Frequent Subgraph Mining Algorithms in Static and Temporal Graph-Transaction Settings: A Survey

Ali Jazayeri and Christopher C. Yang

Abstract—Networks are known as perfect tools for modeling various types of systems. In the literature of network mining, frequent subgraph mining is considered as the essence of mining network data. In this problem, the dataset is composed of networks representing multiple independent systems or one system at multiple time stamps. The cores of mining frequent subgraphs are graph and subgraph isomorphism. Due to the complexities of these problems, the frequent subgraph mining algorithms proposed in the literature employ various heuristics for candidate generation, duplicate subgraphs pruning, and support computation. In this survey, we provide a classification of proposed algorithms in the literature. The algorithms for static networks have found numerous applications. Therefore, these algorithms will be reviewed in detail. Besides, it is discussed that consideration of temporality of data can impact the derived insight and attracted substantial attention in recent years. However, prior surveys have not comprehensively examined the algorithms of frequent subgraph mining in a database of temporal networks represented as network snapshots. Therefore, the algorithms proposed for mining frequent subgraphs in temporal networks are reviewed. Moreover, most of the surveys have focused on main-memory algorithms. Here, we review disk-based, parallel, and distributed algorithms proposed for mining frequent subgraphs.

Index Terms—Subgraph mining, network mining, temporal networks, static networks

1 Introduction

Networks are modeling tools composed of sets of vertices, representing individual and discrete entities, and edges, representing the interactions between pairs of vertices. The potential of networks for modeling of various systems in different domains has been known since the early years of the emergence of general systems theory. Bertalanffy points out to network theory as a representative of "topology or relational mathematics" as one of the "novel developments intended to meet the needs of a general theory of systems" [1]. The advantages of networks for modeling and solving real-world problems have been identified much earlier. The solution of the famous seven bridges of Königsberg (in 1,735) is considered as one of the first applications of network (or graph) theory [2].

Networks have been used as perfect tools for modeling of systems in different disciplines. Hence, many of the network-related concepts have been developed in parallel in different domains, and therefore there is no universally accepted terminology. For example, the first term used is called networks in some domains and graphs in others. There are several important concepts with crucial roles in network mining traditionally using "graph", such as sub*graph* mining, *graph* and sub*graph* isomorphism, and *graph*-transaction setting. We use the terms network and graph interchangeably. Here, a subgraph is considered as a subset of network's components, vertices, and edges, and along with other concepts used throughout this paper will be introduced formally in Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TBDATA.2021.3072001.

The subgraphs that appear more frequently in one network or multiple networks of a set of networks might exhibit essential characteristics of the source system that network(s) represent [3]. For example, in a set of networks representing the chemical compounds of a set of drugs developed for the treatment of a specific disease, the subgraphs that appear in most of them might help to identify essential chemical substances that make this set of drugs good candidates for curing the disease [4]. However, as it will be discussed in this paper, finding such subgraphs is not computationally trivial. Although data mining techniques are being increasingly applied to non-traditional domains, existing frequent pattern discovery approaches, such as algorithms proposed for association rule mining, cannot be directly used for finding frequent subgraphs. This is mainly because the transactional frameworks assumed by these algorithms cannot be directly applied to model the datasets composed of networks effectively.

Some algorithms have been proposed which constrain the outputs to a specific class of subgraphs. These constraints are, for example, based on the structure and frequency, or some measures of significance or interest. Nonetheless, the output of the algorithms proposed for

• *The authors are with the College of Computing and Informatics, Drexel University, Philadelphia, PA 19104 USA. E-mail: {aj629, ccy24}@drexel.edu.*

Manuscript received 10 Aug. 2020; revised 3 Mar. 2021; accepted 6 Apr. 2021. Date of publication 8 Apr. 2021; date of current version 11 Nov. 2022. Corresponding author: Ali Jazayeri.)
Recommended for acceptance by P. D. Urso.
Digital Object Identifier no. 10.1109/TBDATA.2021.3072001

frequent subgraph mining is generally composed of a large number of subgraphs, not all of them are necessarily significant. Besides that, systems work and change over time. In many applications, the time scale of changes in systems is large; therefore, considering the network as a static representation of the system in short periods of time does not significantly impact the derived insight. However, it is not always the case, and the temporality of networks should be taken into account. One common approach is to aggregate and show the temporal aspects of networks as attributes of vertices and edges of static networks. It is shown that this approach might not capture all the temporal information of the network and, consequently, impact the findings [5], [6].

On the other hand, metrics such as distance, diameter, centralities, paths, and connectivity, which have relatively known definitions in static networks, can be differently interpreted and defined in temporal networks. Therefore, the insights which can be drawn from temporal networks might be significantly different from their static or aggregated counterparts, and overlooking the temporal aspects of the network decreases the richness of modeling and analysis [7], [8], [9], [10], [11]. On the other hand, utilizing networks as dynamic and time-varying modeling tools can help to identify essential components of the network more precisely [12], [13]. It is shown that temporal networks can better model most of the natural and social systems. Furthermore, temporal networks can reach controllability (inject some inputs into some of the specific vertices to direct the network toward the desired state) much faster than their static counterparts. This phenomenon is attributed to the dynamic and changing topology and inherent nonlinearity of temporal networks, which are absent in static networks [14]. Despite all these advantages, the addition of the temporal aspect makes the mining of frequent subgraphs even more complicated.

In general, mining and analytical approaches towards networks can be categorized into three, not completely independent categories. In the first category, we are interested in finding some individual nodes or edges based on some pre-defined measures of significance. There are many different metrics developed for mining and finding these individual elements, such as centrality and prestige measures. We call this category microscale mining or analysis of networks. In the second category, we are interested in the characteristics of the network at the macro level. Although these macro-level characteristics are created from the characteristics of the individual elements, they can be considered as emergent characteristics that appear as collective or global features of the network. This category is considered as macroscale analysis and mining of networks. In the third category, we are interested in the mesoscale characteristics of networks. At this scale, the patterns of interest are subsets of the network's components, vertices, and edges, meeting some pre-defined criteria. Many of the proposed algorithms for network analysis and mining belong to this category. In [15], [16], a similar categorization has been proposed for the classification of approaches toward the investigation of the temporal behavior of (web) networks. This survey covers a subcategory of mining algorithms at the mesoscale.

1.1 Survey Scope

This paper covers algorithms proposed for finding frequent subgraphs in a database composed of a set or sequence of networks. This setting is called *network-transaction setting* (also called network database, graph database, or graph-transaction setting in some literature). The output of this problem is traditionally called *frequent subgraphs* (coming after *frequent itemsets* in *frequent itemset mining* literature). Another version of this problem is to discover frequent or significant subgraphs in one *giant* network. The outputs of this latter problem are traditionally called *motifs* (after a study by [17]). Network motifs are defined as inter-connected patterns occurring more than a user-defined threshold in the network of interest than some reference networks (generally randomized versions of the original network). It should be noted that being *frequent* is implied when we use *motifs*; however, for subgraphs, it should be explicitly mentioned. In this survey, we focus on the first family of algorithms, mining frequent subgraphs in a database of network transactions. For the problem of mining frequent subgraphs (motifs) in one giant static or temporal network, refer to [18].

In the graph-transaction setting, there is no specific limitation on the topology of the subgraphs mined. However, some algorithms might limit their search strategies to specific types of subgraphs. These algorithms still follow the same logic for discovering frequent subgraphs. However, the algorithms are modified to narrow down the search space based on the topology of interest. The mined frequent subgraphs then might be used as input features for different mining techniques or machine learning algorithms. Although we cover the algorithms used for mining frequent subgraphs, other data mining and machine learning approaches using the outputs of the frequent subgraph mining algorithms as their inputs or feature set are not covered in this survey. Interested readers may refer to unsupervised and supervised pattern learning [19] such as clustering and construction of decision trees using mined frequent subgraphs [20], [21], [22], [23], [24], integrating a frequent subgraph mining algorithm [25] with a boosting algorithm for a binary classification problem[26], network indexing [27], [28], and neural network applications [29] (Other interesting studies are [30], [31], [32], [33]).

In most of the review papers in the literature of frequent subgraph mining, the focus is on algorithms proposed for mining static networks. Except for [34], which covers three algorithms proposed for dynamic networks, the previous review papers neither cover nor categorize the algorithms proposed for dynamic and temporal networks (refer to Section 2). Some of the algorithms proposed for dynamic and temporal networks are an extension or modification of the algorithms developed for static networks. Therefore, after reviewing the algorithms proposed for static networks, the algorithms proposed for mining dynamic and temporal networks are discussed. Furthermore, one of the main challenges that current popular algorithms face is that they are developed for mining network datasets that can fit into the main memory or be processed by processing units of machines the data stored on. However, with the increasing growth of network data available, for example from high throughput sequencing technologies in bioinformatics,

social network platforms, communication and transportation networks, or sensor-based collected network data, the development of algorithms capable of managing disk-based and streaming data or utilizing parallel computing are in essential need. In [34], the importance of and need for the development of parallel algorithms are noticed. However, just four of the algorithms proposed in the literature are reviewed. Other review papers slightly focus on the parallel algorithms. Therefore another contribution of this survey would be the review of papers proposed in the literature to tackle the challenges of I/O-bound or CPU-bound.

1.2 Related But Out-of-Scope Work

The mining of networks at mesoscale is a vast area of research and study. Some of the very well-known problems in network mining belong to this area. A subcategory of mesoscale mining of networks is subgraph similarity search (for the definition of network-related concepts, refer to Appendix A, available in the online supplemental material). In this problem, a set of networks or a giant network is searched for a specific subtree or subgraph, *query subtree/subgraph* (and it is called reverse similarity search if the objective is to find all the networks in a set of networks which are a subgraph of the query tree/network) [35], [36], [37]. Other very popular subcategories of network mining at mesoscale are link mining and prediction [38], [39], [40] and clustering or community detection in static and dynamic networks [41], [42], [43], [44]. Some of the approaches toward this problem are through mining cliques (or pseudo-cliques) to identify nodes that are densely connected together. For example, MiMAG [45] (also refer to [46]) is one of these approaches proposed for mining dense clusters in multi-layer networks with edges labeled. Although one of the steps in these categories is the detection of subgraphs, because they do not focus on more general types of subgraphs, the network or subgraph isomorphism problem is not covered, and their objective is not to detect frequent subgraphs, they are not covered in this survey. Mining frequent trees in forests (a database of trees) is another very relevant area of research not covered in this paper. In frequent tree mining problems, the objective is to discover all or some of the frequent trees in a forest or a single large tree. As an example, interested readers may refer to TREEMINER [47] as a popular algorithm and [48] for a survey of algorithms proposed for mining frequent subtrees. Another set of problems at the mesoscale (not covered in this survey) is detecting the heaviest subgraph(s) in temporal or static networks. This problem is defined as mining the (top k) subgraphs with the highest score (using a scoring function defined over attributes of vertices or edges). For example, MEDEN [49] is an algorithm proposed for detecting $top-k$ *heaviest* subgraphs in a temporal network. The main application of these algorithms is in communication or transportation networks in which the objective is to find subgraphs with the highest traffic (over sub-intervals of time in temporal networks). Anomaly detection, which might be based on mesoscale mining of networks, is another popular area of research in network studies not covered in this survey. It is defined as mining unusual or rare appearance, disappearance, behaviors, or attributes of vertices, edges, or subgraph in a single or a set of static and temporal networks [50], [51], [52], [53], [54], [55], [56]. For a similar list of sub-domains of network mining approaches, please refer to [48].

1.3 Frequent Subgraph Mining Applications

The first generation of frequent subgraph mining algorithms has emerged mainly to solve the chemo-informatics domain problems, such as evaluating the chemical compounds' toxicology and carcinogenicity [57], [58], [59]. In this problem, each molecule is modeled as a network. The vertices of the network represent the atoms of the molecule, and edges represent the chemical bonds between each pair of atoms. The network might be attributed. The labels of edges indicate the type of chemical bond, and the labels of vertices indicate the atom labels or other characteristics such as chemical charges [4]. A data set of molecules is represented as a data set of graph transactions, in which each transaction models one molecule. The problem is defined as finding molecular substructures in common among a predefined percentage of molecules.

However, a simple search in academic search engines shows that applications of frequent subgraph mining have been generalized to various disciplines over the last two decades. In all these applications, the entities of interest are modeled as networks. Based on the domain of study, these entities would be different. Nonetheless, the problem definition is the same: detection of patterns observed in at least a pre-defined number or percentage of the networks in the data set. In the chemo-informatics applications, the entities are chemical compounds [57], [58], [59]. In health informatics applications related to disease classification and prediction, the entities might be patients' hospitalization [60], [61] or imaging records [62], [63], [64]. In the public health domain, different sources of information might be used for creating network transactions, for example, online threads [65], or geo-sensory, meteorological and air quality data [66], [67]. In bioinformatics application, different biological networks might be used for mining frequent subgraphs [68], such as RNA substructures [69] and protein-protein interactions [70]. The applications of frequent subgraph mining in social network analysis focus mainly on belief and intention inference and interaction analysis of users collected from different online platforms [71], [72], [73]. One of the domains adopted frequent subgraph mining extensively is computer vision, focusing more on image classification [74], [75], [76] and action and event recognition [77], [78], [79]. In these applications, images are represented as networks, for example, using a region adjacency graph [80]. In this representation, the sub-regions in each image are used as vertices, and spatial relations among sub-regions are used for edge description. These networks are then mined for the identification of subgraphs frequent in different image classes. Another set of applications that has grasped frequent subgraph mining algorithms is malware detection [81], [82] and intrusion detection [83] systems. In these applications, the (intrusion and non-intrusion) transactions [84], and system call traces are represented as networks [85]. The set of networks are used as the input data to the frequent subgraph miner. Then, the extracted frequent

TABLE 1
Some Examples of Domains/Problems of Frequent Subgraph Mining Applications

Domain	Problem
Chemo-informatics	Chemical compounds classification [86] Chemical toxicity prediction [57], [58], [59] Biochemical reaction analysis [87]
Health informatics	Public health [65], [66], [67] Disease prediction & classification [62], [63], [64] Disease progression and survival prediction [60], [61]
Bio-informatics	RNA substructure mining [68], [69] Conserved substructures mining [70]
Social networks	Trust and deceptive behaviors [71], [73] Knowledge-sharing [72]
Computer vision	Image classification [74], [75], [76] Image Clustering [88] Action & event recognition [77], [78], [79]
Security	Malware detection [81], [82], [85], [89]

TABLE 2
Summary of Frequent Subgraph Mining Review Papers

Review paper	Year
State of the art of graph-based data mining [90]	2003
A quantitative comparison of the subgraph miners MoFa, gSpan, FFSM, and Gaston [91]	2005
Discovery of Frequent Substructures [92]	2006
Frequent pattern mining: current status and future directions [93]	2007
Mining Graph Patterns [94]	2010
A survey of graph mining techniques for biological datasets [95]	2010
A comparative survey of algorithms for frequent subgraph discovery [96]	2011
Building blocks of biological networks: a review on major network motif discovery algorithms [97]	2012
A survey of frequent subgraph mining algorithms [48]	2012
Performance Evaluation of Frequent Subgraph Discovery Techniques [98]	2014
Frequent Subgraph Mining Algorithms – A Survey [99]	2015
A qualitative survey on frequent subgraph mining [34]	2018

subgraphs are used for downstream analysis, such as classification and discriminatory pattern analysis [81], [85]. Some of the applications of frequent subgraph mining in different domains are listed in Table 1.

1.4 Organization of the Survey

In Section 3 and Appendix A, available in the online supplemental material, the concepts used throughout this paper, and a classification of input data to the frequent subgraph mining algorithms are introduced. Section 2 reviews some of the related surveys available in the literature and the contribution of this paper. In Section 4, the algorithms proposed for mining different types of network data are reviewed. This survey includes a section for the algorithms proposed for mining big network data employing disk-based, parallel, and distributed strategies (Section 5). Then, publicly available tools are discussed in Section 6. The paper concludes with the limitations, critiques, and future directions of the field in Section 7.

2 Related Surveys and Our Contribution

The problem of frequent subgraph mining has attracted substantial attention in the last two decades and is considered as the essence of mining network data [48]. And, consequently, there have been several studies in which the proposed algorithms and applications are reviewed. A list of these review papers is provided in Table 2.

One of the challenges of evaluating and comparing different algorithms is that they are not written and implemented in a common code with the same level of experience and expertise of developers. So, to provide a better comparison, some of the review papers re-implemented different algorithms in the same framework by the same developers. The comparison made adopting this approach is very fair and reliable and can result in significant insights that cannot be easily acquired with a qualitative approach. For example, [91] compared four of the well-known algorithms for network-transaction setting employing this approach (also [98]). They conclude that strategies adopted to prune duplicate candidates are not necessarily the most critical factor, and algorithms should be judged not just by their speed but also their functionalities. Due to the time required to re-implement the algorithms, this approach cannot be applied to a large number of algorithms. Therefore, most of the review papers in Table 2 compared the algorithms descriptively and based on the theoretical advantages of strategies adopted by different algorithms for candidate generation and frequency calculation. In our knowledge, the review paper written by [90] is the first review of the algorithms developed for mining frequent subgraphs. This review paper categorizes the mining approaches into five groups, greedy search-based approaches, inductive logic programming, inductive database-based algorithms, network theory-based approaches, and kernel function-based approaches. Network theory-based approaches have had the most impact on the next generations of algorithms proposed for network-transaction settings. Some of the most well-known algorithms proposed in the literature are covered in the book written by Cook and Holder [100]. Chapter 5 [92] of this book reviews algorithms in network-transaction settings and categorizes these algorithms into two groups, Apriori-based approaches and pattern-growth approaches (also refer to [93], [99]). We follow the same categorization scheme for the classification of algorithms proposed in the literature for mining frequent subgraphs in a database of static networks. A similar categorization is also performed in [96], however, in this paper, the authors considered a larger number of algorithms, and classified them based on their settings (single network or network-transaction), the search strategy adopted by algorithms (breadth-first or depth-first search strategy), and the completeness of the frequent subgraphs mined by each

algorithm. They also summarized the representation of networks, the candidate generation, and frequency evaluation of candidates of different algorithms. [48] provides a detailed review of some of the most well-known algorithms, including adopted strategies by each algorithm for candidate generation and support computation (for both frequent subtree and subgraph mining problems) (also refer to [34]).

3 Terminology and Network Classification

To better understand the algorithms discussed in this survey, it is required to know about the foundational concepts of networks and concepts related to the literature of frequent subgraph mining, such as graph and subgraph isomorphism problems, pruning mechanisms, different concepts of frequency and various types of subgraphs. The terminology and concepts used throughout this survey are defined in Appendix A, available in the online supplemental material. Besides, Appendix A, available in the online supplemental material, briefly reviews the common format of input data to the frequent subgraph mining algorithms as well.

The algorithms proposed in the literature for mining frequent subgraphs in different settings can be categorized based on the characteristics of subgraphs. For example, [101] provides three dimensions based on the type of subgraphs mined: topology (acyclic subgraphs such as paths and trees, and cyclic subgraphs), frequency-based (closed or maximal), and relations between different embeddings of a subgraph (non-overlapping or partial overlapping). Considering the discussed concepts in Appendix A, available in the online supplemental material, the algorithms are different in terms of the temporality of the data, how they cope with complexities of the graph and subgraph isomorphism problems, methods adopted for candidate generation, the approach taken for calling a subgraph frequent (exact\inexact isomorphic, exact\inexact frequency, and general\specific), and the approach adopted for computing frequency or support. In the following, some of the well-known algorithms in the literature proposed for mining frequent subgraphs in graph-transaction setting are described. Furthermore, a detailed discussion of other algorithms in each mining category is provided in Appendices B and C, available in the online supplemental material. The problem of interest is frequent subgraph mining. In this paper, we focused on the algorithms proposed for mining frequent patterns in a dataset of networks. Based on the temporality of the data, the networks are represented either as sets or sequences of networks. In the next level, algorithms are classified based on the most significant differences observed among algorithms proposed in the literature. Fig. 1 visualizes this taxonomy and provides sections and subsections covering each category of mining approaches. In the following sections, when we are reviewing the algorithms proposed in the literature, a table is provided representing different characteristics of algorithms. These characteristics are:

- whether algorithms are exact isomorphic or *minor* topological variations or noises among embeddings of a subgraph are accepted,
- whether the algorithms mine all the frequent subgraphs (i.e., the complete set of frequent subgraphs), and
- whether the algorithms are limited to a specific class of subgraphs (such as closed or maximal), or mines all classes of subgraphs.

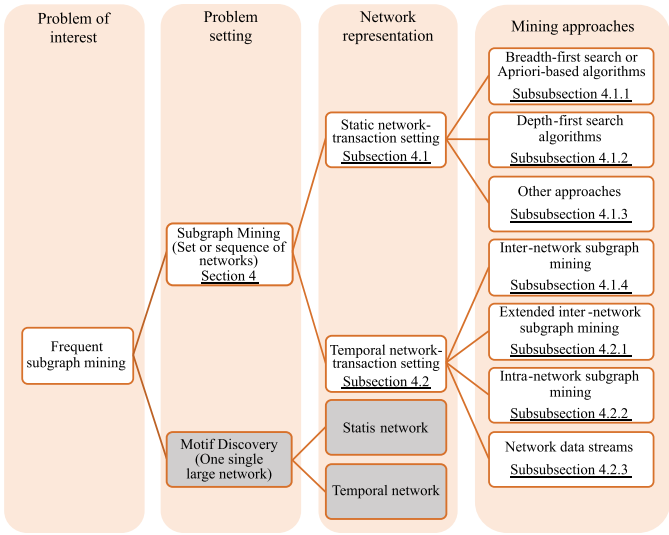

Fig. 1. Classification of frequent subgraph mining algorithms. The boxes grayed out are not covered in this survey. Each mining category is further discussed in Appendices B and C, available in the online supplemental material.

4 Subgraph Mining

4.1 Static Network-Transaction Setting

The proposed algorithms in the literature of network-transaction settings adopt different candidate subgraph generation and frequency computation strategies. The general approach is to generate candidate subgraphs from already identified frequent subgraphs and compute the frequency of candidate in the network database. The frequent subgraphs identified in each iteration are used for the generation of the set of candidates for the next iteration, and these iterations are performed until no further frequent subgraphs are identified. This problem can be formally defined as follows.

> Problem Definition. *Given a network database* $DB = \{\langle i, N_i\rangle \mid i = 1 \ldots n\}$ *composed of* n *network transactions Fig. 2), and a minimum support* $min\_supp \in [0, 1]$, *how we can mine the set of subgraphs appear in at least* $min\_supp \times |DB|$ *number of transactions of* DB.

The networks in the database may be directed/undirected and labeled/unlabeled. And the objective of an

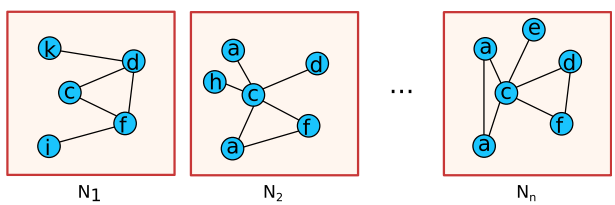

Fig. 2. A network database D composed of n independent undirected labeled networks.

algorithm is either mining all the frequent subgraphs, or a particular class of subgraphs (such as maximal, closed, or induced). In addition, the algorithms may compute the exact frequencies of subgraphs or just provide an approximation of the frequencies. Besides, some of the algorithms output a complete list of frequent subgraphs in the database, and some only mine a subset of the frequent subgraphs.

The algorithms proposed for subgraph mining in static network databases can be categorized into two main categories based on the adopted methods for candidate generations; Apriori-based algorithms and pattern growth algorithms. In Apriori-based algorithms, the candidates at iteration i are generated by merging two subgraphs considered to be frequent at iteration $i-1$. The size of subgraphs at each iteration generally increases by one (size is defined as the number of edges or vertices in different algorithms). To generate a complete list of candidate subgraphs at iteration i, all the frequent subgraphs of size $i-1$ should be known. Therefore, the Apriori-based algorithms follow a breadth-first search strategy (also called level-wise candidate generation). The pattern-growth algorithms start from one subgraph (which can be a single vertex or edge) and mine all the children of this subgraph iteratively by adding a new vertex or edge in each iteration (this strategy is called depth-first search, or DFS). Because the pattern-growth algorithms do not depend on all the known frequent subgraphs at the previous iteration, there is no requirement for detecting all the frequent subgraphs at each iteration. Therefore, pattern-growth algorithms can be implemented with both breadth-first or depth-first search strategies. These two approaches to mining frequent subgraphs in the network databases are explained in detail in [92]. For a summary of the algorithms reviewed in this section, refer to Table 3.

TABLE 3
Summary of Algorithms Developed for Subgraph Mining in Static Network-Transaction Setting

Algorithm	Exact isomorphic	Complete	General
Breadth-first search or Apriori-based algorithms			
AGM [58]			-
AcGM [102]			
FSG [59]			
Inokuchi *et al*.[103]			-
gFSG [104], [105]			
Vanetik *et al*.[106]			
Depth-first search algorithms			
gSpan [25], [107]			
CloseGraph [108]			-
MoFa [4]			
FFSM [109]			
SPIN [110], [111]			-
MARGIN [112]			-
Other approaches to frequent subgraph mining			
GASTON [113]			
TSMiner [114]	both		
CLOSECUT & SPLAT [115]			-
LEAP [116]			-
MULE [117]	-		-
gPrune [118]			-
GraphSig [119]			-

**: This algorithm is discussed in Appendix B, available in the online supplemental material.*
Exact isomorphic: if the algorithm mines the exact isomorphic subgraphs.
Complete: if the algorithm mines all the frequent subgraphs.
General: if the algorithm mines all the general types of subgraphs or limited to a special class, such as induced, maximal, or closed.
"both": the algorithm can mine the exact/inexact isomorphic subgraphs.

4.1.1 Breadth First Search or Apriori Based Algorithms

As discussed, the algorithms in this category generate candidates of each iteration by merging frequent candidates of the previous iterations. One of the most well-known algorithms in this category is FSG [59], [120], [121]. This algorithm uses a sparse network representation of the network database. The candidate generation in FSG is edge-based and composed of joining (including self-joining) two frequent subgraphs of size k, which have identical subgraphs of size $k-1$, called *core*. Therefore, to join two subgraphs, first, the core should be identified, then two subgraphs should join to form a candidate. Some of the candidates may be removed in this step to meet the downward closure property. By recording the history of frequent subgraphs in each iteration, these steps can be expedited. Also, note that two subgraphs with identical cores can be joined differently and produce multiple candidates. For frequency counting, FSG uses an optimized approach. In each iteration, the list of transactions containing the frequent subgraph is stored. Then after producing a candidate from two frequent subgraphs, the intersection of the list of transactions containing both frequent subgraphs is created. If the length of this intersection is less than the threshold, the candidate subgraph can be pruned. Otherwise, the transactions inside the intersection list can be examined for computing the exact frequency. The FSG uses canonical labeling for graph isomorphism. However, to reduce the combinatorial space, vertex invariants such as labels and degrees of vertices and the neighbor lists of vertices are used. An extended version of FSG, gFSG [104], [105], is proposed for mining datasets of geometric networks. This algorithm is discussed in further detail in Appendix B.1.1, available in the online supplemental material.

Another well-known algorithm in this category is Apriori-based Graph Mining (AGM) [58] developed for mining frequent induced subgraphs (the performance of the algorithm has been improved and introduced as AcGM, the new version can find both general and induced subgraphs [102]). Most of the algorithms proposed in the literature mine the network database for general subgraphs. The algorithms proposed for mining induced subgraphs claim that not all the frequent subgraphs necessarily preserve the original input network structure. Mining the network database for induced subgraphs both maintain the structure of the original edges and simultaneously reduce the computational complexity of subgraph isomorphism, because the algorithm does not need to search for non-induced frequent subgraphs [103]. In AGM, the adjacency matrix of all frequent induced subgraphs is vertex-based sorted. Two frequently induced subgraphs are joined to form a candidate if they differ by only one column and one row. The size of this new candidate is one vertex more than the two parent subgraphs. To prevent the redundant generation of candidates, the subgraph with the smaller code is considered the first

subgraph and the second subgraph is joined to the first subgraph (code of a subgraph is defined as the concatenation of elements of the upper triangle of the adjacency matrix along the columns). The vertex-sorted adjacency matrix $X(s)$ of the generated candidate s is called *normal form*, represented as $code(X(s))$. The normal form of a matrix is not necessarily unique. Therefore, in AGM, for correct frequency calculation, *canonical form* of the adjacency matrices $X_c(s)$ is introduced, which for a set of the normal form of identical subgraphs $NF(s)$, is the one with the minimum code.

$$X_c(s) = \arg\min_{X \in NF(s)} code(X(s)). \tag{1}$$

In Vanetik *et al.* [106], an algorithm is proposed which uses edge-disjoint paths for lexicographical ordering and candidate generation. A brief description of this algorithm is provided in Appendix B.1.1, available in the online supplemental material.

Apriori-based algorithms have been very successful in mining the complete set of general or induced subgraphs. Some of the algorithms in this category are customized to mine disconnected subgraphs, subgraphs with self-loops and multi-edges, and subgraphs in a dataset of directed networks [58], [59], [102]. However, the algorithms in this category are considered expensive due to the adopted level-wise candidate generation strategy [92]. In addition, for frequency counting, either the subgraph isomorphism should be performed in each iteration, or the location of all the instances of frequent subgraphs at successive iterations should be recorded. The former strategy is computationally expensive, and the latter increases the memory requirements [101]. The tracking of frequent subgraphs does not require that much memory in algorithms adopting a depth-first search strategy, which is discussed in the next part.

4.1.2 Depth First Search Algorithms

The DFS-based algorithms are developed to improve efficiency and alleviate the challenges associated with Apriori-based algorithms. One of the most well-known algorithms in this category is gSpan [25], [107]. The gSpan utilizes two novel constructs; DFS lexicographic order and minimum DFS code, which make the canonical labeling of networks more efficient for DFS.

Given a support threshold, the gSpan can discover all the frequent subgraphs in a network database. Adopting a DFS strategy, gSpan searches for frequent subgraphs starting from each 1-edge toward all the children of the subgraph. gSpan continues searching until either the support of the subgraph is less than the predefined threshold or the DFS code is not the minimum code. In the latter case, it means that the tree with the minimum DFS code is already found (and all of its children are also evaluated) [108]. The gSpan can be simply modified to mine frequently induced subgraph [107]. CloseGraph [108], an extended version of gSpan, is another algorithm in this category that mine all the closed frequent subgraphs. One, not desired approach, is to generate all the frequent subgraphs and then prune non-closed subgraphs. However, CloseGraph is modified to terminate the iterations for preventing the generation of non-closed subgraphs (through "non-closed graph pruning"). It uses a mechanism (equivalent occurrence) to check for closed-ness of subgraphs and their children and remove them as early as possible if they are not closed.

MoFa (or MoSS) [4] is another algorithm in this category proposed for the identification of frequent subgraphs and contrast subgraphs with minor modifications. Contrast subgraphs are defined as subgraphs frequent in one set of network transactions and infrequent in another set. The approaches proposed for contrast subgraph mining are generally using two support thresholds. One defines the minimum support that subgraphs should have to be considered frequent in the first set. The second one represents the maximum support that subgraphs can have in the second set of transactions to be considered infrequent.

FFSM [109] is another popular algorithm in this category which uses a novel approach to produce canonical forms. Given the adjacency matrix A for a graph N, the $code(A)$ is defined as sequence obtained by concatenating the elements located in the lower triangle of A

$$code(A) = a_{1,1}a_{2,1}a_{2,2}\ldots a_{n,1}\ldots a_{n,n-1}a_{n,n}, \tag{2}$$

where $a_{i,j}$ represents an element of A located at row i and column j. Because the adjacency matrix representation of a network is not unique, the canonical form of network N is defined as

$$A_c(N) = \arg\max code(A(N)). \tag{3}$$

The maximality is checked based on the lexicographic order among the codes obtained from different representations of the adjacency matrix. The matrix $A_c(N)$ is called *canonical adjacency matrix CAM)* of N. FFSM proposes two *FFSM-Join* and *FFSM-Extension* operations on CAM for candidate generation, and a data structure for tracking frequent subgraphs to avoid expensive subgraph isomorphism problem. The developers of FFSM show that these improvements have been able to outperform gSpan, on both real-world and synthetic datasets. SPanning tree-based maximal graph mINing (SPIN) [110], [111] and MARGIN [112] are two other algorithms in this category proposed for mining maximal subgraphs and discussed in further detail in Appendix B.1.2, available in the online supplemental material.

It is shown in multiple studies that algorithms of this category have been successfully outperformed Apriori-based algorithms [25], [92], [113]. Because it is not required to mine and keep all the $(k)-$size frequent subgraphs to create $(k+1)-$size candidates. Besides, the frequency computation and verification can be performed more efficiently, as the frequency of children candidates can be derived from the frequent parent subgraph frequency. However, even with these modifications, the algorithms of this category are not challenge-free. First, the graph isomorphism still exists. Also, the algorithms designed to mine the complete set of general subgraphs produce a large number of frequent subgraphs. For example, in a dataset of 422 chemical compounds with $min\_supp = 0.05$, there are about 1,000,000 frequent subgraphs [108]. Although all these subgraphs are considered frequent at this support level, not all of them are necessarily significant or important for the application of

interest. Increasing the support threshold might eliminate some of the significant but not frequent enough subgraphs. Decreasing the support threshold would increase the mining cost exponentially and may result in many frequent but not significant enough subgraphs. Therefore, some of the other algorithms proposed in the literature taking these challenges into account and are designed to tackle the isomorphism problems more efficiently. Some other algorithms are proposed to mine frequent subgraphs that maximize (minimize) some user-defined objective (loss) functions or enable users to control the mining process by defining structural constraints over the mined subgraphs. These algorithms are covered in the following subsection.

4.1.3 Other Approaches to Frequent Subgraph Mining

The algorithms in this category might still use one of the search strategies discussed in the previous two subsections. However, they generally employ more unique strategies for subgraph mining that differentiate them from other algorithms discussed previously. Therefore, we decided to put these algorithms in a third category. One of the most well-known algorithm in this category is GASTON (GrAph/Sequence/Tree extractiON) [113] (assumed to be a pattern-growth approach [92]). This algorithm first mines all the frequent paths, then trees, and ends with the mining of subgraphs. The main reason is to reduce the computational complexity of an expensive process in subgraph mining as the mining of frequent paths and trees is less complex. Algorithms like GASTON (or SPIN discussed earlier) show their best performance when the frequent subgraphs are trees. TSMiner [114] is another approach developed for mining *frequent topological structures* based on the concept of *topological minor* focusing on replacing independent paths (paths with no inner nodes in common) with edges using appropriate relabeling functions to preserve the edges' information. The topological structures can be obtained from topological minors. They show that the frequent connected subgraph mining problem is a special case of frequent topological structure mining problems. The mining is performed in two steps. First, frequent tree-topological structures are mined (using the approaches introduced in GASTON[113]), then using the frequent tree-topological structures, frequent network-topological structures are mined. The frequent tree-topological structures are the spanning trees of the frequent network-topological structures.

In [115], two algorithms, CLOSECUT and SPLAT, are proposed for mining relational networks. For network N, the edge-cut, E_c, and edge-connectivity, (N) are defined as

$$\begin{aligned} E_c &= \{E' | N(V, E/E') \text{ is disconnected}\} \\ \Rightarrow \quad (N) &= \min_{\mathcal{E} \in E_c} |\mathcal{E}|. \end{aligned} \tag{4}$$

Then, the proposed algorithms discover all the closed frequent subgraphs with minimum edge connectivity of K. These algorithms are different based on the approach they adopt for creating new candidates (pattern-growth versus pattern reduction, respectively). The nodes in relational networks have distinct labels. Therefore, frequent itemset mining approaches can be applied to mine relational networks using edges as items. However, the considered constraint, the minimum edge connectivity of K, is not downward closed, and therefore, the algorithms should adopt appropriate heuristics to mine closed connected subgraphs with the minimum edge connectivity of K.

MULE [117] and the algorithm proposed in [122] are other relevant algorithms. The MULE is a domain-specific algorithm proposed for mining frequent subgraphs in a dataset of a few large directed networks. And, [122] proposes an algorithm for mining frequent subgraphs in outerplanar network-transactions. These two algorithms are briefly described in Appendix B.1.3, available in the online supplemental material.

In many subgraph mining applications, we are interested in subgraphs maximizing or minimizing an objective function or satisfying some constraints. The simple approach is to mine all the frequent subgraphs first, and then among the frequent subgraphs, the ones which maximize or minimize the objective function or meeting the constraints are selected. However, this approach is not very efficient. The ideal case is to mine the frequent subgraphs optimizing objective function or satisfying the constraints in the running time. The gPrune [118] is a constraint-based framework for mining frequent subgraphs in a network database in which the discovered subgraphs are satisfying some structural constraints. A structural constraint is expressed as a boolean predicate, which either a subgraph, s, meets or not. The constraints considered for the gPrune are:

- *Density ratio*: $2 \times |E(s)|/(|V(s)|(|V(s)|-1))$
- *Density*: $|E(s)|/|V(s)|$
- *Diameter*: the maximum length of the shortest path between any pair of nodes
- *Edge vertex) connectivity*: refer to Equation (4)

The gPrune is based on gSpan with two extra checks in the pattern-growth phase. A more in-depth discussion about this algorithm is provided in Appendix B.1.3, available in the online supplemental material.

Instead of constraint-satisfying algorithms, some algorithms are developed to find frequent subgraphs to minimize or maximize an objective function. LEAP is one of these algorithms proposed in [116]. Given a (potentially non-monotonic) user-defined objective functions , F, and an user-defined threshold θ, LEAP is able to mine the set of subgraphs such that $F(s) \geq \theta$, or, the subgraph s^* that maximizing the objective function

$$s^* = \arg\max F(s). \tag{5}$$

This approach is based on this idea that we might be able to use the correlation between the significance similarity (defined based on the objective function) and structural similarity to mine frequent subgraphs optimizing the objective function. This algorithm uses *structural proximity pruning*, meaning similar patterns in structure show similar frequencies and significance, and *frequency association*, meaning that significant patterns also have higher ranks in a list of subgraphs ranked based on their frequency. The GraphSig [119] is another algorithm in this category combining the mining of statistically significant and frequent subgraphs and is briefly discussed in Appendix B.1.3, available in the online supplemental material.

The problem of mining frequent patterns in a set of static networks has been around for a few decades. Numerous research studies are devoted to the extension of theoretical foundations and applications of this problem. Although some of the algorithms proposed can be perfectly applied to many applications, there are still some challenges associated with frequent pattern mining in static network-transaction settings. First, some of the systems modeled by these networks are changing over time. The static representation of these systems might not capture the underlying dynamics. Therefore, the frequent patterns identified by these algorithms might not represent the actual frequent temporal patterns in the systems that the static networks represent. Furthermore, even with the progress made to avoid graph and subgraph isomorphism problems and memory limitations, the adverse computational impacts and the complexities associated with these problems still may limit the proposed algorithms' applications. Therefore, there have been two streams of research to tackle these problems. In the past few years, there has been an increasing interest to include temporality into the network modeling of systems where their dynamics cannot be ignored. Based on the problem and the dataset of interest, there have been several approaches to generalize the problem of frequent subgraph mining to temporal networks. In parallel, some of the studies focus on implementing frequent subgraph mining algorithms using different techniques to solve the computational complexities and memory requirement problems. In the next two sections, first, we review the algorithms proposed for mining frequent patterns in temporal networks. Then, the algorithms proposed for CPU- and I/O-bound problems are reviewed and discussed.

4.2 Temporal Network-Transaction Setting

This category of subgraph mining algorithms can be considered as an extension of frequent subgraph mining in the static network-transaction setting. Therefore, some of the approaches proposed in the literature are a modified version of algorithms discussed in Section 4.1. In the static version of subgraph mining, we were interested in the exact or approximate version of all or a subset of frequent subgraphs. And, a frequent subgraph was defined as a subgraph which has appeared at least in more than a predefined number of network transactions. The embeddings of the subgraph are topologically identical or similar enough. Also, it should be noted that although vertices and edges on network transactions might have identical labels, network transactions are entirely independent. In the dynamic or temporal version of the same problem, on the other hand, network transactions are not independent. In fact, they represent the changing relations among the same system components over a predefined dimension, such as time. In this case, network transactions can be considered as a sequence of small networks or time-series networks. These changes might be realized through insertion, deletion, and substitution of vertices, edges, and labels. The addition of this dimension and sequentiality of network transactions make the subgraph mining much more complicated.

The modeling approaches adopted in temporal network-transaction settings are very similar to those adopted in motif discovery algorithms in a single large network. For a detailed discussion of motif discovery in static and temporal networks, refer to [123]. There are some minor differences that can be helpful to differentiate these two problems. First, the networks in the temporal network-transaction setting usually are small, with less than a hundred or a few hundred nodes. However, the networks in the motif discovery problem are composed of thousands or millions of nodes. Although the networks are labeled in some of the motif discovery algorithms' applications, the focus is generally more on the structure or the structural dynamics. In temporal network-transaction setting, the networks are typically labeled. Even in some of the algorithms proposed in the temporal network-transaction setting, the nodes are uniquely labeled. In this case, the complexities of both graph and subgraph isomorphisms significantly reduce and would be quadratic in the number of nodes [124]. In most of these applications and algorithms, the emphasis might shift toward which components specifically interact with one another over time. Nonetheless, there are no well-defined boundaries between these two problems. Some of the algorithms proposed for motif discovery in a single large network can be customized for network-transaction setting applications. In temporal network-transaction setting, subgraphs are considered frequent if the number of topologically isomorphic embeddings of a subgraph preserving identical temporal changes is more than a user-defined threshold. In the following, some of the proposed algorithms for subgraph mining on temporal networks are discussed. For a summary of the algorithms reviewed in this section, refer to Table 4.

4.2.1 Inter Network Subgraph Mining

In this subclass, the inter-network transformation patterns are defined as changes that convert a network N_{t_i} to the next network in the sequence $N_{t_{i+1}}$. Then, the problem of this subclass can be expressed as follows:

> Problem Definition. *Given a network database* $DB = \{N_{t_1}, N_{t_2}, \ldots, N_{t_n}\}$ *composed of an ordered sequence or time-series of* n *networks representing the states of a system at* n *time-points Fig. 3), a minimum support* min_supp*, how we can mine the patterns appearing in at least* $min\_supp \times |DB|$ *number of inter-network transformation patterns.*

[125] proposed one of the algorithms in this subclass. In [125], a sequence of n networks is considered as $N_{ts} = \{N_1, \ldots, N_n\} = (V, E)$ in which $V = \bigcup_{i=1}^{n} V(N_i)$ and edges might be deleted or inserted over time. Then, the proposed algorithm predicts future interaction patterns using the discovered frequent subgraphs. In this algorithm, it is assumed that vertices are uniquely labeled and can appear maximum once in each timestamp. Because of the unique labeling of vertices, the graph isomorphism test would be trivial. The subgraph mining step of the algorithm is to extract the frequently closed subgraphs for each timestamp. The different ordered pairs of these subgraphs (also taking into consideration the delay between them) are then used to estimate the delay between their co-occurrences as a probability distribution. Because of the elimination of graph isomorphism problem, they have used a modified version of MAFIA

TABLE 4
Algorithms for Subgraph Mining in Temporal Network-Transaction Setting

Algorithm	Exact isomorphic	Complete	General
Inter-network subgraph mining			
Lahiri *et al*. [125]			-
PSEMiner [126]	both		-
You *et al*. [127]	both	-	
GERM [128]			
Ahmed *et al*. [129]			-
Uno *et al*. [130]	-		-
Extended inter-network subgraph mining			
GTRACE & *GTRACE-RS [131], [132], [133]			
FRISSMiner [134], [135]			-
Intranetwork subgraph mining			
EVOLVING-SUBGRAPHS [136]	-	-	-
Wackersreuther *et al*. [137]			
TimeCrunch [138]	-	-	-
Network data streams			
Cuzzocrea *et al*. [139]			
Inc-, Win-, & AdaGraphMiner [140]			-
Braun *et al*. [141]			
Aggarwal *et al*. [142]	-	-	-

*: *This algorithm is discussed in Appendix B, available in the online supplemental material.*
Exact isomorphic: if the algorithm mines the exact isomorphic subgraphs.
Complete: if the algorithm mines all the frequent subgraphs.
General: if the algorithm mines all the general types of subgraphs or limited to a special class, such as induced, maximal, or closed.
"both": the algorithm can mine the exact/inexact isomorphic subgraphs.

[143], a frequent itemset mining approach from a transactional database, for extraction of frequent subgraphs.

A compression-based (instead of a frequency-based) approach in this category is proposed in [127] (also refer to [144]), developed for extracting structural transformation in a sequence of networks. It is composed of two main steps, in the first step, it extracts a set of rules in which, first, the maximum common subgraph between every two consecutive networks is extracted (e.g., N_i and N_{i+1} in $N_{ts} = \{N_1, \ldots, N_n\}$ where $1 \leq i < n$) and then the edges removed and added from and to the first network, N_i, which in combination with the maximum common subgraph have created the second network,N_{i+1}, are recorded (called *rewriting rules*). In other words, rewriting rules, $RR_{i,i+1}$, are a set of network transformations that convert a prior network N_i to a posterior network N_{i+1}

$$N_{i+1} = N_i \oplus RR_{i,i+1}. \tag{6}$$

In the second step, using the rewriting rules, $RR_{i,i+1}$, *description rules* are extracted used to describe the structural changes in a sequence of networks as a set of subgraphs repeatedly removed and added. The complexity of this problem is in finding the maximum common subgraph, which because it is assumed that vertices are uniquely labeled, is not very expensive. In [126] (also refer to [145] for more details), a similar algorithm, PSEMiner, is proposed for detection of structural (or close to) periodic behaviors of sequential networks. The networks in the sequence are composed of uniquely labeled vertices. Their algorithm is designed to mine closed subgraphs, and they define a *periodic subgraph embedding* as a set of subgraphs that appear in regular intervals in the network sequence. A subgraph is frequent if the number of timestamps that this subgraph appears in regular intervals is more than a predefined threshold. They also make their algorithm flexible by allowing some noises in the regularity in the appearances of embeddings. Considering the uniqueness of vertices' labels in sequential networks, they prove that mining periodic subgraphs in dynamic networks are in the time- and space-polynomial class. They also converted the frequently closed subgraph mining problem to a transactional itemsets problem and adopted MAFIA [143] to mine the frequent patterns.

GERM [128] is proposed for *graph evolution rule mining*. Similarly, authors of [129] and [130] propose algorithms for mining persistent maximal evolution paths and *preserving structures*, respectively. The problems defined in these papers are similar in nature to the ones discussed in this section. These algorithms are discussed in Appendix B.2.1, available in the online supplemental material.

Some of the algorithms of this subclass of the temporal network-transaction setting are limited to particular temporal patterns. For example, in [126], the periodic or near-periodic frequent patterns are mined. Or, the algorithm proposed in [130] mines the subgraphs forming a clique or a connected subgraph. Besides, in most of the algorithms in this subclass, the dataset represents one sequence of networks, where each network of the sequence represents the state of the system at a timestamp, and the series represents the system's dynamics over time. In a similar category of algorithms, the frequent transformation rules are of interest. However, instead of one sequence of networks, the dataset is composed of multiple sequences, and the complete set of patterns/sub-sequences are mined. In the next subsection, the algorithms proposed for this type of problem will be discussed.

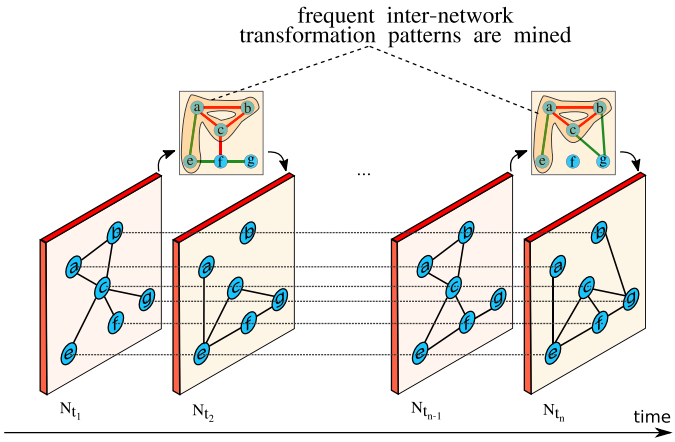

Fig. 3. A sequence or time-series of networks. In the subclass of inter-network subgraph mining, the frequent inter-network transformation patterns are mined. In this subclass, the transformation patterns, represented by insertion or deletion of vertices or edges, and changes in their associated labels, transform one network to the next network in the sequence. In this figure, edges inserted are shown with green, and edges deleted are shown with red, vertices, and the labels remained unchanged.

4.2.2 Extended Inter Network Subgraph Mining

The problem of this category is an extended version of problem discussed in Section 4.2.1. The extension is typically over the number of sequences in the network. Here, the network database $DS = \{S_i = \langle n_i^1, n_i^2, \ldots, n_i^{n_i} \rangle\}$ for $i \in \{1..s\}$ in which S_i represents the sequence i and n_i^j represents the jth observation of the sequence i. Therefore, the database is composed of a set of sequences. Each sequence is an independent time-series of networks similar to the sequence shown in Fig. 3. Then, the problem is defined as follows:

> Problem Definition. *Given a database of network sequences, DS, minimum support min_supp, how can we mine the patterns appearing in at least min_supp × |DS| number of sequences in DS.*

In this problem, similar to the problem of the previous subsection, the inter-network patterns are mined. However, the support is measured over all the sequences of the database, instead of one sequence.

In our knowledge, the first algorithm in this group is GTRACE [131], [132]. The general idea is that there is a database of sequential networks. The sequence i is composed of k_i observations (or *interstates*) of the network. The changes of interstates show the structural changes of the corresponding network over time in the form of vertex or edge insertion or deletion or label substitution. The changes of a network sequence are represented as transformation rules in two consecutive states (*intrastate*), and the sequence of all the consecutive transformation rules of a network sequence is called intrastate transformation sequence. Therefore, each network sequence can be replaced by its initial interstate n^1 and the intrastate transformation sequence. The GTRACE uses a *graph grammatical framework* to formulate these transformation rules. In GTRACE, the union network, N_u, of a sequence S_i is defined as follows:

$$N_u(S_i) = (\mathbb{V}, \quad) \ni$$
$$\mathbb{V} = \bigcup\nolimits_{j=1\ldots|S_i|} \{id(v) | v \in V(n_i^j)\}$$
$$= \bigcup\nolimits_{j=1\ldots|S_i|} \{(id(v), id(v') | (v, v') \in E(n_i^j)\}. \quad (7)$$

A sequence is called relevant if the corresponding union network of the sequence is connected. The objective is to mine the frequent subsequence of transformation rules (FTS), which have connected union network. To find the relevant FTS, any network mining algorithm can be used. It is assumed that identical labels are allowed. However, vertices have unique IDs in S_i. Then, a sequential pattern mining algorithm can be used for mining the FTS. Note that, due to the uniqueness of node IDs, the GTRACE can compute the consecutive transformation rules in polynomial time. The support for each transformation subsequence is computed as the number of S_is in the DS, which includes the subsequence. The modified version of GTRACE, GTRACE-RS [133] and Frequent Relevant, and Induced Subgraph Subsequence (FRISS) Miner [134], [135], are discussed in Appendix B.2.2, available in the online supplemental material.

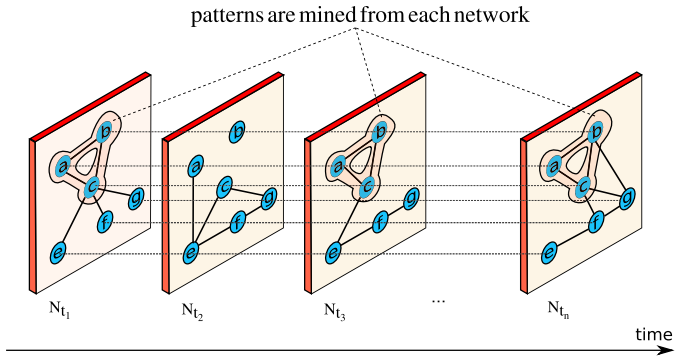

Fig. 4. A sequence or time-series of networks. In this subclass, the networks of the sequence are mined.

4.2.3 Intranetwork Subgraph Mining

A subset of algorithms in the temporal network-transaction setting are the ones that perform mining on each of the networks of the sequence. Therefore, the definition of the problem would be as follows:

> Problem Definition. *Given a network database $DB = \{N_{t_1}, N_{t_2}, \ldots, N_{t_n}\}$ composed of an ordered sequence or time-series of n networks Fig. 4), a minimum support min_supp, then how we can mine the patterns appearing in at least min_supp × |DB| number of networks in the sequence. In this subclass, the proposed algorithm mines the networks of the sequence, instead of the inter-network patterns.*

An algorithm from this category is proposed in [137]. In this algorithm, the vertices are identical over all the networks of the sequence. However, their labels can change. For mining frequent subgraphs, first, all the networks in the sequence are aggregated into one single network. The edge labels are concatenated too. If one of the edges does not exist in one or multiple networks in the sequence, a character is used to make the string concatenation complete. This aggregated network is called *summary graph*. Then, in the first step, all the frequent topological subgraphs of the summary graph are mined using motif discovery algorithms in single networks (In [137], the FANMOD [146] is used). From the mined frequent motifs, the subgraphs that follow the same temporal patterns of edges' insertions, deletions, or label substitutions are considered frequent dynamic subgraphs if their frequencies are more than a predefined threshold. The identification of temporal patterns is performed on the concatenated labels of edges using string algorithms. TimeCrunch [138] is another approach proposed for mining patterns in a sequence of networks and briefly discussed in Appendix B.2.3, available in the online supplemental material.

In all the algorithms discussed, including the algorithms proposed for the temporal network-transaction setting, it is assumed that the entire dataset of interest is available beforehand. Therefore, all the networks' simple frequent components, vertices and edges, can be identified and used for the next iteration of subgraph/sub-sequence candidate generation. Although this assumption that the dataset is accessible in advance holds in many applications, the algorithms discussed so far are not directly applicable to the

cases where the temporal data is generated over time and being gradually accessible. In addition to the challenges discussed so far in these types of problems, the continuity and unboundedness of the input data and unknown distribution of frequent subgraphs in the stream might be problematic. It implies that the subgraphs considered infrequent at some periods might be frequent in others [139], [141]. In this following subsection, some of the algorithms proposed to cope with these complexities are discussed.

4.2.4 Network Data Streams

Another category in this subclass is when the input network data is in the form of sequential networks streaming over time. The problem can be defined based on any of the three previous subclasses. However in this class of problems, designing an efficient data structure to keep track of observed subgraphs for later frequency evaluation is crucial. In [139], three different data structure, DSTree, DSTable, and DSMatrix (DS stands for Data Stream) are briefly reviewed, with the DSMatrix to be the most memory-efficient structure for network data streams (also refer to [141] in which the DSMatrix is proposed for mining frequent subgraphs in a stream of dense networks). These data structures are mainly used when data streaming is in the form of batches of data. Then, two edge-based algorithms are proposed to mine frequent co-occurring connected edges in the network streams. In the first algorithm, the frequent edges are mined first, and then disconnected edges are pruned in the second step. The second algorithm combines these two tasks in one step.

In [140], three algorithms are proposed for mining closed subgraphs in network streams; IncGraphMiner, WinGraphMiner, and AdaGraphMiner, respectively incremental, window-based, and adapting closed subgraph mining algorithms. In IncGraphMiner, batches are composed of a small set of networks. After a new batch arrives, the frequently closed subgraphs are mined, and the previous frequent subgraph repository is updated. In WinGraphMiner, a sliding window is considered. When a new batch of networks arrives, the frequent subgraph repository is updated, however, in cases where the considered sliding window is full, frequently closed subgraphs from the oldest batches might be deleted. Finally, the AdaGraphMiner employs an adaptive approach to the changes in the network stream by using different adaptive sliding window strategies. A probabilistic approach for mining frequent dense subgraphs in streaming edge sets is proposed in [142] and explained in Appendix B.2.4, available in the online supplemental material.

The problem of mining frequent patterns in network data streams is generally very challenging and complex. The algorithms proposed for this subclass either provide approximate estimations of frequencies or produce a subset or particular classes of frequent patterns and subgraphs (e.g., closed or dense subgraphs). As it is discussed in [142], one of the assumptions made in some of these algorithms is sparsity, meaning that even if the networks are composed of a large number of vertices and edges, the streams are in the form of batches of edges connecting a small subset of vertices.

TABLE 5
Algorithms Proposed for CPU-Bound and I/O Bound Problems

Name	Exact isomorphic	Complete	General
ADI-Mine [147]			
PartMiner & IncPartMiner [148]			
Nguyen *et al.* [149]			
[150]			
Parallelized gSpan & MoFa [151]			
FSM-H [152]			
Buehrer *et al.* [153]			
DB-FSG [154], [155]			
OO-FSG [156]			
MapReduce-FSG [157]			
MRFSM [158]			

*: *This algorithm is discussed in Appendix C, available in the online supplemental material.*
Exact isomorphic: if the algorithm mines the exact isomorphic subgraphs.
Complete: if the algorithm mines all the frequent subgraphs.
General: if the algorithm mines all the general types of subgraphs or limited to a special class, such as induced, maximal, or closed.
"both": the algorithm can mine the exact/inexact isomorphic subgraphs.

5 Algorithms for CPU-Bound and I/O Bound Problems

In general, frequent subgraph mining algorithms require extensive computational and memory resources. The computational resources are required for the graph and subgraph isomorphism evaluations and frequency computation. Recording the large number of candidate subgraphs generated and frequent subgraphs identified at each iteration, and the appearance list of subgraphs in the network transactions necessitates a large amount of memory. However, in most of the algorithms reviewed so far, it is assumed that the input data and the data generated during the mining process can be held in the main memory, and processing steps can be completed by computational resources available on the local machine.

An increasing amount of structured data is being generated every day on different biological, communication, transportation, and social media platforms. The modification of previous algorithms or the development of novel algorithms for mining frequent subgraphs that can tackle the input/output (I/O) and CPU-bound problems have been an avenue of extensive research in the past few years. Most of these algorithms focus on either I/O operations' challenges or computational processing limitations as the bottleneck of efficient frequent subgraph mining. Therefore, the strategies adopted by these algorithms for the successful mining of frequent subgraphs are inherently different. In this section, these algorithms are reviewed in detail. For a summary of the algorithms discussed in this section, refer to Table 5.

To alleviate I/O bound limitations, several approaches have been proposed. For example, in [147], the authors developed and proposed an index structure, called ADI (stands for adjacency index). This structure can be used with traditional subgraph mining algorithms for mining frequent subgraphs when the network database cannot be held in the main memory (they adopted gSpan, and in

combination with the index structure, they proposed ADIMine). In this case, the index structure represents the whole database. Instead of scanning the database or its projection, the index is used to check the support of edges, the networks that include them, and support the adjacent edge evaluations for candidate generation and subgraph expansion. The authors show that for a database composed of n network, the space complexity of ADI is

$$\mathcal{O}\left(\sum_{i=1}^{n}|E(N_i)|\right). \tag{8}$$

Some of the other algorithms in this category adopted a data partitioning approach. PartMiner [159] is the first approach proposed in this category. This algorithm is composed of two main phases, each including two steps:

- Phase 1:
  - Bi-partitioning network transactions into subgraphs
  - Grouping subgraphs into units
- Phase 2:
  - Mining frequent subgraph in each unit using traditional memory-based algorithms
  - Performing merge-join operation to aggregate frequent subgraphs

The major reason for partitioning is to reduce the complexity of the problem for each partition. Therefore, the memory-based algorithms proposed in the literature for mining frequent subgraph can be adopted (In [159], GASTON [113] is used). Given the $supp$ threshold, the minimum support for the subgraphs to be considered frequent, and k, the pre-defined number of units, subgraphs in each unit are considered frequent if their frequencies are more than $supp/k$. To reduce the merge-join operation's complexity, PartMiner minimizes the connectivity among the subgraphs of the units. The frequent subgraphs discovered in each unit are recursively merged to find the complete set of frequent subgraphs. In [149], it is discussed that the PartMiner may not produce the correct complete set of frequent subgraphs, and a new algorithm based on a horizontal partitioning framework is proposed to accomplish that objective. Instead of the bi-partitioning approach adopted in PartMiner, the proposed algorithm in [149] adopts a clustering-based approach for partitioning. The authors define k partitions (or fragments). The centroid of partition i is defined as

$$C_i = \{(I_1, w_1), (I_2, w_2), \ldots, (I_m, w_m)\}. \tag{9}$$

Where, I_j and w_j (for $j \in \{1..m\}$) represent 1-edge subgraphs and their frequencies in partition i. The network assignments to partitions are performed based on the similarity of each network (taking into consideration the 1-edge subgraphs and their frequencies in the network) and the centroid of partitions. Adopting this clustering-based partitioning approach helps to create *dissimilar fragments* and consequently reduce the complexity of the next aggregation steps. A subgraph is considered frequent if it is at least frequent in one of the partitions. First, all the locally (at the partition level) frequent subgraphs are mined. The union of these local frequent subgraphs is used to create a global candidate set. Then the partitions are scanned again to find the globally frequent subgraphs.

Another popular set of strategies for I/O bound problems is to store and retrieve data from databases. However, the required processing tasks for mining frequent subgraphs proposed in traditional algorithms cannot be used directly. In the following, some of the algorithms proposed to solve these challenges are discussed. These algorithms are mainly an improved version of previously proposed algorithms modified based on the constraints and available operations and mechanisms in relational databases. The pseudo-code describing the general approach in this category of algorithms is shown in Algorithm 1.

Algorithm 1. Frequent Subgraph Mining in Databases

Require: graph_database (DB), vertex_table, edge_table, min_supp
  one_edge join vertex and edge table
  one_edge $one\_edge \geq min\_supp$
  $FS =$ frequent subgraphs one_edge
  repeat
    $candidates$ $join(FS, one\_edge)$
    $candidates$ $unique(candidates)$ {Eliminate pseudo-duplicates}
    scan DB and compute frequencies of unique $candidates$
    $temp\_FS$ $candidates$ with frequency $\geq min\_supp$
    $FS = FS \cup temp\_FS$
  until $temp\_FS = \emptyset$

DB-FSG [154], [155] and DB-SUBDUE [160] are proposed for mining frequent subgraphs in relational databases. Also, OO-FSG [156] is an object-oriented database approach proposed based on the DB-FSG. The approaches adopted in these three algorithms are discussed in Appendix C, available in the online supplemental material.

Similar to I/O bound problems, several algorithms are proposed for CPU-bound problems employing different strategies for parallel and distributed frequent subgraph mining. In [151], thread-based parallelized implementations of gSpan and MoFa is proposed. In this algorithm, each thread has a copy of the gSpan or MoFa. Each tread mines frequent subgraphs. The network database is kept globally. Also, there is a global data structure keeping the frequent subgraphs in communication with local sets of frequent subgraphs of each thread. The main difference in the implementation of the two algorithms is the work distribution among threads. The MoFa adopts a *work donation* approach for load balancing with a global list of idle workers, and gSpan takes a *work stealing* approach with a global list of active workers.

In [153], a chip multiprocessor (CMP) architecture with gSpan as the subgraph mining algorithm is adopted. To perform the load balancing, distributed queuing is used in which one processor processes each queue. Processors can search for other processors' queues when their queues are empty. The load balancing can be performed through adaptive partitioning (the children of frequent subgraphs are generated and added to the queue which can later be processed by the same processor or other processors) and level-wise partitioning (in which the tasks are added to the queue of each processor to a pre-defined depth in the DFS tree and are processed by the

dequeuing processor). To improve memory utilization, the two adaptive network tiling (packaging the children of the same parent in one task, considering that instances of children of a parent are in the same network transactions as far as this strategy does not negatively impact the load balancing) and adaptive state management strategies are used. In adaptive state management, embedding lists are used to speed up the frequency computation at the expense of memory required for keeping the embedding lists. These lists are defined as mappings between candidates and their location in the database. If the same processor is used for mining children of candidates, adopting this strategy might improve efficiency.

Another algorithm in this category is Frequent Subgraph Mining using Hadoop (FSM-H) [152] developed on top of the MapReduce programming model [161]. FSM-H adopts a breadth-first approach for candidate generation and minimum DFS code (introduced in gSpan [25]) for checking graph isomorphism. The mining is performed in three main phases, data partitioning, the preparation phase, and mining phase (refer to Algorithm 2). The mining phase performs the mining steps iteratively to identify all the frequent subgraphs.

Algorithm 2. Frequent Subgraph Mining Using Hadoop (FSM-H) [152]

Require: graph_database (DB), min_supp

Data partition phase:

- input data is partitioned
- infrequent edges are eliminated

Preparation phase:

- Mappers:
  - partition-specific data structures are prepared including data structures for keeping possible extension from vertices and occurrence list of edges in the partition,
  - mining process is initialized by creating frequent edges as key-value pairs
- Reducers:
  - key-value pairs are written in the Hadoop Distributed File Systems (HDFS)

Mining phase: frequent subgraphs are iteratively mined

- Mappers:
  - mining starts by reading from HDFS
  - required data structures are created
  - candidates are generated
  - isomorphism evaluation are performed
  - occurrence lists are populated and with the DFS codes are emitted
- Reducers:
  - supports are computed for the subgraphs
  - frequent subgraphs are written to HDFS

MapReduce-FSG [157] and MRFSM [158] are two other MapReduce-based approaches (using Apache Hadoop) proposed for CPU-bound problems. These algorithms are described and discussed in Appendix C, available in the online supplemental material.

Similar to the other classes of the frequent subgraph mining problem, the design and development of algorithms able to cope with the CPU-bound or I/O-bound complexities have their own challenges. Considering the downward-closure property as the most critical pruning strategy in frequent subgraph mining, we need to identify infrequent candidates as early as possible. In most of the algorithms proposed, the frequency computation for a candidate requires evaluating the network transactions containing the candidate's parent(s). The networks containing the parents' subgraph might be located on different working nodes [152] and/or distributed over different network partitions and fragments [149], [159]. These networks are not uniformly distributed among various nodes or fragments; therefore, the frequency computation is not a trivial task. Although it is shown that the algorithms proposed for CPU-bound or I/O-bound are more practical, these challenges have made their generalizations to different subclasses of frequent subgraph mining relatively limited. For example, most of these algorithms are limited to mining exact subgraphs without any noise tolerance. Also, most of these algorithms are proposed for static settings.

6 Tools

Some of the proposed algorithms for frequent subgraph mining have been implemented and are publicly available. Table 6 lists some of these tools categorized based on the classification proposed in Fig. 1. In contrast to many other mining paradigms proposed for tabular data that are unified in one library or toolkit, there are just a few toolkits in frequent subgraph mining literature offering an integrated implementation of multiple subgraph mining algorithms with different functionalities. In [162], frequent subgraph mining is packaged with other frequent pattern mining approaches operating on itemsets, sequences, and trees, into one toolkit, called Data Mining Template Library (DMTL). They show that although these patterns are different, all of them can be modeled by networks. Their discovery is composed of a very similar set of tasks: candidate generation, isomorphism verification, and support computation. The algorithm used in DMTL for frequent subgraph mining is a modified version of gSpan.

One package, including the implementation of MoFa, gSpan, FFSM, and GASTON algorithms, is ParMol [163]. Some extensions are also provided in this package, such as mining directed networks and frequently closed subgraphs. The "subgraphMining" [164] is an R package including gSpan, SUBDUE, and SLEUTH (an algorithm proposed in [165] for mining frequent subtrees in a tree database).

7 Conclusion

In this paper, we reviewed some of the most popular algorithms proposed in the literature for mining frequent patterns in static and temporal networks. The frequent subgraph mining problem in a dataset of static networks has been extensively studied in the past decades. Considering the complexities associated with graph and subgraph isomorphism problems, it is shown in multiple studies that some of these algorithms can efficiently mine static networks, such as gSpan [25], [107], FFSM [109], and GASTON [113]. Other algorithms either are not as efficient as these algorithms or are proposed

TABLE 6
Publicly Available Tools for Algorithms Reviewed in This Paper

Name	Address	Programming language
Static network transaction setting		
Breadth-first search or Apriori-based algorithms		
FSG [59]	http://glaros.dtc.umn.edu/gkhome/pafi/overview	ANSI C & C++
Depth-first search algorithms		
gSpan [25], [107]	https://www.cs.ucsb.edu/ xyan/software/gSpan.htm	Java & C++
MoFa (or MoSS) (including gSpan and CloseGraph) [4]	http://www.borgelt.net/moss.html	Java
FFSM [109]	http://web.cs.ucla.edu/ weiwang/software.html	C++
Other approaches to frequent subgraph mining		
GASTON [113]	http://liacs.leidenuniv.nl/ nijssensgr/gaston/	C++
MULE [117]	http://compbio.case.edu/koyuturk/software/mule/	C
GraphSig [119]	http://www.cse.iitd.ac.in/∼sayan/software.html	Java
Temporal network transaction setting		
Inter-network subgraph mining		
PSEMiner [126]	https://compbio.cs.uic.edu/software/periodic/	C++
Intranetwork subgraph mining		
TimeCrunch [138]	http://nshah.net/publications.html	MATLAB & Python
CPU-bounded and I/O bounded		
FSM-H [152]	https://github.com/DMGroup-IUPUI/FSMH	Java

and customized for mining particular categories of subgraphs or specific domains. In general, there are multiple challenges associated with frequent subgraph mining problem. One such challenge is being able to utilize the information carried by frequent subgraphs. It is shown that lowering the support threshold results in an exponential increase in the number of subgraphs discovered in a network database. On the other hand, it is shown that the most frequent subgraphs are not necessarily the significant ones as well [119]. However, by increasing the support threshold, although we might have a smaller set of candidates, we might miss the significant subgraphs. As we discussed, there has been some algorithms proposed combining the discovery of significant and frequent subgraphs in the mining process. However, it seems that this area is an important avenue for future research in subgraph mining literature.

It is discussed that the exact algorithms consider two embeddings different even if the differences are due to a minor variation, such as a single extra edge, or a different label. In practical applications, these variations sometimes can be attributed to the noises related to the uncertainties inherent in the original system or errors attributed to the data collection and modeling phase. As reviewed in this paper, some of the algorithms try to allow some controlled structural variations among subgraphs. In some other cases, it might be helpful to use other *smoothing procedures* to create groups or *clusters* of frequent subgraphs considered as *structural representatives* in which all the frequent subgraphs within the range of accepted structural variations are included [166]. These clusters then can be used as features for secondary analysis of the data.

The frequent pattern mining in temporal networks has attracted more attention in recent years. The functionalities offered or subgraph categories covered by these algorithms are relatively limited compared to their static counterpart. As more and more data is being generated and collected every day in different disciplines and the complexities added to the problem due to the temporality of the data, it seems that this class of algorithms would be at the center of focus in the next coming years. Another avenue for future research in this class is mining frequent patterns and their evolution in network data streams. Developing more efficient data structures for keeping or indexing temporal data and probabilistic and deterministic approaches for mining frequent patterns would be other interesting research areas. Compared with other subclasses of frequent subgraph mining in temporal networks, mining network data streams are more challenging as the input data is unbounded, and the distribution of subgraphs is unknown in the data stream in advance.

Another relevant area not covered in this study is using inductive logic programming or graph grammar learning for mining frequent subgraphs. Inductive logic programming-based approaches are beneficial in mining structures carrying semantic concepts, or when background knowledge can guide the mining process [19] or recursive interactions are embedded in the structure. It is shown that these approaches can mine patterns cannot be easily discovered by frequent subgraph mining algorithms [167].

We reviewed some of the algorithms proposed for I/O and CPU-bound problems. Most of these algorithms are proposed for static settings. Besides, the proposed algorithms mine just the exact isomorphic frequent subgraphs and general forms of subgraphs. Therefore, developing algorithms for temporal and streaming network data and for mining special classes of subgraphs (such as maximal and closed), which allows for structural variations, is an avenue for further research.

In this survey, we also listed publicly available tools for different algorithms. However, in comparison with the number of algorithms reviewed, there are not many publicly available implementations of the algorithms and none for a few mining categories of Fig. 1, neither packages that integrate multiple algorithms from different settings.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation under Grant NSF-1741306, IIS-1650531, and DIBBs-1443019. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] L. V. Bertalanffy, *General System Theory: Foundations, Development, Applications*. New York, NY, USA: G. Braziller, 1972.
[2] M. E. J. Newman, "The structure and function of complex networks," *SIAM Rev.*, vol. 45, no. 2, pp. 167–256, 2003.
[3] K. Yoshida, H. Motoda, and N. Indurkhya, "Graph-based induction as a unified learning framework," *Appl. Intell.*, vol. 4, no. 3, pp. 297–316, 1994.
[4] C. Borgelt and M. R. Berthold, "Mining molecular fragments: Finding relevant substructures of molecules," in *Proc. IEEE Int. Conf. Data Mining*, 2002, pp. 51–58.
[5] L. Kovanen, M. Karsai, K. Kaski, J. Kertész, and J. Saramäki, "Temporal motifs in time-dependent networks," *J. Statist. Mechanics: Theory Experiment*, vol. 2011, no. 11, Nov. 2011, Art. no. P11005. [Online]. Available: https://doi.org/10.1088 2F1742–5468 2F2011 2F11 2Fp11005
[6] V. Nicosia, J. Tang, C. Mascolo, M. Musolesi, G. Russo, and V. Latora, "Graph metrics for temporal networks," in *Temporal Networks*. Berlin, Germany: Springer, 2013, pp. 15–40.
[7] J. Tang, M. Musolesi, C. Mascolo, and V. Latora, "Characterising temporal distance and reachability in mobile and online social networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 40, pp. 118–124, Jan. 2010. [Online]. Available: https://doi.org/10.1145/1672308.1672329
[8] D. Kempe, J. Kleinberg, and A. Kumar, "Connectivity and inference problems for temporal networks," *J. Comput. Syst. Sci.*, vol. 64, no. 4, pp. 820–842, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0022000002918295
[9] R. K. Pan and J. Saramäki, "Path lengths, correlations, and centrality in temporal networks," *Physical Rev. E*, vol. 84, Jul. 2011, Art. no. 016105. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.84.016105
[10] P. Holme and J. Saramäki, "Temporal networks," *Phys. Reports*, vol. 519, no. 3, pp. 97–125, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0370157312000841
[11] P. Holme, "Modern temporal network theory: A colloquium," *Eur. Physical J. B*, vol. 88, no. 9, Sep. 2015, Art. no. 234. [Online]. Available: https://doi.org/10.1140/epjb/e2015–60657-4
[12] J. Tang, M. Musolesi, C. Mascolo, V. Latora, and V. Nicosia, "Analysing information flows and key mediators through temporal centrality metrics," in *Proc. 3rd Workshop Soc. Netw. Syst.*, 2010, Art. no. 3. [Online]. Available: https://doi.org/10.1145/1852658.1852661
[13] V. Kostakos, "Temporal graphs," *Physica A, Statist. Mechanics Appl.*, vol. 388, no. 6, pp. 1007–1023, 2009.
[14] A. Li, S. P. Cornelius, Y.-Y. Liu, L. Wang, and A.-L. Barabási, "The fundamental advantages of temporal networks," *Science*, vol. 358, no. 6366, pp. 1042–1046, 2017. [Online]. Available: https://science.sciencemag.org/content/358/6366/1042
[15] P. Desikan and J. Srivastava, "Mining temporally evolving graphs," in *Proc. 6th WEBKDD Workshop Conjunction 10th ACM SIGKDD Conf.*, 2004, pp. 13–22.
[16] P. Desikan and J. Srivastava, "Mining temporally changing web usage graphs," in *Proc. 6th Int. Conf. Knowl. Discov. Web: Advances Web Mining Web Usage Anal.*, 2004, pp. 1–17. [Online]. Available: https://doi.org/10.1007/11899402_1
[17] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, "Network motifs: Simple building blocks of complex networks," *Science*, vol. 298, no. 5594, pp. 824–827, 2002. [Online]. Available: https://science.sciencemag.org/content/298/5594/824
[18] A. Jazayeri and C. C. Yang, "Motif discovery algorithms in static and temporal networks: A survey," *J. Complex Netw.*, vol. 8, no. 4, Dec. 2020, Art. no. cnaa031.
[19] D. J. Cook, L. B. Holder, and N. Ketkar, *Unsupervised and Supervised Pattern Learning in Graph Data*. Hoboken, NJ, USA: Wiley, 2006, ch. 7, pp. 159–181. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470073049.ch7
[20] P. C. Nguyen, K. Ohara, A. Mogi, H. Motoda, and T. Washio, *Constructing Decision Trees for Graph-Structured Data by Chunkingless Graph-Based Induction*. Berlin, Germany: Springer, 2006, pp. 390–399. [Online]. Available: https://doi.org/10.1007/11731139_45
[21] A. R. Benson, D. F. Gleich, and J. Leskovec, "Higher-order organization of complex networks," *Science*, vol. 353, no. 6295, pp. 163–166, 2016. [Online]. Available: https://science.sciencemag.org/content/353/6295/163
[22] H. Yin, A. R. Benson, J. Leskovec, and D. F. Gleich, "Local higher-order graph clustering," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 555–564. [Online]. Available: https://doi.org/10.1145/3097983.3098069
[23] C. E. Tsourakakis, J. Pachocki, and M. Mitzenmacher, "Scalable motif-aware graph clustering," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 1451–1460. [Online]. Available: https://doi.org/10.1145/3038912.3052653
[24] S. Gunnemann, I. Farber, B. Boden, and T. Seidl, "Subspace clustering meets dense subgraph mining: A synthesis of two paradigms," in *Proc. IEEE Int. Conf. Data Mining*, 2010, pp. 845–850.
[25] X. Yan and J. Han, "gSpan: Graph-based substructure pattern mining," in *Proc. IEEE Int. Conf. Data Mining*, 2002, pp. 721–724.
[26] T. Kudo, E. Maeda, and Y. Matsumoto, "An application of boosting to graph classification," in *Proc. 17th Int. Conf. Neural Inf. Process. Syst.*, 2005, pp. 729–736. [Online]. Available: http://papers.nips.cc/paper/2739-an-application-of-boosting-to-graph-classification.pdf
[27] X. Yan, P. S. Yu, and J. Han, "Graph Indexing: A frequent structure-based approach," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2004, pp. 335–346. [Online]. Available: https://doi.org/10.1145/1007568.1007607
[28] C. Chen, X. Yan, P. S. Yu, J. Han, D.-Q. Zhang, and X. Gu, "Towards graph containment search and indexing," in *Proc. 33rd Int. Conf. Very Large Data Bases*, 2007, pp. 926–937.
[29] A. Sankar, X. Zhang, and K. C.-C. Chang, "Motif-based convolutional neural network on graphs," *CoRR*, vol. abs/1711.05697, 2017.
[30] W. Fan *et al.*, "Direct mining of discriminative and essential frequent patterns via model-based search tree," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2008, pp. 230–238. [Online]. Available: https://doi.org/10.1145/1401890.1401922
[31] S. Pan, J. Wu, X. Zhu, G. Long, and C. Zhang, "Finding the best not the most: Regularized loss minimization subgraph selection for graph classification," *Pattern Recognit.*, vol. 48, no. 11, pp. 3783–3796, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320315001983
[32] M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis, "Frequent substructure-based approaches for classifying chemical compounds," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 8, pp. 1036–1050, Aug. 2005.
[33] J. Ugander, L. Backstrom, and J. Kleinberg, "Subgraph frequencies: Mapping the empirical and extremal geography of large graph collections," in *Proc. 22nd Int. Conf. World Wide Web*, 2013, pp. 1307–1318. [Online]. Available: https://doi.org/10.1145/2488388.2488502
[34] B. Güvenoglu and B. E. Bostanoglu, "A qualitative survey on frequent subgraph mining," *Open Comput. Sci.*, vol. 8, no. 1, pp. 194–209, 2018.
[35] F. N. Afrati, D. Fotakis, and J. D. Ullman, "Enumerating subgraph instances using Map-Reduce," in *Proc. IEEE 29th Int. Conf. Data Eng.*, 2013, pp. 62–73.
[36] D. Shasha, J. T. L. Wang, and R. Giugno, "Algorithmics and applications of tree and graph searching," in *Proc. 21st ACM SIGMOD-SIGACT-SIGART Symp. Princ. Database Syst.*, 2002, pp. 39–52. [Online]. Available: https://doi.org/10.1145/543613.543620

[37] X. Yan, P. S. Yu, and J. Han, "Substructure similarity search in graph databases," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2005, pp. 766–777. [Online]. Available: https://doi.org/10.1145/1066157.1066244

[38] L. Getoor and C. P. Diehl, "Link mining: A survey," *ACM SIGKDD Explorations Newslett.*, vol. 7, no. 2, pp. 3–12, Dec. 2005. [Online]. Available: https://doi.org/10.1145/1117454.1117456

[39] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *J. Amer. Soc. Inf. Sci. Technol.*, vol. 58, no. 7, pp. 1019–1031, May 2007.

[40] T. Tylenda, R. Angelova, and S. Bedathur, "Towards time-aware link prediction in evolving social networks," in *Proc. 3rd Workshop Soc. Netw. Mining Anal.*, 2009, Art. no. 9. [Online]. Available: https://doi.org/10.1145/1731011.1731020

[41] S. Fortunato, "Community detection in graphs," *Phys. Reports*, vol. 486, no. 3, pp. 75–174, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0370157309002841

[42] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 12, pp. 7821–7826, 2002. [Online]. Available: https://www.pnas.org/content/99/12/7821

[43] P. J. Mucha, T. Richardson, K. Macon, M. A. Porter, and J.-P. Onnela, "Community structure in time-dependent, multiscale, and multiplex networks," *Science*, vol. 328, no. 5980, pp. 876–878, 2010. [Online]. Available: https://science.sciencemag.org/content/328/5980/876

[44] P. Pons and M. Latapy, "Computing communities in large networks using random walks," in *Proc. Int. Symp. Comput. Inf. Sci.*, 2005, pp. 284–293.

[45] B. Boden, S. Günnemann, H. Hoffmann, and T. Seidl, "Mining coherent subgraphs in multi-layer graphs with edge labels," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2012, pp. 1258–1266. [Online]. Available: https://doi.org/10.1145/2339530.2339726

[46] B. Boden, S. Günnemann, H. Hoffmann, and T. Seidl, "MiMAG: Mining coherent subgraphs in multi-layer graphs with edge labels," *Knowl. Inf. Syst.*, vol. 50, no. 2, pp. 417–446, Feb. 2017. [Online]. Available: https://doi.org/10.1007/s10115-016-0949-5

[47] M. J. Zaki, "Efficiently mining frequent trees in a forest: Algorithms and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 8, pp. 1021–1035, Aug. 2005.

[48] C. Jiang, F. Coenen, and M. Zito, "A survey of frequent subgraph mining algorithms," *Knowl. Eng. Rev.*, vol. 28, no. 1, pp. 75–105, 2012.

[49] P. Bogdanov, M. Mongiov`, and A. K. Singh, "Mining heavy subgraphs in time-evolving networks," in *Proc. IEEE 11th Int. Conf. Data Mining*, 2011, pp. 81–90. [Online]. Available: https://doi.org/10.1109/ICDM.2011.101

[50] S. Ranshous, S. Shen, D. Koutra, S. Harenberg, C. Faloutsos, and N. F. Samatova, "Anomaly detection in dynamic networks: A survey," *WIREs Comput. Statist.*, vol. 7, no. 3, pp. 223–247, 2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/wics.1347

[51] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: A survey," *Data Mining Knowl. Discov.*, vol. 29, no. 3, pp. 626–688, 2015.

[52] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: A survey," *Data Mining Knowl. Discov.*, vol. 29, no. 3, pp. 626–688, May 2015. [Online]. Available: https://doi.org/10.1007/s10618-014-0365-y

[53] C. C. Bilgin and B. Yener, "Dynamic network evolution: Models, clustering, anomaly detection," *IEEE Netw.*, no. 1, 2006.

[54] P. Shoubridge, M. Kraetzl, W. Wallis, and H. Bunke, "Detection of abnormal change in a time series of graphs," *J. Interconnect. Netw.*, vol. 3, no. 01n02, pp. 85–101, 2002. [Online]. Available: https://doi.org/10.1142/S0219265902000562

[55] C. C. Aggarwal, Y. Zhao, and S. Y. Philip, "Outlier detection in graph streams," in *Proc. IEEE 27th Int. Conf. Data Eng.*, 2011, pp. 399–409.

[56] P. Bindu and P. S. Thilagam, "Mining social networks for anomalies: Methods and challenges," *J. Netw. Comput. Appl.*, vol. 68, pp. 213–229, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804516300029

[57] L. Dehaspe, H. Toivonen, and R. D. King, "Finding frequent substructures in chemical compounds," in *Proc. Int. Conf. Knowl. Discov. Data Mining*, 1998, pp. 30–36.

[58] A. Inokuchi, T. Washio, and H. Motoda, "An apriori-based algorithm for mining frequent substructures from graph data," in *Proc. Eur. Conf. Princ. Data Mining Knowl. Discov.*, 2000, pp. 13–23.

[59] M. Kuramochi and G. Karypis, "Frequent subgraph discovery," in *Proc. IEEE Int. Conf. Data Mining*, 2001, pp. 313–320.

[60] Y. Ren, K. Zhang, and Y. Shi, "Survival prediction from longitudinal health insurance data using graph pattern mining," in *Proc. IEEE Int. Conf. Bioinf. Biomed.*, 2019, pp. 1104–1108.

[61] C. Sun, Q. Li, L. Cui, H. Li, and Y. Shi, "Heterogeneous network-based chronic disease progression mining," *Big Data Mining Analytics*, vol. 2, no. 1, pp. 25–34, 2019.

[62] X. Cui *et al.*, "Classification of Alzheimer's disease, mild cognitive impairment, and normal controls with subnetwork selection and graph kernel principal component analysis based on minimum spanning tree brain functional network," *Front. Comput. Neurosci.*, vol. 12, 2018, Art. no. 31. [Online]. Available: https://www.frontiersin.org/article/10.3389/fncom.2018.00031

[63] J. Du, L. Wang, B. Jie, and D. Zhang, "Network-based classification of ADHD patients using discriminative subnetwork selection and graph kernel PCA," *Computerized Med. Imag. Graph.*, vol. 52, pp. 82–88, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0895611116300350

[64] D. Zhang, L. Tu, L.-J. Zhang, B. Jie, and G.-M. Lu, "Subnetwork mining on functional connectivity network for classification of minimal hepatic encephalopathy," *Brain Imag. Behav.*, vol. 12, no. 3, pp. 901–911, 2018.

[65] R. Kavuluru, M. Ramos-Morales, T. Holaday, A. G. Williams, L. Haye, and J. Cerel, "Classification of helpful comments on online suicide watch forums," in *Proc. 7th ACM Int. Conf. Bioinf. Comput. Biol. Health Informat.*, 2016, pp. 32–40. [Online]. Available: https://doi.org/10.1145/2975167.2975170

[66] Z. Deng *et al.*, "AirVis: Visual analytics of air pollution propagation," *IEEE Trans. Vis. Comput. Graphics*, vol. 26, no. 1, pp. 800–810, Jan. 2020.

[67] X. Li, Y. Cheng, G. Cong, and L. Chen, "Discovering pollution sources and propagation patterns in urban area," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 1863–1872.

[68] A. Mrzic *et al.*, "Grasping frequent subgraph mining for bioinformatics applications," *BioData Mining*, vol. 11, no. 1, pp. 20–24, 2018.

[69] Q. Chen, C. Lan, B. Chen, L. Wang, J. Li, and C. Zhang, "Exploring consensus RNA substructural patterns using subgraph mining," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 14, no. 5, pp. 1134–1146, Sep./Oct. 2017. [Online]. Available: https://doi.org/10.1109/TCBB.2016.2645202

[70] F. C. Queiroz, A. M. Vargas, M. G. Oliveira, G. V. Comarela, and S. A. Silveira, "ppiGReMLIN: A graph mining based detection of conserved structural arrangements in protein-protein interfaces," *BMC Bioinf.*, vol. 21, no. 1, pp. 1–25, 2020.

[71] J. Wu and L. Zhou, "DOBNet: Exploiting the discourse of deception behaviour to uncover online deception strategies," *Behav. Inf. Technol.*, vol. 34, no. 9, pp. 936–948, 2015.

[72] G. A. Wang, H. J. Wang, J. Li, A. S. Abrahams, and W. Fan, "An analytical framework for understanding knowledge-sharing processes in online Q&A communities," *ACM Trans. Manage. Inf. Syst.*, vol. 5, no. 4, Dec. 2014, Art. no. 18. [Online]. Available: https://doi.org/10.1145/2629445

[73] G. Bachi, M. Coscia, A. Monreale, and F. Giannotti, "Classifying trust/distrust relationships in online social networks," in *Proc. Int. Conf. Privacy Secur. Risk Trust and Int. Conf. Soc. Comput.*, 2012, pp. 552–557.

[74] N. Acosta-Mendoza, A. Gago-Alonso, J. A. Carrasco-Ochoa, J. F. Mart´nez-Trinidad, and J. E. Medina-Pagola, "Improving graph-based image classification by using emerging patterns as attributes," *Eng. Appl. Artif. Intell.*, vol. 50, pp. 215–225, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0952197616000348

[75] M. Dammak, M. Mejdoub, and C. B. Amar, "Histogram of dense subgraphs for image representation," *IET Image Process.*, vol. 9, no. 3, pp. 184–191, 2014.

[76] N. Acosta-Mendoza, A. Gago-Alonso, and J. E. Medina-Pagola, "Frequent approximate subgraphs as features for graph-based image classification," *Knowl.-Based Syst.*, vol. 27, pp. 381–392, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950705111002668

[77] N. B. Aoun, M. Mejdoub, and C. B. Amar, "Graph-based approach for human action recognition using spatio-temporal features," *J. Vis. Commun. Image Representation*, vol. 25, no. 2, pp. 329–338, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1047320313001910

[78] N. B. Aoun, H. Elghazel, and C. B. Amar, "Graph modeling based video event detection," in *Proc. Int. Conf. Innovat. Inf. Technol.*, 2011, pp. 114–117.

[79] N. B. Aoun, M. Mejdoub, and C. B. Amar, "Bag of sub-graphs for video event recognition," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2014, pp. 1547–1551.

[80] L. Brun and W. Kropatsch, "Contains and inside relationships within combinatorial pyramids," *Pattern Recognit.*, vol. 39, no. 4, pp. 515–526, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320305003924

[81] A. Abusnaina, H. Alasmary, M. Abuhamad, S. Salem, D. Nyang, and A. Mohaisen, "Subgraph-based adversarial examples against graph-based IoT malware detection systems," in *Proc. Int. Conf. Comput. Data Soc. Netw.*, 2019, pp. 268–281.

[82] N. Asrafi, "Comparing performances of graph mining algorithms to detect malware," in *Proc. ACM Southeast Conf.*, 2019, pp. 268–269. [Online]. Available: https://doi.org/10.1145/3299815.3314485

[83] V. Herrera-Semenets and A. Gago-Alonso, "A novel rule generator for intrusion detection based on frequent subgraph mining," *Ingeniare. Revista chilena de ingenier´a*, vol. 25, no. 2, pp. 226–234, 2017.

[84] V. Herrera-Semenets, N. Acosta-Mendoza, and A. Gago-Alonso, "A framework for intrusion detection based on frequent subgraph mining," in *Proc. 2nd SDM Workshop Mining Netw. Graphs: A Big Data Analytic Challenge*, 2015, pp. 1–8.

[85] T. Wüchner, A. Cis ak, M. Ochoa, and A. Pretschner, "Leveraging compression-based graph mining for behavior-based malware detection," *IEEE Trans. Dependable Secure Comput.*, vol. 16, no. 1, pp. 99–112, Jan./Feb. 2019.

[86] M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis, "Frequent substructure-based approaches for classifying chemical compounds," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 8, pp. 1036–1050, Aug. 2005.

[87] F. Lambusch, D. Waltemath, O. Wolkenhauer, K. Sandkuhl, C. Rosenke, and R. Henkel, "Identifying frequent patterns in biochemical reaction networks: A workflow," *Database*, vol. 2018, Jul. 2018, Art. no. bay051. [Online]. Available: https://doi.org/10.1093/database/bay051

[88] N. Acosta-Mendoza, J. A. Carrasco-Ochoa, J. F. Mart´nez-Trinidad, A. Gago-Alonso, and J. E. Medina-Pagola, "Image clustering based on frequent approximate subgraph mining," in *Proc. Mexican Conf. Pattern Recognit.*, 2018, pp. 189–198.

[89] K. Vacul´k and L. Popel´nský, "DGRMiner: Anomaly detection and explanation in dynamic graphs," in *Proc. Int. Symp. Intell. Data Anal.*, 2016, pp. 308–319.

[90] T. Washio and H. Motoda, "State of the art of graph-based data mining," *ACM SIGKDD Explorations Newslett.*, vol. 5, no. 1, pp. 59–68, Jul. 2003. [Online]. Available: https://doi.org/10.1145/959242.959249

[91] M. Wörlein, T. Meinl, I. Fischer, and M. Philippsen, "A quantitative comparison of the subgraph miners MoFa, gSpan, FFSM, and gaston," in *Proc. Eur. Conf. Princ. Data Mining Knowl. Discov.*, 2005, pp. 392–403.

[92] X. Yan and J. Han, *Discovery of Frequent Substructures*. Hoboken, NJ, USA: Wiley, 2006, ch. 5, pp. 97–115. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470073049.ch5

[93] J. Han, H. Cheng, D. Xin, and X. Yan, "Frequent pattern mining: Current status and future directions," *Data Mining Knowl. Discov.*, vol. 15, no. 1, pp. 55–86, Aug. 2007. [Online]. Available: https://doi.org/10.1007/s10618–006-0059-1

[94] H. Cheng, X. Yan, and J. Han, "Mining graph patterns," in *Managing and Mining Graph Data*, C. C. Aggarwal and H. Wang, Eds. Boston, MA, USA: Springer, 2010, pp. 365–392. [Online]. Available: https://doi.org/10.1007/978–1-4419-6045-0_12

[95] S. Parthasarathy, S. Tatikonda, and D. Ucar, "A survey of graph mining techniques for biological datasets," in *Managing and Mining Graph Data*. Berlin, Germany: Springer, 2010, pp. 547–580. [Online]. Available: https://doi.org/10.1007/978–1-4419-6045-0_18

[96] V. Krishna, N. N. R. R. Suri, and G. Athithan, "A comparative survey of algorithms for frequent subgraph discovery," *Current Sci.*, vol. 100, no. 2, pp. 190–198, 2011.

[97] A. Masoudi-Nejad, F. Schreiber, and Z. R. M. Kashani, "Building blocks of biological networks: A review on major network motif discovery algorithms," *IET Syst. Biol.*, vol. 6, no. 5, pp. 164–174, Oct. 2012.

[98] S. U. Rehman, S. Asghar, Y. Zhuang, and S. Fong, "Performance evaluation of frequent subgraph discovery techniques," *Math. Problems Eng.*, vol. 2014, pp. 1–6, 2014.

[99] R. Raj and R. Prabhakar, "Frequent subgraph mining algorithms – A survey," *Procedia Comput. Sci.*, vol. 47, pp. 197–204, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050915004664

[100] D. J. Cook and L. B. Holder, *Mining Graph Data*. Hoboken, NJ, USA: Wiley, 2006.

[101] M. Kuramochi and G. Karypis, *Finding Topological Frequent Patterns From Graph Datasets*. Hoboken, NJ, USA: Wiley, 2006, ch. 6, pp. 117–158. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470073049.ch6

[102] A. Inokuchi, T. Washio, K. Nishimura, and H. Motoda, "A fast algorithm for mining frequent connected subgraphs," IBM Research Report, IBM Research, Tokyo Research Laboratory, RT0448, 2002.

[103] A. Inokuchi, T. Washio, and H. Motoda, "Complete mining of frequent patterns from graphs: Mining graph data," *Mach. Learn.*, vol. 50, no. 3, pp. 321–354, Mar. 2003. [Online]. Available: https://doi.org/10.1023/A:1021726221443

[104] M. Kuramochi and G. Karypis, "Discovering frequent geometric subgraphs," *Inf. Syst.*, vol. 32, no. 8, pp. 1101–1120, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0306437907000087

[105] M. Kuramochi, M. Desphande, and G. Karypis, "Mining scientific datasets using graphs," in *Data Mining: Next Generation Challenges and Future Directions*, H. Kargupta, A. Joshi, K. Sivakumar, and Y. Yesha, Eds. Palo Alto, CA, USA: AAAI Press, 2004, pp. 315–334.

[106] N. Vanetik, E. Gudes, and S. E. Shimony, "Computing frequent graph patterns from semistructured data," in *Proc. IEEE Int. Conf. Data Mining*, 2002, pp. 458–465.

[107] X. Yan and J. Han, "gSpan: Graph-based substructure pattern mining," University of Illinois at Urbana-Champaign, Champaign, IL, Tech. Rep. UIUCDCS-R-2002–2296, 2002.

[108] X. Yan and J. Han, "CloseGraph: Mining closed frequent graph patterns," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2003, pp. 286–295. [Online]. Available: https://doi.org/10.1145/956750.956784

[109] J. Huan, W. Wang, and J. Prins, "Efficient mining of frequent subgraphs in the presence of isomorphism," in *Proc. 3rd IEEE Int. Conf. Data Mining*, 2003, pp. 549–552.

[110] J. Huan, W. Wang, J. Prins, and J. Yang, "SPIN: Mining maximal frequent subgraphs from graph databases," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2004, pp. 581–586. [Online]. Available: https://doi.org/10.1145/1014052.1014123

[111] J. Huan, W. Wang, J. Prins, and J. Yang, "SPIN: Mining maximal frequent subgraphs from graph databases," Univ. North Carolina at Chapel Hill, Chapel Hill, NC, Tech. Rep. TR04–018, 2004.

[112] L. T. Thomas, S. R. Valluri, and K. Karlapalem, "MARGIN: Maximal frequent subgraph mining," *ACM Trans. Knowl. Discov. Data*, vol. 4, no. 3, Oct. 2010, Art. no. 10. [Online]. Available: https://doi.org/10.1145/1839490.1839491

[113] S. Nijssen and J. N. Kok, "A quickstart in frequent structure mining can make a difference," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2004, pp. 647–652. [Online]. Available: https://doi.org/10.1145/1014052.1014134

[114] R. Jin, C. Wang, D. Polshakov, S. Parthasarathy, and G. Agrawal, "Discovering frequent topological structures from graph datasets," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2005, pp. 606–611. [Online]. Available: https://doi.org/10.1145/1081870.1081944

[115] X. Yan, X. J. Zhou, and J. Han, "Mining closed relational graphs with connectivity constraints," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2005, pp. 324–333. [Online]. Available: https://doi.org/10.1145/1081870.1081908

[116] X. Yan, H. Cheng, J. Han, and P. S. Yu, "Mining significant graph patterns by leap search," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2008, pp. 433–444. [Online]. Available: https://doi.org/10.1145/1376616.1376662

[117] M. Koyuturk, Y. Kim, S. Subramaniam, W. Szpankowski, and A. Grama, "Detecting conserved interaction patterns in biological networks," *J. Comput. Biol.*, vol. 13, no. 7, pp. 1299–1322, 2006.

[118] F. Zhu, X. Yan, J. Han, and P. S. Yu, "gPrune: A constraint pushing framework for graph pattern mining," in *Proc. Pacific-Asia Conf. Knowl. Discov. Data Mining*, 2007, pp. 388–400.
[119] S. Ranu and A. K. Singh, "GraphSig: A scalable approach to mining significant subgraphs in large graph databases," in *Proc. IEEE 25th Int. Conf. Data Eng.*, 2009, pp. 844–855.
[120] M. Kuramochi and G. Karypis, "An efficient algorithm for discovering frequent subgraphs," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 9, pp. 1038–1051, Sep. 2004.
[121] M. Kuramochi and G. Karypis, "An efficient algorithm for discovering frequent subgraphs," Univ. Minnesota, Minneapolis, MN, Tech. Rep. TR 02–026, 2004.
[122] T. Horváth, J. Ramon, and S. Wrobel, "Frequent subgraph mining in outerplanar graphs," *Data Mining Knowl. Discov.*, vol. 21, no. 3, pp. 472–508, Nov. 2010. [Online]. Available: https://doi.org/10.1007/s10618–009-0162-1
[123] A. Jazayeri and C. C. Yang, "Motif discovery algorithms in static and temporal networks: A survey," *J. Complex Netw.*, vol. 8, no. 4, cnaa031, 2020. [Online]. Available: https://doi.org/10.1093/comnet/cnaa031
[124] P. J. Dickinson, H. Bunke, A. Dadej, and M. Kraetzl, "On graphs with unique node labels," in *Proc. Int. Workshop Graph-Based Representations Pattern Recognit.*, 2003, pp. 13–23.
[125] M. Lahiri and T. Y. Berger-Wolf, "Structure prediction in temporal networks using frequent subgraphs," in *Proc. IEEE Symp. Comput. Intell. Data Mining*, 2007, pp. 35–42.
[126] M. Lahiri and T. Y. Berger-Wolf, "Mining periodic behavior in dynamic social networks," in *Proc. 8th IEEE Int. Conf. Data Mining*, 2008, pp. 373–382.
[127] C. H. You, L. B. Holder, and D. J. Cook, "Graph-based data mining in dynamic networks: Empirical comparison of compression-based and frequency-based subgraph mining," in *Proc. IEEE Int. Conf. Data Mining Workshops*, 2008, pp. 929–938.
[128] M. Berlingerio, F. Bonchi, B. Bringmann, and A. Gionis, *Mining Graph Evolution Rules*. Berlin, Germany: Springer, 2009, pp. 115–130.
[129] R. Ahmed and G. Karypis, "Algorithms for mining the evolution of conserved relational states in dynamic networks," *Knowl. Inf. Syst.*, vol. 33, no. 3, pp. 603–630, Dec. 2012. [Online]. Available: https://doi.org/10.1007/s10115–012-0537-2
[130] T. Uno and Y. Uno, "Mining preserving structures in a graph sequence," *Theor. Comput. Sci.*, vol. 654, pp. 155–163, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304397515011469
[131] A. Inokuchi and T. Washio, "A fast method to mine frequent subsequences from graph sequence data," in *Proc. 8th IEEE Int. Conf. Data Mining*, 2008, pp. 303–312.
[132] A. Inokuchi and T. Washio, "GTRACE: Mining frequent subsequences from graph sequences," *IEICE Trans. Inf. Syst.*, vol. E93-D, no. 10, pp. 2792–2804, 2010.
[133] A. Inokuchi, H. Ikuta, and T. Washio, "Efficient graph sequence mining using reverse search," *IEICE Trans. Inf. Syst.*, vol. E95.D, no. 7, pp. 1947–1958, 2012.
[134] A. Inokuchi and T. Washio, "Mining frequent graph sequence patterns induced by vertices," in *Proc. SIAM Int. Conf. Data Mining*, 2010, pp. 466–477. [Online]. Available: https://epubs.siam.org/doi/abs/10.1137/1.9781611972801.41
[135] A. Inokuchi and T. Washio, "FRISSMiner: Mining frequent graph sequence patterns induced by vertices," *IEICE Trans. Inf. Syst.*, vol. E95.D, no. 6, pp. 1590–1602, 2012.
[136] C. Robardet, "Constraint-based pattern mining in dynamic graphs," in *Proc. 9th IEEE Int. Conf. Data Mining*, 2009, pp. 950–955.
[137] B. Wackersreuther, P. Wackersreuther, A. Oswald, C. Böhm, and K. M. Borgwardt, "Frequent subgraph discovery in dynamic networks," in *Proc. 8th Workshop Mining Learn. Graphs*, 2010, pp. 155–162. [Online]. Available: https://doi.org/10.1145/1830252.1830272
[138] N. Shah, D. Koutra, T. Zou, B. Gallagher, and C. Faloutsos, "TimeCrunch: Interpretable dynamic graph summarization," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 1055–1064. [Online]. Available: https://doi.org/10.1145/2783258.2783321
[139] A. Cuzzocrea, Z. Han, F. Jiang, C. K. Leung, and H. Zhang, "Edge-based mining of frequent subgraphs from graph streams," *Procedia Comput. Sci.*, vol. 60, pp. 573–582, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S187705091502311X
[140] A. Bifet, G. Holmes, B. Pfahringer, and R. Gavaldà, "Mining frequent closed graphs on evolving data streams," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2011, pp. 591–599. [Online]. Available: https://doi.org/10.1145/2020408.2020501
[141] P. Braun, J. J. Cameron, A. Cuzzocrea, F. Jiang, and C. K. Leung, "Effectively and efficiently mining frequent patterns from dense graph streams on disk," *Procedia Comput. Sci.*, vol. 35, pp. 338–347, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050914010795
[142] C. C. Aggarwal, Y. Li, P. S. Yu, and R. Jin, "On dense pattern mining in graph streams," *Proc. VLDB Endowment*, vol. 3, no. 1/2, pp. 975–984, Sep. 2010. [Online]. Available: https://doi.org/10.14778/1920841.1920964
[143] D. Burdick, M. Calimlim, and J. Gehrke, "MAFIA: A maximal frequent itemset algorithm for transactional databases," in *Proc. 17th Int. Conf. Data Eng.*, 2001, pp. 443–452.
[144] C. H. You, L. B. Holder, and D. J. Cook, "Learning patterns in the dynamics of biological networks," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2009, pp. 977–986. [Online]. Available: https://doi.org/10.1145/1557019.1557125
[145] M. Lahiri and T. Y. Berger-Wolf, "Periodic subgraph mining in dynamic networks," *Knowl. Inf. Syst.*, vol. 24, no. 3, pp. 467–497, 2010.
[146] S. Wernicke and F. Rasche, "FANMOD: A tool for fast network motif detection," *Bioinformatics*, vol. 22, no. 9, pp. 1152–1153, Feb. 2006. [Online]. Available: https://doi.org/10.1093/bioinformatics/btl038
[147] C. Wang, W. Wang, J. Pei, Y. Zhu, and B. Shi, "Scalable mining of large disk-based graph databases," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2004, pp. 316–325. [Online]. Available: https://doi.org/10.1145/1014052.1014088
[148] G. M. Slota and K. Madduri, "Fast approximate subgraph counting and enumeration," in *Proc. 42nd Int. Conf. Parallel Process.*, 2013, pp. 210–219.
[149] S. N. Nguyen, M. E. Orlowska, and X. Li, "Graph mining based on a data partitioning approach," in *Proc. 19th Conf. Australasian Database*, 2008, pp. 31–37.
[150] W. Garcia, C. Ordonez, K. Zhao, and P. Chen, "Efficient algorithms based on relational queries to mine frequent graphs," in *Proc. 3rd Workshop PhD Students Inf. Knowl. Manage.*, 2010, pp. 17–24. [Online]. Available: https://doi.org/10.1145/1871902.1871906
[151] T. Meinl, M. Worlein, I. Fischer, and M. Philippsen, "Mining molecular datasets on symmetric multiprocessor systems," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, 2006, vol. 2, pp. 1269–1274.
[152] M. A. Bhuiyan and M. Al Hasan, "An iterative MapReduce based frequent subgraph mining algorithm," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 3, pp. 608–620, Mar. 2015.
[153] G. Buehrer, S. Parthasarathy, and Y.-K. Chen, "Adaptive parallel graph mining for CMP architectures," in *Proc. 6th Int. Conf. Data Mining*, 2006, pp. 97–106.
[154] S. Chakravarthy and S. Pradhan, "DB-FSG: An SQL-based approach for frequent subgraph mining," in *Proc. Int. Conf. Database Expert Syst. Appl.*, 2008, pp. 684–692.
[155] S. K. Pradhan, "A relational database approach to frequent subgraph (FSG) mining," Master's thesis, Dept. Comput. Sci. Eng., The University of Texas at Arlington, Arlington, TX, 2006.
[156] B. Srichandan and R. Sunderraman, "OO-FSG: An object-oriented approach to mine frequent subgraphs," in *Proc. 9th Australasian Data Mining Conf.*, 2011, pp. 221–228.
[157] S. Hill, B. Srichandan, and R. Sunderraman, "An iterative MapReduce approach to frequent subgraph mining in biological datasets," in *Proc. ACM Conf. Bioinf. Comput. Biol. Biomed.*, 2012, pp. 661–666. [Online]. Available: https://doi.org/10.1145/2382936.2383055
[158] W. Lin, X. Xiao, and G. Ghinita, "Large-scale frequent subgraph mining in MapReduce," in *Proc. IEEE 30th Int. Conf. Data Eng.*, 2014, pp. 844–855.
[159] J. Wang, W. Hsu, M. L. Lee, and C. Sheng, "A partition-based approach to graph mining," in *Proc. 22nd Int. Conf. Data Eng.*, 2006, pp. 74–74.
[160] S. Chakravarthy, R. Beera, and R. Balachandran, "DB-Subdue: Database approach to graph mining," in *Proc. Pacific-Asia Conf. Knowl. Discov. Data Mining*, 2004, pp. 341–350.
[161] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008. [Online]. Available: https://doi.org/10.1145/1327452.1327492

[162] V. Chaoji, M. Al Hasan, S. Salem, and M. J. Zaki, "An integrated, generic approach to pattern mining: Data mining template library," *Data Mining Knowl. Discov.*, vol. 17, no. 3, pp. 457–495, 2008.
[163] T. Meinl, M. Wörlein, O. Urzova, I. Fischer, and M. Philippsen, "The ParMol package for frequent subgraph mining," *Electron. Commun. EASST*, vol. 1, pp. 1–12, 2007.
[164] N. F. Samatova, W. Hendrix, J. Jenkins, K. Padmanabhan, and A. Chakraborty, *Practical Graph Mining With R*. Boca Raton, FL, USA: CRC Press, 2013.
[165] M. J. Zaki, "Efficiently mining frequent embedded unordered trees," *Fundamenta Informat.*, vol. 66, no. 1/2, pp. 33–52, Nov. 2004.
[166] C. Chen, C. X. Lin, X. Yan, and J. Han, "On effective presentation of graph patterns: A structural representative approach," in *Proc. 17th ACM Conf. Inf. Knowl. Manage.*, 2008, pp. 299–308. [Online]. Available: https://doi.org/10.1145/1458082.1458124
[167] I. Jonyer, *Graph Grammar Learning*. Hoboken, NJ, USA: Wiley, 2006, ch. 8, pp. 183–201. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470073049.ch8

Ali Jazayeri is currently working toward the PhD degree with The College of Computing and Informatics CCI), Drexel University, Philadelphia, Pennsylvania, and his dissertation focuses on mining frequent substructures and their evolution in temporal networks. He is an instructor in the MS in Data Science program at CCI. He is working in the Healthcare Informatics Research Laboratory under the supervision of Professor Christopher C. Yang.

Christopher C. Yang is currently a professor with the College of Computing and Informatics, Drexel University, Philadelphia, Pennsylvania. He also has a courtesy appointment with the School of Biomedical Engineering, Science, and Health Systems. He is the director of Data Science Programs and the program director of MS in Health Informatics. His research interest includes data science, artificial intelligence, machine learning, healthcare informatics, social media analytics, electronic commerce, and intelligence and security informatics. He has more than 300 publications in top-tier journals, conferences, and books, such as the *ACM Transactions on Intelligent Systems and Technology*, *ACM Transaction on Management Information Systems*, *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Computational Social Systems*, the *PLOS One*, the *Journal of Medical Internet Research*, *Artificial Intelligence in Medicine*, and more. He has received more than 5M research fundings from NSF, NIH, PCORI, HK RGC, etc. He is the editor-in-chief of the *Journal of Healthcare Informatics Research* and the *Electronic Commerce Research and Application*. He is the editor of the CRC book series on Healthcare Informatics and the founding steering committee chair of the IEEE International Conference on Healthcare Informatics. He has been the general chair of more than five conferences and program chairs of more than 10 conferences. He is the director of the Healthcare Informatics Research Lab. His recent research includes pharmacovigilance, drug repositioning, predictive modeling of sepsis, predictive modeling of disengagement driving for injury prevention, heterogeneous network mining, distributed graph computing, health intervention through social media for substance use disorders, and social network analytics.