Applied Network Science

RESEARCH Open Access

ANGEL: efficient, and effective, node-centric community discovery in static and dynamic networks

Giulio Rossetti

Correspondence:
giulio.rossetti@isti.cnr.it
KDD-Lab, ISTI-CNR, Pisa, Italy

Abstract
Community discovery is one of the most challenging tasks in social network analysis. During the last decades, several algorithms have been proposed with the aim of identifying communities in complex networks, each one searching for mesoscale topologies having different and peculiar characteristics. Among such vast literature, an interesting family of Community Discovery algorithms, designed for the analysis of social network data, is represented by overlapping, node-centric approaches. In this work, following such line of research, we propose ANGEL, an algorithm that aims to lower the computational complexity of previous solutions while ensuring the identification of high-quality overlapping partitions. We compare ANGEL, both on synthetic and real-world datasets, against state of the art community discovery algorithms designed for the same community definition. Our experiments underline the effectiveness and efficiency of the proposed methodology, confirmed by its ability to constantly outperform the identified competitors.

Keywords: Complex networks, Dynamic networks, Community discovery

Introduction

Community discovery (henceforth CD), the task of decomposing a complex network topology into meaningful node clusters, is one of the oldest and most discussed problems in complex network analysis (Coscia et al. 2011; Fortunato 2010). One of the main reasons behind the attention it has received during the last decades lies in its intrinsic complexity, strongly tied to its overall ill-posedness. Indeed, complex networks researchers agree that it is not possible to provide a single and unique formalization that covers all the possible characteristics a community partition may satisfy. Usually, every CD approach is designed to provide a different point of view on how to partition a graph: in this scenario, the solutions proposed by different authors were often proven to perform well when specific assumptions can be made on the analyzed topology. Nonetheless, decomposing a complex structure in a set of meaningful components represents *per se* a step required by several analytical tasks. Such peculiarity has lead to the definition of several "meta" community definitions, often tied to specific analytical needs. For instance, classic works

© The Author(s). 2020 **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

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. A general, high-level, formulation of the Community Discovery problem definition is the following:

Definition 1 (Community Discovery (CD)) *Given a network G, a community C is defined as a set of nodes in G: $C = \{v_1, v_2, \ldots, v_n\}$. The community discovery problem aims to identify the set $\mathcal{C}$ of all the communities in G.*

The absence of a unique, well-posed, definition of what a community in a complex network should represent is only one of the issues to face when approaching network clustering. Indeed, the evolution through time of a network topology plays a major role in the way communities can be defined and extracted. Even though the CD problem has been classically studied considering the underlying network topology as "*frozen in time*", recently a novel branch of research addressed the problem of studying the dependant evolution of networks and their communities. Complex networks are often used to model dynamic objects – e.g., social phenomena, economic transactions, human mobility – composed by nodes and edges that may appear and vanish as time goes by. When considering this temporally enriched scenario, we need to revise the formulation of the classical Community Discovery problem. We will then talk of Dynamic Community Discovery (henceforth referred as DCD), a problem that can be defined by abstracting the specific CD definition as done in Rossetti and Cazabet (2018):

Definition 2 (Dynamic Community Discovery (DCD)) *Given a dynamic network DG, a Dynamic Community DC is defined as a set of (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.*

Both proposed problem meta-definitions allow multiple solutions to the network clustering under different constraints. As an example, such definitions do not explicitly require complete coverage of the nodes, nor specify if the identified clustering represents a neat nodes partition or, instead, a cover (thus allowing overlaps among communities). In this work, we introduce a CD algorithm, ANGEL, tailored to extract overlapping communities from a complex network. Our approach is primarily designed for social networks analysis and belongs to a well-known subfamily of Community Discovery approaches often identified by the keywords *bottom-up* and *node-centric* (Rossetti et al. 2017b). ANGEL aims to provide a fast way to compute reliable overlapping network partitions in the absence of topology dynamics. However, as we underlined, the unfolding of time plays a significant role in the structures describing social phenomena. To cope with such intrinsic evolution, we leverage ANGEL to design a simple Dynamic Community Discovery approach that can be used to track dynamic communities and their life-cycles. Both the proposed approaches focus on lowering the computational complexity of existing methods proposing scalable sequential – although, easily parallelizable – solutions to a very demanding task: overlapping network decomposition.

The paper is organized as follows. "Related works" section covers the relevant literature on community discovery needed to frame the proposed approach. In "ANGEL: static community discovery" section we introduce our static node-centric algorithm, ANGEL. There we discuss its rationale, the properties it holds as well as its computational complexity. In "ANGEL evaluation" section we evaluate the proposed method on both synthetic and real-world datasets for which ground truth communities are known in advance. To better discuss the resemblance of ANGEL partitions to ground truth ones as well as its execution times, we compare the proposed method with state-of-art competitors sharing the same rationale. In "ANGEL on dynamic networks" section we introduce the extension of ANGEL we designed to cope with dynamic network topologies. There we frame the proposed method in its general class and discuss its computational complexity. In "Dynamic ANGEL evaluation" section, as done for ANGEL, we evaluate its extension on both synthetic benchmarks and real-world dynamic networks. To do so, the concept of community life-cycle is introduced, and qualitative analysis of community event trends is performed. Finally, "Conclusion" section concludes the paper.

Related works

Community discovery is a widely discussed and studied problem. Researchers continuously propose novel approaches with the aim of solving specific declinations of this complex, and ill-posed, problem. Due to the massive literature available in this field, several attempts were made to organize and cluster methods identifying some common grounds. Among the others, the surveys of Fortunato (2010) and Fortunato and Hric (2016) and Coscia (Coscia et al. 2011) propose complete, detailed and extensive taxonomies for classic algorithms. However, due to the intrinsic complexity of the problem, several thematic surveys emerged, each focusing on a different declination (for instance considerint overlapping (Xie et al. 2013), directed (Malliaros and Vazirgiannis 2013), node-centric (Rossetti et al. 2017b) as well as dynamic community discovery (Cazabet et al. 2017; Rossetti and Cazabet 2018)).

Static Community Discovery. The algorithmic solutions we propose share a very specific goal: identify overlapping network partitions following a bottom-up, node-centric, strategy. Such an approach is often adopted while analyzing social network contexts (Rossetti et al. 2015; 2016; Milli et al. 2015), scenarios in which it is important to take into account the individual perspective on their local communities. Most importantly, in social scenarios, neat partitions are rarely semantically coherent or easily identifiable. Following such a rationale, ANGEL leverages individual ego-networks to access the node-centric perspective of the analyzed social graph. The growing availability of social media data has indeed allowed for extensive studies of such ego-centered topologies: among them in (Arnaboldi et al. 2017) Facebook and Twitter datasets were studied to relate online and offline properties of ego-networks. Such procedure, originally proposed in Coscia et al. (2012) and Coscia et al. (2014a) were also extended to parallel implementations, as in Amoretti et al. (2016), and generalized in a high-level framework (Soundarajan and Hopcroft 2015). Moreover, several approaches leverage the concept of ego-network to design heterogeneous community definitions (Epasto et al. 2017; Buzun et al. 2014). Other common strategies to design node-centric approaches, avoiding the use of ego-networks, are the seed set expansion (Moradi et al. 2014;

Whang et al. 2016), and community diffusion ones (Kumpula et al. 2008; Raghavan et al. 2007). The former decompose the community detection into two steps: identification of the seed nodes and definition of an iterative rule that describe how they attract nodes to form communities around them. Conversely, the latter let each node in the graph to autonomously chose its community by observing the choices made by its neighborhood. A classic example of this family of approaches is offered by the Label Propagation algorithm used by ANGEL to identify local communities (Raghavan et al. 2007).

Dynamic Community Discovery. Indeed, a significant number of systems can be modeled as temporal networks: cellular processes, social communications, large infrastructures (i.e., call graphs and web graphs) posses both network and temporal aspects that make them a perfect fit for dynamic network modeling. One of the first works underlining the needs for a dedicated framework for analyzing evolving network structure is indeed (Holme and Saramäki 2012). Several formalisms for representing evolving networks have been proposed to support the definition of such revised analytical framework: Temporal Networks (Holme and Saramäki 2012), Time-Varying Graphs (Casteigts et al. 2012), Interaction Networks (Rossetti et al. 2016), and Link Streams (Viard et al. 2016), to name the most famous. Leveraging such temporally enriched models novel community discovery approaches started taking into account the temporal dimension, following different strategies. In Rossetti and Cazabet (2018), three families of DCD algorithms are identified and discussed:

- *Instant-optimal CD* assumes that communities existing at t only depend on the *current* state of the network at t, as done by our dynamic ANGEL extension.
- *Temporal Trade-off CD* assumes that 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.
- Finally, *Cross-Time CD* shifts the from searching communities relevant at a particular time to searching communities relevant when considering the whole network evolution.

Within the first family are grouped several two-steps, *Identify&Match*, algorithms. The common ground of such approaches, e.g., (Palla et al. 2007; Takaffoli et al. 2011; Morini et al. 2017), is that they are easily parallelizable while suffering from some instability due to the matching phase performed as post-processing. Conversely, Temporal Trade-off approaches focus on smoothly identifying community evolutions as they happen. Such algorithms, e.g., (Cazabet et al. 2010; Zakrzewska and Bader 2015; Rossetti et al. 2017a) , are designed to deal with high-frequency node interactions, are not easily parallelizable and prone to "avalanche effects" (i.e., since they focus on local community perturbations the node groups tend, as time goes by, to increase their sizes). Finally, algorithms of the latter family search for *stable* communities across time, e.g., (Matias and Miele 2016; Mucha et al. 2010; Himmel et al. 2016). They often work upon temporal network aggregations built leveraging the complete knowledge of nodes and edges evolution. As a result, they are usually neither easily parallelizable nor applicable in online scenarios.

ALGORITHM 1: ANGEL

```
Input: $\mathcal{G} : (V, E)$, the graph; $\phi$, the merging threshold.
Output: $\mathcal{C}$ a set of overlapping communities.
1  for $v \in V$ do                                        // Step #1
2      $e \leftarrow$ EgoMinusEgo(v, $\mathcal{G}$) ;       // Step #2
3      $\mathcal{C}(v) \leftarrow$ LabelPropagation(e) ;    // Step #3
4      $\mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{C}(v)$
5  ncoms = $|\mathcal{C}|$
6  acoms = 0
7  while ncoms != acoms do                                 // Step #4
8      acoms = ncoms
9      $\mathcal{C} \leftarrow$ DecreasingSizeSorting($\mathcal{C}$) ;   // Step #5
10     for $c \in \mathcal{C}$ do
11         $\mathcal{C} \leftarrow$ PrecisionMerge(c, $\mathcal{C}$, $\phi$) ;   // Step #6
12     ncoms = $|\mathcal{C}|$
13 return $\mathcal{C}$
```

ANGEL: static community discovery

In this section, we present our bottom-up solution to the community discovery problem. In "Algorithm rationale" section we discuss the core of ANGEL[1]. Our approach follows a well-known pattern composed by two phases: i) construction of local communities moving from ego-network structures and, ii) definition of mesoscale topologies by aggregating the identified local-scale ones. Moreover, in "Properties" and "Complexity" sections we discuss the properties of the proposed algorithm and provide bounds to its complexity.

Algorithm rationale

The algorithmic schema of ANGEL is borrowed from the one firstly adopted in (Coscia et al. 2012) where the authors propose DEMON an approach whose main goal was to identify local communities by capturing individual nodes perspectives on their neighbourhoods and using them to build mesoscale ones. ANGEL follows the same rationale: however, conversely from its predecessor, it focuses on lowering the time complexity while at the same time increasing the partition quality (as will be discussed in "Complexity" section).

ANGEL starts taking as input a graph G, a merging threshold ϕ and an empty set of communities $\mathcal{C}$. The algorithm main loop cycles over each node, so to generate all the possible points of view of the network structure and guarantee complete coverage of its overall topology (Step #1 in Algorithm 1). To do so, for each node v, our algorithm applies the $EgoMinusEgo(v, \mathcal{G})$ (Step #2 in Algorithm 1) operation as defined in Coscia et al. (2014b). Such function extracts the ego-network centred in the node v – e.g., the graph induced on $\mathcal{G}$ and built upon v and its first order neighbours – then removes v from it, obtaining a novel, filtered, graph substructure. ANGEL removes v since, by definition, it is directly linked to all nodes in its ego-network, connections that would lead to noise in the identification of local communities. A single node connecting the entire sub-graph will make

[1]Code available at: https://github.com/GiulioRossetti/ANGEL and within the CDlib python library (Rossetti et al. 2019) https://github.com/GiulioRossetti/cdlib

ALGORITHM 2: PrecisionMerge

```
Input: x, a community; $\mathcal{C}$, a set of overlapping communities; $\phi$, the merging threshold.
Output: $\mathcal{C}$, a set of overlapping communities.
1 com_to_freq ← community_frequency(x) ;                         // Step #A
2 for com, freq ∈ com_to_freq do
3     if $\frac{freq}{|x|} \geq \phi$ then                                  // Step #B
4         $\mathcal{C} = \mathcal{C} - \{x, com\}$
5         $\mathcal{C} = \mathcal{C} \cup \{x \cup com\}$
6 return $\mathcal{C}$
```

all nodes very close, even if they are not in the same local community. Once obtained the ego-minus-ego graph, the next step is to compute the local communities it contains (Step #3 in Algorithm 1). The algorithm performs this step by using a CD approach borrowed from the literature: Label Propagation (LP)(Raghavan et al. 2007). This choice, already adopted in (Coscia et al. 2012), has been made for the following reasons:

1. LP is known as the least complex algorithm in the literature, reaching a quasi-linear time complexity in terms of nodes. However,
2. LP will return results of a quality comparable to more complex algorithms(Coscia et al. 2011).

Reason #1 is particularly important since Step #3 of our pseudocode needs to be performed once for every node of the network, thus making it unacceptable to spend a super-linear time for each node. Notice that instead of LP any other community discovery algorithm (both overlapping or not) can be used (impacting both on the algorithmic complexity and partition quality). Given the linear complexity (in the number of nodes of the extracted ego-minus-ego graph) of Step #3, we refer to this as the inner loop for finding the local communities. Due to the importance of LP for our approach and to shed lights on how it works, we briefly describe its classical formulation (Raghavan et al. 2007). Suppose that a node v has neighbors $v_1, v_2, ..., v_k$ and that each one of them carries a label denoting the community it belongs. Then, during each iteration, the label of v is updated to the majority label of its neighbours. As the labels propagate, densely connected groups of nodes quickly reach a consensus on a unique label. At the end of the propagation process, nodes sharing the same labels identify the resulting communities.

In case of bow-tie situations – e.g., a node having an equal maximum number of neighbors in two or more communities (example in Fig. 1) – the classic definition of the LP algorithm randomly select a single label for the contended node. ANGEL, conversely, handle this situation – that otherwise can led to nondeterministic behaviours – by allowing soft community memberships: each node can thus belong to multiple communities in case of bow-tie configuration. The result of Steps #1-3 of Algorithm 1 is a set of local communities $\mathcal{C}(v)$, according to the perspective of a specific node, v, of the network. Differently, from what done in DEMON, ANGEL does not reintroduce the ego in each local community to reduce the noisy effects hubs play during the merging step. Since local communities can be seen as a biased and partial view of the real community structure of $\mathcal{G}$, the result of ANGEL needs further processing: namely, a merging step that simplifies the local partition present in $\mathcal{C}$.

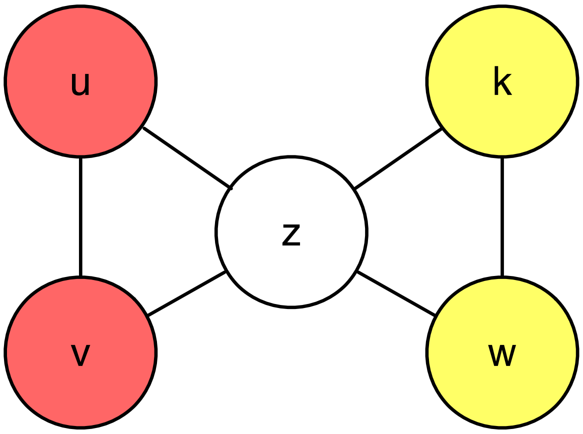


Fig. 1 Label Propagation. In the example, the node z represents the center of a bow-tie structure since it has its neighbors divide equally among two different communities ($\{u, v\}$ belonging the red one, $\{k, w\}$ to the yellow one). In such scenario the classic LP formulation will assign z to only one of the two communities after experiencing a *ping-pong* effect: this will cause nondeterminism. Our approach instead will assign z to both partitions producing a soft, but stable, community membership

Once the outer loop on the network nodes is completed, ANGEL leverages the PRECISIONMERGE function to compact the community set $\mathcal{C}$ so to avoid the presence of fully contained communities in it. Such function (Step #6, detailed in Algorithm 2) implements a deterministic merging strategy and is applied iteratively until reaching convergence (Step #4) – e.g., until the communities in $\mathcal{C}$ cannot be merged further. To assure that all the possible community merges are performed at each iteration $\mathcal{C}$ is ordered from the smallest community to the biggest (Algorithm 1, #Step 6). This merging step is a crucial one since it needs to be repeated for each one of the local communities. In DEMON such operation requires the computation for each pair of communities (x, y), $x \in \mathcal{C}(v)$ and $y \in \mathcal{C}$, of an overlap measure (i.e. Jaccard index) and to evaluate if it overcomes a user defined threshold. This approach, although valid, has a major drawback: given a community $x \in \mathcal{C}(v)$ it requires $O(|\mathcal{C}|)$ evaluations to identify its best match among its peers. Indeed, such kind of strategy represents a costly bottleneck requiring an overall $O(|\mathcal{C}|^2)$ complexity while applied to all the identified local communities. ANGEL aims to drastically reduce such computational complexity by performing the matches leveraging a greedy strategy. To do so, it proceeds in the following way:

i) ANGEL assumes that each node carries, as additional information, the identifiers of all the communities in $\mathcal{C}$ it already belongs to;
ii) in Step #A (Algorithm 2) for each local community x is computed the frequency of the community identifiers associated with its nodes;
iii) in Step #B, for each pair ($community\_id$, $frequency$) is computed its Precision w.r.t. x, namely the percentage of nodes in x that also belong to $community\_id$;
iv) iff the precision ratio is greater (or equal) than a given threshold ϕ the local community x is merged with $community\_id$: their union is added to $\mathcal{C}$ and the original communities are removed from the same set.

Operating in this way it is avoided the time expensive computation of community intersections required by Jaccard-like measures since all the containment testing can be done in place. Figure 2 shows two examples of Angel clustering of the Zachary Karate club network obtained varying the ϕ threshold. As expected, increasing the ψ threshold, we obtain a higher number of communities since lower quality merges cannot take place.

Properties

The proposed approach posses two nice properties: it produces a deterministic output (given the as input a network G and a threshold ϕ), and it allows for a parallel implementation.

Property 1 (Determinism) *There exists a unique* $\mathcal{C}$=ANGEL(G, ϕ) *for any given G and* ϕ*, disregarding the order of visit of the nodes in G.*

To prove the determinism of ANGEL it is mandatory to break its execution in two well-defined steps: (i) local community extraction and (ii) merging of local communities.

i) Local communities: Label Propagation identifies communities by applying a greedy strategy. In its classical formulation (Raghavan et al. 2007) it does not assure convergence to a stable partition due to the so-called "label *ping-pong* problem" (i.e., instability scenario primarily due to bow-tie configurations). However, as already discussed in "Algorithm rationale" section, we solved such problem relaxing the node

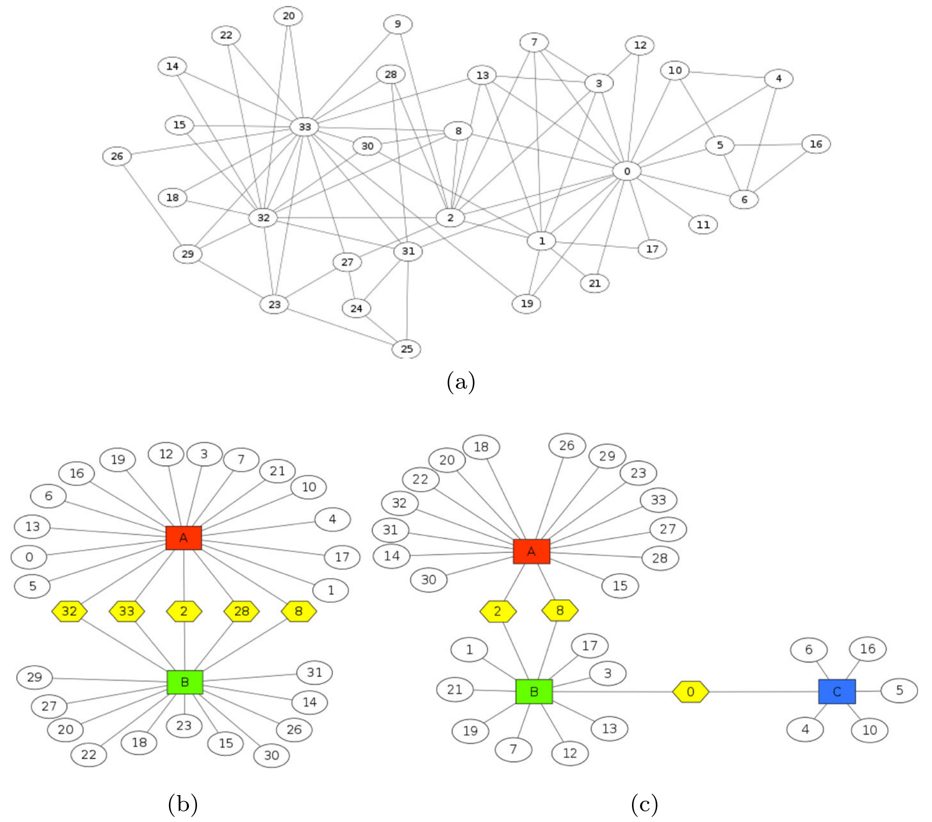


Fig. 2 Zachary Karate Club example. **a** The Zachary Karate Club network. In (**b**-**c**) the bipartite graph showing node-community affiliation (colored squares representing communities, yellow hexagons identifying nodes belonging to more than one community). **b** Clustering for $\phi = 0.33$, (**c**) Clustering for *phi* = 0.66

single label constraint thus allowing for the identification of a stable configuration of overlapping local communities.

ii) Merging: this step operates on a well-determined set of local communities on which the PRECISIONMERGE procedure is applied iteratively. Since we explicitly impose the community visit ordering the determinism of the solution is given by construction.

Property 2 (**Compositionality**) ANGEL *is easily parallelizable since the local community extraction can be applied locally on well defined subgraphs (i.e., ego-minus-ego networks).*

Given a graph $G = (V, E)$ it is possible to instantiate ANGEL local community extraction simultaneously on all the nodes $u \in V$ and then apply the PRECISIONMERGE recursively in order to reduce and compact the final overlapping partition:

$$Angel(G, \phi) = PrecisionMerge\left(\bigcup_{u \in V} LP(EgoMinusEgo(u))\right) \quad (1)$$

The underlying idea is to operate community merging only when all the local communities have already been identified (i.e., LABELPROPAGATION is applied to all the ego-minus-ego of the nodes $u \in V$ – LP(EgoMinusEgo(u)) in Eq. 1 – as shown in Fig. 3). Moreover, this parallelization schema is assured to produce the same network partition obtained by the original sequential approach due to the determinism property.

Complexity

To evaluate the time complexity we proceed by decomposing ANGEL in its main components. Given the pseudocode description provided in Algorithm 1 we can divide our approach into the following sub-procedures:

i) Outer loop (lines 3-6): the algorithm cycles over all the nodes of the network to extract the ego-minus-ego networks and identify local communities. This main loop has thus complexity $\mathcal{O}(|V|)$.

ii) Local Communities extraction: the Label Propagation algorithm has complexity $\mathcal{O}(n + m)$ (Raghavan et al. 2007), where n is the number of nodes and m is the

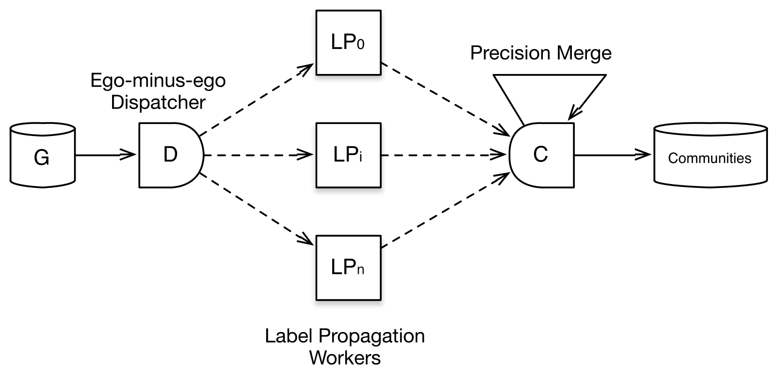


Fig. 3 ANGEL parallelization schema. The graph G is decomposed in $|V|$ ego-minus-ego network by a dispatcher D and distributed to n workers $\{LP_0, \ldots, LP_n\}$ that extract local communities from them. At the end of such parallel process, a collector C iteratively apply PRECISIONMERGE till obtaining the final overlapping partition

number of edges of the ego-minus-ego network. Let us assume that we are working with a scale free network, whose degree distribution is $p_k = k^{-\alpha}$: in this scenario the majority of the identified ego-minus-ego networks are composed by $n << |V|$ nodes and $m << |E|$ edges, thus the average complexity of each iteration will be $\mathcal{O}(n+m) << \mathcal{O}(|V|+|E|)$.

iii) PRECISIONMERGE final cycle (lines 9-14): for each local community ANGEL evaluate if it can be merged with one or more previously identified substructures. To efficiently implement this task, we assume that once identified a community a new identifier is generated and assigned to all the nodes within it. All the nodes will then have attached multiple labels (one representing an identifier of a community the node belongs to). Given a community x the PRECISIONMERGE function (Algorithm 2) leverage such information to efficiently compute – for each community identifier y attached to the nodes in x – the ratio of nodes in it that already belongs to y w.r.t. the size of x. If the ratio is greater than (or equal to) a given threshold, the merge is applied and the node label updated. This step can be performed with constant complexity employing an hash-map, $\mathcal{O}(1)$. Considering the complete loop the overall cost is thus given by the initial sorting of the communities by decreasing size, $\mathcal{O}(|C|log|C|)$ (where C is the community set), and the evaluation of PRECISIONMERGE on each community in C, $\mathcal{O}(|C|)$. Moreover, we can assume the number of iteration $k << |C|$ since at each step the number of communities decreases: thus we can consider k as a constant factor giving as final complexity, $\mathcal{O}(|C|log|C|) + \mathcal{O}(|C|) = \mathcal{O}(|C|log|C|)$.

Considered together such sub procedures gives us a final complexity of $\mathcal{O}(|V|(n+m)) + \mathcal{O}(|C|log|C|)$: considering a scale free network, for which we can reasonably expect $|V| >> (n+m)$ and $|V| > |C|$, the final complexity can be approximated as $\mathcal{O}(|V|)$.

ANGEL evaluation

Evaluating a community discovery approach is not an easy task. In this section, we propose a two-stage evaluation, focusing both on underlining ANGEL efficiency – in terms of scalability and running time – as well as on its ability to retrieve ground-truth communities. As a first step, in "Competitors and datasets" section we identify the competitors of our algorithm, approaches that share with it the same rationale. After that, in "Community resemblance" section, we briefly describe the quality function we adopt to compare the partition produced by the selected algorithms and to assess their resemblance w.r.t. ground-truth communities. Finally, we evaluate ANGEL and its competitors on two different community resemblance tasks: (i) identification of planted ground truth partition in synthetically generated networks, "Synthetic benchmarks" section, and (ii) identification of semantic communities in real-world network datasets, "Evaluation on real world data" section.

Competitors and datasets

We defined ANGEL as a two-phase bottom-up approach that leverage label propagation to extract overlapping communities. To evaluate its performances, we compare it with state-of-art competitors having a similar rationale[2]. In particular, our analysis includes:

[2]All the algorithms were executed on a Linux 4.4.0 machine with an Intel Core i7-5820 CPU @3.3GHzx16 and 32GB of RAM.

DEMON (Coscia et al. 2012; 2014b) is an incremental and limited time complexity algorithm for community discovery. It extracts ego networks, i.e., the set of nodes connected to an ego node u, and identifies the real communities by adopting a democratic, bottom-up merging approach of such structures.

PANDEMON (Amoretti et al. 2016) is a parallel implementation of DEMON designed to increase its scalability and to reduce the computational complexity of its community merging phase.

NODEPERCEPTION. In Soundarajan and Hopcroft (2015) the authors propose a generalization of the DEMON approach: NODEPERCEPTION instantiate the local two-phase schema by employing alternative community discovery approaches to Label Propagation in the local community extraction phase. Thanks to such flexibility, NODEPERCEPTION allows the final user to identify search for network partitions that optimize specific quality functions.

SLPA. In Xie and Szymanski (2012) is introduced an overlapping hierarchical community discovery algorithm designed for large-scale networks. SLPA leverages a label propagation strategy built upon dynamic interaction rules. The time complexity of SLPA scales linearly with the number of edges in the network.

The former three (DEMON, PANDEMON and NODEPERCEPTION) move from the same algorithmic schema of our approach. They all are *node-centric* algorithms (Rossetti et al. 2017b) that, moving from the analysis of ego-networks, generate overlapping partitions following a non-deterministic approach and providing different computational complexity. Conversely, the latter competitor, SLPA, represents a fast implementation of the label propagation algorithm used by ANGEL, DEMON and PANDEMON to identify ego-network local communities. Even though SLPA does not fall in the node-centric algorithmic family, we decided to include it in our analysis since it can be seen as a baseline for all those algorithms employing label propagation as the internal function.

Synthetic benchmarks. To evaluate how ANGEL behave under specific, controlled, settings we tested it, along with its competitors, against synthetic networks having planted ground truth communities generated through the LFR benchmark[3] (Lancichinetti et al. 2008). The networks described by LFR have well-known characteristics: among the others, both their node degrees and community sizes follow a power law distributions. Moreover, similar to the *planted l-partition* model(Condon and Karp 2001), LFR network vertices share a predefined fraction of their links with other vertices of their cluster. Finally, LFR allows the analyst to decide the average cluster density and size of the generated graph. We generated multiple networks varying the following LFR parameters:

- N, the network size (from 100 to 100k nodes);
- C, the network density (from 0.1 to 0.4, steps of 0.1);
- μ, the mixing coefficient describing the average per-node ratio between the number of edges to its communities and the number of edges with the rest of the network (from 0.1 to 0.5, steps of 0.1).

[3]Code available at https://sites.google.com/site/santofortunato/inthepress2

Real world data. TTo understand how ANGEL and its competitors behave on real-world data, we tested them against four network datasets having annotated ground-truth community structure[4]. We analyzed the following datasets (Yang and Leskovec 2015), whose synthetic statistics are briefly summarized in Table 1:

- **emailEU.** Network built upon email exchange data among members of a large European research institution. The ground truth communities identify members' departments.
- **Amazon.** Network built using the *Customers Who Bought This Item Also Bought* feature of the Amazon website. Each product category provided by Amazon defines each ground-truth community.
- **dblp.** Co-authorship network where two authors are connected if they publish at least one paper together. Publication venue, e.g., journal or conference, define ground-truth communities.
- **Youtube.** Subgraph of the Youtube social network. User-defined groups identify ground-truth communities.

Differently from synthetic benchmarks, where the planted communities respect specific topological characteristics, real data annotation provides a semantic partition of network nodes. Since none of the considered algorithms is parameter free in our analysis, we instantiate each one of them multiple times performing a grid-search estimation of the optimal parameter for each target network. Such parameter fitting strategy ensures that, for each network, we compare the performances of the selected algorithms leveraging their partitions that better approximate the ground truth ones.

Community resemblance

One way to asses the effectiveness of a CD algorithm is to compare how much the communities it identifies can provide a good approximation of a given ground truth partition. To quantify the degree of resemblance of two graph partitions we apply an efficient methodology proposed in Rossetti et al. (2016). Given a community set X produced by an algorithm and a ground truth community set Y, for each community $x \in X$ we label its nodes with the ground truth community $y \in Y$ they belong to. Then we match community x with the ground truth community with the highest number of labels in the algorithm community. Such procedure produces (x, y) pairs having the highest homophily between the node labels in x and all the ground truth communities. The quality of the produced mappings is estimated in terms of *precision* and *reacal*:

Table 1 Datasets statistics

	$\|V\|$	$\|E\|$	$\|C\|$	**CC**	***d***
emailEu	1,005	25,571	42	0.3994	7
Amazon	334,863	925,872	75,149	0.3967	44
dblp	317,080	1,049,866	13,477	0.6324	21
Youtube	1,134,890	2,987,624	8,385	0.0808	20

Number of nodes, edges, ground truth communities, average clustering coefficient and diameter for the analyzed datasets

[4]Datasets available at https://snap.stanford.edu/data/.

Precision: identifies the percentage of nodes in x labeled as y. It is defined as:

$$P = \frac{|x \cap y|}{|x|} \in [0, 1] \tag{2}$$

Recall: identifies the percentage of nodes in y covered by x. It is defined as:

$$R = \frac{|x \cap y|}{|y|} \in [0, 1]\,. \tag{3}$$

Given a pair (x, y) the two scores describe the overlap of their members. A perfect match is obtained when both *precision* and *recall* are equal to 1. Indeed, many-to-one mappings can occur: multiple communities in X can be connected to a single ground truth community in Y. This peculiarity allows the adoption of such methodology to evaluate both algorithms producing crisp partitions as well as approaches producing overlapping ones. In Rossetti et al. (2016), *precision* and *recall* are combined into their harmonic mean obtaining the $F1$-measure, a concise quality score for the individual pairing:

$$F1 = 2\frac{precision * recall}{precision + recall}. \tag{4}$$

Given a network, the $F1$ score can be averaged among all the identified pairs in order to summarize the overall correspondence between the algorithm community set and ground truth community set. In the following, we will adopt a normalized version of the F1 score, namely NF1[5], that mitigate the issues related to coverage and redundancy of communities in assessing the final matching quality. In particular, defined as Y_{id} the set of community of Y matched by community in X, we can define *Coverage* as:

$$Coverage = \frac{|Y_{id}|}{|Y|} \in [0, 1] \tag{5}$$

and it identify the percentage of communities in Y that are matched by at least an object of C. Redundancy instead can be defined as:

$$Redundancy = \frac{|X|}{|Y_{id}|} \in [1, +\infty) \tag{6}$$

Redundancy is minimized when no conflicting matches exist among the communities in X and the ones in Y_{id}. Finally NF1 can be defined as:

$$NF1 = \frac{F1 * Coverage}{Redundancy} \in (0, 1] \tag{7}$$

NF1 is maximized when: (i) the average F1 is maximal (perfect match), (ii) the community in X provide a complete coverage for the ones in Y and (iii) the redundancy is minimized (i.e., each community in X is matched with a distinct community in Y). As shown in Rossetti et al. (2016) it is possible to compute F1 (and thus NF1) paying a linear complexity in the size of the community set X. The reduced complexity makes NF1 a suitable alternative to the widely used NMI (Lancichinetti et al. 2008). Moreover, as discussed in Lancichinetti et al. (2009) and McDaid et al. (2011), NMI is not stable while comparing overlapping partitions with non-overlapping ones while NF1 does not suffer such limitation. Since all the compared algorithms produced overlapping partitions, NF1 represents a reasonable resemblance function to adopt. Moreover, considering our analytical setup, we avoid applying aggregate ranking solutions (as proposed in Jebabli et al. (2018) and (Dao et al. 2018)) that are designed to obtain a more comprehensive view of clustering properties. Evaluation approaches belonging to such a family generate an aggregate

[5]Code available at https://github.com/GiulioRossetti/f1-communities

ranking able to summarize the behaviors of alternative CD solutions once that multiple clustering fitness/comparison scores are available (see, for instance, Orman et al. (2012)). However, considering that all the compared algorithms are intended to operate under the same rationale, we feel that focusing on NF1 provides us enough information to draw a few considerations on the obtained clusterings.

Experimental results

Synthetic benchmarks

In Fig. 4 we report the execution time and NF1 score for the compared CD approaches. Our experiments show that our approach is able sensibly to improve the running times of its competitors while increasing the network size. In particular, it is worth noticing that Fig. 4a reports execution times on a log scale: considering the average runtime of ANGEL on the generated 100k nodes graphs it registers a speedup of an order of magnitude w.r.t. its competitors. In Fig.4b-c the NF1 score is used to compare the adherence of the partitions identified by the selected CD algorithms to the ground truth ones: we omitted NODEPERCEPTION's results since their overall NF1 were always lower than 0.4. In particular Fig. 4b compare the average NF1 scores obtained by each algorithm on different sized LFR graphs. To compute the NF1 mean value for the pair $< algorithm, network\_size >$ we considered the results provided by the optimal parameter configuration w.r.t. each network size instantiation (e.g., varying graph density and mixing coefficient). Among the compared methods ANGEL is always able to reach the highest scores, often producing the perfect match for the planted communities. Figure 4c underline the impact of network mixing coefficient on the quality of extracted communities once fixed the network size. We observe that ANGEL and SLPA can assure relatively stable performances while varying μ.

Evaluation on real world data

Table 2 shows the running times – expressed in seconds – of the compared CD approaches when applied to the selected networks. As already underlined in the synthetic scenario, ANGEL outperforms its competitors, often achieving execution times of one or more orders of magnitude less.

Differently from the synthetic scenario, when it comes to assessing community resemblance – quantitative values in Table 3 – we observe a relatively low quality for all the partitions produced by the compared algorithms. Indeed, such results are somehow

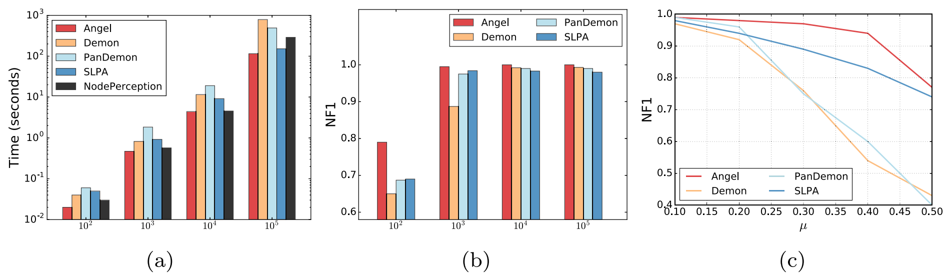


Fig. 4 Synthetic Benchmarks. **a** Running time of the compared algorithms w.r.t. network size (number of nodes); **b** Avg. community resemblance score per network size; **c** Community resemblance varying LFR mixing coefficient (number of nodes 10^6). In (**b**-**c**) the NF1 scores for NODEPERCEPTION are omitted due to their low values (NF1$\leq$0.2)

Table 2 Running times

	Angel	Demon	PanDemon	NodePerception	SLPA
emailEu	3.53	4.72	2.34	9.91	2.42
Amazon	88.49	16862.61	3032.63	256.09	504.61
dblp	115.44	24273.36	1059.54	382.43	321.46
Youtube	2209.20	8362.28	4076.98	11533.74	2860.01

The execution times reported are expressed in seconds and do not include network loading and results serialization on file. PANDEMON has been executed on 16 cores

expected. Conversely, from the synthetic benchmark where the planted communities were designed to follow specific topological characteristics, the semantic annotation provided for the analyzed real-world network do not necessarily reflect structural properties (Hric et al. 2014). Such decoupling makes difficult, if not impossible, for CD algorithms that do not leverage semantic information to capture the same partition identified by the ground truth. However, even in this more complex scenario, ANGEL communities are the ones able to better approximate the provided ground truth node partitions. In order to provide a statistical significance bound to our experiment on real data we also applyed a Friedman test (Friedman 1937) with Li post-hoc evaluation (Li 2008) on the evaluation proposed in Table 3. The test was rejected for the NF1 scores with a p-value of 0.05, thus implying that the compared methods do actually behave differently when tested on multiple datasets. Moreover, the post-hoc underlined that Angel significantly outperforms NODEPERCEPTION under the same confidence interval, and all the others when p-value is imposed equalt to 0.1.

ANGEL on dynamic networks

In this section, we propose an extension of ANGEL tailored to extract communities from dynamic network topologies so to observe their evolution as time goes by. From a modelling point of view, in the following sections, we represent a dynamic network by using snapshot graphs:

Definition 3 (Snapshot Graph) *Let G be an attributed graph $G = (V, E, T)$, where V is a set of nodes, E a set of edges and $T = \{0, 1 \ldots, n\}$ an ordered set of labels (associated to both nodes and edges) identifying different timestamps. Given a label $i \in T$, we call $G_i = (V_i, E_i)$ the graph induced from G composed by the nodes, edges whose labels is i. A Snapshot Graph is defined as the set $\mathcal{G} = \{G_0, G_1 \ldots G_n\}$ composed by n consecutive, non temporal overlapping, partition of G such as $G = \cup_{i=0}^{n} G_i$.*

Using such temporal discretization in Instant-Optimal dynamic community discovery we extend ANGEL to handle dynamic networks and briefly frame the resulting approach

Table 3 Community resemblance

	Angel	Demon	PanDemon	NodePerception	SLPA
emailEu	0.51	0.20	0.04	0.12	0.23
Amazon	0.17	0.10	0.12	0.05	0.09
dblp	0.52	0.32	0.52	0.02	0.27
Youtube	0.19	0.08	0.08	0.04	0.18

NF1 scores achieved by the compared algorithms on the real world datasets. For each model, we report the score achieved by its optimal parameter settings within a specific subclass of DCD methodologies: *Instant Optimal*, *Identify&Match* algorithms. Finally, in Complexity we discuss the computational complexity of the proposed method.

Instant-Optimal dynamic community discovery

As previously discussed, ANGEL efficiently address the classical formulation of the overlapping community discovery problem: however, *per se* it is not designed to take into account the challenges that evolving network topologies generate. The natural way to proceed, to enable ANGEL to dynamic community analysis, is to extend it by applying an algorithmic schema known as Identify&Match (or, equivalently, *"Two-steps"*(Alhajj and Jon 2014)). Such schema characterizes a vast majority of DCD approaches originating from the extension of static methods to dynamic topologies. As suggested by its name, such strategy describes a two steps process:

i) *Identify*: detect static communities on each step of evolution;
ii) *Match*: align the communities identified at step t with the ones at step $t-1$.

The main advantage of two-steps solutions is that they allow reusing static CD techniques, avoiding the definition of novel, often context dependent, methodologies. Moreover, one of the reasons for the abundance of methods belonging to such family lies in the fact that the *matching* step can be derived from existing literature, since set matching is a widely studied problem. Moreover, Identify&Match allows to easily describe parallelizable analytical workflows. As discussed in (Rossetti and Cazabet 2018), *"Two-step"* approaches represent a specialization of a more general class of algorithms, called *Instant-Optimal CD*. Indeed, matching the communities found at different stages of network evolution might involve comparing several sets of temporally disjoint network partitions: however, Instant Optimal CD approaches assume that the partition identified at t is optimal, w.r.t. the topology of the network at t. DCD solutions falling in this class are, by definition, *non-temporally smoothed* and represent 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.

Algorithm 3 details the pseudocode of the dynamic extension of ANGEL. The algorithm required inputs are (i) a set of snapshot graphs, and (ii) the ϕ threshold (as requested by ANGEL). To avoid having a third parameter (e.g., a similarity threshold for the matching phase), we don't make use of the Jaccard similarity – a widely adopted strategy to address this kind of approaches – while aligning community sets extracted from consecutive network snapshots. Conversely, we adopted a matching criterion similar to the one used to merge communities in the second phase of ANGEL, thus providing a coherent merging/matching strategy. We assume that each node at time t carries three sets of labels: i) the identifiers of the communities it currently belongs to; ii) the identifiers of the communities it was part of at $t-1$, and; iii) the identifiers of the communities it will be associated to at $t+1$. Given two community sets – i.e., the ones at time $t-1$ and t – constructing the requested labelling has linear complexity in the number of the nodes. Once the nodes belonging to temporally adjacent partitions are labelled, the following matching procedure is performed:

i) firstly, each community identified in G_{t-1} is matched with the ones in G_t that maximize the precision score;

ALGORITHM 3: Dynamic ANGEL

```
Input: $\mathcal{G}$ ={$G_0, G_1, \ldots G_n$}, a snapshot graph; $\phi$, the merging threshold.
Output: $\mathcal{L}$, a set of community lifecycles.
1 $\mathcal{L}$ = []
2 for $G_i \in \mathcal{G}$ do
3     $\mathcal{C}_\rangle \leftarrow$ ANGEL($G_i, \phi$) ;              // Community Extraction
4     $\mathcal{L} \leftarrow$ Precision($C_i$, $C_{i-1}$) ;              // Merge Detection
5     $\mathcal{L} \leftarrow \mathcal{L} \cup$ Precision($C_{i-1}$, $C_i$) ;  // Split Detection
6 return $\mathcal{L}$
```

ii) secondly, the same criterion is used to match each community in G_t with the more similar ones in G_{t-1}.

Indeed, the precision score (as defined in Equation 2) is not symmetric, thus performing the matching in both directions makes possible the identification of different evolutive patterns involving the observed communities.

Complexity

Since the algorithm core is based on the ANGEL one, it is clear that its community identification step has complexity $\mathcal{O}(|V|)$. Assuming no parallelism, an equal number of nodes in each graph snapshot, and k snapshots composing the network evolution, we get $\mathcal{O}(k|V|)$. Each merging phase, after the labelling step having complexity $\mathcal{O}(|v|)$, has complexity $\mathcal{O}(|C|)$, where C is the identified community set (as discussed in (Rossetti et al. 2016)): thus assuming k community sets of approximately the same size, we get $\mathcal{O}((k-1)|C|)$. Since we can assume that $k << |C| << |V|$ the overall complexity is $\mathcal{O}(|V|)$.

Dynamic ANGEL evaluation

Symmetrically to what we did in "ANGEL evaluation" section, here we evaluate our DCD approach both on synthetic and real-world data. Moving to a dynamic scenario two different quality functions should be taken into consideration while evaluating a CD approach: community *resemblance* and *matching* effectiveness. To assess the former, coherently to what we did for ANGEL, we will proceed leveraging the NF1 score; to evaluate the latter, on the other hand, we will analyze to what extent the community events identified by our approach are in line with the ones annotated in the data. Since community events represent a peculiarity of time evolving contexts in Community events and life-cycle we firstly introduce a high-level discussion of what is usually called *Community Lifecycle*. Finally, in Synthetic benchmarks and Evaluation on real world data we describe and discuss the results obtained on synthetic benchmarks as well as on real data. Conversely, from what previously done for ANGEL we will not compare the obtained results by its extension with the ones produced by others approaches following the same rationale: in this scenario, our focus will be on the analysis of the discovered patterns and the evaluation of the proposed matching strategy.

Moreover, so far most the *two-step* approaches proposed in literature leverage non-overlapping static CD algorithms whose definition is not aligned to the ANGEL node-centric bottom-up one. Finally, a comparison with DCD extensions of the competitors identified in "Competitors and datasets" section can only focus on the matching effectiveness since the resemblance has already been evaluated in the static scenario.

Community events and life-cycle

Several works on evolutionary community discovery focus on the analysis of the events which regulates community life-cycles – birth, merge, split, continue and death of communities. Such events are commonly described as follows:

Birth (B): a community born at time t if there are no network substructures at $t-1$ that can be matched with it;
Merge: two or more communities at time t merge iff they are matched to the same network substructure at $t+1$;
Split (S): a community at t splits if it is matched to multiple network substructures at $t+1$;
Continue (C): a community at t remains the same at $t+1$;
Death (D): a community dies at t if it is not matched with any network substructure at $t+1$.

Indeed, rarely a generic DCD algorithm is able to track all the community events introduced. However, some events can be easily reconstructed as a post-process analytical step from the results of every DCD approach.

The main open issue that affects the evaluation of community life-cycles lies in the so-called *Theseus's ship paradox*. During his adventurous journey, the Greek hero Theseus had to replace his ship piece by piece due to the adversities he had to face on the sea. Once returned home, Theseus's ship is placed in a museum as the iconic memory of the hero legacy. In its essence, the ship was still the same as the one Theseus used when he left Troezen; however, none of its original components were still part of it. Analyzing the evolution of communities leads to a similar dilemma: an evolving community can be considered as such if during his history most (if not all) the nodes that compose it change?

Indeed multiple answers can be provided to such question: in our analysis, we focus our attention on community event trends through time, without directly constructing individual community life-cycle.

Experimental results

Synthetic benchmarks

Similarly to what we have done for ANGEL, we evaluate its extension against synthetic dynamic networks having planted ground truth communities. So far few benchmarks specifically designed for the DCD problem have been proposed: among them, we recall Greene's (Greene et al. 2010; Granell et al. 2015). It composes of four dynamic networks – observed for five snapshots – each one describing a specific kind of community evolution event. In particular, Greene benchmarks aim to evaluate the ability of a DCD approach in identifying the following mesoscale topology perturbations:

- *Intermittence*. In the corresponding network, 10% of communities are removed randomly from each snapshot to its successor;
- *Expansion* and *contractions*. To examine the effect of rapid expansions and contractions of communities, in this network in each snapshot 40 randomly selected

communities, absorb new nodes or lose their former members by 25% of the previous size;

- *Birth* and *death*. At each snapshot, 40 new clusters are created by the nodes which have left their former communities. Furthermore, 40 existing communities are removed randomly;
- *Merging* and *splitting*. In this last setting, during each snapshot, 40 instances of existing clusters are merged two by two, and similarly, 40 communities split into two new communities.

Such benchmarks can be used to evaluate both community resemblance and event matching since they explicitly define both ground truth network partitions for each snapshot as well as the events relating communities of consecutive snapshots.

Indeed, Greene's benchmark is not the only one designed to test DCD approaches. In the last decade, several dynamic graph generators have been proposed to address such issues (Rossetti 2017; Bazzi et al. 2016; Lin et al. 2008), often extending/revising the LFR approach. In the following, we evaluate the proposed algorithm employing only the Greene's synthetic networks corpus since it can be seen as a *de-facto* standard, being the most widely adopted benchmark to test DCD algorithms. Figure 5 show the results that the dynamic extension of ANGEL achieves when tested against Greene's benchmark. As a first result, Fig. 5a underlines the trends for community coverage in all the described scenarios. The reported trend lines underline that ANGEL can identify most of the ground truth community of each snapshot. The partition coverage – i.e., the number of ground-truth communities matched by the ones extracted by our approach – assumes values in [0.94, 0.98]. Moreover, in Fig. 5b, we report the match quality, Ψ, of the communities involved in merge/split, birth/death, Expand and Hide events. Ψ is a function that relates the set of discovered events, $\mathcal{A}$, to the ground truth ones, $\mathcal{B}$:

$$\Psi(\mathcal{A},\mathcal{B}) = \mathcal{A} \cap \mathcal{B} - (\mathcal{A} - \mathcal{B}) \tag{8}$$

The quality function Ψ assumes values in [-$|\mathcal{A}|$, 1]: when maximized, it implies a perfect match of the two event sets when minimized it identify completely disjointed event sets, values in between capturing both the presence of partial matches and wrong ones. Our results underline that the proposed matching strategy can identify the planted events that – with the sole exception of Birth/Death in the first two matching steps – are perfectly reconstructed. Finally, in Fig. 5b we report the overall community quality score computed using the previously introduced NF1 score. Moreover, resemblance scores of ANGEL communities w.r.t. the considered benchmark assume values in [0.91, 0.95], thus

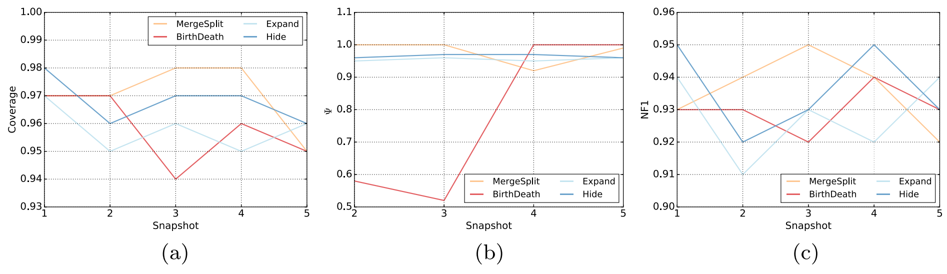


Fig. 5 Greene Benchmarks. **a** Community coverage per snapshot and graph sequence; **b** Percentage of communities matched per snapshots and graph sequence; **c** NF1 score per snapshot and graph sequence

highlighting the ability of the proposed method to extract partitions that adhere to the original ground-truth.

Evaluation on real world data

Temporally evolving networks are often used to describe real-world phenomena. Dynamic network data rarely come with annotated evolving ground-truth communities; however, we decided to test our approach on real-world datasets to provide a characterization of the partitions our approach identifies. To do so, we analyzed the following datasets[6], whose synthetic statistics are reported in Table 4:

- **Enron.** The Enron email network consists of emails sent among the employees of Enron Corporation between the beginning of 1999 and 2002. Graph snapshots are taken monthly.
- **FB-wosn.** Dynamic network built upon the wall posts from the Facebook New Orleans networks. Each edge represents a post to other users wall, and each node represents a Facebook user. The time interval between snapshots is selected one month.
- **Weibo.** This dataset is obtained from the 2012 WISE Challenge[7]. It is built upon the logs of the popular Chinese micro-blog service WEIBO[8]. Edges represent mentions made by users in short messages. We selected a single year, 2011, and used an observation window of one month to build the snapshots.

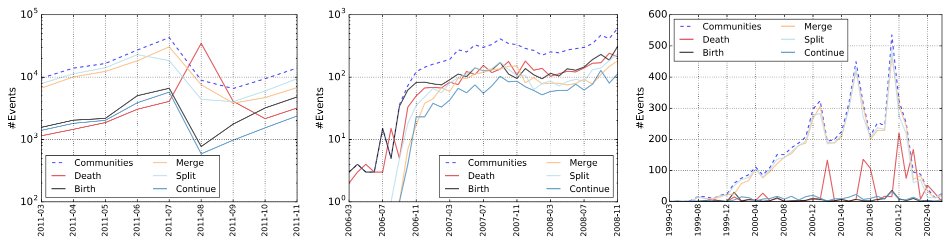
As stated, differently from the real data analyzed in the static scenario, we do not have any ground-truth annotation for the communities nor for the events they experience. To provide some insights on the results obtained, we report in Fig. 6 the trends of community events through time. Communities can be seen as the bricks of a complex network structure, the way they interact (both merging and splitting), form and dissolve, describe the *pulse* of a dynamic system. A leitmotif shared by all the networks analyzed regards the *continue* event being the least represented. As expected, and somehow postulated by the Theseus's ship paradox, it is rare that a community remains stable (i.e., composed by the same exact set of nodes) as time goes by, especially in social interaction networks like the ones we considered. Apart from this common result, the three networks seem to be characterized by different event-patterns. In Fig. 6a we can observe overall stationarity of the Weibo service: all the trends seems to maintain a predictable increasing pattern, interrupted only by a sudden change in the number of communities during August 2011. The FB-wosn data tells a completely different story. Figure 6b describe a progressive, steady, increase in the number of communities, underlying that, during each observation, the relative importance of events remains almost constant. In this scenario, the birth

Table 4 Datasets statistics

	$\|V\|$	$\|E\|$	$\|T\|$	**CC**
Enron	87,110	298,691	48	0.1196
FB-wosn	61,096	614,797	36	0.1886
Weibo	8,335,605	49,595,797	12	0.0146

The number of nodes, edges, snapshots, average clustering coefficient for the analyzed datasets. Network related statistics are computed on flattened graphs

[6]Datasets available at: http://networkrepository.com.
[7]http://www.wise2012.cs.ucy.ac.cy/challenge.html
[8]http://weibo.com.



(a) Weibo (b) FB-wosn (c) Enron

Fig. 6 Event Trends. For each dynamic network are reported the trends (in absolute numbers) of community events. Moreover, the trends of identified communities as time goes by is also shown

and death events are the most frequent ones: communities change quickly and radically from snapshot to snapshot, allowing less gradual merge/splits than in Weibo. Finally, Fig. 6c describes a very peculiar trend emerging from the emails exchanged between Enron employees. As a unique among the observed datasets, in Enron, the merge and split events are constantly the most prominent ones. Conversely, from FB-wosn and Weibo, online social platforms whose users can deeply vary among consecutive observation periods, Enron nodes are particularly stable. Node stability, as well as routinary work-related communications, reduce the likelihood of identifying the formation of completely novel communities. Indeed, changing the snapshot temporal granularity could affect such results: for instance, moving from a monthly to a daily temporal discretization in Enron generates an increase of Death/Birth events during weekends, reflecting reduced email activities.

Conclusion

In this paper, we introduced ANGEL, a node-centric approach to overlapping community discovery. ANGEL was designed with the aim of lowering the computational complexity of existing approaches while ensuring the identification of high-quality partitions. Experimental results, both on synthetic and real-world networks, highlight the efficiency and effectiveness of the proposed approach, underlying its ability to outperform its direct competitors.

Since such topologies usually evolve as time goes by, we also introduced an extension of ANGEL, a dynamic community discovery approach that allows for tracking and analyzing dynamic communities life-cycles. We evaluate our DCD approach both quantitatively on well known synthetic benchmarks and qualitatively on real data. As future work, we plan to apply the proposed algorithms as support for network analytical tasks and, during the process, to implement and evaluate parallel and distributed ANGEL variants able to scale up to big data. Moreover, we also plan to approach the “Theseus’s ship paradox" with the aim of proposing novel post-processing heuristics able to consolidate community life-cycles (independently from the DCD approach used to generate them) thus avoiding instability phenomena and ambivalent interpretations.

Abbreviations
CD: Community discovery; DCD: Dynamic community discovery; LP: Label propagation; NF1: Normalized F1

Acknowledgements
Not applicable.

Authors' contributions
G.R. designed the algorithms, performed the experiments, validated the results and wrote the paper. The author(s) read and approved the final manuscript.

Funding
This work is supported by the scheme 'INFRAIA-01-2018-2019: Research and Innovation action', Grant Agreement n. 871042 'SoBigData++: European Integrated Infrastructure for Social Mining and Big Data Analytics'

Availability of data and materials
The developed algorithms are released under the BSD-license 2 clause, their implementation is available at https://github.com/GiulioRossetti/angel and within the CDlib python library https://github.com/GiulioRossetti/cdlib. The datasets generated and/or analysed during the current study are available in the Netorkrepository repository (http://networkrepository.com).

Competing interests
The authors declare that they have no competing interests.

Received: 12 March 2020 Accepted: 27 May 2020
Published online: 10 June 2020

References
Alhajj R, Jon R (2014) Encyclopedia of social network analysis and mining. Springer Publishing Company, Incorporated
Amoretti M, Ferrari A, Fornacciari P, Mordonini M, Rosi F, Tomaiuolo M (2016) Local-first algorithms for community detection. In: KDWeb
Arnaboldi V, Conti M, Passarella A, Dunbar RIM (2017) Online social networks and information diffusion: The role of ego networks. OSNEM 1:44–55. https://doi.org/10.1016/j.osnem.2017.04.001
Bazzi M, Jeub LG, Arenas A, Howison SD, Porter MA (2016) Generative benchmark models for mesoscale structure in multilayer networks. arXiv preprint arXiv:1608.06196
Buzun N, Korshunov A, Avanesov V, Filonenko I, Kozlov I, Turdakov D, Kim H (2014) Egolp: Fast and distributed community detection in billion-node social networks. In: ICDMW. IEEE. https://doi.org/10.1109/icdmw.2014.158
Casteigts A, Flocchini P, Quattrociocchi W, Santoro N (2012) Time-varying graphs and dynamic networks. Int J Parallel Emergent Distrib Syst 27(5):387–408
Cazabet R, Amblard F, Hanachi C (2010) Detection of overlapping communities in dynamical social networks. In: 2nd International Conference on Social Computing (SocialCom). IEEE. pp 309–314. https://doi.org/10.1109/socialcom.2010.51
Cazabet R, Rossetti G, Amblard F (2017) Dynamic community detection. In: Encyclopedia of Social Network Analysis and Mining. Springer. pp 1–10. https://doi.org/10.1007/978-1-4614-6170-8_383
Condon A, Karp RM (2001) Algorithms for graph partitioning on the planted partition model. Random Struct Algorithm 18(2):116–140
Coscia M, Giannotti F, Pedreschi D (2011) A classification for community discovery methods in complex networks. Stat Anal Data Min. https://doi.org/10.1002/sam.10133
Coscia M, Rossetti G, Giannotti F, Pedreschi D (2012) Demon: a local-first discovery method for overlapping communities. In: International Conference on Knowledge Discovery and Data Mining. ACM. pp 615–623. https://doi.org/10.1145/2339530.2339630
Coscia M, Rossetti G, Giannotti F, Pedreschi D (2014) Uncovering hierarchical and overlapping communities with a local-first approach. ACM Trans Knowl Discov Data (TKDD) 9(1):6
Coscia M, Rossetti G, Giannotti F, Pedreschi D (2014) Uncovering hierarchical and overlapping communities with a local-first approach. Trans Knowl Discov Data 9(1):6
Dao V-L, Bothorel C, Lenca P (2018) Community structure: A comparative evaluation of community detection methods. arXiv preprint arXiv:1812.06598
Epasto A, Lattanzi S, Paes Leme R (2017) Ego-splitting framework: from non-overlapping to overlapping clusters. In: SIGKDD. ACM. pp 145–154. https://doi.org/10.1145/3097983.3098054
Fortunato S (2010) Community detection in graphs. Phys Rep 486(3). https://doi.org/10.1016/j.physrep.2009.11.002
Fortunato S, Hric D (2016) Community detection in networks: A user guide. Phys Rep 659:1–44
Friedman M (1937) The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J Am Stat Assoc 32. https://doi.org/10.1080/01621459.1937.10503522
Granell C, Darst RK, Arenas A, Fortunato S, Gómez S (2015) Benchmark model to assess community structure in evolving networks. Phys Rev E 92(1):012805
Greene D, Doyle D, Cunningham P (2010) Tracking the evolution of communities in dynamic social networks. In: Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference On. IEEE. pp 176–183. https://doi.org/10.1109/asonam.2010.17
Himmel A-S, Molter H, Niedermeier R, Sorge M (2016) Enumerating maximal cliques in temporal graphs. In: IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). IEEE. pp 337–344. https://doi.org/10.1109/asonam.2016.7752255
Holme P, Saramäki J (2012) Temporal networks. Phys Rep 519(3):97–125
Hric D, Darst RK, Fortunato S (2014) Community detection in networks: Structural communities versus ground truth. Phys Rev E 90(6):062805
Jebabli M, Cherifi H, Cherifi C, Hamouda A (2018) Community detection algorithm evaluation with ground-truth data. Physica A Stat Mech Appl 492:651–706
Kumpula JM, Kivelä M, Kaski K, Saramäki J (2008) Sequential algorithm for fast clique percolation. Phys Rev E 78(2):026109

Lancichinetti A, Fortunato S, Kertész J (2009) Detecting the overlapping and hierarchical community structure in complex networks. New J Phys. https://doi.org/10.1088/1367-2630/11/3/033015
Lancichinetti A, Fortunato S, Radicchi F (2008) Benchmark graphs for testing community detection algorithms. Phys Rev E 78(4):046110. https://doi.org/10.1103/PhysRevE.78.046110
Li JD (2008) A two-step rejection procedure for testing multiple hypotheses. J Stat Plan Infer 138(6):1521–1527
Lin Y-R, Chi Y, Zhu S, Sundaram H, Tseng BL (2008) Facetnet: a framework for analyzing communities and their evolutions in dynamic networks. In: Proceedings of the 17th International Conference on World Wide Web (WWW). ACM. pp 685–694. https://doi.org/10.1145/1367497.1367590
Malliaros FD, Vazirgiannis M (2013) Clustering and community detection in directed networks: A survey. Phys Rep 533(4):95–142
Matias C, Miele V (2016) Statistical clustering of temporal networks through a dynamic stochastic block model. J R Stat Soc Ser B (Stat Methodol). https://doi.org/10.1111/rssb.12200
McDaid AF, Derek G, Neil H (2011) Normalized mutual information to evaluate overlapping community finding algorithms. arXiv preprint arXiv:1110.2515
Milli L, Monreale A, Rossetti G, Pedreschi D, Giannotti F, Sebastiani F (2015) Quantification in social networks. In: 2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA) Paris. pp 1–10. https://doi.org/10.1109/DSAA.2015.7344845
Moradi F, Olovsson T, Tsigas P (2014) A local seed selection algorithm for overlapping community detection. In: ASONAM. IEEE. https://doi.org/10.1109/asonam.2014.6921552
Morini M, Flandrin P, Fleury E, Venturini T, Jensen P (2017) Revealing evolutions in dynamical networks. arXiv preprint arXiv:1707.02114
Mucha PJ, Richardson T, Macon K, Porter MA, Onnela J-P (2010) Community structure in time-dependent, multiscale, and multiplex networks. Science 328(5980):876–878
Orman GK, Labatut V, Cherifi H (2012) Comparative evaluation of community detection algorithms: a topological approach. J Stat Mech Theory Exp 2012(08):08001
Palla G, Barabási A-L, Vicsek T (2007) Quantifying social group evolution. Nature 446(7136):664–667
Raghavan UN, Albert R, Kumara S (2007) Near linear time algorithm to detect community structures in large-scale networks. Phys Rev E 76(3). https://doi.org/10.1103/PhysRevE.76.036106
Rossetti G (2017) Rdyn: Graph benchmark handling community dynamics. J Complex Netw. https://doi.org/10.1093/comnet/cnx016
Rossetti G, Cazabet R (2018) Community discovery in dynamic networks: A survey. ACM Comput Surv (CSUR) 51(2):35
Rossetti G, Luca P, Salvatore R (2016) A novel approach to evaluate community detection algorithms on ground truthComplex Networks VII. Springer, Cham. pp 133–144
Rossetti G, Milli L, Cazabet R (2019) Cdlib: a python library to extract, compare and evaluate communities from complex networks. Appl Netw Sci. https://doi.org/10.1007/s41109-019-0165-9
Rossetti G, Pappalardo L, Kikas R, Pedreschi D, Giannotti F, Dumas M (2015) Community-centric analysis of user engagement in skype social network. In: ASONAM. ACM. https://doi.org/10.1145/2808797.2809384
Rossetti G, Pappalardo L, Kikas R, Pedreschi D, Giannotti F, Dumas M (2016) Homophilic network decomposition: a community-centric analysis of online social services. Soc Netw Anal Min J 6(103). https://doi.org/10.1007/s13278-016-0411-4
Rossetti G, Pappalardo L, Pedreschi D, Giannotti F (2017a) Tiles: an online algorithm for community discovery in dynamic social networks. Mach Learn 106(8):1213–1241
Rossetti G, Pappalardo L, Rinzivillo S (2016) A novel approach to evaluate community detection algorithms on ground truth. In: Complex Networks. http://www.giuliorossetti.net/about/wp-content/uploads/2015/12/Complenet16.pdf
Rossetti G, Pedreschi D, Giannotti F (2017b) Node-centric community discovery: From static to dynamic social network analysis. OSNEM 3:32–48
Soundarajan S, Hopcroft JE (2015) Use of local group information to identify communities in networks. Trans Knowl Discov Data. https://doi.org/10.1145/2700404
Takaffoli M, Sangi F, Fagnan J, Zaïane OR (2011) Modec-modeling and detecting evolutions of communities. In: 5th International Conference on Weblogs and Social Media (ICWSM). AAAI. pp 30–41
Viard T, Latapy M, Magnien C (2016) Computing maximal cliques in link streams. Theor Comput Sci 609:245–252
Whang JJ, Gleich DF, Dhillon IS (2016) Overlapping community detection using neighborhood-inflated seed expansion. IEEE Trans Knowl Data Eng 28(5):1272–1284
Xie J, Kelley S, Szymanski BK (2013) Overlapping community detection in networks: The state-of-the-art and comparative study. Comput Surv. https://doi.org/10.1145/2501654.2501657
Xie J, Szymanski BK (2012) Towards linear time overlapping community detection in social networks. In: PAKDD. https://doi.org/10.1007/978-3-642-30220-6_3
Yang J, Leskovec J (2015) Defining and evaluating network communities based on ground-truth. Knowl Inf Syst 42(1):181–213
Zakrzewska A, Bader DA (2015) 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. pp 559–564. https://doi.org/10.1145/2808797.2809375

Publisher's Note
Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.