# Bees colonies for detecting communities evolution using data warehouse

Yasmine Chaabani*

Department of Computer Science,
Higher Institute of Management,
University of Tunis, Tunisia
Email: chaabanijasmin@gmail.com
*Corresponding author

Jalel Akaichi

Department of Computer Science,
Higher Institute of Management,
University of Tunis, Tunisia
and
Department of Computer Science
University of Bisha, Saudi Arabia
Email: j.akaichi@gmail.com

**Abstract:** The analysis of social networks and their evolution has gained much interest in recent years. In fact, few methods revealed and tracked meaningful communities over time. These methods also dealt efficiently with structure and topic evolution of networks. In this paper, we propose a novel technique to track dynamic communities and their evolution behaviour. The main objective of our approach and using the artificial bee colony (ABC) is to trace the evolution of community and to optimise our objective function to keep proper partitioning. Moreover, we use a data warehouse as a mind of bees to store the information of different communities structure in every timestamp. The experimental results showed that the proposed method is efficient in discovering dynamics communities and tracking their evolution.

**Keywords:** social network; community detection; bees colony.

**Reference** to this paper should be made as follows: Chaabani, Y. and Akaichi, J. (2020) 'Bees colonies for detecting communities evolution using data warehouse', *Int. J. Data Mining, Modelling and Management*, Vol. 12, No. 2, pp.192–206.

**Biographical notes:** Yasmine Chaabani is a PhD student of Computer Sciences at the Height Institute of Management in Tunisia. She received her Master's degree in Computer Sciences in the Faculty of Sciences of Monastir. She is a member of the Bestmod Laboratory University of Tunisia. She is also a Lecturer in Computer Sciences at the Umm al Qura University Saudi Arabia since 2016.

Jalel Akaichi received his PhD in Computer Science and Engineering field from the University of Sciences and Technologies of Lille 1. He is currently a Full Professor at the University of Tunis and the University of Bisha. His

research interests include, mainly, business intelligence, knowledge discovery in databases, data warehousing and mining, big data and data science, social networks, etc. He is the Leader and the Founder of the Business Intelligence Research Team and had multiple grants and positions for visiting and working in various international universities. He has published numerous book chapters, and many papers in international journals and conferences, served in different program committees related to international conferences and journals, and provided international courses and tutorials.

## 1 Introduction

A generalisation of the idea of social networks is that of information networks, in which nodes could comprise either actors or entities, and edges denote the relationships between them. Clearly, the concept of social networks is not restricted to the specific case of an internet-based social network such as Facebook. Indeed, the problem of social networking was often studied often in the field of sociology in terms of generic interactions between any group of actors. Such interactions may be in any conventional or non-conventional form, whether they are face-to face interactions, telecommunication interactions, email interactions or postal mail interactions. Generally, the studies of social network analysis did not focus on online interactions. They historically preceded the advent and popularity of computers or the Internet. A classic example of these surveys is the study of Travers and Milgram (1967), who hypothesised the likelihood that any pair of actors on the planet is separated by at most six degrees of separation. While such hypotheses have largely remained conjectures over the last few decades, the development of online social networks has made it possible to test such hypotheses at least in an online setting. This is also referred to as the small world phenomenon.

In recent years, a number of sites have arisen in order to model the interactions between different actors. Some examples of such social networks are Facebook, MySpace, Twitter and LinkedIn. In addition, sites used for sharing online media content (Flickr, Youtube or Delicious, etc.) can also be considered as indirect forms of social networks, because they allow extensive interaction between users. In these cases, the interaction is centered on a specific service such as content sharing. However, many fundamental principles of social networking can be applied. We note that such social networks are extremely rich in the sense that they contain a tremendous amount of content, like texts, images, audios or videos. Quite recently, considerable attention has been paid to analyse mixed numerical data and categorical features (Ahmad and Dey, 2007) where the researchers develop a new method using traditional k-mean algorithm. This content can be leveraged for many purposes. In particular, the interaction between the links and content has provided impetus to a wide variety of mining applications. Furthermore, the problem of clustering in social medias present an important research area, such as authors in Parvin et al. (2013) combines multiple partitions of data into a single clustering solution. The analyses of social network have balanced between dynamic analysis and static analysis. In the case of static analysis, we assume that social network changes slowly over time. We also analyse the whole network in batch mode over particular snapshots.

On the other hand, in the case of many networks such as instant messaging networks, interactions are continuously received over time at very large rate, which may lead to network streams. The analysis of such networks is much more challenging, and intensively discussed in recent studies (Aggarwal and Yu, 2005; Yu et al., 2011). The temporal aspect of networks often arises in the context of dynamic and evolving scenarios. Many interesting temporal characteristics of networks, such as evolving communities, interactions between entities and temporal events in the underlying network, can be determined.

## 2 Literature review

Over the last two decades, social networks have been extensively examined to investigate the interactions between people and define the important structural patterns existing in these interactions. The main focus has been on online social networks in which social networks (e.g., Facebook, LinkedIn and MySpace, etc.) have been considered as internet application. These networks have become popular as there exist no geographical limitations as it is the case in conventional social networks, such as face-to-face meetings or personal friendships, where interactions are defined in more conventional way.

A huge number of data analytic applications, like search, text analysis, image analysis and sensor applications can be provided by the infrastructure constructed around social networks. Moreover, analysing and evolving the social network structure is also an important issue. While some of these problems are also encountered in the more conventional notion of social networks, many of the issues related to the data-analytic aspects of social networks are relevant only in the context of online social networks. Furthermore, online social networks allow more efficient data collection on a large scale. Therefore, the computational challenges are far more significant. However, in most real social networks, communities and their members evolve over time. Dynamic networks also arise in the context of mobile applications, in which moving entities constantly interact with each other. Such dynamic social networks can be modeled as dynamic graphs for which the edges change constantly over time. These graphs result in the appearance of massive challenges because of the extremely large number of connections between the entities which may need to be tracked simultaneously.

Despite the fact that graph stream mining applications are essential for an efficient online analysis, they are typically necessary to summarise the data network structure in real-time and employ it in several mining applications. In this context, some recent developments have been proposed in Aggarwal and Wang (2010) and Yu et al. (2010).

Algorithms of incremental community detection adjusted the assignment of old community in order to create new community structure where the network is updated. Among the introduced approaches, we can mention modularity maximisation applied to detect incremental community detection. This approach uses two basic techniques: iterative execution and rule based revision. In fact, iterative execution is constructed upon base algorithms like the CNM algorithm or Louvain algorithm. Its design rationale consists in converting the community structure in the previous period into inputs on which the base algorithms will be re-run. Dinh et al. suggested in Thai et al. (2009), an iterative execution technique relying on the CNM algorithm. In their approach, novel vertices, neighbour vertices of novel or removed vertices/edges (named changed vertices

since they are included in network updates) are placed into singleton communities if the network is updated.

Afterwards, the network was aggregated to community level and the CNM algorithm was re-applied in order to construct the novel community structure. As the community-level network was considerably smaller than the original one, the algorithm was more effectively applied than the CNM algorithm. Bansal et al. performed, at vertex level, CNM-based iterative execution (Bansal et al., 2011). The vertex merging process of the last period's CNM was replicated prior to reaching novel vertices or neighbour vertices of the novel or the removed edges.

Subsequently, the regular CNM was performed to accomplish the community detectionprocedure. Unlike the algorithm introduced in Bansal et al. (2011), that proposed by Bansal et al. put basically novel changed vertices into singleton communities. It also divided their original community into smaller pieces to allow additional processing by applying CNM. Besides, their algorithm was structured from the final period as the initial state in which each novel vertex was put into a single community. Afterwards, the Louvain algorithm was used to adjust the community assignments of the vertices for the purpose of maximising locally the modularity. In Teow and Chong (2013) applied a similar approach in which new vertices as well as new neighbour vertices of novel or removed vertices/edges were put into singleton communities prior to the re-application of the Louvain algorithm.

Because this algorithm improved the vertices for being inspected for revision, its time complexity rose and community detection quality enhanced. Evidently, rule-based revision employed pre-defined rules in order to identify how to revise vertices' community assignments. Nguyen et al. introduced, in Xuan et al. (2011), a QCA algorithm belonging to this category. The applied algorithm allowed revising the strategies for each class of network updates. In the community assignment of neighbour vertices, new Vertex, remove Vertex, new Edge and remove Edge were revised in or to increase the gain in modularity. In Xie et al. (2012), a similar approach was developed by Shang et al. in Chi et al. (2007), authors proposed a spectral clustering algorithm to detect incremental community. They employed incidence vector/matrix for the representation of the network updates (changes of vertices and edges).

After that, the incidence vector/matrix was utilised for the incremental updating of the eigenvalues of the vertices used to determine the structure of the community in spectral clustering. Sun et al. applied a minimum description length (MDL) approach for the detection of communities and the encoding of a graph with a minimum number of bits (Papadimitriou et al., 2007). The dynamic network was divided into multiple sequential segments and the algorithm parameters were incrementally initialised by considering the last segment community structure as input. Obviously, the application of the previously-mentioned technique can be done just on non-overlapping communities. Moreover, other incremental algorithms, like iLCD (Hanachi et al., 2010), AFOCS (Tokala et al., 2011), FacetNet (Zhu et al., 2008) and the algorithm developed in Zaïane et al. (2013), were proposed to detect the overlapping community structure. We can conclude, from the afore-mentioned ideas, that all algorithms of incremental community detection rely on those of re-execution of static community detection. Therefore, the incremental algorithms' time effectiveness can be defined according to the base algorithms.

## 3 Proposal

Most researchers are interested in tracking the evolution of network by using a method for community detection for static network in different times after trying to understand the evolution of community in different snapshat. In our work, we aim to detecting of community with the evolution of networks over time. In this section, we define and formalise our idea as follow:

### *3.1 Formalisation*

*Definition 1:* Dynamic network.

A dynamic network is defined as a series of graphs $G = G_0, G_1, G_2, ..., G_t$ where $G_t = (V_t, E_t, S_t)$, $V_t$ represents the users in media networks,$E_t$ denotes the the set of edges which present the different interactions between them and $S_t$ is a set of interested subjects by user $V_t$ in his navigation in the media network at time $t$. In fact, we denote the number of communities $n_t$ at the ith snapshot. The community structure of $G_t$ is a collection of vertex sets $C_t = C_t^1, C_t^2, ..., C_t^{n_t}$.

*Definition 2:* Critical evolution events.

To capture the significant changes that the communities and individuals may undergo, we focus on different critical events (Asur et al., 2009) described below:

- *Events involving communities:* We define five basic events that communities can undergo between any two consecutive time intervals or snapshots. Let $G_t$ and $G_{1+t}$ be snapshots of G at two consecutive time intervals with $C_t$ and $C_{t+1}$ denoting the set of communities respectively. The five proposed events are:

    1 *Form:* A form event indicates the former of new community with different members of a community in a particular period of time, which leads to the form of the community.

    2 *Dissolve:* A dissolve event shows the disappearance of interactions among members of a community in a particular period oftime, which results in the breakup of the community.

    3 *Continue:* A continue event indicates that the members of a community remain at two consecutive timesteps. The addition or deletion of edges may happen in the community, which strengthens or weakens the interactions among some members. The community still exists as long as the interactions among its members are strong enough.

    4 *Merge:* A merge event implies that rich interactions were created between nodes which previously were in two different communities.

    5 *Split:* A split event indicates that some nodes in a community stop to interact with each other in a particular period of time, causing the splitting of the original community into two smaller communities.

- *Events involving individuals:* In these events, we analyse not only the evolution of communities, but also the influence of the individuals' behaviour on the communities. In this regard, we introduce four basic transformations involving individuals over snapshots.

    1 *Appear:* A node is considered as new appearing one when it occurs only in the actual time step.

    2 *Disappear:* When some members leave a community, the latter experiences a shrinkage in its events.

    3 *Expand(Join):* When a community absorbs new members, it experiences an expansion in its events.

    4 *Shrink(Leave):* When some members leave a community, the latter experiences a shrinkage in its events.

### *3.2 Artificial bee colony optimisation*

#### *3.2.1 Biological base*

The algorithms inspired from intelligent behaviours of honey bees constitute the third class of population based methods where the fittest individuals are selected through competition. A numerical optimisation algorithm based on foraging behaviour of honey bees, called artificial bee colony (ABC) was proposed by Karaboga in Pham and Castellani (2009). In ABC, the *employer* bees try to find food source and advertise other bees. The *onlooker* bees follow their interesting employer, while the *scout* bee flies spontaneously to find better food sources. Similar to other swarm-based optimisation algorithms, it is important to establish a proper balance between exploration and exploitation in bee swarm optimisation approaches. In a bee swarm, different behaviours of the bees provide this possibility to establish powerful balancing mechanism between exploration and exploitation. This property allows designing more efficient algorithms in comparison with other population based algorithms such as PSO and GA (Xuan et al., 2001).

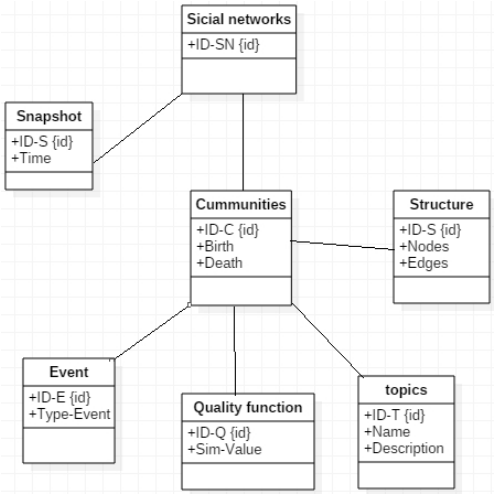
### *3.3 Data warehouse*

Data warehouse (DW or DWH), also known as an enterprise data warehouse (EDW), is a system used to report and analyse data. It is considered as core component of business intelligence. DWs are central repositories of integrated data from one or more disparate sources. They store current and historical data. They are employed to create critical analytical reports for knowledge workers throughout the enterprise. Examples of reports could range from annual and quarterly comparisons and trends to detailed daily sales analysis.

*Data mart* is a simple form of a DW focusing on a single subject (or functional area). Hence, it draws data from a limited number of sources such as sales, finance or marketing. Data marts are often built and controlled by a single department within an organisation. The sources could be internal operational systems, a central DW, or external data. Given that data marts generally cover only a subset of the data contained in a DW, they are often easier and faster to implement.

In this paper, we suggest using data mart to store the evolution of every community over the time which will be stored in the DW. *Besides, DWH is an efficient tool used to save all possible changes and keep their traces by saving all versions of a given community in a data mart.*

**Figure 1** The structure of the used DWH




### *3.3.1 DWH architecture*

To save the differents changes of network evolution in every timestamp, we used a DW. We applied the schema in Figure 1 for our DW, resulting from the application of the algorithm $BCOSMN$ which contained six tables: The network table and the communities table store the history of measurement attributes occurring over a period of time. The different tables are shown in Figure 1.

## *3.4 Algorithm analysis*

Our model is called $BCOCE$.[1] First, we used the $BCOSMN$[2] defined in Chaabani and Akaichi (2016) for communities detection at time $t$ to obtain the structure of networks where $G_t = (V_t, E_t, T_t)$ as it is defined before when we put all information in the DHW. The aim of our work consists in optimising our objective function $DSim$ that characterises the overall quality of a partitioning. The new function of density was defined by adding the average of the Jaccard coefficient to measure the structural and the semantic similarity between two nodes. It is defined in (Chaabani and Akaichi, 2016):

$$DSIM(C) = \frac{1}{\mid V(C) \mid} \sum_{V_i \in C} JS(V_i, V_j) \tag{1}$$

where $V_i$ and $V_j$ in C the comunity and JS is calculated as follows:

$$JS(V_i, V_j) = \frac{J(N_i, N_j) + J(S_i, S_j)}{2} \tag{2}$$

where the $N_i$ is the neighbourhoods of user $V_i$, and $N_j$ is the neighbourhoods of user $V_j$. $S_i$ is the topics interesting the user $V_i$, while $S_j$ is the topics interesting the userr $V_j$.

Bees colonies presents a multi-agents system which monitors the social network and detects its communities. Our model consists of different steps.

**Figure 2** A bees colonies communities at time $T$ (see online version for colours)

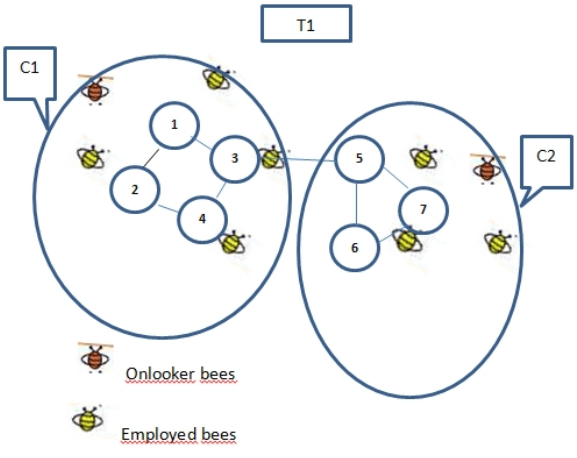
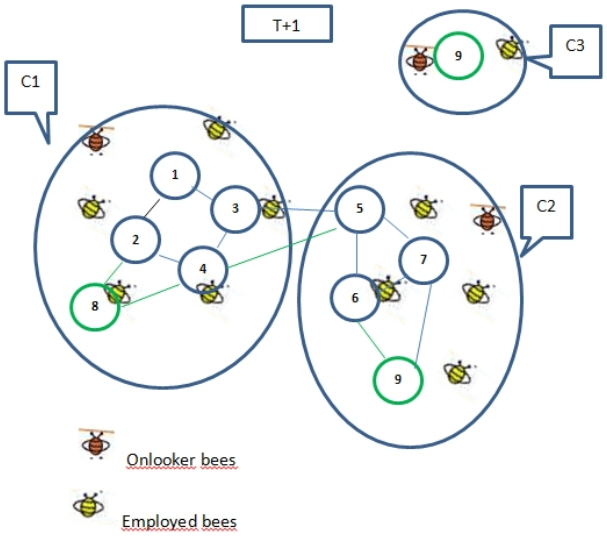



**Figure 3** A bees colonies communities at time $T$ + 1 (see online version for colours)




It is described by Algorithm 1. In the first step, we applied the algorithm $BCOSMN$ to our network at time $t$. Then, we built the DWH of our networks in time $t$. For each community determined in the first step, we put a bee colony in every community as it is shown in Figure 2 to track the evolution of community in the next timestamp. As

we describe in the previous section, we have different types of bees: The employee bees try to discover the change in the time $t + 1$, the appearance or disappearance of relations (edges) or users (nodes) and saved it as visited or not visited. As we described above, each visited node or edge contains a pheromone substance of bees. Every evolution of networks is saved by the onlooker bees in the DWH. At this step, the latest bees use Algorithms 2 and 5 to make the good decision which ameliorate our objective function using different algorithm to make the good choice when we discover the change event of networks described in previous section. So, at each change, the onlooker bee recalculates our objective function and take the decision after discovering the new structure of network and the appearance or disappearance of different topics using the LDA method (Ahuja et al., 2015).

**Algorithm 1** BCOCE

**Input**: A graph $G_t = \{(V_t, E_t, S_t\}$, $C_t = \{C_t^1, C_t^2, ..., C_t^{n_t}\}$ and $S_t = \{S_t^1, S_t^2, ..., S_t^{n_t}\}$
**Output**: A partitioning $C_t = \{C_t^1, C_t^2, ..., C_t^{n_t}\}$
**begin**
built a data warehouse with information obtained from $BCOSMN$ algorithm
Put a bees colony on every community
For each community (colony) The employee bees try to find new members
All employee bees store the non-visited neighbours nodes in the FreeV vector and the non-visited edges to a neighbours nodes in the FreeE vector.
Each employee bees gives the change of networks to the onlooker bees which store them in the DWH
onlookerbees-decision ($G_t, G_{t+1}, C_t^{n_t}$), FreeV, FreeE
onlookerbees-community-decision ($C_t$) The onlooker bee takes the decision and inserts the update data into the DWH
Access to the DWH to know the new structure of communities in t + 1
**end**

**Algorithm 2** Onlookerbees-decision

**Input**: A graph with its evolution $G_t, G_{t+1}$, the community members $C_t^{n_t}$, FreeV, FreeE
**Output**: $C_t^{n_t}$
**begin**
for all nodes in FreeV do IF (BeeFindEventN($G_t, G_{t+1}$, FreeV[i])==’appearN’)
$DSIM - NEW = SIM(C_t^{n_t} \cup FreeV[i])$ IF (DSIM($C_t^{n_t}$) - DSIM-NEW($C_t^{n_t} \cup FreeV[i]) < 0) decision = join$ C_t^{n_t} = C_t^{n_t} \cup FreeV[i]$
IF (BeeFindEventN($G_t, G_{t+1}$, FreeV[i])==’disappearN’)
decision = delete
$C_t^{n_t} = C_t^{n_t} - FreeV[i]$
For all edges in FreeE do IF (BeeFindEventE($G_t, G_{t+1}$, FreeE[i])==’appearE’)
$DSIM - NEW = SIM(C_t^{n_t} \cup FreeE[i])$ IF (DSIM($C_t^{n_t}$) - DSIM-NEW($C_t^{n_t} \cup FreeE[i]) < 0) decision = join$ C_t^{n_t} = C_t^{n_t} \cup FreeE[i]$
IF (BeeFindEventE($G_t, G_{t+1}$, FreeE[i])==’disappearE’)
decision = delete
w1qwacsZ X$C_t^{n_t} = C_t^{n_t} - FreeE[i]$
**end**

**Algorithm 3** BeeFindEventN($G_t, G_{t+1}$, v)

**Input**: A graph with its evolution $G_t, G_{t+1}$, v
**Output**: $event$
**begin**
IF (($v \in V_t$) and ($v \in V_{t+1}$))
event = appearV
IF (($v \in V_t$) and ($v \notin V_{t+1}$))
event = disappearN
return event
**end**

**Algorithm 4** BeeFindEventE

**Input**: A graph with its evolution $G_t, G_{t+1}$, $e$
**Output**: $event$
**begin**
IF (($e \in E_t$) and ($e \in E_{t+1}$))
event = appearE
IF (($e \in E_t$) and ($e \notin E_{t+1}$))
event = disappearE
return event
**end**

**Algorithm 5** Onlookerbees-community-decision

**Input**: A graph $G_{t+1}$
**Output**: $C_t^{n_i}$
**begin**
FOR(All communities for $C_t$ from 1 to n) IF (DSIM($C_t^{i_t}$) - DSIM-NEW($C_t^{i_t} \cup C_t^{i+1}{}_t$)>=0)
decision = merge
$C_t^{n_t} = C_t^{i_t} \cup C_t^{i+1}{}_t$
**end**

## 4 Simulation results

The main purpose of this section is to analyse experimentally our proposal $BCOSE$ and to compare it to other recent proposals for tracking communities evolution in social media networks. We compared $BCOSE$ to the well-known proposal, namely the model in Thai et al. (2009) (noted $CNM$) based on communities detection in every timestamp or every evolution of the network.For evaluating our proposal, we considered large scale real-world networks as well as synthetic networks generated using the LFR benchmarks (Fortunato, 2017). As evaluation criteria for a computed partitioning, we took into account these standard measures: $Q_{sim}$ defined in Chaabani and Akaichi (2016), $D_{sim}$ our poposed objective function and coverage. When the real partitioning

is available (which is the case for LFR benchmarks), we should also consider the normalised mutual information (NMI). Now, we define these criteria. For this reason, let $P = \{C_1, C_2, \cdots, C_k\}$ a computed partitioning using defined algrithms.

$Q_{sim}$ of a partitioning is defined as follows (Chaabani and Akaichi, 2016):

$$Q_{sim} = \frac{1}{(2m)} \sum_{j,k} (a_{v_j,v_k} - \frac{d(v_j)d(v_k)}{2m}) * \delta(v_j, v_k) * \xi(v_j, v_k) \tag{3}$$

with $j = 1...n$ and $k = 1...n$.

Settings have the same meaning as in the definition of the initial modularity, with $\xi$ present the value of the coefficient of Jaccard between $v_j$ and $v_k$ to ensure a quality function which provides the semantic and structural similarity.

As far as the coverage criteria are concerned, it measures the fraction of edges inside a community divided by the total number of edges in the graph and is given by Brandes and Gaertler (2003):

$$Coverage(P) = \sum_{i=1}^{k} \frac{\mid E_i \mid}{\mid E \mid} \tag{4}$$

However, when the real community structure is available, we can compare it to the computed one using the NMI given by Kim et al. (2010):

$$NMI(A, P) = \frac{-2 \sum_{a \in A} \sum_{b \in P} |a \cap b| log(\frac{|a \cap b|n}{|a||b|})}{\sum_{a \in A} |a| log(\frac{|a|}{n}) + \sum_{b \in P} |b| log(\frac{|b|}{n})} \tag{5}$$

where $A$ is the real structure of the network. We notice that $NMI(A, P) = 1$ when both partitions $A$ and $P$ coincide. Given all these performance criteria, we can now outline the experimental results.

### *4.1 Real world network*

In this section, we consider real networks. Unfortunately, the real hidden structure for these networks is generally unknown. However, we suppose that some communities are established around terrorist in Tunisia. We also demonstrate the power of using a bees colony to track the evolution of communities in each timestamp with the integrating DW layer for storing all information and attributes in every change to have a trace to back up at any time.

We applied our proposed model on dataset collected by crawling one week of public tweets through using the 140 dev Twitter framework.[3] Our collection consists of tweets from the 10th till the 17th of July 2016 (one week). We selected the first 100 relevant users according to the follower's number that have tweets number more than 250 tweets about terrorism. For the first network, we applied the algorithm $BCOSMN$ after partitioning the network into times $T$ where all information are stored in the DWH. Then, we tracked the evolution of community by using our proposal method $BCOCE$ employing artificial bees colony optimisation method where every evolution of network

is stored in the data mart. The evolution of topics users over the time is shown in Figure 1.

We compared the structure of networks in every timestamps of our proposal $BCOCE$ with the $CNM$ in Thai et al. (2009) using the qualities functions defined before in Table 1, 2 and 3.

**Table 1** Values of the quality function in time $T$

| | *CNM* | *BCOCE* |
|---|---|---|
| $Q_{sim}$ | *0.56* | 0.45 |
| $D_{sim}$ | 0.62 | *0.72* |
| Coverage | 0.56 | *0.58* |
| CPU | 68 | *142* |

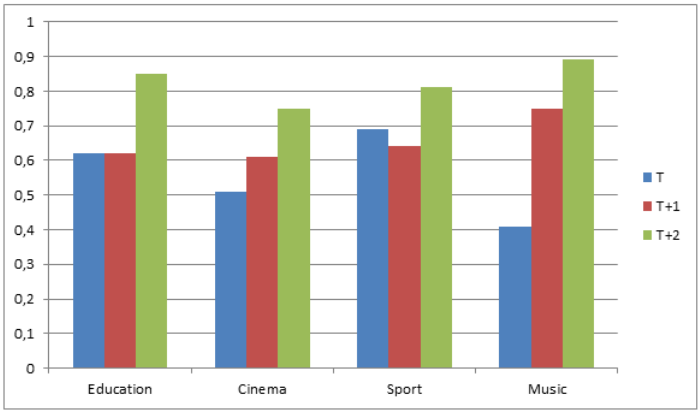
**Table 2** Values of the quality function in time $T + 1$

| | *CNM* | *BCOCE* |
|---|---|---|
| $Q_{sim}$ | 0.58. | *0.69* |
| $D_{sim}$ | 0.56 | *0.75* |
| Coverage | *0.48* | 0.32 |
| CPU | 81 | *162* |

**Table 3** Values of the quality function in time $T + 2$

| | *CNM* | *BCOCE* |
|---|---|---|
| $Q_{sim}$ | 0.62 | *0.71* |
| $D_{sim}$ | 0.42 | *0.51* |
| Coverage | *0.52* | 0.44 |
| CPU(s) | 76 | *156* |

**Figure 4** The evolution of topics over the time (see online version for colours)



For each simulation, we performed runs of CNM and our model $BCOCE$ for different snapshats. Indeed, our model gave the important values of the $Q_{sim}$ and $D_{sim}$ compared to CNM model. But, the latter one provide the best values of coverage in an important execution time.

### *4.2 Evaluation on artificial networks*

In this first set of runs, we used randomly-generated social networks using the LFR benchmarks (Fortunato, 2017) whose structure was imposed during the generation phase. Having the latter as a reference point, we used the $NMI$ in equation (5) to evaluate the quality of the computed partitioning. The LFR generator has several parameters, such as the number of nodes ($N$); the average degree of incoming edges ($k$); the fraction between incoming and outgoing edges inside a community ($\mu$); the minimal community size ($minc$ and the maximal community size ($maxc$). We started our simulation with reference graphs and by varying one or several parameters of the generator. We obtained several graphs of different complexities. For each simulation, we considered two graphs and computed the CPU time (in seconds) taken by the model to output a partitioning.[4] In addition, we took into account the computed number of communities ($\#comm$) compared to the real exact one ($EP$). In this step, we discarded the semantic similarity, ontlu the structure similarity. For each simulation, we performed runs of our method ($BCOCE$).

We considered reference graphs composed of $N$ = 5,000, $k$ = 15, $maxk$ = 50, $\mu$ = 0.1, $minc$ = 20 and $maxc$ = 50. Simulation results are reported in Tables 4, 5 and 6.

**Table 4** NMI for reference graphs

| | *CNM* | *BCOCE* |
|---|---|---|
| $NMI(g_1)$ | 0.512 | *0.892* |
| $NMI(g_2)$ | 0.558 | *0.911* |

**Table 5** Average CPU time (in seconds) for reference graphs

| | *CNM* | *BCOCE* |
|---|---|---|
| $CPU(g_1)$ | 154.123 | *161.434* |
| $CPU(g_2)$ | 162.156 | *165.381* |

**Table 6** Number of computed communities compared to the exact one ($EP$) for reference graphs

| | *CNM* | *BCOCE* |
|---|---|---|
| $\#comm(g_1)$ | 102 | *154* |
| $\#comm(g_2)$ | 105 | *155* |

First, all used graphs are of small size mainly due to the high spatial complexity of the algorithm. We also notice, from Tables 1 and 3 that our algorithm $BCOCE$ computed

better partitioning than $CNM$ (Travers and Milgram, 1967) with respect to $NMI$ and the number of communities. Unfortunately, this resulted in much more CPU time (see Table 2).

## 5 Conclusions and future works

In this paper, we were interested in the issue of tracking the evolution of community in media networks. The main contribution of our method is the use of an ABCO to optimise our objective function defined in our previous works. Then, we integrated the DW technology to improve analysis if huge data stream was generated by social networks in particular twitter. Our method consists of the communities detection in dynamic network. It gave as a more efficient and similar community (semantic and structural similarity).

The key next step is to extend our proposal for reasoning about studying different events over time. As future works, we would like to improve the performance of our method by studying a big real network.

## References

Aggarwal, C.C. and Yu, P. (2005) 'Online analysis of community evolution over data streams', *SIAM Conference on Data Mining*, Vol. 223, No. 5.

Ahmad, A. and Dey, L. (2007) 'A k-mean clustering algorithm for mixed numeric and categorical data', *Data & Knowledge Engineering*, Vol. 63, No. 2, pp.503–527.

Ahuja, A., Wei, W. and Carley, K. (2015) *Topic Modeling in Large Scale Social Network Data*.

Aggarwal, C.C. and Wang, H. (2010) *Managing and Mining Graph Data*, pp.13–68, Springer, Boston, MA.

Asur, S., Parthasarathy, S. and Ucar, D. (2009) 'An event-based framework for characterizing the evolutionary behavior of interaction graphs', *ACM Trans. KDD*, Vol. 3, No. 4, p.16.

Bansal, S., Bhowmick, S. and Paymal, P. (2011) *Complex Networks*, pp.196–207, Springer, Berlin, Heidelberg.

Brandes, U. and Gaertler, M. (2003) 'Experiments on graph clustering algorithms', *LNCS*, Vol. 14, No. 1, pp.51–65.

Chaabani, Y. and Akaichi, J. (2016) 'Bees colonies for meaningfull communities detection in social medias network', in *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld)*, July, pp.1052–1057.

Chi, Y., Gong, Y., Huang, T.S., Ning, H. and Xu, W. (2007) 'Incremental spectral clustering with application to monitoring of evolving blog communities', in *SDM, SIAM*, pp.261–272.

Fortunato, S. (2017) *Benchmark Graphs to Test Community Detection Algorithms* [online] http://sites.google.com/site/santofortunato/inthepress2.

Hanachi, C., Cazabet, R. and Amblard, F. (2010) 'Detection of overlapping communities in dynamical social networks', in *Social Computing (SocialCom) M*, pp.309–314.

Kim, Y., Son, S-W. and Jeong, H. (2010) 'Finding communities in directed networks', *Phys. Rev. E*, January, Vol. 81, p.16103.

Papadimitriou, S., Graphscope, P.S.Y., Sun, J. and Faloutsos, C. (2007) 'Graphscope: parameter-free mining of large time-evolving graphs', in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, pp.687–696.

Parvin, H., Minaei-Bidgoli, B., Alinejad-Rokny, H. and Punch, W.F. (2013) 'Data weighing mechanisms for clustering ensembles', *Computers & Electrical Engineering*, Vol. 39, No. 5, pp.1433–1450.

Pham, D.T. and Castellani, M. (2009) 'The bees algorithm modelling foraging behaviour to solve continuous optimisation problems', *Proc. ImechE, Part C*, Vol. 223, No. 12, pp.2919–2938.

Teow, L.N. and Chong, W.H. (2013) 'An incremental batch technique for community detection', in *Information Fusion (FUSION), 2013 16th International Conference*, pp.750–757.

Thai, M.T., Dinh, T.N. and Xuan, Y. (2009) 'Towards social-aware routing in dynamic communication networks', in *Performance Computing and Communications Conference (IPCCC) IEEE 28th International*, Vol. 223, pp.161–168.

Tokala, S., Thai, M.T., Nguyen, N.P. and Dinh, T.N. (2011) 'Overlapping communities in dynamic networks: their detection and mobile applications', in *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking*, ACM, pp.85–96.

Travers, J. and Milgram, S. (1967) 'The small world problem', *Psychology Today*, Vol. 1, No. 1, pp.61–67.

Xie, F., Chen, Z., Miao, J., Fang, X., Wu, C., Shang, J. and Liu, L. (2012) 'A real-time detecting algorithm for tracking community structure of dynamic networks', in *6th SNA-KDD Workshop*, ACM.

Xuan, Y., Thai, M.T., Nguyen, N.P. and Dinh, T.N. (2001) 'A novel bee swarm optimization algorithm for numerical function optimization', *Commun. Nonlinear Sci. Numer. Simulat*, pp.3142–3155.

Xuan, Y., Thai, M.T., Nguyen, N.P. and Dinh, T.N. (2011) 'Adaptive algorithms for detecting community structure in dynamic social networks', in *INFOCOM, 2011 Proceedings IEEE*, IEEE, pp.2282–2290.

Yu, P., Aggarwal, C.C. and Zhao, Y. (2010) 'On clustering graph streams', *SIAM Conference on Data Mining*.

Yu, P., Aggarwal, C.C. and Zhao, Y. (2011) 'Online analysis of community evolution over data streams', *Outlier Detection in Graph Streams*.

Zaïane, O.R., Takaffoli, M. and Rabbany, R. (2013) 'Incremental local community identification in dynamic social networks', in *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp.90–94.

Zhu, S., Sundaram, H., Tseng, B.L., Lin, Y-R. and Chi, Y. (2008) 'Facetnet: a framework for analyzing communities and their evolutions in dynamic networks', in *Proceedings of the 17th International Conference on World Wide Web*, ACM, pp.685–694.

## Notes

1 BCOCE is an acronym for bee colony optimisation for communities evolution detection in dynamic social medias networks.

2 BCOSMN is an acronym for bee colony optimisation for communities detetion in social medias networks.

3 http://140dev.com/free-twitter-api-source-code-library/.

4 Due to systems' overload, the CPU time is averaged over five trials for each case.