IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 11, 2021, accepted April 19, 2021, date of publication April 29, 2021, date of current version May 11, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3076594

Detection of Community Structures in Dynamic Social Networks Based on Message Distribution and Structural/Attribute Similarities

HEDIA ZARDI[1], BUSHRA ALHARBI[1], WALID KARAMTI[1,2], HANEN KARAMTI[3], AND EATEDAL ALABDULKREEM[3]

[1]Department of Computer Science, College of Computer, Qassim University, Buraydah 51452, Saudi Arabia
[2]BIND Research Group, College of Computer, Qassim University, Buraydah 51452, Saudi Arabia
[3]Computer Sciences Department, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, Riyadh 84428, Saudi Arabia

Corresponding author: Hanen Karamti (hmkaramti@pnu.edu.sa)

This work was funded by the Deanship of Scientific Research at Princess Nourah Bint Abdulrahman University through the Fast-Track Research Funding Program.

ABSTRACT Community detection is a crucial challenge in social network analysis. This task is important because it gives leads to study emerging phenomena. Indeed, it makes it possible to identify the different communities representing individuals with common interests and/or strong connections between them. In addition, it allows tracking the transformation of these communities over time. In this work, we propose a dynamic community detection approach called Attributes, Structure, and Messages distribution-based approach (ASMsg). In addition to the node attributes and the topological structure of the network, we use the rate of transferred messages as the key concept of this approach. Therefore, we obtain communities with similar members that are strongly connected and also frequently interacting. Furthermore, the proposed approach is able to detect all possible communities' transformations even if the communities are overlapped. To demonstrate its efficiency, we widely test ASMsg on artificial and real-world dynamic networks and compare it with representative methods. The results show the superiority of our approach in terms of detected communities.

INDEX TERMS Community detection, dynamic social networks, attributed networks, community evolution, overlapping communities.

I. INTRODUCTION

Social networks such as Twitter, Facebook, and Instagram have attracted millions of users, therefore they get a great deal of attention from researchers. Generally, social network actors are divided into groups called ''communities''. Conventionally, a community is described as a set of members densely interconnected relatively to the rest of the network, and they have a similar interest in topics and properties, for example, may live in the same county, study in the same university or work at the same company [11], [21].

Therefore, the social networks are usually represented by graphs such as the social members are represented by nodes, and the relationships between them are represented by edges. These graphs consist of sets of communities $C = \{C_1, C_2, \cdots, C_k\}$, where the community C_i is a dense group of nodes that have common properties. These nodes are usually strongly connected to each other and loosely connected to the remaining nodes in the network [5].

Due to the individual's tendency to change, the joining and leaving of the network is continuously happening. For instance, new users can register on the social network every moment and, at the same time, many others may delete their accounts. That's led to the evolution of the communities as shown in Figure.1. For this reason, detected communities must be updated according to the dynamic of the network, and this represents one of the challenges related to community detection [36]. The first problem in the community detection process is related to the concept of community structure. In fact, there is no exact definition of the ''community'' concept. In addition, dividing a graph into subsets of nodes is an NP-hard problem [11], so this problem needs to be approached with approximate methods.

The associate editor coordinating the review of this manuscript and approving it for publication was Long Wang.

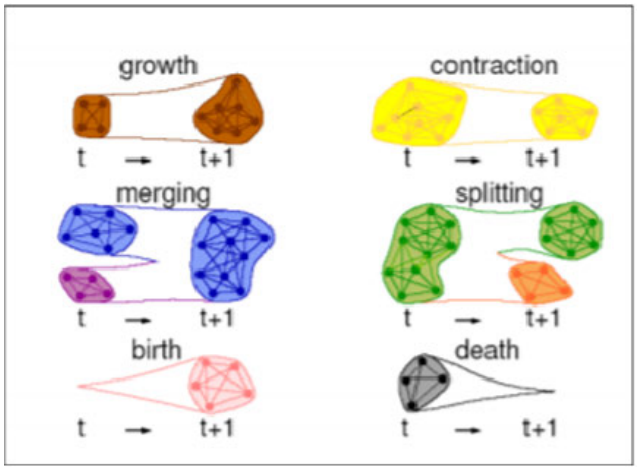

FIGURE 1. Community transformations in evolutionary networks [25].

An additional problem is the lack of an agreed-upon definition for a dynamic community [12], [13]. A simple approach defines it as a succession of static communities, but it is also more realistic to model a dynamic community as a single community with a set of events.

The main goal of this work is to propose an approach for the identification of overlapping communities in dynamic social networks. This approach allows to track the evolution of the communities over time. Our aim is to identify communities with strongly connected members who are homogeneous and daily interacted. For this reason, we use the network structure and also the properties of social members. In addition, we use the rate of transferred messages between the members as an important parameter for our purpose. Moreover, our approach is incremental: it consists of updating existing communities according to the network's evolution rather than recalculating a new partition after each event seen in the network. By this way, the communities in the current time step are detected based on the communities of the past time frame. Therefore, the detection of communities is done by only minor and successive local modifications. In addition, our model is able to allow all transformations on the communities: birth, death, growth, retraction, as well as more complex events such as split and merge. That is another contribution of our model compared to many existing methods that are unable to detect complex events.

The remainder of the paper is organized as follows. Section 2 is dedicated to presenting some known approaches. Section 3 outlines the proposed model. In Section 4, a series of experimental results are given. Section 5 presents a summary and our suggestions for future work.

II. RELATED WORK

The detection of communities has attracted many researchers and many algorithms have been proposed. Historically, the first proposed methods were considered only a simple capture of the network at a given moment, i.e., they view the social network as a static structure in which neither members nor relationships evolve over time [9], [10], [17], [23], [28], [34]. Otherwise, real-world social networks are dynamic. For this reason, nowadays a lot of attention has focused on the question of detecting dynamic communities. We can distinguish two categories of algorithms dealing with dynamic communities: non-incremental and incremental.

A. NON-INCREMENTAL APPROACHES

Non-incremental approaches are based on the idea of dividing the social network into several snapshots; each one is a static network. Thereafter, an algorithm for detecting static communities is applied to each snapshot. Finally, a matching between the detected partitions is applied to track the communities' evaluation [4], [14].

To study the evolution of a community C belonging to a partition $P(t)$, Hopcroft *et al.* proposed in [15] to look for the community C' of the partition $P(t+1)$ which has the largest nodes' number in common with C and which has the size closest to it. Thereby, the authors defined the match measure as the intersection between a community C of the partition $P(t)$ and a community C' of the partition $P(t+1)$. This measure is defined as follows:

$$match\left(C, C'\right) = min\left(\frac{\left|C \cap C'\right|}{|C|}, \frac{\left|C \cap C'\right|}{|C'|}\right) \quad (1)$$

Louvain-OR-Attribute (LOA) and Louvain-AND-Attribute (LAA) are two approaches that are proposed in [3]. The authors combined the gain in modularity with the similarity of users' attributes to detect the communities. The authors used Louvain approach [6] in which each node is considered as an embryo community. By comparing each node with its neighbors, the communities with the greatest possible modularity gain are merged. This gain is defined by:

$$Gain - modularity = [\frac{S_{in} + L_{i,C}}{2w} - (\frac{S_t + L_i}{2w})^2] - [\frac{S_{in}}{2w} - (\frac{S_t}{2w})^2 - (\frac{L_i}{2w})^2] \quad (2)$$

where S_{in} is the sum of edges weights inside the community C, S_t is the sum of weights of edges connecting the different nodes in community C, L_i is the sum of edges weights of the node i, $L_{i,C}$ is the sum of the weights of the edges connecting the node i by C, and w denotes the sum of the weights of all the edges in the network. Afterward, the authors find the common attributes by the intersection of both communities' attributes. The limit of these methods is that their outputs depend on the order in which the nodes are considered.

Hoseini *et al.* [16] proposed an algorithm that uses the amount of the exchanged data between the users of the network. Besides this parameter, this algorithm is implemented with two other parameters: 1) Betweenness centrality of edge and 2) Similarity of the two nodes. A weight is allocated to each parameter that represents the relative importance of each one compared to the other. Thus, a cut parameter is defined

as follows:

$$cut\ (s,t) = \frac{\alpha_1\ (CB\ (s,t))}{\alpha_2\ (RM\ (s,t))\ .\alpha_3\ (S\ (s,t))} \tag{3}$$

where $CB(s,t)$ is the betweenness centrality of the edge connecting the nodes s and t, $RM(s,t)$ is defined as the rate of the sent messages, and $S(s,t)$ represents the level of similarity between s and t.

The major limit of non-incremental community detection approaches is that are very time-consuming on a quickly growing network. Moreover, community detection algorithms are unstable because they can provide very different results for two similar networks with minor differences. Consequently, a modification between two partitions can be explained as a real structural evolution of the communities or is simply due to the used static algorithm.

B. INCREMENTAL APPROACHES

Unlike non-incremental approaches, the incremental approaches maintain a partition having good quality and update it on the basis of the networks evolution in order to avoid repetitive computations [1]. In this case, the detected partition will depend on the previous ones. Moreover, the algorithms of incremental community detection are usually based on the checking of the changed nodes and edges. Since the number of nodes to be updated is reduced, the time efficiency of community detection will be increased [26]. The studies of the incremental approaches can be classified into two subsections: (1) studies dealing with the topological structure but ignoring the nodes' attributes and (2) studies considering the nodes' attributes with the topological structure of the network.

1) COMMUNITY DETECTION IN NON-ATTRIBUTED NETWORKS

Most existing incremental approaches consider only the structure of the networks. This section provides a brief overview of these approaches.

In [32], the authors proposed a filtering technique called δ-screening which can be combined with any community detection algorithm that uses modularity as an objective function. δ-Screening allows to speedily locate the parts of the network that are probably affected by the latest events in the network. At first, this technique detects a subset of nodes then evaluates them at the start of every time step. This is to discover all those nodes that their community could potentially change during δt; the rest of the nodes retain their previous communities. The authors applied their proposed method on two widely-used community detection algorithm: Louvain algorithm [6], and smart local moving (SLM) algorithm [27]. This method is limited to be used only with an incremental algorithm that uses modularity. In addition, it is not able to detect overlapping communities. But in real-world social networks, communities can always overlap since each node may be a member of multiple communities. More recently, this feature has attracted interest from many researchers who proposed approaches to detect overlapping communities.

In [26], Shang *et al.* proposed a machine learning classifier, called learning-based targeted revision approach (LBTR) that is used to predict the nodes to be checked for revision of community assignment. The authors used two classification models: Logistic Regression and Support Vector Machines (SVM). This approach is based on three bases: 1) Local modularity maximization, 2) Learning-based targeted revision to reduce the computations for local modularity maximization, and 3) Small community merging to increase community detection quality. LBTR merges communities when their sizes are smaller than a threshold compared to their neighbor communities. To extract the initial partition, the authors applied Louvain algorithm. Subsequently, they applied their proposed approach and the baseline methods on the dynamic networks every month.

Likewise, in [7], Cazabet *et al.* have developed the incremental *iLCD* (intrinsic Longitudinal Community Detection) model to detect overlapping communities. This approach constructs the network edge by edge. As soon as there is a clique [1] (having 3 or 4 members), a new community will be created. Subsequently, *iLCD* updates existing communities by integrating new nodes such that a node n is added to a community C if the number of neighbors of n belonging to community C is greater than a given limit. The algorithm then calculates the overlap of the communities based on the ratio of nodes that they have in common. If this overlap is high, then the two communities are merged together. The main limit of this model is that the nodes and the links can only be added. As a result, it does not allow the division and the death of the communities.

2) COMMUNITY DETECTION IN ATTRIBUTED NETWORKS

When the nodes are described with a set of attributes, the network is said attributed. In this case, it is crucial to consider both the network structure and the members' properties to have densely connected members within the communities, which share common attributes [35], [37]. A few recent studies have considered attributed networks. In this section, we present some well-known methods.

In [18], Guo *et al.* proposed an incremental algorithm based on Improved Modularity (ICIM). Their proposed algorithm divides the network into nodes and edges attributes. First, the nodes that are affected by the network's evolution are detected, and the node's edges are marked. Then, the community structure is updated based on the evolution of these edges. One of the limits of this algorithm is that it runs slowly in the process to find the neighbors of nodes.

Xia and Tuo [31] proposed an incremental community detection method in dynamic weighted networks called ICDW. This method associates the attributes with the topological graph to form weighted nodes and edges. The authors

[1] A clique is a subset of nodes, such that every two distinct nodes in the clique are adjacent

give a weight to each attribute. In addition, they use a threshold ϵ, which denotes an average change rate of edge weight, to decide whether an attribute's weight changes significantly or not. The edge is added to the increments only if the change rate of its weight is greater than the threshold. This algorithm limited to be used only with social networks that change negligibly in adjacent network snapshots such as a mail network.

More recently, SmaCD model is proposed by Zardi in [33]. It is a Multi-Agent system for the detection of dynamic communities that employ the structural similarity, the topological structure of the network, and the communication between the social members to detect overlapping and dynamic communities. The authors used two types of agents: agent-node and agent-community. The agents-nodes are loaded by observing the network and updating the graph. In the case of a significant change in the network, the agents-nodes announce to the agents-communities entities the observation of events that can distort the existing communities. As a result, the agents-communities react and update the community structure. The drawback of this model is that the overlap is not considered.

Based on our review of existing approaches, we can conclude that the incremental approaches have great advantages over other alternatives. Indeed, the adaptation of a partition according to the evolution of the network (rather than recalculating a new partition at each step) ensures the stability of obtained partitions and reduces the calculation time of the communities. In addition, it can be concluded that the consideration of the members' attributes is very helpful to achieve high-quality community detection and also to understand the characteristics of communities very well.

III. MESSAGE DISTRIBUTION AND STRUCTURAL/ATTRIBUTE SIMILARITIES BASED METHOD

Our proposed model called ASMsg is an incremental method; it consists of finding an initial partition and then adapting existing communities according to the evolution of the dynamic network. ASMsg integrates node attributes with the topological structure of the graph in order to obtain similar and strongly connected members within the same community. In addition, it uses the rate of transferred messages between the social members in order to obtain communities members who are often interacting with each other by exchanging messages.

A. PROBLEM SPECIFICATION

In this work, we design the social network by an attributed graph $G = (V, E, A)$ that is a single graph with a set of modifications on nodes, edges, and the number of transferred messages, such that $V = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes representing the members of the social network, $E = \{e_1, e_2, \ldots, e_m\}$ is the set of edges representing the relationships between the social members, and $A = \{a_1, a_2, \ldots, a_k\}$ is the set of attributes that are associated with each node and that represent the properties associated with the social members. The node v_i has a vector $\{a_{i1}, a_{i2}, \ldots, a_{ik}\}$ where its value on attribute a_j is a_{ij}. Finally, the partition $P = \{C_1, C_2, \ldots, C_k\}$ is defined as the set of detected communities where members within communities are closely connected, having the same properties and they communicate regularly by exchanging messages. The dynamic of the network can be regarded as one of these possible events (1) a new node added, (2) an existing node deleted, (3) a new edge added, and (4) an existing edge deleted. We also consider an extra important dynamic event in the network which is the message exchange.

B. SIMILARITY FUNCTION

In this section, we present a new similarity measure that we use to define the node's attachment to a community. Based on this measure, we carry out the partition of the nodes. This measure is based on the different aspects of the network: the topological structure of the graph, the nodes' attributes, and the rate of transferred message. Thus this measure is composed of three parts as we present as follows.

1) SIMILARITY BASED ON MESSAGE DISTRIBUTION

In this work, we define the rate of exchanged messages between a node n and a community C as the average number of messages exchanged by n and its neighbors belonging to C. It is defined as follow:

$$rate\_msg_t\,(n, C) = \sum_{\forall n' \in N_t(n,C)} \frac{msg_t\,(n, n')}{nb_t\,(n, C)} \tag{4}$$

where $msg_t\,(n, n')$ is the number of messages transferred between the node n and its neighbor n' belonging to the community C at time t, $nb_t\,(n, C)$ is the neighbors' number of n in the community C at time t, and $N_t\,(n, C)$ is the set of neighbors of n belonging to C.

When two members don't exchange message between the two successive time steps, it is important to consider the communication previously carried out. So a node must still close to its community and does not abruptly abandon it as soon as it sends messages to another community. This is why we define the message distribution between a node n and a community C at time t as:

$$msg\_dist_t\,(n, C) = rate\_msg_t\,(n, C) + (1 - \beta)\, msg\_dist_{t-1}\,(n, C) \tag{5}$$

where β is a parameter that models the forgetfulness and it is between [0..1].

2) SIMILARITY OF ATTRIBUTES

In this work, we define the similarity of attributes of a node n with the members of the community C as the average of the similarities of n to its neighbors in C:

$$Sim_{att}\,(n, C) = \sum_{\forall n' \in N_t(n,C)} \frac{sim_{att}\,(n, n')}{|A|.nb_t\,(n, C)} \tag{6}$$

where $sim_{att}(n, n')$ is the number of common attributes' values between n and its neighbor n' belonging to C, $nb_t(n, C)$

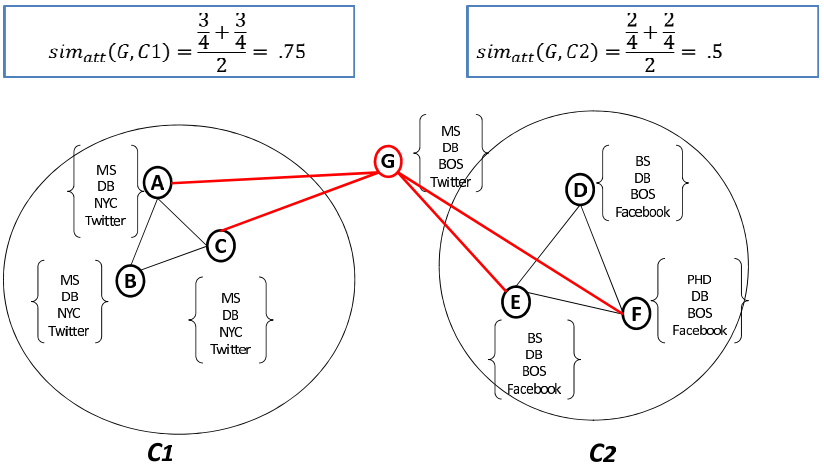

FIGURE 2. Similarity of attributes of a node to two communities.

is the number of neighbors of n in the community C at a time t, $N_t(n, C)$ is the set of neighbors of n belonging to C at a time t, and $|A|$ is the number of attributes.

In Figure.2, we present an example of calculation of the similarity of attributes of a node G to two communities. In this example, we assign to each node four attributes: degree, research area, location and affiliation. Based on the similarity of attributes presented in Equation.6, G is seen more similar to C_1.

3) SIMILARITY BASED ON TOPOLOGICAL STRUCTURE

In order to maximize the number of inter-community edges, we define the similarity based on topological structure of a node n to a community C as the rate of the neighbors of n belonging to C compared to the total number of the neighbors of n:

$$Sim_{str}(n, C) = \frac{nb_t(n, C)}{nb_t(n)} \tag{7}$$

where $nb_t(n, C)$ is the number of neighbors of n in C at time t, and $nb_t(n)$ is the total number of neighbors of n at time t.

To define the attachment of a node to a community, we consider the three defined similarities. Since the significance of each aspect varies from one social network to another, to each aspect is given a weight that measures the relative importance of that aspect compared to the others.

The similarity of a node n to a community C is then defined as follows:

$$Sim(n, C) = \alpha_1.msg_{dist_t}(n, C) + \alpha_2.Sim_{att}(n, C) + \alpha_3.Sim_{str}(n, C), \tag{8}$$

where α_1, α_2, and α_3 are the weights used for controlling the relative importance of each aspect. These parameters can be fixed depending on the importance and the availability of the data in the studied network.

C. ALGORITHM

Our algorithm starts with an initial partition that can be found by any community detection algorithm for a static graph. In this work, we use Louvain method [6] due to its good definition of a community structure that is based on modularity. In addition, this method outperforms many other modularity methods in terms of computation time [6] and it can handle large network datasets (see section II-A). In the next step, we update the graph by adding and removing the nodes and the edges according to the observed dynamic of the network seen between the time steps $t-1$ and t. Then, we select the nodes and the communities affected by the last network's transformations. Thereafter, we update the existing communities. In the end, we merge the communities that have an important number of overlapping nodes.

Figure.3 illustrates the rationale of the proposed method, which consists of two parts. First, the initial partition P_0 is obtained by using the Louvain model at time t_0. Next, a specific process for detecting dynamic communities is introduced.

Initial Community Detection: We employ the Louvain model to detect the partition at the initial time t_0.

Incremental Community Detection: After the detection of the initial partition, an incremental community discovery method is adopted at each time step. The process of incremental community detection mainly involves the following steps:

Step 1: Updating the graph according to the events seen in the network between the time steps $t-1$ and t by adding and/or removing nodes and/or edges, and selecting the nodes and the communities, affected by the last events, to be updated in the next step.

- *New Node:* when a new member joins the social network, a new node is added to the graph. All the new nodes are selected to be handled in the next step, i.e. they will be integrated into existing communities or they will create new communities.

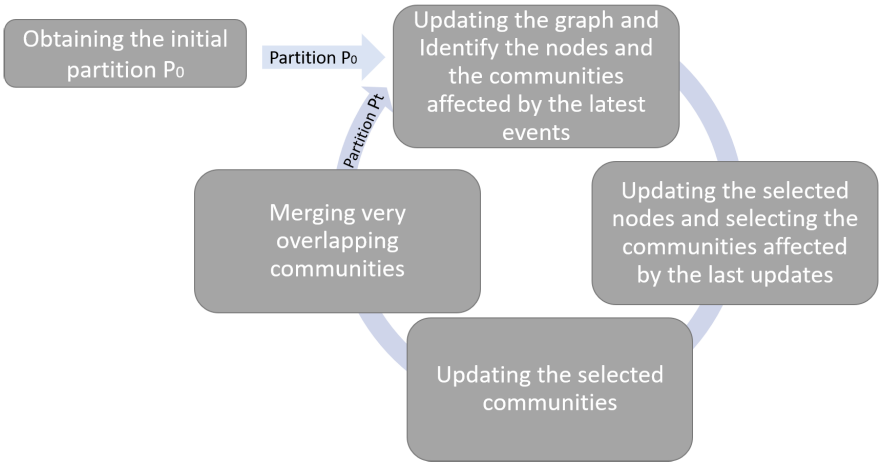

FIGURE 3. The framework of the proposed community detection *ASMsg*.

- *Remove of a Node:* when a member belonging to a community C disappears from the network, the node corresponding to this member in the graph and all its relationships (edges) are removed. The community C is selected to be updated (as it can be divided or removed).
- *New Edge:* when a new relationship is created between two members, a new edge is added to the graph between the nodes representing these members. If these nodes belonging to two different communities, these nodes are selected to be updated afterward.
- *New Message Between Two Nodes Belonging to Two Different Communities:* if two members belonging to two different communities exchange new messages, these nodes are also selected to be updated afterward.
- *Remove of an Edge:* when a relationship between two members is broken, we remove the edge connecting the two corresponding nodes. These nodes are selected to be updated if they belong to the same community. In this case, this community is also selected to be updated.

Step 2: Updating the selected nodes and selecting the communities affected by the last updates,

- *Handling the New Nodes:* In this work, we suppose that each new node n has at least a relationship with an existing member or with at least one new node. So in the first case, i.e. if n has some relationships with existing communities, then it is added to the most similar community using the similarity measure defined in Equation.8. But, if it is related to a set S of new nodes more than it is related to existing communities, a *new community* is created containing n and S.
- *Updating the Existing Nodes:* For each selected node n belonging to a community C, we check the possibility to move n to a neighboring community (with which it has new relationships and/or messages) using the similarity measure presented in Equation.8. n migrate to the community C' if its similarity to C' is greater than its similarity to C and all the neighboring communities. In this case, the community C is selected to be updated in the next step. We note that the node n can also be an overlapping node between two or more communities if it has the same similarity to them.

Step 3: Updating the selected communities

For each selected community, firstly we check its size. If it lost all its members, this community dies. If not, we check if this community has to be divided. For that, we compare the number of its internal edges by a threshold γ that we have chosen to be the half of the maximum number of edges expected inside the community. Indeed, the maximum number of internal edges expected in a community containing n nodes is:

$$Max = n(n-1)/2 \tag{9}$$

The threshold γ that we use in this model is the half of the maximum number of edges expected inside the community, that is:

$$\gamma = Max/2 \tag{10}$$

$$= n(n-1)/4 \tag{11}$$

So, if the number of the internal edges of a community C is less than γ, then the community C has to be divided. To this end, we apply the Louvain model to this community to detect the sub-communities.

Step 4: Merging overlapping communities

At the last step, we check the overlap between the detected communities and we merge those who have important numbers of overlapping nodes. In fact, we choose to merge the community C with the community C' if the number of overlapping nodes shared by C and C' exceeds a threshold δ. In this work, we have chosen that this threshold δ be 75% of the size of C. So if ($|overlap(C, C')| > \delta$.size(C)), then we merge C with C'.

D. COMPLEXITY ANALYSIS

The time complexity of the *ASMsg* is composed of the time complexity of the detection of the initial partition, and the time complexity of the incremental community detection.

1) INITIAL COMMUNITY PARTITION

The initial partition is detected using the Louvain method with the time complexity $O(nlog(n))$ such that n is the number of the nodes in the initial capture of the network.

2) INCREMENTAL COMMUNITY DETECTION

The incremental community detection includes the update of the graph, the selection of the nodes and the communities to be updated, the update of the nodes and the communities, and finally the merge of very overlapping communities.

The update of the graph consists of adding and deleting nodes and edges. The time complexity of adding and deleting nodes is $O(|\Delta V_t|.k_t)$, where ΔV_t is the set of added and deleted nodes and k_t is the average degree of these nodes at time step t. The time complexity of adding and deleting edges is $O(|\Delta E_t|)$ where ΔE_t is the set of added and deleted edges.

The next step requires updating the selected nodes. Let's $\Delta V'_t$ be the set of the selected nodes and k'_t be the average degree of these nodes. In this step, the similarity measure of Equation.8 is calculated for each node in $\Delta V'_t$. The time complexity to calculate the message distribution is $O(k'_t.|\Delta V'_t|)$. The time complexity to calculate the similarity of attributes is $O(k'_t.|A|.|\Delta V'_t|)$, where $|A|$ is the number of the attributes. The time complexity to calculate the similarity based on the topological structure is $O(k'_t.|\Delta V'_t|)$. Thus, the time complexity of this step is $O(|A|.k'_t.|\Delta V'_t|)$.

In the next step, we verify if the selected communities will be divided, that is done on time complexity $O(d_{in}.R)$, such that R is the number of the communities to verify and d_{in} is the average number of internal edges of these communities. The detection of the sub-communities is performed using Louvain model with a complexity equals $O(n_C log n_C)$ where n_C is the size of the community to be divided. Thus, the detection of the sub-communities is done on $O((n'_C.log.n'_C).R)$ such that n'_C is the average size of the communities to be divided and R denotes their number. Thus, the time complexity of this step is $O((d_{in} + (n'_C log n'_C)).R)$

Finally, the verification of the overlap and the merge of the communities is done on $O(n''_C.M)$ such that n''_C is the average size of the communities to be merged and M their number.

For conclusion, the time complexity of the *ASMsg* algorithm is divided into the complexity $O(((nlogn)$ at the time step t_0 and the time complexity $O((|\Delta V_t|.k_t) + |\Delta E_t| + (k'_t.|A|.|\Delta V'_t|) + ((d_{in} + (n'_C.log.n'_C)).R) + (n''_C.M))$ at the time step t. In general, k_t, k'_t, $|A|$, R and M are small. In addition, $|\Delta E_t|$ and d_{in} are negligible compared to the total number of edges existing at a time step t, as to $|\Delta V_t|$, n'_C and n''_C are negligible compared to the total number of nodes existing at a time step t. So the time complexity at each time step is relatively low. Therefore, the *ASMsg* method can be performed to identify the dynamic communities of large-scale networks.

IV. EXPERIMENTAL EVALUATION OF PERFORMANCES

We wanted to evaluate the performance of *ASMsg* using both real-world and artificial datasets. To extensively research the performance of *ASMsg*, we compared it with several state-of-the-art community detection algorithms. Before giving the results of this comparison, we briefly introduce the competing methods.

- *NMF* [20] method based on nonnegative matrix factorization (NMF) for discovering overlapping communities using the node features. First, the authors use the tensor's frontal slices in order to depict the adjacency matrix at each snapshot. Then, they apply a bayesian approach for ranking.
- *Agents* [19] is a multi-agent and incremental method for the detection of overlapping and dynamic communities. This method uses agents to observe the network's evolution and consequently update their communities. The update is performed using a similarity measure based on the topological structure of the network and the similarity of the attributes.
- *SmaCD* [33] is also a multi-agent and incremental method that is based on the similarity of attributes, the number of transferred messages, and the topological structure of the network. It uses two types of agents to observe the network's events and update the detected communities (see section II-B2).
- *iLCD* [7] is a dynamic clique-based method for detecting community structure (see section II-B1). This model is one of the well-known approaches that detect communities in dynamic networks. Several models of community detection have been compared with iLCD, we cite for example [8], [30] and [29].
- *DyPerm* [2] is also an incremental dynamic community detection method. It maximizes the permanence that is a local community-centric measure to detect the dynamic communities.
- *LBTR* [26] is an incremental method that is based on a machine learning. In this approach, the Louvain method is employed to obtain the initial partition. To update the existing communities, a machine learning method is implemented (see section II-B1).

A. DATASETS DESCRIPTION

1) REAL-WORLD DATASETS

To examine the effect of ASMsg, we selected a number of benchmark real-world dataset. Table 1 shows the description and the statistical properties of EIES network,[2] fb-forum network[2], fb-msg network[2], and ia-yahoo-msg network,[3] such that $|V|$ and $|E|$ denote the number of nodes and edges, and $|msg|$ denotes the number of messages.

[2]https://toreopsahl.com/datasets/
[3]http://networkrepository.com/dynamic.php

TABLE 1. Real-world networks characteristics.

Name	Description	|V|	|E|	|msg|
EIES	The Freeman's EIES network is a matrix with the number of messages sent among 32 researchers that used an electronic communication tool.	32	415	460
fb-forum	The Facebook-like Forum Network during $15/05/2004 - 24/10/2004$. The focus in this network is on users' activity in the forum. The weights are assigned based on the number of messages that the users posted.	899	33 720	33 720
fb-msg	The Facebook-like Social Network from an online community for students at University of California during $24/03/2004 - 22/10/2004$.	1 899	61 734	59 835
ia-yahoo-msg	The network message in Yahoo with time presented by link sequences.	100.1 K	6.3 M	2.5 M

2) ARTIFICIAL DATASETS

To get a related evaluation, it is important to know the expected dynamic nature, and to have the exact partitions with which we can compare our results. Since exact partitions are not always available for real-world graphs, we used artificial networks. Accordingly, we used the extended LFR model called LFRDA [24] in order to obtain artificial networks that are similar to real-world networks. Many parameters, such as the number of time steps, community size, mixing parameters, and several events that cause the network structure transformation, can easily monitor the extended LFR benchmark. In Table 2, we present a summary of LFRDA generator parameters. One of the most important parameters of this generator is the parameter for mixing communities that is defined by:

$$\mu(n \in C) = \frac{\sum_{\forall C' \in P, C' \neq C} comp(n, C')}{d(n)} \tag{12}$$

such that:

- $d(n)$: the degree of n (number of its neighbors).
- comp(n,C'): the number of neighbors of n in C'.

To check the validity of a community detection model on an artificial network, a comparison is performed between the obtained partition and the reference partition provided by the generator. The more the partition is closer to the reference partition, the more efficient is the community detection model. In the next section, we present the measure used to perform this comparison.

TABLE 2. Summary of LFRDA generator parameters.

Parameter	Meaning
n	Number of nodes
k	Average degree
μ	Parameter for mixing communities
$[minC, maxC]$	Minimum and maximum community sizes
s	Number of time steps
$\|A\|$	Number of attributes
p	Probability of one node switching community membership between time slices
birth	Number of community births for each time steps
death	Number of community deaths for each time steps
expand	Number of community expansions for each time steps
contract	Number of community contractions for each time steps
merge	Number of community mergers for each time steps
split	Number of community splits for each time steps

B. EVALUATION MEASURES

There are several measures used in the literature to determine the distance between two partitions, but the most used one is the Normalized Mutual Information (NMI) [11]. The mutual information of two random variables (X, Y) defined in probability theory and information theory as a quantity that measures the statistical dependence of these variables. In the probabilistic sense, the mutual information of a couple (X, Y) of variables represents their degree of dependence. The *NMI* is defined as:

$$NMI(A, B) = \frac{-2 \sum_{a \in A} \sum_{b \in B} |a \cap b| \log(\frac{|a \cap b| n}{|a||b|})}{\sum_{a \in A} |a| \log(\frac{|a|}{n}) + \sum_{b \in B} |b| \log(\frac{|b|}{n})}, \tag{13}$$

such that A and B are two distinct partitions of the same graph.

For real-world networks, the obtained results are validated by the weighted modularity that assesses the quality of the obtained partition according to the internal and external edges of its communities. It is the extension of the modularity for weighted graphs, and it is defined as [22]:

$$Q_w(P) = \frac{1}{2m} \sum_{ij} [A_{ij} - \frac{k_i k_j}{2m}] \delta(C_i, C_j) \tag{14}$$

where A_{ij} represents the weight of the edge between i and j, $m = \frac{1}{2} \sum_{ij} A_{ij}$, k_i is the degree of i (it is the sum of the weights of the edges of the node i), k_j is the degree of j, C_i is the community to which the node i is assigned, C_j that of j, and $\delta(x, y)$ equals one if $x = y$ and zero otherwise. Since the modularity is based on the edges of a network, the higher the number of edges inter-communities, the lower the modularity becomes.

The second measure used for evaluation of the competing methods in real-world networks is the performance, which counts the number of correctly ''interpreted'' pairs of nodes, i.e. two nodes that belong to the same community and that are connected by an edge, or two nodes that belong to different communities and are not connected by an edge. The definition of performance, for a partition P, is [11]:

$$Performance(P) = \frac{|\{(i,j) \in E, C_i = C_j\}| + |\{(i,j) \notin E, C_i \neq C_j\}|}{n(n-1)/2} \tag{15}$$

By definition, $0 \leq Performance(P) \leq 1$.

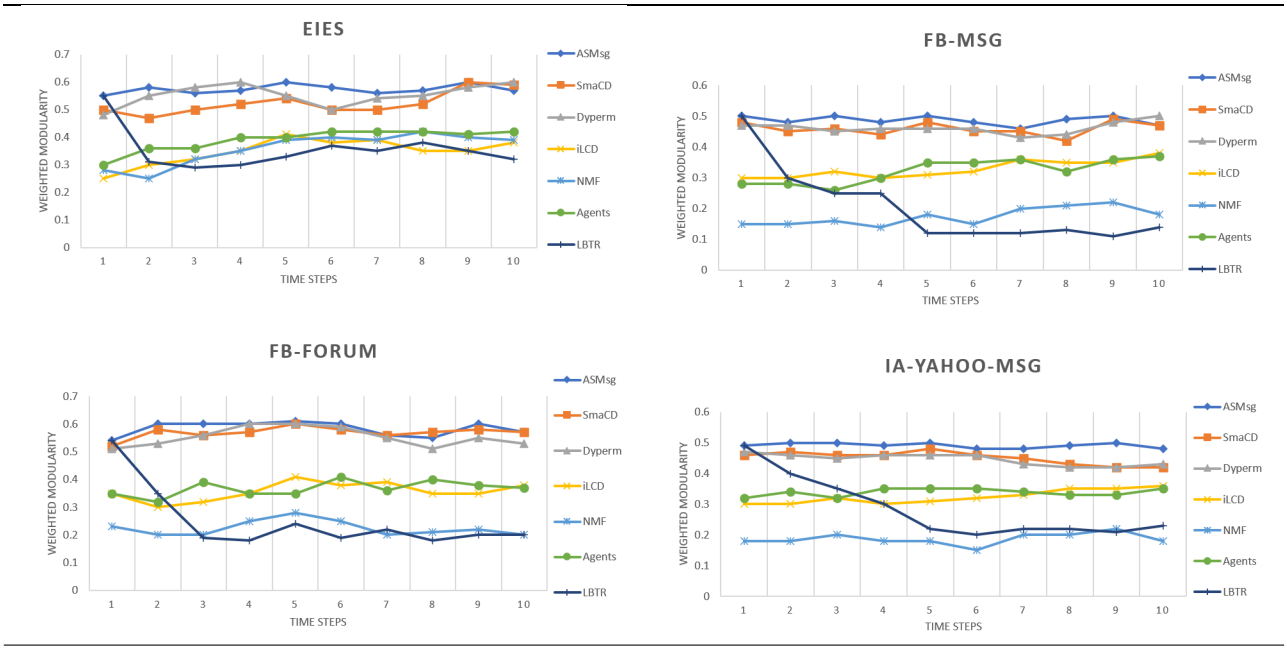

FIGURE 4. Performances of the competing methods on the "weighted modularity metric" for real-world networks.

C. EXPERIMENTAL SETUP

These experiments were performed on Intel Core i5 at 2.5 GHz and 8 GB of RAM under Windows 7 operating system. We used JAVA as the programming language to implement and execute the program codes and NetBeans Java Integrated Development Environment as our standard IDE.

In these experiments, we chose to give the same importance value for each aspect in the defined similarity measure, i.e., (0.33, 0.33, 0.33) for $(\alpha_1, \alpha_2, \alpha_3)$ in Equation.8. However, if the attributes or the number of exchanged messages are not provided in the dataset, we gave the same weight (0.5) for the remaining aspects. For the parameter β in Equation.5, we have varied its value during these experiments, and the best results were given when it equals 0.05.

D. RESULTS AND DISCUSSION

1) REAL-WORLD NETWORKS RESULTS

In these experiments, we tested the performances of the different methods on real-world dynamic networks. The weighted modularity and the performance are used to evaluate the efficacy of these algorithms. The evaluation results can be seen in Figures.4 and 5.

In terms of weighted modularity, *ASMsg*, *DyPerm* and *SmaCD* achieved the highest community detection efficiency and obtained weighted modularity values of approximately 0.6 for EIES network and Fb-forum network, 0.5 for Fb-msg network and ia-yahoo-msg network, with a minor superiority for *ASMsg*. However, *Agents* and *iLCD* obtained reasonable efficiency, and the weighted modularity values were better than those of the *NMF* and *LBTR* methods for Fb-forum network, Fb-msg network and ia-yahoo-messages network. For EIES network, the results of *Agents*, *iLCD*, *NMF* and *LBTR* were very close. For *LBTR*, initially, it introduced the same weighted modularity value as *ASMsg* since this algorithm uses the Louvain method to detect the initial partition. But its results declined gradually over time.

In terms of performance index, as shown in Figure.5, *ASMsg* performed very well for EIES and ia-yahoo-msg networks, where it obtained the highest performance values. *SmaCD*, *Agents*, *Dyperm* and *iLCD* achieved good quality, and the detected partitions were better than those of *LBTR* and *NMF*.

For fb-msg and fb-forum networks, *ASMsg* performed nearly the same as *SmaCD* with the highest performance. Moreover *Agents*, *Dyperm*, and *iLCD* performed similarly as the performance values at each time steps were almost the same. In contrast, *LBTR* declined rapidly. In regards to *NMF*, it did not perform well in these networks and it obtained the lowest values of performance.

2) ARTIFICIAL NETWORKS RESULTS

To assess the efficiency of *ASMsg*, we used the dynamic LFRDA benchmark model to create several synthetic networks with distinct features. We analyzed multiple states of community structures to cover all possible transformation of the communities: node switch, birth and death of the community, extension and contraction of the community, merge and division of the community, and the variation of the complexity of the networks (the rate of inter-communities edges compared to the total number of edges). The common parameters of these networks were set as follows, without loss of generality: time steps s = 10, number of nodes in each time step $n = 5000$, average degree $k = [10 - 20]$, the parameter for mixing communities $\mu = 0.4$, and max degree $max = 40$.

Node Switch: It refers to a node migration from one community to another in a dynamic network in various

FIGURE 5. Performances of the competing methods on the "performance metric" for real-world networks.

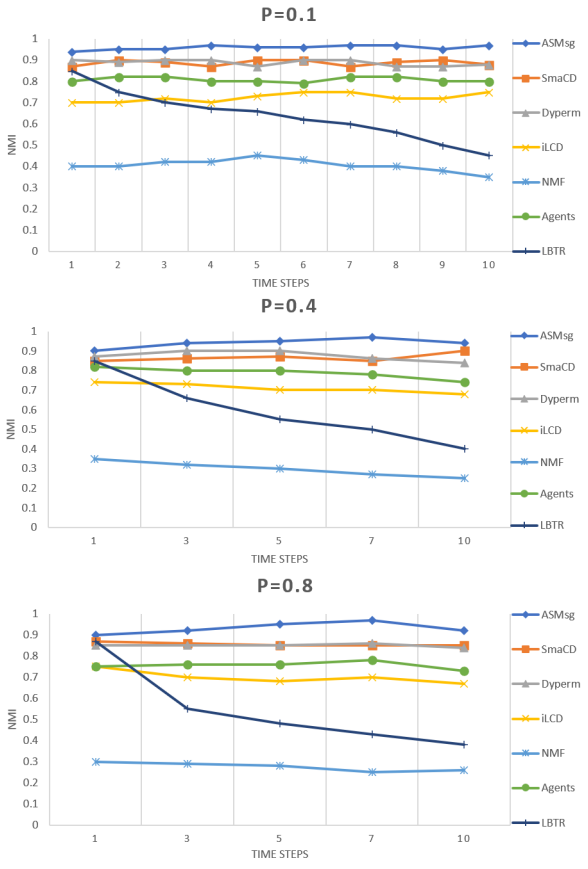

FIGURE 6. NMI of the competing methods on artificial networks with different switch probability *p*.

time steps. The parameter p reflects the probability that a node migrates from its community to another. The value of p has been varied from 0.1 to 0.8 and we have fixed the parameters $k = [10 - 20]$, and $\mu = 0.4$. We only give the results when p is 0.1, 0.4, 0.6 and 0.8, because of the limitation of space.

Figure.6 shows the NMI values of the different approaches for networks with different switch probability p. The best effect was acquired by the *ASMsg* method and the NMI values obtained on each time step were approximately 0.97. This shows that its effect was relatively constant over time. *Dyperm* and *SmaCD* also performed well, reaching 0.90 in the NMI values. The performance of *Agents* was also stable, and the NMI values were effectively held at approximately 0.85. For *iLCD*, the NMI values were successfully maintained at 0.75. For *LBTR*, initially, very high NMI values were introduced, but its results were gradually declined over time. By contrast, *NMF* did not perform well and obtained the lowest values of NMI.

Community Birth and Death: In order to examine the efficiency of the different competing methods in the case of community birth and death, we varied birth (B) and death (D) parameters from 2 to 8 (knowing that the maximum value that can be given to them is limited by the generator at 8). We only give the results when (B) and (D) parameters are 4, 6, and 8.

Figure.7 reveals the performance of competing methods on NMI metric with different numbers of community birth and death. From these results, we can conclude that *ASMsg* was very stable. It obtained the highest values of NMI and it reached 0.96 in these graphs. *DyPerm* and *SmaCD* also performed well with the NMI values stable at approximately 0.85. *Agents* achieved acceptable results, and it obtained an NMI value equals 0.75. *LBTR* started with very good quality, but this performance degraded rapidly over time. In contrast,

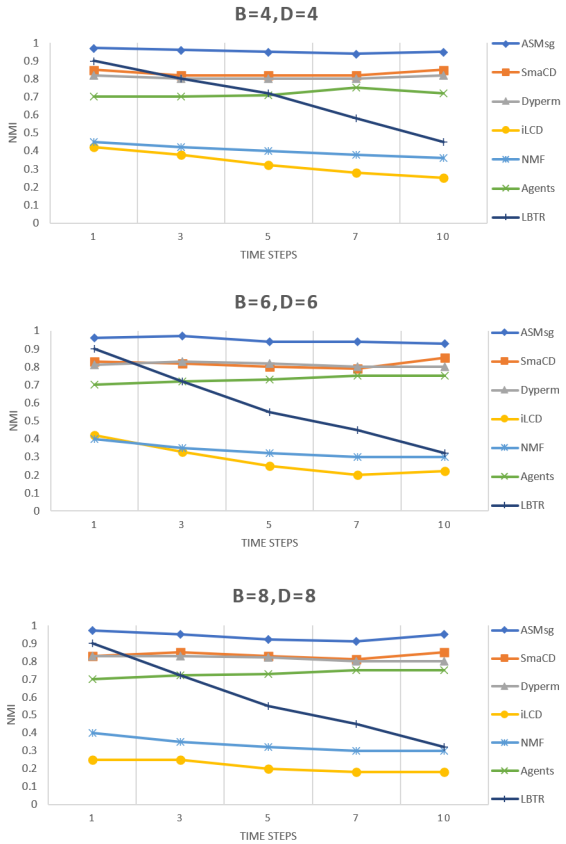

FIGURE 7. NMI of the competing methods on artificial networks with different community birth (B) and death (D) values.

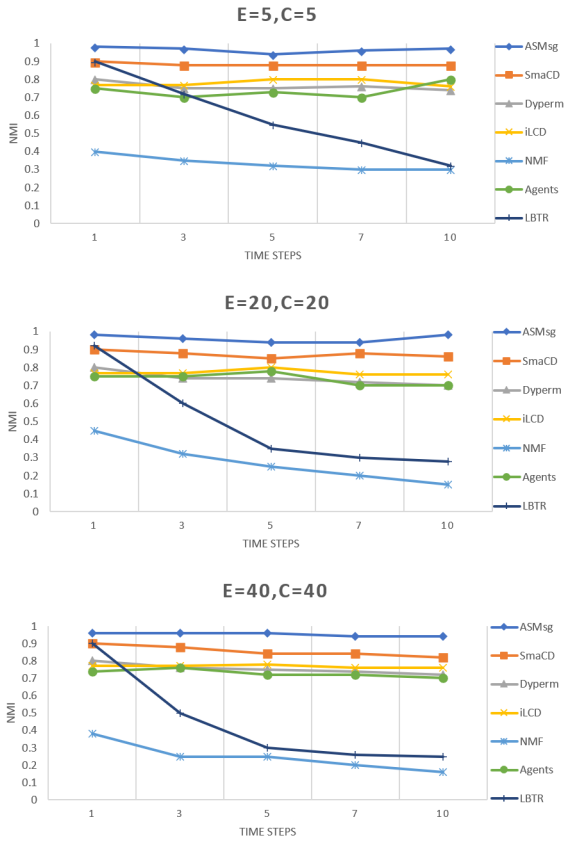

FIGURE 8. NMI of the competing methods on artificial networks with different community expansion (E) and contraction (C) values.

NMF did not perform well and obtained NMI values that did not exceed 0.45 when B and D were less than 6 and 0.4 for the rest of the networks. *iLCD* obtained the lowest values of NMI because this model did not allow the death of communities.

Community Expansion and Contraction: To investigate the efficiency of the competing methods on the expansion and contraction events, the number of expansion (E) and contraction (C) were varied from 5 to 40. We only give the results when these parameters are 5, 20, and 40.

As shown in Figure.8, *ASMsg* acquired the best quality of partitions. Its NMI values varied between 0.94 and 0.98. *SmaCD* also yielded good results, and its NMI values were stable at approximately 0.9. For *DyPerm*, *iLCD*, and *Agents* methods, they achieved stable NMI values at approximately 0.8. Contrariwise, the effectiveness of *NMF* and *LBTR* methods decreased gradually over time.

Community Merge and Split: To more evaluate of the efficiency of each model, we varied the number of merge (M) and split (S) from 5 to 40. We only give the results when these parameters are 5, 20, and 40.

Figure.9 describes the efficacy of the competing methods in these experiments. It can be concluded that *ASMsg* performed better than the other methods, and the NMI values were maintained between 0.9 and 0.95. *SmaCD* also yielded good results, and its NMI values were between 0.80 and 0.87. *Agents* and *DyPerm* achieved acceptable results, and their NMI values were between 0.7 and 0.8. The results of *iLCD* decreased when the number of merged and divided communities increased. The performance of *NMF* and *LBTR* also decreased gradually over time.

Networks With Different Mixing Parameter Values: In these experiments, we varied the mixing parameter μ from 0.1 to 0.8. For the rest of the parameters, the default values were considered.

Figure.10 shows the performance of the competing methods with different μ on NMI metric. We only give the results when μ equals 0.1, 0.6, and 0.8. The best effects were acquired by the *ASMsg*, *Dyperm* and *SmaCD* methods and the obtained NMI values varied between 0.9 and 0.98, when μ was less than 0.5. However, the effectiveness of *Dyperm* and *SmaCD* methods decreased gradually when μ exceeds 0.5, opposite to *ASMsg* that successfully maintained the NMI values at 0.95. This indicates that *ASMsg* was consistent in producing exact partition across different complexity of networks. *Agents* also performed well, reaching 0.80 in the NMI values and its performance was also stable when μ increased. For *iLCD*, the NMI values were successfully

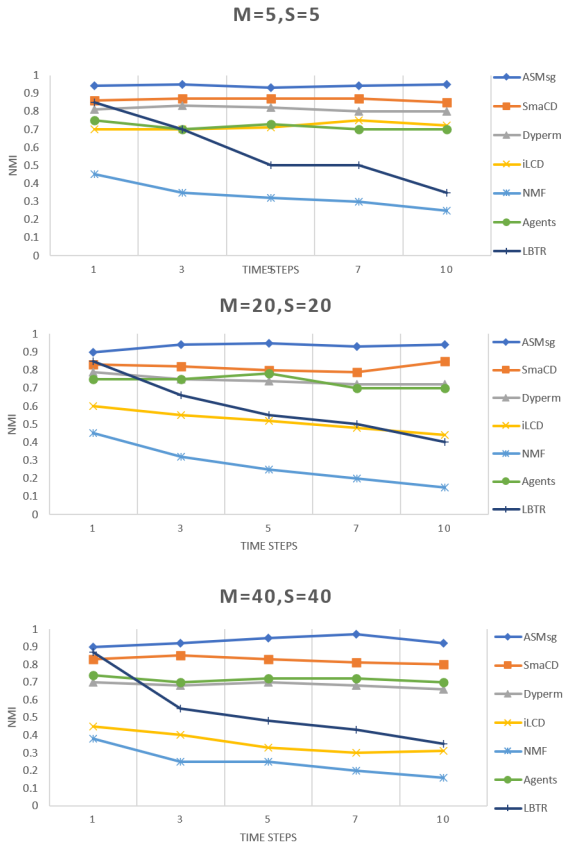

FIGURE 9. NMI of the competing methods on artificial networks with different community merge (M) and split (S) values.

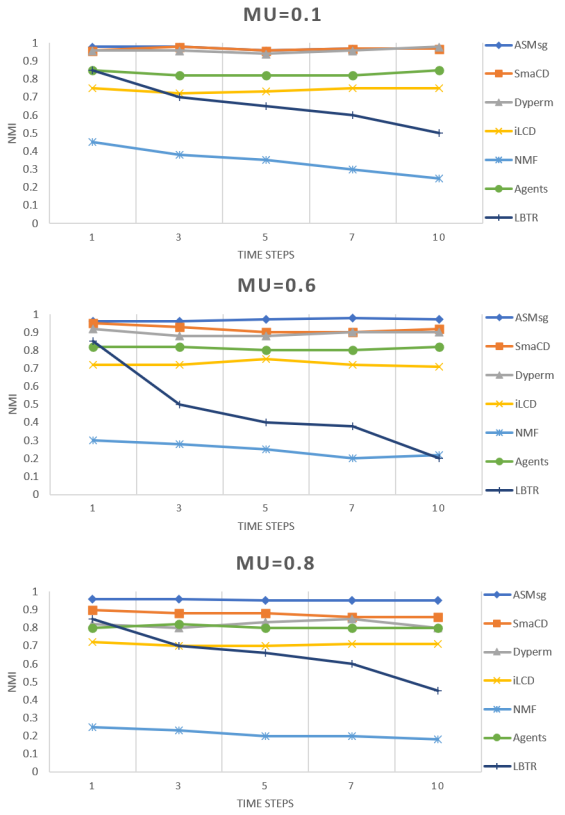

FIGURE 10. Performance, in terms NMI, of the competing methods on artificial networks with different mixing parameter values.

maintained at 0.70. By contrast, *LBTR* and *NMF* did not perform well and obtained the lowest values especially when μ exceeds 0.4.

3) EVALUATION OF SCALABILITY AND RUNTIME

To examine the scalability and the runtime of *ASMsg*, we employed LFRDA to generate dynamic networks with different scales. The number of nodes was varied from 1K to 1M and we fixed the parameters $k = [10 - 20]$, $maxk = 40$, $p = 0.4$, $\mu = 0.4$, and $s = 10$. Figure.11 shows the running time of the competing methods in ten time steps. We noticed that *DyPerm*, *Agents* and *SmaCD* algorithms used the highest runtime. In fact, the *DyPerm* runtime took three days when the number of nodes exceeds 400k nodes. The same runtime was reached by *Agents* and *SmaCD* when the number of nodes exceeded 500k nodes. The *ASMsg* algorithm was faster than these methods. This advantage maintained as the size of the network increased. Therefore, *ASMsg* can handle large-scale dynamic networks. For *NMF*, when the number of nodes reaches 300K, it indicated that the available memory space to run was insufficient. Moreover, although *iLCD* and *LBTR* were faster than *ASMsg*, all the previous experiments prove that its quality was better than these approaches.

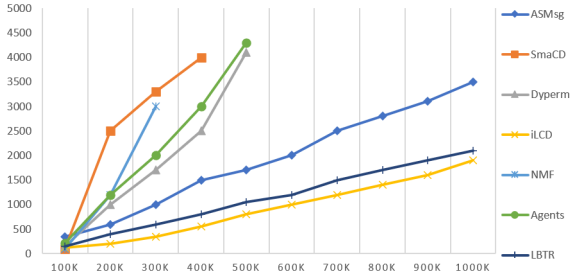

FIGURE 11. The runtimes of the competing methods on the dynamic LFRDA benchmark with nodes ranging from 1K to 1M.

V. CONCLUSION

In this work, we propose a new method for the identification of overlapping communities in dynamic social networks. This model allows to track the evolution of the communities over time. To do so, we started by the definition of a similarity measure to evaluate the node relationship to a community. For this measure, we considered three important aspects: the message distribution, the similarity of attributes, and the topological structure of the network. As a result, the members of each community are similar, they have good communications between them, and structurally they are well related. We have put no restrictions either on social members' dynamics or on

the dynamics of their relationships: members join and leave the network; relationships are created and broken. The results of the experiments on both real-world and artificial networks show that we were able to detect the real-world evolution of the communities whatever the nature of the confronted dynamic.

Throughout this study, we have assumed, like most of the proposed research for the community detection, that individuals are related only by ''positive relationships'' (i.e., generally relationships with friendships). Additionally, we measured the communication between users in terms of the number of exchanged messages. But, in online social networks, users can also communicate by exchanging comments. Thus, each user may be identified according to the semantic content of the data that he creates or manipulates. Therefore, we can be able to define users' communities that share the same semantic connections. Since it is not common to have a single opinion on all topics, users can be negatively linked in the case of disagreement of opinions between users, and that will be our objective in further works.

REFERENCES

[1] E. A. Abdulkreem, H. Zardi, and H. Karamti, ''Community detection in dynamic social networks: A multi-agent system based on electric field,'' *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 1, pp. 493–504, 2019.

[2] P. Agarwal, R. Verma, A. Agarwal, and T. Chakraborty, ''DyPerm: Maximizing permanence for dynamic community detection,'' 2018, *arXiv:1802.04593*. [Online]. Available: https://arxiv.org/abs/1802.04593

[3] Y. Asim, R. Ghazal, W. Naeem, A. Majeed, B. Raza, and A. Kamran, ''Community detection in networks using node attributes and modularity,'' *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 1, pp. 382–388, 2017.

[4] G. A. Bello Lander, ''Multi-objective graph mining algorithms for detecting and predicting communities in complex dynamic networks,'' M.S. thesis, North Carolina State Univ., Raleigh, NC, USA, 2017.

[5] S. K. Bisma and A. N. Muaz, ''Network community detection: A review and visual survey,'' 2017, *arXiv:1708.00977*. [Online]. Available: https://arxiv.org/abs/1708.00977

[6] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, ''Fast unfolding of communities in large networks,'' *J. Stat. Mech.: Theory Exp.*, vol. 2008, no. 10, Oct. 2008, Art. no. P10008.

[7] R. Cazabet, ''Detection of dynamic communities in temporal networks,'' Ph.D. dissertation, Univ. Paul Sabatier-Toulouse III, Toulouse, France, 2013.

[8] T. Chakraborty and A. Chakraborty, ''OverCite: Finding overlapping communities in citation network,'' in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining (ASONAM)*, Aug. 2013, pp. 1124–1131.

[9] A. Clauset, M. E. J. Newman, and C. Moore, ''Finding community structure in very large networks,'' *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 70, no. 6, Dec. 2004, Art. no. 066111.

[10] E. Desmier, M. Plantevit, C. Robardet, and J.-F. Boulicaut, ''Cohesive co-evolution patterns in dynamic attributed graphs,'' in *Proc. Int. Conf. Discovery Sci.*, 2012, pp. 110–124.

[11] S. Fortunato, ''Community detection in graphs,'' *Phys. Rep.*, vol. 486, nos. 3–5, pp. 75–174, Feb. 2010.

[12] S. Fortunato and D. Hric, ''Community detection in networks: A user guide,'' *Phys. Rep.*, vol. 659, pp. 1–44, Nov. 2016.

[13] M. Girvan and M. E. J. Newman, ''Community structure in social and biological networks,'' *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 12, pp. 7821–7826, Jun. 2002.

[14] D. Greene, D. Doyle, and P. Cunningham, ''Tracking the evolution of communities in dynamic social networks,'' in *Proc. Int. Conf. Adv. Social Netw. Anal. Mining (ASONAM)*, N. Memon and R. Alhajj, Eds. Aug. 2010, pp. 176–183.

[15] J. Hopcroft, O. Khan, B. Kulis, and B. Selman, ''Tracking evolving communities in large linked networks,'' *Proc. Nat. Acad. Sci. USA*, vol. 101, no. 1, pp. 5249–5253, Apr. 2004.

[16] S. S. Hoseini and S. H. Abbasi, ''A new method for community detection in social networks based on message distribution,'' *Int. J. Comput. Sci. Netw. Secur.*, vol. 45, no. 5, pp. 298–308, 2017.

[17] D. Jin, X. Wang, D. He, J. Dang, and W. Zhang, ''Robust detection of link communities with summary description in social networks,'' *IEEE Trans. Knowl. Data Eng.*, early access, Dec. 10, 2019, doi: 10.1109/TKDE.2019.2958806.

[18] K. Guo, T. Zhu, and G. Hui Li, ''Incremental dynamic community discovery algorithm based on improved modularity,'' in *Proc. 2nd IEEE Int. Conf. Comput. Commun. (ICCC)*, Oct. 2016, pp. 2536–2541.

[19] A. Mahfoudhi, H. Zardi, and M. A. Haddar, ''Detection of dynamic and overlapping communities in social networks,'' *Int. J. Appl. Eng. Res.*, vol. 13, no. 11, pp. 9109–9122, 2018.

[20] R. Márquez, ''Overlapping community detection in static and dynamic networks,'' in *Proc. 13th Int. Conf. Web Search Data Mining*, New York, NY, USA, Jan. 2020, pp. 925–926.

[21] M. E. J. Newman, ''The structure and function of complex networks,'' *SIAM Rev.*, vol. 45, no. 2, pp. 167–256, 2003.

[22] M. E. Newman, ''Analysis of weighted networks,'' *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 70, no. 5, Nov. 2004, Art. no. 056131.

[23] M. E. J. Newman and M. Girvan, ''Finding and evaluating community structure in networks,'' *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 69, no. 2, Feb. 2004, Art. no. 026113.

[24] G. K. Orman, V. Labatut, M. Plantevit, and J.-F. Boulicaut, ''Interpreting communities based on the evolution of a dynamic attributed network,'' *Social Netw. Anal. Mining*, vol. 5, no. 1, pp. 1–22, Dec. 2015.

[25] G. Palla, A.-L. Barabási, and T. Vicsek, ''Quantifying social group evolution,'' *Nature*, vol. 446, no. 7136, pp. 664–667, Apr. 2007.

[26] J. Shang, L. Liu, X. Li, F. Xie, and C. Wu, ''Targeted revision: A learning-based approach for incremental community detection in dynamic networks,'' *Phys. A, Stat. Mech. Appl.*, vol. 443, pp. 70–85, Feb. 2016.

[27] L. Waltman and N. J. van Eck, ''A smart local moving algorithm for large-scale modularity-based community detection,'' 2013, 2013, *arXiv:1308.6604*. [Online]. Available: https://arxiv.org/abs/1308.6604.

[28] J. J. Whang, D. F. Gleich, and I. S. Dhillon, ''Overlapping community detection using neighborhood-inflated seed expansion,'' *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 5, pp. 1272–1284, May 2016.

[29] J. Xie, M. Chen, and B. K. Szymanski, ''LabelrankT: Incremental community detection in dynamic networks via label propagation,'' in *Proc. Workshop Dyn. Netw. Manage. Mining*, New York, NY, USA, Jun. 2013, pp. 25–32.

[30] J. Xie, S. Kelley, and B. K. Szymanski, ''Overlapping community detection in networks: The state-of-the-art and comparative study,'' *ACM Comput. Surv.*, vol. 45, no. 4, p. 43, 2013.

[31] Y. Xia and L. Tuo, ''An incremental community mining method in dynamic social networks,'' in *Proc. IEEE 3rd Int. Conf. Cloud Comput. Intell. Syst.*, Nov. 2014, pp. 305–309.

[32] N. Zarayeneh and A. Kalyanaraman, ''A fast and efficient incremental approach toward dynamic community detection,'' in *IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining (ASONAM)*, F. Spezzano, W. Chen, and X. Xiao, Eds. Aug. 2019, pp. 9–16.

[33] H. Zardi, ''Community detection in dynamic social network: Multi-agent approach,'' M.S. thesis, Serbone Univ., Paris, France, 2016.

[34] H. Zardi and L. B. Romdhane, ''An O(n^2) algorithm for detecting communities of unbalanced sizes in large scale social networks,'' *Knowl.-Based Syst.*, vol. 37, pp. 19–36, Jan. 2013.

[35] H. Zare, M. Hajiabadi, and M. Jalili, ''Detection of community structures in networks with nodal features based on generative probabilistic approach,'' *IEEE Trans. Knowl. Data Eng.*, early access, Dec. 17, 2020, doi: 10.1109/TKDE.2019.2960222.

[36] Y. Zhang, B. Wu, N. Ning, C. Song, and J. Lv, ''Dynamic topical community detection in social network: A generative model approach,'' *IEEE Access*, vol. 7, pp. 74528–74541, 2019.

[37] C. Zhe, A. Sun, and X. Xiao, ''Community detection on large complex attribute network,'' in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019.

HEDIA ZARDI was born in Sousse, Tunisia, in 1986. She received the bachelor's and master's degrees in computer science from the University of Monastir, Tunisia, in 2009 and 2011, respectively, and the Ph.D. degree in computer science from Sorbonne University, France, in 2016.

From 2017 to 2019, she was an Assistant Professor with the University of Monastir. She is currently an Assistant Professor with Qassim University, Saudi Arabia. She is the author/coauthor of several articles in international conferences and journals. Her current research interests include social networks, graph mining, and multi-agents systems.

BUSHRA ALHARBI was born in Qassim, Saudi Arabia, in 1992. She received the bachelor's degree in information technology and the master's degree in science of informatics from Qassim University, Saudi Arabia, in 2015 and 2020, respectively.

Her research interests include graph mining and social networks.

WALID KARAMTI was born in Sfax, Tunisia, in 1985. He received the bachelor's, master's, and Ph.D. degrees in computer science from the University of Sfax, Tunisia, in 2008, 2010, and 2015, respectively.

From 2015 to 2019, he was an Assistant Professor with the University of Monastir, Tunisia. He is currently an Assistant Professor with Qassim University, Saudi Arabia. He is the author/coauthor of several articles in international conferences and journals. His current research interests include formal methods for the analysis of behaviors in real time systems and social networks.

HANEN KARAMTI received the bachelor's degree in computer science and multimedia from the Higher Institute of Computer Science and Multimedia, Sfax University, Tunisia, the Master of Computer Science and Multimedia degree from Sfax University, in 2011, and the Ph.D. degree in computer science from the National Engineering School of Sfax, Tunisia, in 2017, in cooperation with the University of La Rochelle, France, and the University of Hanoi, Vietnam.

She is currently an Assistant Professor with Princess Nourah Bint Abdulrahman University, Riyadh, Saudi Arabia. Her research interests include information retrieval, multimedia systems, image retrieval, health informatics, big data, and data analytics.

EATEDAL ALABDULKREEM received the Ph.D. degree in computer science from Brunel University, U.K.

She is currently an Assistant Professor with the Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, Riyadh, Saudi Arabia. Her research interests include artificial intelligence, cognitive computing, and networks. She has published several research articles in her field.