Information Sciences 546 (2021) 1097–1112

Contents lists available at ScienceDirect

Information Sciences

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

Analyzing a large and unobtainable relationship graph using a streaming activity graph

Alon Bartal *, Gilad Ravid

Dept. Industrial Engineering and Management, Ben Gurion University of the Negev, Beer Sheva 84105, Israel

ARTICLE INFO

Article history:
Received 11 August 2019
Received in revised form 21 September 2020
Accepted 25 September 2020
Available online 7 October 2020

Keywords:
Large relationship graph
Unobtainable graph representation
Streaming graph
Activity graph

ABSTRACT

The availability of large-scale data about interactions of social media users allows the study of complex human behavior. Graphs are typically employed to represent user interactions, but several algorithms become impractical for analyzing large graphs. Hence, it can be useful to analyze a small sub-graph instead in a practice known as graph sampling. However, if the graph is unobtainable, for example, due to privacy limitations, graph sampling is impossible. We introduce an innovative algorithm for representing a large unobtainable graph of user relationships such as Facebook friendships, using a streaming graph of user activity that can include, for example, wall posts on Facebook. We applied different methods of the proposed algorithm to two large datasets. The results show that averages and distribution statistics of nodes in a large, unobtainable relationship graph are well represented by a graph of about 20% of the size of the unobtainable graph. Finally, we apply the proposed algorithm to identify influencers in an unobtainable graph by analyzing a representative graph. We find that 63% to 76% of identified influencers in the representative graph act as influencers in the unobtainable graph, suggesting that the developed algorithm can effectively capture properties of the unobtainable graph.

1. Introduction

Online social networks (OSNs) are increasingly popular, allowing billions of users to form relationships that can be represented by a *relationship graph* such as the Facebook friendship graph. Some OSNs enable users to leverage their relationships by interactions such as content sharing [5], which are typically represented by an *activity graph*.

The activity graph provides a good approximation of the strength of the users' relationships [26], but differs substantially from the relationship graph [47]. For example, the ties in a Twitter activity graph, representing tweeting interactions of a user with others, can substantially differ from the size of her Follower/Following lists.

The relationship and activity graphs affect the evolution of each other. For example, the structure of the Facebook relationship graph affects the user's tendency to share a photo [18]. An edge in the activity graph expresses this sharing activity. In turn, the activity graph can affect the evolution of the relationship graph. For example, a Twitter user who posts interesting content attracts others to follow her by creating a link in the relationship graph. Analysis of the activity graph might provide clues about the structure of the relationship graph.

* Corresponding author.
E-mail address: bartala@post.bgu.ac.il (A. Bartal).

https://doi.org/10.1016/j.ins.2020.09.063
0020-0255/

Analyzing a large graph is computationally complex [12]. Graph sampling methods make such analyses easier by analyzing a small sample of the original graph instead. A good sampling algorithm preserves many properties of the original graph [12]. However, utilizing sampling approaches assume that access to the original graph is possible. Methods for dealing with missing data might help study partially obtainable graphs but they are inapplicable when all (or most) of the data are missing [22].

We aim to study the properties of an unobtainable relationship graph (i.e., the original graph, G). Since the activity graph contains information about the structure of G [26], it can be used to study G. Hence, we aim to find a representative graph that would best match a set of G's properties that can be used to study the unobtainable graph G.

The proposed algorithm treats the activity graph as a stream of edges that are processed as they arrive to generate a representative graph. We test the performance of the representative graphs for the Facebook, and Twitter datasets in terms of preserving the averages and distribution statistics of nodes in G. The results show that a representative graph of about 20% of the size of G accurately represents the statistics of G. We demonstrate the applicability of our algorithm by identifying influencers in a representative graph. We find that 63% to 76% of the identified influencers act as such in the original graph G.

Contrary to sampling algorithms, which must access the original graph, or missing data techniques that cannot handle large amounts of missing data, our approach is innovative in several ways: First, we demonstrate how to represent a large, unobtainable relationship graph using a smaller activity graph. Second, we estimate the properties of the original graph and evaluate the performance of the representative graphs in capturing those properties. Third, we demonstrate an application for identifying influencers in an unobtainable graph by analyzing a representative graph.

Revealing the properties of an unobtainable graph can be used to analyze hidden networks such as those of terrorists, networks with extensive user privacy limitations, or networks that take a long time to collect.

The next section sets the theoretical background. Section 3 develops an algorithm and methods for generating a representative graph. Section 4 describes the datasets. The results are detailed in Section 5. Lastly, we discuss the contributions, innovations, and limitations of this work in Section 6.

2. Related work

OSNs have revolutionized how people create and consume information. Currently, millions of users maintain online relationships that can be represented as a *relationship graph*. Examples include friendships on Facebook or Follower-Followee relationships on Twitter.

Some OSNs allow users to leverage their relationships through interactive activities. Twitter, for example, supports posting messages (tweets), replying to tweets, mentioning, and content sharing (retweeting). Online activities can be represented as an *activity graph* where nodes are active users and edges represent interactions between users.

The activity graph is a good approximation of users' relationships [45,26]. However, the activity graph differs substantially from the relationship graph [47]. The relationship graph typically contains more nodes than the activity graph because it is difficult to maintain interactions with many people [36].

The relationship and activity graphs affect the evolution of each other. For example, the structure of the relationship graph largely determines the information diffusion paths [8]. A user who is exposed to content from her friends is more likely to re-share it [18], thus creating a new edge in the activity graph. On the other hand, user interactions expressed as online activities affect the evolution of the relationship graph. The public nature of online activities (e.g., postings) in many OSNs [36] allow user A to follow interesting content posted by user B, thus, creating a social link in the relationship graph. Studying user interactions through the activity graph can provide insights about the structure, properties, and evolution of an unobtainable relationship graph.

2.1. Graph evolution models

Both graph sampling methods and network evolution models try to explain the properties of a graph [25]. Leskovec et al. [32] presented a graph sampling method, assuming that the sampled graph was similar to what the original graph looked like when it was the size of the sampled graph. Network evolution models can simulate a graph with a given size that matches the evolving nature of the original network [25]. We can then analyze the small graph.

The activity graph provides a good approximation of user relationships [26]. Therefore, applying a graph evolution model to the activity graph to make it more similar to the relationship graph seems to be an attractive, intuitive idea to create a representative graph.

Three properties are particularly important in representing the topology of a graph: Degree, Clustering coefficient (CC), and Path-length (PL) [25]. Degree centrality helps us understand the connectivity of graphs. The CC captures local topological features of the graph (transitivity) [46]. The PL presents the trail sequence of distinct edges between two nodes, capturing global topological features of the graph [2,25].

The Small-world Model [46] provides a good representation of the CC but does not maintain the power-law degree distribution that exists in many real-world graphs. The Barabasi and Albert [6] Model (BA model) generates a power-law degree distribution graph, but assumes an undirected graph, ignoring a crucial feature of networks. [27] extended the BA Model to directed graphs, generating Indegree and Outdegree power-laws distributions. Seeking to account for the dynamic nature of

networks, [34] introduced a model with evolving edges, where an edge grows by a linear function of time, and a deletion follows a Poisson process.

[20] extended the network evolution further with a Poisson process, proposing three models based on the homogeneous Poisson, non-homogeneous Poisson, and birth-death process. However, the two latter models are unable to preserve Degree, CC, and PL simultaneously in the same graph.

Kronecker graphs [33] reproduced both densification and small diameter, while preserving the graph properties of heavy-tailed Indegree and Outdegree distributions, eigenvalues, and eigenvectors. However, the CC of the generated graphs are smaller than real-world graphs, and the deterministic nature of the model misses the dynamic features of real-world graphs [7]. Thus, the Stochastic Kronecker Graphs Model [41] was developed. The model adjusts the parameters that strongly affect the properties of the generated graphs [41]. Leverage Lu Model [13].

In the Chung Lu Model, the probability of an edge formation between two nodes is proportional to the product of their degrees. The model preserves degree distribution, but misses other graph properties such as the CC. The Block Two-Level Erdos and Renyi (BTER) Model [30] extended the Chung Lu Model and matches both the degree distribution and CC. One deficit of the BTER is that it does not preserve community structure parameters such as community densities and membership that are retained in the Generalized BTER Model [11]. However, the Generalized BTER Model misses the CC per degree [19]. Two other important models are the Forest Fire Model [12] and the Random-Surfer Model [10]. We discuss them further in Section 2.2.

To summarize, network evolution models can simulate a graph that preserves only a few of the properties of the original graph. Contrary to network evolution models, graph-sampling methods are more suitable for capturing structures that are similar to the original graph [25], as reviewed next.

2.2. Sampling from static graphs

Graph sampling is the task of selecting a set of nodes and edges from an original graph G, while preserving specific properties of G in the sampled graph G_s [2,25]. Then, we can analyze G_s instead of G.

Two main graph statistics are node-level, and graph-level [2]. A node-level statistic describes a characteristic of a node such as Degree. A graph-level statistic describes a characteristic of the entire graph such as its diameter. Another distinction is made between point and distribution statistics [2]. A point statistic is a single value such as an average PL, and a distribution statistic is a vector of statistics (e.g., PL distribution for all nodes).

Recent graph sampling studies [25,2] used Degree, CC, and PL measures to test the extent to which G_s preserves the properties of G. Sampling methods can be categorized into three main groups: Edge Sampling, Node Sampling, and Traversal Based Sampling.

Edge sampling (ES) selects a subset of edges from the original graph and then selects the nodes that are connected to those edges [2]. ES does not preserve many graph properties such as the CC and connectivity [3], due to the independent sampling of edges. However, ES is expected to preserve the PL of the original graph due to its bias towards high-degree nodes [31].

Node sampling (NS) selects a subset of nodes according to a distribution, neglecting the structure of the original graph. Then, only edges between the sampled nodes are kept [31]. For example, random degree node-sampling samples a node proportional to its degree [3]. NS does not preserve power-law degree distributions [44] or the original level of connectivity [31].

NS with Neighborhood expands NS by first selecting a subset of nodes without topology information, and then keeping all edges of the selected nodes with their neighbors [49]. It represents the original graph better than ES and NS but like NS and ES, it assumes that access to the original graph is feasible. In reality, accessing a large relationship graph in many OSNs, such as Facebook or Twitter is often limited by API calls or by user privacy regulations. Traversal Based Sampling algorithms are more suitable for this task [2].

Traversal Based Sampling (TBS) algorithms assume partial access to the original graph [2]. Examples of TBS algorithms are Breadth-First Sampling [29], Depth-First Sampling [29], Snowball Sampling [48], and Random-First Sampling [17]. Breadth-First Sampling starts with a root node and explores its neighboring nodes first, before moving to the next level neighbors. Breadth/ Depth/ Random-First Sampling produces samples biased towards high-degree nodes [31], and they differ in their implementation. Breadth/ Depth/ Random-First Sampling selects the first/ last/ random nodes respectively [17].

Snowball Sampling is another TBS algorithm that starts by selecting a set of nodes. Then for each selected node, k unvisited neighbors are added randomly to the node-set [48]. Snowball Sampling preserves the connectivity of the original graph. However, it is biased because the nodes sampled on the last round miss many neighbors [31].

Forest Fire (FF) [32] is a probabilistic version of Snowball Sampling. FF starts by sampling a node uniformly at random. Then, it "burns" a random number of its outgoing edges that is geometrically distributed with mean $p_f/(1-p_f)$. These edges are added to the sampled node-set along with their neighbors. The process is repeated for each visited node until no new node is selected. Then, a new node is randomly chosen to continue the process until reaching the desired sample size. The FF algorithm well represents densification law, shrinking diameter, and community guided attachment [32], but misses high-degree nodes [2].

The Random Walk algorithm is related to Snowball Sampling [37]. Random Walk starts with a selected node and at each iteration, the node and all of its neighbors are discovered. Then, one of the neighbors of the visited node is selected uniformly at random (or according to some weight) to be visited [37]. The PageRank algorithm [39] illustrates a random walk of various lengths on the web graph. The number of random walks is determined by a positive real value α, where $\alpha \in [0, 1)$. Snowball Sampling becomes Random Walk for $k = 1$. Random Walk is memoryless whereas, in Snowball Sampling, selected nodes are excluded from future steps. Thus, Random Walk obtains a stationary distribution of a Markov Chain analysis [37]. Random Walk is biased since high-degree nodes are more likely to be sampled, resulting in a uniform distribution of edges [37].

The transition probability between nodes in the Markov Chain Model can be changed to output uniform NS in undirected graphs, preserving the node degree distribution of the original graph, as the Metropolis-Hastings Random Walk algorithm does [28]. Calculating the transition probabilities requires knowledge of the adjacent nodes' degree distribution, which is hard to obtain. The Metropolis-Hastings Random Walk algorithm was expanded to sample from directed graphs by considering unidirectional edges as bidirectional edges. Frontier Sampling [40] is an edge sampling algorithm that was also designed to preserve the degree distribution of the original graph by obtaining a sample graph with the least mean square error compared with the original graph.

The Random Walk with Jumps algorithm [32] avoids the Random Walk's tendency to be trapped in dense regions of the graph by allowing a user to jump to a random node when trapped [40]. The Multi-Dimensional Random Walk algorithm [40] also addressed this trapping problem by conducting a node sampling (NS) to obtain an initial set of nodes. Then, at each step, the algorithm chooses a node v with a probability proportional to the node's degree, chooses a random neighbor w of v, and collects edges between v and w.

TBS algorithms can be used with partial access to the original graph by accessing only a node's neighborhood. Various techniques have been developed to handle partially available data [22] as discussed next.

Partially available data. [23] estimated the parameters of an original graph by using Exponential Random Graph Models and then utilized the estimated parameters to simulate missing ties. [21] used observational data to infer the edges of the graph. [24] introduced small biases in the of parameter estimates of up to 20% of missing users. [22] analyzed dynamic graphs with missing data in multiple periods but considered only users observed at the start and the end of the analyzed period.

The Inclusive Sampling Method [22] has been proposed for dealing with missing data in dynamic graphs with multiple periods. This method involves forming subgroups of the data for each period where each subgroup includes only members that are fully observed at the start and end of the period.

The reviewed graph-sampling algorithms in this section assume a static graph structure. This assumption is inappropriate for large and dynamic graphs. Moreover, these algorithms ignore the time-evolving streaming nature of OSNs. Therefore, an algorithm for sampling from a streaming graph, which assumes that the graph can be accessed only as a stream of edges, is preferable when the graph is dynamic [50].

2.3. Sampling from dynamic graphs

Streaming graphs are continuous and dynamic over time, and their structure is not fully observable [2]. A sampling algorithm from a streaming graph produces a sampled sub-graph by sequentially sampling edges or nodes of the streaming graph. [1] proposed an algorithm that uniformly samples a subset of edges from a streaming graph. Recent studies [35,42,43] focus on estimating transitivity and local triangle counts in graph streams. Streaming graph sampling methods typically estimate the temporal distribution of Degree, CC, and PL [2].

All of the discussed methods so far assume that the sampled graph has: 1) full access, 2) restricted access by crawling (e.g., by the Random Walk), or 3) streaming access when the graph is dynamic. In the last case, we cannot explore the neighbors of the nodes. Thus, the graph is accessed only sequentially as a stream of edges.

Directly sampling from the original graph is often impractical due to graph size, API or privacy limitations, or hidden relationships. As a result, the original graph is essentially unobtainable. Methods for dealing with missing data are also inapplicable since all or most of the data are missing [24].

This research assumes that access to the original relationship graph is not possible. However, public activities of users are obtainable. We propose an algorithm that utilizes a smaller, dynamic, streaming activity graph to generate a representative graph of the large relationship graph (G). The proposed algorithm preserves the properties of the latter graph and allows us to study it.

In this study we ask two research questions (RQ):

1. Can an activity graph represent a relationship graph?
   The activity graph is a stream of edges, which represents users' interactions over time. The larger the time window is, the larger the activity graph. Thus, a natural question is:
2. How small can the activity graph be to represent the relationship graph?

To answer these research questions, we created an algorithm to represent an unobtainable relationship graph based on the activity graph.

3. Algorithm for inferring the properties of an unobtainable graph

The developed algorithm (*RAG*) generates a Representative Activity Graph $G_R(V_R, E_R, w_R)$ of the unobtainable relationship graph $G(V, E)$.

Since G is unobtainable, *RAG* uses the observed edge stream of the activity graph within a predefined time window to generate G_R. In G_R: (i) nodes represent users ($V_R \subseteq V$), (ii) directed edges arrive in a streaming fashion in any arbitrary order [2], representing interactions between users, and (iii) the weight of the edges denote the frequency of interactions between a pair of users within a defined time interval.

RAG (Algorithm 1) involves three steps: 1) select all streaming edges of the activity graph within a predefined time interval, 2) select nodes that are attached to the selected edges from Step #1, and 3) add new nodes from Step #2 to G_R, add all streaming edges to E_R, and update the weights of the edges (w_R). These steps are repeated until the desired number of nodes (n) is collected.

Algorithm 1 Representative Activity Graph (*RAG*).

Input: Desired number of nodes n, Activity graph stream G_{AS}, Window size Δ_t

initialize $V_R \leftarrow \phi, E_R \leftarrow \phi, w_R \leftarrow 0$
while $|V_R| \leq n$ **do**
  $E_t \leftarrow$ observed edges in G_{AS} within Δ_t
  for each $e_{ij} = (v_i, v_j) \in E_t$ **do**
    if $v_i \notin V_R$ **then**
      $V_R \leftarrow V_R \cup \{v_i\}$
    if $v_j \notin V_R$ **then**
      $V_R \leftarrow V_R \cup \{v_j\}$
    $w_{R,ij} \leftarrow w_{R,ij} + w(e_{ij})$
    $E_R \leftarrow E_R \cup \{e_{ij}\}$
    if $|V_R| \geq n$ **then break**
  move the window forward by Δ_t
return $G_R(V_R, E_R, w_R)$

RAG captures infrequent user interaction activity and increases the connectivity of the collected nodes by considering their cumulative interaction over time. Given that very active nodes appear more frequently in the activity graph, G_R might be biased. *RAG* counters this bias by selecting all nodes within a time interval, including less active nodes. *RAG* also selects all streaming edges and aggregates them in G_R. This process resembles edge sampling (ES) if one thinks of the streaming edges as a sample of edges from the original graph (G). Similar to ES, *RAG* is expected to preserve the PL of G but not the CC.

Recall that sampling algorithms sample nodes or edges directly from G to generate a sub-graph (G_s) of G. Since RAG cannot access G, it generates a representative graph of G based on user activity.

Sampling algorithms and RAG aim to preserve the properties of G. However, by generating smaller graphs than G they create graphs with biased properties compared to G. RAG generates a representative graph (G_R) that is smaller and differs substantially from G [47]. Hence, we expect that the properties of G_R will be biased compared to G.

3.1. Algorithm methods

We developed several methods based on the RAG algorithm, with the goal of better representing the properties of G. We first generate G_R by Algorithm 1 and then apply the methods proposed in this section to G_R.

Since the representative graph (G_R) is smaller than G, there is a scaling effect on some of its statistics [32]. Our goal is to compensate G_R for the lack of knowledge about user relationships in G. We follow a similar approach to [32] by assuming that G looked similar to G_R at some previous point. Seeking to mimic the temporal evolution of G, we apply theoretical network evolution principles (Section 2.1) to G_R. Hence, compensation is achieved by the addition of edges to G_R, aiming to scale up its properties to resemble those of G.

We assume that G does not evolve during data collection or during analysis. However, the activity graph (the input to Algorithm 1) does evolve.

Typically, user relationships in real networks are not independent events [16]. Assuming mutuality, if $e_{ij} \in E_R$, then $e_{ji} \in E_R$. However, a single interaction might not imply a relationship, as defined in the First method (*RAG*1).

- **RAG1**: if $e_{ij} \in E_R$ and $w_{R,ij} > 1$, then add e_{ji} to E_R with $w_{R,ji} = 1$.

Adding multiple edges to G_R might sharply reduce the PL. We expect *RAG*1 to do well representing the mutuality in G. However, it is likely to miss the CC, since triangles are not the focus of the *RAG*1 method.

The Small-world Model preserves the CC but does not maintain the power-law degree distribution. The BA Model generates a scale-free degree distribution but does not possess a high CC. To scale up the properties of G_R, the next method (RAG2) aims to maintain both the power-law degree distribution and high CC. The CC is tunable by applying *RAG*2 to G_R with different parameters.

- **RAG2** involves three steps: (i) select a graph evolution model M_i (Table 1), ii) sample a set of nodes of size S from G_R, according to a probability $P_{k,j}$ (Table 1iii) for each node from Step #2, connect its neighbors according to M_i. Add new edges between unconnected neighbors to G_R.

RAG2 has three tunable parameters: $M_i, P_{k,j}$, and S. The first two parameters create 18 combinations: three graph evolution models (M_i) times six node sampling probabilities ($P_{k,j}$). In addition, the size of S changes. These parameters control for the CC and graph evolution. However, they might over-represent nodes with high Indegree, or low CC, depending on the selected parameters.

Each method creates a new representative graph by adding edges to G_R.

3.2. Algorithm complexity

Let $|V_R|$ be the number of nodes, and $|E_R|$ be the number of edges in G_R. Step #1, line 2 in *RAG* (Algorithm 1), is repeated $k \leqslant |V_R|$ iterations until the desired number of nodes is collected, assuming that G_R contains at least the desired number of nodes. In each iterations (time steps $t_1, t_2, \ldots, t_k$), Algorithm 1 processes $|E_t|$ edges (Steps #2 and #3 starting at line 4 in Algorithm 1). Hence, the time complexity of *RAG* is $O(|E_{t_1}| + \ldots + |E_{t_k}|) = O(|E_R|)$.

RAG1 begins by generating G_R using *RAG* with $O(|E_R|)$ time complexity. Then, it adds opposite edges to existing edges in G_R having weights larger than 1 with $O(|E_R|)$ time complexity. Thus, the time complexity of *RAG*1 is $O(|E_R|)$.

RAG2 begins by generating G_R using RAG with $O(|E_R|)$ time complexity. Then, a set of S nodes is sampled from G_R and their neighbors are connected by an edge according to a graph evolution model, having the highest time complexity for the CC algorithm of $O(|V_R| * d^2)$ where d is the average node degree [14]. Since d is defined by dividing the sum of all nodes' degrees by the total number of nodes, $d^2 = O((|E_R|/|V_R|)^2)$. We can write $O(|V_R| * d^2) = O(|E_R|^2/|V_R|)$. Since $|E_R| \leqslant |V_R|^2, O(|V_R|^4/|V_R|) = O(|V_R|^3)$. Thus, the time complexity of *RAG*2 is $O(|V_R|^3 + |E_R|)$. Since $E_R \leqslant V_R$, the time complexity is $O(|V_R|^3)$.

Table 1
Tunable parameters of RAG2.

Graph evolution models (M_i)	
M_{Br}	A Bernoulli process with a success probability P_B^*
M_{FF}	Forest Fire Model with a forward burning probability P_f^*
M_{BA}	Barabasi and Albert [6] Model with α^* - the power of the preferential attachment Every new node is born with a single edge
Node sampling probability ($p_{k,j}$)	
with ($j = 1$) or without ($j = 0$) replacement; $k \in \{Deg, PR, CC\}$	
$P_{Deg,j}$	Nodes with high Indegree are more likely to be sampled
$P_{PR,j}$	Nodes with high PageRank [39] are more likely to be sampled
$P_{CC,j}$	Nodes with low CC are more likely to be sampled
Sample size of the node-set	
S^*	Number of sampled nodes from G_R according to a probability $p_{k,j}$

** Different values were tested (see Section 5.4).

Space complexity for all suggested algorithms requires locating G_R in memory, meaning, $O(|V_R| + |E_R|)$.

Blagus et al., [9] who empirically compared seven graph sampling methods, found that random node sampling by degree and BFS achieved the best results in preserving a property of the original graph with $O(|V|)$ time complexity. However, these methods do not preserve several graph properties simultaneously. On the other hand, the FF algorithm [32] preserves many graph properties (Section 2.2) with $O(|V|^3)$ time complexity simultaneously. When the original graph is unobtainable, graph sampling is impossible. Our proposed algorithm can preserve properties of an unobtainable graph (elaborated in Section 5) with $O(|V|^3)$ time complexity simultaneously.

Next, we elaborate on the evaluation methodology that measures the extent to which the representative graphs represent G.

3.3. Evaluation method

We follow four steps to evaluate the accuracy of each representative graph.

1. Define a set of graph statistics. Following similar studies [32,12,2] that represent a large graph by a smaller graph, we consider: Degree, PL, and CC which are evaluated as point and distribution statistics.
   We do not limit the discussion or the evaluation of the statistic to a specific distribution such as a power-law degree distribution. Our goal is to find the best algorithm that generates a representative graph that gives the most similar statistics to G.
2. Generate a representative graph of G by each method (Section 3.1) and RAG (Algorithm 1).
3. Compute the statistics from Step #1 for each representative graph.
4. Evaluate the accuracy of the statistics' estimations of each representative graph by comparing them to the true statistics of G.

For evaluation, in Step #4 only, we assume that the relationship graph G is obtainable. In Steps #1 to #3, G is unobtainable.

To compare the distribution statistics of a representative graph with that of G, we use the Kolmogorov-Smirnov (KS) D-statistic test like [32]. The D-statistic is defined as the maximum distance: $D = max(|F_1(x) - F_2(x)|)$, where x represents the range of the random variable, and F_1 and F_2 represent the cumulative distribution functions of G, and the representative graph respectively. The smaller the distance, the more similar the distribution curves, indicating the better the representation of the statistic. Hence, we seek to minimize the distance between the two distributions (F_1 and F_2).

4. Data description

The First Dataset (DS1) includes two Twitter (TW) graphs [15].

1. **Activity graph (DS1.1)** contains 985,590 tweets: retweets, mentions, or replies regarding the Higgs boson particle with at least one of the keywords or hashtags "LHC", "cern", "boson", or "Higgs". The interactions were sent between 00:00 AM, July 1, 2012 and 11:59 PM, July 7, 2012. An edge e_{ij} indicates that user v_i posted a tweet addressed to v_j.
2. **Relationships graph (DS1.2)** consists of nodes representing the authors of the tweets in the activity graph (DS1.1), and edges representing the Following relationships. In a directed edge e_{ij} node v_i Follows node v_j.

The Second Dataset (DS2) includes two Facebook (FB) graphs [45].

1. **Activity graph (DS2.1)** contains 838,092 wall posts of FB users who appear in the FB relationship graph (DS2.2). A directed edge e_{ij} indicates that user v_i posted on the wall of v_j. The wall activity contains interactions between September 14, 2004, and January 22, 2009. Each post contains information about the wall owner, the posting user, and the posting time.
2. **Relationship graph (G) (DS2.2)** was collected between December 29, 2008, and January 3, 2009, representing the New Orleans regional static, and directed friendship graph on Facebook. A directed edge represents a friendship. Note, on Facebook, any friendship imposes bi-directional edges.

Table 2
Characteristics of the relationship (G) and the activity graphs.

Graph Metric	Twitter graphs (DS1)		Facebook graphs (DS2)	
	Relationship (G)	Activity	Relationship (G)	Activity
Nodes	456,626	456,626	63,731	46,952
Edges	14,855,842	985,590	1,545,686	838,092
Window size (Δ_t)	–	1-hr	–	1-month
OBS	–	168	–	52

OBS: number of observations (time steps) of the activity graph.

Table 2 summarizes the two datasets.

Next, we generate representative graphs of G in both datasets by using the RAG algorithm and the proposed methods, and evaluate their performance.

5. Analysis and results

Before generating representative graphs, we preprocess the two datasets and explore their dynamics over time.

To test if a single observation of the activity graph is sufficient to represent the properties of the relationship graph, we split the activity graphs of both datasets by time intervals (Δ_t in Table 2), creating graph observations that will be used later in Algorithm 1.

For DS1.1, we split the Twitter interactions by 1-hr intervals, resulting in 168 observations of the activity graph. A time interval of 1-hr was selected because the decay time scale of user activity was ~1.13 h [15]. Thus, $\Delta_t = 1_{hr}$ is sufficient to capture the temporal activity in the graph.

For DS2.1, we split the Facebook interactions by 1-month intervals, following [45]. This process resulted in 52 monthly observations of the Facebook activity graph.

Fig. 1 presents the monthly and hourly average point statistics for each temporal observation of the activity graph of the Facebook and Twitter datasets respectively, versus the average point statistics of their relationship graphs (G). Since the Indegree and Outdegree statistics were highly correlated, hereafter, we report only the Indegree statistics. Fig. 1 shows that the temporal observations of the activity graphs do not preserve the average statistics of G. Users are infrequently active, affecting the statistics of the activity graphs.

To better represent the properties of G, we create the G_R representative graph for each dataset by using *RAG* (Algorithm 1). *RAG* collects the edges of the streaming activity graph until the number of nodes in G_R (the size of G_R) equals a predefined fraction of the of nodes in G (the size of G). Previous studies [32,2] tested sampled graphs that were 10% to 40% of the size of the original graph. We test larger fractions, from 10% to 60% of the size of G.

To verify whether G_R differs significantly from G, we compare their structure by using a quadratic assignment procedure (QAP) test [16], one for each dataset, with 1,000 replications. QAP is a non-parametric, restricted permutation test for the significance of an association between two matrices with complex dependencies. QAP enables us to evaluate the extent to which the graphs are correlated in terms of their structure, and whether that correlation is significant.

QAP requires the compared graphs to be the same size. Therefore, using Algorithm 1 we first generate G_R at the earliest time step where its size (n) is 10% of the size of G. Then, we create a sub-graph of G, containing nodes that appear in G_R, and edges that connect those nodes in G. Both G_R, and the sub-graph of G have the same node-set but can have different edges. We perform six QAP tests beginning with 10% of the size of G and increasing to 60% in 10% increments.

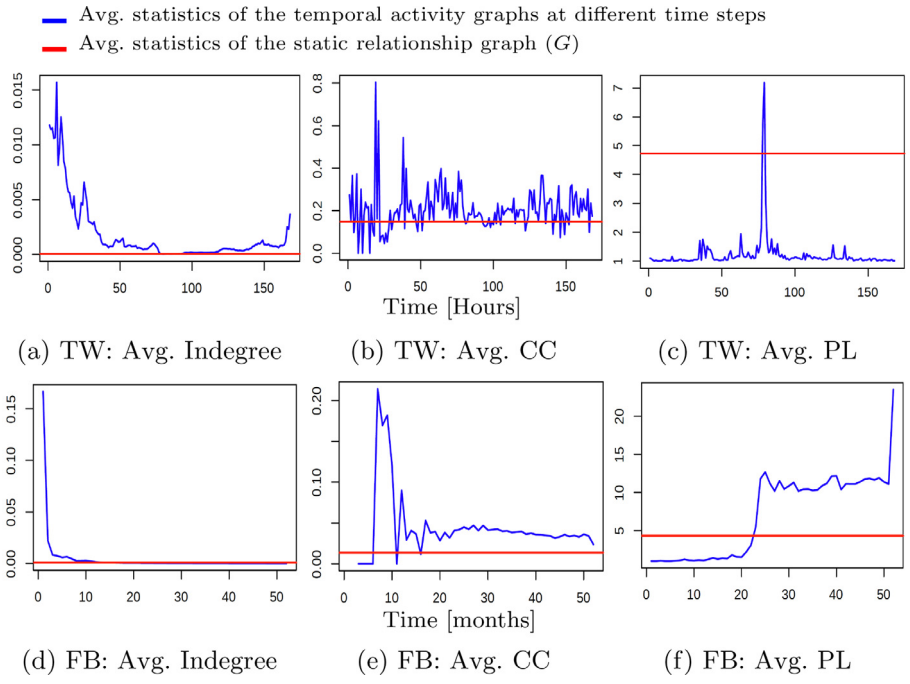


Fig. 1. Point statistics of the activity graphs vs. the relationship graph.

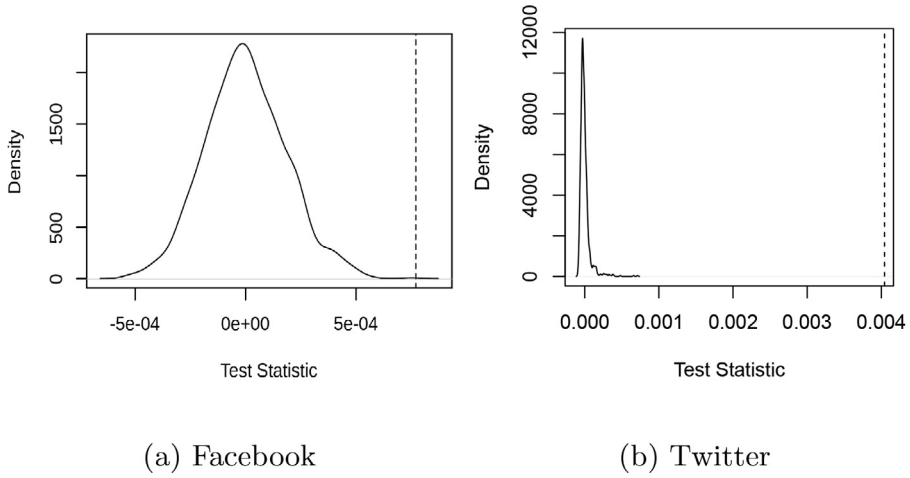


Fig. 2. Estimated density of the QAP replications.

Fig. 2a and b present the QAP test results at 10% of the size of G for Facebook and Twitter, respectively. Both graphs (G_R, and the sub-graph) in both datasets differ significantly with a low correlation of 7×10^{-4} on Facebook, and 4×10^{-3} on Twitter, and $P_{value} \approx 0$. The results are similar for graph sizes 10% to 60% but are not presented for brevity.

In accordance with the evaluation steps (Section 3.3), in Step #1 we use three statistics: the Degree, CC, and PL. In Step #2, we generate representative graphs (with sizes from 10% to 60% of the size of G) in each dataset by using the RAG algorithm and the proposed methods (Section 3.1). In Step #3, we compute the point and distribution statistics of each representative graph. Finally, in Step #4, we compare the statistics of the representative graphs to the original relationship graphs (Facebook and Twitter), as elaborated next.

5.1. Comparing point statistics

Fig. 3 depicts the average point statistics of the representative graphs for each dataset generated by RAG (Algorithm 1), and by the two methods (Section 3.1) versus the average point statistics of the original graph (G).

RAG2 provides 18 potential combinations of $P_{k,j}$ and M_i (Table 1). We present only the best performing parameters of the methods, as denoted next.

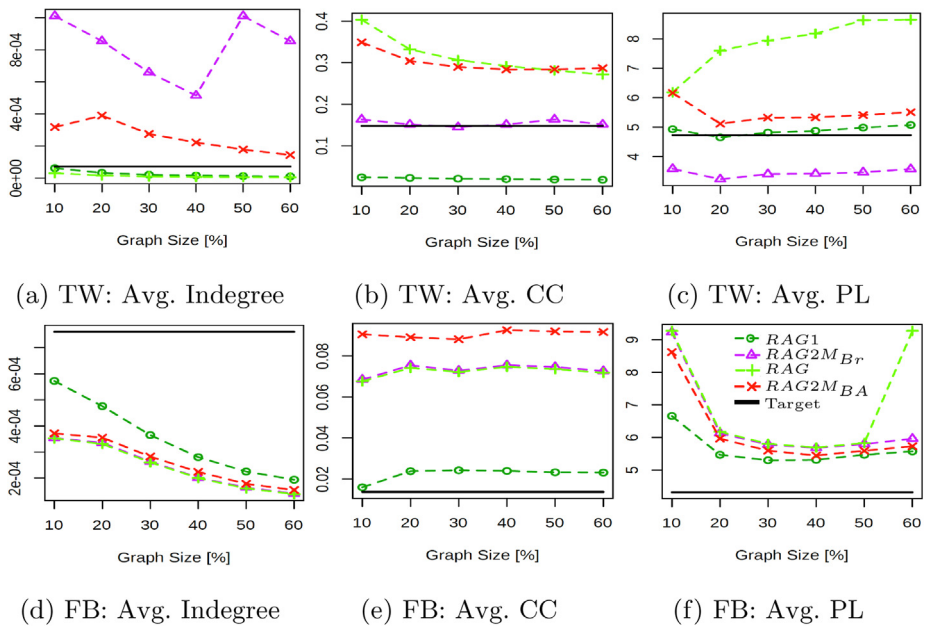


Fig. 3. Point statistics of G and the representative graphs at various sizes.

Table 3
A summary of the evaluations of the point statistics.

Model	Avg. Indegree		Avg. CC		Avg. PL	
	TW	FB	TW	FB	TW	FB
RAG	$-^*$	–	+	+	+	+
*RAG*1	$-^*$	–	–	$+^*$	$+^*$	+
$RAG2M_{Br}$	+	–	$+^*$	+	–	+
$RAG2M_{BA}$	+	–	+	+	+	+

(+) Overestimation; (−) Underestimation; (∗) Close estimation.

- $RAG2M_{Br}$: RAG2(M_{Br} with $P_B = 0.05, S = 0.02, P_{Deg,0}$).
- $RAG2M_{BA}$: RAG2(M_{BA} with $\alpha = 1, S = 0.02, P_{Deg,1}$).
- *RAG*: generates the G_R graph using Algorithm 1.
- *RAG*1: if $e_{ij} \in E_R$ and $w_{R,ij} > 1$, then add e_{ji} to E_R with $w_{R,ji} = 1$.
- Target: the original relationship graph *G*.

Table 3 summarizes the observations regarding the over/under estimation of the target statistics based on Fig. 3. Estimations that reach the target statistic in at least one evaluation (10% to 60%) are denoted by '∗' in Table 3.

Average normalized Indegree Fig. (3a and 3d). Since most edges in *G* are not observed in the activity graph, *RAG* and *RAG*1 miss these edges, producing lower estimations of the average Indegree.

$RAG2M_{Br}$ and $RAG2M_{BA}$ overestimate the average Indegree of *G* on Twitter but underestimate it on Facebook. This finding can be attributed to the different definitions of community in both datasets.

The Twitter dataset represents interactions among users who form a community based on shared topics of interest (Higgs bosson) [15]. In contrast, the Facebook dataset represents interactions among users who actively chose to affiliate with a regionally based community. Facebook users can invite friends to join a community, which increases the connectivity of a community [4]. Thus, users who interact in DS2 (Facebook) are more likely to maintain social relationships than on DS1 (Twitter). It is more likely that $RAG2M_{Br}$ and $RAG2M_{BA}$ will underestimate the average Indegree on Facebook than on Twitter.

Average CC (Fig. 3b and e). *RAG* and $RAG2M_{BA}$ produce more clustered graphs on average than *G* in both datasets.

*RAG*1 produces close estimations on Facebook by capturing the bidirectional nature of Facebook edges. However, it underestimates the average CC on Twitter by adding opposite edges that reduce the CC of the representative graph.

$RAG2M_{Br}$ overestimates the average CC of *G* on Facebook but produces close estimations on Twitter.

Both $RAG2M_{BA}$ and $RAG2M_{Br}$ connect neighbors of high-degree nodes, creating densely connected areas in the graph.

Average PL (Fig. 3c and f). On Facebook, all algorithms overestimate the average PL due to a lack of node connectivity in the representative graphs compared to *G*. The estimations are improved as the size of the representative graphs grows to 40% of the size of *G*. After 40% the estimations deteriorate, as more nodes are introduced to the graph with insufficient connectivity.

On Twitter, *RAG* misses the average PL, and *RAG*1 slightly overestimates it. $RAG2M_{Br}$ underestimates the average PL. $RAG2M_{BA}$ overestimates the average PL, with close estimations at 20% of the size of *G*.

Overall, most of the graphs generated by the algorithm methods improve the performance of *RAG* and present consistent estimation trends (Table 3).

Although point statistics are essential in evaluating representative graphs, they do not provide insights into the specific parameters of individual nodes. Therefore, in the next section, we discuss the distribution statistics.

5.2. Comparing distribution statistics

In this section, we plot and analyze the distributions of the Indegree, CC, and PL statistics of the representative and original graphs. For the sake of brevity, we present only the distribution statistics of the representative graphs at 20% of the size of *G*. Fractions of 10% to 60%, showed similar trends which were less accurate at 10%, and less extreme at 30%. Above 40% the results were less accurate since the algorithm methods added too many edges to G_R.

Fig. 4 presents the distributions of the three statistics that were calculated on the representative graphs and the relationship (target) graph *G*.

Indegree distribution (Fig. 4a and d). Both *RAG* and *RAG*1 present a similar slope and shape to the target distribution in both datasets. However, by capturing all streaming activity edges they over-represent low-degree nodes.

On Twitter (Fig. 4a), $RAG2M_{Br}$ overestimates high-degree nodes, but at a degree larger than ∼1,000, $RAG2M_{Br}$ underestimates high-degree nodes. On Facebook (Fig. 4d), $RAG2M_{Br}$ captures the shape of the distribution but underestimates high-degree nodes.

The Twitter activity graph is more clustered than the Facebook activity graph (Fig. 1b and e). By connecting unconnected neighbors of selected high-degree nodes, $RAG2M_{Br}$ overestimates high-degree nodes for degrees $< 1,000$ on Twitter (Fig. 4a).

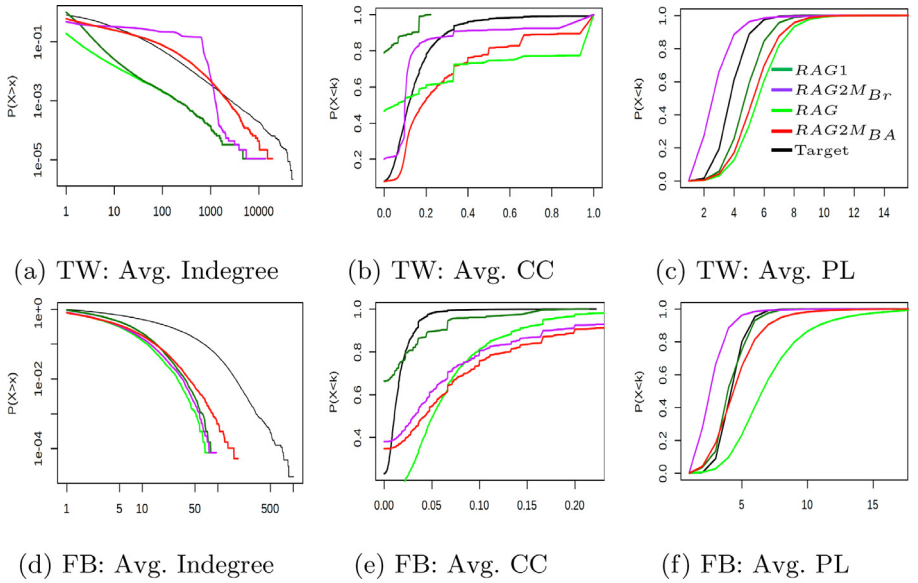


Fig. 4. Distribution statistics of representative graphs at 20% of the size of *G*.

On Twitter (Fig. 4a), $RAG2M_{BA}$ presents a similar shape to the target degree distribution, slightly underestimating high-degree nodes. On Facebook (Fig. 4d), $RAG2M_{BA}$ preserves the shape of the target degree distribution but underestimates high-degree nodes. The different trends of $RAG2M_{BA}$ in Fig. 4a and d can be attributed to the differences between the communities of the two datasets.

CC distribution (Fig. 4b and e). The representative graphs are highly clustered, over-representing nodes with high CC. The only exception is for *RAG*1 on Twitter. *RAG*1 creates low clustered graphs when applied to the Twitter dataset but presents better distributions when applied to the Facebook dataset. *RAG*1 captures the symmetric edge structure of the Facebook friendship graph but misses the non-symmetric edge structure of the Twitter relationship graph.

$RAG2M_{Br}$ creates a CC distribution that is similar to the Twitter target distribution in terms of shape. However, it produces less accurate estimations on Facebook, generating more clustered graphs.

$RAG2M_{BA}$ captures the shape of the curve of the statistics better than *RAG* on Twitter. However, on Facebook, this trend is reversed.

The representative graphs generated by $RAG2M_{BA}$, and $RAG2M_{Br}$ are highly clustered since they connect a fraction of unconnected neighbors of selected high-degree nodes.

The vertical jumps in the CC distributions (Fig. 4b and e) reflect graphs with non-continuous CC. Given that *RAG* uses the activity graph, the algorithm methods on which it relies are sensitive to changes in user activities. Hence, the bursty nature of Twitter affects the estimations by different representations.

PL distributions (Fig. 4c and f). *RAG* produces longer PLs on both datasets compared to the target statistic. It underestimates high-degree nodes in *G*, resulting in sparsely connected graphs that lead to long PLs.

*RAG*1 presents a high percentage of PL estimations on Twitter and produces estimations that are very close to the Facebook target statistic. On Facebook (Fig. 4f), *RAG*1 performs best by capturing the structure of Facebook's relationship graph where a friendship imposes bidirectional edges.

$RAG2M_{Br}$ produces biased estimations of the target statistics on both datasets by creating representative graphs with shorter PLs compared to the target statistic. $RAG2M_{Br}$ models edge formation between unconnected neighbors of high-degree nodes by a Bernoulli process, creating many short paths.

Table 4
A summary of the evaluations of the distribution statistics.

Model	Indegree		CC		PL	
	TW	FB	TW	FB	TW	FB
RAG	–	–	+	+	+	+
*RAG*1	–	–	–	+	+	*
$RAG2M_{Br}$	+[1]/–	–	+	+	–	–
$RAG2M_{BA}$	–	–	+	+	+	+

Over (+)/ under (–) representation of high Indegree nodes, high clustered nodes (CC), or large fraction of path-length (PL); (∗) Matches target.
[1]Up to degree ∼ 1,000.

For both datasets, $RAG2M_{BA}$ generates representative graphs with a higher percentage of longer PLs than the target distribution. $RAG2M_{BA}$ samples high degree nodes from G_R with replacements and connects their unconnected neighbors according to the BA Model, which does not create enough short paths.

Table 4 summarizes the observations regarding the over/under representation of the target statistic distributions based on Fig. 4.

In accordance with the evaluation steps (Section 3.3), we next measure the extent to which each representative graph represents the original graph.

5.3. Performance evaluation

To evaluate the performance of the representative graphs, we compute the KS-distance between the three distributions of the statistics of the original graphs and the representative graphs with varying sizes. A smaller distance indicates more similar cumulative distribution curves - a better representation.

For $RAG2M_{Br}$, and $RAG2M_{BA}$ we compute the average KS-distance of 10 runs (following [12]), and 6 graph sizes (10% to 60%) per dataset. We measured the average KS-distance between the distributions of the three statistics of G and the representative graphs (Fig. 5). Since *RAG* and *RAG*1 are deterministic, we measure their KS-distances only once. Fig. 5d presents the average KS-distances of Fig. 5a–c.

Indegree distribution (Fig. 5a). Assuming that G is unobtainable, $RAG2M_{BA}$ is the closest in terms of the KS-distance, in both datasets.

CC distribution (Fig. 5b). *RAG*1 outperforms all other methods on Facebook, and $RAG2M_{Br}$ outperforms all other methods on Twitter.

PL distribution (Fig. 5c). *RAG*1 outperforms all other methods on Facebook. $RAG2M_{Br}$ outperforms all other methods on Twitter.

As Fig. 5 indicates, no algorithm method performs well consistently across all statistics and all datasets. In case one needs to choose a single representation method, the average performance of each method was calculated in terms of KS-distance (Fig. 5d). According to Fig. 5d, $RAG2M_{Br}$ performs best on Twitter, and *RAG*1 performs best on Facebook.

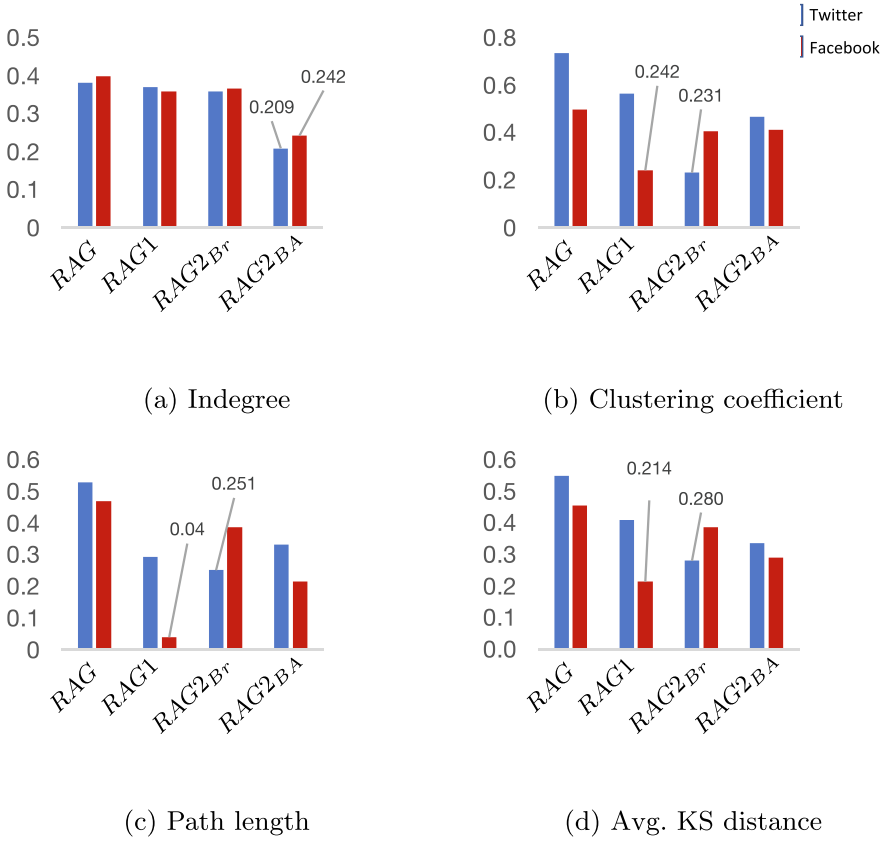


Fig. 5. Kolmogorov Smirnov distances for different methods at 20% of the size of G.

Based on the results of Fig. 5, $RAG2M_{Br}$ performs best in capturing the CC and the PL distributions on Twitter, which presents an asymmetric user relationship structure. $RAG2M_{BA}$ performs best in capturing the Indegree distribution on both Twitter and Facebook. *RAG*1 well represents the CC and PL distributions of the Facebook symmetric user relationship graph, since it captures the bi-directional nature of friendship edges on Facebook.

Low KS-distances and the similar shape of the distributions of the statistics indicate that the activity graph can represent a relationship graph, which answers RQ #1. The results show that there is no single representation method that best fits all of the analyzed graph statistics. Nevertheless, depending on the properties of interest to be matched and the structure of the relationship graph (asymmetric or not), one can choose an algorithm method that best preserves the desired properties of a large, unobtainable relationship graph by a streaming activity graph. For example, according to the results, Indegree distribution is best captured by $RAG2M_{BA}$ for both types of graphs.

To answer RQ #2, we discuss the parameters tested in each algorithm method.

5.4. Sensitivity analysis of parameters

Using an iterative computational method, we test different values of the *RAG*2 parameters (Table 1). Our goal is to minimize the average KS-distance of the Indegree, CC, and PL statistics, similar to Fig. 5d.

We test the following parameters (Table 1):

- M_{Br} with an edge formation probability $0.01 \leqslant P_B \leqslant 0.1$ with 0.01 steps. The best results ($RAG2M_{Br}$) are for $P_B = 0.05$.
- M_{FF} with varying parameters: $0.1 \leqslant P_f \leqslant 0.7$with 0.1 steps. We chose this range since FF was found to perform best when $p_f = 0.7$ [32]. We obtain the best results with $P_f = 0.2$. The results are similar to the results of $RAG2M_{BA}$ (elaborated next) and therefore, are not presented.
- M_{BA} with varying power $1 \leqslant \alpha \leqslant 3$ where $\alpha \in \mathbb{N}$ of the BA Model. We chose this range since scale-free graphs typically have $2 < \alpha < 3$ [13]. The best results ($RAG2M_{BA}$) are for linear preferential attachment ($\alpha = 1$).

We test each of the *RAG*2 configurations with a varying fraction of sampled nodes $0.01 \leqslant S \leqslant 0.05$ with 0.01 steps, from G_R. Nodes were sampled with a varying probability $P_{k,j}$, where $k \in \{Deg, PR, CC\}$, with ($j = 1$) or without ($j = 0$) replacement.

To expedite computations, the smaller the representative graph is, the better. We find that representative graphs at 20% of the size of the original graph (G) yield good estimations of the target statistics. The results show that a representative graph with 20% of the size of G is sufficient to represent the statistics of G. This finding answers RQ #2.

In the next section, we demonstrate an application for identifying influential users (influencers) in an unobtainable graph.

5.5. Identifying influencers

Different measures were suggested to identify influencers in an OSN [38]. The leading measures include Degree and k-core [38]. Specifically, the Indegree of a node reflects its popularity and can indicate its influence on others [38].

Our goal is to identify influencers in a large relationship graph (G). Since G is unobtainable, we identify influencers in the representative graphs of G. Influencers are a small fraction of the nodes ranked at the top of the k-core or at the top of the Indegree lists [38].

To evaluate our success in identifying influencers, we compute an overlap measure between the influencers identified in G and the influencers identified in the representative graphs. We define an overlap between the graphs (1).

$$Overlap(f_{\%}) = |I_{f_{\%}} \cap P_{f_{\%}}|/|I_{f_{\%}}| \tag{1}$$

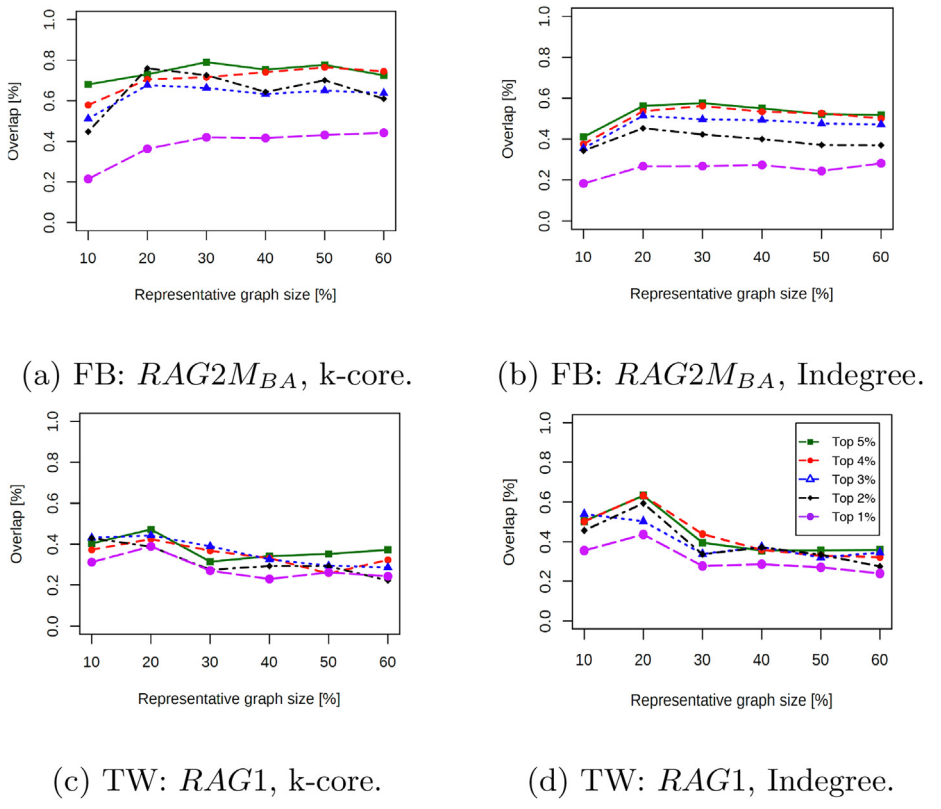
$I_{f\%}$ – The set of identified influencers (nodes), located at the top $f_{\%}$ of a ranking, based on an influence measure (k-core or Indegree) in G.

$P_{f_{\%}}$ - The set of influencers that are located at the top $f_{\%}$ of a ranking, based on an influence measure in a representative graph.

We evaluate the effect of different algorithm methods on a node's influence ranking under the k-core and the Indegree measures using four steps.

1. Calculate the k-core and Indegree for each G graph (TW and FB).
2. Generate representative graphs at different sizes (10% to 60% of G) by the most accurate method. According to our findings in Section 5.3, these are $RAG1, RAG2M_{Br}$, and $RAG2M_{BA}$.
3. Calculate the k-core and Indegree for each representative graph.
4. Calculate the overlap measure (Eq. (1)) with different fractions of the top $f_{\%} = \{x \in \mathbb{N} | 1\% \leqslant x \leqslant 5\%\}$ ranking under each influence measure.

Our goal is to find the top $f_{\%}$ influencers in an unobtainable graph by ranking users in a representative graph. The smaller the $f_{\%}$ and the higher the overlap are, the more successful we are at identifying influencers in G.



(a) FB: $RAG2M_{BA}$, k-core. (b) FB: $RAG2M_{BA}$, Indegree.

(c) TW: $RAG1$, k-core. (d) TW: $RAG1$, Indegree.

Fig. 6. Overlap score achieved by different representative graphs at varying sizes.

Among the three tested methods, $RAG2M_{BA}$ produces the highest overlap score for Facebook in both the k-core (Fig. 6a) and Indegree (Fig. 6b) measures. On Twitter, $RAG1$ outperforms other methods, presenting the highest overlap score for both k-core (Fig. 6c) and Indegree (Fig. 6d).

We focus on representative graphs at 20% of the size of G since they well preserve the properties of G (Section 5.2). At 20% with $f_{1\%}$, the overlap on Facebook under the k-core measure (Fig. 6a) is higher than the overlap under the Indegree measure (Fig. 6b). At 20% with $f_{1\%}$, the overlap on Twitter under the Indegree measure (Fig. 6d) is higher than the k-core measure (Fig. 6c).

Larger representative graphs ($> 20\%$) at $f_{1\%}$ improve the overlap on Facebook slightly but worsen the overlap on Twitter.

The best Facebook overlap score (0.76) is achieved by the k-core measure (Fig. 6a) using $RAG2M_{BA}$ at 30% of the size of G with $f_{5\%}$. In Fig. 6a, $f_{2\%}$ closely follows (0.74) the best overlap score at 20% of the size of G.

The best Twitter overlap score (0.63) is achieved by the Indegree influence measure (Fig. 6d) using $RAG1$ at 20% of the size of G with $f_{\%} \geqslant 4\%$. In Fig. 6d, $f_{2\%}$ at 20% of the size of G presents a close overlap score (0.58).

The results show that it is possible to identify influencers in an unobtainable relationship graph by using the proposed algorithm with different methods, depending on whether or not the unobserved graph is symmetric.

6. Discussion and conclusions

Popular OSNs are huge and the algorithms required to analyze them are of greater computational complexity than can be handled easily. Therefore, it is extremely hard to study and analyze the structure of these massive graphs.

Generating a small representative sample of a large graph is an essential tool for large graph analysis. However, directly sampling from a relationship graph is often impractical due to the graph's size, API or privacy limitations, or if the graph is hidden. Hence, these graphs are essentially unobtainable.

We proposed an algorithm that utilizes the smaller activity graph rather than generating a sample of the large relationship graph. Applying the proposed algorithm with different methods to the activity graph showed that a representative graph at 20% of the size of the large original relationship graph exhibits close parameter estimations to the original graph. However, no method performs best consistently across all statistics and all datasets. Even in the simpler case when the original graph is obtainable, there is no single sampling algorithm that best fits all distributions [32]. Depending on the properties of interest to be matched, and the structure of the relationship graph (asymmetric or not), one can choose an appropriate

algorithm method that best preserves the desired properties of a large, unobtainable relationship graph by a streaming activity graph.

Finally, we demonstrated an application for using the proposed algorithm with different methods to identify influencers in an unobtainable graph. The results show that determining the top 2% of influencers ($f_{2\%}$) in a representative graph that is 20% of the size of the original graph yields a high overlap score (0.74) on Facebook with the k-core measure. On Twitter, a representative graph with $f_{2\%}$ produces an overlap of 0.58 with the Indegree measure. Higher overlaps (0.63 on Twitter, and 0.76 on Facebook) require larger representation graphs (30%), and larger $f\%$.

This study has three main limitations. First, we experimented with a Twitter dataset, which is bursty and can affect the outcome of the results by over-representing nodes and edges or under-representing nodes with little activity. Second, the proposed algorithm generates a graph with weighted edges but does not try to preserve the weights of these edges as a graph property that can give insights into user interactions. Lastly, using a computational approach to search for the best model parameters, we tested different parameter values with varying intervals, which might "jump" across the optimal point, thereby missing it.

Our contribution is threefold: First, we develop an algorithm and methods that can represent a large unobtainable relationship graph by using a streaming activity graph. Second, we show that a representative graph that is 20% of the size of the relationship graph is sufficient to approximate the properties of the relationship graph. Third, we demonstrate how to analyze a small representative graph instead of a large, unobtainable relationship graph to determine the leading influencers in an OSN.

The activity graph enables us to analyze large unobtainable graphs, as well as graphs with extensive user privacy protections or graphs whose information takes a long time to collect. The proposed algorithm can be used to build patterns for graph mining applications such as role discovery in large OSNs or detecting influential users by centrality measures. Future research can utilize the results of this study to analyze privacy protected networks that can pose a serious threat to user privacy in OSNs.

CRediT authorship contribution statement

Alon Bartal: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing - original draft, Writing - review & editing, Visualization. **Gilad Ravid:** Conceptualization, Methodology, Investigation, Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Charu C. Aggarwal, Yuchen Zhao, S. Yu Philip, Outlier detection in graph streams, in: 2011 IEEE 27th International Conference on Data Engineering, IEEE, 2011, pp. 399–409.
[2] Nesreen K. Ahmed, Jennifer Neville, Ramana Kompella, Network sampling: from static to streaming graphs, ACM Trans. Knowl. Discovery Data 8(2) (2014) 7. ISSN 1556–4681. .
[3] Mohammad Al Hasan, Methods and applications of network sampling, INFORMS, 2016, 115–139. .
[4] Lawrence Ang, Community relationship management and social media, J. Database Market. Customer Strategy Manage. 18 (1) (2011) 31–38.
[5] Eytan Bakshy, Solomon Messing, Lada A. Adamic, Exposure to ideologically diverse news and opinion on facebook, Science 348 (6239) 1130–1132, 2015. ISSN 0036–8075. .
[6] Albert-László Barabási, Réka Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512. ISSN 0036–8075. .
[7] Alon Bartal, Gilad Ravid, Member behavior in dynamic online communities: role affiliation frequency model, IEEE Trans. Knowl. Data Eng. (2019).
[8] Alon Bartal, Nava Pliskin, Gilad Ravid, Modeling influence on posting engagement in online social networks: Beyond neighborhood effects, Soc. Networks 59 (2019) 61–76.
[9] Neli Blagus, Lovro Šubelj, Marko Bajec, Empirical comparison of network sampling: how to choose the most appropriate method?, Phys A 477 (2017) 136–148.
[10] T.H. Avrim Blum, Hubert Chan, Mugizi Robert Rwebangira, A random-surfer web-graph model, in: 2006 Proceedings of the Third Workshop on Analytic Algorithmics and Combinatorics (ANALCO) SIAM , SIAM, 2006, pp. 238–246.
[11] Robert A. Bridges, John Collins, Erik M. Ferragut, Jason Laska, Blair D. Sullivan, A multi-level anomaly detection algorithm for time-varying graph data with interactive visualization, Soc. Network Anal. Min. 6 (1) (2016) 99.
[12] Kai Cheng, Sampling from large graphs with a reservoir, in: 2014 17th International Conference on Network-Based Information Systems, IEEE, 2014, pp. 347–354.
[13] Fan Chung, Linyuan Lu, The average distance in a random graph with given expected degrees, Internet Math. 1 (1) (2004) 91–113.
[14] Gábor Csárdi, Tamás Nepusz, igraph reference manual. URL: http://igraph.sourceforge.net/documentation.html (accessed April, 20, 2010. .
[15] Manlio De Domenico, Antonio Lima, Paul Mougel, Mirco Musolesi, The anatomy of a scientific rumor. arXiv preprint arXiv:1301.2952, 2013. .
[16] David Dekker, David Krackhardt, Tom A.B. Snijders, Sensitivity of mrqap tests to collinearity and autocorrelation conditions. Psychometrika 72(4) (2007) 563–581. ISSN 0033–3123. .
[17] Christian Doerr, Norbert Blenn, Metric convergence in social network sampling, in: Proceedings of the 5th ACM workshop on HotPlanet , ACM, 2013, pp. 45–50.
[18] P. Alex Dow, La.da.A. Adamic, Adrien Friggeri, The anatomy of large facebook cascades, in: Seventh International AAAI Conference on Weblogs and Social Media, 2013.

[19] Omar El-Daghar, Erik Lundberg, Robert Bridges, Egbter: capturing degree distribution, clustering coefficients, and community structure in a single random graph model, in: 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), IEEE, 2018, pp. 282–289.
[20] Minyu Feng, Hong Qu, Zhang Yi, Xiurui Xie, Jürgen Kurths, Evolving scale-free networks by poisson process: modeling and degree distribution, IEEE Trans. Cybern. 46 (5) (2016) 1144–1155.
[21] David Hallac, Youngsuk Park, Stephen Boyd, Jure Leskovec, Network inference via the time-varying graphical lasso, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining , ACM, 2017, pp. 205–213.
[22] Kayla Haye, Joshua Embree, Marc Punkay, Dorothy L. Espelage, Joan S. Tucker, Harold D. Green, Jr., Analytic strategies for longitudinal networks with missing data, Soc. Netw. 50 (2017) 17–25. ISSN 0378–8733. .
[23] John R. Hipp, Cheng Wang, Carter T. Butts, Rupa Jose, Cynthia M. Lakon, Research note: the consequences of different methods for handling missing network data in stochastic actor based models, Soc. Netw. 41 (2015) 56–71. ISSN 0378–8733. .
[24] Mark Huisman, Christian Steglich, Treatment of non-response in longitudinal network studies, Soc. Netw. 30(4) (2008) 297–308. ISSN 0378–8733. .
[25] Bo Jiao, Jianmai Shi, Wensheng Zhang, Lining Xing, Graph sampling for internet topologies using normalized laplacian spectral features, Inf. Sci. 481 (2019) 574–603.
[26] Indika Kahanda, Jennifer Neville, Using transactional information to predict link strength in online social networks, ICWSM 9 (2009) 74–81.
[27] Paul L. Krapivsky, Sidney Redner, A statistical physics perspective on web growth, Comput. Netw. 39(3) (2002) 261–276. ISSN 1389–1286. .
[28] Maciej Kurant, Athina Markopoulou, Patrick Thiran, On the bias of bfs (breadth first search), in: Teletraffic Congress (ITC), 2010 22nd International, IEEE, pp. 1–8. ISBN 1424488362. .
[29] Maciej Kurant, Athina Markopoulou, Patrick Thiran, On the bias of bfs. arXiv preprint arXiv:1004.1729, 2010. .
[30] Silvio Lattanzi, D. Sivakumar, Affiliation networks, in: Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing, Citeseer, 2009, pp. 427–434.
[31] Sang Hoon Lee, Pan-Jun Kim, Hawoong Jeong, Statistical properties of sampled networks, Phys. Rev. E 73 (1) (2006) 016102.
[32] Jure Leskovec, Christos Faloutsos, Sampling from large graphs, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2006, pp. 631–636.
[33] Jurij Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, Realistic, mathematically tractable graph generation and evolution, using kronecker multiplication, in: European Conference on Principles of Data Mining and Knowledge Discovery, Springer, 2005, pp. 133–145.
[34] Zhimin Li, Lu. Min, Wu. Xiaotai, The connectivity probability of edge evolving network driven by compound poisson process, Neurocomputing 218 (2016) 13–16.
[35] Yongsub Lim, Minsoo Jung, U. Kang, Memory-efficient and accurate sampling for counting local triangles in graph streams: from simple to multigraphs, ACM Trans. Knowl. Discov. Data 12(1) (2018) 4. ISSN 1556–4681. .
[36] Danielle Lottridge, Frank R Bentley, Let's hate together: how people share news in messaging, social, and public networks, in: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems , ACM, 2018, p. page 60..
[37] László Lovász, Random walks on graphs. Combinatorics, Paul erdos is eighty 2 (1993) 1–46. .
[38] Linyuan Lü, Duanbing Chen, Xiao-Long Ren, Qian-Ming Zhang, Yi-Cheng Zhang, Tao Zhou, Vital nodes identification in complex networks, Phys. Rep. 650 (2016) 1–63. ISSN 0370–1573. .
[39] Lawrence Page, Sergey Brin, Rajeev Motwani, Terry Winograd, The pagerank citation ranking: bringing order to the web. Technical Report 1999–66, Stanford InfoLab, November 1999, 1999. .
[40] Bruno Ribeiro, Don Towsley, Estimating and sampling graphs with multidimensional random walks, in: Proceedings of the 10th ACM SIGCOMM conference on Internet measurement , ACM, 2010, pp. 390–403.
[41] Comandur Seshadhri, Tamara G. Kolda, Ali Pinar, Community structure and scale-free collections of erd s-rényi graphs, Phys. Rev. E 85 (5) (2012) 056109.
[42] Kijung Shin, Euiwoong Lee, Jinoh Oh, Mohammad Hammoud, Christos Faloutsos, Dislr: distributed sampling with limited redundancy for triangle counting in graph streams. arXiv preprint arXiv:1802.04249, 2018. .
[43] Lorenzo De Stefani, Alessandro Epasto, Matteo Riondato, Eli Upfal, Triest: counting local and global triangles in fully dynamic streams with fixed memory size, ACM Trans. Knowl. Discov. Data 11(4) (2017) 43. ISSN 1556–4681. .
[44] Michael P.H. Stumpf, Carsten Wiuf, Robert M. May, Subnets of scale-free networks are not scale-free: sampling properties of networks, Proc. Natl. Acad. Sci. U.S.A. 102(12) (2005) 4221–4224. ISSN 0027–8424. .
[45] Bimal Viswanath, Alan Mislove, Meeyoung Cha, Krishna P. Gummadi, On the evolution of user interaction in facebook, in: Proceedings of the 2nd ACM Workshop on Online Social Networks , ACM, 2009, pp. 37–42.
[46] Duncan J. Watts, Steven H. Strogatz, Collective dynamics of'small-world'networks, Nature 393(6684) (1998) 440. ISSN 0028–0836. .
[47] Christo Wilson, Bryce Boe, Alessandra Sala, Krishna P.N. Puttaswamy, Ben Y. Zhao, User interactions in social networks and their implications, in: Proceedings of the 4th ACM European Conference on Computer Systems, ACM, 2009, pp. 205–218. .
[48] Bowen Yan, Steve Gregory, Identifying communities and key vertices by reconstructing networks from samples, PLoS One 8(4) (2013) e61006. ISSN 1932–6203. .
[49] Libo Yang, Steven M. Lavalle, The sampling-based neighborhood graph: an approach to computing and executing feedback motion strategies, IEEE Trans. Robot. Autom. 20(3) (2004) 419–432. ISSN 1042–296X. .
[50] Jian Zhang, A survey on streaming algorithms for massive graphs, Manag. Min. Graph Data (2010) 393–420.