Online Social Networks and Media 3–4 (2017) 32–48

Contents lists available at ScienceDirect

Online Social Networks and Media

journal homepage: www.elsevier.com/locate/osnem

Node-centric Community Discovery: From static to dynamic social network analysis

Giulio Rossetti[b,*], Dino Pedreschi[a], Fosca Giannotti[b]

[a] Department of Computer Science, University of Pisa, Italy
[b] Knowledge Discovery and Data mining Laboratory, ISTI-CNR, Pisa, Italy

ARTICLE INFO

Article history:
Received 6 October 2017
Revised 27 October 2017
Accepted 27 October 2017

Keywords:
Community Discovery
Complex networks
Dynamic networks

ABSTRACT

Nowadays, online social networks represent privileged playgrounds that enable researchers to study, characterize and understand complex human behaviors. Social Network Analysis, commonly known as SNA, is the multidisciplinary field of research under which researchers of different backgrounds perform their studies: one of the hottest topics in such diversified context is indeed Community Discovery. Clustering individuals, whose relations are described by a networked structure, into homogeneous communities is a complex task required by several analytical processes. Moreover, due to the user-centric and dynamic nature of online social services, during the last decades, particular emphasis was dedicated to the definition of node-centric, overlapping and evolutive Community Discovery methodologies.

In this paper we provide a comprehensive and concise review of the main results, both algorithmic and analytical, we obtained in this field. Moreover, to better underline the rationale behind our research activity on Community Discovery, in this work we provide a synthetic review of the relevant literature, discussing not only methodological results but also analytical ones.

1. Introduction

In the era of big data, Online Social Networks, also known as OSN, have allowed the collection and analysis of huge – and fine-grained – information regarding human sociality and behaviors. Such semantic rich analytical contexts opened the way to both methodological and social studies: during the last decades, novel algorithms were devised to cope with the continuously increasing quantity of data and, thanks to them, unprecedented analytical studies were performed to validate/disprove sociological theories.

Among the Social Network Analysis tasks one has received growing attention: Community Discovery (henceforth, CD). The problem of extracting communities of users from a social graph, or of dividing its vertexes into clusters, has been approached from several perspectives. Algorithms for community extraction have appeared in practically all scientific fields, not only in SNA: physics, biology, and computer science are only a few contexts in which, even with a different perspective, the CD problem has been deeply investigated.

Indeed, the multiplicity and heterogeneity of the existing CD approaches is probably due to the absence of a formal and shared definition of what a "community" really is. Such problem ill-posedness reflects the fact that, even though each algorithm extracts coherent network substructures, it does so by maximizing a given, even ad-hoc, topological objective function. Unfortunately, every quality function potentially describes a different optimal network partition. Thus several families of Community Discovery approaches emerged, each one preserving different clustering peculiarities.

From a Social Network Analysis perspective, communities are the most basilar bricks that make possible the analysis of complex phenomena: indeed, being able to identify tightly connected sets of nodes allow an in-depth analysis of the human sociality. In this scenario, due to the increasing availability of OSNs data, we have witnessed the appearance of a vast number of algorithms tailored to capture specific characteristics expressed by human interactions.

Observing the social structure of several OSNs, it was noticed that the neighborhood of a single node is often composed of a vast number of peers belonging to different semantic contexts (i.e., school, work, sports related...). However, not all the node's acquaintances can be considered as "real" ones: conversely from the real world experience, in online services the action of establishing new ties has no cost. Thus, most of the links observed in an OSN do not represent relations existing in real life. For such reason, CD solu-

* Corresponding author.
E-mail addresses: giulio.rossetti@isti.cnr.it, giulio.rossetti@gmail.com (G. Rossetti), dino.pedrechi@di.unipi.it (D. Pedreschi), fosca.giannotti@isti.cnr.it (F. Giannotti).

https://doi.org/10.1016/j.osnem.2017.10.003
2468-6964/

tions in OSN contexts need, at least, to take care of two analytical constraints: (i) the identified social groups need to be as homogeneous as possible w.r.t. some semantic information, and (ii) the produced clustering must avoid the overestimation of user's sociality.

Another pressing issue related to OSN data analysis, affecting CD approaches, lies in the temporal granularity of the observations. As time goes by people tend to modify their social relationships: travels, job changes, rising of new interests are only a few examples of the causes that lead to a perturbation (and in some cases even to the ending) of interactions and social connections. Such topological evolution is most evident in the social tissue described by communities, substructures that can born, die, and change through time.

In such complex scenario, a peculiar category of CD methods has been able to increasingly gather the attention of researchers as well as data analysts: *node-centric* (also called *local-first*, or *bottom-up*) approaches. CD algorithms that fall into such family pose each individual at the center of the community identification: to do so, they usually decompose the observed networks by iteratively analyzing each node neighborhoods, thus proceeding agnostically w.r.t. the complete graph topology. Node-centric approaches are often adopted to speed up the community extraction as well as to ensure homophilic behaviors among the nodes of the identified mesoscale topologies. Moreover, when dealing with temporally evolving networks, the adoption of local approaches enable to design incremental decomposition strategies that can operate by exploiting local topology perturbations.

In this work we organize and discuss some of our contribution to Community Discovery, placing them within the relevant literature of the field. To do so, in Section 2 we introduce the Community Discovery problem and fix some preliminary concepts that will allow the reader to understand the need for node-centric, overlapping, approaches in the analysis of social data better. Section 3 focuses on node-centric algorithms designed to partition and analyze static networks: moreover, in such section, we review some example of data-driven studies built upon them. Symmetrically, Section 4 focuses on local Dynamic Community Discovery approaches, providing a classification and exemplifying their use-cases. In Section 5 are reviewed different strategies to evaluate network partitions, synthetic benchmarks, and dynamic communities related issues. Finally, Section 6 concludes the paper, providing a discussion and overview of future research directions.

2. Community Discovery

Community Discovery is a very relevant problem in the complex network analysis field [15,26]. One of the main reason behind the attention it has received in the last decades lies in its intrinsic complexity, primarily due to its ill-posedness. Indeed, a precise and unique definition of what a community is cannot be proposed for the problem itself can be defined from multiple points of views. In the absence of a general topological ground truth partition of a given graph each algorithmic approach designed to solve the CD problem relates its results to a specific quality function. Such peculiarity has lead to the definition of several "meta" community definitions: for instance, classic works intuitively describe communities as sets of nodes closer among them than with the rest of the network, while others, looking at the same problem from another angle, only define such topologies as dense network subgraphs.

To maintain a more general perspective, we will adopt a definition proposed in [15] so to create an underlying concept able to generalize to all the variants found in the literature:

Definition 1 (Community)**.** A community in a complex network is a set of entities that share some closely correlated sets of actions with the other entities of the community. We consider a direct connection as a particular and very important, kind of action.

Moving from such general definition, in the following sections, we will describe a specific class of approaches that were designed with the aim of extract meaningful partitions from social networks: our analysis will thus focus on node-centric Community Discovery for both static and dynamic social networks.

2.1. A social networks perspective

Social networks, both online and offline, describe peculiar complex systems where the nodes, or *agents*, are individuals and the edges connecting them model some kind of social relation (kinship, co-working, direct interaction...) or shared attribute (nationality, age...). The particular semantics attached to nodes and edges are often used to guide the interpretation of analytical results: for such reason, the tools used to study social networks contexts need to be carefully chosen so to preserve, as much as possible, non-topological characteristics.

In particular, when applied to social networks, Community Discovery approaches usually aim to bound social contexts (i.e., schools, organizations...) as well as homogeneous actor characteristics (i.e., age, education level, income...). Such final goals make some classes of approaches more suitable than others. In order to better highlight the characteristics we would expect from the result of a social network decomposition two Community Discovery dichotomies needs to be tackled: *top-down* vs. *bottom-up* extraction and *crisp* vs. *overlapping* partitions.

2.1.1. Bottom-up vs. top-down

Usually Community Discovery algorithms are designed to follow either a divisive or an agglomerative schema.

Top-down approaches, as [30,50,54,60], follow the former strategy: starting from the whole graph they recursively break down communities by disconnecting its components removing either nodes or edges. An archetypal example is provided by the Grivan–Newman [50] algorithm. Such approach, during each iteration, identifies the edge having highest edge betweenness and removes it: its execution generates a dendrogram describing the hierarchy of communities present in the analyzed graph.

Bottom-up methods, as [7,16,17,59], conversely, build the community hierarchy in an agglomerative fashion starting from the leafs (i.e., the individual nodes), not from the root (i.e., the whole graph). A well-known example of bottom-up approach is indeed provided by the Louvain [7] algorithm. Louvain aims to optimize a specific quality function, modularity [49,50]. It works following an iterative two-step procedures. First, each node joins one community among the ones of its neighbors, choosing it so to maximize the modularity increase. Secondly, the identified communities are collapsed in *meta-nodes* and a novel graph – which identifies a new level in the hierarchy – is built connecting them. At the end of the second step, a new iteration begins.

As we can easily observe, bottom-up and top-down algorithms represent opposite approaches to the Community Discovery problem. In SNA, often, the center of the investigation is the individual: apart analyzing the network as a whole, the final goal of Social Network Analysis is to understand the behavior of the actors that compose it. From such viewpoint, bottom-up partitioning strategies allow to maintain a favorable analytical position: they keep the computation as near as possible to the objects of the analysis and allow the researcher to decide when, and how, to stop performing aggregations. Moreover such strategies, due to their nature, often enable for parallelization schemas thus making easier and less costly the analysis of big data sources.

2.1.2. Crisp vs. overlapping

The second dichotomic choice that needs to be made when choosing a Community Discovery approach is related to the expected separation among the identified clusters: indeed, some approaches are designed to produce *crisp*, neat, node partitions while others identify *overlapping* ones allowing nodes to belong to several communities at once. Crisp partitions are usually obtained, for instance, by methods that optimize modularity [49,50], as the already discussed Louvain [7], and conductance [71]. Overlapping communities, on the other hand, are usually identified by pattern-based approaches, such as ego-networks [16], cliques [52], as well as flow-based approaches, i.e., label propagation [59].

Indeed, slightly different problem definitions, analytical goals as well as semantic contexts profoundly affect the choice among those two families of approaches. Often, when social networks are the objects of analysis, a crisp node partition oversimplifies the complexity underlying the observed topology. Social communities are often used to identify and separate different contexts an actor is involved in: professional contacts, sports ones, family, are all "communities" each node is expected to participate at the same time. Such semantic distinction, however, is rarely completely neatly reflected by the network topology the node is embedded into: for such reason, overlapping approaches, at the cost of a reduced effectiveness concerning classical quality measures, are often considered as valuable options in SNA.

3. Node-centric communities in social networks

In the era of big data analytics, where networks of billions of nodes are generated and collected on a daily basis by a plethora of online and offline human-related activities, temporal constraints are indeed a pressing issue for both analysts and researchers. Being able to process huge networks so to extract in real-time useful knowledge from them is for sure one of the main objectives of cutting-edge companies. Indeed, partition a social network is, usually, a computationally expensive task: for this reason, the definition of easily parallelizable CD approaches (even at the cost of degrading the quality of the identified mesoscale structures) has become a relevant problem for the research community.

As discussed, to preserve some non-topological characteristics often observed in social network contexts as well as to speed up the computation time, several classes of node-centric and overlapping Community Discovery approaches have been proposed so far. Among them we can recall seed set expansion [43,44,48,85,88], diffusion based [19,36,59,76] and ego-network based CD [9,16,17,34,57,72].

In Section 3.1 we will focus our attention on the latter subclass of approaches, describing the rationale behind them and the general algorithmic pattern they implement. In Section 3.2, for sake of completeness, we briefly describe the idea behind *seed set expansion* and *diffusion based* approaches. Moreover, in Section 3.3, we briefly introduce real-world data-driven Social Network Analysis enabled by node-centric algorithms. Finally, in Section 3.4, we provide a discussion on the analytical investigation enabled by bottom-up approaches w.r.t. the top-down ones.

3.1. Ego-networks analysis

A billion nodes social graph can, in a first instance, easily be broken down directly extracting the set of ego-networks that compose it. An ego-network is defined as follows:

Definition 2 (Ego-network)**.** Given a graph $G = (V, E)$, where V identifies the set of nodes and E the set of edges, and a node $n \in V$ the ego-network $Ego(G, n)$ is a graph induced on G, centered in n and composed by n itself (called *ego*), the n first order neighbors V_n (also called *alters*) and all the edges among them.

Indeed, several variations of such definition have been proposed to capture slightly different node-centered topologies: for instance, several works do not consider edges connecting alters within the ego-network, as in [4], or remove the ego-node from it, as in [16,17]. In the latter scenario, the resulting topology is referred to as *ego-minus-ego* graph.

Each ego-network provides a node-centric perspective of the social graph, as shown in Fig. 1, capturing all those relations that are meaningful for the *ego* and discarding those that do not affect its immediate surroundings. The growing availability of social media data has indeed allowed for extensive studies of such peculiar, local, topologies leading to valuable insights also regarding offline social contexts. In particular, in [4] several Facebook and Twitter datasets were analyzed to show that ego-networks extracted from online social networks maintain the same qualitative and quantitative properties of human ego networks in general. Moreover, in the same work the role of ego-networks in information diffusion processes were observed showing that, by considering their structural properties, it is possible to model information diffusion cascades both at the individual level, as well as at the entire network level. Indeed, ego-networks are the minimal bricks that constitute complex social tissue.

3.1.1. Ego-network based Community Discovery

Due to their nature, ego-networks can be easily seen as prototypical community units – at least in social network contexts. Moving from such observation in [16] we introduced one of the first local, bottom-up, approaches aimed to identify overlapping communities: Demon.[1] Demon explicitly makes use of the first order node surroundings to build up mesoscale structures: conversely, from classical bottom-up approaches, like Louvain, it does not explicitly optimize any quality function.

The rationale behind our approach lies in the observation that different egos should have different perspectives over the same neighbors and it is the union of all these perspectives that creates an optimal partition of the network. In other words: only if two nodes are placed in the same community by all the nodes connected to both of them, then they should be grouped in the same cluster. Such result is achieved by a democratic, bottom-up mining approach: in turn, each node gives the perspective of the communities surrounding it and then all the different perspectives are merged into an overlapping structure.

Demon, as well as most of the ego-network based approaches that followed it, implements a simple but effective algorithmic schema, logically composed of three steps (see Algorithm 1 for the pseudocode):

Algorithm 1 Algorithmic schema for ego-network bottom-up algorithms.

Require:
$\mathcal{G} = (V, E)$: an undirected graph
Ensure: set of communities $\mathcal{C}$

1: $\mathcal{C} = \emptyset$
2: **for all** $n \in V$ **do**
3: $e \leftarrow Ego(\mathcal{G}, n)$
4: $\mathcal{C}(n) \leftarrow LocalCommunities(e)$
5: **end for**
6: $\mathcal{C} \leftarrow AggregateLocalCommunities(\mathcal{C})$
7: **return** $\mathcal{C}$

(i) firstly, for each node its ego-minus-ego graph is extracted (i.e., the ego-network from which the ego is removed);

[1] Code available at https://goo.gl/J1YVCM.

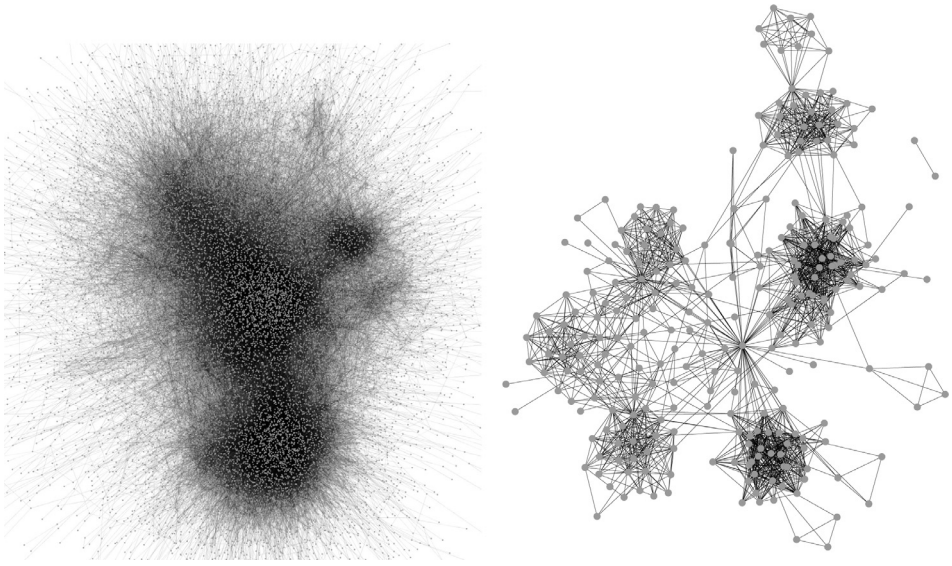

Fig. 1. Global vs. local. A global view of a 15k nodes sample Facebook vs. the ego-mins-ego graph of a single node of such graph.

(ii) secondly, from each ego-minus-ego graph local communities are extracted (in the Demon case using an overlapping Label Propagation [59] approach);
(iii) finally, all the local communities are merged using a thresholding function to obtain the final overlapping partition.

The last step in Demon is performed by applying as similarity function a variant of the Jaccard measure. In particular, given two communities C_i and C_j, where $|C_i| \leq |C_j|$ their similarity is computed as $\frac{C_i \cap C_j}{C_j}$ in order to evaluate the percentage of overlap w.r.t. the smallest topology. Indeed, since such comparison is pairwise, it represents one of the critical aspects of bottom-up ego-network based approaches, an issue that can pose a serious computation bottleneck due to its algorithmic complexity.

To address such issue, in [17] we extended Demon proposing a hierarchical merge function. HDemon follows the same rationale of Demon but implements community merge using a community induced graph (a strategy previously applied in Louvain, see Section 2.1.1).

Other works address the same issue, both proposing a parallel implementation of our approach, PanDemon [3], as well as changing the local community detection approach, NodePerception [75]. Following a similar rationale SONIC [72] is designed to extract ego-networks based communities in a dynamic, streamed, network scenario. Finally, in [21] a scalable and easily parallelizable non-overlapping Community Discovery ego-network based approach is defined. Indeed, going further from Demon derived approaches, ego-networks have proven to be a fertile starting point to perform community detection analysis [9,14].

3.2. Seed set expansion and diffusion-based approaches

Similarly, from ego-network based approaches, seed-centric algorithms focus their attention on the node perception of the clustered graph structure. The basic idea underlying these approaches consists of identifying particular nodes in the network, called *seeds*, around which communities can then be identified. Indeed, different algorithms adopt different seed definitions and community expansion strategies, as discussed in [33]. As for ego-network based Community Discovery, we can identify a general algorithmic schema for seed-set expansion strategies.

As shown in Algorithm 2, seed set expansion approaches can be easily decomposed in two steps: identification of the seed nodes and definition of an iterative rule that specify how the community they identify forms around them. An example of seed set expansion approach is indeed LICOD [32]. Such approach, implementing a leader-based community detection strategy, operates the following choices:

Algorithm 2 Algorithmic schema for seed-set expansion algorithms.

Require:
$\mathcal{G} = (V, E)$: an undirected graph
Ensure: set of communities $\mathcal{C}$

1: $\mathcal{C} = \emptyset$
2: $S \leftarrow$ ComputeSeeds($\mathcal{G}$)
3: **for all** $s \in S$ **do**
4: $C_s \leftarrow ComputeLocalCommunity(s, \mathcal{G})$
5: $\mathcal{C} \leftarrow \mathcal{C} - C_s$
6: **end for**
7: **return** $\mathcal{C}$

- the seed set is identified by all nodes having a centrality higher that $\sigma\%$ of their direct neighbors, then it is reduced by grouping together seeds (i.e., *leaders*) estimated to belong to the same community – a community must be associated with a single seed.
- Each non-seed node compute its, ranked, degree of community membership for all the seeded communities, a rank subsequently updated by considering the ones of its direct neighborhood.
- Finally, each node is assigned to its top-ranked community.

The idea of identify special nodes as community *representatives* indeed allows the definition of scalable and easily parallelizable bottom-up approaches. For such reason, the same strategy has also been used in dynamic network context [83] where the seeds acquire the critical role of connecting temporally displaced network partitions.

Another family of local approaches to community diffusion is the one commonly referred to as *diffusion* based [15]. A diffusion community can be defined as:

Definition 3 (Diffusion community)**.** A diffusion community in a complex network is a set of nodes that are grouped by the propagation of the same property, action or information in the network.

Diffusion Community Discovery algorithms let each node in the graph to autonomously chose its community by observing the choices made by its neighborhood – or, in some variations, by evaluating its participation in particular local patterns [19,36] (e.g., cliques, quasi-cliques). The final partition is thus obtained by performing a diffusion or percolation procedure on the network – applying algorithm specific rules – and then group nodes that end up in the same state. The algorithmic schema behind diffusion based CD is the one shown in Algorithm 3.

Algorithm 3 Algorithmic schema for diffusion based algorithms.

Require:
$\mathcal{G} = (V, E)$: an undirected graph
Ensure: set of communities $\mathcal{C}$

$\mathcal{C} = \emptyset$
while True **do**
  $\mathcal{C}_{new} \leftarrow \emptyset$
  for all $n \in V$ **do**
    $\mathcal{C}_{new} \leftarrow UpdateNodeLabel(n, \mathcal{G}, \mathcal{C})$
  end for
  if $\mathcal{C} == \mathcal{C}_{new}$ **then**
    return $\mathcal{C}$
  else
    $\mathcal{C} \leftarrow \mathcal{C}_{new}$
  end if
end while

A classic example of this family of approaches is offered by the Label Propagation algorithm (also known as LP) [59]. LP assumes that each node in the network joins the community to which the maximum number of its neighbors belongs. Node labels, identifying community affiliation, propagates through a densely connected group of nodes until a consensus is reached. Several variants of LP has been proposed to cope with peculiar community definitions allowing, for instance, to identify overlapping structures [76]. Due to the fast computation offered by some diffusion based algorithms (i.e., LP having quasi-linear time complexity) their partitions are often used as preprocessing step by other CD approaches (i.e., DEMON [16]) that upon them builds more refined mesoscale topologies.

3.3. Applications

The ability of efficiently identify partitions of large social graphs keeping a node-centric perspective enables for several analytical applications of Community Discovery. In the following, we discuss how ego-centric communities, in particular, the ones produced by Demon and HDemon, can provide bound to homophilic behaviors observed in online social networks. In particular we will describe two analytical works that tackle relevant SNA problems: *homophilic network decomposition*, Section 3.3.1 and *network quantification*, Section 3.3.2.

3.3.1. Homophilic network decomposition

The increasing availability of Big Data describing customers behavior has changed the way Companies advertise their services. Online shopping site, as well as OSNs, are nowadays collecting data on their users' activities and sociality to extract information which can guarantee them an edge on competitors. In this scenario, being able to identify how much users are engaged in a specific product/service offered by a brand (i.e., Skype video call, Facebook chat, Dropbox file sharing, Google online document editing...) is a powerful tool. Indeed, such analysis that can be used to drive decision on future commercial strategies [6,20] as well as analyze churn rates and characteristics [51,62]. Moreover, the success of a product/service is often due to the virality it is able to achieve. To broaden the diffusion of a particular product, it becomes mandatory identify a fertile ground, a set of potential users that are likely to be interested in it. Several SNA studies have shown that *homophily* is a property that can be observed in almost all human social networks: people tend to cluster homogeneously by age, location, interests and, more important in our scenario, tastes.

Social communities are, perhaps, the smallest topologies that can be used to bound such phenomenon and that can provide indicators able to show who to target when programming advertising campaigns. In [66,68] we studied the effectiveness of a supervised learning model aimed to predict user engagement level starting from community topological features. We train such model on features computed on the outcome of four different CD approaches: Ego-networks, HDemon, Louvain and BFS sampling.

Our analysis was carried on three real-world datasets: the whole Skype contact graph, a 75k user Last.fm network and a 33k Google+ user dataset [29]. For each dataset a different proxy for user engagement was identified: monthly number of usage for Video/Chat/Audio in Skype, number of song listenings in Last.fm, homogeneity/heterogeneity of education level in Google+. In Skype and Last.fm, we discretized the user engagement levels in high/low by analyzing the distribution of the chosen target variable. Doing so, we identify two scenarios, a balanced one, where we separate on the 50th percentile, and an unbalanced one, built upon the 75th percentile. However, such different settings did not impact significatively on the final analytical results.

On the Skype network, our results showed that algorithms producing overlapping micro-communities like HDemon and ego-networks reach the best performances (in Fig. 2 are shown the performances of the predictors built on top of each CD approach w.r.t. average community density and size). Conversely, modularity-based approaches like Louvain do not guarantee good performance and are often outperformed by naive algorithms such as BFS. Similar analytical results were obtained on Last.fm, while in the balanced scenario of Google+, the only dataset among the analyzed ones that do not take into account explicit user activities, Louvain communities were able to outperform the other partitioning methods slightly. In such specific scenario, the target regards personal information, which may represent one of the reasons behind the presence of some network connections (i.e., if they studied together): however, such semantics it is not necessarily the glue that keeps communities together. Such peculiarity indeed privileged approaches that break the network tissue in few large communities covering a high percentage of nodes, that even if misclassified have reduced impact on the overall accuracy, and a high number of small ones. That phenomenon becomes more evident considering the classification results in the unbalanced scenario where, as in Skype and Last.fm, the most effective partitions are again the ones produced by HDemon and ego-networks due to the more selective criterion used to filter out medium-size communities of produced by such bottom-up approaches.

3.3.2. Quantification in social networks

Another interesting problem which still poses its grounds on the existence of homophilic behaviors within well-defined network substructures is the *quantification* one.

Many real-world applications require estimating and monitoring the distribution of a population across different classes. An example of such applications is the crucial task of determining the percentage (or "prevalence") of unemployed people across different geographical regions, genders, age ranges or even temporal observations. In the literature, this task has been called *quantification* [22,25,46,79].

Quantification is closely related to *classification*. However, the goal of classification differs from the quantification one since, in

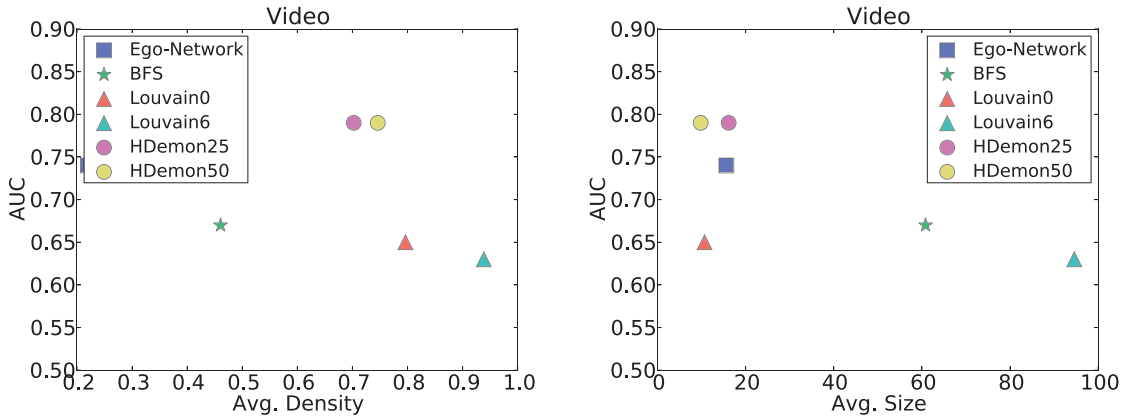

Fig. 2. CD approach and their effectiveness in predicting the user engagement on the Skype Video service. Dense and small communities, as the one identified by HDemon provides a better proxy for user engagement prediction.

the former, the focus is on correctly guessing the real class label of every single individual while, in the latter, the aim is to estimate class prevalence. Classification and quantification differ because, while a perfect classifier is also a perfect quantifier, not necessarily a good classifier is also a good quantifier. Indeed, a classifier that generates on the test set a similar number of misclassified items over the different classes is a good quantifier because the compensation of the misclassifications leads towards a perfect estimation of the class distribution.

Most of the works address the quantification problem taking into consideration data presented in conventional attribute format. Since the ever-growing availability of web and social media, we have a flourish of networking data representing a new valuable source of information. In this scenario an interesting question arises: *how can the quantification be performed in contexts where the observed entities are related to each other?*

The impact of quantification techniques for networking data is potentially high: this because today we are witnessing an ever more effective dissemination of social networks and social media where people express their interests and disseminate information on their opinions, about their habits, and their wishes. The possibility to analyze and quantify the percentage of individuals with specific characteristics or a particular behavior could help the analysis of many social aspects. For example, analyzing social platforms like Facebook or Google+, as we did in [47], where – as we have already discussed – users can choose to specify their education level, we could estimate the level of education of an entire population even in the presence of missing, or evolving, data. Following the same rationale, using a quantification approach, we could determine the distribution of the political orientation or the geographical origin of the social network population. In [47] we compare quantification approaches based on Demon [16], Infohiermap [71] and ego-networks on three different datasets: (i) Google+, where the target variable is the education level, (ii) CoRa, a reference based graph built upon a computer science bibliographic library where class labels represent topics, and (iii) IMDB a movie-to-movie network where each node's label capture whether the opening weekend box-office sales have exceeded $2 million or not. In such scenarios, we leveraged social networks homophilic behaviors to assign labels to the unlabeled nodes belonging to each of the clusters generated by the selected algorithms. We applied two strategies: (i) *density based*, where the class label selected is the highest frequency one of the denser community to which the unlabeled node belongs, and (ii) *frequency based*, where to each unlabeled node is assigned the class having the greatest overall relative frequency across all the communities the nodes belongs to. The latter strategy in case of ego-network partitioning lead the selection of the highest frequency class among the direct neighbors of the unlabeled ego node, as shown in Fig. 3. Experimental results highlight that the latter approach constantly outperforms the former concerning quantification quality (measured via KLD, Kullback–Leibler divergence [25]). Moreover, the more the topologies used to assign class labels to unlabeled nodes are small the more likely that the predicted class distribution will better approximate the real one, as shown in Table 1 for Google+. In this particular scenario, ego-networks represent the best choice since they enable for a tighter bound of homophilic phenomena, confirming what already observed in Section 3.3.1.

3.4. Discussion

In this section, we focused our attention on node-centric approaches to Community Discovery. As discussed, such strategies answer two different demands: (i) they, in principle, allow for a reduction of computational costs since approach the partitioning problem at the local level, and (ii) they better capture node perspective while partitioning the overall network.

As shown in the applications reported in Section 3.3, while analyzing data extracted from social contexts (ONSs, call graphs...) CD approaches able to produce small and strongly connected communities – as node-centric ones are – allow to bound homophilic behaviors. Indeed, such property is primarily due to the bottom-up approach they follow and to their tendency toward the discovery of a relatively high number of communities composed of a reduced number of nodes. Top-down partitioning approaches, on the other hand, tend in general to break the graph into few communities, some of which covering a significant percentage of its nodes. This latter kind of partitions – at least in a social context – do not allow for the extraction of valuable insights due to the heterogeneous distribution of information that medium-large sized communities contain. Indeed, what observed in a social context cannot be transferred directly to other, different, ones.

Such issue is not limited to bottom-up vs. top-down analysis. An alternative example that underlines the importance of contextual information on the selection of a CD algorithm is related to the community shape. If while partitioning a social graph overlapping mesoscale structures make sense – due to the multiple environments individuals can participate in – when analyzing a different context they may not (e.g., scenarios modeling mutually exclusive choices, or tasks requiring to identify well separate modules).

Indeed, there not exist the *perfect* Community Discovery approach able to always outperform its competitors. Conversely, for each context and analytical request, it is possible to identify a set

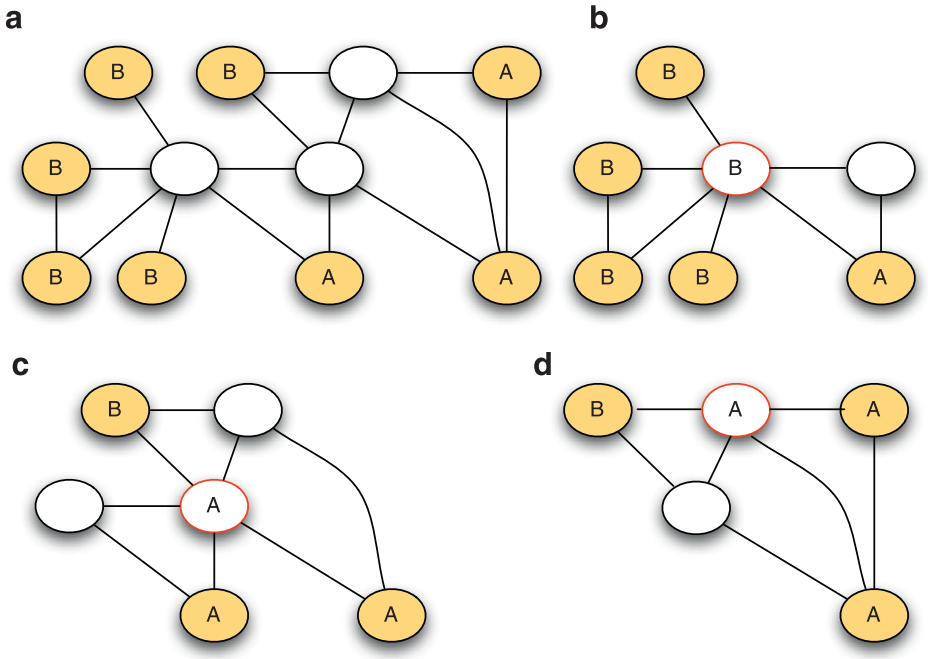

Fig. 3. Frequency based ego-network quantification. For each unlabeled node is extracted its one-hop ego-network and the higher frequency label within it is selected.

Table 1
Google+ Edu: mean of KLD scores for the frequency based approaches. Tests were performed neglecting class labels for 20% and 30% of the graph nodes sampled using a random strategy (RS) bottom-degree (BS) and top-degree (TS).

RS			BS		TS	
Method	20	30	20	30	20	30
Infohiermap	2.744e−3	2.641e−3	5.385e−1	4.822e−1	1.061e+0	9.135e−2
Demon	3.092e−1	3.150e−1	2.840e−1	2.816e−1	2.152e−1	1.887e−1
Ego 1-hop	**1.160e−3**	**1.354e−3**	3.424e−3	2.508e−3	**1.860e−3**	**2.531e−3**
Ego 2-hop	3.504e−3	4.177e−3	**2.531e−3**	**1.795e−3**	1.407e−2	1.167e−2

of approaches capable of preserving some of the network semantics within the partitions they generate.

4. Social network dynamics

So far our attention has been focused on the analysis of *static* social graphs, however, a peculiar characteristic of social phenomena is that they naturally evolve as time goes by.

Since its beginning, complex network analysis has been approached through the definition of very specific, task-oriented, mining problems. The almost complete absence of the time dimension in such definitions comes from historical reasons that can be identified in (i) the graph theory ancestry of the field, and in (ii) the reduced number of dynamic data sources available at the time the area of complex networks analysis emerged. Indeed, such scenario radically changed during the last decades' thanks to the explosion of human-generated data collected via socio-technical platforms: wide repositories of time-aware data that can be easily modeled as dynamic networks.

Graphs have often been used to model and study dynamic phenomena: to better describe these realities, in which relationships among agents change through time, several works in the last few years have started to lay the foundations of temporal network analysis.

Indeed, there is a significant number of social systems that can, potentially, be modeled as temporal networks. In addition to cellular processes and social communications, large infrastructures (i.e., call graphs and web graphs) posses both network and temporal aspects that make them attractive for temporal network modeling.

The emergence of such theoretical grounds has been highlighted in the book "Temporal Networks" [31] where the curators, Holme and Saramaki, propose an ensemble of works covering different dynamic network analysis methodologies. As a first step, several works have tried to transpose known problems on static networks to temporal networks: Temporal motifs mining [35], Diffusion [41,56], Link prediction [77], are only a few examples.

Moreover, to support the definition of such revised analytical framework, several formalisms have been proposed to represent evolving networks without loss of information: Temporal Networks [31], Time-Varying Graphs [10], Interaction Networks [67], and Link Streams [81], to name the most famous. Henceforth, we use the term *dynamic network* to encompass all those formalisms. In order to be as general as possible, we will define a dynamic network as done in [64]:

Definition 4 (Dynamic network)**.** A dynamic network is a graph $DG = (V, E, T)$ where: V is a set of triplets of the form (v, t_s, t_e), with v a vertex of the graph and $t_s, t_e \in T$ are respectively the birth and death timestamps of the corresponding vertex (with $t_s \leq t_e$); E is a set of quadruplets (u, v, t_s, t_e), with $u, v \in V$ are vertices of the graph and $t_s, t_e \in T$ are respectively the birth and death timestamps of the corresponding edge (with $t_s \leq t_e$).

In a dynamic context, all network entities can vary as time goes by. In a social scenario, such flexibility naturally models the

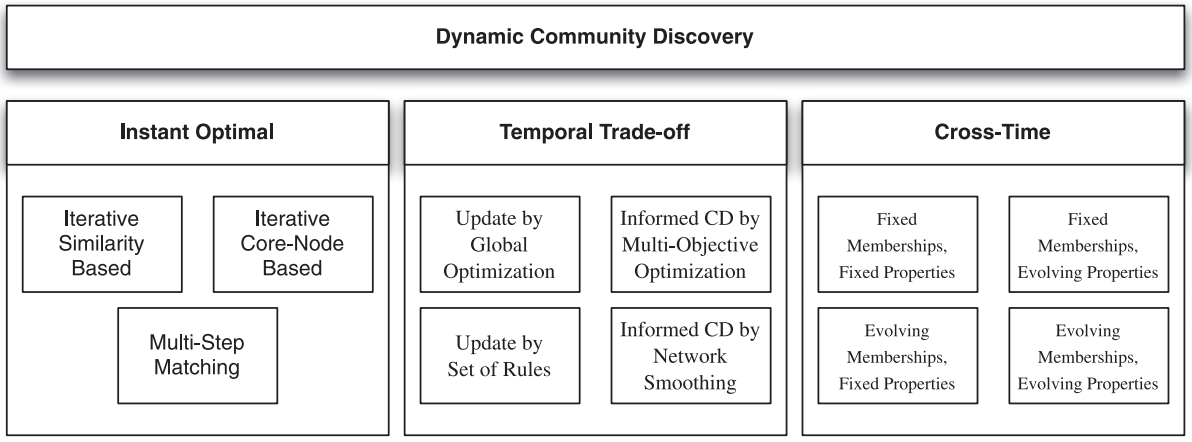

Fig. 4. Classification of DCD approaches. Two level taxonomy proposed in [64].

volatility of some human interactions (e.g., telephone call, face to face meetings, email exchange...) whose valence/duration tend to be overestimated when they are analyzed through classical graph theory tools.

4.1. Dynamic community discovery

Indeed, moving from a static to a dynamic model deeply impacts the definition of community and the analytical tasks that can be built around such concept.

As done for in Section 2 to avoid making any assumption on the nature of communities we adopt a generic definition (borrowed from [64]) to describe the Dynamic Community Discovery (henceforth DCD) problem.

Definition 5 (Dynamic Community Discovery)**.** Given a dynamic network $DG = (V, E, T)$, a Dynamic Community DC is defined as a set of distinct (node, periods) pairs: $DC = \{(v_1, P_1), (v_2, P_2), \ldots, (v_n, P_n)\}$, with $P_n = ((t_{s0}, t_{e0}), (t_{s1}, t_{e1}) \ldots (t_{sN}, t_{eN}))$, with $t_{s^*} \leq t_{e^*}$. Dynamic Community Discovery aims to identify the set $\mathcal{C}$ of all the dynamic communities in DG. The partitions described by $\mathcal{C}$ can be neat as well as overlapping.

The time evolving clustering $\mathcal{C}$ (or set of Dynamic Communities) of a dynamic network DN captures, as time goes by, how nodes organize their self into mutable topological substructures.

4.1.1. Local Dynamic Community Discovery approaches

Even though the DCD problem started receiving significative attention only recently the number and variety of algorithms designed to operate on a dynamic network is non-negligible. To allow practitioners to quickly identify the DCD approach that best suit their needs in [64] we introduced a first comprehensive classification of existing methods.

In our taxonomy, synthesized in Fig. 4, the higher level corresponds to the different definition of what are Dynamic Communities, without assumptions on the technique used to find them. These high-level classes are then divided into subcategories, which correspond to different methods used to find communities corresponding to this definition.

We identified three major families of approaches:

- The first one (*instant-optimal CD*) assumes that communities existing at t only depend on the *current* state of the network at t. Matching communities found at different steps might involve looking at communities found in previous steps, or considering all steps, but communities found at t are considered optimal, w.r.t. the topology of the network at t. Approaches falling in this class are *non-temporally smoothed*.
- In the second class (*temporal trade-off CD*), communities defined at an instant t do not only depend on the topology of the network at that time, but also on the *past* evolutions of the topology, *past* partitions found, or both. Communities at t are therefore defined as a trade-off between optimal solution at t and known past. They do not depend on future modification, an important point for "on the fly" CD. Conversely, from the approaches falling in the previous class, temporal trade-off ones are *incrementally temporally smoothed*.
- In the third class (*cross-time CD*), the focus shifts from searching communities relevant at a particular time to searching communities relevant when considering the whole network evolution. Methods of this class search a single partition directly for all time steps. Communities found at t depends *both on past and future* evolutions. Methods in this class produce communities that are *completely temporally smoothed*.

Such taxonomy does not take in consideration the top-down/bottom-up nature of the DCD approaches on purpose. Conversely, it aims to highlight the assumptions a DCD approach made on community stability – the highest level of the hierarchy – as well as how such assumptions reflect in the algorithmic solution it proposes.

For instance, within the *instant optimal* class fall all those approaches – we called them *iterative similarity based* – that leverage classical algorithms designed for static networks applying them to temporally discretized network snapshots. Such algorithmic schema, also known as *two-step* since it separates community *identification* and *alignment* (i.e., the reconciliation of consecutive, distinct, instances of the same community), is often exploited when DCD represent a preprocessing step of more complex analytical contexts – as we will see in Section 4.2.

Indeed, as previously discussed, social interactions can model volatile connections, e.g., phone calls: in such scenarios identify a significative temporal unit to discretize the observed phenomenon is not trivial. For such reason recent research moved from snapshot community analysis to online community detection, thus trying to keep track of community changes immediately as they occur [11,12].

In [69] we introduce *tiles*, a *temporal trade-off* approach specifically designed to track community evolution by observing local perturbations caused by individual interaction

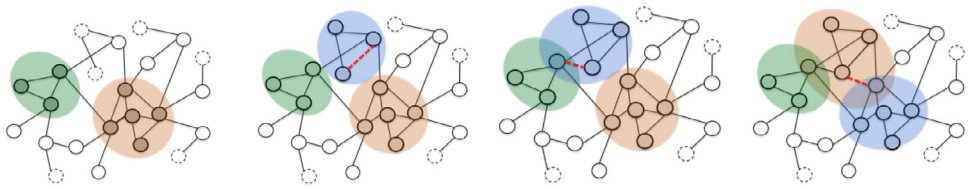

Fig. 5. Tiles Dynamic Communities. Example of how simple local perturbations – new edges represented as dashed red lines – can affect community structure. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

appearance/vanishing. Tiles[2] works by analyzing an interaction stream: it applies a constrained label propagation strategy that, enforced by a predefined set of rules, allows to efficiently discover and track through time communities, thus avoiding the need for the temporal alignment step imposed by classic two-step procedures. Tiles is, indeed, a node-centric algorithm: for each novel local perturbations, community updates are evaluated only analyzing the surroundings of the nodes involved. The rationale behind TILES is shown in Algorithm 4. Given an interaction stream – e.g., a temporally ordered log of phone calls – it operates following steps:

Algorithm 4 TILES pseudocode.

Require:
S: interaction stream,
τ: observation threshold,
ttl: interaction duration
Ensure: Set of dynamic communities $\mathcal{DC}$

$t_{old} \leftarrow 0$
$\mathcal{DC} \leftarrow \emptyset$
$\mathcal{G} \leftarrow CreateEmptyGraph()$
for all $(u, v, t) \in S$ **do**
  $RemoveExpiredInteractions(\mathcal{G}, t, ttl)$
  $\mathcal{G} \leftarrow AddNewInteraction((u, v, t))$
  $\mathcal{DC} \leftarrow LocalUpdateCommunities(\mathcal{G})$
  if $t - t_{old} \geq \tau$ **then**
    yield $\mathcal{DC}$
  end if
  $t_{old} \leftarrow t$
end for

(i) *Community contraction.* From the graph are removed all the expired interaction, i.e., all the ones for which *ttl* time units are elapsed from appearance. This cause an implicit update of those communities that contained them (death/split/shrink events);
(ii) *Graph update.* The novel interaction is added to the graph;
(iii) *Community expansion.* All the communities affected by the appearance of the interaction are evaluated and their structure modified if needed (birth/merge/growth events);
(iv) *Community observation.* Finally, if a temporal window of width τ units is elapsed from the last observation, the actual status of $\mathcal{DC}$ is returned.

Community updates are evaluated locally, evaluating only the communities whose nodes were endpoints of the processed interaction. Fig. 5 provides a toy example that illustrates how dynamic community memberships are updated by our approach.

The locality of community updates choice makes TILES easily parallelizable although defined to operate on streamed dynamic networks. Indeed, it is possible to evaluate at runtime local dependencies on consecutive community updates and, in their absence, automatically decompose Tiles execution in multiple parallel processes. We applied our approach to several data sources, both extracted from OSN (Facebook and Sina Weibo) and synthetically generated: our results suggest that it outperforms its competitor for both running time and community quality.

[2] Code available at https://goo.gl/zFRfCU.

4.2. Applications

Dynamic Community Discovery is not only an interesting problem *per se*: DCD approaches, as well as CD ones, are often used to support other graph mining tasks. While static Community Discovery enables the analyst to focus its attention on structural network model and their properties (e.g., homophilic behaviors as discussed in Section 3.3), dynamic one allows analyzing how functional graph modules unfold through time. To provide an example of how DCD approaches can be embedded into more complex analytical processes in Section 4.2.1 we describe a particular case study: *interaction prediction*.

4.2.1. Interaction prediction

Reasoning on networks evolution a very urgent question arises: *is there any rule that regulates the rising of new edges?* or similarly, *there exists couples of nodes that are most likely to establish a connection than the others?*

In the last decades, an extensive set of models were proposed with the aim to understand and reproduce real networks traits: all those models mimic, to some extent, the processes behind networks growth over time. Here, we are interested in addressing a slightly different issue: we know the nodes of our network (we assume that no other nodes could be added in successive time steps) and want to study the probability that two of them became neighbors in the future. Suggest new friendships on a social network, co-authorship on a professional network or interesting products of an online-market are for sure facilities that online services nowadays need to offer to their users. Link prediction group together all those problems. It is defined as the problem of identifying, given a snapshot of a network G at a time t_0, the top-k edges that are most likely to appear among its set of nodes, at a time t_1, restricting the prediction to those nodes that are not connected by edges during the first observation.

Correctly predict a new link in a, often, sparse network is a hard task to accomplish: for this reason, several approaches were proposed to study this evolutive aspect of complex networks, using both supervised and unsupervised methodologies. In particular, unsupervised approaches are built upon on local (neighborhood-based), or global (path-based), topological aspects relative to the pairs of nodes for which a prediction is needed. Those methodologies, given their simple nature, were shown to be able to guarantee around 10% of correct predictions in several OSN datasets: due to the complexity of the problem, this value that could seem very low is, actually, an excellent result. Often link prediction methodologies are designed to cope with static networks: indeed, classical solutions discretize the network history in two epochs the *past* – i.e., the network structure used to make predictions – and the *future* – i.e., the network structure that needs to be predicted.

Indeed, when the analyzed phenomenon is continuously evolving the adoption of temporal network modeling allows designing

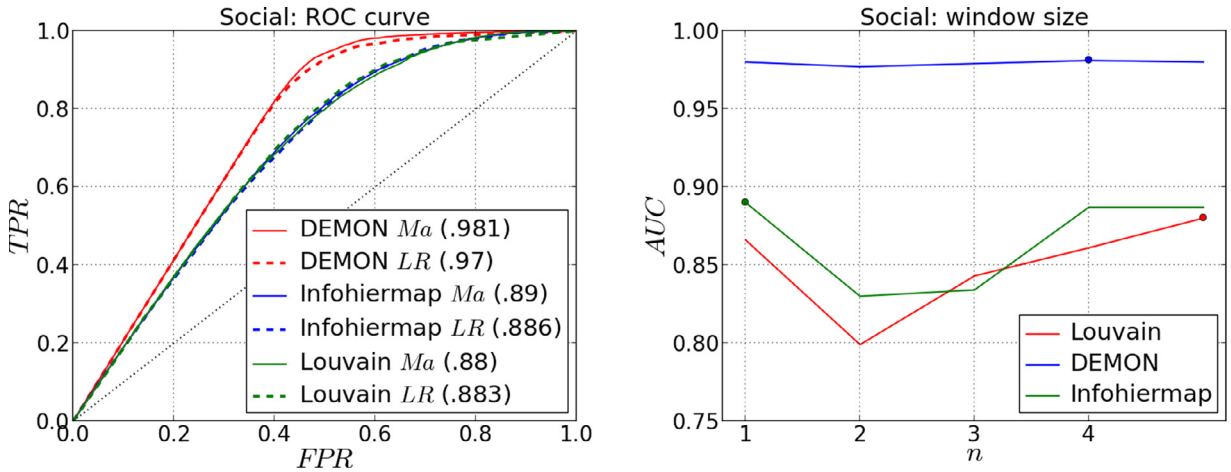

Fig. 6. Interaction prediction. Predictive accuracy of the supervised model while varying the DCD approach used in the preprocessing step (*left*) and the length, in terms of observed weeks, of the training set (*right*).

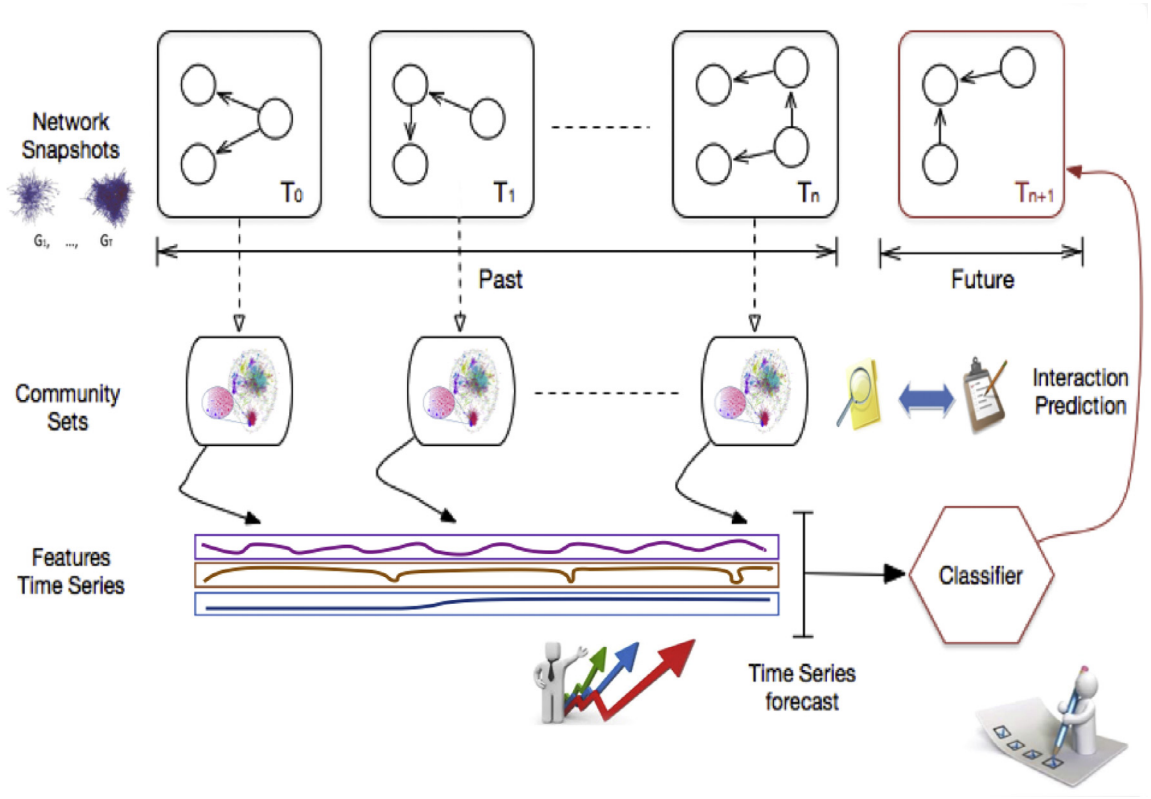

Fig. 7. Interaction prediction. Supervised learning approach workflow. Starting from a snapshot representation of the dynamic graph: (i) communities are extracted independently for each network observation; (ii) the topological features of each dynamic community are used to build features time series; (iii) the future value of each time series are forecasted; (iv) the forecasted values are used to train a supervised classifier aimed to predict the absence/presence of interactions within future evolutions of the community.

time-aware predictive models: in such cases, we talk about *interaction prediction*. In [65,67] we designed a time-aware approach to interaction prediction that exploits two-step DCD approaches as preprocessing phase.

The overall analytical process is composed of four stages, as shown in Fig. 7: (i) DCD community extraction, (ii) extraction of community-based temporal feature vector, (iii) forecasting of feature vector future value through time-series analysis and (iv) definition of a supervised learning strategy that leveraging such forecasts predict future node–node interactions. The DCD step was introduced to bound the set of eligible interactions to predict: indeed, interactions occurring among nodes that belong to the same community are more likely to occur than one that relates nodes far apart in the observed graph.

The idea behind the described workflow is to build a synthetic and holistic representation of the future state of the observed dynamic network through the forecast of a set of – interaction related – topological features. As an example, after such step, the approach will know the expected number of common neighbors (as well as the value of Jaccard coefficient, betweenness, ...) among node i and j in the future network observation. Such representation is then used as an unlabeled input for a classifier trained on

the observed dynamic graph snapshots (e.g., a dataset relating the presence/absence of an interaction to the values of the same set of topological features computed for its endpoints).

Our approach was tested both for predicting intra-community interactions as well as, with some minor modification, inter-community ones. For both scenarios, we tested our approach against real-world temporal OSN datasets: our results highlight the importance of selecting the right CD approach to boost the predictive performances. In particular, as shown in Fig. 6 for the intra-community prediction in Social (a Facebook-like dataset), we observed how ego-network based approaches (e.g., the DCD version of Demon) were able to outperform modularity and conductance based ones in terms of AUC, even while varying the length of the temporal observation used to train the model. Such results confirm what observed in Section 3.3: node-centric approaches represent a reasonable choice when designing analytical tasks in OSNs, both in static and dynamic contexts.

4.3. Discussion

As we have seen in this section, we can recognize three classes of DCD approaches, each one having advantages and drawbacks, none superior to the others. Nevertheless, each one of them is more suitable for some use cases.

For instance, if the final goal is to provide on-the-fly community detection on a network that will evolve in the future, *instant optimal* as well as *temporal trade-off* approaches represent the most suitable fit. If the analytical context requires working with a fine temporal granularity, therefore modeling the observed phenomena with temporal networks, it is strongly suggested to avoid methods of *instant optimal* approaches, since they almost exclusively deal with snapshots.

Indeed, the first layer the taxonomy proposed in [64] can be used to provide guidance and recommendations on which approach (or class of approaches) select given a specific problem formulation. For instance, we can observe how,

- *Instant optimal* approaches are the best choice when the final goal is to provide communities which are as good as possible at each step of the evolution of the network.
- *Cross-time* approaches are the best choice when the final goal is to provide communities that are coherent in time, in particular over the long-term.
- *Temporal trade-off* approaches represent a tradeoff between these other two classes: they are the best choice in case of continuous monitoring, rapidly evolving data, and in some cases limited memory applications.

Identifying the particular family a DCD approach belongs to is valuable to understand which are its real competitors and, doing so, to better organize comparative analysis and also refine the problem definition.

5. Evaluating community partitions

A major issue that profoundly affects Community Discovery approaches lies in a direct drawback of the ill-posedness of the problem itself: partition quality evaluation. In this section, we will review the classical methods used to compare and rank the outcome of different CD, as well as DCD, approaches on the same graph.

In particular, we discuss in Section 5.1 *external* evaluation and in Section 5.2 *internal* evaluation strategies. Within the former we also approach *ground-truth* comparison and *synthetic benchmarks*. Finally, in Section 5.2.1 we propose an example of *internal* evaluation strategy tailored for DCD algorithms: *community life-cyle* analysis.

5.1. External evaluation

External evaluation methodologies assume the existence of an external ground truth that needs to be retrieved or a specific partition quality score to optimize.

A common way to compare different algorithms is to rank their partitions w.r.t. a quality score. *Modularity* [49,50] is probably the most widely used quality function. It is defined as [82]:

Definition 6 (Modularity)**.** Let σ_i be the community to which node i is assigned: the expected number of edges between nodes i and j, if edges are drawn at random, is $\frac{k_i k_j}{2m}$, where k_i and k_j are the degrees of the nodes and m is the total number of edges in the network. The modularity is given by

$$Q = \frac{1}{2m} \sum_{i \neq j} \left(A_{i,j} - \frac{k_i k_j}{2m} \right) \delta(\sigma_i, \sigma_j) \tag{1}$$

where the function $\delta(\sigma_i, \sigma_j)$ is 1 if $\sigma_i = \sigma_j$ and 0 otherwise. If the number of within-community edges is lesser than the expected number of edges in a random graph, we will get $Q = 0$. Values of Q approaching 1 indicate networks with strong community structure.

Indeed, the legitimacy of modularity has been challenged in recent years: in particular [27] shows that partitions of optimal modularity do not necessarily correspond to what one expect as good communities. There the authors introduced the problem of "resolution limit", that may prevent from detecting small communities in large networks, and vice-versa. Indeed, such issue deeply limits the interpretability of the results provided by modularity based approaches to social contexts.

Although being the most famous community scoring function, modularity is not the only measure used to evaluate partition quality; we report a synthetic list of them in Table 2.

Studies about the relations between these quality functions can be found in [18,87]. Evaluating solutions based on golden quality scores has a major drawback: it favors methods that are designed to maximize it. Even though it can be used fruitfully to compare methods that optimize it, its application to approaches that search for communities having different definition may produce misleading, or inconclusive/irrelevant, comparisons. For these reasons, an alternative approach used to compare CD algorithms is often used: ground-truth testing.

5.1.1. Ground truth communities

When network semantic plays a major role – as in SNA contexts – a common strategy is often used to evaluate the identified communities: ground truth comparison [87]. Although several criticisms were opposed to this evaluation methodology [55], it is still the most widely spread to compare different algorithmic approaches. The common way to assess how a given partition resembles the ground-truth one is to compute the *Normalized Mutual Information score* (NMI, [38,39,45]) a measure of similarity borrowed from information theory, defined as:

$$NMI(X, Y) = \frac{H(X) + H(Y) - H(X, Y)}{(H(X) + H(Y))/2} \tag{2}$$

where $H(X)$ is the entropy of the random variable X associated to an identified community, $H(Y)$ is the entropy of the random variable Y associated to a ground truth one, and $H(X, Y)$ is the joint entropy. NMI is defined in the interval [0,1] and is maximal when the compared communities are identical. One drawback of NMI is that assuming an approximate size z for the compared community sets its computation requires $O(z^2)$ comparisons, a complexity that makes it often unusable to evaluate partitions of large-scale networks.

Table 2
External community quality functions. Detailed definitions can be found in the referenced works.

Measure	Description	Reference
Conductance	The percentage of edges that cross the cluster border	[58,87]
Expansion	The number of edges that cross the community border	[58]
Internal density	The ratio of edges within the cluster w.r.t. all possible edges	[58]
Cut/normalized cut	The fraction of all possible edges leaving the cluster	[26,74]
Maximum/average ODF	The maximum/average fraction of nodes' edges crossing the cluster border	[23]
Flake ODF	The fraction of nodes involved in fewer edges within the community than outside it	[23]
Volume	The sum of degrees of nodes in the community	

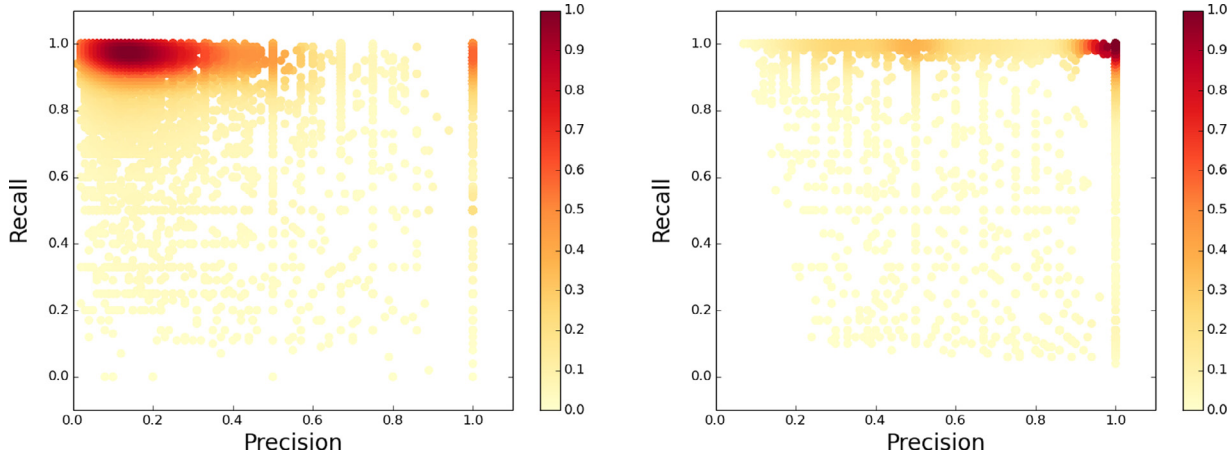

Fig. 8. *F1-community* scatter plot: inforhiermap (left) vs. Demon (right) on DBLP. Each point represents the (*precision, recall*) pairs for a given community match: the deeper the color the more the communities sharing the same values of (*precision, recall*). Points in the upper right corner (precision and recall equals to 1) identify perfect matches: communities having high precision and low recall (bottom-right corner) underestimate the ground-truth ones, communities having low precision and high recall (top-left corner) overestimate the ground truth ones.

In order to cope with the high computational complexity of such method in [69,70] we introduced the *F1-community* score.[3] In such work we address the evaluation problem as a classification task:

(i) network nodes are labeled according to their ground-truth community;
(ii) each community identified by the tested algorithm is matched to the ground-truth one whose label is shared by the majority of its nodes;
(iii) precision and recall scores are computed for each community by considering observed nodes and expected ones;
(iv) the *F1-community* score is calculated as the average of the harmonic mean of precision and recall of the matched communities.

Such approach requires $O(z)$ to be computed and allows to graphically visualize the performances of a given algorithm via density scatter plots: as an example in Fig. 8 is shown the comparison among communities identified by two different methods on the same dataset having ground-truth communities. In [63,69], in order to cope with community overlap and redundancy, we introduced a normalized version of the *F1-community* score, namely *NF1*.

5.1.2. Synthetic benchmarks

Generally, network datasets come without an explicit ground truth community annotation. For such reason, during the last decade, several synthetic network generators with tunable structure embedded communities have been proposed. The aim of synthetic models [5,61,84,86] is to provide an understanding of the dynamics of network formation and evolution. Among them, the most famous benchmarks used to assess the performances of Community Discovery algorithm are LFR [37,38] and Girvan–Newman, GN [28].

However, one relevant aspect of real-world networks have been ignored by such benchmarks so far: community dynamics. To provide a coherent environment to test both CD and DCD algorithms, we proposed RDyn [63], a flexible network generator able to simulate not only interaction dynamics but also community ones.

RDyn[4] allows tuning the generated topology and related dynamics through the introduction of several controlling variable. It describes network evolution as an iterative process composed of three main components, (i) degree/community size distribution configuration, (ii) dynamic network generation and (iii) community event generation. We designed RDyn as an iterative process (an example of its execution in Fig. 9): the topologies it generates are the results of subsequent choices made by the graph nodes that, during each iteration, are allowed to perform a specified set of actions (i.e., create/destroy edges). Moreover, once completed each iteration the status of the resulting communities is evaluated and returned if considered stable, e.g., if the partition satisfies a specified quality threshold (modularity, conductance, density...): then community dynamics are planted. Nodes act as independent agents in a social network: each one of them can decide when, how and with whom to establish or broke an interaction as well as for how long. The rationale behind RDyn is reported in Algorithm 5. RDyn takes five parameters: (i) a set of nodes, (ii) the probability of interaction renewal, (iii) the probability that an interaction will appear within (across) a community, (iv) the minimum quality required to define a community *stable*, and (v) the quality function used to evaluate the planted partition. Each of those parameters affects the topologies generated by RDyn. In particular:

[3] Code available at: https://goo.gl/9tUrcG.

[4] Code available at: https://goo.gl/WtLg4V.

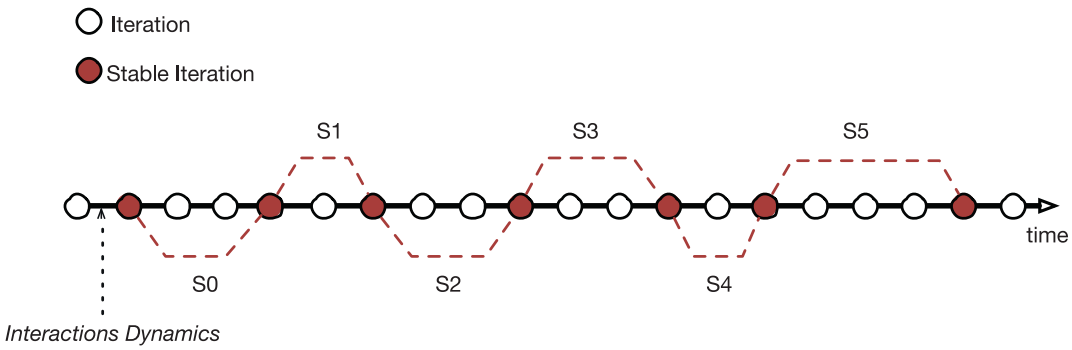

Fig. 9. RDyn execution timeline: ground-truth communities are generated only during stable iterations (black circles). Interactions between two consecutive stable iterations compose a snapshot (here identified with S0,...,S5). Interaction dynamics (as well as community ones) happens between consecutive iteration.

Algorithm 5 RDyn.

Require:
V: node set
ν: interaction renewal probability,
p_{in}: intra/inter community interaction probability,
κ: community quality threshold,
QF: quality function
Ensure:
$\mathcal{G}$: a dynamic graph,
$\mathcal{C}$: a set of, planted, evolving communities

$\mathcal{G} = (V, \emptyset)$
$d_{degree}, d_{comSize} \leftarrow$ InstantiateDistributions()
$\mathcal{C} \leftarrow$ NodeToCommunityAssignment($\mathcal{G}, d_{degree}, d_{comSize}$)
while True **do**
  for $u \in V$ **do**
    $\mathcal{G} \leftarrow$ RemoveExpiredInteractions(u, ν)
    $\mathcal{G} \leftarrow$ GenerateNewInteraction(u, p_{in})
  end for
  quality $\leftarrow$ EvaluatePartitionQuality(QF,$\mathcal{G}$, $\mathcal{C}$)
  if quality $\geq \kappa$ **then**
    yield $\mathcal{G}$, $\mathcal{C}$
    $\mathcal{C} \leftarrow$ GenerateCommunityEvents($\mathcal{G}$, $\mathcal{C}$)
  end if
end while

- The node set, V, defines the size of the graph;
- varying the interaction removal probability we can tune the overall dynamic graph stability, moving from extremely volatile structures, $\nu = 0$, to almost static ones, $\nu = 1$;
- varying the inter/intra community probability, p_{in}, we can act both on the degree of separation among network clusters and on their relative density;
- finally, varying the community quality function and threshold, QF and κ, we can take control on the kind of mesoscale topology will be planted in the graph and decide when a programmed community event (either split or merge) completed.

As an example, Fig. 10 shows the quality, computed in terms of *NF1*, of CD partitions extracted from RDyn generated graphs by both static (Louvain [7], Infohiermap [71], Demon [16]) and dynamic (iLCD [11], Tiles [69], D-GT [2]) Community Discovery approaches. The proposed example underlines how, for the chosen parameters, Infohiermap and iLCD are, respectively, the best performing algorithms among static and dynamic ones. Such results are primarily due to: (i) the measure used to evaluate community quality (density), and (ii) the fact that RDyn generates crisp, nonoverlapping, communities. The adoption of density as quality function works in favor of iLCD since, due to its definition, it searches for compact communities: the definition of planted crisp communities, on the other hand, privilege those approaches (like Inforhiermap) that search for well separated mesoscale topologies. This latter peculiarity is also the main reason for which, in Fig. 10, the compared dynamic algorithms perform worst than the static ones: indeed, all the former search for overlapping communities.

Moreover, the analytical studies performed in [63] shows that RDyn offers a more challenging benchmark than LFR and GN. Moreover, the possibility to specify a custom quality function for embedded communities (density in case of Fig. 10) enables RDyn to simulate different topological contexts, thus modeling several social network realities.

5.2. Internal evaluation

Internal evaluation methodologies, conversely from the previously discussed ones, focus on the inspection and description of the identified topologies as well as on the assessment of approach complexity and scalability.

The adoption of internal evaluation strategies is often made to overcome the issue of determining a qualitative ranking among different algorithms by proposing a quantitative evaluation. Instead of comparing the obtained partition with a ground-truth or computing quality measures, authors that exploit internal evaluation focus on the algorithmic complexity [8,69,80], running time [42,69,73], scalability [24,78,89] or analysis enabled by their approaches (i.e., identification of specific "events" [12,40,53]). *Internal* evaluation does not want to produce a direct comparison among different partitions: it assumes that each algorithm is based on a different community definition. Starting from such assumption internal evaluation strategies measure quantitative performances defining a context of applicability for the proposed algorithm. Indeed, internal evaluation techniques are often used as support for external one, as in [1,24,69]. An example of this class of evaluation methodologies, tailored to the Dynamic Community Discovery context, is the study and analysis of community events and life-cycle.

5.2.1. Dynamic community events, life-cycle tracking and stability

The vast literature on dynamic networks agrees on the basilar set of simple actions that involve entities of complex time-varying networked systems: node/edge appearance and vanishing. Those simple operations are able, as time goes by, of generating significant perturbations of the whole network topology thus affecting community structures.

Such atomic actions allow for the definition of more complex topological *events* and to a novel, interesting, problem that stands side-by-side to Community Discovery in dynamic networks: community life-cycle tracking.

The persistence along time of communities subjected to progressive changes is a significant problem while analyzing social

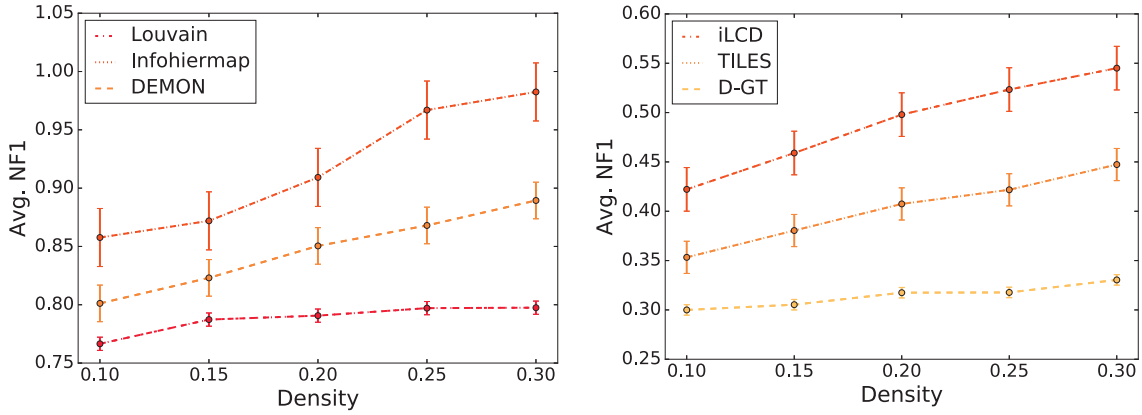

Fig. 10. RDyn benchmarks. Average *NF1* scores w.r.t. the synthetic communities (whose lower bound quality is defined in terms of density) for both static (left) and dynamic (right) Community Discovery approaches.

Table 3
Dynamic community events.

Event	Description
Birth	Identifies the first appearance of a new community
Death	Describe the vanishing of a community
Contraction	Some nodes leave the community thus decreasing its size
Growth	New nodes increase the size of a community
Merge	Two communities or more merge into a single one
Split	A community breaks into two or more components
Continue	A community remains unchanged
Resurgence	A community vanishes for a period, then comes back without significant perturbations as if it has never stopped existing

network contexts. The perturbations caused by node/edge appearance/vanishing can be used to describe a set of community transformation *events*. Such transformations, or operations, were firstly introduced in [53], which listed six of them (Birth, Death, Growth, Contraction, Merge, Split). A seventh operation, "Continue", is sometimes added to these ones, while in [13], an eighth operation was proposed (*Resurgence*). We report the complete list of community events, along with their description, in Table 3. Indeed, not all those operations are necessarily handled by a generic DCD approach and, most importantly, even if their semantics is fixed their implementation often vary from an approach to the other. This latter peculiarity, along with the heterogeneity of DCD definitions, makes impossible to directly compare community life-cycles extracted from a same dynamic graph by different DCD approaches.

Despite the particularities introduced by each approach when defining topology transformations, the identified events allow to describe for each community its so-called *life-cycle* [63,64]:

Definition 7 (Community life-cycle)**.** Given a community C, its life-cycle (which univocally identifies C's complete evolutive history) is composed of the polytree such that: (i) the roots are *Birth* events, of C and its potential predecessors if C has been implicated in *Merge* events; (ii) the leafs are *Death* events, corresponding to deaths of C and of its successors, if C has been implicated in *Split* events; (iii) the central nodes are the remaining actions of C, its successors, and predecessors. The edges of the tree represent transitions between subsequent actions in C life.

An example of community life-cycle is shown in Fig. 11. It is possible – in principle – to perform a life-cycle analysis starting from the output of any generic Dynamic Community Discovery approach.

Indeed, as illustrated by the famous Theseus' Ship Paradox, deciding if an element composed of several entities at a given instant is the same or not as another one consisting of some – or even none – of such entities at a later point is necessarily arbitrary, and cannot be answered unambiguously.

Such issue deeply affects community life-cycle analysis, especially for all those approaches where community identification is performed following a non-smoothed approach (e.g., *instant optimal* two-step algorithms). Indeed, since different DCD approaches make different assumptions on both network and community dynamics, community life-cycle tracking is used only as an *internal* evaluation strategy: a methodology to underline some characteristics of the topologies extracted by a given algorithm

As previously discussed, in [69] we introduced an online algorithm to avoid the community reidentification step: in the same work, we also provided an example of the community stability analysis. In the same work, moving from communities life-cycles, we studied Tiles' community stability through time in a specific dataset: the dynamic interaction graph built upon Sina Weibo users' conversations. Fig. 12 shows how community structure is affected by the dynamic unfolding of user interactions. In the observed scenario Tiles' communities remain stable, unchanged, as time goes for on average 40% of their nodes and, symmetrically, almost 36% of the dynamic network nodes do not change communities across consecutive observations – Fig. 12 left. Moreover, observing the unfolding of community events – Fig. 12 right – we can notice two peculiarities: (i) all events follow a same *pattern*, a pulse that characterize the evolution of the particular network, and, (ii) the sample of Sina Weibo observed describes a phenomenon in which is more likely to observe the rise and merge of communities rather than their splits (community death events may also capture community vanishing due to merging events). Indeed, such kind of evaluation represents also an example of the analytical insights that can be driven by the results of DCD approaches and the reconstruction of communities evolution history.

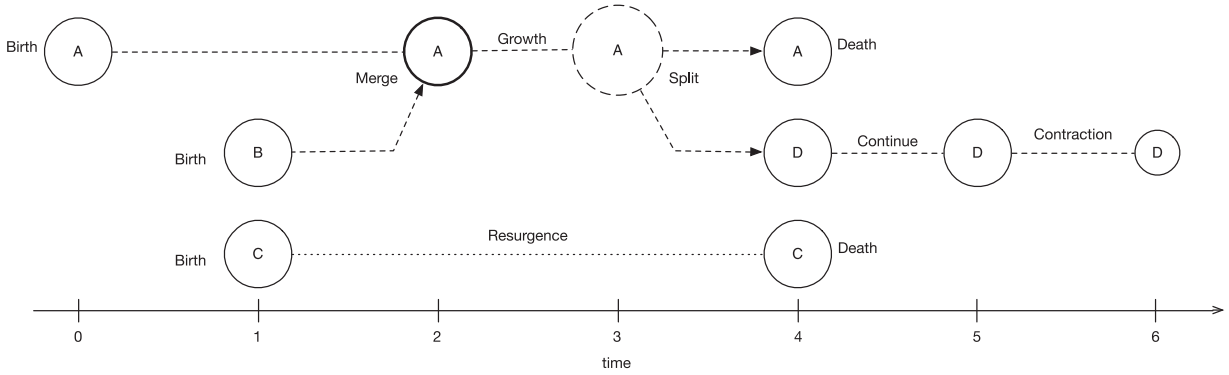

Fig. 11. Community lifecycle. As time goes by, dynamic communities experience mutations that are often abstracted by adopting a predefined set of operations. This toy example highlights lifecycles of communities *A* and *C*.

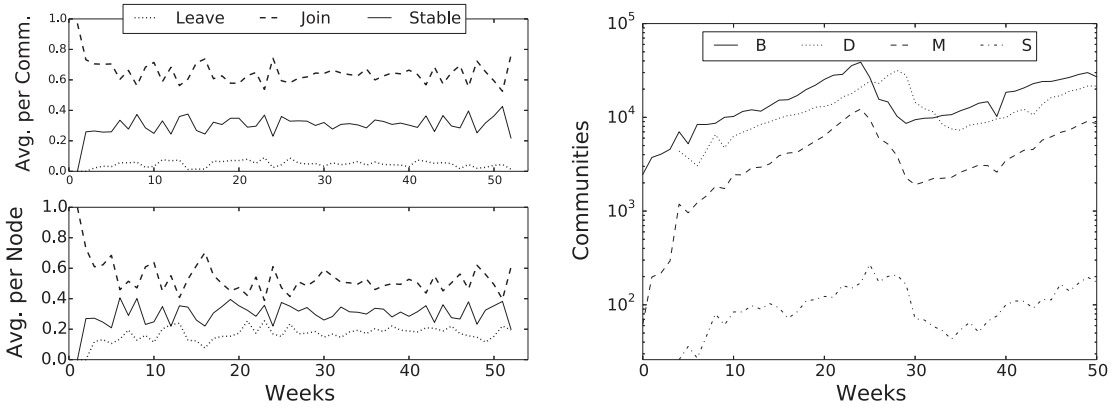

Fig. 12. Community stability and community events in Facebook. (*left*) Evolution through time of node join/leave events and (*rigth*) community events. In the right plot are reported the trends for birth (B), death (D), merge (M) and split (S) events.

6. Conclusion and future research directions

Community Discovery is one of the hottest topics in complex network analysis: indeed, countless SNA analysis and applications are nowadays built on top of mesoscale network topologies.

In this work, we discussed several different perspectives offered by CD literature, both algorithmic and analytical. Moving from classic static network analysis, we illustrated how considering the temporal dimension affects the definition and extraction of communities. We discussed a specific family of approaches, namely local bottom-up and overlapping Community Discovery, that best suits the analysis of OSNs due to their node-centric approach as well as their scalability. Finally, we showed some example of analytical tasks that benefit from such kind of methods and discuss evaluation strategies that are commonly adopted to assess community quality. All the results we proposed, framed in their relevant literature, represent our recent contribution to the Community Discovery field.[5]

Indeed, especially in the dynamic network scenario, several relevant lines of research remain open. Moving from the tracking of communities life-cycles one complex issue to address regards the forecasting of future community events. Indeed, being able to track the evolutive history of a social network and its components is just the first step: being able to leverage such information to accurately predict what will be the future shape of an analyzed social tissue is of paramount importance. Several applications, from peer-to-peer load balancing through the study of diffusive phenomena, can greatly benefit from a preemptive analysis that allows anticipating the system needs. Moreover, a very relevant topic for SNA is the study of the relations among dynamics *of* and *on* networks. Correlate the topological evolution, as well as community life-cycle, with spreading processes on social networks is a fascinating topic to address, a topic whose thorough analysis can open an entirely novel field of research. So far diffusive phenomena were studied prevalently assuming static social network context. Such oversimplification, however, does not allow to model the complex system carefully and, implicitly affects analytical results. Indeed, network topology evolution and diffusive processes are strongly tied and affect one another reciprocally, i.e., think for instance to the spread of a virus and to the possible containment strategies that can be applied to reduce it as well as to the perturbation it causes on the network topology once infected nodes are stably removed. Still pursuing a similar line of research, another relevant topic that keeps together dynamics of and on networks regards the analysis of competing diffusion processes: how does graph topology evolve when several diffusion processes compete for mutually exclusive resources (e.g., nodes)? Can we forecast the stable state, if any, of such complex intertwined phenomena?

As already discussed, Community Discovery is a very important, still ill-posed, problem that deeply affects the outcome of several network analytical tasks. In particular, its dynamic formulation has the potential to become a key step in data-driven investigations: for that to happen, it is indeed necessary that research effort will be dedicated not only to the definition of novel approaches but also to the investigation of sound and shared evaluation methodologies, specifically tailored for dynamic contexts.

[5] The code of our algorithms is available at: https://goo.gl/x4KrXN.

Acknowledgments

This work is funded by the European Community's H2020 Program under the funding scheme "FETPROACT-1-2014: Global Systems Science (GSS)", grant agreement #641191 CIMPLEX "Bringing CItizens, Models and Data together in Participatory, Interactive SociaL EXploratories".[6]

This work is supported by the European Community's H2020 Program under the scheme "INFRAIA-1-2014-2015: Research Infrastructures", grant agreement #654024 *"SoBigData: Social Mining & Big Data Ecosystem"*.[7]

References

[1] M.K. Agarwal, K. Ramamritham, M. Bhide, Real time discovery of dense clusters in highly dynamic graphs: identifying real world events in highly dynamic environments, Proc. VLDB Endow. 5 (10) (2012) 980–991.
[2] H. Alvari, A. Hajibagheri, G. Sukthankar, Community detection in dynamic social networks: a game-theoretic approach, in: Proceedings of the 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), IEEE, 2014, pp. 101–107.
[3] M. Amoretti, A. Ferrari, P. Fornacciari, M. Mordonini, F. Rosi, M. Tomaiuolo, Local-first algorithms for community detection., in: Proceedings of the 2016 International Workshop on Knowledge Discovery on the WEB (KDWeb), 2016.
[4] V. Arnaboldi, M. Conti, A. Passarella, R.I. Dunbar, Online social networks and information diffusion: the role of ego networks, Online Soc. Netw. Media 1 (Supplement C) (2017) 44–55, doi:10.1016/j.osnem.2017.04.001.
[5] A.-L. Barabási, R. Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512.
[6] R. Bhatt, V. Chaoji, R. Parekh, Predicting product adoption in large-scale social networks, in: Proceedings of the Nineteenth ACM International Conference on Information and Knowledge Management, ACM, 2010, pp. 1039–1048.
[7] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech. Theory Exp. 2008 (10) (2008) P10008.
[8] R. Bourqui, F. Gilbert, P. Simonetto, F. Zaidi, U. Sharan, F. Jourdan, Detecting structural changes and command hierarchies in dynamic social networks, in: Proceedings of the 2009 International Conference on Advances in Social Network Analysis and Mining (ASONAM), IEEE, 2009, pp. 83–88.
[9] N. Buzun, A. Korshunov, V. Avanesov, I. Filonenko, I. Kozlov, D. Turdakov, H. Kim, Egolp: fast and distributed community detection in billion-node social networks, in: Proceedings of the 2014 IEEE International Conference on Data Mining Workshop (ICDMW), IEEE, 2014, pp. 533–540.
[10] A. Casteigts, P. Flocchini, W. Quattrociocchi, N. Santoro, Time-varying graphs and dynamic networks, Int. J. Parallel Emergent Distrib. Syst. 27 (5) (2012) 387–408.
[11] R. Cazabet, F. Amblard, C. Hanachi, Detection of overlapping communities in dynamical social networks, in: Proceedings of the Second IEEE International Conference on Social Computing (SocialCom), IEEE, 2010, pp. 309–314.
[12] R. Cazabet, H. Takeda, M. Hamasaki, F. Amblard, Using dynamic community detection to identify trends in user-generated content, Soc. Netw. Anal. Min. 2 (4) (2012) 361–371.
[13] R. Cazabet, G. Rossetti, F. Amblard, Dynamic community detection, in: Encyclopedia of Social Network Analysis and Mining, Springer, 2014, pp. 404–414.
[14] Y. Cohen, D. Hendler, A. Rubin, Node-centric detection of overlapping communities in social networks, in: Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), IEEE, 2016, pp. 1384–1385.
[15] M. Coscia, F. Giannotti, D. Pedreschi, A classification for community discovery methods in complex networks, Stat. Anal. Data Min. 4 (5) (2011) 512–546.
[16] M. Coscia, G. Rossetti, F. Giannotti, D. Pedreschi, Demon: a local-first discovery method for overlapping communities, in: Proceedings of the Eighteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2012, pp. 615–623.
[17] M. Coscia, G. Rossetti, F. Giannotti, D. Pedreschi, Uncovering hierarchical and overlapping communities with a local-first approach, ACM Trans. Knowl. Discov. Data (TKDD) 9 (1) (2014) 6.
[18] J. Creusefond, T. Largillier, S. Peyronnet, On the evaluation potential of quality functions in community detection for different contexts, in: Proceedings of the 2016 International Conference and School on Network Science, Springer, 2016, pp. 111–125.
[19] I. Derényi, G. Palla, T. Vicsek, Clique percolation in random networks, Phys. Rev. Lett. 94 (16) (2005) 160202.
[20] P. Domingos, M. Richardson, Mining the network value of customers, in: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2001, pp. 57–66.
[21] A. Epasto, S. Lattanzi, R. Paes Leme, Ego-splitting framework: from non-overlapping to overlapping clusters, in: Proceedings of the Twenty-third ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2017, pp. 145–154.
[22] A. Esuli, F. Sebastiani, Sentiment quantification, IEEE Intell. Syst. 25 (4) (2010) 72–79.
[23] G.W. Flake, S. Lawrence, C.L. Giles, Efficient identification of web communities, in: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2000, pp. 150–160.
[24] F. Folino, C. Pizzuti, An evolutionary multiobjective approach for community discovery in dynamic networks, Trans. Knowl. Data Eng. 26 (8) (2014) 1838–1852.
[25] G. Forman, Quantifying counts and costs via classification, DMKD 17 (2) (2008) 164–206.
[26] S. Fortunato, Community detection in graphs, Phys. Rep. 486 (3) (2010) 75–174.
[27] S. Fortunato, M. Barthélemy, Resolution limit in community detection, Proc. Natl. Acad. Sci. 104 (1) (2007) 36–41.
[28] M. Girvan, M.E. Newman, Community structure in social and biological networks, Proc. Natl. Acad. Sci. 99 (12) (2002) 7821–7826.
[29] N.Z. Gong, W. Xu, L. Huang, P. Mittal, E. Stefanov, V. Sekar, D. Song, Evolution of social-attribute networks: measurements, modeling, and implications using Google+, in: Proceedings of the 2012 ACM Conference on Internet Measurement Conference, ACM, 2012, pp. 131–144.
[30] S. Gregory, An algorithm to find overlapping community structure in networks, Proceedings of the Eleventh European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2007), 2007, pp. 91–102.
[31] P. Holme, J. Saramäki, Temporal networks, Phys. Rep. 519 (3) (2012) 97–125.
[32] R. Kanawati, LICOD: leaders identification for community detection in complex networks, in: Proceedings of the 2011 IEEE Third International Conference on Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third International Conference on Social Computing (SocialCom), IEEE, 2011, pp. 577–582.
[33] R. Kanawati, Seed-Centric Approaches for Community Detection in Complex Networks, Springer International Publishing, pp. 197–208. 10.1007/978-3-319-07632-4-19
[34] H. Kanezashi, T. Suzumura, An incremental local-first community detection method for dynamic graphs, in: Proceedings of the 2016 IEEE International Conference on Big Data (Big Data), IEEE, 2016, pp. 3318–3325.
[35] L. Kovanen, M. Karsai, K. Kaski, J. Kertész, J. Saramäki, Temporal motifs in time-dependent networks, J. Stat. Mech. Theory Exp. 2011 (11) (2011) P11005.
[36] J.M. Kumpula, M. Kivelä, K. Kaski, J. Saramäki, Sequential algorithm for fast clique percolation, Phys. Rev. E 78 (2) (2008) 026109.
[37] A. Lancichinetti, S. Fortunato, Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities, Phys. Rev. E 80 (1) (2009) 016118.
[38] A. Lancichinetti, S. Fortunato, F. Radicchi, Benchmark graphs for testing community detection algorithms, Phys. Rev. E 78 (4) (2008) 046110.
[39] A. Lancichinetti, S. Fortunato, J. Kertész, Detecting the overlapping and hierarchical community structure in complex networks, New J. Phys. 11 (3) (2009) 033015.
[40] P. Lee, L.V. Lakshmanan, E.E. Milios, Incremental cluster evolution tracking from highly dynamic network data, in: Proceedings of the Thirtieth International Conference on Data Engineering (ICDE), IEEE, 2014, pp. 3–14.
[41] S. Lee, L.E. Rocha, F. Liljeros, P. Holme, Exploiting temporal network structures of human interaction to effectively immunize populations, PloS One 7 (5) (2012) e36439.
[42] X.-L. Li, A. Tan, S.Y. Philip, S.-K. Ng, Ecode: event-based community detection from social networks, in: Proceedings of the 2011 International Conference on Database Systems for Advanced Applications (DASFAA), Springer, 2011, pp. 22–37.
[43] L. Ma, H. Huang, Q. He, K. Chiew, J. Wu, Y. Che, GMAC: a seed-insensitive approach to local community detection, in: Proceedings of the 2013 International Conference on Data Warehousing and Knowledge Discovery, Springer, 2013, pp. 297–308.
[44] L. Ma, H. Huang, Q. He, K. Chiew, Z. Liu, Toward seed-insensitive solutions to local community detection, J. Intell. Inf. Syst. 43 (1) (2014) 183.
[45] A.F. McDaid, D. Greene, N. Hurley, Normalized mutual information to evaluate overlapping community finding algorithms, arXiv preprint:1110.2515 (2011).
[46] L. Milli, A. Monreale, G. Rossetti, F. Giannotti, D. Pedreschi, F. Sebastiani, Quantification trees, in: Proceedings of the Thirteenth IEEE International Conference on Data Mining (ICDM), IEEE, 2013, pp. 528–536.
[47] L. Milli, A. Monreale, G. Rossetti, D. Pedreschi, F. Giannotti, F. Sebastiani, Quantification in social networks, in: Proceedings of the 2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA), IEEE, 2015, pp. 1–10.
[48] F. Moradi, T. Olovsson, P. Tsigas, A local seed selection algorithm for overlapping community detection, in: Proceedings of the 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), IEEE, 2014, pp. 1–8.
[49] M.E. Newman, Fast algorithm for detecting community structure in networks, Phys. Rev. E 69 (6) (2004) 066133.
[50] M.E. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2) (2004) 026113.
[51] R.J. Oentaryo, E.-P. Lim, D. Lo, F. Zhu, P.K. Prasetyo, Collective churn prediction in social network, in: Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012), IEEE Computer Society, 2012, pp. 210–214.

[6] CIMPLEX: https://www.cimplex-project.eu.
[7] SoBigData: http://www.sobigdata.eu.

[52] G. Palla, I. Derényi, I. Farkas, T. Vicsek, Uncovering the overlapping community structure of complex networks in nature and society, Nature 435 (7043) (2005) 814–818.
[53] G. Palla, A.-L. Barabási, T. Vicsek, Quantifying social group evolution, Nature 446 (7136) (2007) 664–667.
[54] S. Papadimitriou, J. Sun, C. Faloutsos, P. Yu, Hierarchical, parameter-free community discovery, Mach. Learn. Knowl. Discov. Databases (2008) 170–187.
[55] L. Peel, D.B. Larremore, A. Clauset, The ground truth about metadata and community detection in networks, Sci. Adv. 3 (5) (2017) e1602548.
[56] D. Pennacchioli, G. Rossetti, L. Pappalardo, D. Pedreschi, F. Giannotti, M. Coscia, The three dimensions of social prominence, in: Proceedings of the 2013 International Conference on Social Informatics, Springer, 2013, pp. 319–332.
[57] G. Petkos, S. Papadopoulos, Y. Kompatsiaris, Social circle discovery in ego-networks by mining the latent structure of user connections and profile attributes, in: Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), IEEE, 2015, pp. 880–887.
[58] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, D. Parisi, Defining and identifying communities in networks, Proc. Natl. Acad. Sci. U.S.A. 101 (9) (2004) 2658–2663.
[59] U.N. Raghavan, R. Albert, S. Kumara, Near linear time algorithm to detect community structures in large-scale networks, Phys. Rev. E 76 (3) (2007) 036106+, doi:10.1103/PhysRevE.76.036106.
[60] M.J. Rattigan, M. Maier, D. Jensen, Graph clustering with network structure indices, in: Proceedings of the Twenty-fourth International Conference on Machine Learning, ACM, 2007, pp. 783–790.
[61] A. Rényi, P. Erdos, On random graphs, Publ. Math. 6 (290–297) (1959) 5.
[62] Y. Richter, E. Yom-Tov, N. Slonim, Predicting customer churn in mobile networks through analysis of social groups, in: Proceedings of the 2010 SIAM International Conference on Data Mining, SIAM, 2010, pp. 732–741.
[63] G. Rossetti, RDYN: graph benchmark handling community dynamics, J. Complex Netw. (2017).
[64] G. Rossetti, R. Cazabet, Community discovery in dynamic networks: a survey, arXiv preprint:1707.03186 (2017).
[65] G. Rossetti, R. Guidotti, D. Pennacchioli, D. Pedreschi, F. Giannotti, Interaction prediction in dynamic networks exploiting community discovery, in: Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015, ACM, 2015, pp. 553–558.
[66] G. Rossetti, L. Pappalardo, R. Kikas, D. Pedreschi, F. Giannotti, M. Dumas, Community-centric analysis of user engagement in Skype social network, in: Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015, ACM, 2015, pp. 547–552.
[67] G. Rossetti, R. Guidotti, I. Miliou, D. Pedreschi, F. Giannotti, A supervised approach for intra-/inter-community interaction prediction in dynamic social networks, Soc. Netw. Anal. Min. 6 (1) (2016) 86.
[68] G. Rossetti, L. Pappalardo, R. Kikas, D. Pedreschi, F. Giannotti, M. Dumas, Homophilic network decomposition: a community-centric analysis of online social services, Soc. Netw. Anal. Min. J. 6 (103) (2016).
[69] G. Rossetti, L. Pappalardo, D. Pedreschi, F. Giannotti, Tiles: an online algorithm for community discovery in dynamic social networks, Mach. Learn. 106 (8) (2016) 1213–1241.
[70] G. Rossetti, L. Pappalardo, S. Rinzivillo, A novel approach to evaluate community detection algorithms on ground truth, in: Complex Networks VII, Springer, 2016, pp. 133–144.
[71] M. Rosvall, C.T. Bergstrom, Maps of random walks on complex networks reveal community structure, Proc. Natl. Acad. Sci. 105 (4) (2008) 1118–1123.
[72] A.E. Sarıyüce, B. Gedik, G. Jacques-Silva, K.-L. Wu, Ü.V. Çatalyürek, SONIC: streaming overlapping community detection, Data Min. Knowl. Discov. 30 (4) (2016) 819–847.
[73] J. Shao, Z. Han, Q. Yang, Community detection via local dynamic interaction, arXiv preprint arXiv:1409.7978 (2014).
[74] J. Shi, J. Malik, Normalized cuts and image segmentation, Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 888–905.
[75] S. Soundarajan, J.E. Hopcroft, Use of local group information to identify communities in networks, ACM Trans. Knowl. Discov. Data (TKDD) 9 (3) (2015) 21.
[76] H. Sun, J. Liu, J. Huang, G. Wang, X. Jia, Q. Song, LinkLPA: a link-based label propagation algorithm for overlapping community detection in networks, Comput. Intell. 33 (2) (2017) 308–331.
[77] L. Tabourier, A.-S. Libert, R. Lambiotte, Predicting links in ego-networks using temporal information, EPJ Data Sci. 5 (1) (2016) 1.
[78] B. Tan, F. Zhu, Q. Qu, S. Liu, Online community transition detection, in: Proceedings of the 2014 International Conference on Web-Age Information Management, Springer, 2014, pp. 633–644.
[79] L. Tang, H. Gao, H. Liu, Network quantification despite biased labels, in: Proceedings of the Eighth Workshop on Mining and Learning with Graphs (MLG 2010),
[80] C. Tantipathananandh, T. Berger-Wolf, D. Kempe, A framework for community identification in dynamic social networks, in: Proceedings of the Thirteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2007, pp. 717–726.
[81] T. Viard, M. Latapy, C. Magnien, Computing maximal cliques in link streams, Theor. Comput. Sci. 609 (2016) 245–252.
[82] Q. Wang, Détection de Communautés Recouvrantes Dans Des Réseaux de Terrain Dynamiques, Lyon, École Normale Supérieure, 2012 Ph.D. thesis.
[83] Y. Wang, B. Wu, X. Pei, COMMtracker: a core-based algorithm of tracking community evolution, Adv. Data Min. Appl. 8 (2008) 229–240.
[84] D.J. Watts, S.H. Strogatz, Collective dynamics of small-worldnetworks, Nature 393 (6684) (1998) 440–442.
[85] J.J. Whang, D.F. Gleich, I.S. Dhillon, Overlapping community detection using neighborhood-inflated seed expansion, IEEE Trans. Knowl. Data Eng. 28 (5) (2016) 1272–1284.
[86] J. Yang, J. Leskovec, Structure and overlaps of ground-truth communities in networks, Trans. Intell. Syst. Technol. (TIST) 5 (2) (2014) 26.
[87] J. Yang, J. Leskovec, Defining and evaluating network communities based on ground-truth, Knowl. Inf. Syst. 42 (1) (2015) 181–213.
[88] J.-X. Yang, X.-D. Zhang, Finding overlapping communities using seed set, Phys. A Stat. Mech. Appl. 467 (2017) 96–106.
[89] A. Zakrzewska, D.A. Bader, A dynamic algorithm for local community detection in graphs, in: Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), ACM, 2015, pp. 559–564.