Computational Intelligence

AN INTERNATIONAL JOURNAL

Computational Intelligence, Volume 33, Number 2, 2017

EVOLUTIONARY CLUSTERING FOR MINING AND TRACKING DYNAMIC MULTILAYER NETWORKS

ALESSIA AMELIO AND CLARA PIZZUTI

National Research Council of Italy (CNR), Institute for High Performance Computing and Networking (ICAR), Rende (CS), Italy

This article proposes a framework for community discovery in temporal multiplex networks by extending the evolutionary clustering approach to encompass both time and multiple dimensions. In this extended framework, the problem of finding community structures for time-evolving networks with multiple types of ties is reformulated by adding the concept of dimensional smoothness, relative to a single timestamp, to that of temporal smoothness, at the base of evolutionary clustering. At each timestamp, the method tries to maximize the quality of the clustering obtained for the current multidimensional network and to minimize the differences with respect to that obtained at the previous timestamp. Moreover, the evolution of a community between two consecutive timestamps is maintained by exploiting the Hungarian approach, which determines the best cluster correspondence between two consecutive timestamps. Experiments on synthetic and real-world networks show the capability of the approach in discovering and tracking group organization of actors constituting the network.

Received 18 November 2014; Revised 31 July 2015; Accepted 31 July 2015

Key words: multilayer networks, dynamic networks, evolutionary clustering, multiobjective clustering.

1. INTRODUCTION

In recent years, we are witnessing an increasing interest in the analysis of complex systems represented as networks. Characterizing network topology is an important aspect to understand its properties and dynamic behavior. Basic components of real-world systems are generally connected by multiple and diverse relationships, often occurring at different time points. For instance, the flow of information generated by people who communicate and connect through social media, mobile telephone calls, or e-mail messages contains detailed information on the time the event took place and the kind of interaction that was established. In system biology, proteins can interact at different times and have different types of interactions. Research in network science recently realized that the classical approach of aggregating these composite interactions among the actors constituting a network provokes the loss of important information regarding the original system, which could be exploited to better analyze and describe its features. As pointed out in Kivelä et al. (2014) and Battiston, Nicosia, and Latora (2013), networks with multiple types of interactions provide a much more complete description of a system than monoplex networks, generated by the aggregation of these connections on a single kind of tie, because each connection can have different meanings and roles.

In this article, a new framework based on multiobjective optimization (Deb 2001; Coello et al. 2007) and local search is proposed to deal with the problem of detecting community structure in temporal multilayer networks and tracking their evolution along time. The method, named *DMultiMOGA* (Dynamic Multilayer MultiObjective Genetic Algorithm), extends the evolutionary clustering framework (Chakrabarti, Kumar, and Tomkins 2006) to encompass both time and multiple dimensions. In this extended framework, the problem of finding community structures for time-evolving networks with multiple types of

Address correspondence to Clara Pizzuti, National Research Council of Italy (CNR), Institute for High Performance Computing and Networking (ICAR), Rende (CS) 87036, Italy; e-mail: pizzuti@icar.cnr.it

© 2016 Wiley Periodicals, Inc.

ties is reformulated by adding the concept of dimensional smoothness, relative to a single timestamp, to that of temporal smoothness, at the base of evolutionary clustering (Chakrabarti et al. 2006). *DMultiMOGA* optimizes the two objectives of facet quality $\mathcal{FQ}$ and dimensional sharing $\mathcal{SQ}$ inside a fixed timestamp, and facet quality $\mathcal{FQ}$ and temporal cost $\mathcal{TC}$ when a new timestamp starts. The first two objectives guarantee dimensional smoothing among the layers of a multidimensional network at a current timestamp, while the substitution of $\mathcal{SQ}$ with $\mathcal{TC}$, when a new timestamp begins, ensures temporal smoothness between consecutive timestamps.

It is worth to point out that, at each timestamp t, layers can be rather different because both nodes and interactions could be present in a layer but missing in another one. As a consequence, there could be isolated nodes not assigned to any community. To dampen this phenomenon, *DMultiMOGA* performs a local label propagation for each isolated node v by considering the neighbors of v in all the dimensions and then assigning to v the most recurring class label of its neighbors in the current community structure.

Moreover, *DMultiMOGA*, by exploiting the Hungarian algorithm for optimal assignment problem (Kuhn 1955), determines the best correspondence among the communities obtained between two consecutive timestamps and maintains a trace of the evolution of each community along the time.

Experiments on synthetic and real-world networks show that the approach is capable to detect accurate community structures in temporal multilayer networks and maintain a track of their evolution as time passes.

The article is organized as follows. The next section gives an overview of the most recent proposals in the field. Section 3 defines temporal multidimensional networks and the problem of community detection. Section 4 formalizes the problem as a multiobjective optimization problem. Section 5 describes the proposed approach. In Section 6, the results of the experiments are reported. Finally, Section 7 concludes the article.

2. RELATED WORK

In the last few years, the research in multilayer networks is receiving more attention. Many studies are focusing on their analysis and mathematical formulation, along with extensions of the metrics defined for characterizing single-layer networks to the multilayer ones (Battiston et al. 2013; De Domenico et al. 2013, 2014; Magnani, Micenkova, and Rossi 2013; Kivelä et al. 2014).

Although there exists a plenty of methods for community detection in single-layer networks, approaches for multilayer networks are still at very beginning. Moreover, as regards the temporal information, a number of methods have been proposed for analyzing and tracking dynamic single-layer networks, but approaches that detect communities in multilayer networks and track these communities over time have not been yet developed.

A popular framework for studying single-layer dynamic networks is evolutionary clustering (Chakrabarti et al. 2006). Evolutionary clustering has been proposed by Chakrabarti et al. for data clustering, and it relies on the idea of temporal smoothness, that is, it assumes that rapid changes in a short time period are unlikely to happen. Chi et al. (2009) specialized this concept to dynamic networks and defined a cost function $cost = \alpha \cdot \mathcal{SC} + (1-\alpha) \cdot \mathcal{TC}$ composed of two terms: the snapshot quality $\mathcal{SC}$, which measures how much the obtained clustering reflects latent community structure at the current timestamp, and the temporal cost $\mathcal{TC}$, which biases consecutive clusterings to do not sensibly differ from one timestamp to the next one. α is an input parameter used by the user to emphasize one of the two objectives. When $\alpha = 1$, the approach returns the clustering without temporal smoothing. When $\alpha = 0$, the same clustering of the previous timestamp is produced. This cost function has

been adopted by Lin et al. (2009), Kim and Han (2009), and Folino and Pizzuti (2014). In particular, Lin et al. employ a stochastic block model for generating communities and a probabilistic model based on the Dirichlet distribution to catch community evolutions. One of the main drawbacks of this method is that it assumes a fixed number of communities over time. In Folino and Pizzuti (2014), the detection of community structure with temporal smoothness has been formulated as a multiobjective optimization problem, where the first objective is the maximization of the snapshot quality, achieved through the optimization of the modularity concept (Newman and Girvan 2004), and the second objective is the minimization of the temporal cost, fulfilled by maximizing the normalized mutual information (NMI) (Danon et al. 2005) between the community structure obtained at the current timestamp and that obtained at the previous one.

Mucha et al. (2010) introduced a general framework encompassing time-evolving, multiscale, and multidimensional networks for determining community structure in multislice networks. These networks consist of multiple adjacency matrices, where each adjacency matrix can indifferently represent variations across time (dynamic networks), variations across different types of connections (multidimensional networks), or even the same network at different scales. Communities can then be detected by applying one of the known methods for single-layer networks by optimizing a generalized modularity function based on a suitably redefined null model.

Tang, Liu, and Zhang (2012a) proposed an approach for community discovery in dynamic and multimode (also called heterogeneous) networks, that is, networks with different types of nodes that evolve. Interactions between two modes, however, are of only one type. The approach is based on the idea that the interactions between modes can be approximated by the membership of nodes to their latent community. Thus, communities are obtained by minimizing the difference between the original interaction graph and the network reordered according to a block model representing the density of group interactions. To include snapshots of consecutive timestamps, the notion of temporal regularization, similar to the concept of temporal smoothness of Chakrabarti et al. (2006), is introduced. To this end, the objective function to minimize is extended with a term that measures the difference between the clustering obtained at the current timestamp and that returned at the previous one.

MetaFac (metagraph factorization) is an approach proposed by Lin et al. (2011) to extract community structure from dynamic multirelational and multidimensional social data. The concept of multirelational and multidimensional network is analogous to that of multimode network of Tang et al. (2012a), that is, there can exist a number of different types of nodes, interacting through diverse relationships. Networks are represented with a multirelational hypergraph, where each vertex corresponds to a set of nodes of the same type, and an hyperedge connects all the nodes of the same type. Communities of a metagraph are extracted by applying a multirelational factorization method. Data are represented as multiple conjunct data tensors, where each conjunction is realized through a multigraph. The approach is able to deal with time-varying relations by applying an incremental metagraph factorization. Also, *MetaFac* assumes that the community structure to search for should not deviate too much from that obtained at the previous timestamp. Moreover, $MetaFac$ needs the number of communities as input parameter. This number cannot change during the evolution.

The most recent proposals to find groups in multidimensional networks can be found in Tang, Wang, and Liu (2009, 2012b) and Dong et al. (2014). In particular, Tang et al. (2009) observe that there are two main approaches to deal with multidimensional networks. The former is a naive strategy that considers a multidimensional network as one dimensional, by using the average interaction network among nodes. The other approach consists in

optimizing the score function for all the dimensions. Because Tang et al. use modularity, they propose to optimize the total layer modularity. Besides these two approaches, they propose a new method, named principal modularity maximization, that consists of two main steps. First, for each dimension, the so-called structural features, corresponding to the top eigenvectors with positive eigenvalues, are extracted; then these features are combined to obtain latent communities. Tang et al. (2012b) extended their approach by analyzing four different strategies to integrate structural features. One of the main drawbacks of the proposal is that the number of communities must be given as input parameter.

An approach similar to principal modularity maximization has been proposed by Dong et al. (2014). They combine the first k eigenvectors of the graph Laplacian matrices corresponding to layers, to compute a jointly smooth spectrum of couples of layers, by solving an optimization problem that tries to minimize the differences among the eigenvectors of these matrices. Then they apply the K-means clustering method to group nodes in k clusters. The number k must be fixed in advance.

Li, Ng, and Ye (2014), instead of finding a network partitioning, deal with the problem of building a seed-based community for a multidimensional network, that is, given a seed node, neighboring nodes are added to the seed community provided that they are similar.

It is worth to point out that methods proposed in Lin et al. (2011), Tang et al. (2012a, 2009, 2012b), and Dong et al. (2014) all assume that the number of communities is given beforehand. Furthermore, Chi et al. (2009), Lin et al. (2009), and Folino and Pizzuti (2014) are able to deal with dynamic single-layer networks, Lin et al. (2011) and Tang et al. (2012a) work with dynamic multimode networks, and Tang et al. (2009, 2012b) and Dong et al. (2014) consider multidimensional, but not dynamic, networks. Our proposal is different from all the methods described previously. We consider multiplex networks that evolve over time and discover communities from these time-varying multidimensional social data by exploiting the concept of temporal smoothness of evolutionary clustering to smooth both time and multiple types of interactions. Moreover, *DMultiMOGA* automatically determines the number of communities to find and does not need it as input parameter.

3. PROBLEM DEFINITION

In this section, we define the most general concept of multilayer network, as proposed by Kivelä et al. (2014); then the restrictions to represent a temporal network with different types of interactions are introduced.

A single-layer network is a graph $G = (V, E)$, where V is the set of nodes and E is the set of links that connect the objects of V. A multilayer network consists of a network at multiple levels, that is, with multiple types of edges. Thus, the notion of layer, along with nodes and edges, must be considered. Each layer represents a combination of different features of the network, called aspects or facets. More formally, a multilayer network $\mathcal{M}$ is defined as a quadruple:

$$M = (V_M, E_M, V, \mathbf{L})$$

where V is the set of nodes, $\mathbf{L} = \{\mathbf{L_a}\}_{\mathbf{a=1}}^{\mathbf{l}}$ is a sequence of sets of elementary layers L_a, $V_M \subseteq V \times L_1 \times \ldots \times L_l$ contains only the set of combinations of nodes and elementary layers effectively present in a layer, $E_M \subseteq V_M \times V_M$ is a set of couples of possible combinations (Kivelä et al. 2014). Nodes could be connected to any other both inside the same layer and across layers.

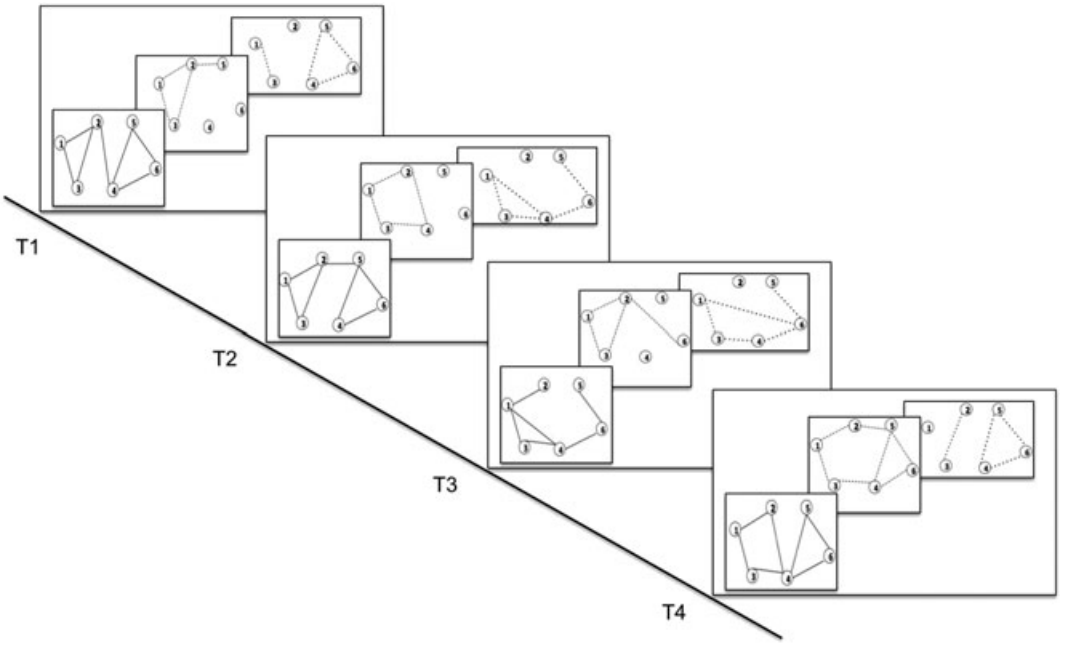


FIGURE 1. Example of dynamic multilayer network with four timestamps and three types of relationships.

In this article, we consider networks constituted by only two aspects. The first aspect $L_1 = \{T^1, \ldots, T^T\}$ represents the temporal information, that is, the time in which a connection between two nodes occurred, while the second one $L_2 = \{D_1, \ldots, D_d\}$ gives the type of interaction among nodes. The set of combinations of a fixed elementary layer $T^t \in L_1$ with all the elementary layers $D_j \in L_2$, $j = 1, \ldots, d$ will be called multiplex (or multidimensional) network at time t and denoted as $\mathcal{T}^t = \{\mathcal{N}_1^t, \mathcal{N}_2^t, \ldots, \mathcal{N}_d^t\}$, where each $\mathcal{N}_i^t$ is the network representing one of the elementary layers of L_2.

A temporal multilayer network is then defined as a sequence $\mathcal{DM} = \{\mathcal{T}^1, \ldots, \mathcal{T}^T\}$ of multiplex networks, where each $\mathcal{T}^t$, $t = 1, \ldots, T$ is a snapshot of the multiplex network at time t, referred as timestamp or timestep.

In the following, we assume that each node represents the same entity in each layer it appears and that cross-edges of the same node at different snapshots are implicit. We do not consider any other type of cross-edges; thus, links between two nodes exist only inside the same elementary layer.

An example of temporal multilayer network can be seen in Figure 1. The first aspect $L_1 = \{T_1, T_2, T_3, T_4\}$ is the temporal information about the network and consists of four timestamps. The second aspect $L_2 = \{D_1, D_2, D_3\}$ represents three different types of connections between nodes. Thus, there are 12 different layers $\{(T_1, D_1), (T_1, D_2), \ldots, (T_4, D_3)\}$. By grouping them with respect to the same timestamp, there are four multiplex networks $\mathcal{T}^t = \{\mathcal{N}_1^t, \mathcal{N}_2^t, \mathcal{N}_3^t\}$, $t = 1, \ldots, 4$. The temporal network $\mathcal{DM} = \{\mathcal{T}^1, \ldots, \mathcal{T}^4\}$ thus consists of four multiplex networks. At each timestamp, different kinds of links between nodes can be observed. For instance, at time $T1$, nodes 4 and 6 are connected with respect to the first and third relationships, that is, in $\mathcal{N}_1^1$ and $\mathcal{N}_3^1$, but not with respect to the second type of connection. In $\mathcal{N}_2^1$, they are isolated nodes. Along the time axis, it is possible to observe how layers evolve. For example, at time $T4$, nodes 4 and 6 are connected in all the $\mathcal{N}_1^4$, $\mathcal{N}_2^4$, and $\mathcal{N}_3^4$ networks.

4. COMMUNITY DISCOVERY WITH TEMPORAL AND DIMENSIONAL SMOOTHING

The discovery of latent community structure in time-varying multidimensional networks poses two main challenging tasks. The first is that the communities obtained at each timestamp t should be consistent with the structure of all the layers present at time t, and, at the same time, they should not differ too much from those found at the previous timestamps. The second one is that, to understand their evolution, communities should be tracked over time.

To deal with the former problem, it is first necessary to define the concept of community structure for a multidimensional network; then the generalization for evolving ones must be considered.

In this article, we propose to extend the evolutionary clustering framework to encompass both time and multiple dimensions. In this extended framework, the problem of finding community structures for time-evolving networks with multiple types of ties is reformulated by adding the concept of dimensional smoothness, relative to a single timestamp, to that of temporal smoothness, at the base of evolutionary clustering (Chakrabarti et al. 2006).

To this end, we first need to define the concept of shared latent community structure for a single timestamp. Tang et al. (2009), for multidimensional networks, formulated this notion as a grouping of nodes such that a quality function is maximized for each layer. We modify this idea by adding the concept of dimensional smoothness, achieved by computing the similarity among the divisions obtained for each layer.

Thus, let $\mathcal{DM} = \{\mathcal{T}^1, \ldots, \mathcal{T}^T\}$ be a temporal multilayer network, where each $\mathcal{T}^t = \{\mathcal{N}_1^t, \mathcal{N}_2^t, \ldots, \mathcal{N}_d^t\}$ is a snapshot of the multiplex network at time t. We enrich the evolutionary clustering framework by introducing the concepts of facet quality $\mathcal{FQ}$ and sharing cost $\mathcal{SQ}$, analogous to those of snapshot quality $\mathcal{SC}$ and temporal cost $\mathcal{TC}$, respectively. Facet quality guarantees that the clustering found for the ith layer under consideration maximizes the adopted quality function as much as possible, while the sharing cost means that the clustering of the current facet agrees as much as possible with the clustering obtained for the other $i-1$ layers. In this extended framework, a shared community structure among the networks $\mathcal{N}_i^t$ of $\mathcal{T}^t$ is obtained by iteratively optimizing both facet quality and sharing cost. The community structure obtained for the last layer d is considered the best sharing community structure among the d layers.

Let $\mathcal{CS}_1^t, \ldots, \mathcal{CS}_d^t$ be the community structures obtained for each elementary layer of a multiplex network $\mathcal{T}^t = \{\mathcal{N}_1^t, \mathcal{N}_2^t, \ldots, \mathcal{N}_d^t\}$, at timestamp t. We say that $\mathcal{CS} = \{C_1, \ldots, C_k\}$ is a shared community structure of $\mathcal{T}^t$ if the two functions are maximized:

$$f_q(\mathcal{CS}, \mathcal{N}_i^t), i = 1, \ldots, d \tag{1}$$

$$f_s(\mathcal{CS}, \mathcal{CS}_i^t), i = 1, \ldots, d \tag{2}$$

where (1) is the quality function computed on the network $\mathcal{N}_i^t$ by using the community structure $\mathcal{CS}$, and (2) is a function that computes the similarity between $\mathcal{CS}$ and the community structure obtained for $\mathcal{N}_i^t$ by maximizing f_q, independently from the other layers. f_q and f_s can be any functions computing the quality of a clustering and the similarity between two clusterings, respectively.

Thus, we search for a community structure $\mathcal{CS}$ that maximizes a fitness function on each elementary layer $\mathcal{N}_i^t$ while taking into account the similarity with the clustering obtained on the other layers. For instance, a shared community structure of the example network of

Figure 1 at time $T1$ could be given by the clusters $C_1 = \{1, 2, 3\}$ and $C_2 = \{4, 5, 6\}$ because this is presenting the highest intersection level with the clusters of the other layers.

This framework is then utilized between couples of consecutive timestamps $t - 1$ and t by resorting to the dynamic evolutionary approach where the temporal cost $\mathcal{TC}$ is guaranteed by considering the similarity between the community structure $\mathcal{CS}^{t-1}$ obtained for the previous timestamp and that found for the first elementary layer $\mathcal{CS}_1^t$ of the current timestamp.

In the next section, a detailed description of the method *DMultiMOGA* is given, along with the functions used to ensure facet quality, sharing cost, and temporal cost. Moreover, a methodology, based on the Hungarian algorithm for optimal assignment problems (Kuhn 1955), to deal with community tracking, is also proposed.

5. THE *DMULTIMOGA* METHOD

DMultiMOGA is a multiobjective genetic algorithm (Coello et al. 2007) that uncovers community structure with temporal and dimensional smoothing by optimizing two competitive objectives both inside a fixed timestamp and when a new timestamp starts. Before giving a detailed description, a brief description of evolutionary multiobjective optimization is reported, along with the genetic representation adopted and the objective functions used by the method.

5.1. Evolutionary Multiobjective Optimization

A multiobjective problem is a problem where a number k of objective functions must be simultaneously optimized. A general multiobjective problem $(\Omega, \mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_h)$ is defined as

$$\min \mathcal{F}_i(\mathcal{S}), \quad i = 1, \ldots, h \qquad \text{subject to } \mathcal{S} \in \Omega$$

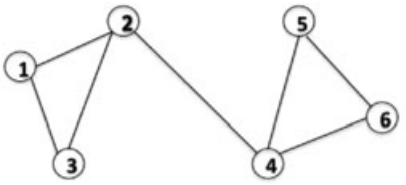
where $\Omega = \{\mathcal{S}_1, \ldots, \mathcal{S}_k\}$ is the set of feasible solutions, and $\mathcal{F} = \{\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_h\}$ is a set of h single criterion functions. Because $\mathcal{F}$ is a vector of competing objectives that must be simultaneously optimized, there is not one unique solution to the problem, but a set of solutions are found through the use of Pareto optimality theory (Coello et al. 2007). Given two solutions $\mathcal{S}_1$ and $\mathcal{S}_2 \in \Omega$, solution $\mathcal{S}_1$ is said to dominate solution $\mathcal{S}_2$, denoted as $\mathcal{S}_1 \prec \mathcal{S}_2$, if and only if

$$\forall i : \mathcal{F}_i(\mathcal{S}_1) \leq \mathcal{F}_i(\mathcal{S}_2) \ \wedge \ \exists\, i \ s.t. \ \mathcal{F}_i(\mathcal{S}_1) < \mathcal{F}_i(\mathcal{S}_2)$$

A Pareto-optimal solution is a nondominated solution for which an improvement in one objective requires a degradation of another. More formally, the set of Pareto-optimal solutions Π is defined as

$$\Pi = \{S \in \Omega : \quad \nexists S' \in \Omega \text{ with } S' \prec S\}$$

This definition means that there are several trade-off solutions, called Pareto-optimal front for the plot of nondominated solutions in the objective space. Multiobjective evolutionary algorithms are stochastic optimization techniques that find such Pareto-optimal solutions. It is worth pointing out that evolutionary methods avoid divergence problems, often present in natural evolution, because of the fitness-based search that selects individuals optimizing an objective function.



(a)

1	2	3	4	5	6
3	4	1	2	6	5

(b)

FIGURE 2. Locus-based representation (b) of an individual for $\mathcal{N}_1^1$ (a) of the toy example of Figure 1 corresponding to the division $\{\{1,3\},\{2,4\},\{5,6\}\}$.

When the problem to solve is that of community detection in temporal multilayer networks, $\Omega = \{\mathcal{S}_1, \ldots, \mathcal{S}_k\}$ is the set of feasible clusterings of a network, while each $\mathcal{F}_i : \Omega \rightarrow \mathcal{R}$ is an objective function determining the feasibility of the obtained clustering.

DMultiMOGA optimizes the two competitive objectives $\mathcal{FQ}$ and $\mathcal{SQ}$ inside a fixed timestamp, and $\mathcal{FQ}$ and $\mathcal{TC}$ when a new timestamp starts. The first two objectives guarantee dimensional smoothing among the layers of a multidimensional network at a current timestamp, while the substitution of $\mathcal{SQ}$ with $\mathcal{TC}$, when a new timestamp begins, ensures temporal smoothness between consecutive timestamps. Thus, the *DMultiMOGA* method, at each timestamp t, tries to maximize the quality of the clustering obtained for the multidimensional network $\mathcal{T}^t = \{\mathcal{N}_1^t, \mathcal{N}_2^t, \ldots, \mathcal{N}_d^t\}$ at time t and to minimize the differences with respect to that obtained at time $t-1$.

5.2. Genetic Representation and Operators

DMultiMOGA uses the locus-based adjacency representation (Park and Song 1989). An individual of the population consists of N genes $g_1, \ldots, g_N$, where N is the number of nodes of the network, assuming values in the range $\{1, \ldots, N\}$. A value j assigned to the ith node means that there is a link between nodes i and j, and that in the clustering solution, i and j will be in the same cluster. Figure 2(b) shows the locus-based representation of an individual when the network $\mathcal{N}_1^1$, at timestamp 1, of the toy example of Figure 1 is considered. In such a case, node 1 is connected with node 3, node 3 with node 1, and so on. The initialization process assigns to each node i one of its neighbors j. The crossover operator adopted is uniform crossover. Given two parents, a random binary vector is created. Uniform crossover selects the genes where the vector is a 0 from the first parent and the genes where the vector is a 1 from the second parent and combines the genes to form the child. The mutation operator, analogously to the initialization process, randomly assigns to each node i one of its neighbors.

5.3. Fitness Functions

We adopted the very popular modularity concept of Girvan and Newman (2004) as facet quality $\mathcal{FQ}$ to optimize. Given a network $\mathcal{N}$ and a partitioning of $\mathcal{N}$ in k communities $S = \{S_1, \ldots, S_k\}$, the modularity $Q(S, \mathcal{N})$ of S is defined as

$$Q(S,\mathcal{N}) = \sum_{s=1}^{k} \left[\frac{l_s}{m} - \left(\frac{d_s}{2m} \right)^2 \right] \tag{3}$$

where the first term of each summand is the fraction of edges inside a community, S_s, while the second one is the expected value of the fraction of edges that would be in the network if edges fell at random without considering the community structure.

As regards the computation of similarity between two clusterings, necessary to guarantee temporal and dimensional smoothness, we used two measures: the well-known entropy-based measure of NMI (Danon et al. 2005) and the Hungarian accuracy (Grappiolo, Togelius, and Yannakakis 2013), based on the Hungarian algorithm for optimal assignment problems (Kuhn 1955). Let $A = \{A_1, \ldots, A_{c_A}\}$ be the true partitioning of a network, consisting of N nodes, in c_A communities, $B = \{B_1, \ldots, B_{c_B}\}$ be the community structure inferred from an algorithm, and C be the $c_A \times c_B$ confusion matrix whose element C_{ij} is the number of nodes of the community $A_i \in A$ that are also in the community $B_j \in B$.

Normalized mutual information. The NMI of A and B is defined as follows:

$$NMI(A, B) = \frac{-2 \sum_{i=1}^{c_A} \sum_{j=1}^{c_B} C_{ij} \log(C_{ij} N / C_{i.} C_{.j})}{\sum_{i=1}^{c_A} C_{i.} \log(C_{i.}/N) + \sum_{j=1}^{c_B} C_{.j} \log(C_{.j}/N)} \tag{4}$$

where c_A (c_B) is the number of groups in the partitioning A (B), $C_{i.}$ ($C_{.j}$) is the sum of the elements of C in row i (column j), and N is the number of nodes. If $A = B$, $NMI(A, B) = 1$. If A and B are completely different, $NMI(A, B) = 0$.

Hungarian accuracy. Let $L = \{l_{v_1}, \ldots, l_{v_N}\}$, $1 \leq l_{v_i} \leq c_A$ and $P = \{p_{v_1}, \ldots, p_{v_N}\}$, $1 \leq p_{v_i} \leq c_B$ be two vectors containing the true and predicted, respectively, community labels of each node. The Hungarian algorithm tries to align L and P, considered as strings, by using the confusion matrix C. It determines the correspondence of the labels from L to P, which maximizes their alignment. P is then relabeled based on this correspondence. The output of the procedure is the number lm of label matches between L and P. The normalized Hungarian accuracy (NHA) is defined as follows:

$$NHA(L, P) = \frac{lm}{N} \tag{5}$$

$NHA(L, P)$ takes values from 0 (total mismatch) to 1 (total match).

Input: Given a dynamic multilayer network $\mathcal{DM} = \{\mathcal{T}^1, \ldots, \mathcal{T}^T\}$, and the number T of timestamps.

Output: A clustering for each multilayer network $\mathcal{T}^i = \{\mathcal{N}_1^i, \ldots, \mathcal{N}_d^i\}$ of $\mathcal{DM}$ and the evolving community tracking.

Method: Perform the following steps:

1 **Let** $\mathcal{G}^1 = \{G_1^1, \ldots, G_d^1\}$ be the set of graphs modeling $\mathcal{T}^1$
2 $Track = \emptyset$
3 $\mathcal{CS}^0 = \emptyset$
4 **Perform** *MultiMOGA*($\mathcal{G}^1$,1, $\mathcal{CS}^0$)
5 **for** $t = 2$ to T
6 **Let** $\mathcal{G}^t = \{G_1^t, \ldots, G_d^t\}$ be the set of graphs modeling $\mathcal{T}^t$, and $\mathcal{CS}^{t-1}$ the clustering obtained at the previous timestamp
7 **Perform** *MultiMOGA*($\mathcal{G}^t$, t, $\mathcal{CS}^{t-1}$)
8 **Let** L^{t-1} and L^t be the cluster labels of nodes at timestamp $t-1$ and t, respectively
9 **Perform** $Track^t = RelabelHungarian(L^t, L^{t-1})$
10 $Track = Track \cup Track^t$
11 **end for** t % t-th timestamp

FIGURE 3. The pseudo-code of the *DMultiMOGA* algorithm.

MultiMOGA($\mathcal{G}^t$, t, $\mathcal{CS}^{t-1}$) **Method:**

Input: A multidimensional network $\mathcal{T}^t = \{\mathcal{N}_1^t, \mathcal{N}_2^t, \ldots, \mathcal{N}_d^t\}$ at timestamp t, modeled as a sequence of graphs $\mathcal{G}^t = \{G_1^t, \ldots, G_d^t\}$, and the clustering $\mathcal{CS}^{t-1}$ obtained at the $t-1$ timestamp

Output: A node cluster labeling that partitions $\mathcal{T}^t$ in the optimal shared community structure

1 **if** (t==1)
2 **Perform** ***ClustGA*** on the G_1^t graph
else
3 **Perform** ***MultiDim***(G_1^t, CS^{t-1}) on the G_1^t graph
4 **for each** node v_i of $\mathcal{G}^t$ not appearing in G_1^t
5 **Perform** ***LabelAssignment***
6 **for** $i = 2$ to d
7 **Create** a population of random individuals whose length equals the number $n_i =\mid V_i \mid$ of nodes of G_i^t;
8 **Perform** a multiobjective GA with objectives
8.1 $\mathcal{FQ} = f_q(\mathcal{CS}_i^t, G_i^t)$
8.2 $\mathcal{SQ} = f_s(\mathcal{CS}_i^t, \mathcal{CS}_{i-1}^t)$
9 **choose** the solution $\mathcal{CS}_i^t = \{C_{i_1}^t, \ldots C_{i_{k_i}}^t\}$ of the Pareto front having the maximum f_q value;
10 **for each** node v_j of $\mathcal{G}^t$ not appearing in G_i^t
11 **Perform** ***LabelAssignment***
12 **end for**

FIGURE 4. The pseudo-code of the *MultiMOGA* algorithm for a single timestamp.

ClustGA **Method:**

Input: The graph G_1^1 modeling the first dimension of a multiplex network

Output: The clustering of G_1^1 having the best fitness value, the node cluster labeling $L^1 = \{l_{v_1}, \ldots, l_{v_N}\}$

1 **create** an initial population of random individuals whose length equals the number N of nodes
2 **while** not maxGen
3 **decode** each individual $I = \{g_1, \ldots, g_N\}$ to obtain the partitioning $C = \{C_1, \ldots, C_k\}$ of the graph G in k connected components.
4 **evaluate** the fitness of the translated individuals
5 **create** a new population by applying the variation operators
6 **end while**

FIGURE 5. The pseudo-code of the *ClustGA* algorithm for the first dimension of multiplex network at timestamp $t = 1$.

5.4. Method Description

The pseudo-code of the *DMultiMOGA* method is reported in Figure 3. *DMultiMOGA* is composed of two main procedures: the multiobjective genetic algorithm *MultiMOGA*, which finds a clustering of a multiplex network of a generic timestamp, and the $RelabelHungarian$ method that, by exploiting the Hungarian algorithm, tries to determine the best correspondence among the communities obtained between two consecutive timestamps and maintains a trace of the evolution of each community along the time. *DMultiMOGA* executes *MultiMOGA* for the first timestamp (step 4). Then, from the second timestamp on, it repeatedly executes *MultiMOGA* and $RelabelHungarian$ (steps 5–11) to obtain the community structure of each multiplex network and the evolution of each community. In the following, a description of these methods is given.

***MultiDim* Method:**

Input: the first graph $\mathcal{G}_1^t$ at timestamp t, and the clustering $\mathcal{CS}^{t-1}$ at timestamp $t-1$

Output: A clustering of $\mathcal{G}_1^t$

1 **Perform** a multiobjective GA on G_1^t with objectives
  1.1 $\mathcal{FQ} = f_q(\mathcal{CS}_1^t, G_1^t)$
  1.2 $\mathcal{SC} = f_s(\mathcal{CS}_1^t, \mathcal{CS}^{t-1})$
2 **choose** the solution $\mathcal{CS}_1^t$ of the Pareto front having the maximum modularity value;

FIGURE 6. The pseudo-code of the $MultiDim$ algorithm for the first dimension of multiplex network at a timestamp $t > 1$.

***LabelAssignment* Method:**

Input: the set of graphs $\mathcal{G} = \{G_1, \ldots, G_d\}$ modeling a multiplex network
the current node cluster labeling $L = \{l_{v_1}, \ldots, l_{v_N}\}$ of nodes of $\mathcal{G}$

Output: Updated L

2 **for each** node v_j with no cluster label
3 **let** $v_{n_1}, \ldots, v_{n_u}$ be the neighbors of v_j in $\mathcal{G}$
  and $l_{v_{n_1}}, \ldots, l_{v_{n_u}}$ be their cluster labels
4 assign to v_j $argmax_l$ $\{l_{v_{n_1}}, \ldots, l_{v_{n_u}})\}$

FIGURE 7. The pseudo-code of the $LabelAssignment$ algorithm.

MultiMOGA (Figure 4) must distinguish between the first timestamp $t = 1$ and a generic timestamp t. In fact, in the former case (step 2), it executes *ClustGA* (Figure 5), a standard GA method on the graph G_1^1 of the first dimension to find a division of G_1^1 that optimizes only the facet quality $\mathcal{FQ}$. In the latter case (step 3), it detects a grouping for the first graph G_1^t of the tth timestamp by running the multiobjective GA method $MultiDim$ (Figure 6), which optimizes the facet quality of G_1^t and the temporal cost $\mathcal{TC}$ between the current clustering and that obtained at the previous timestamp $\mathcal{CS}^{t-1}$. The standard GA method *ClustGA* (described in Figure 5) on the graph G_1^1 representing the first layer optimizes the fitness function to obtain a clustering $\mathcal{CS}_1^1$. *ClustGA*, after a random population has been created (step 1 in Figure 5), runs for a fixed number of generations by applying the variation operators described previously (steps 2–6).

ClustGA and $MultiDim$ return a clustering $\mathcal{CS}^t$ where each node is associated with a class label. However, because any two objects may interact in one dimension but not in another one, some nodes may be isolated in some dimension, and thus, they do not have a cluster label. For these nodes, we then perform a local label propagation (steps 4 and 5 in Figure 4) that, given such a node v_j, considers its neighbors $v_{n_1}, \ldots, v_{n_u}$ in all the dimensions and then assigns to v_j the most recurring class label of its neighbors in $\mathcal{CS}^t$, as explained in the $LabelAssignment$ algorithm (Figure 7). An example of label assignment can be seen in Figure 8. Consider the network $\mathcal{N}_3^1$ of the toy example reported in Figure 1, whose representation is in Figure 8(a). Node 2 has no connections with the other nodes of $\mathcal{N}_3^1$; thus, the method does not assign it to any cluster. However, node 2 has two types of links with nodes 1, 3, 4, and 5, that is, nodes 1, 3, and 4 are its neighbors in $\mathcal{N}_1^1$ and 1, 3,

1	2	3	4	5	6
3	0	1	5	4	5

(a)

1	2	3	4	5	6
3	1	1	5	4	5

(b)

FIGURE 8. Locus-based representation of an individual for N_{31} of the toy example of Figure 1. Individual before label assignment (a). Individual after label assignment (b). The network division of N_{31} is now $\{\{1, 2, 3\}, \{4, 5, 6\}\}$.

RelabelHungarian **Method:**

Input: the node cluster labeling L^t at timestamp t and the node cluster labeling L^{t-1} at timestamp $t-1$

Output: the updated node cluster labeling L^t where each node label has been changed on the base of Hungarian alignment

1 **Let** CM be the $|\mathcal{CS}^{t-1}| \times |\mathcal{CS}^t|$ confusion matrix between L^{t-1} and L^t
2 **Execute** the Hungarian algorithm to obtain the match matrix $\mathcal{MM}$ from CM
3 $Corr = \emptyset$
4 **for** $j = 1, \ldots, |\mathcal{CS}^t|$
5 Let i be the class label at timestamp $t-1$ corresponding to j in $\mathcal{MM}$, i.e. such that $\mathcal{MM}_{ij} = 1$
6 Update L^t by substituting all the class labels j with i
7 $Corr = Corr \cup < i, j >$
8 **end for j**
9 **Renumber** L^t in progressive order

FIGURE 9. The pseudo-code of the RelabelHungarian procedure.

and 5 in $\mathcal{N}_2^1$. Because nodes 1 and 3 are clustered together in $\mathcal{N}_3^1$, the $LabelAssignment$ procedure will assign node 2 to the same cluster of its neighbors. The individual after $LabelAssignment$ is thus that reported in Figure 8(b).

After label assignment, the multiobjective genetic algorithm is iteratively executed for the $d-1$ dimensions (steps 6–12) by optimizing the two objectives $\mathcal{FQ}$ and $\mathcal{SQ}$ (steps 8.1 and 8.2). For each iteration, the clustering having the best f_q value is chosen from the Pareto front as current solution (step 9). Then again, the procedure $LabelAssignment$ (steps 10 and 11) is executed on this solution to assign labels to those nodes having no cluster membership.

The other main procedure of *DMultiMOGA* is $RelabelHungarian$, whose task is to find the best match between communities of two consecutive timestamps. The algorithm (Figure 9) receives the node cluster labelings L^t and L^{t-1}, builds the confusion matrix whose rows correspond to communities in L^{t-1} and columns to those in L^t (step 1), and executes the Hungarian algorithm to obtain a match matrix $\mathcal{MM}$ (step 2), that is, a matrix that if the value at entry (i, j) is 1, it means that community j at time t corresponds to community i at time $t-1$. Then, for each cluster label j in L^t, it finds the corresponding cluster label i determined by the Hungarian algorithm (step 5) and substitutes all the occurrences of j with i in L^t (step 6). The correspondence (i, j) is maintained in a vector $Corr$. Finally, L^t is renumbered to maintain a progressive enumeration of cluster labels.

5.5. Layer Ordering

The *MultiMOGA* method considers the elementary layers of a snapshot network $\mathcal{T}^t = \{\mathcal{N}_1^t, \mathcal{N}_2^t, \ldots, \mathcal{N}_d^t\}$ sequentially; thus, processing first one layer instead of another could

produce different results. Choosing the best ordering is not an easy task. However, we propose a heuristic to sort the layers based on the Jensen–Shannon distance between graphs, adopted in De Domenico et al. (2014) for layer aggregation. In Section 6, we show that this ordering gives good results with respect to a random choice of layers.

Given two layers $\mathcal{N}_i^t$ and $\mathcal{N}_j^t$ of $\mathcal{T}^t$, let ρ and σ be the normalized Laplacian matrices associated, respectively, with the graphs G_i^t and G_j^t modeling these layers, where the normalized Laplacian matrix associated with a graph G of N nodes is defined as $L_G = I - D^{-1}A$, A being the adjacency matrix of G, I the identity matrix, and D the diagonal matrix of the node degrees of G. The Von Neumann entropy h of G can be computed from the eigenvalues $\{\lambda_1, \ldots, \lambda_N\}$ of L_G:

$$h = -\sum_{i=1}^{N} \lambda_i \log_2(\lambda_i) \tag{6}$$

The Jensen–Shannon divergence between ρ and σ is defined as follows:

$$D_{JS}(\rho||\sigma) = h(\mu) - \frac{1}{2}[h(\rho) + h(\sigma)] \tag{7}$$

where $\mu = \frac{1}{2}(\rho + \sigma)$ and h is the Von Neumann entropy. The Jensen–Shannon distance is then

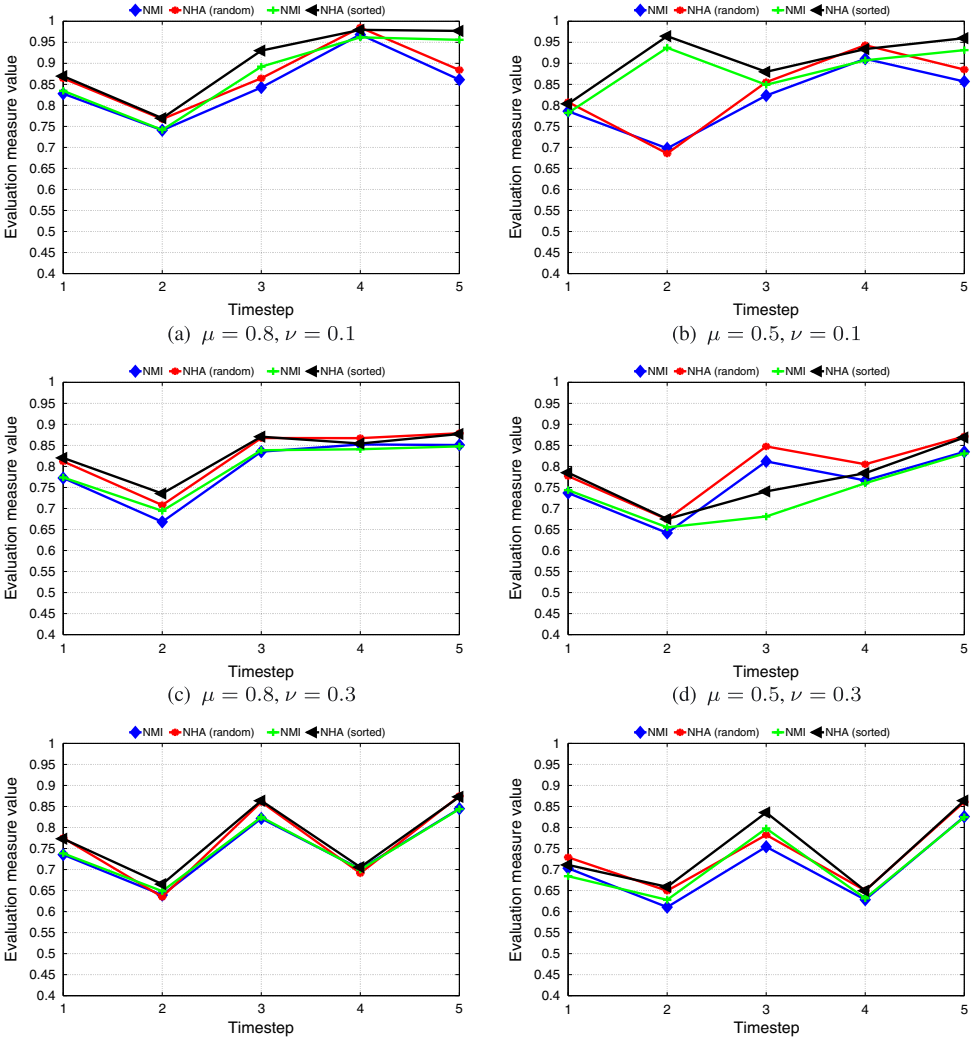
$$d_{JS} = \sqrt{D_{JS}} \tag{8}$$

d_{JS} has been proven to be symmetric and to range from 0 to 1.

The layers $\mathcal{N}_1^t, \ldots, \mathcal{N}_d^t$ of the multidimensional network $\mathcal{T}^t$ for each timestamp t are sorted according to the Jensen–Shannon distance in the following way. For each layer $\mathcal{N}_i^t$, the Jensen–Shannon distance is calculated between this layer and all the other layers of $\mathcal{T}^t$, and the average of these values is computed. The layers are then sorted in ascending order according to this average distance values. Because d_{JS} measures the distance between the normalized Laplacian matrices associated with two graphs G_i^t and G_j^t in terms of information gain/loss, the graph having the lowest average value of d_{JS} can be considered the more informative and representative dimension of a multiplex network because it shares more information with all the other layers.

6. EXPERIMENTAL RESULTS

In this section, we provide an experimental evaluation of the *DMultiMOGA* algorithm on synthetic networks. Then we consider a real-world network generated by De Domenico et al. (2014) and compare *DMultiMOGA* with two state-of-the-art methods. As regards the *DMultiMOGA* parameters, after properly tuning them, we set population size 500, number of generations 150, crossover fraction 0.8, and mutation rate 0.2. The elite reproduction is 10% of the population size, and a roulette selection function is employed. For all the experiments, a t-test at the 5% significance level has been performed to evaluate the statistical significance of the produced results. Being the returned p-values very small, the significance level was very high. The implementation of *DMultiMOGA* has been written in MATLAB 7.14 R2012a (MathWorks, Inc., Natick, MA), using the Genetic Algorithms and Direct Search Toolbox 2.



(a) $\mu = 0.8, \nu = 0.1$ (b) $\mu = 0.5, \nu = 0.1$

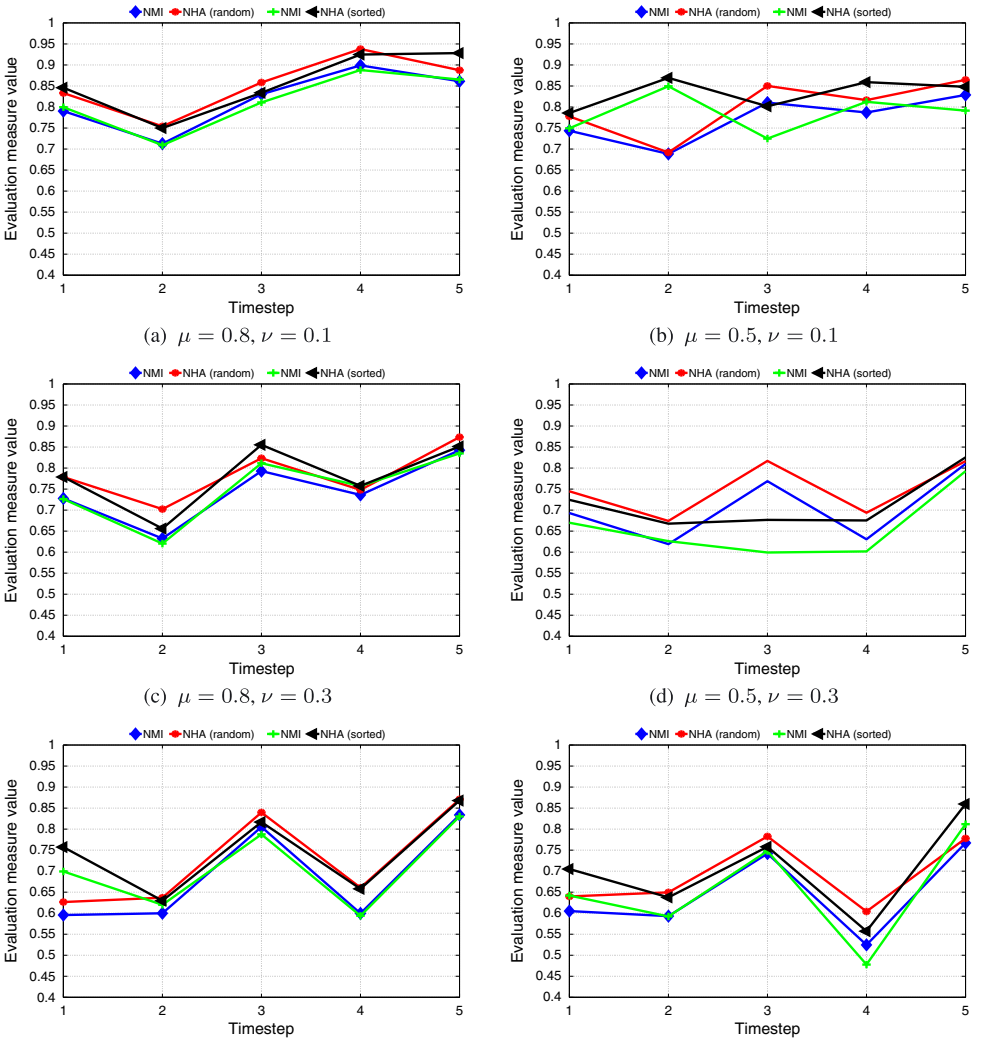
(c) $\mu = 0.8, \nu = 0.3$ (d) $\mu = 0.5, \nu = 0.3$

(e) $\mu = 0.8, \nu = 0.5$ (f) $\mu = 0.5, \nu = 0.5$

FIGURE 10. Normalized mutual information (NMI) and normalized Hungarian accuracy (NHA) obtained by *DMultiMOGA* for five timestamps with $\mu = \{0.5, 0.8\}$ and $\nu = \{0.1, 0.3, 0.5\}$, when NMI is used as second objective function. [Color figure can be viewed at wileyonlinelibrary.com]

6.1. Synthetic Networks

The synthetic networks have been generated by modifying the generator of Tang et al. (2009) for multidimensional networks by adding the temporal component. The synthetic data set is composed of 1,024 objects divided into eight clusters of 128 objects each. The number of dimensions is 4, that is, the objects are involved in four different kinds of relations. The number of timestamps for each dimension is 5. For each timestamp, objects belonging to the same cluster are connected with a random-generated within-group probability μ. Interaction probability changes between groups for each dimension. To control noise, a probability parameter ν is used to connect any two nodes. If the μ value is high and the ν value is low, the network has a clear cluster structure. The μ and ν values are always



(a) $\mu = 0.8, \nu = 0.1$ (b) $\mu = 0.5, \nu = 0.1$

(c) $\mu = 0.8, \nu = 0.3$ (d) $\mu = 0.5, \nu = 0.3$

(e) $\mu = 0.8, \nu = 0.5$ (f) $\mu = 0.5, \nu = 0.5$

FIGURE 11. Normalized mutual information (NMI) and normalized Hungarian accuracy (NHA) obtained by *DMultiMOGA* for five timestamps with $\mu = \{0.5, 0.8\}$ and $\nu = \{0.1, 0.3, 0.5\}$, when NHA is used as second objective function. [Color figure can be viewed at wileyonlinelibrary.com]

the same for each multilayer network in each timestamp. To introduce dynamics, 10% of nodes are moved among communities at each timestamp.

The algorithm has been executed 50 times on 10 randomly generated synthetic networks by considering different combinations of μ and ν values, in particular $\mu = \{0.5, 0.8\}$ and $\nu = \{0.1, 0.3, 0.5\}$. Figures 10 and 11 show the NMI and NHA obtained by *DMultiMOGA* when NMI (Figures 10) and NHA (Figures 11) are used as second objective function. Moreover, we show these values when layers are considered either at random or with respect to the Jensen–Shannon (in parenthesis sorted) order.

The first observation is that values of NMI and NHA are rather similar, although the Hungarian accuracy values are slightly higher than NMI for all the parameter combinations and layer orders.

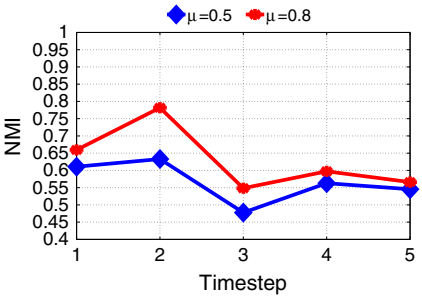


FIGURE 12. Normalized mutual information (NMI) obtained by *DMultiMOGA* for five timestamps with $\mu = \{0.5, 0.8\}$ and $\nu = 0.1$ for the synthetic networks with 10,000 nodes. [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 1. Normalized Mutual Information Values of *DMultiMOGA* on Synthetic Networks for the Events: Birth and Death (B&D), Expansion and Contraction (E&C), and Merging and Splitting (M&S).

Timestamp	B&D	E&C	M&S
T_1	0.9306	0.7035	0.7438
T_2	0.9978	0.8298	0.9476
T_3	0.9720	0.7660	0.7766
T_4	0.9650	0.8059	0.9258
T_5	0.9934	0.8269	0.8939

When the objectives $\mathcal{SQ}$ and $\mathcal{TC}$ are computed as NMI (Figure 10), we can observe that the Jensen–Shannon layer order gives, in general, better values of both NMI and NHA with respect to a random choice of the dimensions. This is not true any more when as second objective NHA is used (Figure 11). Moreover, in such a case, the values of NMI and NHA are lower with respect to those obtained with NMI as $\mathcal{SQ}$ and $\mathcal{TC}$.

However, the figures point out that our approach achieves good accuracy in finding the true community structure, with NMI and NHA values above 0.95 in some timestamps. From a deeper analysis on the topology of the obtained communities, we found that when the noise level ν increases, the method has the tendency to split some communities. In particular, it can split one or at most two communities for $\nu = 0.1, 0.3$, while it can split also three communities for $\nu = 0.5$, for example, at timesteps 2 and 4 when $\mu = 0.5$ and $\nu = 0.5$.

To test the method on large data sets, we generated synthetic networks with two layers of 10,000 nodes each for five timesteps, $\mu = 0.5, 0.8$, $\nu = 0.1$, and executed *DMultiMOGA* 50 times. The average NMI values are depicted in Figure 12. In this case, we can observe that these values are lower than those of the 1,024 nodes networks, because of the behavior of the method to find many small communities instead of the big ground truth clusters consisting of 1,250 nodes each.

The synthetic data set described previously maintains constant the number of communities for all the timestamps. However, dynamic networks are characterized by some types of events that better describe their evolution, such as birth and death, expansion and contraction, and merging and splitting (Asur, Parthasarathy, and Ucar 2009; Greene, Doyle, and Cunningham 2010). To this end, we modified the synthetic data set to include these types of events as follows:

- Birth and death: We choose four communities at the first timestamp, and, from the second timestamp on, a new community appears, and an existing one is deleted.
- Expansion and contraction: Two out of eight communities are randomly selected and expanded or contracted by 25% of their size.
- Merging and splitting: At each timestamp, a community is split, and two communities are merged.

Table 1 shows the average NMI values for each of the described events, along the five timestamps. In particular, *DMultiMOGA* behaves very well in case of birth and death, because it finds communities that correspond almost perfectly to the ground truth division, by just assigning less than 1% of nodes to a different cluster. Regarding the other two network topologies, expansion and contraction, and merging and splitting, in two out of five timesteps, *DMultiMOGA* splits one or two communities. In any case, the table points out that it is able to successfully face these types of events that a network can incur during its evolution.

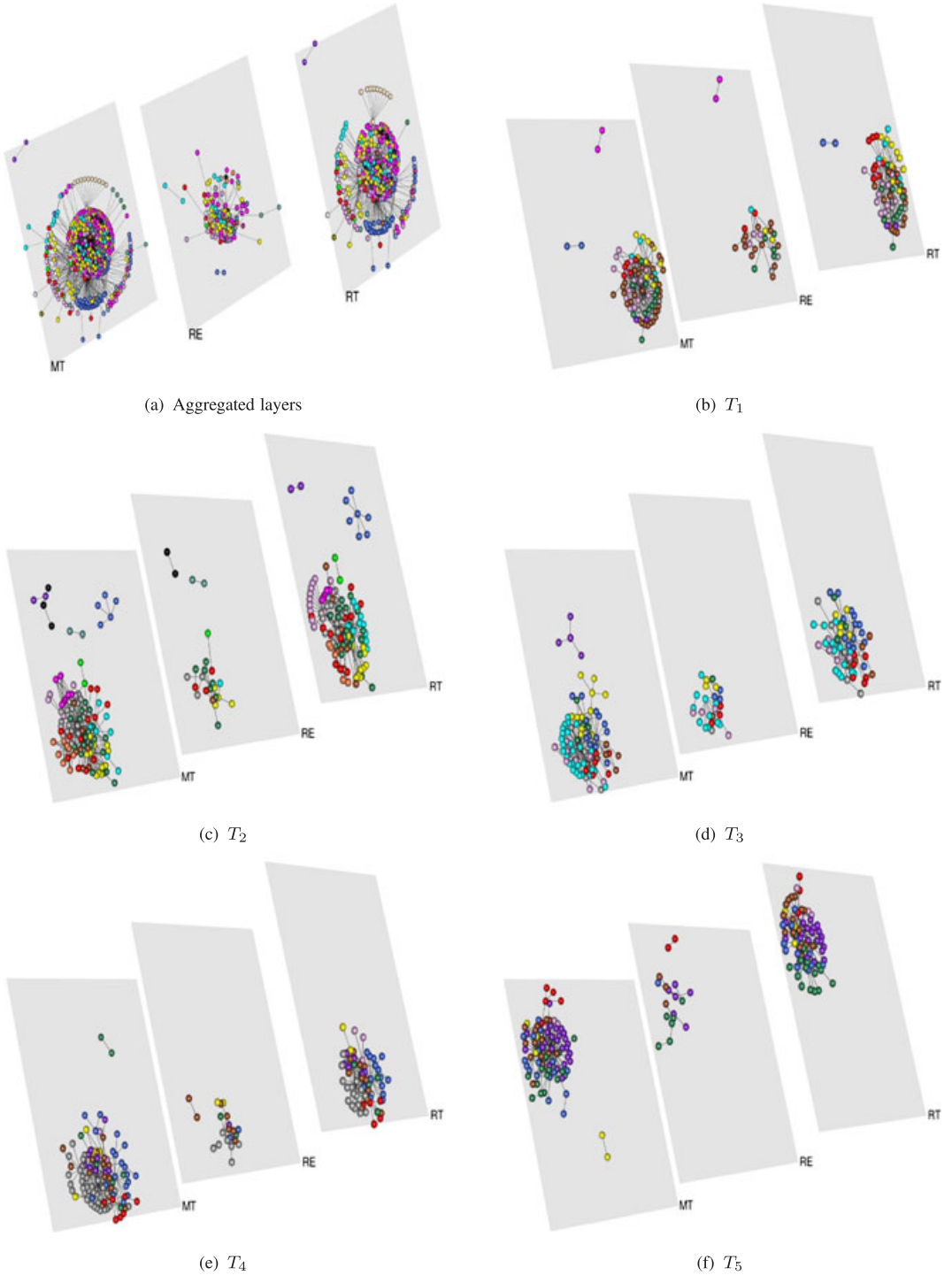
6.2. Real-World Data Set

The real-world temporal multidimensional data set, generated by De Domenico (2014), describes the interactions among the participants to the European Conference on Complex Systems (ECCS) 2013, an international conference covering general aspects of complex systems. The conference lasted 5 days, from September 16 to 20, with 2 days, September 18 and 19, dedicated to satellite meetings. Every day had two sections, the first one in the morning and the second one in the afternoon. The data set describes the daily interactions over Twitter between 412 people attending the conference. Three different kinds of relations are considered: RT (message retweet), RE (reply), and MT (mention). The data set is composed of 5,943 rows, each containing a pair of node identifiers, a timestamp describing the time in minutes when the relation took place, and the type of relation.

Data set rows have been grouped based on timestamp and type of relationship. This grouping generated a total of 2,617 timestamps. Because of the too high number of

TABLE 2. Number of Nodes and Edges for Each Timestamp $\{T_1, ..., T_{11}\}$ and Dimension {MT, RE, RT} for the European Conference on Complex Systems Temporal Multiplex Network.

	MT		RE		RT	
Timestamp	Nodes	Edges	Nodes	Edges	Nodes	Edges
T_1	84	184	30	27	69	100
T_2	109	248	24	21	105	138
T_3	83	190	27	21	58	91
T_4	96	178	21	16	62	105
T_5	83	209	18	14	72	129
T_6	91	199	23	24	56	103
T_7	70	174	21	14	62	107
T_8	90	208	16	12	71	111
T_9	103	215	17	14	65	95
T_{10}	88	248	26	24	67	109
T_{11}	112	223	23	15	57	84

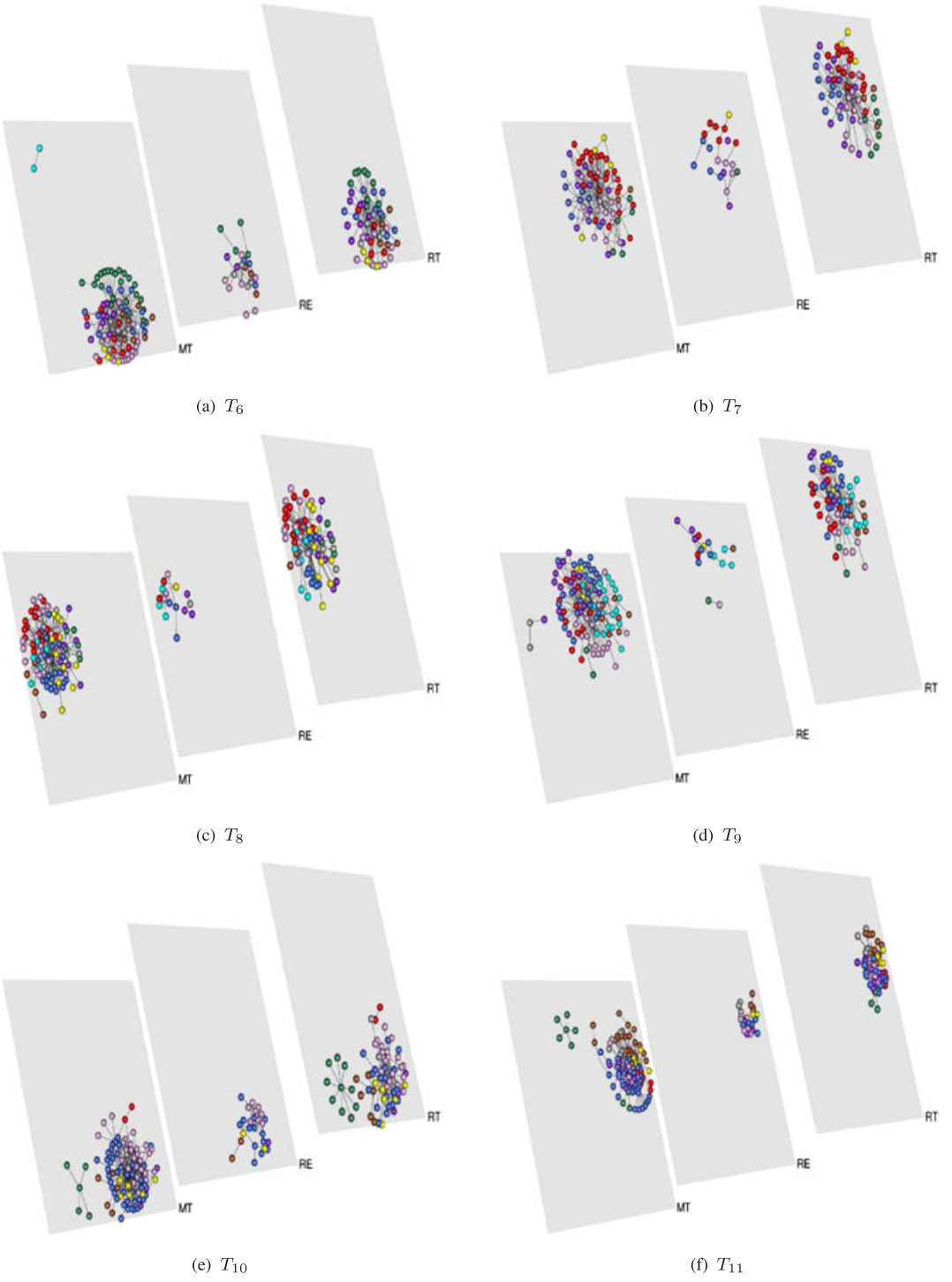


(a) Aggregated layers (b) T_1

(c) T_2 (d) T_3

(e) T_4 (f) T_5

FIGURE 13. Clustering obtained from *DMultiMOGA* for the (a) aggregated European Conference on Complex Systems network and for (b–f) timestamps 1–5. [Color figure can be viewed at wileyonlinelibrary.com]



(a) T_6 (b) T_7

(c) T_8 (d) T_9

(e) T_{10} (f) T_{11}

FIGURE 14. Clustering obtained from *DMultiMOGA* for the European Conference on Complex Systems network at timestamps 6–11. [Color figure can be viewed at wileyonlinelibrary.com]

timestamps, each often containing only very few nodes and interactions, we merged them in temporal intervals of 4 h, representing the morning and afternoon sessions of every day. In such a way, we obtained 11 time intervals. The number of nodes and edges for each timestamp and type of interactions is reported in Table 2. Note that edges are directed. From the table, we can observe that the networks are rather sparse and that nodes are not present in each timestamp.

The result of running *DMultiMOGA* on this real-life directed temporal multiplex network is displayed in Figures 13 and 14, where colors distinguish the different communities found by the method.[1] To understand how communities form and evolve along time, in Figure 15, we report, for each timestamp, the clusters obtained by *DMultiMOGA* with the number of nodes (in parenthesis) that each contains. An arrow from a cluster i at time T_t to a cluster j at time T_{t+1} means that community i evolved into community j, as determined by the algorithm $RelabelHungarian$, described in Figure 8. It is worth noting that some communities disappear, while new ones form. For instance, from timestamp T_1 to T_2, communities 3, 7, and 10, all composed of only two nodes, die. However, one node of community 3 moves to community 1 of T_2, while the other five nodes disappear. At time T_2, although some nodes do not continue to interact, new nodes become active, and new

[1] The figures have been generated with the $muxViz$ software, downloadable from http://deim.urv.cat/~manlio.dedomenico/muxviz.php

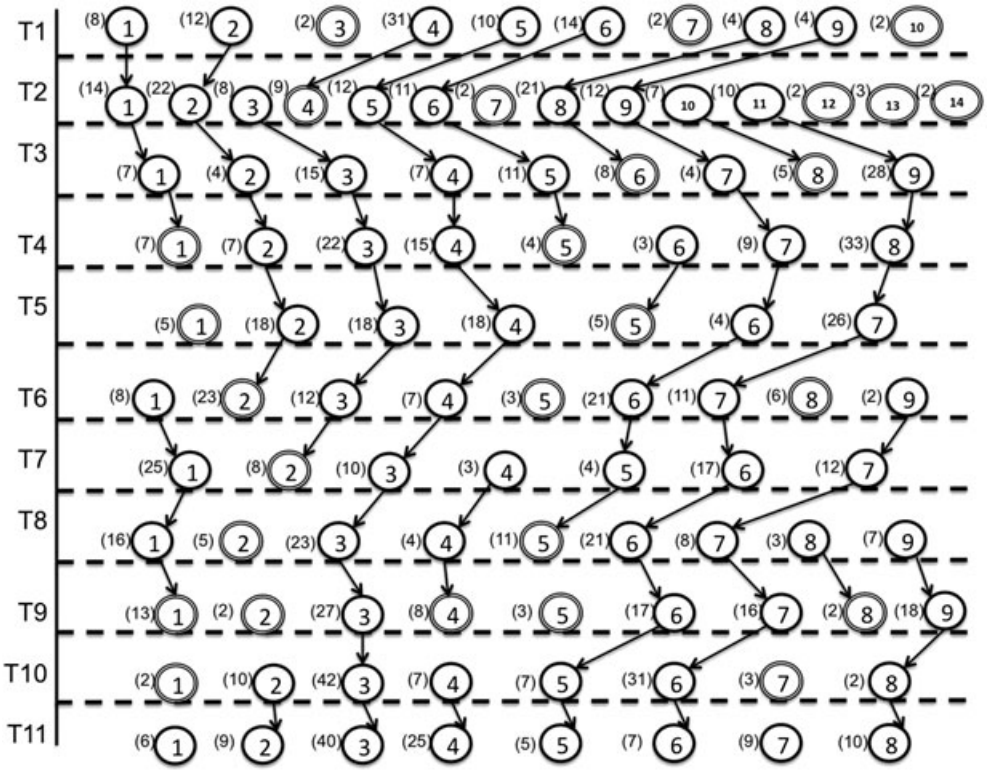


FIGURE 15. Tracking of the clusters obtained from *DMultiMOGA* on the European Conference on Complex Systems network for the 11 timestamps. The cluster size is reported in parenthesis.

communities rise. In fact, the number of nodes increases from 89 to 135 and the number of communities from 10 to 14.

The analysis of the network has pointed out the high relationship dynamism of participants to the conference. Many of them interact in a time period and then are not interested any more to communicate with the same people in the next timestamp. For example, from time T_5 to T_6, although the total number of nodes is almost the same, 94 and 93, respectively, 47 nodes out of 94 in T_5 do not appear any more in T_6, while the other nodes reaggregate and distribute in the new born communities. For instance, class 7 in T_5 looses 15 nodes in T_6, and its remaining 11 nodes rearrange among the new communities 1–3 and 6–8. Thus, even if community 7 in T_5 evolves to community 7 in T_6, its structure is completely changed. This behavior is more evident when computing, for each node, the number of timestamps that it is active, as reported in Table 3. From the table, we can observe that 235 out of 412 exchange messages in two (not necessarily consecutive) timestamps, while only eight people (i.e., nodes numbered 21, 160, 208, 212, 229, 234, 290, and 307) are active for all the 11 timestamps. These nodes can play an important role in information diffusion over the network.

The ground truth division of the ECCS network is not known; thus, we adopted the suggestion of Lin et al. (2009) to use the clustering obtained on the aggregated network as reference and then compute the NMI between the communities obtained at each timestamp with these reference communities. To this end, we aggregated the ECCS network for all the timestamps and layers and then computed the community structure by running the

TABLE 3. Number of Nodes That Are Active for a Number of Timestamps, Eventually Not Consecutive, in the European Conference on Complex Systems Temporal Multiplex Network.

Number of timestamps	1	2	3	4	5	6	7	8	9	10	11
Number of active nodes	412	235	131	86	69	51	41	32	21	12	8

TABLE 4. Confusion Matrix of the First Timestamp on the European Conference on Complex Systems Temporal Multiplex Network.

	T_1									
	C_1^1	C_2^1	C_3^1	C_4^1	C_5^1	C_6^1	C_7^1	C_8^1	C_9^1	C_{10}^1
C_1	2	9	0	8	10	7	0	1	2	2
C_2	0	2	0	7	0	3	2	0	0	0
C_3	5	0	2	4	0	0	0	0	0	0
C_4	1	0	0	7	0	0	0	0	0	0
C_5	0	0	0	1	0	1	0	0	0	0
C_6	0	0	0	0	0	3	0	0	0	0
C_7	0	0	0	0	0	0	0	0	2	0
C_8	0	1	0	1	0	0	0	0	0	0
C_9	0	0	0	0	0	0	0	3	0	0
C_{10}	0	0	0	2	0	0	0	0	0	0
C_{11}	0	0	0	1	0	0	0	0	0	0
C_{12}	0	0	0	0	0	0	0	0	0	0

Blondel et al. (2008) algorithm, which is one of the most efficient and successful methods for community discovery in monoplex networks that optimizes modularity. This algorithm, on the aggregated network of 412 nodes, returned 12 communities each containing 104, 90, 53, 30, 28, 25, 24, 18, 13, 13, 9, and 2 nodes, respectively. Tables 4–14 show the confusion matrices obtained by using these communities as reference ground truth and the communities obtained by *DMultiMOGA* for each timestamp. It is interesting to note that, generally, the nodes of the reference communities are distributed among the predicted communities. However, often, *DMultiMOGA* finds subcommunities entirely contained in the ground truth

TABLE 5. Confusion Matrix of the Second Timestamp on the European Conference on Complex Systems Temporal Multiplex Network.

	T_2													
	C_1^2	C_2^2	C_3^2	C_4^2	C_5^2	C_6^2	C_7^2	C_8^2	C_9^2	C_{10}^2	C_{11}^2	C_{12}^2	C_{13}^2	C_{14}^2
C_1	3	11	0	6	3	1	0	5	9	0	0	0	2	2
C_2	0	0	0	0	0	0	0	9	2	7	10	0	2	1
C_3	2	0	8	0	0	0	0	4	0	0	0	0	0	0
C_4	6	1	0	3	0	0	0	0	0	0	0	2	0	0
C_5	0	2	0	0	1	8	0	0	0	0	0	0	0	0
C_6	3	1	0	0	0	1	0	0	1	0	0	0	0	0
C_7	0	0	0	0	0	1	0	1	0	0	0	0	0	0
C_8	0	5	0	0	0	0	0	1	0	0	0	0	0	0
C_9	0	1	0	0	0	0	0	0	0	0	0	0	0	0
C_{10}	0	1	0	0	0	0	0	1	0	0	0	0	0	0
C_{11}	0	0	0	0	8	0	0	0	0	0	0	0	0	0
C_{12}	0	0	0	0	0	0	2	0	0	0	0	0	0	0

TABLE 6. Confusion Matrix of the Third Timestamp on the European Conference on Complex Systems Temporal Multiplex Network.

	T_3								
	C_1^3	C_2^3	C_3^3	C_4^3	C_5^3	C_6^3	C_7^3	C_8^3	C_9^3
C_1	7	0	3	2	0	1	4	1	7
C_2	0	0	0	0	0	0	0	1	15
C_3	0	0	10	1	0	0	0	0	0
C_4	0	0	1	4	2	0	0	3	2
C_5	0	0	0	0	4	1	0	0	0
C_6	0	0	0	0	1	0	0	0	0
C_7	0	0	0	0	0	0	0	0	1
C_8	0	4	0	0	4	0	0	0	2
C_9	0	0	0	0	0	5	0	0	0
C_{10}	0	0	1	0	0	1	0	0	1
C_{11}	0	0	0	0	0	0	0	0	0
C_{12}	0	0	0	0	0	0	0	0	0

TABLE 7. Confusion Matrix of the Fourth Timestamp on the European Conference on Complex Systems Temporal Multiplex Network.

	T_4							
	C_1^4	C_2^4	C_3^4	C_4^4	C_5^4	C_6^4	C_7^4	C_8^4
C_1	6	2	1	9	3	1	6	5
C_2	0	0	0	0	0	0	3	17
C_3	0	0	11	1	0	0	0	0
C_4	1	0	5	2	1	0	0	2
C_5	0	1	0	0	0	0	0	2
C_6	0	1	0	0	0	0	0	1
C_7	0	0	1	1	0	2	0	1
C_8	0	3	0	1	0	0	0	1
C_9	0	0	0	0	0	0	0	1
C_{10}	0	0	1	0	0	0	0	3
C_{11}	0	0	3	1	0	0	0	0
C_{12}	0	0	0	0	0	0	0	0

TABLE 8. Confusion Matrix of the Fifth Timestamp on the European Conference on Complex Systems Temporal Multiplex Network.

	T_5						
	C_1^5	C_2^5	C_3^5	C_4^5	C_5^5	C_6^5	C_7^5
C_1	0	4	3	9	2	0	15
C_2	0	2	4	1	0	4	6
C_3	0	0	4	1	0	0	0
C_4	0	2	5	1	0	0	2
C_5	0	2	0	4	0	0	1
C_6	5	0	0	2	1	0	0
C_7	0	7	0	0	2	0	0
C_8	0	1	1	0	0	0	1
C_9	0	0	1	0	0	0	0
C_{10}	0	0	0	0	0	0	1
C_{11}	0	0	0	0	0	0	0
C_{12}	0	0	0	0	0	0	0

ones. For instance, at time T_1, it detects two subcommunities composed of 10 and two nodes (C_5^1 and C_{10}^1) of the 104 node C_1 community, the two node subcommunities C_3^1 and C_7^1 of C_3 and C_2, respectively. At time T_2, *DMultiMOGA* obtains the subcommunities C_3^2, C_7^2, C_{10}^2, C_{11}^2, and C_{12}^2 of C_3, C_{12}, C_2, and C_4, respectively. More notably, community C_{12} (corresponding to C_7^2) of two nodes is found by *DMultiMOGA* only at time T_2, meaning that these two researchers became active at timestamp T_2 and then did not send messages any more. Moreover, C_{10}^2 and C_{11}^2 are both subgroups of C_2, thus providing a more detailed

TABLE 9. Confusion Matrix of the Sixth Timestamp on the European Conference on Complex Systems Temporal Multiplex Network.

	T_6								
	C_1^6	C_2^6	C_3^6	C_4^6	C_5^6	C_6^6	C_7^6	C_8^6	C_9^6
C_1	3	2	8	2	0	8	7	3	2
C_2	5	1	0	0	3	6	0	2	0
C_3	0	9	0	0	0	0	1	0	0
C_4	0	2	0	0	0	1	0	0	0
C_5	0	0	4	4	0	1	0	0	0
C_6	0	1	0	1	0	2	0	0	0
C_7	0	6	0	0	0	1	0	0	0
C_8	0	1	0	0	0	0	1	1	0
C_9	0	0	0	0	0	1	0	0	0
C_{10}	0	0	0	0	0	0	1	0	0
C_{11}	0	1	0	0	0	1	1	0	0
C_{12}	0	0	0	0	0	0	0	0	0

TABLE 10. Confusion Matrix of the Seventh Timestamp on the European Conference on Complex Systems Temporal Multiplex Network.

	T_7						
	C_1^7	C_2^7	C_3^7	C_4^7	C_5^7	C_6^7	C_7^7
C_1	17	0	0	0	0	7	5
C_2	2	0	0	0	3	0	0
C_3	0	0	0	3	0	2	0
C_4	4	0	0	0	0	1	0
C_5	0	1	10	0	0	0	2
C_6	0	0	0	0	1	1	0
C_7	1	6	0	0	0	2	0
C_8	1	0	0	0	0	0	0
C_9	0	1	0	0	0	0	5
C_{10}	0	0	0	0	0	4	0
C_{11}	0	0	0	0	0	0	0
C_{12}	0	0	0	0	0	0	0

information of C_2 at this timestamp. Analogous situations can be observed for the other timestamps, confirming the advantages of temporal and multiplex networks in analyzing the original complex system, avoiding to lose important information regarding it, that could happen when an aggregated network representation is used.

Finally, because at the moment there exist algorithms that are able to deal either with multiplex or with dynamical networks, but not both, we consider the multiplex approach

TABLE 11. Confusion Matrix of the Eighth Timestamp on the European Conference on Complex Systems Temporal Multiplex Network.

	T_8								
	C_1^8	C_2^8	C_3^8	C_4^8	C_5^8	C_6^8	C_7^8	C_8^8	C_9^8
C_1	5	1	9	1	5	10	1	0	1
C_2	0	0	3	0	1	0	0	0	0
C_3	0	4	2	1	0	1	0	0	0
C_4	0	0	0	0	5	3	0	0	0
C_5	0	0	7	0	0	1	2	0	0
C_6	0	0	0	0	0	0	1	0	0
C_7	6	0	0	0	0	0	0	0	2
C_8	4	0	1	1	0	1	0	0	3
C_9	0	0	0	0	0	0	4	3	1
C_{10}	0	0	1	1	0	3	0	0	0
C_{11}	1	0	0	0	0	2	0	0	0
C_{12}	0	0	0	0	0	0	0	0	0

TABLE 12. Confusion Matrix of the Ninth Timestamp on the European Conference on Complex Systems Temporal Multiplex Network.

	T_9								
	C_1^9	C_2^9	C_3^9	C_4^9	C_5^9	C_6^9	C_7^9	C_8^9	C_9^9
C_1	7	0	6	5	1	2	2	0	12
C_2	0	0	8	0	1	0	0	0	0
C_3	2	2	2	1	0	4	1	0	0
C_4	1	0	0	0	1	1	0	0	1
C_5	0	0	2	0	0	1	1	0	0
C_6	0	0	1	0	0	3	11	0	0
C_7	3	0	5	0	0	0	0	0	2
C_8	0	0	1	0	0	4	0	0	2
C_9	0	0	1	0	0	0	1	2	1
C_{10}	0	0	0	1	0	1	0	0	0
C_{11}	0	0	1	1	0	1	0	0	0
C_{12}	0	0	0	0	0	0	0	0	0

of Mucha et al. (2010)[2] and the dynamic framework of Lin et al. (2009) and compare *DMultiMOGA* with the community structure that these two methods find on each timestamp. Regarding the former method, we provide as input the multidimensional networks of each timestamp, one at a time, and thus execute the method 11 times. For $FacetNet$, however, because it can analyze single-dimensional networks, at each timestamp, we provide the network where the three layers are merged. Table 15 shows, for each timestamp, the

[2] We used the algorithm implemented in the $muxViz$ software.

TABLE 13. Confusion Matrix of the 10th Timestamp on the European Conference on Complex Systems Temporal Multiplex Network.

	T_{10}							
	C_1^{10}	C_2^{10}	C_3^{10}	C_4^{10}	C_5^{10}	C_6^{10}	C_7^{10}	C_8^{10}
C_1	0	0	9	4	4	20	1	1
C_2	0	10	11	1	0	0	0	0
C_3	0	0	2	0	0	4	0	0
C_4	0	0	6	0	0	0	0	1
C_5	0	0	1	1	0	0	0	0
C_6	0	0	5	0	0	6	0	0
C_7	2	0	2	0	0	0	0	0
C_8	0	0	0	0	3	0	0	0
C_9	0	0	2	1	0	0	0	0
C_{10}	0	0	4	0	0	0	2	0
C_{11}	0	0	0	0	0	1	0	0
C_{12}	0	0	0	0	0	0	0	0

TABLE 14. Confusion Matrix of the 11th Timestamp on the European Conference on Complex Systems Temporal Multiplex Network.

	T_{11}							
	C_1^{11}	C_2^{11}	C_3^{11}	C_4^{11}	C_5^{11}	C_6^{11}	C_7^{11}	C_8^{11}
C_1	0	0	13	12	0	6	4	3
C_2	6	6	19	0	0	1	0	0
C_3	0	1	1	2	0	0	0	0
C_4	0	0	1	4	1	0	0	1
C_5	0	0	2	3	0	0	0	0
C_6	0	0	0	0	0	0	4	1
C_7	0	0	3	0	0	0	0	0
C_8	0	0	0	0	4	0	1	0
C_9	0	2	1	4	0	0	0	0
C_{10}	0	0	0	0	0	0	0	5
C_{11}	0	0	0	0	0	0	0	0
C_{12}	0	0	0	0	0	0	0	0

number of active nodes at that time and, for each method, the number of clusters that the method obtains and the NMI with respect to the reference ground truth. The table clearly points out the superiority of *DMultiMOGA* with respect to the other two approaches. It is worth to observe that both $FacetNet$ and $Mucha$ methods seem not to be able to properly group nodes when working only with subparts of the overall network. In fact, at each timestamp, the number of active nodes is between 20% and 30% of the total number of nodes. *DMultiMOGA*, instead, is capable to find a more granular community structure, obtaining a finer and more detailed organization of the people constituting the network, consistent with the division of the aggregated network. This experiment points out the capability

TABLE 15. Comparison of *DMultiMOGA* with *FacetNet* and *Mucha* Methods on the ECCS Temporal Multiplex Network.

		FacetNet		*DMultiMOGA*		*Mucha*	
Time	Nodes	nClust	NMI	nClust	NMI	nClust	NMI
T_1	89	12	0.1288	10	0.6617	8	0.1175
T_2	135	12	0.2141	14	0.6796	12	0.2036
T_3	89	12	0.2220	9	0.7293	9	0.1854
T_4	100	12	0.2278	8	0.6481	11	0.1601
T_5	94	12	0.2346	7	0.6345	10	0.1446
T_6	93	12	0.2349	9	0.6371	8	0.1132
T_7	79	12	0.2706	7	0.7648	8	0.1998
T_8	98	12	0.2787	9	0.6494	8	0.1705
T_9	106	12	0.2631	9	0.6097	8	0.1609
T_{10}	104	12	0.2579	8	0.6543	7	0.1689
T_{11}	111	12	0.2555	8	0.6581	10	0.1752

ECCS, European Conference on Complex Systems; NMI, normalized mutual information.

of *DMultiMOGA* to successfully deal with the problem of community detection in dynamic multidimensional networks, by providing an automatic way of finding community structure in multiplex networks, and to track their evolution along time. Moreover, it allows to study the behavior of each node as time passes.

7. CONCLUSIONS

The article presented a multiobjective method to uncover community structure in temporal multiplex networks, relying on the new concepts of facet quality and dimensional sharing, and to track their evolution, by exploiting the Hungarian method. The algorithm applies also a local label propagation strategy to properly assign nodes, not appearing in all layers, to clusters. A heuristic that exploits the concept of distance between graphs is also introduced to select the ordering under which networks should be examined at a particular timestamp. Experimental results showed the ability of the approach in obtaining meaningful node clustering, also when compared with state-of-the-art methods. The proposed framework is an advancement in the network analysis field because it allows to study the topology of complex systems consisting of objects connected by multiple types of interactions, which can change as time goes by. Traditional approaches, treating different links as either undistinguishable or with a weight quantifying the strength of interactions, have been recognized to provide descriptions unable to capture the details of real-world phenomena, and, in some cases, these descriptions could also be incorrect. Temporal multiplex networks, instead, yield a richer and more natural representation of many real-world systems. The advantages of employing such networks have been confirmed on the ECCS network, for which a finer and more detailed group organization, with respect to the community structure of the aggregated network, has been obtained by *DMultiMOGA*. The generalization of the evolutionary clustering approach to multiplex and time-varying networks, by exploiting multiobjective optimization, can be considered a step forward in the research field of network science.

ACKNOWLEDGMENT

The authors thank Dr. Manlio De Domenico for providing the ECCS network.

REFERENCES

ASUR, S., S. PARTHASARATHY, and D. UCAR. 2009. An event-based framework for characterizing the evolutionary behavior of interaction graphs. ACM Transactions on Knowledge Discovery from Data, **3**(4): Paper 16.

BATTISTON, F., V. NICOSIA, and V. LATORA. 2013. Metrics for the analysis of multiplex networks. *In* arXiv:1308.3182v2.

BLONDEL, V. D., J.-L. GUILLAUME, R. LAMBIOTTE, and E. LEFEVRE. 2008. Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment, **P10008**.

CHAKRABARTI, D., R. KUMAR, and A. TOMKINS. 2006. Evolutionary clustering. *In* Proceedings of the 12th ACM International Conference on Knowledge Discovery and Data Mining (KDD'06), August 20–23, Philadelphia, pp. 554–560.

CHI, Y., X. SONG, D. ZHOU, K. HINO, and B. L. TSENG. 2009. On evolutionary spectral clustering. ACM Transactions on Knowledge Discovery from Data, **3**(4, Article 17).

COELLO, C., A. COELLO, G. B. LAMONT, and D. A. VAN VELDHUIZEN. 2007. Evolutionary Alg. for Solving Multi-Objective Problems. Springer: New York.

DANON, L., A. DÍAZ-GUILERA, J. DUCH, and A. ARENAS. 2005. Comparing community structure identification. Journal of Statistical Mechanics, **P09008**.

DEB, K. 2001. Multi-objective Optimization using Evolutionary Algorithms. Wiley: Chichester, England.

DE DOMENICO, M. 2014. Personal communication. *In* Departament d'Enginyeria Informática i Matemátiques, Universitat Rovira I Virgili, Tarragona, Spain.

DE DOMENICO, M., V. NICOSIA, A. ARENAS, and V. LATORA. 2014. Layer aggregation and reducibility of multilayer interconnected networks. *In* arXiv:1405.0425v1.

DE DOMENICO, M., M. A. PORTER, and A. ARENAS. 2014. Multilayer analysis and visualization of networks. *In* arXiv:1405.0843v1.

DE DOMENICO, M., A. SOLÉ-RIBALTA, E. COZZO, M. KIVELÄ, Y. MORENO, M. A. PORTER, S. GÓMEZ, and A. ARENAS. 2013. Mathematical formulation of multilayer networks. *In* Physical Review X 3. p. 041022.

DONG, X., P. FROSSARD, P. VANDERGHEYNST, and N. NEFEDOV. 2014. Clustering on multi-layer graphs via subspace analysis on grassmann manifolds. IEEE Transactions on Signal Processing, **62**(4): 905–918.

FOLINO, F., and C. PIZZUTI. 2014. An evolutionary multiobjective approach for community discovery in dynamic networks. IEEE Transactions on Knowledge and Data Engineering, **26**(8): 1838–1852.

GRAPPIOLO, C., J. TOGELIUS, and G. N. YANNAKAKIS. 2013. Artificial evolution for the detection of group identities in complex artificial societies. *In* IEEE Symposium on Artificial Life, ALife 2013, Singapore, pp. 126–133.

GREENE, D., D. DOYLE, and P. CUNNINGHAM. 2010. Tracking the evolution of communities in dynamic social networks. *In* International Conference on Advances in Social network Analysis and Mining (ASONAM'10), August 9–11, Odense, Denmark, pp. 176–183.

KIM, M., and J. HAN. 2009. A particle-and-density based evolutionary clustering method for dynamic networks. *In* Proceedings of the VLDB Endowment, Vol. 2, No. 1, August 24–28, Lyon, France, pp. 622–633.

KIVELÄ, M., A. ARENAS, M. BARTHELEMY, J. P. GLEESON, Y. MORENO, and M. A. PORTER. 2014. Multilayer networks. *In* arXiv:1309.7233v3.

KUHN, H. W. 1955. The Hungarian method for the assignment problem. Naval Research Logistics Quarterly, **2**(1-2): 83–97.

LI, X., M. K. NG, and Y. YE. 2014. Multicomm: finding community structure in multi-dimensional networks. IEEE Transactions on Knowledge and Data Engineering, **26**(4): 929–941.

LIN, Y.-R., J. SUN, H. SUNDARAM, A. KELLIHER, P. CASTRO, and R. B. KONURU. 2011. Community discovery via metagraph factorization. TKDD, **5**(3): 17.

LIN, Y.-R., S. ZHU, H. SUNDARAM, and B. L. TSENG. 2009. Analyzing communities and their evolutions in dynamic social networks. ACM Transactions on Knowledge Discovery from Data, **3**(2, Article 18).

MAGNANI, M., B. MICENKOVA, and L. ROSSI. 2013. Combinatorial analysis of multiple networks. *In* arXiv:1303.4986v1.

MUCHA, P. J., T. RICHARDSON, K. MACON, M. A. PORTER, and J.-P. ONNELA. 2010. Community structure in time-dependent, multiscale, and multiplex networks. Science, **328**(5980): 876–878.

NEWMAN, M. E. J., and M. GIRVAN. 2004. Finding and evaluating community structure in networks. Physical Review, **E69**: 026113.

PARK, Y. J., and M. S. SONG. 1989. A genetic algorithm for clustering problems. *In* Symposium on Genetic Algorithms (SGA-98), July 22–25, University of Wisconsin Madison, Wisconsin, pp. 568–575.

TANG, L., H. LIU, and J. ZHANG. 2012a. Identifying evolving groups in dynamic multi-mode networks. IEEE Transactions on Knowledge and Data Engineering, **24**(1): 72–85.

TANG, L., X. WANG, and H. LIU. 2009. Uncovering groups via heterogeneous interaction analysis. *In* The Ninth IEEE International Conference on Data Mining (ICDM'09), April 30–May 2, Sparks, NV, pp. 503–512.

TANG, L., X. WANG, and H. LIU. 2012b. Community detection via heterogeneous interaction analysis. Data Mining and Knowledge Discovery, **25**(1): 1–33.