ComTector algorithm

Community Detection in Large-Scale Social Networks

Alex Delbono

alex.delbono@mail.polimi.it

Politecnico di Milano

April 14 2016

Overview

Table of Contents

POLITECNICO MILANO 1863

Introduction

The algorithm was named *ComTector* because it is a Community Detector algorithm.

The algorithm was designed by:

- Nan Du - dunan@bupt.edu.cn
- Bin Wu - wubin@bupt.edu.cn
- Xin Pei - peixin@tseg.org
- Bai Wang - wangbai@bupt.edu.cn
- Liutong Xu - xliutong@bupt.edu.cn

Introduction

POLITECNICO
MILANO 1863

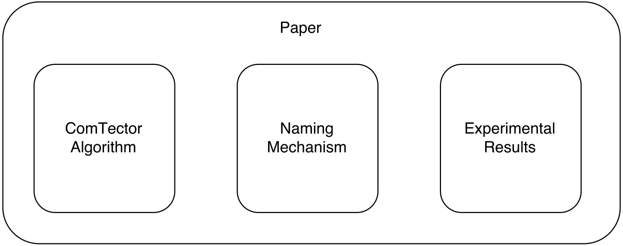
The algorithm which will be presented is more efficient and accurate than the other known algorithm in community detection because:

- It is focused on *Large-Scale Social Networks*
- It takes advantages of the *sparsity* of the Social Adjacent Matrixes
- It considers the *overlapping* nature of the social relationships

There is no need of a-priori knowledge of the network communities.

Presentation structure

This is the structure of the paper and the presentation will follow it.



In the paper the ComTector presentation is structured using the bottom-up approach, while in this presentation it is top-down, in order to make an initially clearer global vision of the algorithm.

Table of Contents

Definitions

POLITECNICO
MILANO 1863

Definition 1 : $S \subseteq V(G)$,$\forall u, v \in S, u \neq v$, such that $(u, v) \in E$, then S is a clique in G. If any other S' is a **clique** and $S' \supseteq S$ iff $S' = S$, S is a **maximal clique** of G.

Definition 2 : For a given vertex v, $N(v) = \{u | (v, u) \in E(G)\}$. We call $N(v)$ the set of all **neighbors** of v. Given a set $S \subseteq V(G)$, $N|_S = \bigcup N(v_i) - S$, $v_i \in S$, $N|_S$ is the set of all neighbors of S.

Definition 3 : Let $Com(G)$ be the set of all components in G. The giant component is denoted by C_G and $M(C_G)$ is the set of all the maximal cliques in C_G. We use $V_M \subseteq V(G)$ to represent the set of all vertices covered by $M(C_G)$.

Definitions

Definition 4 : Let $P_0, P_1, ..., P_{n-1}$ be the subgraph of G such that $\forall P_i, P_j, V(P_i) \cap V(P_j) = \emptyset$, and $V(P_0) \cup, ..., V(P_{n-1}) = V(G)$. For any pair of P_i and P_j, if $|E(P_i)| > |(N|_{P_i} \cap P_j)|$, P_i is defined as a **community** of G.

Definition 5 : Given vertex $v_i \in V_M$, define $C_i = \{S|S \in M(C_G), v_i \in S\}$ to be the set of all maximal cliques containing v_i, and C the set of all C_i's. $\forall C_i, C_j \in C$, if $\frac{|C_i \cap C_j|}{|C_j|} \geq f$, which is a threshold to describe the extent to which C_i overlaps with C_j, we call C_j is contained in C_i, denoted by $C_j < C_i$. If C_i is not contained by any other element in C, C_i is called the **kernel** of G and v_i is the **center** of C_i

Structure

The following is the algorithm of the structure of ComTector presented in the paper at the end of the relative section.

Algorithm 5 ComTector(G)

1. Read Graph G
2. $Com(G) \Leftarrow$ all components of G
3. $M(C_G) \Leftarrow$ all maximal cliques in C_G
4. FilterOutKernels(C,f)
5. DeDuplication(K)
6. AssignVertex(K)
7. AdjustDivision(K)
8. return $K \cup (Com(G) - C_G)$

Filter out kernels

Now we generate the kernel K as shown in the algorithm.

Algorithm 1 FilterOutKernels(C,f)

```
K ⇐ ∅
sort C by the descending order of |C_i|, C_i ∈ C
{core stores the centers of the filtered out kernels}
core ⇐ ∅
for C_i ∈ C do
  contained ⇐ C_j, j ≠ i, C_j < C_i
  independent ⇐ k, k ≠ i, C_k ≮ C_i
  delete C_i from C
  C ⇐ C − contained
  for s ∈ C_i do
    if s ∩ (independent ∪ core) ≠ ∅ then
      delete s from C_i
    end if
  end for
  if C_i ≠ ∅ then
    K ⇐ C_i
  end if
  core ← v_i
end for
return K
```

DeDuplication

Now we have to choose only one kernel for those vertices that belong to multiple kernels.

Algorithm 2 DeDuplication(K)

```
I_K ⇐ ∅
for k_i ∈ K do
    for k_j ∈ K, i < j do
        I_K ← I_K ⋃ (k_i ∩ k_j)
    end for
end for
for v ∈ I_K do
    remove v from all the kernels except for the one having
    the maximum distance
end for
```

Assign vertex

Next step is to assign to the closest kernel all the remaining vertices.

Algorithm 3 AssignVertex(K)

for $v_i \in V_K$ **do**
    v_i is marked as old
end for
$V_E \leftarrow$ vertices not marked as old in $\bigcup N(k_i) - V_K$
while $V_E \neq \emptyset$ **do**
    for $v_i \in V_E$ **do**
        assign v_i to its closest kernel k_i
        v_i is marked as old
    end for
    V_E'$\leftarrow \emptyset$, V_E'$\leftarrow$ vertices not marked as old in $N|_{V_E}$
    $V_E \leftarrow \emptyset$, $V_E \leftarrow V_E$'
end while

Adjust division

This is the last part of ComTector algorithm:

Algorithm 4 AdjustDivision(K)

```
calculate ΔQ from pairs of connected communities
while maximal ΔQ > 0 do
    select the maximal ΔQ
    join the pair of communities with the maximal ΔQ
    update the ΔQ matrix
end while
```

Table of Contents

POLITECNICO MILANO 1863

Naming Mechanism

POLITECNICO
MILANO 1863

Now we have found the communities, we can be interested in knowing:

- the kind of relationship among the vertices
- the most important characteristics of the community
- the common attributes of the nodes

For this reason the authors propose an algorithm to extrapolate this information.

Naming Mechanism

The main idea is to take advantage of the social structure of the network in which we have several hubs.

They are **central entities** in the communities and they put important impacts on the overall formation and development of the given community.

We can order them using some types of centrality such as:

- degree
- betweenness
- Page Rank
- closeness
- eigenvector

CentralEntityResolution

Now we sort the nodes according to descending order and we choose the central ones according to the following algorithm.

Algorithm 6 CentralEntityResolution(C,p)

```
{C is the given community and p is a threshold value}
Calculate the centrality of each vertex v_i in C
v_0, v_1, ..., v_{n-1} is arranged by the descending order of
    their centrality c_0, c_1, ..., c_{n-1}
i ⇐ 0
while i < n − 1 do
    if (c_i − c_{i−1}) / (c_i − c_{n−1}) > p then
        return {c_k | 0 ≤ k ≤ i}
    else
        i ⇐ i + 1
    end if
end while
```

NamingCommunity

Finally we can associate their attributes to each community.

Algorithm 7 NamingCommunity(R,$Center$,p_1,p_2)

```
{R is the attribute set, Center is the community set, p1,p2 are two threshold values}
if |Center| > 1 then
    for each attribute a ∈ R do
        if a is discrete then
            C_a ⇐ the most frequent value of a among the central entities
        else
            C_a ⇐ average value of a over the central entities
        end if
    end for
else
    for each attribute a ∈ R do
        C_a ⇐ value of a
    end for
end if
```

NamingCommunity

```
for each attribute $a \in R$ do
    if $a$ is discrete and $F_{C_a} > p_1$ then
        $a$ is selected as the key attribute
        return $C_a$
    else
        if $\frac{C_a - A_a}{A_a} < p_2$ then
            $a$ is selected as the key attribute
            return $C_a$
        end if
    end if
end for
```

Table of Contents

Experimental results

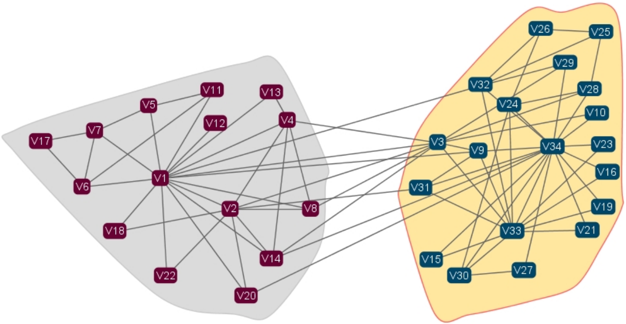
POLITECNICO
MILANO 1863

The paper reports three applications of the ComTector algorithm:

- Zachary Karate Club
- American college football
- Scientific collaboration

Zachary Karate Club

It is characterised by 34 vertices and 78 edges among members of the club based on their social interactions.



The algorithm detects two communities which exactly match with the result of Zachary's study.

American college football

This network represents the schedule of Division I games for the 2000 season.

It consists of **115 vertices** and **616 edges** which are the representations of football teams and regular season games among them respectively.

115 teams are divided into **12 conferences** containing around 8 to 12 teams each.

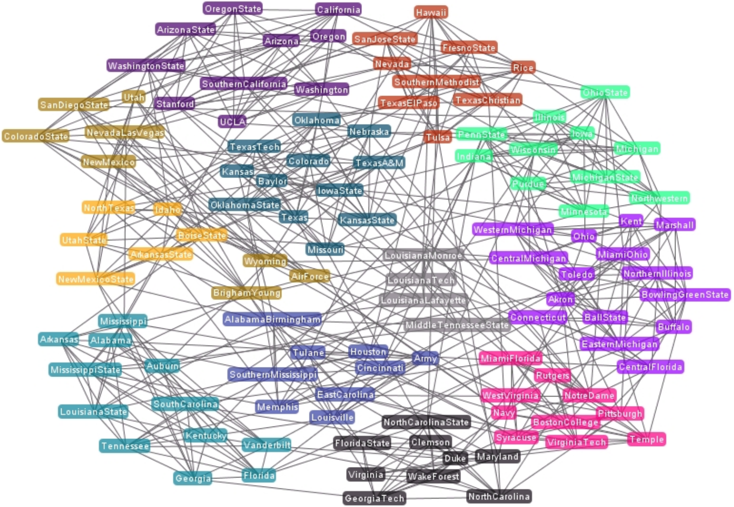
Games are more frequent between members of the same conference than between members of different conferences.

Apparently, each conference can be considered as one community of the network.

American college football

POLITECNICO
MILANO 1863

The algorithm shows an accuracy of 93.8%.



Scientific collaboration

The data of the collaboration network is obtained according to the 1990 published papers from the year 1998 to 2005 indexed by SCI, EI and ISTP in Beijing University of Posts and Telecommunications(BUPT).

Each **author corresponds to a vertex** of the network and **there is an edge between two vertices if the two correspondent authors have collaborated in a paper**.

For each author it is created **a set of attributes** using the most relevant term in their papers.

The **Periphery** area is an independent small component of the network, and the **Core** area corresponds to the giant component.

Scientific collaboration

POLITECNICO
MILANO 1863