Multi-order Egocentric Friendship Network and Its Egocentric Betweenness

Abstract

In this section, we define terms that represent two types of egocentric networks, egocentric networks and multi-layered egocentric networks (Section **??**), as well as the betweenness of a vertex in its ego and x-egocentric networks (Section 3). We then present several properties of the multi-layered egocentric networks (Section **??**). Our betweenness computation algorithm in Section 6 takes advantage of these properties.

Keywords: Egocentric, Friendship Networks, Betweenness Centrality

1. Introduction

In this section, we define terms that represent two types of egocentric networks, egocentric networks and multi-layered egocentric networks (Section **??**), as well as the betweenness of a vertex in its ego and x-egocentric networks (Section 3). We then present several properties of the multi-layered egocentric networks (Section **??**). Our betweenness computation algorithm in Section 6 takes advantage of these properties.

An egocentric network (also called egocentric network) consists of the alters connected to ego, along with the links between ego and alters, and links among alters (Brea L. Perry (2018)).

Our betweenness computation algorithm in Section 3 takes advantage of these properties.

2. Multi-order Egocentric Friendship Networks and Their Betweenness

In this section, we formally define *multi-order egocentric friendship networks*, while we also introduce *multi-order egocentric networks* formally, and then provide the definition of betweenness centrality over such networks.

Table 1: Summary of notation

Symbol	Description
$V_i(v)$	set of i-hop neighbors of vertex v ($V_0(v) = \{v\}$)
$V_{\leq i}(v)$	set of vertices that are at most i hops away from vertex v (i.e., $V_{\leq i}(v) = \cup_{k=0}^{i} V_k(v)$)
$E_i(v)$	set of edges connecting two vertices in $V_i(v)$ ($E_0(v)$=$\emptyset$)
$E_{\leq i}(v)$	set of edges connecting two vertices in $V_{\leq i}(v)$

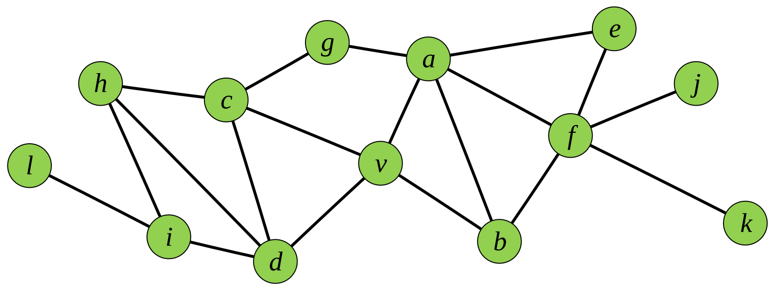
2.1. Definitions

In this paper, we consider a graph $G(V, E)$[1] where V is a set of vertices and E is a set of undirected edges representing social links between vertices. In the literature Marsden (2002); Everett and Borgatti (2005); Nanda and Kotz (2008); Daly and Haahr (2009), given a graph $G(V, E)$ and a vertex $v \in V$, the *ego network* of v is defined as the subgraph of G consisting of v and its 1-hop neighbors (i.e., vertices with an edge to v) as well as the edges between these vertices. Using the notation summarized in Table 1, this ego network can be formally extended to *n-th order ego network* as follows:

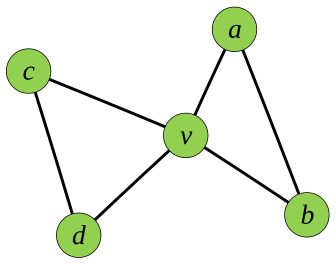
Definition 2.1. Given a graph $G(V, E)$ and a vertex $v \in V$, the *n-th order ego network* of v is defined as $\mathcal{E}_v^n(V_{\leq n}(v), E_{\leq n}(v))$ where $V_{\leq n}(v)$ is the set of vertices whose geodesic distance from v is no longer than n and $E_{\leq n}(v)$ denotes the set of edges between the vertices in $V_{\leq n}(v)$.

In a given graph shown at Fig. 1 (a), $V_{\leq 1}(v) = V_0(v) \cup V_1(v) = \{v, a, b, c, d\}$ and $E_{\leq 1}(v) = \{(v, a), (v, b), (v, c), (v, d), (a, b), (c, d)\}$. The first-order ego network (i.e., just ego network) of vertex v, $\mathcal{E}_v^1(V_{\leq 1}(v), E_{\leq 1}(v))$, is shown in Fig. 1 (b). Egocentric networks well model the relationships/interactions between an actor and others in a social network. However, egocentric networks have the limitation that they do not capture a substantial amount of information if the networks are formed via a way of information diffusion. For example, in Fig. 1 (a), assume that actor v received from actor a the information about a's 1-hop neighbors (i.e., b, e, f, g, and v). Despite this information, the ego network of v cannot record the social links between a

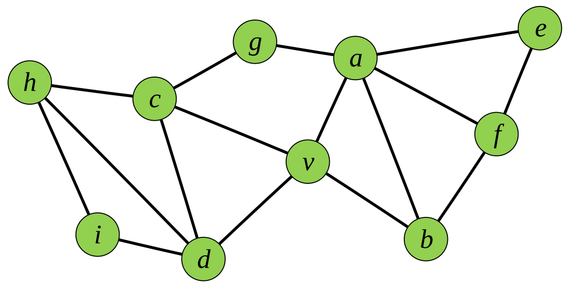
[1]We use the term *actors* and *social links* to refer to individuals, groups or organizations and their relationships in a social network. On the other hand, the graph representing a social network consists of *vertices* and *edges* representing actors and social links, respectively.



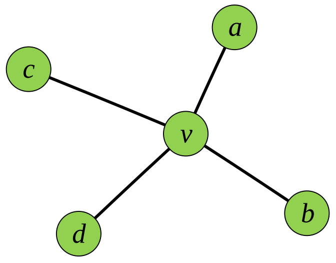
(a) A given network



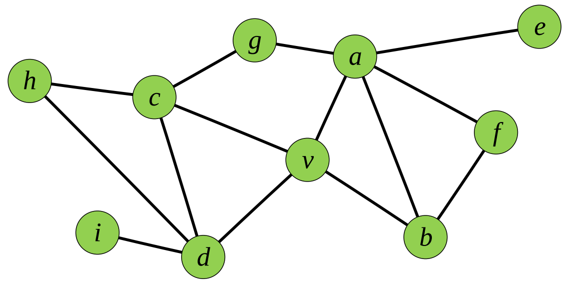
(b) Ego network



(c) Second-order ego network



(d) Egocentric friendship network



(e) Second-order egocentric friendship network

Figure 1: The given network, and the n-order ego and egocentric egocentric friendship networks of vertex v

and e, between a and f, and between a and g since it can represent only the social links between v and each 1-hop neighbor of v, and between two 1-hop neighbors of v. On the other hand, n-th order egocentric networks can hold more information than egocentric networks. For example, the second-order ego network of vertex v, $\mathcal{E}_v^2(V_{\leq 2}(v), E_{\leq 2}(v))$, is shown in Fig. 1 (c). However, it still does not contain all the linkage information of an ego's first and second-order neighbors. For example, f, a second-order neighbor of v, has its neighbors j and k, which does not belong to $\mathcal{E}_v^2(V_{\leq 2}(v), E_{\leq 2}(v))$. If there were a way that all the neighbor information of f can be delivered to v via a or b, the linkage information (e.g., (f, j) and (f, k)) should have been used in any way.

Egocentric friendship networks overcome the above limitation. Egocentric friendship network for an actor consists of the alters directly connected to the actor, along with the links between the actor and the alters. We extend the concept of egocentric friendship network into *n-th order egocentric friendship network*, also called *n-th order f-ego network*, and define it formally as follows:

Definition 2.2. Given a graph $G(V, E)$ and a vertex $v \in V$, the *n-th order f-ego network* of v is $\mathcal{F}_v^n(V_{\leq n}(v), E_{\leq n}(v) - E_n(v))$, where $V_{\leq n}(v)$ is the set of vertices that are at most n hops away from v, $E_{\leq n}(v)$ is the set of edges between vertices that are at most n hops away from v, and $E_n(v)$ is the set of edges between n-hop neighbors of v.

Fig. 1 (d) and (e) show the first-order f-ego network and the second-order f-ego network of v, respectively. $\mathcal{E}_v^n$ has the same vertices with $\mathcal{F}_v^n$, while $\mathcal{F}_v^n$ is different from $\mathcal{E}_v^n$ in that $\mathcal{F}_v^n$ does not contain $E_n(v)$. For example, as shown in Fig. 1 (e), the second-order f-ego network of v does not include the edge sets $E_2(v) = \{(f, e), (h, i)\}$. It is noted that a vertex v can generate its second-order f-ego network by using only neighbor information of its neighbors, along with its first-order f-ego network. As the same manner, for $n = 1, 2, ..., n$, the n-th order f-ego network of an actor v can be constructed with the first-order neighbor and linkage information of vertices in $V_{n-1}(v)$, along with its $(n-1)$th-order f-ego network. Formally, the n-th order f-ego network of a vertex v is defined recursively as follows:

$$\mathcal{F}_v^n = \begin{cases} \mathcal{F}_v^{n-1} + \sum\limits_{v_k \in V_{n-1}(v)} \mathcal{F}_{v_k}^1 & \text{if } n > 1 \\ \mathcal{F}_v^1 & \text{if } n = 1 \end{cases} \tag{1}$$

2.2. Multi-order Ego and F-Ego Betweenness

In the literature Freeman (1979); Everett and Borgatti (2005); Brandes (2001), given a graph $G(V, E)$, the betweenness $B(v)$ of a vertex v is defined as:

$$B(v) = \frac{\sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}}}{(|V|-1)(|V|-2)} \tag{2}$$

where σ_{st} is the number of shortest paths from vertex s to vertex t and $\sigma_{st}(v)$ is the number of those shortest paths that pass through vertex v. In the above definition, the denominator represents the total number of pairs of all vertices except v. It normalizes $B(v)$ to a value between 0 and 1. Given an undirected graph, $\sigma_{st} = \sigma_{ts}$ and $\sigma_{st}(v) = \sigma_{ts}(v)$ for all vertices s, t, and v, so that it is sufficient to find either σ_{st} or σ_{ts} and either $\sigma_{st}(v)$ or $\sigma_{ts}(v)$. The globally computed betweenness based on (2) requires much information about the whole network topology and causes large computational overheads. If the whole network were constructed via a way of information diffusion, a kind of information exchange load should be also very heavy. Therefore, getting local information from neighbor nodes and generating multi-order egocentric friendship networks in a distributed way are practically feasible, and calculating betweenness based on such local information is also acceptable if the betweenness is not much different from the one computed based on the global information of the whole network.

A node v's *n-th order ego betweenness* $B_n^{\mathcal{E}}(v)$ on its ego network $\mathcal{E}_n^v$ is locally computed as follows:

$$B_n^{\mathcal{E}}(v) = \frac{\sum_{s \neq v \neq t \in V_n(v), s<t} \frac{\sigma_{st}^{\mathcal{E},n}(v)}{\sigma_{st}^{\mathcal{E},n}}}{(|V_n(v)|-1)(|V_n(v)|-2)/2} \tag{3}$$

and the *n-th order f-ego betweenness* $B_n^{\mathcal{F}}(v)$ on its f-ego network $\mathcal{F}_n^v$ is also locally obtained as follows:

$$B_n^{\mathcal{F}}(v) = \frac{\sum_{s \neq v \neq t \in V_n(v), s<t} \frac{\sigma_{st}^{\mathcal{F},n}(v)}{\sigma_{st}^{\mathcal{F},n}}}{(|V_n(v)|-1)(|V_n(v)|-2)/2} \tag{4}$$

where $\sigma_{st}^{\mathcal{E},n}$ and $\sigma_{st}^{\mathcal{F},n}$ denote the number of shortest paths from vertex s to vertex t in $\mathcal{E}_v^n$ and $\mathcal{F}_v^n$, while $\sigma_{st}^{\mathcal{E},n}(v)$ and $\sigma_{st}^{\mathcal{F},n}(v)$ denote the number of shortest paths from vertex s to vertex t through vertex v in $\mathcal{E}_v^n$ and $\mathcal{F}_v^n$.

In this paper, we consider the situations where each node computes its betweenness using either its n-th order ego network or n-th order f-ego network, and then uses the result as an estimate of its true betweenness in the entire network.

Table 2: Comparison of betweenness, first-order ego betweenness, and first-order f-ego betweenness for the graph shown in Fig. 5 (a), (b), and (d) (the Pearson correlation is 0.63 between $B(v)$ and $B_n^{\mathcal{E}}(v)$ and 0.90 between $B(v)$ and $B_n^{\mathcal{F}}(v)$, and the Spearman correlation is 0.79 between $B(v)$ and $B_n^{\mathcal{E}}(v)$ and 0.93 between $B(v)$ and $B_n^{\mathcal{F}}(v)$)

Nodes		v	a	b	c	d	e	f	g	h	i	j	k	l
$B(v)$	Value	0.405	0.383	0.124	0.131	0.286	0.000	0.318	0.049	0.030	0.167	0.000	0.000	0.000
	Rank	1	2	7	6	4	10	3	8	9	5	10	10	10
$B_n^{\mathcal{E}}(v)$	Value	0.667	0.750	0.167	0.583	0.333	0.000	0.833	1.000	0.167	0.667	0.000	0.000	0.000
	Rank	4	3	8	6	7	10	2	1	8	4	10	10	10
$B_n^{\mathcal{F}}(v)$	Value	0.383	0.500	0.125	0.269	0.339	0.000	0.524	0.214	0.133	0.400	0.000	0.000	0.000
	Rank	4	2	9	6	5	10	1	7	8	3	10	10	10

Table 2 shows, for every vertex v in Fig. 5, the betweenness ($B(v)$), ego betweenness ($B_n^{\mathcal{E}}(v)$), and x-ego betweenness ($B_n^{\mathcal{F}}(v)$). In this table, for most of the vertices, x-ego betweenness is closer to betweenness than ego-betweenness mainly because it is derived from a larger number of vertices and edges. For this reason, the correlation coefficient (also known as the Pearson correlation coefficient) of x-ego betweenness and betweenness (0.9) is higher than that of ego-betweenness and betweenness (0.63). The advantage of x-ego betweenness over ego betweenness can also be observed in terms of Spearman's rank correlation, which indicates, given two series $\mathcal{X} = (X_1, X_2, \cdots, X_N)$ and $\mathcal{Y} = (Y_1, Y_2, \cdots, Y_N)$, the correlation between $(r_{\mathcal{X}}(X_1), r_{\mathcal{X}}(X_2) \ , \cdots, r_{\mathcal{X}}(X_N))$ and $(r_{\mathcal{Y}}(Y_1), r_{\mathcal{Y}}(Y_2), \cdots, r_{\mathcal{Y}}(Y_N))$, where $r_{\mathcal{X}}(X_i)$ and $r_{\mathcal{Y}}(Y_i)$ represent the rank of X_i in $\mathcal{X}$ and that of Y_i in $\mathcal{Y}$, respectively. In Table 2, Spearman's correlation is 0.93 between x-ego betweenness and betweenness and 0.79 between ego betweenness and betweenness. In Section 4, we further demonstrate the benefit of x-ego betweenness using wireless trace data.

2.3. Properties of Multi-order Egocentric Friendship Networks

In this section, we present four properties of multi-order egocentric egocentric friendship networks. These properties enable efficient multi-order betweenness computation (Section 6). As in Brandes' work Brandes (2001), we denote the *dependency* of vertices s and t on vertex v as $\delta_{st}(v) = \frac{\sigma_{st}(v)}{\sigma_{st}}$. Then, the betweenness of v (Equation (2)) can be computed by adding the dependency values for all pairs of vertices excluding v. To quickly calculate such dependency values in multi-order egocentric egocentric friendship networks, we have identified the properties explained below.

Theorem 2.1. Assume a vertex v, its x-ego network $\mathcal{X}_v$, and its two different 1-hop neighbors s and t (i.e., $s, t \in V_1(v)$ and $s \neq t$). Then, $\delta_{st}(v) = 0$ if there is an edge between s and t (i.e., $\{s, t\} \in E_1(v)$).

Proof. Since $\{s, t\} \in E_1(v)$, $d(s, t) = 1$. Furthermore, $s, t \in V_1(v)$, meaning that $d(s, v) + d(v, t) = 1 + 1 = 2 > d(s, t) = 1$ (i.e., no shortest path from s to t passes through v). Therefore, $\delta_{st}(v) = \sigma_{st}(v)/\sigma_{st} = 0/\sigma_{st} = 0$. □ □

Example 2.2. In Fig. **??**(b), $\delta_{ab}(v) = 0$ since $a, b \in V_1(v)$ and there is an edge between a and b.

Theorem 2.1 allows us to quickly compute dependencies particularly when x-egocentric networks exhibit a strong community structure (i.e., 1-hop neighbors of a node tend to have direct social links between them). In Section 4, we show this benefit using wireless trace data. When Theorem 2.1 cannot be applied to v's 1-hop neighbors s and t (i.e., there is no edge between s and t), the dependency of s and t on v can be computed using the following theorem.

Theorem 2.3. Assume a vertex v, its x-ego network $\mathcal{X}_v$, and its two different 1-hop neighbors s and t (i.e., $s, t \in V_1(v)$ and $s \neq t$). Then, $\delta_{st}(v) = 1/|V_1(s) \cap V_1(t)|$ if there is no edge between s and t (i.e., $\{s, t\} \notin E_1(v)$).

Proof. Since $\{s, t\} \notin E_1(v)$, $d(s, t) > 1$. In this case, $d(s, t) = 2$ due to the path $s - v - t$. Furthermore, (i) the number of shortest paths from s to t (i.e., paths whose length is 2) can be expressed as $\sigma_{st} = |V_1(s) \cap V_1(t)|$. On the other hand, (ii) the path $s - v - t$ is the only shortest path from s to t that passes through v (i.e., $\sigma_{st}(v) = 1$) since the length of that path is 2, which must be smaller than the length of any other path from s to t that passes through v. By (i) and (ii), $\delta_{st}(v) = \sigma_{st}(v)/\sigma_{st} = 1/|V_1(s) \cap V_1(t)|$. □ □

Example 2.4. In Fig. **??**(b), vertices a and c are 1-hop neighbors of v and there is no edge between a and c. Furthermore, $V_1(a) = \{b, e, f, g, v\}$ and $V_1(c) = \{d, g, h, v\}$. Therefore, $\delta_{ac}(v) = 1/|V_1(a) \cap V_1(c)| = 1/|\{g, v\}| = 1/2$.

Just like Theorem 2.1, the following theorem quickly computes the dependency of two vertices on a vertex v. While the former applies to a pair of 1-hop neighbors of v, the latter applies to a pair of a 1-hop or 2-hop neighbor and a 2 hop-neighbor of v.

Theorem 2.5. Assume a vertex v, its x-ego network $\mathcal{X}_v$, a vertex $s \in V_{\leq 2}(v)$, and another vertex $t \in V_2(v)$ such that $s \neq t$. Then, $\delta_{st}(v) = 0$ if $\delta_{sn}(v) = 0$ for a 1-hop neighbor vertex n of vertex t (i.e., for $n \in V_1(t)$).

Proof. Since $\delta_{sn}(v) = 0$ (i.e., no shortest path from s to n passes through v), (i) $d(s,n) < d(s,v) + d(v,n)$. Then, (ii) $d(s,t) \leq d(s,n) + d(n,t) < d(s,v) + d(v,n) + d(n,t)$ by (i). Furthermore, $n \in V_1(t)$ and $t \in V_2(v)$, meaning that vertex n is either a 1-hop or a 3-hop neighbor of v. In $\mathcal{X}_v$, however, any vertex including n is at most 2 hops away from v. For this reason, n is a 1-hop neighbor of v. Since $d(v,n)$=$d(n,t)$=1 and $d(v,t) = 2$, (iii) $d(v,n) + d(n,t) = d(v,t)$. By (ii) and (iii), $d(s,t) < d(s,v) + d(v,t)$ (i.e., no shortest path from s to t passes through v). Therefore, $\delta_{st}(v) = \sigma_{st}(v)/\sigma_{st} = 0/\sigma_{st} = 0$. □ □

Example 2.6. In Fig. **??**(b), $b \in V_{\leq 2}(v)$ and $g \in V_2(v)$. Furthermore, for a 1-hop neighbor a of g, $\delta_{ba}(v) = 0$ (Theorem 2.1). Therefore, by Theorem 2.5, $\delta_{bg}(v) = 0$.

Given vertices $s \in V_{\leq 2}(v)$ and $t \in V_2(v)$ such that $s \neq t$, Theorem 2.5 cannot be applied if $\delta_{sn}(v) > 0$ for every 1-hop neighbor n of t. In this case, the dependency of s and t on v can be obtained using the following theorem.

Theorem 2.7. Assume a vertex v, its x-ego network $\mathcal{X}_v$, a vertex $s \in V_{\leq 2}(v)$, and another vertex $t \in V_2(v)$ such that $s \neq t$. Then, $\delta_{st}(v) = \bar{H}(\{\delta_{sn}(v) : n \in V_1(t)\})$ if $\delta_{sn}(v) > 0$ for every 1-hop neighbor n of vertex t, where $\bar{H}(\{\delta_{sn}(v) : n \in V_1(t)\})$ denotes the *harmonic mean* computed over $\{\delta_{sn}(v) : n \in V_1(t)\}$.

Proof. We prove this theorem considering the following two cases. [Case I. s is a 1-hop neighbor of v (i.e., $s \in V_1(v)$)] Let n_i ($i = 1, 2, \cdots, m$) denote the ith 1-hop neighbor of t. Then, each n_i is a 1-hop neighbor of v since, in $\mathcal{X}_v$, any 2-hop neighbor (including t) of v can be connected only to a 1-hop neighbor of v (Definition **??**). Thus, by Theorem 2.3, $\delta_{sn_i}(v) = 1/|V_1(s) \cap V_1(n_i)|$. Since $|V_1(s) \cap V_1(n_i)|$ represents the number of shortest paths from s to n_i (i.e., σ_{sn_i}) and each n_i has an edge to t, the total number of shortest paths from s to t can be expressed as (i) $\sigma_{st} = \sum_{i=1}^{m} |V_1(s) \cap V_1(n_i)| = \sum_{i=1}^{m} 1/\delta_{sn_i}(v)$. On the other hand, the path $s - v - n_i - t$ is the only shortest path from s to t via both v and n_i since the length of the path is 3 and any other path from s to t via both v and n_i must be longer. For this reason, (ii) there are m shortest paths from s to t via v (i.e., $\sigma_{st}(v) = m$). By (i) and (ii),

$\delta_{st}(v) = \frac{m}{\sum_{i=1}^{m} 1/\delta_{sn_i}(v)} = \bar{H}(\delta_{sn}(v))$. [Case II. s is a 2-hop neighbor of v (i.e., $s \in V_2(v)$)] Let n_i ($i = 1, 2, \cdots, m$) denote the ith 1-hop neighbor of t. Let also p_j ($j = 1, 2, \cdots, k$) denote the jth 1-hop neighbor of s. In $\mathcal{X}_v$, 2-hop neighbors (including s and t) of v can be connected only to a 1-hop neighbor of v. For this reason, any shortest path from s to t must contain a shortest path from p_j to n_i for some j and i (i.e., $\sigma_{st} = \sum_{i=1}^{m} \sum_{j=1}^{k} |V_1(n_i) \cap V_1(p_j)|$). For a pair of i and j, on the other hand, the path $n_i - v - p_j$ is the only shortest path from n_i to p_j via v and so is the path $t - n_i - v - p_j - s$ (i.e., $\sigma_{st}(v) = mk$). Therefore,

$$\begin{aligned}
\delta_{st}(v) &= \frac{mk}{\sum_{i=1}^{m} \sum_{j=1}^{k} |V_1(n_i) \cap V_1(p_j)|} \\
&= \frac{mk}{\sum_{i=1}^{m} \sum_{j=1}^{k} 1/\delta_{n_i p_j}(v)} \\
&= \frac{m}{\sum_{i=1}^{m} \frac{\sum_{j=1}^{k} 1/\delta_{n_i p_j}(v)}{k}} \\
&= \frac{m}{\sum_{i=1}^{m} 1/\delta_{n_i s}(v)} \\
&= \frac{m}{\sum_{i=1}^{m} 1/\delta_{s n_i}(v)} \\
&= \bar{H}(\delta_{sn}(v)).
\end{aligned}$$

□

Example 2.8. In Fig. **??**(b), $V_1(h) = \{c, d\}$. By Theorem 2.3, $\delta_{ac}(v) = \frac{1}{2}$ and $\delta_{ad}(v) = 1$. Therefore, by Theorem 2.7, $\delta_{ah}(v) = \bar{H}(\{\delta_{an}(v) : n \in V_1(h)\}) = \bar{H}(\{\delta_{ac}(v), \delta_{ad}(v)\}) = \frac{2}{\frac{1}{\frac{1}{2}} + \frac{1}{1}} = \frac{2}{3}$.

3. Betweenness in Multi-order Friendship Networks

In this section, we provides an overview of previous work on betweenness computation (Section 3.1), presents our algorithm for quickly computing nodal betweenness in multi-order friendship networks (Section 3.2), and analytically shows the benefits of our algorithm (Section 3.3).

3.1. Previous work on betweenness computation

To the best of our knowledge, the fastest algorithm for computing the betweenness of every vertex in a given graph is developed by Brandes Brandes (2001). Given an unweighted graph, the Brandes algorithm performs a breadth first search to compute the number of shortest paths from any pair of two vertices. It then derives the betweenness of each vertex by aggregating the previously computed count values backwards along the edges. Given an unweighted graph $G(V, E)$, this Brandes algorithm takes $O(|V||E|)$ time.

The proposed node-centric betweenness of a vertex can be obtained by calculating the betweenness of that single vertex in the corresponding multi-order friendship network. Applying the Brandes algorithm to the multi-order network, however, results in finding the betweenness of every vertex in the network (i.e., the Brandes algorithm incurs overhead to find needless information). Section 3.2 provides our algorithm that quickly computes the nodal betweenness by computing the betweenness of only the target vertex and by skipping computations according to the properties of multi-order friendship networks explained in Section 2.3.

3.2. Our nodal betweenness computation

3.3. Algorithm Complexity

4. Evaluation

5. EEGO

6. Computation

S. P. B. Brea L. Perry, Bernice A. Pescosolido, Egocentric Network Analysis: Foundations, Methods, and Models (Structural Analysis in the Social Sciences), Cambridge University Press, 2018.

P. V. Marsden, Egocentric and Sociocentric Measures of Network Centrality, in: Social Networks, volume 24, 2002, pp. 407–422.

M. Everett, S. P. Borgatti, Ego Network Betweenness, Social Networks 27 (2005) 31–38.

S. Nanda, D. Kotz, Localized Bridging Centrality for Distributed Network Analysis, in: Proc. of IEEE ICCN, 2008.

E. M. Daly, M. Haahr, Social Network Analysis for Information Flow in Disconnected Delay-Tolerant MANETs, IEEE Trans. Mobile Comput. 8 (2009) 606–621.

L. C. Freeman, Centrality in Social Networks: Conceptual Clarification, Social Networks 1 (1979) 215–239.

U. Brandes, A faster algorithm for betweenness centrality, Journal of Mathematical Sociology 25 (2001) 163–177.