Approximation of the Maximal $\alpha$—Consensus Local Community detection problem in Complex Networks

Patricia Conde Céspedes, Blaise Ngonmang and Emmanuel Viennet

Université Paris 13
L2TI

COMPLEX NETWORKS (LIP6) -January 22nd, 2016

This work is supported by REQUEST and Open Food System projects.
Table of contents

1. Introduction and objective
2. Proposed Solutions
3. Evaluations
4. Conclusions and perspectives
The problem, we want to tackle, can be summarized as follows:

**Objective:**

"Given a node \( n_0 \) of a graph \( G(V, E) \) and a parameter \( \alpha \) \((0 < \alpha < 1)\), the purpose is to find the biggest \( \alpha \)-consensus community \( C(n_0) \) containing \( n_0 \)."

That is, the biggest group of nodes \( C(n_0) \) where each node is connected to more than a proportion \( \alpha \) of the other nodes. We call this rule, the Condorcet's majority rule. Mathematically, this problem can be formalized as follows:

\[
\begin{align*}
\max_C & \quad |C| \\
\text{subject to} & \quad n_0 \in C \\
\text{and} & \quad |\Gamma(n_i) \cap C| > \alpha(|C| - 1), \forall n_i \in C. \\
\end{align*}
\]

(1)

where \( \Gamma(u) \) is the neighborhood of a node \( u \).
Examples of $\alpha$–cliques

$\alpha$–cliques for nodes 5 and 28 of Zachary karate club networks for $\alpha = 0.5$. 
Some applications of $\alpha$–cliques

- In SNA to find groups of people who all know each other.
- In social recommendation.
- Link prediction.
- In electrical engineering to analyze communications networks.
- In bioinformatics, many problems have been modeled using cliques.
Definitions and notations

$D$: set of nodes identified as members of $C(n_0)$
$S$: set of neighbors of nodes in $D$ (possible candidates to enter $D$).

The local community starts with one node $D = \{n_0\}$, at each iteration one or more nodes from $S$ are added to $D$. Each algorithm has its own criterion to choose the best candidate (s) from $D$ to enter $S$. 
Table of contents

1. Introduction and objective
2. Proposed Solutions
3. Evaluations
4. Conclusions and perspectives
SOLUTION 1, the RANK-GAIN algorithm

At each iteration one node from $S$ is chosen to be added to $D$. The new node is chosen according to the gain $g_n$ caused by its addition to $D$, given by:

$$g_n = \ell_n - \alpha \times |D|$$

The RANK-GAIN algorithm chooses the node whose gain is maximum and positive to enter $D$. If there are ties the new node is chosen according to the number of its neighbors in $S$. If there are still ties the new node is chosen randomly.
Advantages of the *RANK-GAIN* algorithm

**Figure:** Advantages of choosing the new node according to the number of its neighbors in $S$
Introduction to SOLUTION 2: the internal degree

When looking for the local community of node 0, the algorithm will give $C(n_0) = \{0, 1, 2, 3, 4\}$ whereas 0 does not respect the rule anymore.
SOLUTION 2, the *RANK-GAIN+* algorithm

- At any iteration, the maximum possible size of $D$, $D_{\text{max}}$, given the minimal internal degree of nodes in $D$, $d_{\text{min}}$ will be: $D_{\text{max}} = \lceil \left( \frac{d_{\text{min}}}{\alpha} \right) \rceil$

- If $D_{\text{max}}$ has been reached the eventual new node is chosen from the set of neighbors of nodes with $d_{\text{min}}$, denoted $S_{\text{min}}$. 

![Diagram of the community detection process]

**Approximation of the Maximal $\alpha$—Consensus Local Community detection problem**
Shortcomings not solved by the \textit{RANK-GAIN+} algorithm

**Shortcoming 1:**

![Diagram showing optimal and suboptimal communities for node 0](image)

**Shortcoming 2:**

![Diagram showing optimal and suboptimal communities for node 5](image)

**Shortcoming 3:**

![Diagram showing optimal and suboptimal communities for node 1](image)
Introduction to SOLUTION 3

The gain is not the most important thing when choosing the new node to enter $D$. Reminder:

**Objective:**

Given a node $n_0$ in the graph, the purpose is to find the biggest community containing $n_0$ where each node is connected to more than $\alpha\%$ of the other nodes.

So, the most important thing is the size of the community. We are not looking for the densest community but for the biggest community where all nodes must respect the majority rule.

Why not to let enter more than one node at each iteration? Why not to prioritize the number of neighbors in $S$ rather than the gain?
Solution 3: The $RANK\text{-}NUM\text{-}NEIGHS$ (RNN) algorithm

To face this problem we propose an algorithm, called $RANK\text{-}NUM\text{-}NEIGHS$, that prioritizes the entry of nodes with the highest number of connections in $S$ over the gain and allows to enter one node or more at each iteration.

1. It prioritizes the number of common neighbors over the gain when choosing a new node (or more nodes) to enter the local community.

2. At each iteration more than one node might enter the local community.

If a node $n$ has a negative gain, the number of neighbors $n$ needs to enter $D$ is:

$$x = \left\lfloor \left( \alpha \times |D| - \ell_n \right) \left( 1 - \alpha \right) \right\rfloor + 1$$
Solution 3, The *RANK-NUM-NEIGHS*: example of the 4 situations

For each situation the optimal solution is the set of red nodes on the right.
Comparison of the RANK-GAIN+ and the RANK-NUM-NEIGHS algorithms
Table of contents

1. Introduction and objective
2. Proposed Solutions
3. Evaluations
4. Conclusions and perspectives
Sizes of the detected communities vs $\alpha$, comparison to the global Louvain method

Local Community sizes for artificial graphs

Network Size: 1000

Network Size: 5000

Network Size: 10000

Local Community sizes for real networks

karate

football

polblogs

polbooks

Approximation of the Maximal $\alpha$—Consensus Local Community detection problem
Community-Sizes vs $\alpha$, comparison to the *QUICK* method

Local community sizes, comparison with QUICK method

$N = 5000$, $\alpha = 0.7$

$N = 5000$, $\alpha = 0.8$

$N = 5000$, $\alpha = 0.9$

- **RANK–NUM–NEIGHS**
- **Quasi clique (QC)**

Approximation of the Maximal $\alpha$—Consensus Local Community detection problem
Densities of the detected communities versus $\alpha$

### Density of local communities for artificial graphs

<table>
<thead>
<tr>
<th>Network Size</th>
<th>Method</th>
<th>Density</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>0.5 ZC</td>
<td></td>
</tr>
<tr>
<td>5000</td>
<td>0.6 ZC</td>
<td></td>
</tr>
<tr>
<td>10000</td>
<td>0.7 ZC</td>
<td></td>
</tr>
</tbody>
</table>

### Density of communities for real networks

<table>
<thead>
<tr>
<th>Network</th>
<th>Method</th>
<th>Density</th>
</tr>
</thead>
<tbody>
<tr>
<td>karate</td>
<td>0.5 ZC</td>
<td></td>
</tr>
<tr>
<td>football</td>
<td>0.6 ZC</td>
<td></td>
</tr>
<tr>
<td>polblogs</td>
<td>0.7 ZC</td>
<td></td>
</tr>
<tr>
<td>polbooks</td>
<td>0.8 ZC</td>
<td></td>
</tr>
</tbody>
</table>

Approximation of the Maximal $\alpha$—Consensus Local Community detection problem
Stability of the algorithm

\[ J(n) = \frac{|\bigcap_{i=1}^{10} C(n)_i|}{|\bigcup_{i=1}^{10} C(n)_i|} \]

Approximation of the Maximal $\alpha$—Consensus Local Community detection problem
Execution time

The computer used for the experimentations has a Quad Core processor running at 3.33 GHz and 15GB RAM. The algorithm is written in python.

- **alpha = 0.4**
- **alpha = 0.5**
- **alpha = 0.6**
- **alpha = 0.7**
- **alpha = 0.8**
- **alpha = 0.9**

**Mixing parameter $\lambda$**
- 0.1
- 0.2
- 0.3

**Time in seconds**

Approximation of the Maximal $\alpha$—Consensus Local Community detection problem
Bound on the optimal solution

$C^*(n_0)$ is the maximal $\alpha$–consensus community of node $n_0$, so the optimal solution, we have the following bounds:

- Bound $B_0(n_0) = \left\lceil \left( \frac{d(n_0)}{\alpha} \right) \right\rceil$.

- Bound $B_1(n_0) = \min \left( \max_{n \in \Gamma(n_0)} B_0(n), B_0(n_0) \right)$.

- Bound $B_2$ found iteratively.

- $|C(n_0)^*| \leq B_2(n_0) \leq B_1(n_0) \leq B_0(n_0)$.

- Example: for node 1 of Zachary Karate Club $B_0(1) = 32, B_1(1) = 20$ and $B_2(1) = 10$. The solution found by RNN is 8.

Exemple for the calculation of bound $B_2$
Bound on the optimal solution

Histogram of the difference

Histogram of the difference between the upper bound $B_2$ and the results of RNN algorithm for real networks.

Approximation of the Maximal $\alpha$—Consensus Local Community detection problem
Table of contents

1. Introduction and objective
2. Proposed Solutions
3. Evaluations
4. Conclusions and perspectives
Conclusions and perspectives

- We presented a **novel method**, called *RANK-NUM-NEIGHS*, for the problem of mining the maximal local $\alpha-$ consensus community of a given node of a network. Starting from a baseline method, our method is introduced to address some shortcomings not solved by the previous ones.

- Our method contains **two important characteristics**. First, it prioritizes the nodes who have the biggest number of neighbors in the neighborhood of the community, over the gain. Second, it allows to let enter more than one node at each iteration.

- The *RANK-NUM-NEIGHS* gives good results in real and generated networks in terms of size, stability and execution time. It also performs better in terms of quality than the existing method *QUICK* and than global community detection *LOUVAIN* method.