A Fast Algorithm for Nodal Betweenness Centraility

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 social network analysis, the betweenness of an actor indicates the extent to which that actor is between all other actors within the network (Freeman (1979)).

Calculating the betweenness of each node, however, requires finding all of the shortest paths between every pair of nodes in the given network. Since carrying out this task in a large wireless network will incur prohibitively expensive computational costs, techniques for estimating betweenness have been developed Daly and Haahr (2009); Marsden (2002); Everett and Borgatti (2005); Nanda and Kotz (2008); Pantazopoulos et al. (2013). In these techniques, each node identifies its *ego network*, a logical network consisting of that node, its 1-hop neighbors, and all links between these nodes. Then, each node calculates, as an estimate of its betweenness in the entire network, its betweenness only in its ego network, thereby saving both network and computational resources.

Table 1: Summary of notation

Symbol	Description
$V/\{v\}$	set of all vertices except v
$V_i(v)$	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. Multi-order Egocentric Friendship Networks and Their Betweenness

3. Betweenness in Multi-order Friendship Networks

4. Evaluation

5. EEGO

6. Computation

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

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

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.

P. Pantazopoulos, M. Karaliopoulos, I. Stavrakakis, On the Local Approximations of Node Centrality in Internet Router-level Topologies, in: 7th Int'l Workshop on Self-Organizing Systems (IFIP IWSOS'13), 2013.

Algorithm 1 *nodal_betweenness*$(G(V,E), v)$

```
Input: a graph G(V, E) and a vertex v(∈ V)
Output: betweenness of v
create an 2-dimensional array D[1, 2, ..., |V|][1, 2, ..., |V|]
sum ← 0
for p : 1 to u do
    for q : p + 1 to u do
        D[p][q] ← dependency1(p, q, N)
        sum ← sum + D[p][q]
    end for
    for q : u + 1 to u + w do
        D[p][q] ← dependency2(p, q, N, D)
        sum ← sum + D[p][q]
    end for
end for
for p : u + 1 to u + w do
    for q : p + 1 to u + w do
        D[p][q] ← dependency2(p, q, N, D)
        sum ← sum + D[p][q]
    end for
end for
return 2·sum / ((u+w)(u+w−1))
```

Algorithm 2 *dependency*$1(p,q,N)$

```
Input: p, q, N[0, 1, ..., u + w]
if q ∈ N[p] then
    return 0                          ▷ by Theorem 1
else
    return 1/| N[p] ∩ N[q] |          ▷ by Theorem 2
end if
```

Algorithm 3 *dependency*$2(p, q, N, D)$

Input: p, q, $N[0, 1, ..., u + w]$, $D[1, 2, ..., u + w][1, 2, ..., u + w]$
$C \leftarrow []$
for each $r \in N[q]$ **do**
  if $p < r$ **then**
    $\tau = D[p][r]$
  else
    $\tau = D[r][p]$
  end if
  if $\tau == 0$ **then**
    return 0 ▷ by Theorem 3
  else
    append τ to C
  end if
end for
return $\bar{H}(C)$ ▷ by Theorem 4