State-of-the-art in Community Detection in Temporal Networks

Kostas Christopoulos[1] and Kostas Tsichlas[1]

Computer Engineering & Informatics Department, University of Patras, Greece.
kchristopou@ceid.upatras.gr,ktsichlas@ceid.upatras.gr

Abstract. Community detection is a prominent process on networks and has been extensively studied on static networks the last 25 years. This problem concerns the structural partitioning of networks into classes of nodes that are more densely connected when compared to the rest of the network. However, a plethora of real-world networks are highly dynamic, in the sense that entities (nodes) as well as relations between them (edges) constantly change. As a result, many solutions have also been applied in dynamic/temporal networks under various assumptions concerning the modeling of time as well as the emerging communities. The problem becomes quite harder when the notion of time is introduced, since various unseen problems in the static case arise, like the identity problem. In the last few years, a few surveys have been conducted regarding community detection in time-evolving networks. In this survey, our objective is to give a rather condensed but up-to-date overview, when compared to previous surveys, of the current state-of-the-art regarding community detection in temporal networks. We also extend the previous classification of the algorithmic approaches for the problem by discerning between global and local dynamic community detection. The former aims at identifying the evolution of all communities and the latter aims at identifying the evolution of a partition around a set of seed nodes.
Keywords: Temporal Graphs/Networks, Community Detection

1 Introduction

Networks are widely used as a method for analyzing data in many scientific fields, such as social sciences, transportation and biology. The prrocess of community detection (henceforward also referred as CD) , that has its origins in graph partitioning, is concerned with node intra-connectivity and its goal is to identify highly linked groups (communities) of nodes. For example, finding clusters of users in social networks and functional protein complexes in bioinformatics networks are two widely used applications of this problem.

In general, a static network is represented as $G = (V, E)$, where V is the set of vertices (entities) and E is the set of edges (interactions/relations between entities). An edge can be directed, such as the connection between two people where one sends an email to another or undirected, such as the connection between two collaborating peers. Lastly, edges among nodes can be associated with weights

(e.g., frequency of interactions) or nodes can be associated to weights (e.g., specific properties of nodes). In many cases, real-world networks are dynamic, in the sense that new edges or nodes appear and existing edges or nodes disappear. As a result, the communities themselves change because of the evolution of the network. The appearance of a new community, the disappearance of an existing and the split of an existing into two or more, are examples of changes in the community structure.

For the purpose of representing a temporal network, it can be assumed as a sequence of static graphs (snapshots) or as a network with time annotations on its nodes/edges that represent its time evolution. The former approach requires to specify the size of our time window that defines the time instances of snapshot construction. The later, is related to events, like edge/node insertion or deletion or its existence interval. The notion of a time annotation may have different aspects/interpretations depending on the application. The following three aspects have been used in the literature [27]:

1. *Point Networks:* every link among two vertices x and y, which has been created at certain time t, can be represented as a triplet $e = (x, y, t)$.
2. *Time Interval Networks:* the time-interval connection of two nodes is represented as a quadruplet $e = (x, y, t, \Delta t)$. Δt is the duration of the link between the vertices x and y.
3. *Incremental Networks:* edges/nodes can only be added and deletions are forbidden.

In the last few years, many surveys in the field of CD in temporal networks have been published. These are discussed briefly in Section 2. The main contribution of this paper is in Section 3, which can be summarized as follows: 1) we provide an updated overview of the current state-of-the-art methods for CD in temporal networks since the last years there are quite a few new related results, and 2) we further classify the approaches in global CD and local CD in temporal networks. The latter contribution concerns the discussion on new methods related as to how a community around a given set of nodes evolves in time. This approach is appropriate in cases where one is not interested at discovering all communities, leading to large efficiency and effectiveness gains. Finally, we conclude in Section 4.

2 Related Work

In this section are discussed existing surveys on CD in temporal networks. In [4] a general classification of methods is proposed into two classes: 1) *Online* (real time, incremental detection) and 2) *Offline* (prior knowledge of network changes). Similarly, in [20], authors identify the same two classes but they focus on online approaches dividing them into two sub-classes: 1) *Temporal Smoothness*, where at each snapshot a static CD algorithm is run from scratch and 2) *Dynamic Update*, where the communities are updated based on the differences of two consecutive snapshots.

A very good survey is [32], where the dynamic CD algorithms have been classified based on the strategy they use for detecting meaning-full evolving communities. They propose three classes of approaches: 1) *Instant - optimal*, where the algorithms detect communities from scratch at each snapshot and then match them between consecutive snapshots, 2) *Temporal Trade - off*, where the algorithms detect communities comparing the topology of two consecutive snapshots and 3) *Cross - Time*, where the algorithms discover communities using the information of all snapshots. Similarly in [2], the authors identify three similar categories as well: 1) *Two - Stage methods* that detect the communities from scratch at each snapshot and then match them across different snapshots, 2) *Evolutionary Clustering*, that detect communities based on the changes in the topology between two consecutive snapshots and 3) *Coupling Graph* that creates an aggregate network containing all snapshots and then uses a static CD algorithm.

In [12] the authors classified the dynamic CD algorithms into four classes: 1) *Independent Detection*, here the communities are detected from scratch at each snapshot and then they are matched among consecutive snapshots, 2) *Dependent Detection*, where communities are identified based on the changes of the topology between two consecutive snapshots, 3) *Simultaneous Detection*, where communities are detected by using the information from all snapshots and 4) *Dynamic Detection*, where the communities are updated based on the network updates. Additionally, in [17] four classes of evolving clustering methods are provided that are similar to the preceding classification: 1) *Sequential mapping-driven*, 2) *Temporal smoothing-driven*, 3) *Milestone detection-driven* and 4) *Incremental adaptation-driven*.

In [7], a survey is conducted exclusively for incremental (online) CD methods in temporal networks. The proposed classification contains two subcategories of incremental methods: 1) *Community Detection in Fully Temporal Networks*, where insertions and deletions of nodes and edges are permitted and 2) *Community Detection in Growing Temporal Networks*, where only insertions of nodes and edges are permitted.

Finally, it is worth mentioning that multilayer networks can be used for dynamic community detection [23]. In particular, a multilayer network is a network made of multiple networks, called layers, where each layer has the same number of nodes, but different edge connections. The multilayer network model is commonly employed in the study of temporal networks, in which each snapshot is represented as a layer and all layers are interconnected based on their time relationship.

3 Detecting Communities in Temporal Networks: Classification

In this section, is provided a classification of dynamic CD methods. At first, the different versions of the dynamic CD problem are discerned into two: *Global* and *Local*. The former concerns the identification of all communities and their

evolution in the temporal network, while the latter concerns the identification of a community around a given set of seed nodes and its evolution in the temporal network. This corresponds to the division between global and local CD in static graphs. The main body of the literature concerns the global version of the problem, however there is recently an admittedly small number of publications on the local version. Although the local version uses techniques especially from the global online dynamic CD category we believe that it constitutes a class by itself since there are many differences in terms of efficiency as well as effectiveness of the methods. Thus we identify the following 5 classes of methods:

1. Global:
   (a) Community Detection from scratch and match
   (b) Dependent or Temporal Trade - off Community Detection
   (c) Simultaneous or Offline Community Detection
   (d) Online Community Detection in fully Temporal Networks and in growing Temporal Networks
2. Local: Community Detection in Temporal Networks using Seed Nodes

Table 1. Overview of the proposed temporal community detection classification.

Global Temporal Community Detection		
Class	Description	References
From Scratch and Match	Static algorithm at each snapshot and matching	[10] [30] [31], [24], [40]
Dependent or Temporal Trade-off	Based on the topology of two adjacent snapshots	[13], [34], [35] [43] [26] [36] [18]
Simultaneous or Offline	Creation of single graph - run static algorithm on it	[16] [22] [29], [28], [39], [15]
Online Community Detection	Update in proportion to network modifications	[46] [11] [19] [42] [41] [38] [6] [33] [48], [1], [9], [47]
Local Temporal Community Detection		
Class	Description	References
Using Seed Nodes	Update only the area around the seed node	[21] [44] [45] [14] [3]

In the following, we present in detail these five classes by discussing recent representative methods.

3.1 Community Detection from Scratch and Match

In this class, a static CD algorithm is applied on each snapshot from scratch and then the communities that have been found at snapshot $t + 1$ are matched (by using a similarity metric like Jaccard similarity) with the communities found

at snapshot t. The advantage is that communities can be detected in parallel and existing methods for static CD can be used. On the other hand, instability (the communities may have a lot of changes between consecutive snapshots) and inefficiency (in each snapshot a static community detection algorithm is invoked) are its main two drawbacks. In the following, we discuss some representative methods of this category (more such methods can be found in [31,24,40]).

In [10], given as input a sequence of snapshots of a network, they initially find the network representatives based on the common nodes between two consecutive snapshots and list the communities. Then, the community representatives are identified and consequently, the relation of communities between different snapshots (G_t, G_{t+1}, G_{t+2}) is established by finding the predecessor(s) and successor(s) for each community. Finally, the dynamic CD is performed by looking at six different events that may happen to a community (Grown, Merged, Split, Shrunken, Born and Vanished). For all the events they track forward the network sequence of snapshots with the exception of the shrunken and split communities where a backtracking process is applied.

In [30] the authors use sliding windows to track the dynamics by computing partitions for each time slice and by modifying the community description at time t using the structures found at times $t-1$ and $t+1$. More specifically, the data set is divided into time windows and for each one a static CD algorithm is used. Then, the similarity scores between communities at times $t-2, t-1, t, t+1, t+2$ are computed. This information allows to easily distinguish noise from real evolution. Consequently, this information is used to smooth out the communities evolution. Then, communities which have been generated by unduly splits are merged, while communities that have been generated by artificial merges are separated. At the end of the procedure, a description of the network evolution is obtained.

3.2 Dependent or Temporal Trade-off Community Detection

Methods in this class process repeatedly network changes. Initially, by using a static CD algorithm they find partitions for the initial state (first snapshot) of the network, and then they find communities at snapshot t by using information from both the current snapshot (t) and previous snapshots ($< t$). Methods in this subcategory don't suffer from the instability problem and are faster than those from the previous category. On the other hand, the avalanche effect [1] and the fact that this method is not parallelizable are its two main drawbacks. Global and multiobjective optimization methods are the most common subcategories in this class. In the following, we discuss some representative methods of this category (more such methods can be found in [13,34,35]).

In [18] they detect evolutionary community structure in a weighted dynamic network. For each snapshot the follow process is iteratively applied: i) Firstly, the

[1] The avalanche effect describes the phenomenon when communities can experience substantial drifts compared to what a static algorithm would find on the static network at a particular time instance.

initial partition is detected (using the input matrix that describes the previous) and then ii) the community is expanded based on the assumption that the nodes are attached to the cluster that provides the highest modularity gain. Finally, in the last phase, the merging process starts if and only if the modularity of the merged community is higher than each partition separately. Another recent dynamic community detection algorithm is proposed in [36]. This method utilizes all the past information from the network and by using the algorithm C-Blondel, which is a modification of the Louvain algorithm, manages to compress the Network. Thus, the compressed network consists of all the historical snapshots and the changes which have been occurred on the network.

In [26], a multi-objective optimization approach has been adopted. Initially, the probability fusion method is adopted and two different approaches (neighbor diversity and neighbor crowd) are used. In this way, suitable communities are created in a fast and accurate way. Moreover, by utilizing a progression metric, the authors can detect the similarities of formed communities between two successive snapshots. The same approach has also been used by [43]. Their method, called DYN-MODPSO, is suitable for large-scale dynamic CD. Like the previous method, they use two different approaches optimizing NMI (Normalized Mutual Information) and CS (Community Score) metrics.

3.3 Simultaneous or Offline Community Detection

Methods in this class discover partitions by considering all states of the temporal network at the same time. A single multilayer network is created from all snapshots using edges based on the relationship between nodes at the same snapshot and at adjacent (preceding and succeeding) snapshots. Then, the communities are detected by using an appropriately modified static algorithm on the multilayer network. Methods in this category don't suffer from instability and the avalanche effect. On the other hand, they have certain limitations like a requirement for a fixed number of communities, or lacking a mechanism to determine operations between temporally successive partitions (like merge), etc.. In the following, we discuss some representative methods of this category (more such methods can be found in [29,28,39,15]).

A significant study of the fundamental limits of discovering community structure in dynamic networks is done in [16]. The authors analyze the boundaries of detectability for a Dynamic Stochastic Block Model (DSBM) that nodes affiliations can change over time (from one community to other), and edges are created separately at each time step. The method exploits the powerful tools of probabilistic generative models and Bayesian inference, and by utilizing the cavity method, they obtain a clearly defined detectability threshold as a function of the rate of change and the communities strength. Below this threshold, they claim that no efficient algorithm can identify the communities better than chance. Then, they give two algorithms that are optimal in the sense that they succeed in detecting the correct communities up to this up to this threshold. The first algorithm utilizes belief propagation, which provides an asymptotically

optimal accuracy, while the second is an efficient spectral algorithm, founded on linearizing the belief propagation equations.

Another algorithm based on clique enumeration is proposed in [22]. They use an adaptation for temporal networks of a well-known recursive static backtracking algorithm, Bron-Kerbosch [8]. The parameter "Δ-slice degeneracy" is introduced, which is a modification of the degeneracy parameter that is often used in static graphs, and it is an easy way to measure the sparsity of the network.

3.4 Online Community Detection

In these methods, the temporal network is not considered as a sequence of snapshots, but as a succession of network transformations instead. The methods are initialized by discovering partitions at time 0 and then the community structure is updated in each update of nodes/edges. This class of methods is further divided into two main subcategories of incremental methods: 1) *Community Detection in Fully Temporal Networks*, where insertions and deletions of nodes and edges are allowed and 2) *Community Detection in Growing Temporal Networks*, where only insertions of nodes and edges are allowed. In addition, the methods of this class can either handle network updates in batches of arbitrary size (one extreme is to consider batches of size 1, that is after each update the community structure is updated). One advantage of this method is that algorithms for static CD can be used with easy modifications. Moreover, this method does not suffer from instability, and it is quite efficient since updates for the community structure are usually applied locally. In the following, we discuss some representative methods of this category (more such methods can be found in [48,1,9,47]).

In [46], a filtering technique is introduced, which is called "Δ-Screening". The technique of Δ-Screening captures in each time step, new inserted/deleted nodes (V_t) that impact the structure of the network. Initially, a static algorithm discovers the communities at time $t = 0$. Then, for each time step, all the added nodes are assigned a new community label. Then Δ-Screening captures a subset R_t ($R_t \subseteq V_t$) of these nodes. Consequently, the static algorithm (mentioned above) is invoked in order to detect the evolution of the communities, visiting only the subset of nodes R_t, of the most significant changes.

An incremental, modified Louvain algorithm [5] is proposed in [11]. Nodes and edges are inserted in or deleted from the network as time evolves, and Louvain is implemented only for those communities that are affected. The local modularity metric is applied only in a part of the network, where the changes are taking place. As a result, stability and efficiency are the two main advantages of the method. A dynamic CD algorithm, which is a modified version of the static algorithm in [37], based on distance dynamics is proposed in [19]. By utilizing the local interaction model, based on the Jaccard distance, the method in [19] overcomes the well-known disadvantages of modularity-based algorithms by detecting small communities or outliers. This is achieved regardless of the processing order of the increment set and the algorithm can achieve the same community partition results in near-linear time.

One recent, efficient and parameter-free incremental method, based on the Matthew effect, is proposed in [42]. Unlike other incremental approaches, changes are processed in batches. Between two consecutive snapshots, deletion and insertion of nodes and edges are performed. The degree of nodes as well as node and group attractiveness for the purpose of the changed sub-graph to be extracted are used. Then, the affected and non-affected communities are calculated iteratively between each pair of consecutive snapshots. The same dynamic CD framework, based on information dynamics, is used in [41].

An online version of the Clique Percolation Method (CPM), combined with the Label Propagation Algorithm (LPA) is presented in [6].The proposed algorithm OLCPM (Online Label Propagation Clique Percolation Method) is a two step framework which firstly uses the Dynamic CPM to update the communities locally by utilizing a stream model, in order to improve the efficiency. Then, by using LPA it solves the problem of nodes affiliation while a node can be allocated to one or more communities. Finally, in [33], the algorithms named Tiles is presented. Tiles is a a streaming algorithm, treating each topological perturbation as a domino tile fall: whenever a new interaction emerges in the network, Tiles first updates the communities locally, then propagates the changes to the surrounding nodes modifying the neighbors' partition memberships.

3.5 Local Community Detection in Temporal Networks

Given a set of seed nodes Z, our goal is to detect the community which includes Z. The main assumption in this case is that Z is of high importance (e.g., high degree centrality) and act as the community reference point. This problem differs from general temporal CD approaches since our objective is to discover the community defined around the set of seed nodes. Notice that the online methods described in 3.4 are global in the sense that they maintain a partition of the network in communities. The algorithms in this class are very efficient since it is required from them to maintain a single community.

In [21] a hierarchical algorithm is presented. This method discovers communities in temporal networks based on hubs (nodes of high degree centrality) by grouping nodes in their vicinity. Each node carries hub information (e.g., distance between nodes, hub and parent nodes, threshold level, etc.) and the idea is based on propagating this information through the network. Then, the intra-node hubs transfer the information (message) to the outer nodes and in this way, all non-hub nodes are assigned to the closest hub and the fuzzy membership of each node is calculated. This method can be readily adopted for the case of a set of seeds propagating the information only in their vicinity. An advantage of this method is that only a small number of processing steps (adding or removing edges) have to be performed too update the partitions while at the same time is parameter-free.

A dynamic algorithm for local community detection using a set of seeds is proposed in [44]. Initially, a greedy static algorithm is used in order to discover the local community. During this process, the community initially contains only the seed node and in each iteration one neighbor node is added maximizing a

chosen fitness score. At the end of this step there is a collection of sequences (vertex,interior/border edge sum and fitness score), in increasing order of fitness score. Then, the next phase start and in each network change, the algorithm modify the collection of sequences. If after the modification the fitness score of a position is higher than the fitness score of the next position, then the node is removed and interior/border edges are modified as well. When this step is finished, the collection of modified sequences is scanned and if in one position the increasing order of the fitness score is violated then the set from this position until its end is removed. Finally, the static algorithm is used one more time in order to add new nodes to the local community. A full streaming version of the seed set expansion method is described in [45].

Another approach is Evoleaders [14] that employs leader nodes with followers, in order to identify the evolution of the communities. The "Top leaders" algorithm [25] initialize D leader nodes, one for each community, and associate the nodes of the network to an appropriate leader. In this way, the communities are constructed and, by utilizing the highest centrality node, a new leader is picked and the old one is replaced. Then, in each time step, for the initialization of the leader nodes, their common neighbors from the previous and the current time step is taken into consideration and the Top leaders algorithm is used iteratively. Consequently, the process of community splitting starts and then all small communities can be merged, in an appropriate manner, so that the quality (in terms of modularity) of the communities is improved. Finally, very recently, [3] described a framework that strengthens the vicinity of the seed set (called anchors) exploiting the fact that the seed set is of central importance for the evolving community.

4 Conclusion

Our aim in this survey is to reexamine all the recent surveys in the field of CD in temporal networks and to propose a new category of methods based on local community detection. Thus, we propose five classes of algorithms and discuss some representative methods. The advantages and drawbacks of each class are also discussed. In future work , it will be beneficial to delve into, in more detail, the local community detection class and to enrich the current survey with more recent literature.

Acknowledgments

This research was supported by the Hellenic Foundation for Research and Innovation (H.F.R.I.) under the “2nd Call for H.F.R.I. Research Projects to support Faculty Members & Researchers” (Project Number: 3480).

References

1. Elyazid Akachar, Brahim Ouhbi, and Bouchra Frikh. Acsimcd: A 2-phase framework for detecting meaningful communities in dynamic social networks. *Future Generation Computer Systems*, 125:399–420, 2021.
2. Thomas Aynaud, Eric Fleury, Jean-Loup Guillaume, and Qinna Wang. Communities in evolving networks: definitions, detection, and analysis techniques. In *Dynamics On and Of Complex Networks, Volume 2*, pages 159–200. Springer, 2013.
3. Georgia Baltsou and Konstantinos Tsichlas. Dynamic community detection with anchors. 2022.
4. Shweta Bansal, Sanjukta Bhowmick, and Prashant Paymal. Fast community detection for dynamic complex networks. In *Complex networks*, pages 196–207. Springer, 2011.
5. Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, oct 2008.
6. Souâad Boudebza, Rémy Cazabet, Faiçal Azouaou, and Omar Nouali. Olcpm: An online framework for detecting overlapping communities in dynamic social networks. *Computer Communications*, 123:36–51, 2018.
7. Fariza Bouhatem, Ali Ait El Hadj, Fatiha Souam, and Abdelhakim Dafeur. Incremental methods for community detection in both fully and growing dynamic networks. *Acta Universitatis Sapientiae, Informatica*, 13(2):220–250, 2021.
8. Coen Bron and Joep Kerbosch. Algorithm 457: finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577, 1973.
9. Zhe Chen, Aixin Sun, and Xiaokui Xiao. Incremental community detection on large complex attributed network. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(6):1–20, 2021.
10. Zhengzhang Chen, Kevin A. Wilson, Ye Jin, William Hendrix, and Nagiza F. Samatova. Detecting and tracking community dynamics in evolutionary networks. In *2010 IEEE International Conference on Data Mining Workshops*, pages 318–327, 2010.
11. Mário Cordeiro, Rui Portocarrero Sarmento, and Joao Gama. Dynamic community detection in evolving networks using locality modularity optimization. *Social Network Analysis and Mining*, 6(1):1–20, 2016.
12. Narimene Dakiche, Fatima Benbouzid-Si Tayeb, Yahya Slimani, and Karima Benatchba. Tracking community evolution in social networks: A survey. *Information Processing and Management*, 56(3):1084–1102, 2019.
13. Francesco Folino and Clara Pizzuti. An evolutionary multiobjective approach for community discovery in dynamic networks. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1838–1852, 2013.
14. Wenhao Gao, Wenjian Luo, and Chenyang Bu. Evolutionary community discovery in dynamic networks based on leader nodes. In *2016 International Conference on Big Data and Smart Computing (BigComp)*, pages 53–60, 2016.
15. Laetitia Gauvin, André Panisson, and Ciro Cattuto. Detecting the community structure and activity patterns of temporal networks: A non-negative tensor factorization approach. *PLOS ONE*, 9(1):e86028, 2014.
16. Amir Ghasemian, Pan Zhang, Aaron Clauset, Cristopher Moore, and Leto Peel. Detectability thresholds and optimal algorithms for community structure in dynamic networks. *Physical Review X*, 6(3):031005, 2016.

17. Maria Giatsoglou and Athena Vakali. Capturing social data evolution using graph clustering. *IEEE Internet Computing*, 17(1):74–79, Jan 2013.
18. Chonghui Guo, Jiajia Wang, and Zhen Zhang. Evolutionary community structure discovery in dynamic weighted networks. *Physica A: Statistical Mechanics and its Applications*, 413:565–576, 2014.
19. Qian Guo, Lei Zhang, Bin Wu, and Xuelin Zeng. Dynamic community detection based on distance dynamics. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 329–336, 2016.
20. Tanja Hartmann, Andrea Kappes, and Dorothea Wagner. Clustering evolving networks. In *Algorithm engineering*, pages 280–329. Springer, 2016.
21. Pascal Held and Rudolf Kruse. Detecting overlapping community hierarchies in dynamic graphs. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1063–1070. IEEE, 2016.
22. Anne-Sophie Himmel, Hendrik Molter, Rolf Niedermeier, and Manuel Sorge. Enumerating maximal cliques in temporal graphs. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 337–344. IEEE, 2016.
23. Xinyu Huang, Dongming Chen, Tao Ren, and Dongqi Wang. A survey of community detection methods in multilayer networks. *Data Mining and Knowledge Discovery*, 35:1–45, 2021.
24. Kaveh Kadkhoda Mohammadmosaferi and Hassan Naderi. Evolution of communities in dynamic social networks: An efficient map-based approach. *Expert Systems with Applications*, 147:113221, 2020.
25. Reihaneh Rabbany Khorasgani, Jiyang Chen, and Osmar R Zaiane. Top leaders community detection approach in information networks. In *4th SNA-KDD workshop on social network mining and analysis*. Citeseer, 2010.
26. Weimin Li, Xiaokang Zhou, Chao Yang, Yuting Fan, Zhao Wang, and Yanxia Liu. Multi-objective optimization algorithm based on characteristics fusion of dynamic social networks for community discovery. *Information Fusion*, 79:110–123, 2022.
27. Panagiotis Liakos, Katia Papakonstantinopoulou, Theodore Stefou, and Alex Delis. On compressing temporal graphs. 2022.
28. Catherine Matias and Vincent Miele. Statistical clustering of temporal networks through a dynamic stochastic block model series b statistical methodology. 2017.
29. Catherine Matias, Tabea Rebafka, and Fanny Villers. A semiparametric extension of the stochastic block model for longitudinal networks. *Biometrika*, 105(3):665–680, 2018.
30. Matteo Morini, Patrick Flandrin, Eric Fleury, Tommaso Venturini, and Pablo Jensen. Revealing evolutions in dynamical networks. *arXiv preprint arXiv:1707.02114*, 2017.
31. Gergely Palla, Albert-László Barabási, and Tamás Vicsek. Quantifying social group evolution. *Nature*, 446(7136):664–667, Apr 2007.
32. Giulio Rossetti and Rémy Cazabet. Community discovery in dynamic networks: A survey. *ACM Comput. Surv.*, 51(2), feb 2018.
33. Giulio Rossetti, Luca Pappalardo, Dino Pedreschi, and Fosca Giannotti. Tiles: an online algorithm for community discovery in dynamic social networks. *Machine Learning*, 106(8):1213–1241, 2017.
34. Polina Rozenshtein, Nikolaj Tatti, and Aristides Gionis. Discovering dynamic communities in interaction networks. In Toon Calders, Floriana Esposito, Eyke Hüllermeier, and Rosa Meo, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 678–693, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

35. Hadiseh Safdari, Martina Contisciani, and Caterina De Bacco. Reciprocity, community detection, and link prediction in dynamic networks. *Journal of Physics: Complexity*, 3(1):015010, 2022.
36. Mahsa Seifikar, Saeed Farzi, and Masoud Barati. C-blondel: An efficient louvain-based dynamic community detection algorithm. *IEEE Transactions on Computational Social Systems*, 7(2):308–318, 2020.
37. Junming Shao, Zhichao Han, Qinli Yang, and Tao Zhou. Community detection based on distance dynamics. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1075–1084, 2015.
38. Xing Su, Jianjun Cheng, Haijuan Yang, Mingwei Leng, Wenbo Zhang, and Xiaoyun Chen. Incnsa: Detecting communities incrementally from time-evolving networks based on node similarity. *International Journal of Modern Physics C*, 31(07):2050094, 2020.
39. Jimeng Sun, Christos Faloutsos, Spiros Papadimitriou, and Philip S. Yu. Graphscope: Parameter-free mining of large time-evolving graphs. KDD '07, page 687–696, New York, NY, USA, 2007. Association for Computing Machinery.
40. Yang Sun, Junhua Tang, Li Pan, and Jianhua Li. Matrix based community evolution events detection in online social networks. In *2015 IEEE international conference on smart city/SocialCom/SustainCom (SmartCity)*, pages 465–470. IEEE, 2015.
41. Zejun Sun, Jinfang Sheng, Bin Wang, Aman Ullah, and FaizaRiaz Khawaja. Identifying communities in dynamic networks using information dynamics. *Entropy*, 22(4):425, 2020.
42. ZeJun Sun, YaNan Sun, Xinfeng Chang, Feifei Wang, Zhongqiang Pan, Guan Wang, and Jianfen Liu. Dynamic community detection based on the matthew effect. *Physica A: Statistical Mechanics and its Applications*, page 127315, 2022.
43. Ying Yin, Yuhai Zhao, He Li, and Xiangjun Dong. Multi-objective evolutionary clustering for large-scale dynamic community detection. *Information Sciences*, 549:269–287, 2021.
44. Anita Zakrzewska and David A. Bader. A dynamic algorithm for local community detection in graphs. ASONAM '15, page 559–564, New York, NY, USA, 2015. Association for Computing Machinery.
45. Anita Zakrzewska and David A Bader. Tracking local communities in streaming graphs with a dynamic algorithm. *Social Network Analysis and Mining*, 6(1):1–16, 2016.
46. Neda Zarayeneh and Ananth Kalyanaraman. A fast and efficient incremental approach toward dynamic community detection. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '19, page 9–16, New York, NY, USA, 2019. Association for Computing Machinery.
47. Hedia Zardi, Bushra Alharbi, Walid Karamti, Hanen Karamti, and Eatedal Alabdulkreem. Detection of community structures in dynamic social networks based on message distribution and structural/attribute similarities. *IEEE Access*, 9:67028–67041, 2021.
48. Di Zhuang, J Morris Chang, and Mingchen Li. Dynamo: Dynamic community detection by incrementally maximizing modularity. *IEEE Transactions on Knowledge and Data Engineering*, 33(5):1934–1945, 2019.