Intelligent Data Analysis 26 (2022) 1071–1096
DOI 10.3233/IDA-205698
IOS Press

ITeM: Independent temporal motifs to summarize and compare temporal networks

Sumit Purohit[a,*], George Chin[a] and Lawrence B. Holder[b]
[a]*Pacific Northwest National Laboratory, Richland, WA, USA*
[b]*Washington State University, Pullman, WA, USA*

Abstract. Networks are a fundamental and flexible way of representing various complex systems. Many domains such as communication, citation, procurement, biology, social media, and transportation can be modeled as a set of entities and their relationships. Temporal networks are a specialization of general networks where every relationship occurs at a discrete time. The temporal evolution of such networks is as important to understand as the structure of the entities and relationships. We present the *Independent Temporal Motif* (ITeM) to characterize temporal graphs from different domains. ITeMs can be used to model the structure and the evolution of the graph. In contrast to existing work, ITeMs are edge-disjoint directed motifs that measure the temporal evolution of ordered edges within the motif. For a given temporal graph, we produce a feature vector of ITeM frequencies and the time it takes to form the ITeM instances. We apply this distribution to measure the similarity of temporal graphs. We show that ITeM has higher accuracy than other motif frequency-based approaches. We define various ITeM-based metrics that reveal salient properties of a temporal network. We also present *importance sampling* as a method to efficiently estimate the ITeM counts. We present a distributed implementation of the ITeM discovery algorithm using Apache Spark and GraphFrame. We evaluate our approach on both synthetic and real temporal networks.

Keywords: Temporal graph, temporal motif, independent motif, graph comparison, embeddings

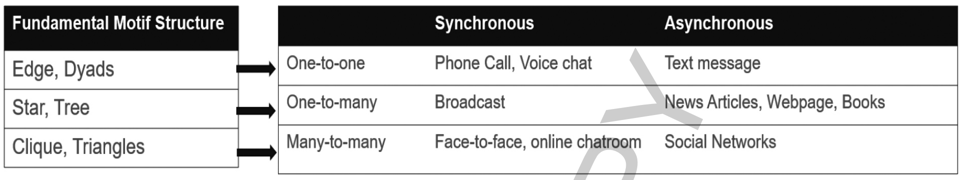
1. Introduction

Networks have been widely used to represent entities, relationships, and behaviors in many real-world domains including power grids [10], social networks [22], microbial interaction networks [49], corporate networks [51], the food web [19], and modeling adversarial activities [11]. These complex systems do not show a temporal or structural continuum, but rather show a characteristic non-linear dynamic behavior [4,52]. Count-based metrics such as the number of entities (i.e., nodes), the number of interactions (i.e., edges), and the average connectivity of the entities in the network (i.e., degree) are important measures that represent the population and the interaction density of the entities involved in the network. However, these measures are limited in their ability to describe non-linear, localized, and dynamic properties of the systems. To uncover structural, temporal, and functional insights of complex systems, *network motifs* have been used extensively in recent years as they provide a tractable approximation of the networks that can be measured and updated within given memory and compute constraints. Holme et al. [15] define a library of motifs that can represent six fundamental interactions types in *synchronous* and *asynchronous* communication [45], as shown in Fig. 1. In the case of synchronous communication, both

[*]Corresponding author: Sumit Purohit, Pacific Northwest National Laboratory, Richland, WA, USA. E-mail: Sumit.Purohit@pnnl.gov.

Table 1
Temporal ordering of events

Event	*edge_id*	*time:t*	*edge:<source, edge type, destination>*
E1	e1	0	*<bob, rent, car>*
E1	e2	10	*<bob, driveTo, shopping mall>*
E1	e3	30	*<bob, starts, shooting>*
E2	e4	10	*<john, rent, car>*
E2	e5	30	*<john, driveTo, shopping mall>*
E2	e6	0	*<john, starts, shooting>*



Fundamental Motif Structure
Edge, Dyads
Star, Tree
Clique, Triangles

	Synchronous	Asynchronous
One-to-one	Phone Call, Voice chat	Text message
One-to-many	Broadcast	News Articles, Webpage, Books
Many-to-many	Face-to-face, online chatroom	Social Networks

Fig. 1. Fundamental interaction types define by motifs.

participants are active and engaged, whereas, in the case of asynchronous communication, the receiver may not be present when the sender initiates the message. In addition to the structure of a graph, [8,25], the motifs can also measure temporal patterns in a graph. Recent temporal motif-based approaches [17,40] count all the isomorphic instances of a motif that are formed within a given duration, and use the count to characterize the graph. In addition to the global temporal patterns of the graph, it is also desirable to discover the local temporal evolution of the graph which is important for many domains such as social networks, communication networks, and terrorism activity knowledge graphs. Table 1 shows a notional example of two events. Both the events that take 30 time units to form and can be represented as *star* temporal motifs but {e1, e2, e3} and {e6, e4, e5} differ in the order they are formed. Such an ordering of events is a critical piece of information to measure the spread of misinformation in social networks, packet switching in communication networks, etc.

We present the **I**ndependent **Te**mporal **M**otif (ITeM) as the elementary building block of temporal networks. The core contribution of ITeM is an unsupervised, temporal motif based graph characterization approach that encodes temporal ordering of the edges, in addition to the structure of the motif. Another research contribution of ITeM is the use of edge-disjoint (or *Independent*) motifs to characterize the network. In contrast to the count of isomorphic instances used in [17,40], ITeM better characterizes a network because it contains more information about the structure and temporal evolution of the network which is useful to measure the resilience of a network topology such as communication or power-grid network.

The rest of the paper is organized as follows. Section 2 presents related work and highlights out research contributions of this work. Section 3 lays out various definitions. We start by defining *Atomic Motif* and extend it to *Temporal Motif* and *Independent Temporal Motif*. We also define various ITeM-based metrics used to characterize salient properties of a temporal graph. We present our core approach and distributed algorithms in Section 4. Section 5 shows our experimentation with synthetic and real-world datasets to summarize the temporal networks and measure their similarity. Section 6 presents conclusions and future work.

2. Related work

Extensive research has been done on the appropriate definition of *network motifs* [37,53] and their application to various network analytical tasks. Cao et al. [8] use network motifs to define the *network backbone*, which is a collection of relevant nodes and edges in the large-scale network. They define a motif-based extraction method to extract the *functional backbone* of the complex network. The functional backbone is indicative of certain functional properties of the network that cannot be explained by centrality-based backbones. Similarly, Shen et al. [49] use a *weighted motif* to cluster microbial interaction networks. Network motifs can also be used to identify the exchange of emotions in online communication networks, such as Twitter [25], using *emotion-exchange motifs*. The emotion-exchange motifs containing reciprocal edges manifest anger or fear, either in isolation or in any combination with other emotions. Conversely, positive emotions are characteristic of one-way motifs. Jin et al. [18] define *TrendMotif* that describes a recurring subgraph of weighted vertices and edges in a dynamic network over a user-defined period. The *TrendMotif* can indicate the increasing and/or decreasing intervals for the weighted vertices or edges over the time period. Borgwardt et al. [7] extend pattern mining on static graphs to time series of graphs where each graph has the same set of vertices and observed addition and deletion of edges.

A *temporal network* is a generalization of a static network that changes with time. Many system modeling approaches model time as an attribute of the entity or the interaction, which makes temporal graphs a special case of *attributed graphs*. We interchangeably use *network* and *graph* in this paper. Incorporating time into static graphs has given rise to a new set of important and challenging problems that cannot be modeled as a static graph problem [21,36]. Network motifs are also used to visualize and summarize large dynamic graphs [32]. TimeCrunch [46,47] discovers five different temporal patterns of some common substructures and summarizes the network in terms of a sequence of substructures that minimizes the Minimum Description Length (MDL) cost of describing the graph. Adhikari et al. [2] use local substructures to condense a temporal network. Liu et al. [29] propose a Bayesian framework to estimate the number of temporal motifs in communication networks. A majority of the prior research does not account for the temporal evolution of the motif. Recent work [40] defines δ-temporal motif as an elementary unit of the temporal network and provides a general methodology for counting such motifs. It computes the frequency of *overlapping* temporal motifs, where one interaction can be part of more than one temporal motif. In a δ-temporal motif, all the edges in a given motif have to occur inside the time period of δ time units. Li et al. [28] propose temporal Heterogeneous Information Networks (HIN) and develop a set of algorithms to count HINs. Aparício et al. [3] use *orbit* transitions to compare a set of temporal networks. Dynamic Graphlet (DG) [17] extends static graphlets to analyze structure and function of molecular network. DG distinguishes *graphlet* from motif as induced subgraphs that are not defined based on the statistical significance of the substructure. DG defines orbit in a graphlet to measure automorphism in the graphlet. Sarkar et al. [43] use the temporal motif to understand information flow in social networks. Liu et al. [31] define *event-pairs* and use a library of temporal motifs to focus on behavior of intermediate events and temporal correlations.

We propose the **I**ndependent **Te**mporal **M**otif (ITeM) as the elementary building block of temporal networks. In contrast to the related work, ITeMs are edge-disjoint temporal motifs that provide insight into the temporal evolution of a graph, such as its rate of growth, neighborhood, ordering of temporal edges, and the change in the role of a vertex over time. The independence of the temporal motif leads to mutually exclusive motif instances by restricting each edge to participate in only one temporal motif instance. We use a set of temporal motifs that are simple to compute but at the same time representative of temporal, structural, and functional properties of the network. We also define properties to measure the temporal evolution of the motifs, which informs the rate at which motifs are formed in the network. In contrast to previous work, no limit is put on the δ time window of the motif, but it can be restricted optionally. We provide algorithms to compute the independent temporal motif distribution of a given graph. Additionally, we also provide a new distributed implementation using the Apache Spark graph analytic framework.

Table 2
Symbols and their descriptions

Symbol	Description
T	Temporal graph
G_i	i^{th} window
t	Total number of windows
K	Set of Atomic Motifs
m_k	k^{th} Atomic motif
m_{kl}	l^{th} Temporal motif of k^{th} atomic Motif
$\mathcal{T}$	Set of time-steps associated with motif edges
M	Motif instance
$\hat{M}$	ITeM instance
v_k	Number of vertices in k^{th} motif
$\hat{V}_k$	# unique vertices in ITeM instances of k^{th} motif
I	Set of Importance values for each window
I_i	Importance of i^{th} window
F	Temporal motif distribution for a given T
d	*Order* of a motif
o	*Orbit* of a motif

3. Definitions

We present the ITeM-based approach to characterize a temporal network. In the following sections, we present definitions and algorithms used by ITeM to model a temporal network. We also review the Maximum Independent Set (MIS) problem, which is a subproblem of the proposed algorithm. MIS has been proved to be an NP-complete problem, and we present a heuristic-based approach to finding the lower bound on the ITeM frequency [33]. We also outline a sampling method to estimate the true frequency of a temporal motif in the network. The sampling approach is based on the *importance* of the sampled network [30].

A temporal graph is a specialization of a static graph, where each edge of the static graph appears at a time unit such as second, day, year, etc. Various representations of temporal graphs that are useful in different scenarios are proposed [35]. We use a window-based representation, where each window corresponds to a temporal sub-graph between two time-steps.

Definition 1. Temporal Graph: A temporal graph T is an ordered sequence of graphs $T = G_1, \dots, G_t$, indexed by a window id $i = 1, \dots, t$. We define $Gi = (V_i, E_i)$, where V_i and E_i denote the vertex and edge sets, respectively, in the window i, arriving since the window $i - 1$. We say the temporal graph T is on vertex set $V_T = V_1 \cup \cdots \cup V_t$ and edge set $E_T = E_1 \cup \cdots \cup E_t$.

This definition allows for the representation of a large graph with a single window. It is useful for datasets that are small in size and cover a small period of time.

3.1. Atomic motif

Atomic motifs are small subgraphs that serve as interesting indicators for complex networks. They can reveal patterns of association among entities in the network. Figure 2 shows a library of atomic motifs used in the current work. A k-order motif is defined by a subgraph with k vertices. Lower-order motifs such as isolated vertex (order d = 1), self-loop (d = 1), and isolated edge (d = 2) are examples of *fringe* motifs as they have less (sometimes zero) connectivity to the rest of the network. Whereas, higher-order motifs such as wedge (d = 3), triangle (d = 3), and square (d = 4) are an example of *core* motifs, which

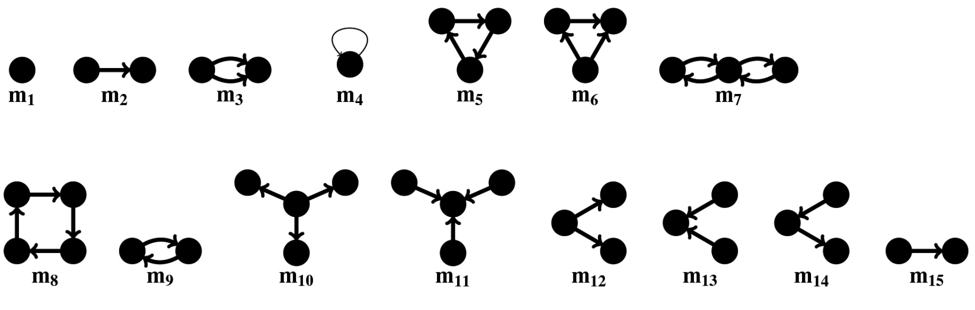


Fig. 2. Atomic motifs.

have been found to constitute a major fraction of real-world graphs. Our experimentation shows that the relative frequencies of *fringe* and *core* motifs in a temporal network can be used to compute graph similarity.

We can define atomic motifs of any number of vertices and edges, but the larger motifs are more difficult to search for in a network due to the intractability of the subgraph isomorphism, leading to an exponential increase in the runtime. Previous work shows that the computational cost of motif counting increases exponentially with k in $\mathcal{O}(|V|)^k$ [50]. Conversely, smaller atomic motifs are easier to find and yield better dividends in terms of modeling temporal and structural characteristics of the graph.

We limit our motif library to 4-order motifs. The selection of d-order motifs to include in the search library has been influenced by previous research in this area, functional interpretation of the motifs in real-world domains, and computational pragmatism. In addition to the higher-order motifs (d $>$ 2), we also make use of a few *fringe* motifs that provide insight about a complex network that is not captured by such higher-order motifs. m_1 and m_2 correspond to isolated vertices and isolated edges in the network that are not part of any higher-order motif. An abundance of such motifs is a clear indicator of a sparse, disconnected state of the network and is important to model some domains, such as power-grids [12]. Similarly, m_3 and m_4 correspond to self-loop and multi-edges between the same set of entities. Frequencies of such motifs show important functional properties of the network and can be used to convert it into a smaller weighted network, where the self-loops and the multi-edges are converted into vertex and edge weights, respectively. At the same time, they also contribute to the combinatorial explosion of the higher-order motifs. The current set of motifs also allows us to analyze multiple domains without mining important subgraphs specific to that domain. While such subgraphs may better represent the domain, they require time and data to discover and would need to be limited in size to avoid the search complexity. We focus on using ITeM distribution for various downstream graph applications such as summarization, generation, and classification.

Dyads and triads are the most used motifs to model complex networks. Larger acyclic and dense patterns do not uniquely explain different phases of temporal diffusion, whereas both the triads and linear chains do a better job [43]. Motifs such as m_5 and m_6 are examples of feed back and feed forward loops [5] and are fundamental to understanding transcriptional regulation networks [34], social networks, and biological systems. Adversarial activities exhibit patterns such as m_7 and m_9 among groups of adversaries, representing interactions such as communications and procurements [11]. m_7 is also found on or around structural hubs in brain networks [16]. The simple 4-cycle motif m_8 is an easy to find and informative structure. Star motifs m_{10} and m_{11} are ubiquitous in many social networks. Two-hop paths such as m_{12}, m_{13}, and m_{14} are essential to understanding air-traffic patterns [5] and procurement patterns [11]. Directed wedges such as m_{12} and m_{13} are also fundamental building blocks of bipartite graphs, which by definition do not show any triad or 4-cycle motifs. We also define *residual*

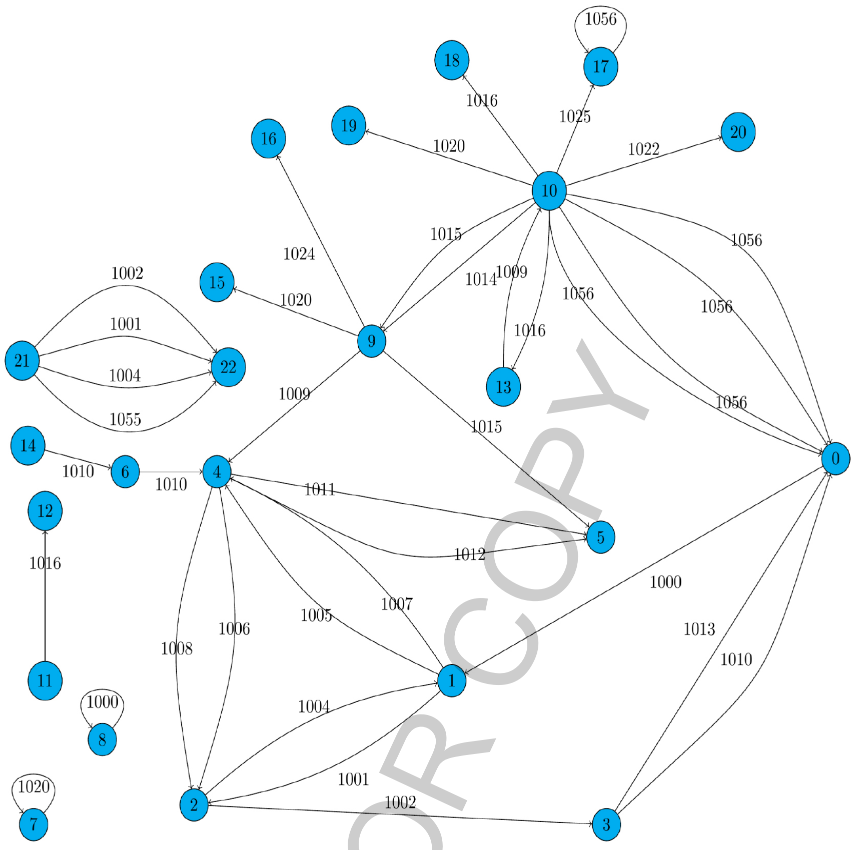


Fig. 3. Example input graph.

edge motif m_{15}, which is a single edge, 2-vertex motif that represents instances of the interactions that are not discovered as part of any higher-order motif pattern and can be generated using a randomized network. The *residual edge motif* m_{15} differs from *isolated edge motif* m_2. An isolated edge motif m_2 is disconnected from the rest of the graph and the number of m_2 instances is a characteristic of the domain represented by the temporal graph. It is searched before any higher order motif is searched in the graph. In contrast, m_{15} represents the leftover edges in the graph. At the end of the search order, these are not discovered as part of any larger motif. So the number of m_{15} instances depends on motif search algorithm. Figure 3 shows an example graph to illustrate atomic and temporal motifs observed in real-world graphs. Every edge in the input graph can be represented as a triple $<$ *source_id*, *destination_id*, *edge_time* $>$ and describes a temporal interaction between *source_id* and *destination_id* at a time (*edge_time* > 0). As shown in Fig. 3, the edge $< 11, 12, 1016 >$ is an example of a temporal edge between nodes 11 and 12 at a time $= 1016$. This edge is discovered as m_2 since it is disconnected from the rest of the input graph. Whereas, edge $< 10, 17, 1025 >$ is never discovered as a m_2 but it may be a *residual edge* if it is not part of any other motif type around nodes 10 and 17.

3.2. *Temporal motif*

Definition 2. Temporal Motif: A Temporal Motif $\mathcal{M}_t = (V, E, \mathcal{T})$ consists of a connected graph with time-steps on edges where:

- V is a set of vertices of the motif.
- E is a set of edges $e \in E$, e: $(u, v, t), u \in V, v \in V, t \in \mathcal{T}$ where $\mathcal{T}$ is a set of time steps associated with motif edges.
- Edges have a temporal ordering such that for an edge e_1: (u_1, v_1, t_1) and e_2: (u_2, v_2, t_2) if $t_1 < t_2$ then e_1 arrives before e_2.

A Temporal Motif is a specialization of the atomic motif, where every interaction between two vertices occurs at a specific time-step. The time step t_e of an edge e defines a temporal ordering of the edge within the temporal motif $\mathcal{M}_t$. However, it does not correspond to the actual time of the interaction in the temporal graph. Using this definition, we extend the atomic motif to model its temporal evolution in terms of size and structure. Characterization of the temporal network using a set of static motifs can be misleading and inaccurate because the static motifs fail to capture the temporal properties of the network, such as the scale at which transactions occur [5], burstiness of the transactions, and temporal dependency among the set of transactions. Additionally, many temporal systems are characterized as a dense *multi-graph*, where a pair of entities share many temporal transactions as the network evolves. This poses additional combinatorial complexity challenges beyond discovering structural motifs in the network. Figure 5 shows a set of temporal motifs used in this work. We extend the atomic motifs defined in Fig. 2 and measure two additional properties of a sub-structure using a temporal motif. Firstly, it measures the temporal ordering of the edges that informs the rate at which local interactions occur in a temporal graph. It also measures the change in the size of the temporal graph, in the number of nodes, by categorizing the motif nodes into two states: *new* and *reused*. Neither of these properties can be measured using atomic motifs since they don't encode *edge_time* and node state. Sections 3.3.1 and 3.3.2 provide concrete examples and more detail about using these properties.

3.3. Independent temporal motif (ITeM)

Schreiber and Schwobbermeyer [44] describe three different ways to measure the frequency of any pattern in a graph. They categorize them as $F1$, $F2$, and $F3$ *concepts*. In the context of motif computation, $F1$ includes every occurrence of a motif instance without any restriction, such as reusing a vertex or an edge while computing the frequency of motif instances. Paranjape et al. [40] use this definition to compute overlapping δ-motif frequencies. $F2$ and $F3$ concepts put restrictions on the reuse of a vertex or edge. $F2$ is an edge-disjoint concept and does not allow the reuse of an edge in more than one instance of the motif. Similarly, $F3$ is more restrictive as it is a vertex and edge-disjoint concept and does not allow reuse of any vertex and edge in more than one instance of the motif.

A major contribution of our work is the ITeM, which is an edge-disjoint temporal motif such that no two motif instances share any edge between them. ITeM is different from the temporal network modeling approaches mentioned in the related work, which use overlapping motif instances where some of them can share any number of edges. Overlapping motif instances can be used to model a network where it is common to have nodes and edges participate in multiple functional processes such as biological networks [9] but fails to model a network where each edge represents one transaction between two entities as in communication and financial networks. Independent motif instances capture a more accurate state of the network as no two transactions are part of any two motifs. Figure 4 shows a set of ITeM instances discovered for the input graph shown in Fig. 3. ITeM provides a lossless characterization of the input graph and can be used to reconstruct the input graph. Algorithms presented in Section 4 use the order defined in Fig. 5 to discover ITeM instances. ITeMs also define the temporal ordering of the edges within a motif. Such an ordering is independent of the actual timestamps used to define the temporal

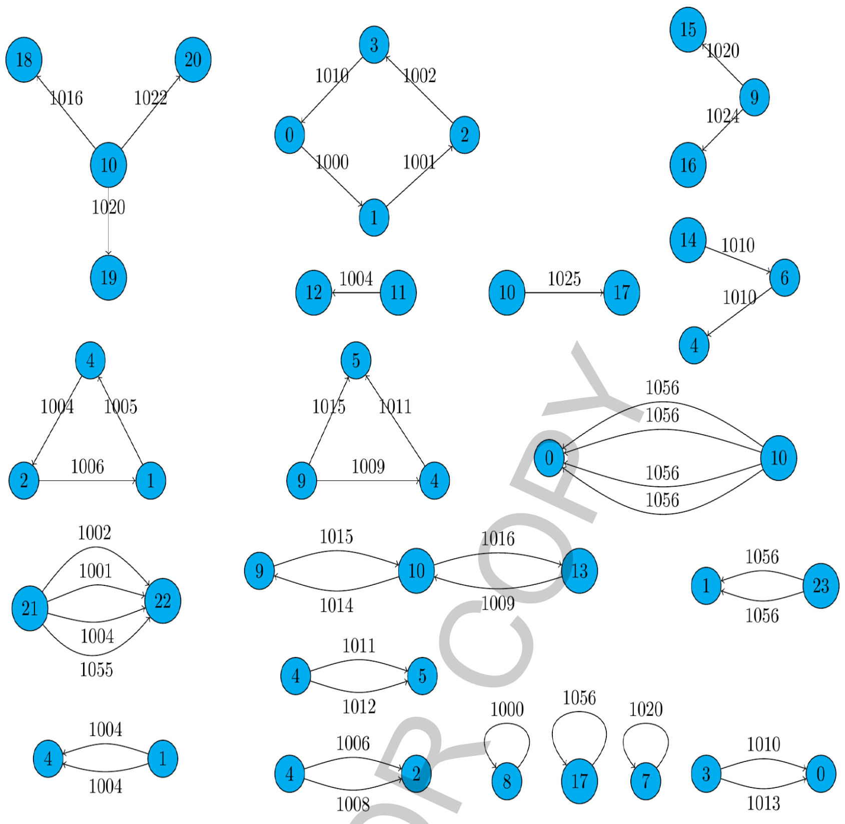


Fig. 4. ITeMs for example input graph.

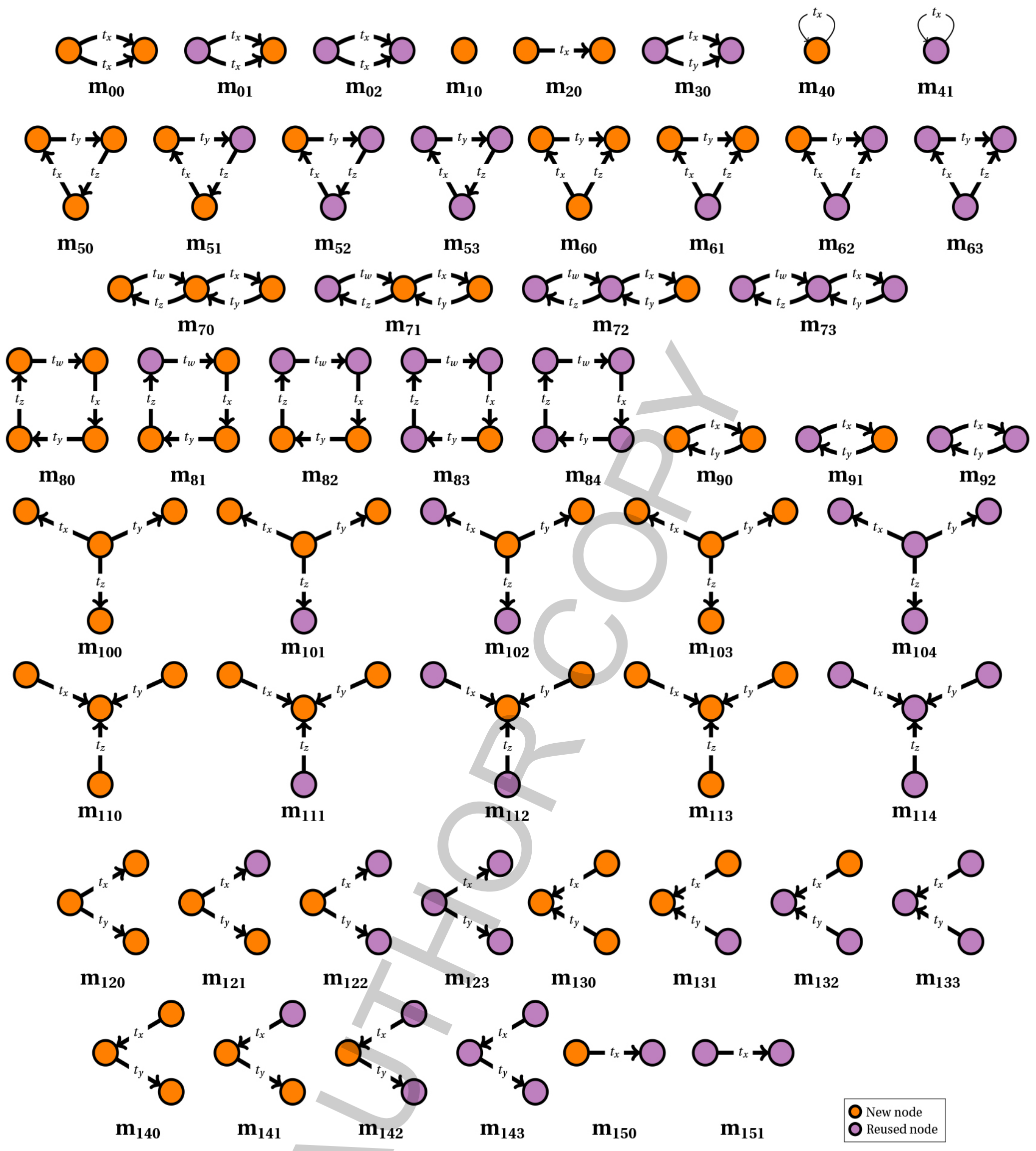
edges. This flexible approach allows us to use ITeMs with different granularities of the temporal graph. Additionally, for a temporal network, ITeM can be used to model the rate at which the network grows as it distinguishes between adding a transaction using new nodes to the network and reusing them for multiple future transactions. Overlapping instances fail to capture this phenomenon as they compute all the isomorphic instances of a motif type. This restriction also poses a greater complexity issue as finding temporal motifs is proved to be an NP-Complete problem [30]. In the following sub-sections, we define some key concepts used by ITeM to model a temporal network.

3.3.1. Vertex birth-time

We define the *birth-time* of a vertex in the temporal network as the time of the first transaction involving the vertex. The *birth* of a vertex increases the network size by one vertex. For the rest of the life of the network, that entity is treated as *reused* and it never increases the network population.

3.3.2. Structural contribution

Structural Contribution of an ITeM instance is a measure of the growth in the graph size as a result of adding the instance. The Structural Contribution of an independent temporal motif in terms of the number of edges is always equal to the number of temporal edges in the temporal motif. Figure 5 shows a set of temporal motifs and their *structural contributions*. As shown in Fig. 5, every instance of m_{62} adds three new temporal edges to an existing network. The *structural contribution* in terms of the number of vertices is impossible to measure using static atomic motifs because an atomic motif instance fails



m00 m01 m02 m10 m20 m30 m40 m41
m50 m51 m52 m53 m60 m61 m62 m63
m70 m71 m72 m73
m80 m81 m82 m83 m84 m90 m91 m92
m100 m101 m102 m103 m104
m110 m111 m112 m113 m114
m120 m121 m122 m123 m130 m131 m132 m133
m140 m141 m142 m143 m150 m151
New node
Reused node

Fig. 5. Library of temporal motifs used in this work.

to distinguish between the introduction of a new vertex to the network and reusing an existing vertex. Temporal motifs are required to encode this information to model the size and structure of the graph as it evolves. As shown in Fig. 5, every instance of the temporal motif m_{62} adds only one new vertex to an existing network. Whereas, every instance of the temporal motif m_{60} adds three new vertices to the temporal network.

3.3.3. Motif orbit

An orbit of a motif is defined as distinct positions in which a vertex can appear within the motif. An o-orbit motif is defined as a motif with o distinct vertex positions. A motif of d nodes can have maximum d orbits depending on its structure. A directed edge is a simple example of 2-orbit motif as a vertex can have two distinct positions: source or destination. As shown in Fig. 2, star motifs m_{10} and m_{11} have two orbits each: center and periphery of the star. Whereas m_5 has just one orbit but m_6 has three different orbits. The orbit of a vertex in a motif encapsulates its functional role within the motif. A combination of *structural contribution* and a change in the orbit of vertices allow us to model the evolution of a network without measuring the frequency of every *automorphic* instance. Graph automorphism is a measure of the symmetry of a structure. It is defined as a mapping from the vertices of a given graph T to itself.

3.3.4. Independence

We also define *Independence* of a temporal motif as a measure of its uniqueness in a given temporal graph. The *independence* can be measured for temporal motifs, temporal edges, or vertices of the temporal graph. The edge-disjoint concept defined above leads to maximal independent temporal edges because every edge has a bijection to the set of independent temporal motifs. We define the independence of a temporal motif and a vertex as follows:

Definition 3. Motif Independence: For a given temporal motif m_k, the independence of the motif is defined as a ratio of the number of ITeM instances to the number of overlapping motif instances.

$$DM_k = \begin{cases} \frac{|\hat{M}_k|}{|M_k|}, \text{if } |M_k| \geqslant 0 \\ 0, \quad \text{otherwise} \end{cases}$$

where $|\hat{M}_k|$ is the total number of ITeM instances, and $|M_k|$ is the total number of motif instances ($|M_k| \geqslant |\hat{M}_k|$).

This frequency-based metric identifies unique temporal motifs in the graph. Highly independent motifs exhibit the lower average cost of finding isomorphic combinatorial instances because of their uniqueness.

Definition 4. Vertex Independence: For a given temporal motif m_k, independence of the involved vertices is defined as a ratio of the number of unique vertices in ITeM instances to the maximum number of vertices possible in those instances.

$$DV_k = \begin{cases} \frac{|\hat{V}_k|}{|M_k * v_k|}, \text{if } |M_k| \geqslant 0 \\ 0, \quad \text{otherwise} \end{cases}$$

where $|\hat{V}_k|$ is the number of unique vertices in the ITeM instances of the k^{th} motif, $|M_k|$ is the total number of motif instances, and v_k is the number of vertices in the k^{th} motif.

Temporal motifs with high vertex independence lead to high *structural contribution*, whereas low vertex independence leads to co-located independent temporal motifs with a higher number of shared vertices among them.

4. Approach

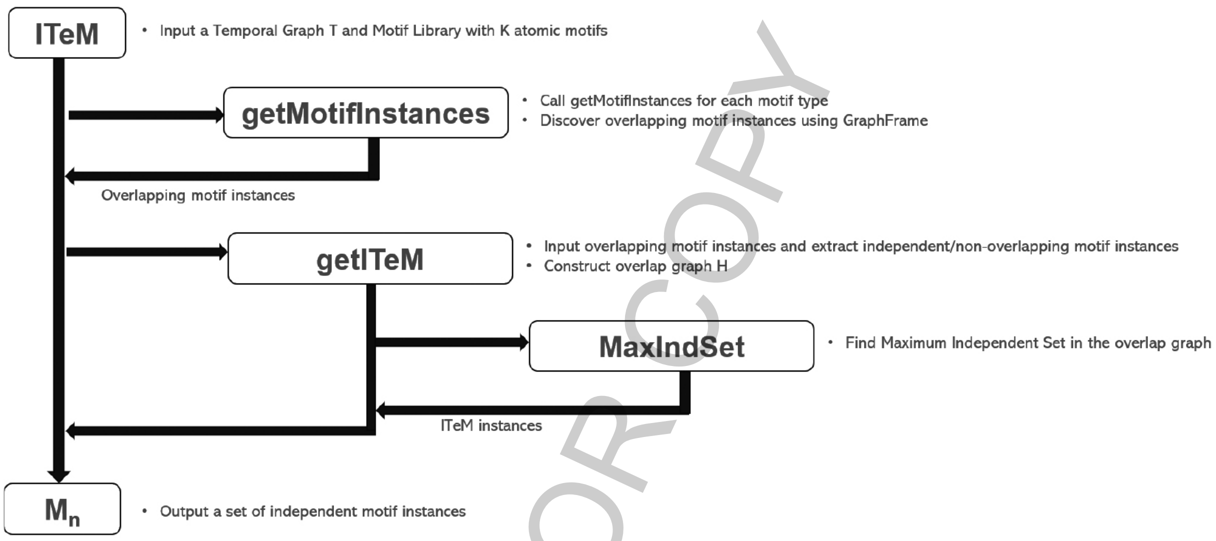
In this section we present the algorithm to count ITeM frequency. We also present a variant of it using window-based Importance sampling.

4.1. Algorithm to count ITeM frequency

Figure 6 shows a simple flow diagram for ITeM discovery. The ITeM discovery process takes a temporal

Algorithm 1: ITeM (T, K)

```
Data: $T$: Temporal Graph
Data: $K$: Set of Atomic Motifs
Result: $M_n$: Independent motif instances
1 foreach $m_k \in K$ do
2     $M \leftarrow$ getMotifInstances $(m_k, T)$
3     $M_n \leftarrow M_n \cup$ getITeM $(M)$
4 end
5 return $M_n$
```



ITeM
• Input a Temporal Graph T and Motif Library with K atomic motifs
getMotifInstances
• Call getMotifInstances for each motif type
• Discover overlapping motif instances using GraphFrame
Overlapping motif instances
getITeM
• Input overlapping motif instances and extract independent/non-overlapping motif instances
• Construct overlap graph H
MaxIndSet
• Find Maximum Independent Set in the overlap graph
ITeM instances
M_n
• Output a set of independent motif instances

Fig. 6. ITeM discovery algorithm flow diagram.

graph and a library of atomic motifs as input and iteratively measures the count and temporal evolution of the motifs. ITeM is a simple and efficient approach to graph characterization because of an abundance of small motifs in real-world systems. Algorithms 1–3 present the pseudocode to find independent temporal motif instances in a given temporal graph. Algorithm 1 inputs a temporal graph and a set of atomic motif types to discover as shown in Figs 3 and 2 respectively. Line 1 discovers all overlapping motif instances of a given motif type m_k. We generate all the temporal motif types corresponding to m_k (Fig. 5) and use GraphFrame [13] to discover the overlapping temporal motif instances. Overlapping motif discovery is a run-time bottleneck and GraphFrame provides optimized motif discovery using graph-aware dynamic programming algorithms. It also provides a simple Domain-Specific Language (DSL) to express all the temporal motifs. Algorithm 2 inputs a set of overlapping temporal motif instances discovered in Line 1 and returns ITeM instances (as shown in Fig. 4). We use the temporal ordering of the edges to define $\mathcal{L}(m)$, a lexical representation of the motif instance. The lexical representation is used as a vertex label to construct a motif overlap graph H. The motif overlap graph H is an abstract graph that represents clusters of motif instances sharing at least one edge in the input graph T as defined in Definition 1. Lines 2–6 map an edge and its associated set of motif instances. Lines 8–10 create a set of vertices H_v in the abstract graph. Lines 12–16 construct an edge-list H_e using all the motifs that share a temporal edge in the input graph. H_e is constructed by creating an edge in the abstract graph H for every shared edge in the input graph T. H_e and H_v are used to construct the abstract graph H on Line 18. The final

Algorithm 2: getITeM (M)

```
Data: $M$: All motif instances
Result: $M_n$: Independent motif instances
/* Create a mapping $EM$ between an edge $e$ and all associated motif instances. $\mathcal{L}(i)$
   is the string representation of a motif instance $i$.                         */
foreach $i \in M$ do
    foreach $e \in i$ do
        $EM(e) \leftarrow EM(e) \cup \mathcal{L}(i)$
    end
end
/* For every motif instance label $i$, create a vertex in the overlap graph.      */
foreach $i \in M$ do
    $H_v \leftarrow H_v \cup \mathcal{L}(i)$
end
/* Create an edge in the overlap graph, between every motif-instance label pair
   $(l_r, l_{r+1})$ that share an edge $e$ in the input graph.                     */
foreach $e \in EM$ do
    foreach $(l_r, l_{r+1}) \in EM(e)$ do
        $H_e \leftarrow H_e \cup (l_r, l_{r+1})$
    end
end
/* Create the motif overlap graph                                                  */
$H \leftarrow G(H_v, H_e)$
/* Find non-overlapping temporal instances                                         */
$M_n \leftarrow MaxIndSet(H)$
return $M_n$
```

Algorithm 3: MaxIndSet (H)

```
Data: $H$: An undirected abstract graph
Result: $I$: Maximum Independent set of vertices
/* Set every vertex in its own Independent Set                                     */
foreach $v \in H_v$ do
    $I_v = \mathcal{L}(v)$
end
repeat
    send $I_v : v \in H_v$ to every $u \in Neighbor(v)$
    receive $I_u$ for every $u \in Neighbor(v)$
    update $I_v$ by the lowest $I_u$ received
until $I_v$ does not change;
/* Get Independent Set as unique values of $I_v$                                   */
return $unique(I)$
```

result is computed using Algorithm 3 on Line 20, which uses a distributed Maximum Independent Set implementation to compute the ITeM instances. The ITeM instances represent a set of edge-disjoint motif instances in the input graph. Since finding matches to temporal motifs is proved to be an NP-Complete problem [30], We use Luby's Algorithm [33] to discover ITeMs which provides a lower bound on the ITeM frequency.

Algorithm 3 presents the pseudocode of a distributed implementation of the MIS algorithm. We use Pregel API, available in Apache Spark, to implement Luby's Algorithm [33]. We initialize all vertices in their own independent set as shown in lines 2–4. At lines 5–9 of Algorithm 3, each vertex exchanges messages with its neighbors and updates its independent set value based on the minimum values received from all neighbors. This process stops when no vertex in the graph changes its independent set.

4.2. *Complexity analysis*

Performance of Algorithms 1–3 is influenced by different structural properties such as degree distribution and clustering of a real-world graph. Worst-case performance is observed for complete large graphs, but the real-world graphs are not fully connected and instead show domain dependent characteristics such as power-law degree distribution, low clustering coefficient, and shrinking diameter as the graph evolves. For the maximum number of edges E in a given window, the Algorithm 1 has $O(|E|^2)$ space and time complexity, although for real-world graphs it is constrained by the factor of maximum motif size d. We also extract multi-edges from the input temporal graph which leads to space and run-time improvements. This leads to $\Omega(\lceil E/d \rceil)$ space complexity to discover overlapping instances at line 2 of Algorithm 1. Similarly, Algorithm 1 has $O(|V|^d)$ time complexity [50]. Algorithm 2 creates a huge overlapping graph as shown in lines 12 to 18. The worst-case space complexity for a fully connected graph is $O(|E|^4)$, but due to optimizations in Algorithm 1, Algorithm 2 has $O(|E|^2)$ space complexity. The approximate Algorithm 3 uses graph message-passing techniques to identify independent sets in the overlapping graph and has $O(\log(n))$ run-time complexity [33] that leads to $O(\log(|E|))$ run-time complexity for Algorithm 3. The overall space and run-time complexities for ITeM discovery are $O(|E|^2)$ and $O(k * \log(|E|))$ respectively, for total k motif types shown in Fig. 5.

4.3. *Importance sampling to count ITeM frequency*

Our approach includes three major algorithmic components: searching for overlapping temporal atomic motifs, finding independent temporal motifs, and computing information content and temporal evolution of such motifs. Out of the three components, finding independent temporal motifs is an NP-Complete problem, and we use a heuristic to find a lower bound of the actual count. As explained in the previous section, we construct a motif overlap graph where every vertex is a motif instance and an edge between two vertices exists if the corresponding motif instances share an edge in the original temporal graph T. This abstract formulation may lead to a highly-cliqued abstract graph, which is a characteristic of various real-world domains, such as a social network. A highly-cliqued abstract graph leads to excessive message-passing in the distributed computing environment. To address this, we use an *importance* based sampling approach to approximate the $F2$ motif frequency computation.

Importance sampling for motifs is presented by Liu et al. [30]. The importance sampling is based on the assumption that each distribution has some interesting or important regions and the samples drawn from those regions must be normalized to get an unbiased estimate [38]. Window-based importance sampling splits the time series dataset into multiple temporal windows and performs computation on each window. We create window graphs with equal temporal window size, each with a different number of edges within the window. Each window is assigned an *importance*, based on the fraction of all the edges present in the window. The importance is used to normalize the computed metric across all randomly-selected windows. The normalization reduces the overall variance for real-world domains that do not show a *burst*. The current approximation approach does not model such anomalies in the ITeM distribution but allows us to model the evolution of a network as shown in Section 5. Future work will address this challenge using an importance decay approach that gives more importance to recent windows. We compute the distribution of all temporal motifs present in the window graph. At the end of all the windows, we compute the weighted average of all the distributions, which gives a lower bound estimate of the distribution that can maintain a relative error tolerance of 5% in the count [30].

For a given temporal graph T with t windows, the importance vector $\mathcal{I}$ is an ordered sequence of *window importance* I_i: $\mathcal{I} =< I_1, I_2, \dots, I_{t-1}, I_t >$ where the I_i is defined as: $I_i = \frac{|E_i|}{|E_T|}$ where E_i is the number of edges in a window i and E_T is the total number of edges in the temporal graph. For a given motif m_k, the expected motif frequency $\mathcal{F}_k$ in the temporal graph can be computed from the frequency Δ_{ki} of the motif in the i^{th} window with importance I_i as:

$$f_{ki} = \frac{\Delta_{ki}}{I_i} \quad \text{and} \quad \mathcal{F}_k = \frac{1}{t}\sum_{i=1}^{t} f_{ki}$$

We also define a random variable $X_i \in \{0,1\}$ that selects a specific window in the entire population. The expected frequency $\mathcal{F}_k$ is computed as:

$$\mathcal{F}_k = \frac{1}{t_x}\sum_{i=1}^{t_x} X_i * f_{ki}$$

where t_x is the number of windows selected ($X_i = 1$) for the ITeM disovery. The ITeM distribution $\mathcal{F}$ for a given temporal graph is the distribution of all such temporal motifs over the window population. $\mathcal{F} =< \mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_K >$ where $|K|$ is the total number of motifs.

5. Experiments

To evaluate the performance and scalability of our approach, we analyzed a rich set of synthetic and real-world temporal datasets. Datasets used in this study exhibit real-world properties of different domains such as sparsity, preferential attachment, and long-tail degree distribution. The synthetic dataset is generated with Gaussian noise in temporal edges. The real datasets show temporal bursts, observed in social networks. The experiment provides support for our following core contributions:

– ITeMs are a novel way of capturing discerning temporal properties of a temporal network that cannot be measured using static and overlapping temporal motifs.
– ITeMs outperform the Stanford SNAP temporal motif algorithm (referred as δ-Motif hereinafter) and Dynamic Graphlet (DG) [17] in measuring the similarity of temporal graphs.
– Our approach is scalable and configurable to analyze a temporal network as one large graph or a sequence of windows using sampling.

All the experiments are done on a cluster using Apache Spark 2.3.0 and GraphFrame 0.7.0. All the algorithms are implemented in Scala 2.11.8, and the source code is available at https://github.com/temporal-graphs/STM.

5.1. Results on synthetic networks

ITeMs can efficiently model the evolution of a temporal network using the properties defined in the section above. To present the accuracy of modeling temporal changes in the network using ITeMs, we generate a set of synthetic temporal graphs using a stochastic generation method and measure the change in the similarity as the networks evolve. We benchmark against δ-Motif and DG and show that ITeMs are better at measuring the changes in the similarity as the networks evolve. For a given population size $|V| = 100$, we create a temporal graph G_0 of one-day time duration, where every vertex creates an edge with a random target vertex with a low probability p at every second. Then, we create variations of the base graph by stretching it one day at a time and perturbing timestamps using a Gaussian distribution with zero mean and 1/6 day as standard deviation. We create thirty such variations ($G_1, G_2, \ldots, G_{30}$). For example, the time between edge arrivals in G_{10} is 10 days longer than in G_1. All the graphs in the sequence have

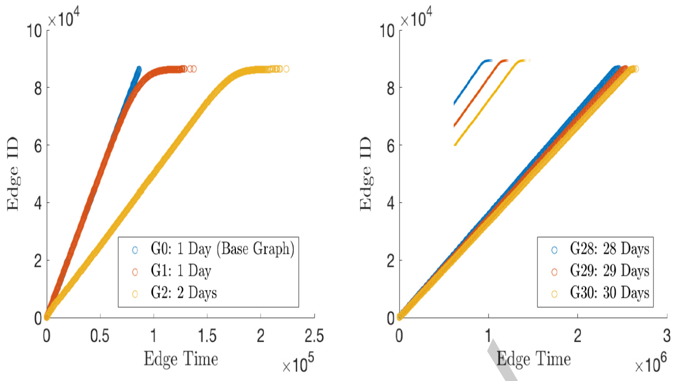


Fig. 7. Synthetic graphs.

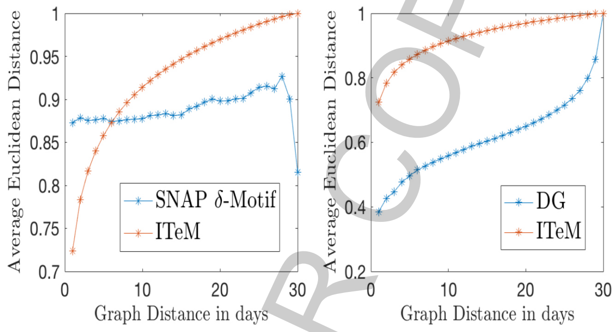


Fig. 8. Temporal graph similarity.

the same structure and only the edge timestamps vary. This allows us to compare capabilities of δ-Motif, DM, and ITeM to measure temporal variations in graphs.

Figure 7 shows the rate of the addition of temporal edges to the graph. We also show a zoomed-in version (right) of G_{28}, G_{29}, and G_{30} to visualize linearity in the temporal stretch as we increase the total time of the graph. We compute motif frequencies using both algorithms. Similarly, we also compute temporal, structural, and orbital features using our ITeM approach. For a given approach, we compute a feature matrix with 31 rows where each row corresponds to one synthetic network. Additionally, each row represents a fixed-column vector where the length of the row corresponds to total features computed by the tool. These feature vectors are used to measure the pairwise similarity of the temporal networks. Once we compute the pairwise similarities for all the networks, we aggregate them for the networks with same temporal stretch. This explains Fig. 8 where each entry (i, j) represents j avg. Euclidean distance for all the networks with the total duration i days apart. The experiment is repeated for all the three approaches. For ITeM, we use non-sampling algorithm as described in Sub-section 4.1.

Figure 8 shows the change in normalized graph similarity as a function of the difference in the time duration of the synthetic graphs. A point (i, j) on the plot represents the average Euclidean distance j over all the graphs that are i days apart. Intuitively, the set of graphs that are i days apart have lower average Euclidean distance than the set of graphs that are $i + 1$ days apart. Also, the difference in two average Euclidean distances decreases as i increases. The δ-Motif allows the use of arbitrarily large δ values (the limit on the time window spanned by motifs), and we use this feature to identify motifs without any temporal restriction on the time difference between any two motif edges. Figure 8 (left) shows that the

Table 3
Temporal graphs datasets

	$\|V\|$	$\|E_{temporal}\|$	$\|E_{static}\|$	Time
CM	1,899	59,835	20,296	193 days
BA	3,783	24,186	24,186	1,901 days
EE	986	332,334	24,929	803 days
TT	34,800	171,403	155,507	21 hours
IA	545,196	1,302,439	1,302,253	1,153 days
HT	304,691	563,069	522,618	7 days
RH	55,863	571,927	561,483	3 yrs 4 mos
WT	1,140,149	7,833,140	3,309,592	6 yrs 4 mos

temporal-spatial-orbital features computed by ITeM outperform graph similarity accuracy using δ-Motif features that are based only on motif counts. The δ-Motif does not capture the temporal variations of discovered motif instances, whereas ITeM can successfully measure it as the graph is *stretched* in time and the average δ time between edges and the time to form a motif increases. For maximum distant graphs such as G_0 and G_{30}, we observe an unexpected sharp change in the similarity using δ-Motif. This requires a deeper analysis of the algorithm and the output generated by the tool.

DG also characterizes a temporal network in terms of *graphlet* count for the entire network and individual nodes. DG also provides a δ parameter to restrict time difference between two edges of the graphlet, but due to out-of-memory errors, we could not run it in the unbounded setup that was used in the previous experiment. To benchmark against DG, we used a δ restrictive mode of our algorithm with δ set to 600 seconds.

Figure 8 (right) shows the result comparing DG and ITeM. As shown in the Fig. 7, the base graph shifts from a stochastic base model to a Gaussian distribution based temporal network, which explains the initial sharp increase in the graph distance measured by both algorithms. Both the approaches also show sub-linear trends afterward but only ITeM continues as the time difference between graphs increases. DG shows sudden exponential changes in the distance (or similarity) that do not correspond to the linear temporal evolution of the graphs as shown in Fig. 7 (right). Overall, both the approaches exhibit similar trends that show the importance of modeling temporal variations and orbital information of the graph, in addition to the frequency count.

5.2. Results on real-world networks

We analyze various real-world networks and measure the difference in their temporal evolution. The following list introduces all the datasets used for the experiments. Table 3 describes their static and temporal scale. We generate tITeM distribution and use it for the measurement. We also use the change in the distribution over time to detect an event in the network.

- **CollegeMsg (CM)**: CollegeMsg [39] is comprised of private messages sent on an online social network at the University of California, Irvine. An edge (u, v, t) means that user u sent a private message to user v at time t.
- **Bitcoin-Alpha (BA)**: Bitcoin-Alpha [24] is a who-trusts-whom network of people who trade on *Bitcoin Alpha* platform. An edge (u, v, t) in the network exists if person u gives a rating to person v at time t.
- **Email-EU (EE)**: Email-EU [27,54] is an anonymized network about all incoming and outgoing emails between members of a large European research institution. An edge (u, v, t) in the network exists if person u sent an email to person v at time t.

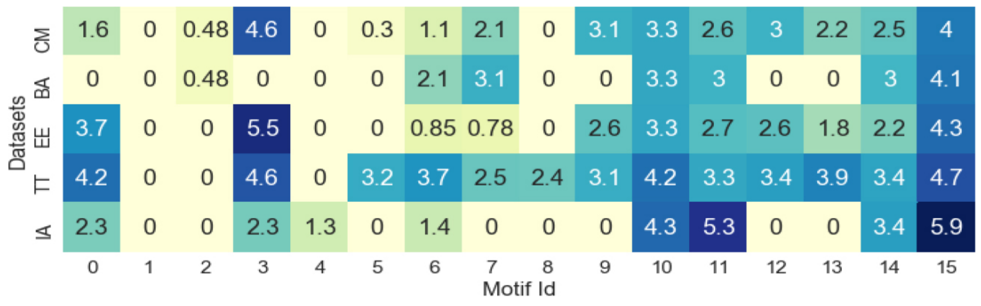


Fig. 9. ITeM distribution (log10) of different datasets.

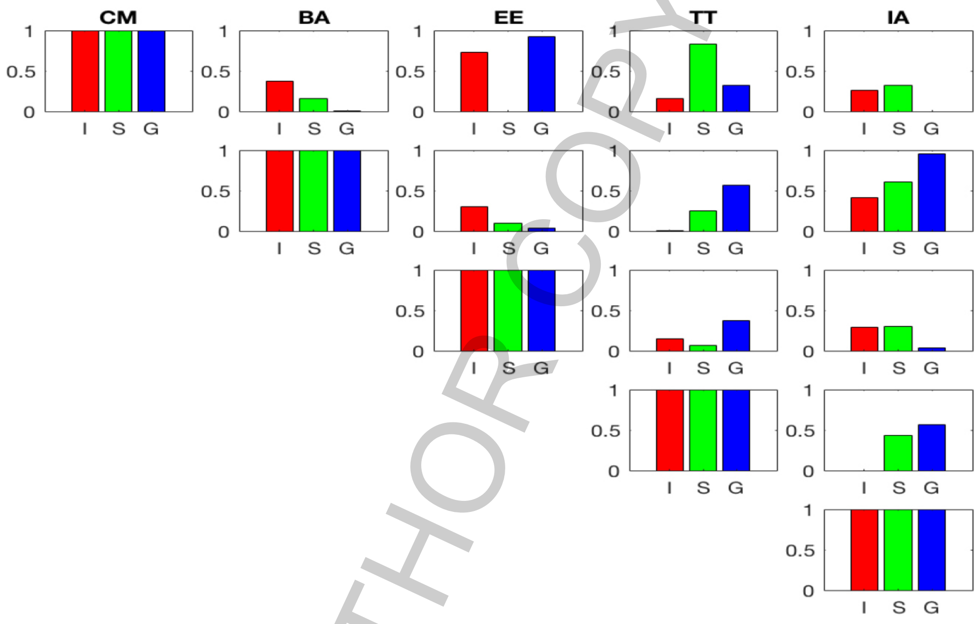


Fig. 10. Real-world graph similarity (y-axis) using ITeM, SNAP, and DG (x-axis) for CM, BA, EE, TT, and IA.

- **Tech-As-Topology (TT)**: Tech-As-Topology [42] is a temporal network of Autonomous Systems (AS) where an edge (u, v, t) represents a link between AS u and AS v at time t.
- **IA-Stackexch (IA)**: IA-Stackexch-User-Marks-Post is a bipartite Stack Overflow favorite network [42]. Nodes represent users and posts. An edge (u, v, t) denotes that a user u has marked a post v as a favorite at time t.
- **Higgs Twitter: (HT)** Higgs dataset [14] is an anonymized network that has information about messages posted on Twitter between the 1st and the 7th of July 2012 about the announcement of the discovery of Higgs boson particle. An edge (u, v, t) represents a Twitter interaction between user u and v at time t. An interaction can be a *re-tweet*, *mention*, or *reply*.
- **Reddit Hyperlink (RH)**: Reddit hyperlink [23] represents the directed connections between two subreddits. An edge (u, v, t) represents a hyperlink from subreddit u to subreddit v at time t.
- **Wiki-talk (WT)**: Wiki-talk [40] represents Wikipedia users editing each other's Talk page. A directed edge (u, v, t) means that user u edited user v's talk page at time t.

Figure 9 shows the independent temporal motif distribution of different datasets. Similarly, Fig. 11

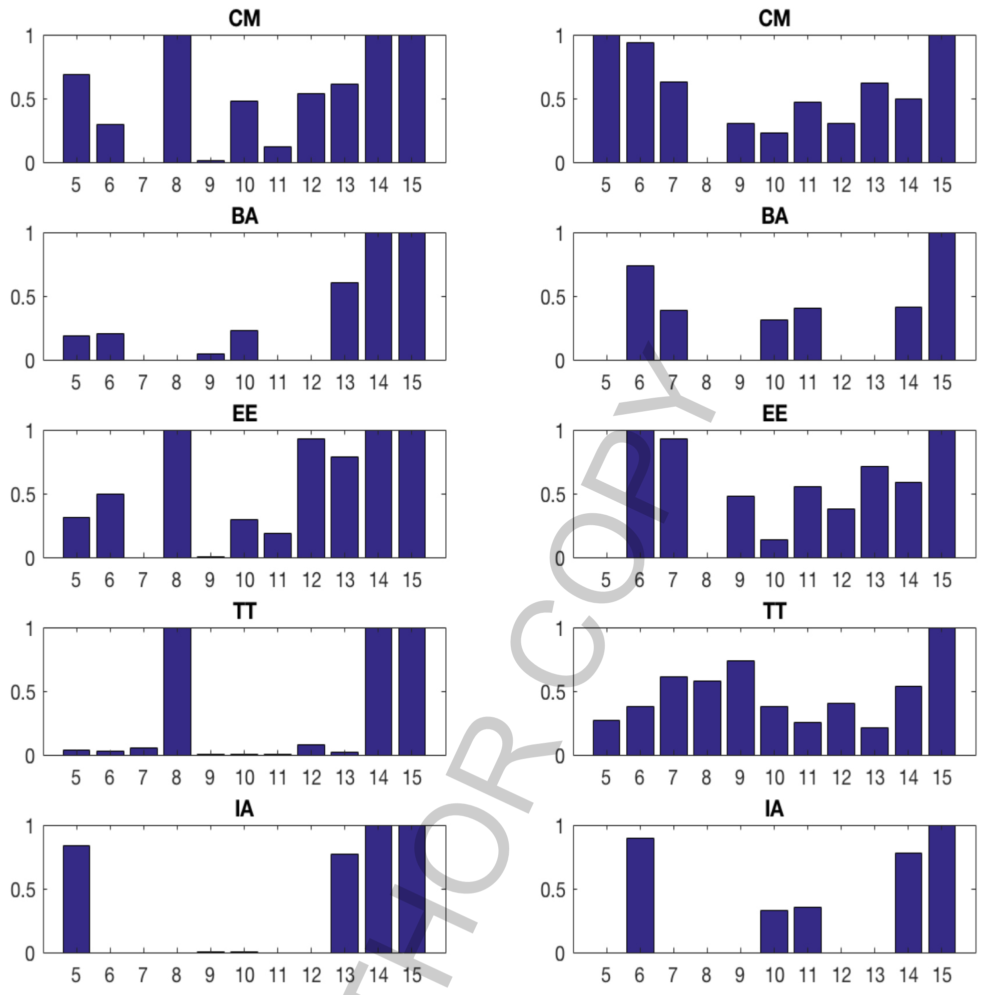


Fig. 11. Motif and vertex independence of different datasets. X-axis represents motif-id and y-axis represents motif independence (left) and vertex independence (right).

shows motif independence and vertex independence for the datasets. These results give initial clues that similar domain networks such as **CM** and **EE** exhibit similar motif and vertex independence, whereas **BA** and **TT** have a different distribution. We also analyze real-world datasets using ITeM, SNAP δ-Motif and DG. For ITeM, we use non-sampling algorithm as described in Sub-section 4.1. In the absence of any ground-truth, we observe the types of motifs and their frequencies discovered by the tools. We restrict our analysis to the motifs of maximum 4 vertices. ITeM can identify *fringe* motifs such as isolated nodes, isolated edges, and self loops. DG does discover single edges but not the isolated nodes and self-loops. Both DG and SNAP δ-Motif focus on connected networks but ITeM can also measure the fraction of the graph that exists as disconnected nodes and edges. All three tools discover multi-edges in the network.

We also compute network similarity across all network pairs, using Euclidean distance between the normalized frequency vectors. Figure 10 shows the similarity between each pair of real-world datasets using the three approaches. Both DG and ITeM identify CollegeMsg (**CM**) more similar to Email-EU (**EE**) than Bitcoin-Alpha (**BA**) and Tech-As-Technology (**TT**). We could not run DG in the unbounded setup so we restricted δ to 6000 seconds and that leads to single edge only motifs in the case of **BA** and

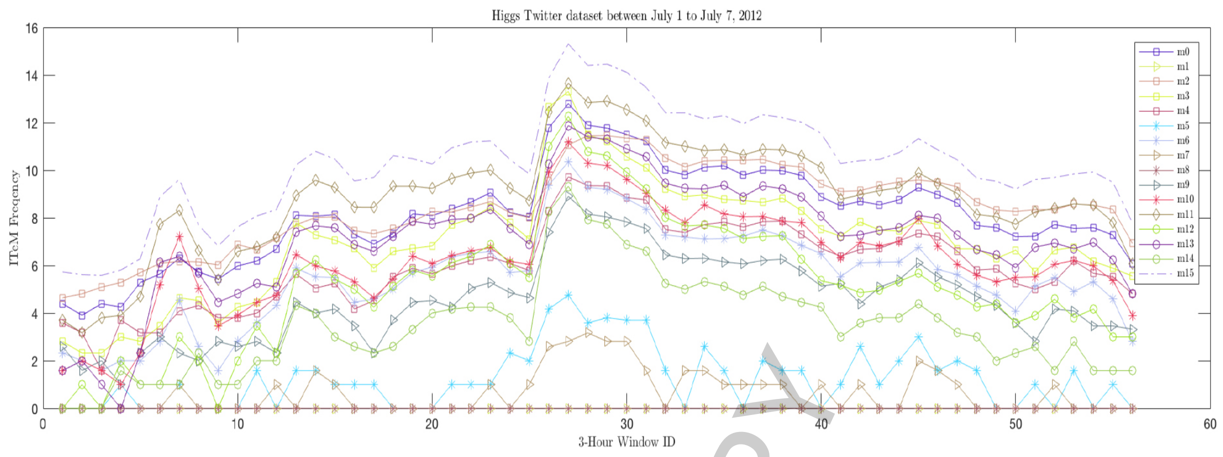


Fig. 12. ITeM frequency changes in the Higgs Twitter (HT) temporal network.

IA. Similarly, SNAP δ-Motif discovers few non-negative motif instances and we treat them as zero for the analysis. ITeM and δ-Motif also generate a fixed size feature vector for a given input which makes it easier to use in downstream applications.

5.2.1. ITeM-based temporal evolution measurement

ITeM can also model the temporal evolution of a network using a sequence of temporal graphs, each with a given time window. We use the Higgs Twitter (**IT**) dataset and monitor 3-hour windows from July 1st to July 7th. Our approach iteratively analyzes each window and updates the temporal summary of the network as it progresses. This allows us to not only analyze a large graph using multiple smaller graphs but also to identify an anomalous event in the network and to understand how the behavior of vertices changes in the temporal network. Figure 12 shows a change in ITeM frequencies to reflect a burst event in the graph. The ITeM frequencies peak at the event on July 4th and then gradually return to a normal state. ITeM also provides more insight into the event than basic graph density-based measures. As shown in Fig. 12, the maximum increase is observed in the *fringe* part of the network, such as self-loops, isolated edges, and residual edges. Similarly, a higher number of stars and wedges are also observed. These observations correspond to a network growth phenomenon where a burst of new interactions occurs in the network among newly-added entities. In the case of **HT**, this is explained by a higher number of Twitter users tweeting about the Higgs boson particle discovery for a short period of time.

Figure 13 shows motif independence over time for the same window of the **HT**. Figures 12 and 13 show that the *core* motif, such as the star, increases in count but the *motif independence* decreases sharply. This happens as the temporal network exhibits the emergence of a hub-like structure with a small number of extremely-high degree vertices. In contrast to the *burst* observed in the HT, Wiki-talk (**WT**) shows a linear evolution of the graph for a very long time (76 months) as shown in Fig. 16.

5.3. Scalability analysis

A major contribution of this paper is a distributed algorithm to analyze a large temporal graph or a sequence of temporal graph windows. All the algorithms are developed using the Apache Spark 2.3.0, GraphFrame 0.7.0, and Scala 2.11.8 environment. This allows the use of scalable distributed data structures to handle large graphs in the order of millions of edges and to iteratively update the

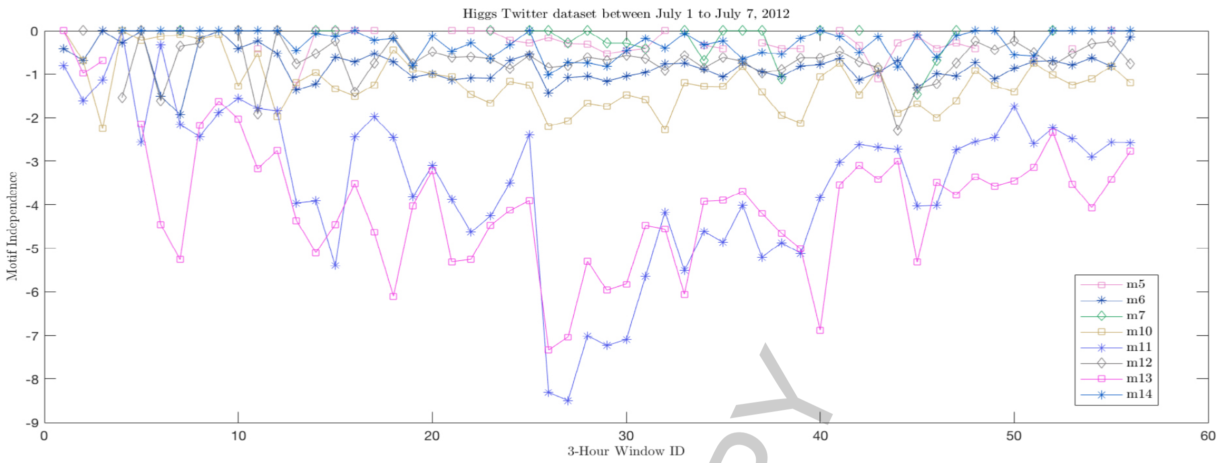


Fig. 13. ITeM Independence changes in the Higgs Twitter (HT) temporal network.

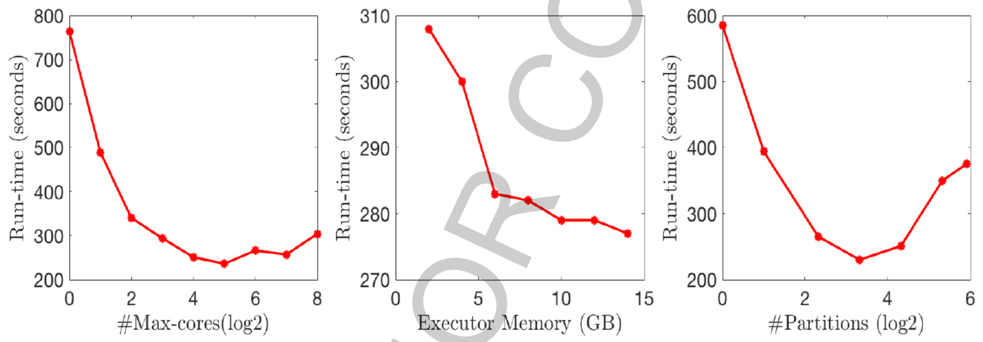


Fig. 14. ITeM runtime analysis on Email-EU (EE) dataset: Single Graph.

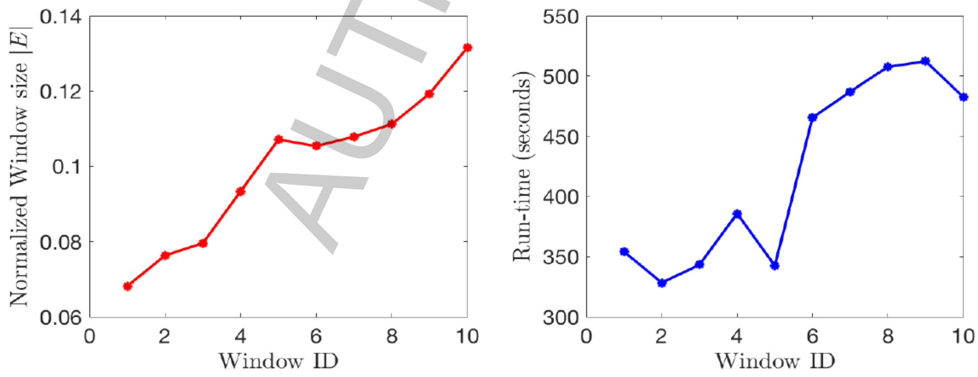


Fig. 15. ITeM runtime analysis on Reddit (RH) dataset: Sequence of temporal graphs.

temporal-structural and orbital properties of the graph. To analyze the scalability of the core algorithm, we use a Snakemake [20] based automation pipeline and a SLURM [55] based resource manager. We experiment with different combinations of hardware resources and distributed partitions. Figure 14 shows the results of the scalability experiment using the EmailEU dataset. ITeM shows initial speed-up up to a

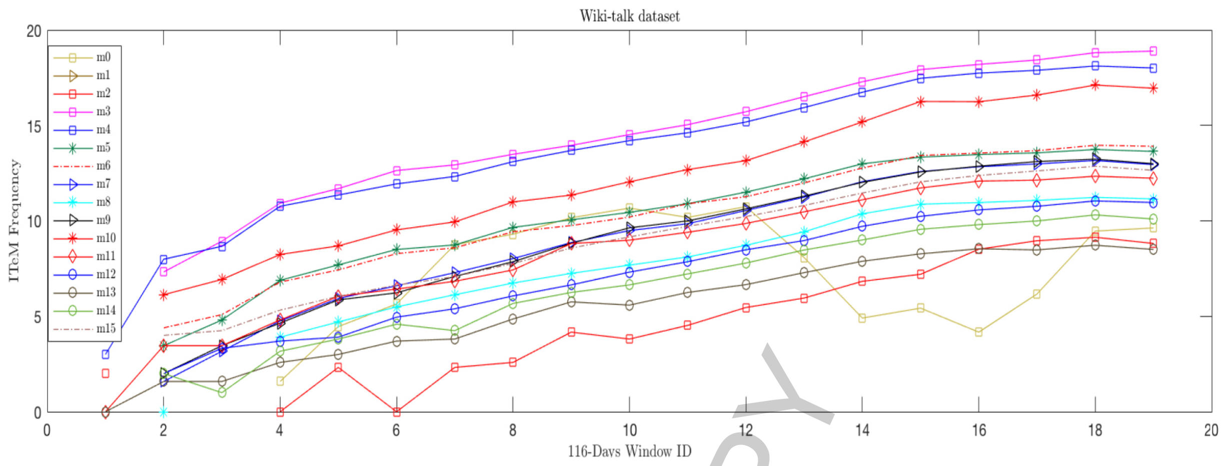


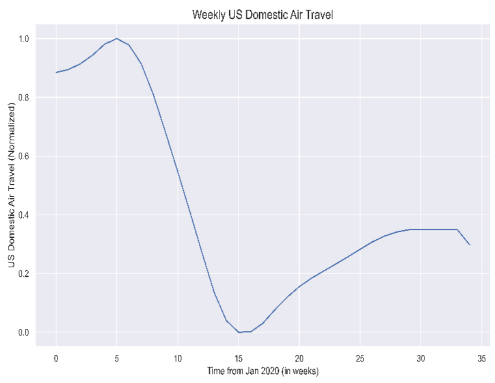
Fig. 16. ITeM frequency changes in the Wiki-talk (WT) temporal network.

maximum of 32 cores available to the Spark application. Beyond this point, the application suffers from communication and data serialization overhead. A similar trend was observed as we increased the number of data partitions, keeping the maximum number of cores fixed. The run-time sharply decreases as we increase the executor memory from 2 GB to 6 GB, and the decrease slows down after that.

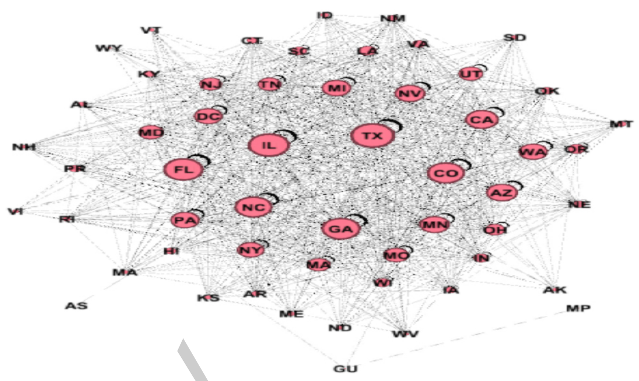
Temporal analysis of an evolving network using a window-based approach poses memory constraints and scalability challenges as the number of windows increases. We preserve minimum information across the windows to maintain a global summary of the temporal network and to save window-specific summaries and vertex features to files, to be used by other analytic processes. This allows us to use our method in a longer running streaming fashion. Although we do not observe a strong sub-linear trend as the windows progress, as shown in Fig. 15, further analysis of the window graph structure using ITeM suggests that the run times depend on both the window size and the *fringe* structure of the graph. The runtime of Window 5 and 10 decreases even as the graph size increases because those windows have a higher number of multi-edges in comparison to windows of similar size, which leads to aggressive subgraph reduction while discovering larger motifs. Future work will perform a more detailed analysis of the impact of a specific ITeM count on the runtime. For all three approaches, overall run-time complexity depends on enumerating larger motifs in the network but δ-Motif has developed a set of specialized algorithms that count certain motif classes faster. Similarly, DG uses *constrained dynamic graphlet counting*, a modified counting process to examine fewer instances of a given dynamic graphlet. In contrast, ITeM uses a general purpose framework to discover temporal motifs. This leads to faster run-times for δ-Motif and DG but ITeM provides a fault-tolerant framework to analyze large graphs. Future work will also develop specialized distributed algorithms to find certain classes of motifs instances.

5.4. ITeM-based analysis of real-world COVID-19 indicators

In this section, we present a real-world use case and ITeM-based analysis to extract actionable knowledge from it. Coronavirus Disease 2019 (COVID-19) is an ongoing outbreak and the latest threat to global health. Understanding the implications of social interaction on COVID-19 indicators is an important research objective to help formulate policies and guidelines by governments and local authorities. We use curated state-level COVID-19 indicators [48] such as Active Cases, Deaths, and Hospitalization Rates for the United States. We also curate domestic US air travel data as shown in Fig. 17a and present its impact on COVID-19 indicators.



(a) Weekly Travel Trend From Jan 2020-Aug 2020 for domestic US air-travel based on Bureau of Transportation Statistics data.



(b) Temporal Graph representing domestic US air-travel.

Fig. 17. Temporal Graph representing domestic US air-travel.

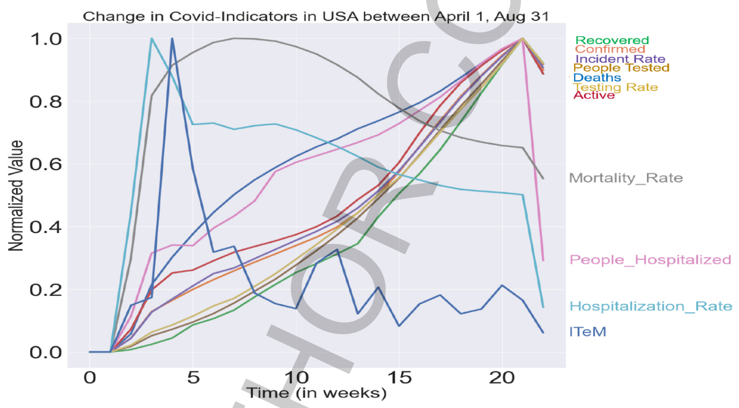


Fig. 18. Weekly real-world Indicators and ITeM trends from April 2020–Aug 2020.

We create a dynamic graph to model air travel between different US states as shown in Fig. 17b. Each US state is modeled as a node in the graph. We create edges using the total air-travel passenger count. We create a logarithmic bin to reduce the number of edges between any two states for a given day. We also create a sequence of weekly temporal graphs to measure temporal trends in the COVID-19 air-travel dataset. We generate weekly ITeM distributions to observe temporal patterns in the COVID-19 travel graph. The ITeM allows us to measure the air-travel magnitude and also the travel behavior between US states. In addition to weekly ITeM distributions, we also compute the weekly distribution of COVID-19 indicators such as Active Cases, Deaths, and Hospitalization Rates for the United States. As shown in Fig. 18, we qualitatively observe three different classes of real-world COVID-19 indicators. The "Hospitalization Rate" and the absolute number of people hospitalized in the US averaged over a week show a similar trend. In contrast, indicators such as "Active Cases", "Confirmed Cases", "Deaths" show a similar upward trend. The "Mortality Rate" does show an initial sharp upward and slowly decreasing trend in contrast to all the other indicators.

We use Dynamic Time Warping (DTW) [6] to quantify ITeM similarity with the rest of the indicators. DTW measures the similarity (or distance) between two time series by computing optimal matches

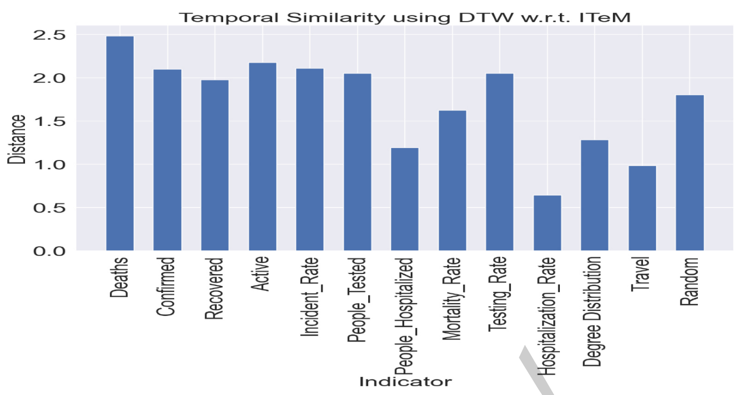


Fig. 19. COVID-19 real-world indicator similarity with ITeM distribution.

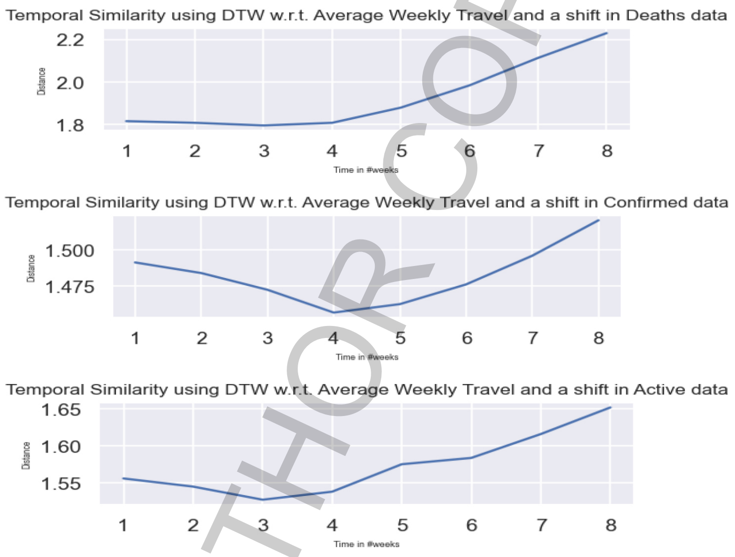


Fig. 20. Average travel trend follows COVID-19 real-world indicators with a shift of one month.

between them. We compute the pair-wise distance between all real-world indicators and ITeM distribution for each week from April 2020 to August 2020. We used the degree distribution of the travel graph as the baseline similarity with each indicator. ITeM distribution shows high similarity with average travel and "Hospitalization Rate" as shown in Fig. 19.

ITeM can also be used as an early indicator of a shift in a real-world indicator. COVID-19 contact tracing research has estimated a 2 to 14 day incubation period for the virus [1]. Lauer et al. [26] predicts that more than 97% of people who contract SARS-CoV-2 show symptoms within 11.5 days of exposure. So the early identification of real-world indicators is beneficial to contact tracing and local government policy recommendations. Baseline measures such as aggregated travel statistics take about a month to accurately predict the trend as shown in Fig. 20. In contrast, ITeM provides a shorter prediction window where real-world indicators show similar trends within three weeks, as shown in Fig. 21. The gain of one week to estimate the indicators based on higher-order analysis such as ITeM is significant and can improve policy planning.

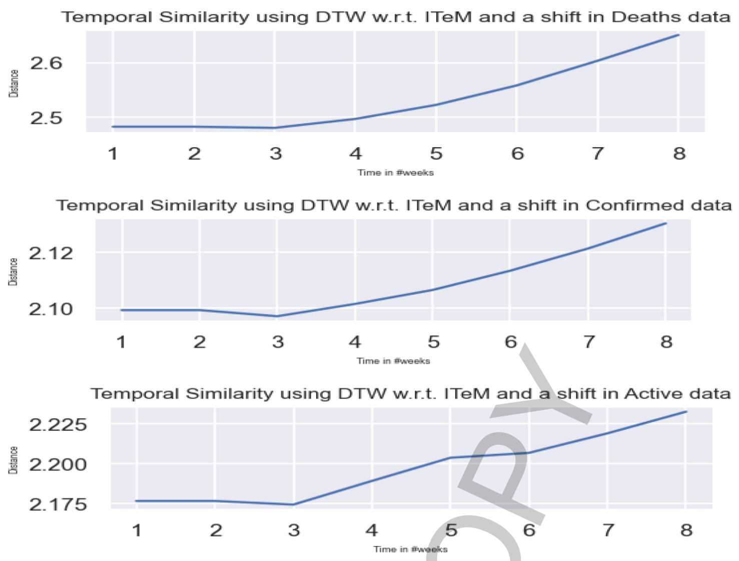


Fig. 21. ITeM-based travel trend follows COVID-19 real-world indicators with a shift of three weeks.

6. Conclusion and future work

Complex temporal networks are observed in the real world, and a better understanding of them is required to effectively handle real-world applications. We present Independent Temporal Motif (ITeM) as a building block to characterize temporal graphs. ITeM reveals many salient features of the temporal graph, such as its core structure, *fringe* vertices and edges, temporal evolution, and uniqueness. Graphs from different domains are found to exhibit varied structural and temporal distributions. Likewise, graphs from similar domains are found to exhibit similar structural properties, but many of them show varied temporal characteristics. We use these observations to characterize individual graphs and define a metric to quantitatively measure the similarity among them. We also present the *importance sampling* based approach to analyze a large graph as a sequence of smaller windows. We use this to show a change in the distribution that exhibits a behavioral shift in the way entities interact in a transactional graph, such as a social network.

The rate at which temporal motifs are formed can also be used to generate synthetic graphs that exhibit similar evolution as a given real-world graph, as shown in [41]. Additionally, these features can also be used in a diverse set of applications, such as approximate sub-graph matching, graph mining, and network embedding learning. We will compare ITeM to other temporal network embedding generation techniques to measure the benefits of ITeM over other approaches for use in such applications. Future work will also address scalability challenges by estimating the number of ITeMs using specialized algorithms for different motif classes and perform a sensitivity analysis of the sampling approach.

Acknowledgments

We thank the DARPA Modeling Adversarial Activity (MAA) program for funding this project. The associated PNNL project number is 69986. A portion of the research was performed using PNNL Institutional Computing (PIC) at Pacific Northwest National Laboratory. We also thank Patrick Mackey and Joseph Cottam for providing feedback and help setting up the experimentation.

References

[1] Centers for Disease Control and Prevention. https://www.cdc.gov/coronavirus/2019-ncov/symptoms-testing/symptoms.html. Accessed: April 2, 2021.

[2] B. Adhikari, Y. Zhang, S.E. Amiri, A. Bharadwaj and B.A. Prakash, Propagation-based temporal network summarization, *IEEE Transactions on Knowledge and Data Engineering* **30**(4) (2017), 729–742.

[3] D. Aparício, P. Ribeiro and F. Silva, Graphlet-orbit transitions (got): A fingerprint for temporal network comparison, *PloS One* **13**(10) (2018), e0205497.

[4] E. Ben-Naim, H. Frauenfelder and Z. Toroczkai, Complex networks, Vol. 650, Springer Science & Business Media, 2004.

[5] A.R. Benson, D.F. Gleich and J. Leskovec, Higher-order organization of complex networks, *Science* **353**(6295) (2016), 163–166.

[6] D.J. Berndt and J. Clifford, Using dynamic time warping to find patterns in time series, in: *KDD Workshop*, Seattle, WA, USA, Vol. 10, 1994, pp. 359–370.

[7] K.M. Borgwardt, H.-P. Kriegel and P. Wackersreuther, Pattern mining in frequent dynamic subgraphs, in: *Sixth International Conference on Data Mining (ICDM'06)*, IEEE, 2006, pp. 818–822.

[8] J. Cao, C. Ding and B. Shi, Motif-based functional backbone extraction of complex networks, Physica A: Statistical Mechanics and its Applications, 2019, 121123.

[9] J. Chen, W. Hsu, M.L. Lee and S.-K. Ng, Nemofinder: Dissecting genome-wide protein-protein interactions with meso-scale network motifs, in: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006, pp. 106–115.

[10] C.-C. Chu and H.H.-C. Iu, Complex networks theory for modern smart grid applications: A survey, *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* **7**(2) (2017), 177–191.

[11] J.A. Cottam, S. Purohit, P. Mackey and G. Chin, Multi-channel large network simulation including adversarial activity, in: *2018 IEEE International Conference on Big Data (Big Data)*, IEEE, 2018, pp. 3947–3950.

[12] L. Cuadra, S. Salcedo-Sanz, J. Del Ser, S. Jiménez-Fernández and Z. Geem, A critical review of robustness in power grids using complex networks concepts, *Energies* **8**(9) (2015), 9211–9265.

[13] A. Dave, A. Jindal, L.E. Li, R. Xin, J. Gonzalez and M. Zaharia, Graphframes: an integrated api for mixing graph and relational queries, in: *Proceedings of the Fourth International Workshop on Graph Data Management Experiences and Systems*, ACM, 2016, p. 2.

[14] M. De Domenico, A. Lima, P. Mougel and M. Musolesi, The anatomy of a scientific rumor, *Scientific Reports* **3** (2013), 2980.

[15] P. Holme and J. Saramäki, Temporal network theory, Vol. 2, Springer, 2019.

[16] C.J. Honey, R. Kötter, M. Breakspear and O. Sporns, Network structure of cerebral cortex shapes functional connectivity on multiple time scales, *Proceedings of the National Academy of Sciences* **104**(24) (2007), 10240–10245.

[17] Y. Hulovatyy, H. Chen and T. Milenković, Exploring the structure and function of temporal networks with dynamic graphlets, *Bioinformatics* **31**(12) (2015), i171–i180.

[18] R. Jin, S. McCallen and E. Almaas, Trend motif: A graph mining approach for analysis of dynamic complex networks, in: *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, IEEE, 2007, pp. 541–546.

[19] J. Klaise and S. Johnson, The origin of motif families in food webs, *Scientific Reports* **7**(1) (2017), 16197.

[20] J. Köster and S. Rahmann, Snakemake: A scalable bioinformatics workflow engine, *Bioinformatics* **28**(19) (2012), 2520–2522.

[21] L. Kovanen, M. Karsai, K. Kaski, J. Kertész and J. Saramäki, Temporal motifs in time-dependent networks, *Journal of Statistical Mechanics: Theory and Experiment* **2011**(11) (2011), P11005.

[22] R. Kumar, J. Novak and A. Tomkins, Structure and evolution of online social networks, in: *Link Mining: Models, Algorithms, and Applications*, Springer, 2010, pp. 337–357.

[23] S. Kumar, W.L. Hamilton, J. Leskovec and D. Jurafsky, Community interaction and conflict on the web, in: *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, International World Wide Web Conferences Steering Committee, 2018, pp. 933–943.

[24] S. Kumar, F. Spezzano, V. Subrahmanian and C. Faloutsos, Edge weight prediction in weighted signed networks, in: *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, IEEE, 2016, pp. 221–230.

[25] E. Kušen and M. Strembeck, An analysis of emotion-exchange motifs in multiplex networks during emergency events, *Applied Network Science* **4**(1) (2019), 8.

[26] S.A. Lauer, K.H. Grantz, Q. Bi, F.K. Jones, Q. Zheng, H.R. Meredith, A.S. Azman, N.G. Reich and J. Lessler, The incubation period of coronavirus disease 2019 (COVID-19) from publicly reported confirmed cases: Estimation and application, *Annals of Internal Medicine* **172**(9) (2020), 577–582.

[27] J. Leskovec, J. Kleinberg and C. Faloutsos, Graph evolution: Densification and shrinking diameters, *ACM Transactions on Knowledge Discovery from Data (TKDD)* **1**(1) (2007), 2.

[28] Y. Li, Z. Lou, Y. Shi and J. Han, Temporal motifs in heterogeneous information networks, in: *MLG Workshop@ KDD*,

2018.
[29] K. Liu, W.K. Cheung and J. Liu, Detecting stochastic temporal network motifs for human communication patterns analysis, in: *2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013)*, IEEE, 2013, pp. 533–540.
[30] P. Liu, A. Benson and M. Charikar, A sampling framework for counting temporal motifs, arXiv preprint arXiv:1810.00980, 2018.
[31] P. Liu, V. Guarrasi and A.E. Sariyuce, Temporal network motifs: Models, limitations, evaluation, IEEE Transactions on Knowledge and Data Engineering, 2021.
[32] Y. Liu, T. Safavi, A. Dighe and D. Koutra, Graph summarization methods and applications: A survey, *ACM Computing Surveys (CSUR)* **51**(3) (2018), 1–34.
[33] M. Luby, A simple parallel algorithm for the maximal independent set problem, *SIAM Journal on Computing* **15**(4) (1986), 1036–1053.
[34] S. Mangan, A. Zaslaver and U. Alon, The coherent feedforward loop serves as a sign-sensitive delay element in transcription networks, *Journal of Molecular Biology* **334**(2) (2003), 197–204.
[35] N. Masuda and R. Lambiotte, A guidance to temporal networks, World Scientific, 2016.
[36] O. Michail, An introduction to temporal graphs: An algorithmic perspective, *Internet Mathematics* **12**(4) (2016), 239–280.
[37] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii and U. Alon, Network motifs: Simple building blocks of complex networks, *Science* **298**(5594) (2002), 824–827.
[38] A.B. Owen, Monte carlo theory, methods and examples, Monte Carlo Theory, Methods and Examples, Art Owen, 2013.
[39] P. Panzarasa, T. Opsahl and K.M. Carley, Patterns and dynamics of users' behavior and interaction: Network analysis of an online community, *Journal of the Association for Information Science and Technology* **60**(5) (2009), 911–932.
[40] A. Paranjape, A.R. Benson and J. Leskovec, Motifs in temporal networks, in: *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, ACM, 2017, pp. 601–610.
[41] S. Purohit, L. Holder and G. Chin, Temporal graph generation based on a distribution of temporal motifs, in: *Proceedings of the 14th International Workshop on Mining and Learning with Graphs*, 2018.
[42] R.A. Rossi and N.K. Ahmed, The network data repository with interactive graph analytics and visualization, in: *AAAI*, 2015.
[43] S. Sarkar, H. Alvari and P. Shakarian, Understanding information flow in cascades using network motifs, arXiv preprint arXiv:1904.05161, 2019.
[44] F. Schreiber and H. Schwöbbermeyer, Frequency concepts and pattern detection for the analysis of motifs in networks, in: *Transactions on Computational Systems Biology III*, Springer, 2005, pp. 89–104.
[45] V. Sekara, A. Stopczynski and S. Lehmann, Fundamental structures of dynamic social networks, *Proceedings of the National Academy of Sciences* **113**(36) (2016), 9977–9982.
[46] N. Shah, D. Koutra, L. Jin, T. Zou, B. Gallagher and C. Faloutsos, On summarizing large-scale dynamic graphs, *IEEE Data Eng. Bull.* **40**(3) (2017), 75–88.
[47] N. Shah, D. Koutra, T. Zou, B. Gallagher and C. Faloutsos, Timecrunch: Interpretable dynamic graph summarization, in: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 1055–1064.
[48] F. Shelobolin, COVID-19 Dynamic Graph. https://github.com/fshelobolin/C19DynamicGraph, 2021. [Online; accessed 19-July-2021].
[49] X. Shen, X. Gong, X. Jiang, J. Yang, T. He and X. Hu, High-order organization of weighted microbial interaction network, in: *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, IEEE, 2018, pp. 206–209.
[50] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn and K. Borgwardt, Efficient graphlet kernels for large graph comparison, in: *Artificial Intelligence and Statistics*, 2009, pp. 488–495.
[51] F.W. Takes, W.A. Kosters, B. Witte and E.M. Heemskerk, Multiplex network motifs as building blocks of corporate networks, *Applied Network Science* **3**(1) (2018), 39.
[52] Z. Toroczkai, Complex networks, Science-Based Prediction, 2005, 94.
[53] A. Vazquez, R. Dobrin, D. Sergi, J.-P. Eckmann, Z. Oltvai and A.-L. Barabási, The topological relationship between the large-scale attributes and local interaction patterns of complex networks, *Proceedings of the National Academy of Sciences* **101**(52) (2004), 17940–17945.
[54] H. Yin, A.R. Benson, J. Leskovec and D.F. Gleich, Local higher-order graph clustering, in: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, 2017, pp. 555–564.
[55] A.B. Yoo, M.A. Jette and M. Grondona, Slurm: Simple linux utility for resource management, in: *Workshop on Job Scheduling Strategies for Parallel Processing*, Springer, 2003, pp. 44–60.