Ego-Community Evolution Tracking in Instant Messaging Networks

Ahmed Ould Mohamed Moctar(✉) and Idrissa Sarr

Department of Mathematics and Computer Science, Cheikh Anta Diop University, Dakar - Fann, BP 5005, Dakar, Senegal
amed.mohameden@gmail.com, idrissa.sarr@ucad.edu.sn

Abstract. Community detection is one of the most topics that are covered by social network analysis researchers. Early works focused mainly on partitioning networks into several global communities before they target communities evolution over time. The main drawback of such approach is the difficulty to obtain the entire network on which we may process a set of algorithms to track evolution. As a result, many researchers focused on detecting and tracking local dynamic communities.

Basically, detecting and tracking local dynamic communities is a process that moves from a set of nodes to build community structure followed by a structure evolution tracking. A particular type of local communities is called "ego-communities". Unlike local communities, ego-centered ones have the advantage of being able to expand the community according to the neighborhood step of interest node. They allow, among others, to better identify and track the network elements activities.

Existing ego-community detection algorithms do not support directed networks and are designed for static networks. This failure led us to propose a new solution allowing to detect and track ego-community evolution in directed, weighted and dynamic social networks. We present some illustrative examples to explain the working principle of our solution.

Keywords: Ego-community · Dynamic networks · Instant messaging

1 Introduction

The development of on-line social media has created many opportunities to communicate, access, and share information from anywhere and at anytime. The kind of application such as *Viber*, *WhatsApp*, *Imo*, *Line*, as well as *Facebook* affords plenty of possibilities for getting in touch with friends, colleagues, and relatives at every moment with real-time messages, photos, videos, etc. Data collected from those applications integrate the time from which users send and/or receive data. Therefore, it is worthwhile to build a social network based on instant messaging content to find out "who talk to whom" and/or "who is closed to whom". Such a network that we call Instant Messaging Network (IMN) can be represented as a graph where individuals are the nodes and their interactions the links. In

C. M. F. Kebe et al. (Eds.): InterSol 2018, LNICST 249, pp. 13–22, 2018.
https://doi.org/10.1007/978-3-319-98878-8_2

such a network, finding out the sub-groups of individuals, *a.k.a* communities, is worthwhile. In fact, detecting communities within a network affords opportunities to place network elements in classes associated with specific needs or characteristics. For example, identifying the group of people who had a contact with an Ebola patient helps to identify the target either to be monitored or to be informed about the exposed danger. However, in a dynamic social network, such a detection may be challenging at many points. For instance, how to detect changes whitin a community structure and when they occurs is not trivial.

In this study, we rely on IMN which is an illustration of dynamic social networks. To detect dynamic communities in IMN, we use the snapshots approach which relies on capturing a series of snapshots whose each one is aggregated over a time window. For each snapshot, one may apply a set of processes and finally compare the resulting communities with the ones obtained in the other snapshots. This strategy is widely used in dynamic community studies in order to track changes over time. In short, we plan to face three challenges, namely:

1. Data gathering and interpretation;
2. Time window size management;
3. Monitoring the evolution of dynamic communities.

The first challenge is the extraction and interpretation of data from instant messaging platforms, which provide huge and various information with a high velocity. Actually, in many studies, data contents are extracted and analyzed for unveiling knowledge. Important parts of these approaches rely on semantic rules, and/or more simply, on keywords to identify the network structures (nodes and their links). However, when it comes to deal with dynamic networks, it is worthwhile to integrate the fact that links can either vanish or intensify over time. That is, everything (nodes as well as links) must be set as dynamic. Moreover, based on the exchange flows, a link direction may switch over time and leads to many changes in the community structure. Therefore, instant messaging networks should be modeled as a directed and weighted dynamic graph in order to deal with the frequency and the direction of interactions over time.

The second challenge is how to manage the size of time window which has a real impact on a community structure evolution since it determines the data that belong to each snapshot. In other words, a bad time windows setting can lead to miss interesting structure changes. To manage the time widow size, we define a quality function in order to quantify the changes that the network undergoes during a given period. If the quality difference between two times exceeds a given threshold, this means that the network has undergone a considerable changes, and then so, we make a new snapshot.

The third challenge is the monitoring of dynamic community evolution. To handle this aspect, we propose a solution, based on snapshots approach, for detecting and tracking the evolution of dynamic ego-community over time. It consists in selecting some nodes of interest among the most important nodes in network. Then, it seeks to track changes in the ego-communities over time.

In brief, our solution proceeds in 3 phases. First, it decomposes the network evolution into several snapshots. Next, it detects in each snapshot the static

ego-communities based on the algorithm we proposed in [5]. Finally, it applies a dynamic tracking algorithm to interpret the communities evolution over time. In our best knowledge, there is a glaring lack of studies that target the challenges we pointed out while we count hundreds of works in community detection topic.

The remainder of this paper is organized as follows. First, we present, in Sect. 2, the challenges and problem statement as well as the data gathering and interpretation. Next, we discuss, respectively, in Sects. 3 and 4 time window size management and monitoring the evolution of dynamic communities. Then, we present, in Sect. 5, some illustrative examples in order to illustrate the working principle of our solution before concluding in Sect. 6.

2 Network Set-Up

To model the instant messaging network, we use a set of nodes and links attributes explained below. In this perspective, a set of nodes constitutes the network initial composition, and when an existing node communicates with a new element, the latter it's added as a new node in the network, provided with the corresponding link. The links weights indicate the communications number between nodes. An interaction between two existing nodes implies the appearance of a new link. This is how the network grows over time.

2.1 Challenges and Problem Statement

The design of a community detection method unveils several questions that one can summarize in two categories, namely, supported networks and detected community properties. Several works have been proposed in order to find a method able to face the challenges pointed out above [1–4,6]. However, in our best knowledge, there is no method allowing to handle all these issues.

Our goal is to propose a solution that supports dynamic, directed and weighted networks and allows to detect and track the evolution of overlapping local ego-communities.

2.2 Nodes Attributes

Each node is identified by the following information:

- Identifier: this attribute uniquely identifies each user (*e.g.,* phone number, ID of the user profile, etc.);
- Instant of appearance: this attribute indicates the moment where a node comes into the network.

2.3 Links Attributes

Each link is characterized by:

- Direction: It shows who initiates the interaction and who reacts;

– Duration: It represents the time that lasts each interaction. If the communication is instant, we set a short-time threshold beyond which we consider that the interaction is over;
– Weight: We compute weight depending on a set of durations so that the link weight during a time window is equal to interactions number multiplied by the sum of interactions durations.

2.4 Operations of Network Evolution

Related works analyze the networks evolution according to a macroscopic vision. That's to say, they focus only on community changes. In fact, existing works suffer from two weaknesses: *(i)* they neglect the communication intensity and are limited to the topological changes; *(ii)* they interpret the community evolution according to a global scope, in other words, they can't interpret the microscopic changes of network (nodes and links changes).

The approach we propose overcomes these two weaknesses as we introduce the communication intensity aspect into the definition of network evolution operations. In addition, we define 5 nodes/links evolution operations, namely:

– Incoming node (IN): a node comes into the network if it receives a communication from a node of network;
– Vanishing node (VN): a node is considered as vanishing if its communication rate is equal to zero during a given time window;
– Incoming link (IL): it reflects the first time two nodes start exchanging;
– Vanishing link (VL): a link is considered as vanishing if the communication between nodes ends or also if one of the link endpoints vanishes;
– Growing link (GL): a link with an increasing weight, thanks to the higher communication rate of the corresponding nodes at a given time window.

3 Time Window Size Management

In this section, we propose a new method allowing to decompose the dynamic network evolution into several snapshots, each of which represents the state of the network during a given period. Our method works periodically as it consists of capturing snapshots by period. Note that the period is not fixed, it depends on degree of changes network over time.

The method we propose consists of two parts:

1. a variant, whose scope is global, of the quality function that we proposed in [5]. This variant serves to measure the state of network during a given period taking into account the links cohesion and the communication intensity between nodes;
2. a strategy, based on the quality function, to decompose the dynamic network evolution into several snapshots.

In the following, we present, in Sect. 3.1, our global quality function. Then, we explain, in Sect. 3.2, how we decompose the evolution of dynamic networks.

3.1 Network Quality

Let $\mathcal{N}$ a dynamic network and $\mathcal{N}_t$ a network capture at time t. To get an idea of the communication intensity between nodes within $\mathcal{N}_t$, we compute the inverse of the weights sum of all links $\frac{1}{\sum_{w_{\mathcal{N}_t}}}$. The intuition behind is that the more the communication is intense within $\mathcal{N}_t$ (high value of weights), the more $\frac{1}{\sum_{w_{\mathcal{N}_t}}}$ tends to 0.

Regarding to the internal cohesion of $\mathcal{N}_t$, we take again the idea of the local quality function: the more the topological structure of the network at instant t approaches a clique, the more the snapshot is considered dense. Therefore, the proportion $\frac{|V_{\mathcal{N}_t}|}{|E_{\mathcal{N}_t}|}$ allows to evaluate at what level the snapshot $\mathcal{N}_t$ is cohesive. $|V_{\mathcal{N}_t}|$ designate the nodes number of $\mathcal{N}_t$ and $|E_{\mathcal{N}_t}|$ the links number.

This, our global quality function is defined as follows:

$$\psi(\mathcal{N}_t) = \frac{1}{\sum_{w_{\mathcal{N}_t}}} \times \frac{|V_{\mathcal{N}_t}|}{|E_{\mathcal{N}_t}|} \tag{1}$$

Note that we choose the multiplication between tow parts of the quality function for weighting the communication intensity to the cohesion between nodes. The overall complexity of $\psi(\mathcal{N}_t)$ is approximately equal to $O(n + 2m)$.

3.2 Dynamic Network Evolution

The optimal size of the time window should enable us to decompose the network evolution into several snapshots, each of which includes an interesting number of changes. To this end, we propose a strategy that works in two phases:

1. First, we rely on the previous global quality function to measure the changes that network has undergone during a given period;
2. Next, we model the changes rate across a threshold ε. If the quality difference of the network between two instants exceeds the threshold value, it means that the network has been considerably changed.

Formally, we consider that the network has undergone a considerable change if the difference between its quality at time t and that at instant $t + 1$ becomes greater than a given threshold ε:

$$|\psi(\mathcal{N}_t) - \psi(\mathcal{N}_{t+1})| \geqslant \varepsilon \tag{2}$$

Note that increasing the threshold value implies the abundance of changes between two instants. In this paper, we are not interested in the way that the right threshold should be determined. The choice of threshold value will be addressed in our future work.

To predict the next instant $t+1$, we use a time series (built from the quality scores over time) that we expose in another paper.

4 Tracking the Changes of Dynamic Ego-Community

The question we have to address first is how do we track the changes: should we do it in a continuous manner or a discontinuous one? In our case, we decide to use the discontinuous fashion since we do not aim to have the trace of every single change but the overall change in some particular points. The first problem raised is to define these particular points. After creating the snapshots, we detect, on each one, the community structure using a static ego-community detection algorithm that we detail in another paper. Its working principle consists in optimizing a quality function that allows to evaluate the relevance of an ego-community according to its internal cohesion, on the one hand, and the communication intensity between its nodes, on the other hand.

Basically, we take a snapshot for each time point and see how the ego-communities evolve over time. To this end, we apply our static ego-community detection algorithm in a sequential manner on each snapshot. The outcome of the snapshot at t is compared with the one of $t + 1$ in order to identify changes if ever.

As in the literature, the operations indicating the evolution of dynamic communities are categorized in seven classes: *growth*,*contraction*, *continuing*, *division*, *merging*, *birth* and *death*. In opposite with the existing solutions, we do not just limit our analysis on finding the class of the evolution, but also we clarify what kind of changes in terms of nodes (IN or VN), links (IL and VL), as well as communication intensity (GL). The reason of doing so is to be able to explain what factors underline the evolution. In our context, this strategy is very useful since an evolution of a community is not always conducted by the topological aspects but with also the behavior of a node regarding to an ego node during a time window.

Moreover, we found interesting to introduce vanishing node (VN) in order to identify nodes that stay inactive for a long time. However, an ego node cannot be considered as a VN since it can stop communicating with its neighbors that keep exchanging between them. However, a community disappears when the number of its nodes is lesser than a fixed number. For example, if we consider that a community is composed at least of n nodes, then an ego community will disappear if we have less than $n - 1$ nodes.

5 Application Case

In this section, we present, first of all, an illustrative example in order to clarify the role of our static ego-community detection algorithm. Next, we show an example of dynamic ego-community tracking over time. In these examples, we detect 2-steps ego-communities. Let $C_{\{u,2\}}$ the 2-steps ego-community of interest node u and $S_{\{u,2\}}$ the seed used to build the ego-community.

5.1 Static Ego-Community Detection

On Fig. 1, we present the initial structure of a dynamic, oriented and weighted network, consisting of 33 nodes and 61 links. The weights values are shown on

links. As for the interest nodes, we choose 4 thanks to their central positions, namely, the nodes 1, 10, 17 et 27, denoted, respectively, by a, b, c et d.

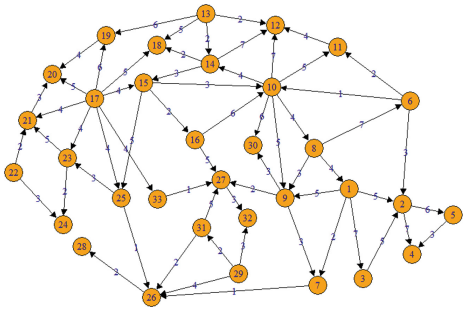


Fig. 1. Initial structure of a dynamic network, oriented and weighted.

Table 1 shows the 2-steps a's ego-community detection procedure. In this procedure, we considered that the seed of a's ego-community includes the whole 2-steps neighborhood. Then, at each iteration, we calculated the affinity degree[1] of the ego's neighbors and we chose the neighbor having the smallest value of affinity degree. Such node is removed from the community if its removing decreases the quality score; otherwise, we keep it and move on to the next neighbor. Following this process, we noticed that the removing of nodes 26, 27 and 10 implies a decrease in the quality score unlike the other nodes. In Fig. 1, we see that node 26 has been removed since it is loosely linked to the nodes of a's community. We also note that nodes 10 and 27 are more connected to the outside as to nodes of a's community, hence the reason for their removing.

After building the 2-steps a's ego-community, the algorithm repeats the same process to detect the ego-communities of nodes 10, 17 and 27. Figure. 2 shows the detected ego-communities. Note that a node can belong to several ego-communities as our algorithm supports overlap.

5.2 Tracking Ego-Communities Evolution

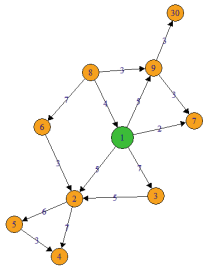
In this part, we illustrate our method of tracking the evolution of dynamic ego-communities. To this end, we consider two snapshots captured, respectively, at the time t and $t+1$, represented on Fig. 3. The snapshot t represents the initial state of the evaluation network already shown in Fig. 1 and whose communities are presented individually on Fig. 2. We present on Fig. 3a all detected ego-communities in snapshot t to highlight the overlap aspect.

On Fig. 3b, we find that the microscopic changes (relative to nodes or links) that took place at time $t+1$ are:

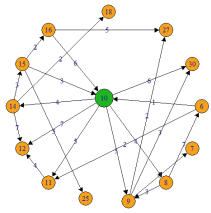
[1] To select the ego's neighbors, we proposed a new metric allowing to classify the nodes based on the number of their neighbors in ego-community as well as on their weights of adjacent links. This metric is detailed in another paper.

Table 1. Illustration of the procedure of 2-steps a's ego-community detection.

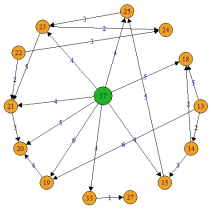
Community current nodes	Selected node	Affinity degree	Quality score	Decision
$S_{\{a,2\}} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 26, 27, 30\}$			0.930	—
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 26, 27, 30}	26	0.02	0.903	Removed
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 27, 30}	27	0.025	0.876	Removed
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 30}	10	0.173	0.809	Removed
{1, 2, 3, 4, 5, 6, 7, 8, 9, 30}	30	0.166	0.842	Kept
{1, 2, 3, 4, 5, 6, 7, 8, 9, 30}	9	0.261	0.144	Kept
{1, 2, 3, 4, 5, 6, 7, 8, 9, 30}	7	0.111	0.922	Kept
{1, 2, 3, 4, 5, 6, 7, 8, 9, 30}	8	0.305	0.940	Kept
{1, 2, 3, 4, 5, 6, 7, 8, 9, 30}	6	0.057	0.938	Kept
{1, 2, 3, 4, 5, 6, 7, 8, 9, 30}	3	0.208	1.004	Kept
{1, 2, 3, 4, 5, 6, 7, 8, 9, 30}	2	0.2	1.513	Kept
{1, 2, 3, 4, 5, 6, 7, 8, 9, 30}	4	0.15	1.051	Kept
{1, 2, 3, 4, 5, 6, 7, 8, 9, 30}	5	0	0.972	Kept
Detected ego-community: $C_{\{a,2\}} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 30\}$				



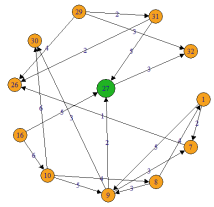
(a) 2-steps a's ego-community.



(b) 2-steps b's ego-community.

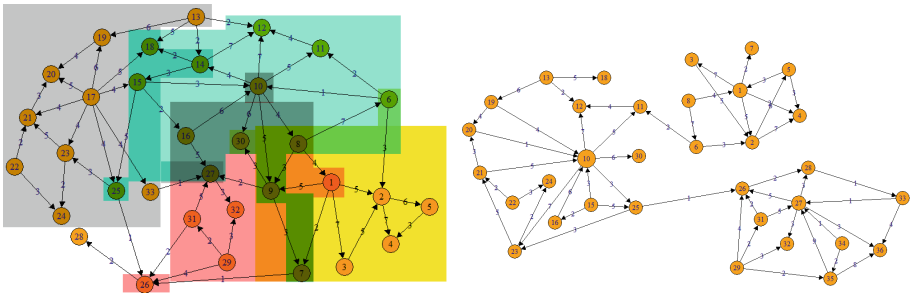


(c) 2-steps c's ego-community.



(d) 2-steps d's ego-community.

Fig. 2. 2-steps ego-communities detected by our algorithm on the network evaluation.



(a) 2-steps ego-communities detected in network at time t.

(b) Dynamic network structure at time $t+1$.

Fig. 3. Two successive snapshots of our dynamic evaluation network.

- 03 vanishing nodes: 9, 14 et 17;
- vanishing of links (16, 27) and (10, 8) as well as all links that were adjacent to vanishing nodes;
- 03 incoming nodes: 34, 35 et 36;
- 16 incoming links: (5, 1); (1, 4); (27, 26); (29, 35); (27, 28); (28, 33); (33, 36); (35, 36); (34, 35); (27, 36); (35, 27); (34, 27); (20, 10); (10, 23); (21, 10); (19, 10).

As for the macroscopic changes, after having applied our dynamic ego-community tracking algorithm, we found 3 ego-communities and 5 kinds of evolution:

1. A contraction with $C_{\{a,2\}}$ loosing two nodes: 9 et 30;
2. A growth with $C_{\{a,2\}}$ gaining two new links (5, 1) et (1, 4). Thus, $C_{\{a,2\}}$ ={1, 2, 3, 4, 5, 6, 7, 8};
3. A death of $C_{\{c,2\}}$;
4. A growth with $C_{\{b,2\}}$ after gaining the nodes that were in the community of $C_{\{b,2\}}$. Therefore, $C_{\{a,2\}}$ ={10, 11, 12, 13, 15, 16, 18, 19, 20, 21, 22, 23, 24, 25, 30};

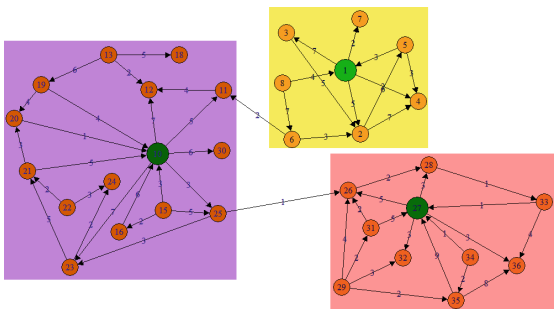


Fig. 4. 2-steps ego-communities detected by our algorithm in snapshot $t + 1$.

5. A growth with $C_{\{d,2\}}$ thanks to the arrival of new nodes and links. Thereby $C_{\{d,2\}}$ ={26, 27, 28, 29, 31, 32, 33, 34, 35, 36}.

In Fig. 4, we portray the new community structures of the snapshot $t + 1$.

6 Conclusion

The objective of this paper is to propose a new method able to track the evolution of dynamic ego-communities in instant messaging networks. To this end, we consider data from an instant messaging platform such as Facebook and build a social network based on "who talk to whom" and/or "who is closed to whom". We aim therefore at finding out communities centered on special nodes due to their characteristics or social positions. This is very useful since identifying individuals that are closely exposed to a disease is a starting point for controlling an epidemic. With this insight, finding out nodes that share a community with an infected individual may help to point out who is exposed or not and where it is more relevant to make specific actions to break down the disease spread.

References

1. Danisch, M.: Mesures de proximité appliquées à la détection de communautés dans les grands graphes de terrain. Ph.D. thesis, Paris 6 (2015)
2. Hou, J., Yu, Z., Hong, X., Wang, L.: Social circle detection based on micro-blogging topic and user follow relationship. In: 2016 International Conference on Asian Language Processing (IALP), pp. 222–227. IEEE (2016)
3. Lan, C., Yang, Y., Li, X., Luo, B., Huan, J.: Learning social circles in ego-networks based on multi-view network structure. IEEE Trans. Knowl. Data Eng. **29**(8), 1681–1694 (2017)
4. Mcauley, J., Leskovec, J.: Discovering social circles in ego networks. ACM Trans. Knowl. Discov. Data (TKDD) **8**(1), 4 (2014)
5. Moctar, A.O.M., Sarr, I.: Ego-centered community detection in directed and weighted networks. In: Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017, ASONAM 2017, New York, USA, pp. 1201–1208. ACM (2017)
6. Rees, B.S., Gallagher, K.B.: EgoClustering: overlapping community detection via merged friendship-groups. In: Özyer, T., Rokne, J., Wagner, G., Reuser, A. (eds.) The Influence of Technology on Social Network Analysis and Mining. LNSN, vol. 6, pp. 1–20. Springer, Vienna (2013). https://doi.org/10.1007/978-3-7091-1346-2_1