive correlation between the two measures. As discussed earlier, C_{CI} underestimates non-border nodes in a community as conveyed by the bottom cluster on the plot indicating nodes with low C_{CI} but average-valued betweenness. However, the results also clearly suggest that high C_{CI} values corresponds well to nodes with high betweenness. Nodes with betweenness value exceeding 1×10^6 are all picked up by high C_{CI} scorings. The correlation between the two measures on this subgraph is 0.72 and increases to 0.79 if we focus on the top 5% nodes exhibiting high betweenness.

Figure 5 shows the Pearson correlation between five different measures and the actual betweenness of the four graphs. We compare our approximation against random samples of Brandes' SSSP path counts as well as the proposed C_{CI} measure and the degree. Recall that our approximation technique works on weighted path accumulation on the Focused Abstract Graph, here we also compare a modification where we restrict the number of SSSP accumulations to the square root of each of the Focused Abstract Graph's size. This is inspired by the common random sampling techniques and further reduces the running time of our Abstract Graph approximation.

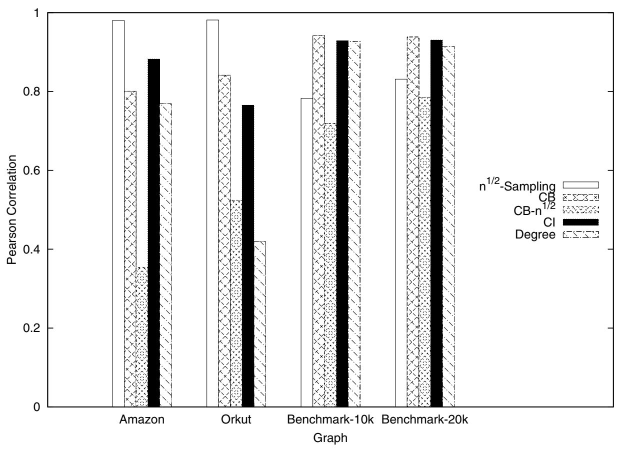

Figure 5: Pearson correlations of five betweenness approximation algorithms against the actual betweenness on four networks. CB **corresponds to our approximation and** $CB{-}n^{1/2}$ **corresponds to** $\sqrt{n}$ **sampling on the Focused Abstract Graph. Results are averaged over 10 realizations.**

The C_{CI} measure has the degree of the node as one of its term, so it is not surprising to see that where there are high degree-betweenness correlation in the networks, it has high correlation as well. However, it also consistently performs well across different networks, indicating that the entropy term also provides good indication of the node's centrality. It is perhaps not surprising to see that the $\sqrt{n}$-sampling on the Focused Abstract Graph performs worse than without sampling in all cases. Our approximation technique performs respectably in all networks and outperforms the $\sqrt{n}$ SSSP sampling in the benchmark networks. The benchmark networks are highly modular and as a result fit better to the assumptions of our algorithm than the Amazon and Orkut networks.

4.2 Performance Analysis

4.2.1 *Closeness*

Referring to Section 3.1, the creation of Abstract Graph requires running time $O(m)$, as we are required to go through each community to detect all out-going edges. Calculating the APSP in the Abstract Graph $\widetilde{G}$ requires $O(|\widetilde{V}||\widetilde{E}|)$. Since $|\widetilde{V}|$ equals the number of communities, which we denote by c, and if we assume the average case of $c \leq \sqrt{n}$ and $|\widetilde{E}| \leq \sqrt{m}$, the closeness approximation algorithm then requires running time $O(m)$.

4.2.2 *Betweenness*

The computation of Community Inbetweenness requires running time $O(m)$ since each node requires only to examine its immediate neighbors' community information.

As to the abstraction framework, the following analysis assumes the average case of each community to contain $\frac{n}{c}$ nodes and $\frac{m}{c}$ edges. Identifying neighboring communities for all communities requires running time $O(m)$, similar to the that of the creation of an Abstract Graph. Obtaining the ATC for each community node requires $O(c \cdot \frac{m}{c})$ for SSSPs from each neighbor community. Repeated for each community node gives time $O(cm)$. Assuming the existence of the Abstract Graph, each Focused Abstract Graph requires $O(\frac{m}{c})$ time to replace the abstract node by expanding nodes and edges of the focused community, so the total cost for all Focused Abstract Graphs is $O(m)$. Lastly, running c times the variant of Brandes' algorithm on the Focused Abstracted Graph each requires $O((\frac{n}{c} + c) \cdot \frac{m}{c})$, giving an overall running time of $O(\frac{nm}{c} + cm)$. Therefore, the total running time complexity dominated by the term $O(\frac{nm}{c} + cm)$. The optimal running time is therefore achieved when $c = \sqrt{n}$, giving an overall running time of $O(\sqrt{n}m)$.

It is important to notice also a fundamental drawback of traditional sampling algorithms for computing centralities values in real-world system. It is widely acknowledged that the derivation of shortest paths in large-scale real-world systems often suffer from constraints imposed by the system itself. For instance, there are significant overheads in terms of the communication cost in a mobile ad-hoc network, say, that prohibits the derivation of shortest paths between all nodes in a network as effectively as one would anticipate in an offline analysis. A similar argument can also be made on large-scale distributed databases that is found on most online social networks. Our approach is inherently decentralized due to the focuses on closely connected communities. The abstraction of network significantly reduces the aforementioned communication overheads despite the similar computation complexities between traditional sampling approaches and our approximation framework.

4.3 Limitation

There are several inaccuracies involved in the approximations. Firstly, when a shortest path between two community nodes is devised, it is implicitly assumed that all shortest paths between nodes in the two communities in the original graph also follow the same path. It is possible, however, that a proportion of the shortest paths actually deviates from the calculated path. The algorithm attempts to account for this by allowing a margin of error ϵ in the path length estimations when deciding whether two paths are considered of equal length (lines a and b in Algorithm 1).

Another type of inaccuracy in the approximation is in the path length estimations. The current ATC takes the rough assumption that the shortest paths within a community are utilized by the rest of the network uniformly. A more accurate ATC would have to incorporate estimations on the ratio of the shortest paths being visited. However, this requires the estimation on the global shortest paths involving the rest of the network, which in turn requires the use of the ATCs of other communities. One possible improvement to resolve the recursive dependency would be an iterative approach. After initial estimations on the ATCs and the subsequent discovery of the global shortest paths, we could feed the information back to each community to improve the actual estimation of the ratio. This mechanism is subject to further investigation.

Finally, recall that an edge between community nodes A and B would be assigned the ATC value $\frac{ATC^{A}(B)+ATC^{B}(A)}{2}-1$. If the difference between the two values to be averaged is significant, the assigned weight to the edge may not be representative for the link between the two communities/nodes in either direction.

5. CONCLUSION AND FUTURE WORK

In this literature, we have proposed a novel centrality measure and a framework for approximating closeness and betweenness centralities based on community information. Results and analysis suggest that our proposed techniques run efficiently with respectable accuracy on different real-world and benchmark graphs. Our investigation opens the opportunity for real-time centrality estimation in large-scale real-world networks where modularity is often exhibited. Importantly, our approach is easily decentralizable and highly scalable, which is vital for any efficient and online analysis in real-world large-scale networks.

One of the major future directions for this work is to further improve the ATC estimation accuracy which dictates the final outcome of our approximation. It is worth investigating how an iterative approach (as suggested in previous section) would improve on the ATC estimations. As mentioned, our framework can be adapted to different centrality measures, but further investigations will be required, for example, to abstract Newman's random-walk betweenness of communities. We also conjecture that less modular graph has a higher ATC variance and thus a higher number of potential shortest paths amongst communities which violates some of the assumptions in the algorithms. Future work is needed to explore the relationship between the performance of our approximation and the modularity of a network by different community detection techniques. Lastly, a detailed analysis on the average running time on different real-world networks as well as the error bound of our approximation is of considerable interest.

Our abstraction framework opens up a new paradigm to large-scale and real-time network analysis. With further investigations, we believe the proposed methodologies would be of substantial value to the field.

6. ACKNOWLEDGMENTS

This work is supported by the EC IST SOCIALNETS project, grant agreement number 217141, and the International Technology Alliance, sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defense.

7. REFERENCES

[1] R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Rev. Mod. Phys.*, 74:47–97, Jan 2002.

[2] D. Bader, S. Kintali, K. Madduri, and M. Mihail. Approximating betweenness centrality. *Lecture Notes in Computer Science*, 4863:124–137, 2007.

[3] A. Barabási and Z. Oltvai. Network biology: understanding the cell's functional organization. *Nature Reviews Genetics*, 5(2):101–113, 2004.

[4] S. Borgatti. Centrality and network flow. *Social Networks*, 27(1):55–71, 2005.

[5] U. Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(2):163–177, 2001.

[6] U. Brandes. On variants of shortest-path betweenness centrality and their generic computation. *Social Networks*, 30(2):136–145, 2008.

[7] U. Brandes and C. Pich. Centrality estimation in large networks. *International Journal of Bifurcation and Chaos in Applied Sciences and Engineering*, 17(7):2303, 2007.

[8] D. Eppstein and J. Wang. Fast approximation of centrality. *Journal of Graph Algorithms and Applications*, 8(1):39–45, 2004.

[9] L. Freeman. Centrality in social networks: Conceptual clarification. *Social networks*, 1(3):215–239, 1979.

[10] L. Freeman, S. Borgatti, and D. White. Centrality in valued graphs: A measure of betweenness based on network flow. *Social Networks*, 13(2):141–154, 1991.

[11] P. Holme, B. Kim, C. Yoon, and S. Han. Attack vulnerability of complex networks. *Phys. Rev. E*, 65(5):56109, 2002.

[12] A. Lancichinetti, S. Fortunato, and F. Radicchi. Benchmark graphs for testing community detection algorithms. *Phys. Rev. E*, 78:046110, 2008.

[13] V. Latora and M. Marchiori. A measure of centrality based on network efficiency. *New Journal of Physics*, 9(6):188, 2007.

[14] I. X. Y. Leung, P. Hui, P. Liò, and J. Crowcroft. Towards real-time community detection in large networks. *Phys. Rev. E*, 79(6):066107, 2009.

[15] P. Marsden. Egocentric and sociocentric measures of network centrality. *Social Networks*, 24(4):407–422, 2002.

[16] S. Milgram. The small world problem. *Psychology today*, 2(1):60–67, 1967.

[17] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and analysis of online social networks. In *IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pp. 29–42, New York, NY, USA, 2007.

[18] M. Newman. A measure of betweenness centrality based on random walks. *Social Networks*, 27(1):39–54, 2005.

[19] M. E. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69:026113, 2004.

[20] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, June 1998.