Distributed Identification of Overlapping Network Communities

Zoran Nikoloski

2006
Distributed Identification of Overlapping Network Communities

Zoran Nikoloski

Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic

nikoloski@kam.mff.cuni.cz

Abstract. We present the concept of overlapping communities defined with respect to a novel node-centrality index, called attenuation centrality. The attenuation centrality combines two types of node-centrality indices, distance-based and feedback, to describe the importance of a node in a deterministic spreading process on a graph $G$. The relationships between attenuation centrality and the existing centrality indices are theoretically derived. An efficient distributed algorithm for evaluating the attenuation centrality index is designed and analyzed. This algorithm is, further, used in devising two distributed algorithms for identification of overlapping communities—one, for the granular community structure, and the other, for the coarse community structure with respect to a general node-centrality index. In addition, we develop a procedure for determining the hierarchical decomposition into overlapping communities in the context of the attenuation centrality index.

1 Introduction

The problems of identifying communities in large graphs and analyzing their properties are important tasks in fields, such as: computer science, for designing efficient synoptic Web search engines, social studies, for identifying tight human relationships, biology, for analyzing metabolic and protein networks, and other areas that deal with optimization of large infrastructures. The notion of community is quite general and, depending on the context, can be synonymous to: module, class, cohesive group, and cluster. In computer science, the problem of community identification is sometimes considered identical to the extensively studied problem of graph partitioning [6]. In the community identification problem, for a given definition of community, graph $G$, and a node $u \in V(G)$, is that of finding the maximal community to which $u$ belongs.

Qualitatively, a community in a graph $G = (V, E)$ is defined as a subset of nodes $S \subseteq V(G)$ that induces a subgraph $G[S]$ of size (number of edges) greater than the size of the cut $(S, V \setminus S)$. In other words, community is a subset of nodes $S \subseteq V(G)$ that induces locally-dense connected subgraph $G[S]$. The reader is directed to [10] for a comprehensive survey of existing definitions for community.

It is not surprising that there is a large body of algorithms intended at identifying communities in networks, due to the growing number of applications. The algorithms for community identification can be divided into two classes: (1) algorithms based on an index (measure) [8] and (2) flow-based
algorithms [1]. Giving a survey of the algorithms is not in the scope of this paper. We point out that our algorithms, developed in Section 4, belong to the class of algorithms based on a node-index.

It is natural that communities in a given real-world network are overlapping. For instance, communities in the World Wide Web would overlap, since a given web-page may deal with several topics. To our knowledge, there is only one approach for determining overlapping communities, and is based on clique percolation [11]. As the problem of finding cliques of certain order in a given graph $G$ is NP-hard, the need for more efficient algorithms for identification of overlapping communities, amenable to rigorous analysis and applicable in the context of large networks, is even more pronounced.

**Our contribution:** We design and analyze two algorithms *distributed detection of overlapping communities*. Our approach is based on the process of deterministic spreading of an agent on a given graph $G$. In this process, an agent, on node $u$, spreads to all neighbors of $u$ that do not have agents residing on them. The importance of a node $u \in V(G)$ in this process can be naturally defined as the inverse of the sum of the distances to the remaining nodes. However, this definition does not account for the individual distance of a given node from $u$. Thus, we propose a novel centrality index, called *attenuation centrality*. Our centrality index integrates the feedback centrality indices defined in social network analysis with the distance-based centrality indices from graph theory. By using the idea of "information comes to those who need it," we devise a distributed algorithm for calculating this node-centrality index. This algorithm, in turn, is used for devising two distributed algorithms—one for the granular, and the other, for the coarse community structure of a given graph.

The paper is structured as follows: The definition of the attenuation centrality index is given in Section 2, where contrived examples and theoretical results for the attenuation centrality index are given. The details of the distributed algorithm for calculating the attenuation centrality index are given in Section 3. The distributed algorithms for determining (1) the granular community structure of a given graph and (2) coarse community structure are described and analyzed in Section 4. In addition, we design a procedure for calculating the hierarchical decomposition of a given graph into overlapping communities, in the context of the attenuation centrality index.

## 2 Attenuation Centrality Index

The concept of centrality indices, proceeding from social network analysis [12], has already been applied in some problems in computer science, such as: propagation of computer viruses [9], understanding and managing the evolution of networks with desired properties, and community identification [3]. Given a graph $G$, the centrality of a node specifies the importance (role) that node has in a specified (random) process taking place on the network (the reader is directed to [8] for a survey). For instance, one may seek the centrality of a node in network processes, such as: transport of commodities, spread of rumors, or voting.

One of the oldest centrality measure, defined by Katz in 1953 [7], is based on the idea of *feedback*, *i.e.* a node is more central the more its neighbors are. In other words, the longer the path between two nodes $u$ and $v$, the smaller the impact of node $u$ on the centrality of node $v$. The formal mathematical formulation of the Katz's centrality index is the following: Given a directed simple graph $G$ with adjacency matrix $A$ and a damping factor $\alpha > 0$ for the distance between any two nodes, the centrality of node $u$, denoted by $c_f(u)$, is given by $c_f(u) = \sum_{k=1}^{\infty} \sum_{v \in V(G)} \alpha (A^k)_{uv}$. The matrix $A^k$ contains the number of walks of length $k$ between every pair of nodes. The value of the damping factor $\alpha$ should be
smaller than the reciprocal value of the largest eigenvalue of $A$ for the infinite sum, in the expression for $c_f(u)$, to converge [7].

The centrality of a node can also be specified in terms of the distance between that node and chosen subset of nodes. The formal description will require the following notation: Given a graph $G = (V, E)$, let $l(u, v)$ represent the distance between the nodes $u$ and $v$. Furthermore, let $N_i(u)$ denote the set of nodes at distance $l$ from node $u$.

The simplest of the distance-centrality indices is the node-degree: For a node $u$, the degree, denoted by $d(u)$, is defined as: $d(u) = |N_1(u)|$. One can also consider the eccentricity of a node $u$, defined as the maximum over the distances between $u$ and any other node in $V(G)$, i.e. $e(u) = \max \{l(u, v) | v \in V(G)\}$. Haryary and Hage [4] proposed a distance-centrality measure based on the eccentricity, defined as: $c_e(u) = \frac{1}{e(u)}$. The closeness centrality index for a node $u$, instead of using only the maximum distance, employs the sum of all distances between the node $u$ and any other node in $V(G)$, as follows: $c_c(u) = \frac{1}{\sum_{v \in V(G), v \neq u} l(u, v)} = \frac{1}{\sum_{i=1}^{\infty} |N_i(u)|}$. Note that all of the distance-centrality indices just described, except the node-degree, cannot be employed for the case of disconnected graphs. In this case, the eccentricity of every node is $\infty$, while the Hage and Haryary’s index and the closeness centrality index are both equal to 0.

Here, we define the attenuation centrality index, to remedy the problems that (1) the feedback node-centrality indices based on the number of paths passing through a node cannot be efficiently computed for large networks and (2) the distance-centrality indices cannot be applied to disconnected graphs. The formal definition of the attenuation centrality index appears below:

**Definition 1.** Given a graph $G = (V, E)$ and a node $u \in V(G)$, the attenuation centrality index of $u$, denoted by $c_a(u)$, is given by:

$$c_a(u) = \sum_{v \in V(G), v \neq u} \frac{1}{l(u, v)} = \frac{1}{\sum_{i=1}^{\infty} |N_i(u)|}.$$

Note that the attenuation centrality index integrates the distance between two nodes with the idea of feedback: The larger the distance, the smaller the influence of the nodes’ centralities. Moreover, the attenuation centrality index can be easily applied to disconnected graphs without encountering the problem of infinite distance. Next, we give the relation between the attenuation centrality index and the described distance-centrality indices:

**Proposition 1.** Given a connected, undirected graph $G = (V, E)$ and a node $u \in V(G)$: (1) $c_a(u) > c_e(u)$ and (2) $c_a(u) \geq (n-1)^2 c_c(u)$.

**Proof:** The claim in (1) follows from $c_a(u) = \frac{\sum_{v \in V(G)} l(u, v)}{\sum_{v \in V(G), v \neq u} l(u, v)} > \frac{1}{\max \{l(u, v) | v \in V(G)\}} = \frac{1}{e(u)}$. The claim (2) follows from the relation between the arithmetic and harmonic means:

$$\sum_{v \in V(G), v \neq u} \frac{1}{l(u, v)} \geq \frac{(n-1)^2}{\sum_{v \in V(G), v \neq u} \frac{1}{l(u, v)}} = \frac{(n-1)^2}{\sum_{v \in V(G), v \neq u} l(u, v)}.$$ The equality holds for the complete graph $K_n$. 
The quality of a centrality index is expressed via its ability to differentiate two nodes based on their importance in a given (random) process on a graph $G$. Note that the eccentricity and the attenuation centrality index are qualitatively different: Consider the graph $G$ shown in Figure 1, below; nodes $b$ and $c$ have equal eccentricity (or value 2); however, they have different attenuation centrality indices; namely, $c_a(b) = 4.5$, while $c_a(c) = 4$. Next, we determine the existence of a graph $G$ where there are at least two nodes $u, v \in V(G)$ for which $c_e(u) = c_e(v)$ does not imply $c_a(u) = c_a(v)$; i.e.,

$$
\sum_{i=1}^{e(v)} i |N_i(u)| = \sum_{i=1}^{e(v)} i |N_i(v)| \quad \text{and there exists } 1 \leq k \leq \min \{ e(u), e(v) \} \quad \text{such that}
$$

$$
|N_k(u)| \neq |N_k(v)|.
$$

The smallest such graph appears in Figure 2, where $c_e(a) = c_e(d) = \frac{1}{8}$ and $c_a(a) = 3.5$, $c_a(d) = 3.88$.

**Figure 1.** Two nodes ($b$ and $c$) have equal eccentricity but different attenuation centrality.

**Figure 2.** Smallest graph where two nodes ($a$ and $d$) of equal closeness centrality have different attenuation centrality.

### 3 Distributed Algorithm for Attenuation Centrality Index

From Definition 1, it is clear that attenuation centrality indices for all nodes of $G$ can be calculated with a modification of the centralized all-paths-shortest-paths algorithm [1]. However, such algorithms become inefficient when the input graph is of a large order and size, typical for real-world networks. Here, we devise a distributed algorithm for determining the attenuation centrality indices for the nodes of a given graph $G$. Unlike the existing distributed algorithms for the all-pairs-shortest-path problem, which guarantee that at most two nodes will contain the distance matrix of the entire graph $G$ with optimal number of transmitted messages [5], our algorithm deals only with the problem of distributed calculation of the attenuation centrality index without the need of storing the entire distance matrix at a certain node.

A network consists of a set of communication devices that know the communication links incident on them. The network is modeled by a graph $G = (V, E)$, where nodes represent communication devices and edges represent communication links. Let $n$ denote the number of nodes and $m$, the number of edges. Let $D$ denote the diameter of the graph $G$. No node knows the topology of $G = (V, E)$, and no common memory is shared among nodes. For the duration of the algorithm, it is assumed that no node or edge can be removed. Each node $u$ has a unique identifier $ID(u)$. A packet is a set of elementary messages that can be sent together. Packets are only transmitted between two adjacent nodes. We assume that the system can be synchronized. Let $T$ be the maximum time needed for a packet to traverse an edge. Thus, any packet broadcast at time $t$ by node $u$ will be received by the neighbors of $u$ at most by time $t + T$. The local computation at a node is assumed to take time
negligible to the time required to transmit a packet. With an introduction of a global clock, it is now possible to synchronize the system. At a given time \( t_0 \), each node broadcast packets to all of its neighbors. Every node waits for the packets from its neighbors during the time interval \( T \). The new packets will be sent at time \( t_0 + T \).

The **computational complexity** of a distributed algorithm is defined as the number of packets sent through the algorithm. The **time complexity** is defined as the number of time units necessary for termination of the algorithm. The **packet complexity** of a distributed algorithm is defined as the number of packets that have to be sent through the algorithm.

**Algorithm 1** Distributed calculation of attenuation centrality indices at node \( u \)

//initial step at time \( t_0 \)
1: for every neighbor \( v \) of \( u \)
2: send packet \((ID(u),ID(u),0)\) to node \( v \)
3: end for
4: \( c_u \leftarrow 0 \)
//start receiving and calculating
5: while any packet has been received in time \( t_i \)
6: if packet \((x,y,l)\) received and \( x \) is not hashed
7: hash \( x \) and \( l \)
8: \( c_u \leftarrow c_u + \frac{1}{l} \)
9: forward packet \((x,ID(u),l+1)\) to all neighbors of \( u \)
10: end if
11: end while

**Figure 3.** Distributed algorithm for attenuation centrality indices

In the following, packets that carry new data, not stored at a node, will be called relevant. Packets that contain data already stored at a node \( u \) will be called backfiring, and will not be forwarded (through broadcasting) to the neighbors of \( u \). Each packet contains three elementary messages—the identifier of the origin of the packet, the identifier of the forwarding node, and the number of edges traversed by the packet. Upon receiving a packet \((x,y,l)\), the receiving node \( u \) builds a new packet in which: the first message (origin of the packet) is not modified, the identifier of the forwarding node is changed from \( y \) to \( ID(u) \), and, finally, the number of traversed edges, \( l \), is incremented by 1. Each node \( u \) is endowed with a hash table in which it stores the origin of every packet (unless the packet is backfiring) and the distance between the origin and \( u \), extracted from a relevant packet. If a node receives more than one packet with the same origin, it does not forward the packet to its neighbors, and discards it. From the last, it is clear that flooding of the network down a BFS tree is performed from every node—the root of the BFS tree. Finally, every node stores the sum of the reciprocal values of the distances stored in the hash table, which is the attenuation centrality index of the node. A formal description of the algorithm is given in Figure 3, above: In lines 1 – 3, packets are broadcast to all neighbors. Line 8 specifies the calculations of the attenuation centrality index, while in line 9 – 10 a new packet is designed and broadcast to all neighbors.

The time complexity is in the order \( O(D) \), since each packet traverses a path of length bounded by \( D \). To calculate the communication complexity, we observe that every node must send one packet over
every edge. Therefore, the communication complexity is \(O(n^2 m)\). This is optimal, since every node needs to know the distance to every other node to calculate the attenuation centrality index. The number of packets sent to determine the BFS tree from a node is in the order \(O(Dm)\), since each node can receive and send no more than \(3(n^2 + n)\) elementary messages.

4 Distributed Algorithms for Community Identification

In this section, we define two types of community structure—granular and coarse, design distributed algorithms for their calculation, and analyze their complexity. To expand the scope of the granular and the coarse community structures, we propose a general framework for distributed algorithms that determine these structures based on a given centrality index.

4.1 Granular community structure

Given a graph \(G = (V, E)\) and a node \(u \in V(G)\), the granular community structure for node \(u\) specifies the subset of nodes with which \(u\) forms increasing (decreasing) communities with respect to an arbitrary centrality index \(c\). Consider the following:

**Definition 2.** An increasing community for a given node \(u \in V(G)\) is the set of nodes \(S(u)\) that induces a connected subgraph \(G[S]\), such that for every pair of nodes \(x, y \in S(u)\), \(c(y) > c(x)\) if and only if \(l(u, x) < l(u, y)\). Decreasing community is defined equivalently.

To define the concept of overlapping boundary, we first define a boundary and an internal boundary for a subset \(S\) of nodes.

**Definition 3.** Given a graph \(G = (V, E)\) and a set \(S \subseteq V(G)\), the boundary of the set \(S\), denoted by \(\partial S\), is \(N(S) \setminus S\), where \(N(S) = \bigcup_{v \in S} N_1(v)\).

**Definition 4.** Given a graph \(G = (V, E)\) and a set \(S \subseteq V(G)\), the internal boundary of the set \(S\), denoted by \(\sigma S\), is \(N(\partial S) \cap S\). If \(\partial S = \emptyset\), then \(\sigma S = V(G)\).

The overlapping boundary of an increasing community with respect to the centrality index \(c\) is, then, defined as follows:

**Definition 5.** Given a graph \(G = (V, E)\) and an increasing community \(S \subseteq V(G)\) with respect to a centrality index \(c\), the overlapping boundary of the set \(S\), denoted by \(\psi S\), is given by:

\[
\psi S = \{ u \mid u \in \partial S \land \exists v, (u, v) \in E(G) : c_u(u) = c_v(v) \} \cup \\
\{ w \mid w \in \sigma S, \forall v \in \partial S, (v, w) \in E(G) : c_u(w) > c_v(v) \}.
\]

Overlapping boundary of a decreasing community with respect to a centrality index \(c\) is defined equivalently.

Finally, an overlapping increasing (decreasing) community is the union of the increasing (decreasing) community and its overlapping boundary. These notions are illustrated in Figure 4, below, with attenuation centrality: The initial node is node \(k\), seeking increasing community, i.e., only nodes with attenuation indices greater than \(c_u(k) = 4.95\), satisfying Definition 2, are included. The nodes in the community are \(S_{inc}(k) = \{d, g, h, i, j\}\). The boundary of this community is
\[ \partial S_{\text{inc}}(k) = \{c,e,f,k\}, \] while the internal boundary is \[ \sigma S_{\text{inc}}(k) = \{d,g,j\}. \] From Definition 5, it follows that the overlapping boundary is \[ \psi S_{\text{inc}}(k) = \{d,k\}. \] The increasing overlapping community for node \( k \) is given by the subset of nodes \( \{d,g,h,i,j,k\} \).

Algorithm 2, shown in Figure 5, below, calculates the increasing (overlapping) community for a given node \( u \) in a distributed fashion. The setting for this algorithm is the same as for Algorithm 1. However, the packets contain different elementary messages: initially, each node chooses which type of overlapping community needs to determine (increasing or decreasing). This information is contained in an identifier denoted by \( \text{INC} \), for increasing, or \( \text{DEC} \), for decreasing. A packet initiated at node \( u \), then, contains four messages: the identifier \( \text{ID}(u) \) of the origin, the identifier of the forwarding node, the type of community, and the centrality index of the forwarding node \( c(v) \). A node that receives this message, stores the identifier of the origin and the type of community.

In the case of increasing type of community, upon receiving a packet \((x,y,\text{INC},c(y))\), node \( v \) forwards two packet \((x,v,\text{INC},c(v))\) and \((x,v)\), if \( c(v) > c(y) \), and only one packet \((x,v)\), otherwise. Upon receiving a packet \((u,x)\), node \( u \) stores the value of \( x \). Thus, after termination of the algorithm, every node knows the nodes that are in its own increasing (decreasing) community, and also knows the type and the communities to which it belongs. The time, communication, and packet complexities for this algorithm are the same with those of Algorithm 1 for calculation of the attenuation centrality index.

**4.1.1 Hierarchical decomposition into maximal overlapping communities**

The output of Algorithm 2 can also be used in determining a hierarchical decomposition into maximal overlapping communities. An increasing (decreasing) community is maximal if addition of another node violates the increasing (decreasing) criteria. The hierarchical decomposition with respect to the attenuation centrality indices can be obtained by the greedy approach: The nodes of the graph are ordered such that the pairs of attenuation indices and degrees of nodes form a non-decreasing sequence, i.e. node \( u \leq v \) if and only if \( c(u) \leq c(v) \) and \( d(u) \geq d(v) \). At step \( i \), the maximal community and its overlapping boundary are determined for the first node in the ordered sequence, say node \( u_j \) whose degree in \( G_i \) is not 0. The graph \( G_i \) is obtained iteratively: \( G_1 = G \),
\[ E(G_i) = E(G_{i-1}) \setminus E\{G[S_{\text{inc}}(u_j)]\} \]. The procedure is repeated until no node of degree greater than 0 exist in the sequence (i.e. until \( E[G_i] \neq \emptyset \)).

**Algorithm 2 Distributed calculation of increasing overlapping community for node \( u \)**

//initial step at time \( t_0 \)
1: for every neighbor \( v \) of \( u \)
2: send packet \((ID(u),ID(u),INC,c(u))\) to node \( v \)
3: end for
//start receiving and calculating
4: while any packet has been received in time \( t_i \)
5: if packet \((x,y,INC,c(y))\) received and \( x \) is not hashed
6: hash \( x \)
7: if \( c(u) > c(y) \)
8: forward packet \((x,y,INC,c(u))\) to all neighbors of \( u \)
9: end if
10: forward packet \((x,v)\) to all neighbors of \( u \)
11: end if
12: if packet \((u,x)\) received and \( x \) is not hashed
13: hash \( x \)
14: end if
15: end while

**Figure 5.** Algorithm for calculating increasing (overlapping) community for node \( u \)

Suppose that the procedure is repeated \( p \) times, and let the collection of overlapping communities be \( S = \{S_1,S_2,\ldots,S_p\} \). To obtain the hierarchical decomposition, the following procedure is repeated: find the first pair of communities, \( S_i \) and \( S_j \), for which \( S_i \cap S_j \neq \emptyset \). The collection is modified to \( \{S_1,\ldots,S_{i-1},S_i \cup S_j,\ldots,S_{j-1},S_{j+1},\ldots,S_p\} \). This procedure is repeated until there are two communities in \( S \) whose intersection is not an empty set (thus, the step is repeated at most \( (p-1) \) times). Figure 6, below, illustrates the hierarchical decomposition when attenuation centrality is used as centrality index, where progressively thicker lines indicate higher levels in the hierarchy.

### 4.2 Coarse community structure

The coarse community structure of a graph \( G \), with respect to a centrality index \( c \), is given by the collection of maximal increasing overlapping communities for the set, \( I \), of seed nodes. The set of seed nodes is obtained by the following procedure, which strongly relies on the centrality index employed: First, the nodes are ordered in a non-increasing order. We say that \( u \geq v \) if and only if \( c(u) \geq c(v) \) and \( d(u) \geq d(v) \). The algorithm proceeds deleting nodes in a greedy fashion from the ordered sequence until only isolated nodes remain. Nodes creating ties are deleted at one step. After each step of node(s) deletion, the centrality index is re-computed and another sequence of ordered nodes is obtained. The remaining isolated nodes comprise the set of seed nodes \( I \), and are used as seeds (initial nodes) in determining the increasing overlapping communities. This algorithm can be used in determining the community structure of graphs for which most of the nodes have equal centrality indices.
To re-compute the attenuation centrality index, one can use Algorithm 1, with additional constraint that the nodes can be in two states active or inactive. An active node sends, receives, and forwards packets. An inactive node receives but does not forwards any packets. To determine the set of inactive nodes, one only has to calculate the maximum over centrality indices in a distributed fashion, such that every node is knows it. Algorithm 3, in Figure 7, below, produces the desired result. Note that if a node $u$ does not receive any packets after sending the initial one, it means that it is isolated, and thus belongs to the set of seed nodes. The node $u$ can then start executing Algorithm 2 to determine its overlapping increasing community. Figure 7 shows the execution of Algorithms 1 and 3, until the set $I$ is obtained, and the result of running Algorithm 2 on every node in $I$: Nodes $d$ and $i$, having equal degrees and attenuation indices, are deleted first; followed by nodes $c$ and $j$, in the second step, nodes $b$ and $k$, in the third, and nodes $h$ and $e$ in the fourth and last step. There are four overlapping communities: \{a, b, c, d, g\} with seed $a$, \{c, d, e, f, k\} with seed $f$, \{g, h, i, j\} with seed $g$, and \{i, j, k, l\} with seed $l$.

5 Conclusion

We presented the concept of overlapping communities and its relation with a novel centrality index called attenuation centrality. The attenuation centrality, defined here, determines the importance of a node in the process of deterministic spreading on a given graph $G$. We established theoretical connections between our and the existing distance-based centrality indices. Furthermore, we designed and analyzed a distributed algorithm for calculating the attenuation index. The concepts of granular and coarse community structure with respect to a general centrality index are also algorithmically defined. In addition, two distributed algorithms for determining the community structure of a graph $G$ are designed.
and theoretically analyzed. Experimental results on real-world scale-free networks will be presented in the full version of the paper.

**Algorithm 3** Distributed calculation of inactive nodes

//initial step at time $t_0$
1: for every neighbor $v$ of $u$
2: \quad send packet $(ID(u), c(u))$ to node $v$
3: end for
//start receiving and calculating
4: inactive ← true
4: while any packet has been received in time $t_i$
5: \quad if packet $(x, c(x))$ received and $x$ is not hashed
6: \quad \quad hash $x$
7: \quad \quad if $c(u) < c(y)$
8: \quad \quad inactive ← false
9: \quad end if
10: \quad forward packet $(x, c(x))$ to all neighbors of $u$
11: end if
12: end while
//change status after all nodes have been hashed
13: if inactive = true stop forwarding

Figure 6. Determining the status of a node

6 References