Unstable Communities in Network Ensembles

Ahsanur Rahman* Steve T. K. Jan* Hyunju Kim† B. Aditya Prakash*
T. M. Murali*‡

Abstract

Ensembles of graphs arise in several natural applications. Many techniques exist to compute frequent, dense subgraphs in these ensembles. In contrast, in this paper, we propose to discover maximally variable regions of the graphs, *i.e.,* sets of nodes that induce very different subgraphs across the ensemble. We first develop two intuitive and novel definitions of such node sets, which we then show can be efficiently enumerated using a level-wise algorithm. Finally, using extensive experiments on multiple real datasets, we show how these sets capture the main structural variations of the given set of networks and also provide us with interesting and relevant insights about these datasets.

1 Introduction

Ensembles of graphs arise in several natural applications such as disease spreading, mobility tracking, and social networks. Mining such graph ensembles is an increasingly important problem, especially in the current era of 'big-data.' A typical problem solved by many existing techniques is to identify subgraphs of interest in these graphs. A well-studied problem is to compute frequent dense subgraphs (see Section 2). For example, a group of people who are well-connected in a large fraction of the graphs may indicate a close-knit community.

Here, we study the opposite problem. Consider a set of nodes such that the subgraphs induced by this group show considerable variation across a network ensemble. We can think of such a group as an "unstable community." Figure 1 displays the social network of three characters in the popular TV show *Friends*: Chandler, Joey and Kathy. Chandler and Joey experience breakup and renewal of their friendship because of their dating and/or breakup with Kathy, which is reflected in their social networks. Over time, these three characters induce as many as five of the eight possible subgraphs among them. Such unstable communities can arise in many different applications including contact networks, communication networks and citation networks (see Experiments in Section 6). Moreover, several such groups may be present in an ensemble of networks. Formalizing the concept of unstable communities and computing them efficiently are challenging tasks that have not been addressed by the data mining community.

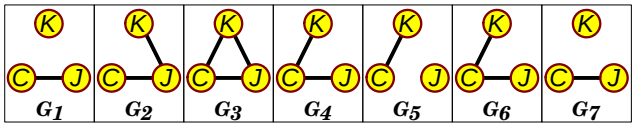


Figure 1: An example of an unstable community from the TV Show *Friends* consisting of three characters: Chandler (C), Joey (J), and Kathy (K). Their relationship network varies over time as follows: C and J have long been best friends (G_1). K starts dating J (G_2). Later, K and C fall in love with each other even while she continues to date J (G_3). Then, K breaks up with J (G_4). When J learns about C's love for K, he shuns C (G_5). After a while, J forgives C while C continues to date K (G_6). Finally, C breaks up with K (G_7).

The frequency distribution of the five subgraphs in Figure 1 is almost uniform (three graphs occur once and two graphs occur twice). Intuitively, an unstable community should induce several subgraphs and the frequency distribution of these subgraphs should be near-uniform. Hence, the "instability" of such a community may be captured by computing the difference between the frequency distribution of its induced subgraphs and the uniform distribution. We use this idea to make the following contributions herein:

(a) **Problem Formulation:** We propose the first formal definitions of unstable communities (UCs in short) in network ensembles and introduce a new class of graph mining problems based on these definitions (Section 3). We also compare these definitions and discuss their trade-offs (Sections 4 and 6.3.1).

(b) **Algorithm:** We show that our definitions have the desirable property of anti-monotonicity, which we leverage to develop efficient algorithms to enumerate UCs (Section 5).

(c) **Results:** We apply our algorithms to diverse

*Department of Computer Science, Virginia Tech, VA, USA. Email: {ahsanur, tekang, badityap, murali}@cs.vt.edu.

†Beyond Center for Fundamental Concepts in Science, Arizona State University, AR, USA. Email: hyunju.kim@asu.edu.

‡ICTAS Center for Systems Biology of Engineered Tissues, Virginia Tech, VA, USA.

datasets such as phone call, proximity, citation, co-authorship and communication networks. Apart from showing that the distribution of UCs varies from one dataset to another (Section 6.3.2) and that UCs cannot be computed by methods that mine frequent, dense subgraphs (Section 6.3.3), we demonstrate two powerful applications of UCs: (i) **Network sensors:** We show that a small number of UCs can succinctly track the structural variation within the entire network ensemble (Section 6.3.4). (ii) **Epidemiological monitoring:** We show that the pattern of infection of nodes within UCs is a leading indicator of epidemic spread in the entire network (Section 6.3.5).

2 Related Work

We survey related work in this section. Briefly, none of the research discussed below considers structural variation in network ensembles to find informative and interesting sets of nodes.

Mining Static Graphs: This work can be grouped into three levels. First, on the graph level, representative works include patterns and "laws" discovery, e.g., power law distributions [9], small world phenomena [21], and numerous other regularities. Next, on the subgraph level, work includes frequent subgraph mining and indexing [13] and community detection [15]. Finally, at the node level, work includes node proximity and recommendation systems [17], link prediction [20] and ranking [10]. There is active research on blogs and social networks, trying to model link behavior in large-scale on-line data [18], with work on viral marketing [16], and virus propagation [22]. In this paper, in contrast, we focus on ensembles of networks.

Mining Dynamic Graphs: Most research here has focused on community evolution [2, 6] and dynamic tensor analysis [24]. A line of work in databases looks into supporting continuous queries over streaming graph data [4], including work on streaming algorithms for specific problems (e.g., counting triangles, PageRank computation, sketching, etc.) [12], and on theoretical models and approximation algorithms. While this line of research is very valuable, our paper focuses on extracting structurally *variable* sets of nodes, as opposed to *cohesive* sets or stable communities.

Frequent Subgraph Mining: Recent work of frequent graph mining [14] takes a compendium of labeled graphs as input and finds all connected and/or dense graphs (suitably defined) that occur *frequently* as subgraphs of the graphs in the compendium. Similar techniques for mining labeled graphs (also called "relational graphs") have been proposed in the data mining community [5, 25]. While very few algorithms were developed for optimal graph pattern mining, there are an abundant number of optimal itemset mining algorithms available [8]. As we show later, UCs are *fundamentally* different from frequent subgraphs as they represent *variation* rather than constant/stable patterns.

Computational Biology: Battle *et al.* discovered sets of genes which induced diverse linear paths across multiple Bayesian networks inferred from genetic interaction data [3]. Rahman *et al.* [23], computed balance-and support-based UCs in the same dataset. In practice, there are two challenges with their approach: (i) it is difficult to assign meaningful values to their parameters and (ii) their heuristic algorithm could not guarantee the computation of all UCs. In this paper, we resolve both issues. First, we develop novel definitions of UCs that depend only on one parameter. Second, we design algorithms to enumerate all UCs exactly.

3 Problem Formulation and Definitions

Informally, a UC is a set of nodes with the following property: if we count how many times each distinct graph among these nodes appears as a subgraph in a given ensemble of graphs, this distribution is as close to uniform as possible. In other words, a set of nodes is a UC if the relative entropy between the subgraph probabilities of these nodes and the uniform distribution is at most a user-specified threshold.

We start by introducing some notation. Let $\mathcal{G}$ be a set of undirected and unweighted graphs. We assume that every graph in $\mathcal{G}$ contains the same set V of vertices but the edges may vary between the graphs. Each vertex has a label that identifies it uniquely. Given a set $U \subseteq V$ of k nodes and a graph $G \in \mathcal{G}$, let $G(U)$ denote the subgraph of G induced by U. Let $\mathcal{G}(U) = \{G(U) | G \in \mathcal{G}\}$ be the multiset of subgraphs induced by U in each of the graphs in $\mathcal{G}$. Note that $\mathcal{G}(U)$ is a multiset since U may induce the same subgraph in different graphs in $\mathcal{G}$. Let $\mathcal{P}(U)$ denote the set of $2^{\binom{k}{2}}$ possible graphs on the nodes in U. For a graph $G \in \mathcal{P}(U)$, let $\psi_{\mathcal{G}}(G)$ denote the number of times G is a member of $\mathcal{G}(U)$. Let X and Y be two discrete random variables whose sample space is $\mathcal{P}(U)$, i.e., each value that X or Y can take is a graph defined over U. For every graph G in $\mathcal{P}(U)$, we define the probability that X and Y equals G as follows:

$$\Pr(X = G) = \frac{\psi_{\mathcal{G}}(G)}{|\mathcal{G}|} \qquad \Pr(Y = G) = \frac{1}{2^{\binom{k}{2}}}$$

In other words, X is the observed distribution of these graphs in $\mathcal{G}$ while Y is the uniform distribution. We define $S_{\mathcal{G}}(U)$, the *subgraph divergence (SD) of U in $\mathcal{G}$*, as the relative entropy of X from Y, i.e.,

$$\begin{aligned} S_{\mathcal{G}}(U) &= \sum_{\substack{G \in \mathcal{P}(U) \\ \psi_{\mathcal{G}}(G) \neq 0}} \Pr(X = G) \log_2 \left(\frac{\Pr(X = G)}{\Pr(Y = G)} \right) \\ &= \binom{k}{2} + \sum_{\substack{G \in \mathcal{P}(U) \\ \psi_{\mathcal{G}}(G) \neq 0}} \frac{\psi_{\mathcal{G}}(G)}{|\mathcal{G}|} \log_2 \frac{\psi_{\mathcal{G}}(G)}{|\mathcal{G}|} \end{aligned}$$

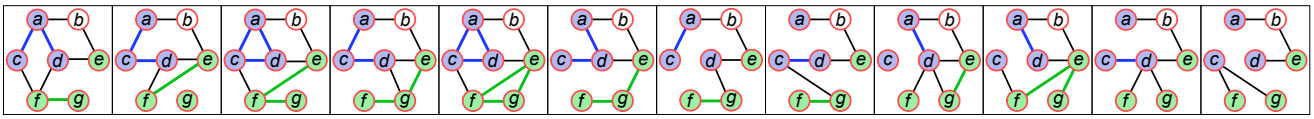


Figure 2: Illustrate of our definitions of UCs. $\{e, f, g\}$ is a 0.1446-SD-UC as well as a 0.0482-SSD-UC (Section 3). $\{a, b, d, e\}$ is a bad 5-SD-UC (Section 4).

SD-UC. Given a set $\mathcal{G}$ of graphs and a parameter $\rho \geq 0$, we say that a set of nodes U is a *ρ-SD-UC* if its subgraph divergence $S_{\mathcal{G}}(U) \leq \rho$. Note that the closer the distribution of $p_{\mathcal{G}}(G)$ is to being a uniform distribution, the smaller $S_{\mathcal{G}}(U)$ is. Moreover, $0 \leq S_{\mathcal{G}}(U) \leq \binom{|U|}{2}$. Therefore, the range of possible values for the subgraph divergence of a set of nodes increases with the size of the set. Next, we propose a normalized definition that addresses this point.

We define $T_{\mathcal{G}}(U)$, the *scaled subgraph divergence (SSD) of U in $\mathcal{G}$*, as the ratio of the subgraph divergence of U and its maximum possible value, i.e., $T_{\mathcal{G}}(U) = S_{\mathcal{G}}(U)/\binom{|U|}{2}$. We now give an analogous definition of UC based on the scaled subgraph divergence.

SSD-UC. Given a set $\mathcal{G}$ of graphs and a parameter $\sigma \geq 0$, we say that a set of nodes U is a *σ-SSD-UC* if (a) its scaled subgraph divergence $T_{\mathcal{G}}(U) \leq \sigma$ and (b) every subset of U is a *σ-SSD-UC*. Finally, the following definition captures when an SD-UC is maximal.

Maximal SD-UC: Given a set $\mathcal{G}$ of graphs and a parameter $\rho \geq 0$, a set of nodes U is a *maximal ρ-SD-UC* if no proper superset of U is a ρ-SD-UC. We formulate an analogous definition for a maximal σ-SSD-UC.

Figure 2 explains the two definitions of UC using a set $\mathcal{G}$ of 12 graphs on the set of nodes $\{a, b, c, d, e, f, g\}$. Consider the set of nodes $\{e, f, g\}$. All the eight possible graphs among these nodes appear in $\mathcal{G}$, with two graphs appearing twice each, one graph appearing thrice, and each of the remaining ones appearing once. So $S_{\mathcal{G}}(\{e, f, g\}) = 0.1446$ and $\{e, f, g\}$ is a 0.1446-SD-UC. Further, $T_{\mathcal{G}}(\{e, f, g\}) = 0.0482$ and each subset of $\{e, f, g\}$ has scaled subgraph divergence lower than 0.0482. Therefore, $\{e, f, g\}$ is a 0.0482-SSD-UC.

4 Comparison between Definitions

In this section, we provide the rationale for developing two definitions of unstable communities and give insights into their relative advantages and disadvantages.

Subgraph divergence is a natural way to measure the difference between the observed distribution of subgraphs and the uniform distribution. Moreover, we can prove that the property of being an SD-UC is anti-monotone (Lemma 5.2 in Section 5). Unlike SD-UC, the anti-monotonicity of SSD-UC has to be enforced via condition (b) in its definition. Therefore SD-UC is theoretically more attractive to SSD-UC. However, the values of subgraph divergence are not comparable across UCs of different sizes. For example, a set of three nodes may have a subgraph divergence of at most three while a set of four nodes may have a subgraph divergence of at most six. In contrast, the value of scaled subgraph divergence is bounded from above by one, allowing us to consider UCs of different sizes on the same scale.

Another advantage of the SSD-UC definition over SD-UC lies in the following fact: an SSD-UC forces an upper bound on both the scaled subgraph divergence and the subgraph divergence of each of its subsets while an SD-UC guarantees an upper bound only on the subgraph divergence of each of its subsets. Specifically, a σ-SSD-UC of size k cannot contain a subset of size $l < k$ whose scaled subgraph divergence is larger than σ or, equivalently, whose subgraph divergence is larger than $\sigma\binom{l}{2}$. Similarly, the anti-monotonicity of the SD-UC definition guarantees that a ρ-SD-UC of k nodes cannot contain a subset whose subgraph divergence is larger than ρ. However, it may contain a subset whose scaled subgraph divergence is larger than $\rho/\binom{k}{2}$, *i.e.,* an SD-UC may contain a subset with a larger scaled subgraph divergence than its own. We formalize this idea in the next definition.

Bad SD-UC: Given a set $\mathcal{G}$ of graphs and a parameter $\rho \geq 0$, a ρ-SD-UC U is *bad* if it has a subset $W \subset U$ such that $T_{\mathcal{G}}(W) > \rho/\binom{|U|}{2} \geq T_{\mathcal{G}}(U)$. Note that the second inequality in the definition is simply a restatement of the fact that U is a ρ-SD-UC.

The set $A = \{a, b, d, e\}$ in Figure 2 is an example of a bad SD-UC. Since A induces two subgraphs in $\mathcal{G}$ each with frequency six, $S_{\mathcal{G}}(A) = 5$. However, the subset $B = \{a, b, e\}$ of A induces the same subgraph across all graphs in $\mathcal{G}$. Hence $T_{\mathcal{G}}(B) = 1 > 5/\binom{|A|}{2} = 5/6$. By the above definition, A is a bad 5-SD-UC. Intuitively, A is a bad UC because one of its subsets (B) induces the same three-node subgraph across all the input graphs. We will show in Section 6.3.1 that bad SD-UCs are common in real networks.

5 Mining UCs

Problems: We seek to solve the following pair of problems in this paper:

P1. *Given a set of graphs $\mathcal{G}$ and a parameter $\rho \geq 0$, enumerate all maximal ρ-SD-UCs.*

P2. *Given a set of graphs $\mathcal{G}$ and a parameter $0 \leq \sigma < 1$, enumerate all maximal σ-SSD-UCs.*

We now state two theorems whose proofs we omit due to lack of space.

THEOREM 5.1. *Given a set $\mathcal{G}$ of graphs and a parameter $\rho \geq 0$ (respectively, $0 \leq \sigma < 1$), the problem of deciding whether G contains a maximal ρ-SSD UC (respectively, σ-SSD UC) is NP-complete.*

Anti-monotonicity: Our algorithms exploit the fact that the property of being an SD-UC is anti-monotone.

THEOREM 5.2. *Let $\mathcal{G}$ be a set of graphs and U be a set of nodes. For every node a in U, we have $S_{\mathcal{G}}(U \setminus \{a\}) \leq S_{\mathcal{G}}(U)$,* i.e., *removing a node from U does not increase its subgraph divergence.*

Algorithm 1 COMPUTESDUCS ($\mathcal{G}$, ρ)

Require: A set $\mathcal{G}$ of graphs, $0 \leq \rho$.
Ensure: All ρ-SD-UCs.
1: $\mathcal{S} \leftarrow \{(u, v) \in V \times V \mid S_{\mathcal{G}}(\{u, v\}) \leq \rho\}$
2: **while** $\mathcal{S}$ is not empty **do**
3: $\quad\mathcal{T} \leftarrow \phi$
4: $\quad$**for** every set $U \in \mathcal{S}$ **do**
5: $\quad\quad$Compute $S_{\mathcal{G}}(U)$
6: $\quad\quad$**if** $S_{\mathcal{G}}(U) \leq \rho$ **then**
7: $\quad\quad\quad$Output U
8: $\quad\quad\quad$Insert U into $\mathcal{T}$
9: $\quad\mathcal{S} \leftarrow$ GENERATE-CANDIDATES($\mathcal{T}$)

Algorithms: We exploit the anti-monotonicity of SD-UCs to develop a level-wise algorithm (Algorithm 1) for computing all maximal ρ-SD-UCs. Note that SSD-UCs are anti-monotone, by definition. Thus we can use this algorithm to mine both kinds of UCs. Our algorithm is in the Apriori style [1]. We stress that we can adapt any algorithm for mining maximal itemsets to our purpose.

In the algorithm, $\mathcal{S}$ is the current candidate set of UCs; each candidate in $\mathcal{S}$ contains the same number of nodes, say $k \geq 2$. We start with two-node UCs (Step 1), specifically pairs that are connected by an edge in at least one graph in $\mathcal{G}$. Each iteration of the while loop (Step 2) increases the size of the candidate UCs in $\mathcal{S}$ by one. We compute the subgraph divergence of each candidate UC in $\mathcal{S}$ (Step 5). We output only those candidates whose subgraph divergence is at most ρ (Step 7), while also storing these UCs in the set $\mathcal{T}$. Given the set $\mathcal{T}$ of all UCs with k nodes, we construct the set $\mathcal{S}$ of candidate UCs with $k+1$ nodes using GENERATE-CANDIDATES (Step 9).

The GENERATE-CANDIDATES subroutine is identical to the one in the *Apriori* algorithm [1]; we do not provide pseudo-code for this subroutine. Here, we exploit Theorem 5.2 to generate candidate UCs with $k+1$ nodes by merging two k-node UCs when they share $k-1$ nodes [1]. For each candidate UC U, we ensure that every subset of U with k nodes is a UC, *i.e.,* an input to GENERATE-CANDIDATES. If not, we discard U. Note that this algorithm may output non-maximal UCs. It is not difficult to modify it to compute only non-maximal UCs.

Running Time: Let the input ensemble $\mathcal{G}$ contain m graphs, each with n vertices. There are at most $\binom{n}{k}$ k-node UCs. To compute the SD or SSD of each such UC, we examine the subgraphs induced by it across $\mathcal{G}$ and count their frequencies, which takes $O\left(m\binom{k}{2}\right)$ time. Hence the worst case running time of our miner is $O\left(m\sum_{k=2}^{n}\binom{k}{2}\binom{n}{k}\right)$ *i.e.,* $O(mn^2 2^{n-2})$, i.e., linear in the number of graphs but exponential in the number of vertices. However, as we will show in Section 6.3.7, our miner is very efficient in practice.

6 Experiments

6.1 Experimental Setup We used a Linux 64-bit server with 128 GB RAM and 16 2.4 GHz Intel(R) Xeon(R) CPU cores for all our operations. We implemented our UC miners in C++. Our code is available as part of the `biorithm` package at `http://bioinformatics.cs.vt.edu/~murali/software/biorithm`.

6.2 Datasets We applied our UC miner on six sets of networks from different domains (Table 1). We chose these datasets to show that unstable communities can exist in different types of networks. The largest dataset (DBLP) contained nearly 1.5M nodes and 10.6M edges. We ignored the directionality of edges in directed networks because we sought to focus solely on the existence or the lack of interaction between two nodes. Thus all of our networks were undirected.

6.3 Results We divide our results into multiple parts: (a) superiority of SSD-UCs over SD-UCs, (b) differences of scaled subgraph divergence values among networks, (c) discovering network sensors, (d) using sensors as epidemiological monitors, (e) qualitative analysis of SSD-UCs, and (f) the efficiency of SSD-UC miner. We restricted our attention to UCs with three or more nodes for all our analyses. Due to lack of space, we show results only for a subset of the datasets in this paper and note that the results were qualitatively similar for the other datasets.

6.3.1 Superiority of SSD-UCs over SD-UCs As we discussed in Section 4, a potential problem with the SD-UC definition is that it can yield bad SD-UCs. To examine this possibility, for each network, we computed ρ-SD-UCs for $\rho \in \{1, 2, 3, 4, 5, 6\}$. For each value ρ and for $k \in \{3, 4\}$, we computed the number of bad ρ-SD-UCs among all k-node ρ-SD-UCs. Figure 3 shows how

Table 1: Properties of analyzed network ensembles. #Nets is the number of networks and #Edges is the total number of edges in the dataset.

Network	Domain	Start time	End time	Interval	#Nets	#Nodes	#Edges
1. SE-Prox	Mobility	Oct 1, 2008	May 31, 2009	1 month	8	75	11,184
2. SE-Phone	Phone calls	Oct 1, 2008	May 31, 2009	1 month	8	75	1,075
3. Hospital	Public health	Dec 6, 2010 1:00 PM	Dec 10, 2010 2:00 PM	1 hour	97	75	32,424
4. HEP-PH	Citations	1992	2002	1 year	11	30,566	349,944
5. LBNL	Network traces	Dec 15, 2004 3:42 PM	Dec 15, 2004 4:42 PM	1 min	61	2,751	68,947
6. DBLP	Co-authorship	1938	2015	1 year	73	1,482,029	10,597,016

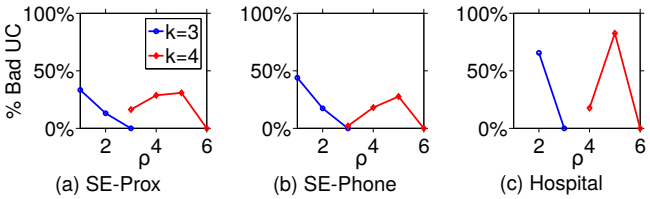


Figure 3: Percentages of bad k-node SD-UCs for different values of ρ for the (a) SE-Prox, (b) SE-Phone, and (c) Hospital networks. The Hospital networks did not contain any 1-SD-UC. HEP-PH networks did not yield any bad SD-UC.

the percentage of bad SD-UCs changes with ρ and k. Note that a k-node SD-UC cannot be bad for $\rho = \binom{k}{2}$ because it is not possible for any set of nodes to have scaled subgraph divergence larger than one. Therefore, the percentage of bad SD-UCs is always zero for $k = 3, \rho = 3$ and for $k = 4, \rho = 6$. For most other (ρ, k) combinations, a significant number of SD-UCs were bad. For some pairs, 50% or more of the SD-UCs were bad. In the extreme, *all* 1-SD-UCs in the LBNL dataset were bad. Note that even using a restrictive value of $\rho < 1$ did not guarantee the elimination of all bad SD-UCs. For example, more than 63% of the 0.9-SD-UCs in LBNL networks were bad SD-UCs. Moreover, in practice, we did not find any ρ-SD-UCs with four or more nodes when $\rho < 1$. Due to this inherent weakness of SD-UCs, *we focus on SSD-UCs in the rest of this paper.*

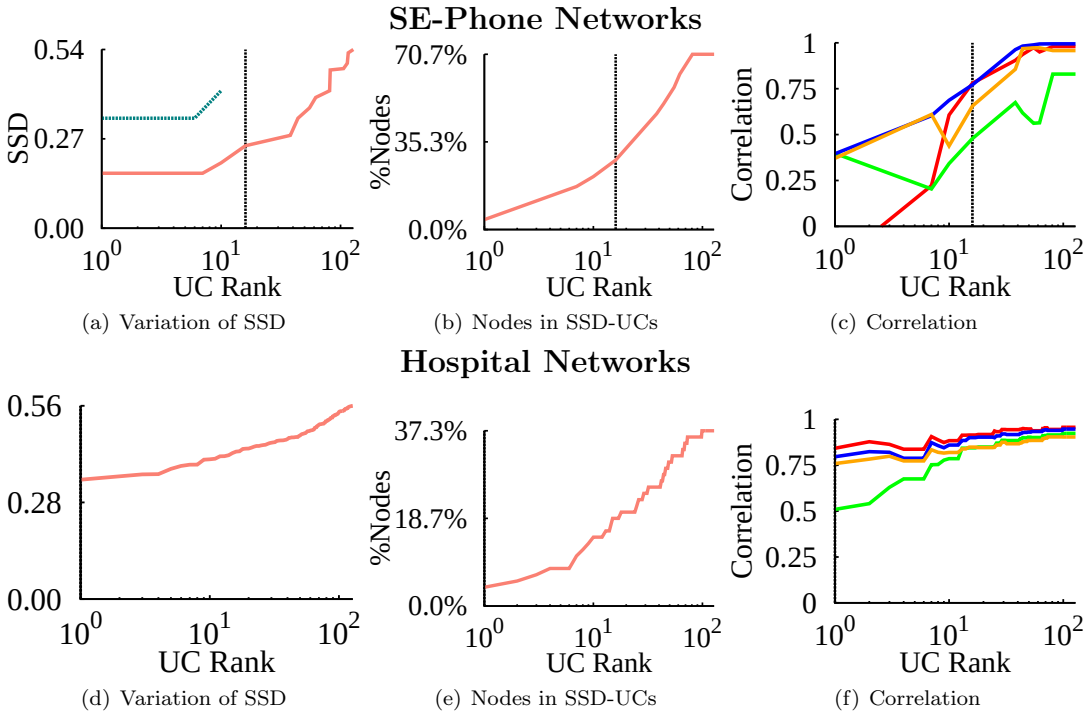
6.3.2 Differences among Networks SSD-UCs capture the structural variations in a network ensemble. To determine if SSD values varied from one ensemble to another, we computed a spectrum of σ-SSD-UCs in each dataset for $\sigma \in \{0.1, 0.2, \ldots, 0.6\}$. We sorted the SSD-UCs in each dataset in increasing order of their scaled subgraph divergences. For every rank $r \leq 128$, we plotted the scaled subgraph divergence of the rth UC and the percentage of nodes in the network that were members of the top r UCs. The results are available in the following figures: 4(a) and 4(b) (SE-Phone) and 4(d) and 4(e) (Hospital). Note that the x-axes of these figures are in the logarithmic scale and that the ranges of the y-axes of these plots are different.

We observed that the variation of scaled subgraph divergence with UC rank differed from one dataset to another. We also noted that the percentage of the nodes in the top 128 UCs varied considerably from one dataset to another: 70.7% for SE-Phone (Figure 4(b)) and 37.3% for Hospital (Figure 4(e)). Hence, we concluded that SSD values indeed differed among datasets.

Across all σ values, the LBNL datasets yielded only 26 UCs covering only 0.8% of the nodes. UCs in the DBLP dataset covered a much smaller fraction of the nodes. Since these datasets did not contain much variation, we omitted them from further analysis.

6.3.3 Comparison to Dense Subgraphs We sought to demonstrate that our notion of UC is quite dissimilar to the concept of frequent dense subgraphs in graph ensembles. Since all our UCs were of size three, we attempted to compute all frequent dense subgraphs with three nodes. The NetsTensor algorithm [19], which was designed for this problem, was not able to compute any such subsets. Therefore, we applied a cluster-based algorithm [23] to compute three-node sets that induced the same subgraph in at least half the graphs in the ensemble. We retained a node set only if the most frequent subgraph induced by it contained at least one edge. We refer to such node sets as "clusters." We computed the SSD of each cluster and plotted the variation of these SSD values with rank for the top 128 clusters. The teal curves in Figure 4 display these results. Note that we did not find any clusters in the Hospital dataset. These results show that clusters had much higher scaled subgraph divergences than UCs, supporting our claim that UCs are quite distinct from frequent dense subgraphs.

6.3.4 Discovering Network Sensors Graph properties such as density and average clustering coefficient are often used to summarize the structures of graphs. We chose four commonly used structural properties [11]:



(a) Variation of SSD (b) Nodes in SSD-UCs (c) Correlation

(d) Variation of SSD (e) Nodes in SSD-UCs (f) Correlation

Figure 4: Variation of scaled subgraph divergence (Section 6.3.2), percentage of nodes in SSD-UCs, and the correlation of structural properties (Section 6.3.4) with SSD-UC rank for (a-c) SE-Phone and (d-f) Hospital networks. Teal curves indicate SSD values for frequent subgraphs (FS). The vertical dotted line represents the rank cutoff we report in Table 2. The cutoff for the Hospital dataset is one. Colors in the correlation plots: red for DEN, green for ACC, blue for MEV, and orange for MD.

density (DEN), average clustering coefficient (ACC), maximum eigen value (MEV), and maximum degree (MD). We noticed that the value of each structural property varied among the networks in a dataset, raising an interesting question: can we summaries the differences among these networks using SSD-UCs? If so, will the nodes in the "summary" be able to represent all the networks and thereby act as "network sensors?" In this section, we try to answer these questions.

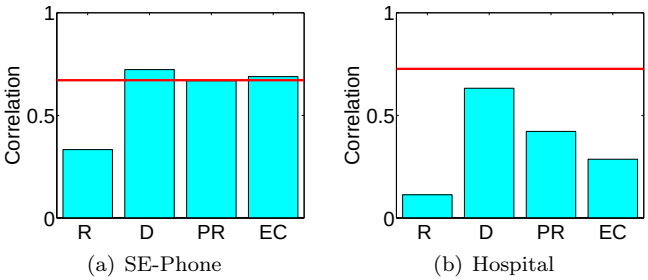
Methodology. In principle, the SSD-UCs computed in Section 6.3.2 captured all possible variations in the structures of the input ensembles. Moreover, by definition, UCs with lower scaled subgraph divergences captured greater variations than those with higher scaled subgraph divergences. Therefore we hypothesized that the UCs with the lowest scaled subgraph divergences would suffice to represent the structural variations present in each ensemble. To test this hypothesis, for each dataset, we chose a structural property (say, density) and computed the values of that property for each network in that dataset, thereby obtaining a vector u. Then we (a) chose a value r of the rank, (b) selected the set of SSD-UCs with the r smallest scaled subgraph divergences, (c) determined the set of nodes in the union of these UCs, (d) computed the subgraphs induced by those nodes in each of the networks in the ensemble, (e) evaluated the density in each subgraph, thereby obtaining another vector v, and (f) computed the Pearson correlation coefficient between u and v. We repeated this procedure for different values of r and plotted the resulting correlations against r (rightmost column of Figure 4). We performed these analyses for SE-Prox, SE-Phone, Hospital, and HEP-PH networks and for each structural property.

Quantitative analysis of sensors. We observed in Figure 4 that the value of correlation often increased with the rank of UC. This trend is expected because as the rank increases, UCs include more nodes in the ensemble. Therefore, the induced subgraphs resemble the networks more and more. We computed the minimal set of UCs for which we obtained a correlation of > 0.5 for three or more of our structural properties. Surprisingly, as highlighted in Table 2, a small number of the highest-ranked SSD-UCs sufficed to achieve this correlation for all datasets. The vertical dotted lines in Figure 4 represent these ranks. These UCs collectively

contained fewer than 30% of the nodes in each dataset. Informally we could "sense" how the structure of the networks varied across the entire dataset by considering only the set of nodes in these SSD-UCs. Henceforth, we will refer to these nodes as *sensors*.

Comparison with baseline methods. We compared the correlations obtained using SSD-UC-based sensors with those obtained from four baseline methods, namely Random (R), Degree (D), PageRank (PR), and Eigenvector-Centrality (EC). Average values of these quantities per unit time have previously been used as features to detect structural anomalies and changes/segments in time-varying graphs [11]. Hence, we intuitively expect nodes with high values of these quantities to act as good sensors. For each dataset, we used each method to select the same number of nodes as in UC-based sensors (Table 2). The "Random" method selected nodes uniformly at random from the set of all nodes. We computed the average correlation across 100 such sets. The other methods selected the nodes with the highest average degree, average PageRank score, and average Eigenvector centrality, respectively, where we computed the average over all graphs in the dataset.

We applied our SSD-UC miner (or, a baseline method) on a dataset to compute a set of nodes, used those nodes to compute the correlation for each structural property, and calculated the average of these correlations. We found that the average correlations obtained by SSD-UC-based sensors were considerably better than (for Hospital networks) or comparable (for SE-Phone networks) to those from any of the baseline methods (Figure 5). The Jaccard indices (JIs) between our sensors and the sets of nodes computed by the other algorithms were not very high (Table 2). These trends, which also held true for the datasets whose results we did not present in this paper, indicated that UC-based sensors achieved superior or comparable correlation to baseline methods and were quite different in composition from the nodes computed by the baselines.



(a) SE-Phone (b) Hospital

Figure 5: The average correlation for the sensors computed by SSD-UCs (red line) compared to the average correlation for the same number of nodes computed using the baseline methods Random (R), Degree (D), PageRank (PR), or Eigenvector Centrality (EC).

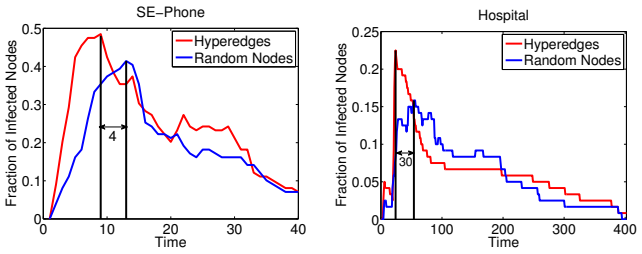
6.3.5 UCs as Epidemiological Monitors To showcase the application of the discovered UCs, we performed experiments using the so-called 'epidemiological monitoring' problem. Given a graph and a contagion spreading on it, can we monitor a subset of nodes to get *ahead* of the overall epidemic (i.e., detect the peak earlier)? This problem is of interest in multiple settings. In public health surveillance, such monitors can provide valuable lead time to react and implement containment policies. Similarly in a computer security scenario, it can provide anti-virus companies lead time to develop solutions. Most current methods for such detection problems typically yield indicators that lag behind the epidemic. Recent work [7] uses the so-called 'Friend-of-Friend' (FOF) approach to select such monitors. After implementing it among the students at Harvard, the authors found that the peak of the daily incidence curve in the monitor set occurred 3.2 days earlier than that of a same-sized random set of students. Intuitively, this implies that if public-health officials monitor this set, they can a get a significant lead time before the outbreaks happen in the *population at large*.

Finding such sets is expensive—more so in the realistic scenario of temporally changing contact networks, which has not been explored before. It is also not straightforward how to extend the FOF approach to dynamic networks as the study assumed static networks. Therefore, we sought to investigate if sets of nodes in UCs can act as good monitors in temporal networks. Our intuition was that rate of epidemic spread is closely related to the largest eigenvalue of a graph [22]. We have shown that the UCs with the smallest scaled subgraph divergences are well-correlated with the change in this quantity. Hence monitoring nodes in these UCs may prove beneficial in tracking the epidemic early. Figure 6 shows the number of infected nodes per unit time (averaged over 100 simulation runs) in two sets of monitors (Random and UCs) in the SE-Phone and Hospital networks under the well-known 'mumps-like' SIR infection model. We chose the set of nodes in the top seven SSD-UCs in the SE-Phone and the top five in the Hospital dataset. We selected an equal number of nodes uniformly at random for the Random set. Clearly, the fraction of infected nodes in the UCs set peaks significantly earlier than the fraction infected in the Random set (the gains are 4 and 30 units of time in Figure 6). This result shows that UC-based sensors may be useful for early detection of the spread of a rumor or a disease.

6.3.6 Qualitative Analysis of SSD-UCs In this section, we interpret the SSD-UCs we identified in SE-Phone dataset. To interpret these SSD-UCs, we

Table 2: Statistics on the sensors identified: smallest rank at which at least three correlations are > 0.5, the value of SSD at that rank, #sensors, %sensors (as a percentage of the nodes in the dataset), and maximum Jaccard Index (JI) between the set of sensors and the same number of nodes computed using a baseline method.

Network	SSD cutoff	#UCs	#Sensors	%Sensors	Max. JI
SE-Phone	0.25	16	21	28%	0.615
Hospital	0.347	1	3	4%	0.200



(a) SE-Phone (b) Hospital

Figure 6: SSD-UCs (Red) monitors lead Random sets (blue) to give valuable lead time. Each point is an average over 100 runs.

used three auxiliary sets of information available in the SocialEvolution dataset, namely the close-friend, socialization, and political-discussant networks. Note that these networks were based on infrequent surveys.

SE-Phone networks. 15 of the top 16 UCs showed some variation among their auxiliary networks. Three of them induced two or more unique subgraphs in each of the three auxiliary networks. The UC with the third lowest scaled subgraph divergence (0.25) contained three students: $\{12, 23, 44\}$. All three students in this SSD-UC discussed politics with each other during October 2008, December 2008, and April–May 2009. Student 44 discussed politics with each of the other two students in Mar 2009 (information about other months are not available). Student 44 socialized with 23 in September 2008. Afterwards, s/he socialized with 23 and 12 separately. Their friendship network was even more dynamic, with three of eight possible subgraphs being observed during the five months for which data is available (Figure 7).

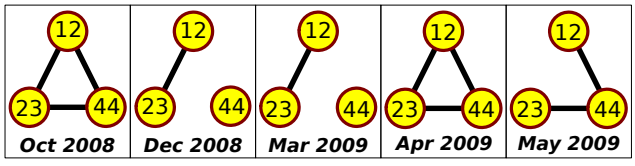


Figure 7: Variation of friendships in an SSD-UC mined from SE-Phone networks. Friendship information for other months are not available.

6.3.7 Efficiency of SSD-UC Miner Our miner was very efficient. (a) For the largest dataset (DBLP), it took less than 14.7 min to mine all 0.6-SSD-UC. On all other datasets, it ran in less than one minute. (b) In the sensor mining application (Section 6.3.4), it was competitive with or faster than the other approaches. It took less time than both PageRank and Eigenvector-Centrality on HEP-PH dataset. It computed 56 sensors from HEP-PH dataset in 23.65 sec whereas Degree, PageRank, Eigenvector-Centrality took 3.28, 98.04, 87.94 sec, respectively to compute the same number of nodes from HEP-PH. All methods ran in less than a minute for the other datasets.

7 Conclusions

We have introduced a novel graph mining problem of discovering unstable communities, i.e., sets of nodes that support highly varying subgraphs in an ensemble of graphs. This problem stands in contrast to the more commonly studied dense subgraph mining problem. We have developed two related concepts—subgraph divergence and scaled subgraph divergence—to quantify these variations. We exploited anti-monotonicity to design levelwise algorithms to enumerate all maximal SD-UCs and SSD-UCs. We applied these algorithms to diverse network ensembles from different domains and demonstrated the usefulness of UCs in summarizing structural variations. We also showed a novel application of UCs as monitors in an epidemiological setting.

There are many avenues for future work such as developing other definitions of UCs, especially those that explicitly consider the temporal order of the networks. Comparing to a distribution other than the uniform may be useful in cases when we know that some classes of subgraphs will not appear in the ensemble. We may also seek to compute the k UCs with the largest SSD/SD or the set of UCs that maximize the correlation with a graph property. Rather than enumerating UCs, as we studied in this paper, we may instead seek to compute diverse sets of UCs or approximations to the largest UC. Thus, we believe our work can inspire further research in this new direction in graph mining.

Acknowledgments This work was supported by grants to TMM from the Environmental Protection Agency (EPA-RD-83499801), the National Science Foundation (DBI-1062380), and the National Institute

of General Medical Sciences of the National Institutes of Health (R01-GM095955), grants to BAP from the National Science Foundation (IIS-1353346), the National Endowment for the Humanities (HG-229283-15), ORNL (Task Order 4000143330) and from the Maryland Procurement Office (H98230-14-C-0127), and a Facebook faculty gift to BAP. Any opinions, findings and conclusions or recommendations express in this material are those of the author(s) and do not necessarily reflect the views of the respective funding agencies.

References

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proc. Very Large Data Bases*, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[2] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: Membership, growth, and evolution. In *Proc. Knowledge Discovery and Data Mining (KDD)*, pages 44–54, New York, NY, USA, 2006. ACM.

[3] A. Battle, M. C. Jonikas, P. Walter, J. S. Weissman, and D. Koller. Automated identification of pathways from quantitative genetic interaction data. *Molecular Systems Biology*, 6(1):379, 2010.

[4] Z. Cai, D. Logothetis, and G. Siganos. Facilitating real-time graph mining. In *Proc. International Workshop on Cloud Data Management (CloudDB)*, pages 1–8, New York, NY, USA, 2012. ACM.

[5] D. Chakrabarti and C. Faloutsos. Graph mining: Laws, generators, and algorithms. *ACM Computing Surveys*, 38(1), 2006.

[6] Y. Chi, X. Song, D. Zhou, K. Hino, and B. L. Tseng. Evolutionary spectral clustering by incorporating temporal smoothness. In *Proc. Knowledge Discovery and Data Mining (KDD)*, pages 153–162, New York, NY, USA, 2007. ACM.

[7] N. Christakis and J. Fowler. Social network sensors for early detection of contagious outbreaks. *PloS one*, 5(9):e12948, 2010.

[8] G. Dong and J. Li. Efficient mining of emerging patterns: Discovering trends and differences. In *Proc. Knowledge Discovery and Data Mining (KDD)*, pages 43–52, New York, NY, USA, 1999. ACM.

[9] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. *SIGCOMM Comput. Commun. Rev.*, 29(4):251–262, 1999.

[10] F. Geerts, H. Mannila, and E. Terzi. Relational link-based ranking. In *Proc. Very Large Data Bases (VLDB)*, pages 552–563. VLDB Endowment, 2004.

[11] K. Henderson, T. Eliassi-Rad, C. Faloutsos, L. Akoglu, L. Li, K. Maruhashi, B. A. Prakash, and H. Tong. Metric forensics: a multi-level approach for mining volatile graphs. In *Proc. Knowledge Discovery and Data Mining (KDD)*, pages 163–172. ACM, 2010.

[12] M. Jha, C. Seshadhri, and A. Pinar. A space efficient streaming algorithm for triangle counting using the birthday paradox. In *Proc. Knowledge Discovery and Data Mining (KDD)*, pages 589–597, New York, NY, USA, 2013. ACM.

[13] R. Jin, C. Wang, D. Polshakov, S. Parthasarathy, and G. Agrawal. Discovering frequent topological structures from graph datasets. In *Proc. Knowledge Discovery in Data Mining (KDD)*, pages 606–611, New York, NY, USA, 2005. ACM.

[14] R. Jin, C. Wang, D. Polshakov, S. Parthasarathy, and G. Agrawal. Discovering frequent topological structures from graph datasets. In *Proc. Knowledge Discovery in Data Mining (KDD)*, pages 606–611, New York, NY, USA, 2005. ACM.

[15] G. Karypis and V. Kumar. Multilevel algorithms for multi-constraint graph partitioning. In *Proc. ACM/IEEE Conference on Supercomputing*, pages 1–13, Washington, DC, USA, 1998. IEEE Computer Society.

[16] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *Proc. Knowledge Discovery and Data Mining (KDD)*, pages 137–146, New York, NY, USA, 2003. ACM.

[17] Y. Koren. Collaborative filtering with temporal dynamics. *Commun. ACM*, 53(4):89–97, Apr. 2010.

[18] R. Kumar, J. Novak, P. Raghavan, and A. Tomkins. On the bursty evolution of blogspace. In *Proc. International Conference on World Wide Web (WWW)*, pages 568–576, New York, NY, USA, 2003. ACM.

[19] W. Li, C. Liu, T. Zhang, H. Li, M. Waterman, and X. Zhou. Integrative analysis of many weighted co-expression networks using tensor computation. *PLoS Computational Biology*, 7(6):e1001106, 2011.

[20] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *Proc. International Conference on Information and Knowledge Management (CIKM)*, pages 556–559, New York, NY, USA, 2003. ACM.

[21] S. Milgram. The small world problem. *Psychology today*, 2(1):60–67, 1967.

[22] B. A. Prakash, D. Chakrabarti, M. Faloutsos, N. Valler, and C. Faloutsos. Threshold conditions for arbitrary cascade models on arbitrary networks. In *Proc. International Conference on Data Mining (ICDM)*, pages 537–546, Washington, DC, USA, 2011. IEEE Computer Society.

[23] A. Rahman, C. Estep, T. M. Murali, C. L. Poirel, and D. J. Badger. Reverse Engineering Molecular Hypergraphs. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 10(5):1113–1124, 2013.

[24] J. Sun, D. Tao, and C. Faloutsos. Beyond streams and graphs: Dynamic tensor analysis. In *Proc. Knowledge Discovery and Data Mining (KDD)*, pages 374–383, New York, NY, USA, 2006. ACM.

[25] X. Yan, H. Cheng, J. Han, and P. S. Yu. Mining significant graph patterns by leap search. In *Proc. ACM SIGMOD International Conference on Management of Data*, pages 433–444, New York, NY, USA, 2008. ACM.