An Event-based Framework for Characterizing the Evolutionary Behavior of Interaction Graphs

Sitaram Asur, Srinivasan Parthasarathy, and Duygu Ucar
Department of Computer Science and Engineering
Ohio State University
{srini}@cse.ohio-state.edu

ABSTRACT

Interaction graphs are ubiquitous in many fields such as bioinformatics, sociology and physical sciences. There have been many studies in the literature targeted at studying and mining these graphs. However, almost all of them have studied these graphs from a static point of view. The study of the evolution of these graphs over time can provide tremendous insight on the behavior of entities, communities and the flow of information among them. In this work, we present an event-based characterization of critical behavioral patterns for temporally varying interaction graphs. We use non-overlapping snapshots of interaction graphs and develop a framework for capturing and identifying interesting events from them. We use these events to characterize complex behavioral patterns of individuals and communities over time. We demonstrate the application of behavioral patterns for the purposes of modeling evolution, link prediction and influence maximization. Finally, we present a diffusion model for evolving networks, based on our framework.

1. INTRODUCTION

Many social and biological systems can be represented as complex interaction networks where nodes represent entities (i.e. individuals, proteins) and edges mimic the interactions among them. These interaction networks arise from a wide variety of scientific domains like computer science, physics, biology and sociology. Online communities such as Flickr, MySpace and Orkut, e-mail networks, co-authorship networks and WWW networks are examples of interesting interaction networks. The study of these complex interaction networks can provide insight into their structure, properties and behavior.

Early research in this area [13, 24, 4, 5] has primarily focused on the static properties of these networks, neglecting the fact that most real-world interaction networks are dynamic in nature. In reality, many of these networks constantly evolve over time, with the addition and deletion of edges and nodes representing changes in the interactions among the modeled entities. Recently, there has been some interest in studying dynamic graphs [19, 3, 6, 10]. Identifying the portions of the network that are changing, characterizing the type of change, predicting future events (e.g. link prediction), and developing generic models for evolving networks are challenges that need to be addressed. For instance, the rapid growth of online communities has dictated the need for analyzing large amounts of temporal data to reveal community structure, dynamics and evolution.

Interaction networks are often modular in nature. The interactions existing between nodes can be used to group them into clusters or communities. For instance, in a social network, these clusters represent people with similar contact patterns or interests. The problem of identifying these clusters or communities from static graphs has been extensively studied [13, 7, 25, 11] over the past decade. However, in the case of evolving graphs, the clusters are typically not static. Instead they constantly change over time, as the network evolves. We believe that studying the evolution of these clusters, in particular their formation, transitions and dissolution, can be extremely useful for effectively characterizing the corresponding changes to the network over time.

Another important aspect is the behavior of the nodes of the network. Nodes of an evolving interaction network represent entities whose interaction patterns change over time. The movement of nodes, their behavior and influence over other nodes, can help make inferences regarding future interactions as well as predicting changes to communities in the network. For instance, in a social network, if a person is very sociable, the chances of him/her interacting with new people and joining new groups is very high. In the case of a collaboration network, if a person is known to collaborate frequently with different people, then the chance of a new collaboration involving this person is high. The influence exerted by a node can be studied in terms of its effect on other nodes. If several other people join a community when a particular individual does, it indicates a high degree of positive influence for that person.

The study of diffusion or flow of information in an evolving network is important for social science research, viral marketing applications and epidemiology. For instance, pandemic viruses pose a severe threat to society due to their potential to spread rapidly and cause tremendous widespread illnesses and deaths. In viral marketing, the goal is to propagate an idea or innovation through an interaction network. Analysis of the evolution of interactions in a social network and the identification of influential nodes can be used to devise effective containment policies for pandemic disease

.

spread in the case of epidemiological studies, as well as "word-of-mouth" advertising in marketing.

In this paper, we provide an event-based framework for characterizing the evolution of interaction networks. We begin by converting an evolving graph into static snapshot graphs at different time points. We obtain clusters at each of these snapshots independently. Next, we characterize the transformations of these clusters by defining and identifying certain critical events. We define efficient incremental algorithms involving bit-matrix computations for this purpose. We use these critical events to compute and reason about novel behavior-oriented measures, which offer new and interesting insights for the characterization of dynamic behavior of interaction graphs. We illustrate our framework on two different evolving networks - the DBLP co-authorship network and a clinical trials patient network. In each case, the behavioral patterns that we discover using our framework help us make useful inferences about cluster evolution and link prediction. For the DBLP dataset, we use the patterns for predicting future trends (link prediction) with good success. In the case of the clinical trials network, we show how the behavioral patterns we discover can help detect signs of hepatoxic side-effects for a particular drug. Finally, we use the behavioral measures to detail a diffusion model for evolving networks and demonstrate their application for the task of influence maximization.
In short, the key contributions of this work are

- The identification of key critical events that occur in evolving interaction networks.
- Efficient incremental algorithms for the discovery of these critical events
- Novel behavioral measures for stability, sociability, influence and popularity that can be computed incrementally over time
- A diffusion model for evolving networks based on our framework
- Application of the events and behavioral measures on two real datasets for modeling evolution, predicting behavior and trends (link prediction) and influence maximization.

2. RELATED WORK

There has been enormous interest in mining interaction graphs for interesting patterns in various domains. However, the majority of these studies [13, 24, 4, 5, 7, 25, 11] have focused on mining static graphs to identify community structures, patterns and novel information. Recently, the dynamic behavior of clusters and communities have attracted the interest of several groups. Leskovec *et al* [19] studied the evolution of graphs based on various topological properties, such as the degree distribution and small-world properties of large networks. They proposed a graph generation model, called *Forest Fire* model, to explain their findings about evolutionary behaviors of graphs. Backstrom *et al* [3] studied formation of groups and the ways they grow and evolve over time. To estimate probability of an individual joining a community, they proposed using features of communities and individuals, applying decision-tree techniques. To identify communities that are likely to grow, they also used community features on a decision-tree based analysis.

Chakrabarti *et al* [6] proposed evolutionary settings for two widely-used clustering algorithms (k-means and agglomerative hierarchical clustering). They define evolutionary clustering as the task of incrementally obtaining high-quality clusters for a set of objects while also maintaining similarity with clusters identified in previous timestamps. To obtain the clusters for a particular snapshot, they also use history information to obtain a clustering consistent with earlier snapshots. Falkowski *et al* [10] analyze the evolution of communities that are stable or fluctuating based on subgroups. Although they analyze interaction graphs, their focus is different from ours. They examine overlapping snapshots of interaction graphs and apply standard statistical measures to identify persistent subgroups. Our focus is on identifying key events and behavioral patterns that can characterize, model and predict future behavioral trends. In this regard, we specifically target nodes of the network and analyze their evolutionary behavior.

The seminal paper by Samtaney *et al* [28] described an approach for extracting coherent regions from 2-dimensional and 3-dimensional scalar and vector fields for tracking purposes. To study the evolution of these regions over time, they present certain evolutionary events for objects. Event-based methods have also been applied on spatial data [33] and clustered stream data [29]. Although they use events, they do not deal with evolving graphs.

The use of Semantic similarity based on ontologies has been studied many times in the past [21, 12]. It has been successfully applied on various taxonomies such as WordNet [27] and the Gene Ontology [8]. Resnik [26] suggested a novel way to evaluate semantic similarity in an ontology based on notion of information content. In our work, semantic similarity concepts are used to quantify the similarity between authors and clusters.

3. PROBLEM DEFINITION

Before describing our event-based framework in detail, we introduce the basic notations used throughout the paper. As we mentioned earlier, our focus in this work is to study the evolution of graphs, in particular to understand behavioral patterns for communities and individuals over time. In order to fully understand the temporal evolution of graphs, it becomes necessary to study and characterize the transformations undergone by the graph at different time instants along the way. In this regard, we make use of temporal snapshots to examine static versions of the evolving network at different time points.

Definition: An interaction graph G is said to be evolving if its interactions vary over time. Let $G = (V, E)$ denote a temporally varying interaction graph where V represents the total unique entities and E the total interactions that exist among the entities. We define a temporal snapshot $S_i = (V_i, E_i)$ of G to be a graph representing only entities and interactions active in a particular time interval $[T_{s_i}, T_{e_i}]$, called the snapshot interval.

As the graph evolves, new nodes and edges can appear. Similarly, nodes and edges can also cease to exist. This dynamic behavior of a graph over time can thus be represented as a set of S equal, non-overlapping temporal snapshots.

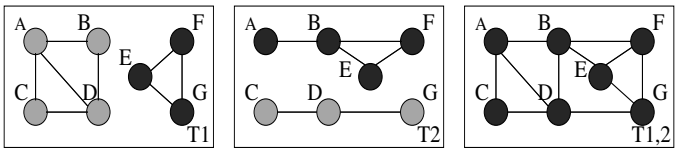

Figure 1: Temporal Snapshots a) at Time t=1 b) at Time t=2 c) Cumulative snapshot at Time t=2

Note that, different snapshots are mutually exclusive. They do not contain any information in common. This is in contrast to the representation provided in some earlier research [6, 22] which define a snapshot considering the entire set of interactions upto the current time interval. Figure 1(a-b) illustrates an example evolving graph over two time intervals. We find that in the first time interval, interactions exist between A and C, and between A and D. In the second time interval, these interactions do not continue to exist. Figure 1(c) depicts a cumulative snapshot of the second time interval. We find that the information regarding the loss of interactions AC and AD is lost. Also, the community structure depicted in Figure 1(c) does not reflect the actual structure. To prevent this loss of information, we choose short time intervals and generate snapshots representing only the information of that specific interval. The collection of all T temporal snapshots is represented by $S = \{S_1, S_2, \ldots, S_T\}$.

To study the evolution of the graph, we need a representation of its structure at different snapshots. For this purpose, we generate clusters for each snapshot graph. Each S_i is partitioned into k_i communities or clusters denoted by $C_i = \{C_i^1, C_i^2, \ldots, C_i^{k_i}\}$. The j^{th} cluster of S_i, C_i^j is also a graph denoted by (V_i^j, E_i^j) where V_i^j are nodes in S_i^j and E_i^j denotes the edges between nodes in V_i^j. Finally, for each $S_i = (V_i, E_i)$, $V_i^1 \cup V_i^2 \cup \ldots \cup V_i^{k_i} = V_i$.

To choose a clustering algorithm for this work, we examined the performance of various graph clustering algorithms on several interaction graphs in terms of modularity of clusters. We found that the MCL algorithm [31], a fast and scalable unsupervised clustering algorithm, consistently yielded clusters of high modularity. Hence, we use MCL to obtain the clusters at different timestamps [1]. The MCL algorithm does not require a parameter specifying the number of clusters. Instead it uses a granularity parameter and the cluster structure prevalent in the graph to determine the number of partitions. Accordingly, for each snapshot, the number of clusters may vary depending on the interactions in that time interval. We used a granularity parameter of 1.2 for our experiments, since the graphs were fairly sparse.

Algorithm 1 shows the outline of the framework we propose. We design an *incremental strategy* to mine the clusters over time to identify significant changes that occur among snapshots, referred to as *critical events*. These events are then used to study more complex behavioral patterns. In Section 5, we will describe the critical events and how we find them. In Section 6, we mine these events further to find complex behavioral patterns for analysis.

4. DATASETS

We employ two different datasets in this work.

[1]Note that, the event-based framework we propose is relatively independent of the clustering algorithm used to obtain the snapshot clusters.

Algorithm 1 Mine-Events(G,T)

Input: Interaction graph $G = (V, E)$ and T, the number of intervals
Convert graph $G = (V, E)$ into T temporal snapshots $S = \{S_1, S_2, \ldots, S_T\}$.
for $i = 1$ to T **do**
  Cluster S_i
  $C_i = \{C_i^1, C_i^2, \ldots, C_i^{k_i}\}$
end for
for $i = 1$ to $T - 1$ **do**
  $Events$ = Find_events(S_i,S_{i+1}) [Section 5]
  Mine $Events$ for complex patterns [Section 6]
end for

DBLP co-authorship network: The DBLP bibliography maintains information on more than 800000 computer science publications. We used the DBLP data to generate a co-authorship network representing authors publishing in several important conferences in the field of databases, data mining and AI. We chose all papers over a 10 year period (1997-2006) that appeared in 28 key conferences spanning mainly these three areas. We converted this data into a co-authorship graph, where each author is represented as a node and an edge between two authors corresponds to a joint publication by these two authors. The graph spanning 10 years contained 23136 nodes and 54989 edges. We chose the snapshot interval to be a year, resulting in 10 consecutive snapshot graphs. These graphs are then clustered and analyzed to identify critical events and patterns. We believe that studying the evolution of the DBLP dataset can afford information about the nature of collaborations and the factors that influence future collaborations between authors.

Clinical Trials Data: In clinical trials, pharmaceutical companies test a new drug for efficacy and toxicity - efficacy to evaluate its effectiveness in curing or controlling the disease in question and toxicity to determine if the drug is safe for consumption and with minimal side effects. Releasing a drug that turns out to be toxic can cost companies billions of dollars and more importantly lead to loss in life. In this paper we use a dataset obtained from a major pharmaceutical company, consisting of both healthy people as well as patients suffering from certain diseases(diabetes and hepatic impairment). As part of the study, they were given either a placebo (a formulation that includes only the inactive ingredients) or the drug under study. Liver toxicity information can be obtained from eight serum analytes (often referred to in the literature as the liver panel) This dataset consisted of patients on the placebo and the drug in the ratio 40:60. The initial snapshot of this data is composed of the measurements of the analytes obtained before patients were treated with the drug or the placebo. The subsequent snapshots correspond to measurements taken every week. The data thus consisted of 7 snapshots spanning a 6 week period since the beginning of the treatment. We transformed the data for each snapshot into a graph, based on the correlations that exist between the analyte values of patients. If there exists a high correlation (greater than a threshold T_{corr}) in the analyte values between two patients, the two patients have an edge between them in the snapshot graph [2]. Note that, if we consider each patient separately, we are limited to only intrinsic information, whereas by modeling patients

[2]We examined the distribution of correlations and picked a T_{corr} value of 0.7 for our experiments

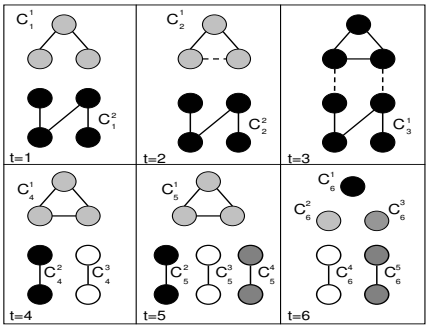

Figure 2: Temporal Snapshots at time t=1 to 6

as a graph, we are able to utilize intrinsic as well as extrinsic properties.

5. CRITICAL EVENTS

In this section, we introduce and afford a formal definition to certain critical events that occur in evolving graphs. Some of the critical events described in this section are inspired by a similar notion described by Samtaney *et al* [28]. in the context of tracking and visualizing features. They have also been used since, for tracking spatial objects [33] and clustered text streams [29].

The events that we define are primarily between two consecutive timestamps but it is possible to coalesce events from contiguous timestamps by analyzing the meta-data collected from the event mining framework. We proceed to use these events in the later sections to define more complex behavior. We distribute the critical events which graphs can undergo into two categories - events involving communities and events involving individuals.

Figure 2 displays a set of snapshots of the network which will be used as a running example in this section. At time $t = 1$, 2 clusters are discovered (shown in different colors).

Events involving communities: We define 5 basic events which clusters can undergo between any two consecutive time intervals or steps. Let S_i and S_{i+1} be snapshots of S at two consecutive time intervals with C_i and C_{i+1} denoting the set of clusters respectively. The five proposed events are:

1) Continue: A cluster C^j_{i+1} is marked as continuation of C^k_i if V^j_{i+1} is the **same** as V^k_i. We do not impose the constraint that the edge sets should be the same.

$Continue(C^k_i, C^j_{i+1}) = 1$ iff $V^k_i = V^j_{i+1}$

The main motivation behind this is that if certain nodes are always part of the same cluster, any information supplied to one node will eventually reach the others. Therefore, as long as the vertex set remains same, the information flow is not hindered. The addition and deletion of edges merely indicates the strength between the nodes. An example of a continue event is shown at t=2 in Figure 2. Note that an extra interaction appears between the nodes in Cluster C^1_2 but the clusters do not change.

2) κ-Merge: Two different clusters C^k_i and C^l_i are marked as merged if there exists a cluster in the next timestamp that contains at least κ% of the nodes belonging to these two clusters. The essential condition for a merge is :

$Merge(C^k_i, C^l_i, \kappa) = 1$ iff $\exists C^j_{i+1}$ such that

$$\frac{|(V^k_i \cup V^l_i) \cap V^j_{i+1}|}{Max(|V^k_i \cup V^l_i|, |V^j_{i+1}|)} > \kappa\% \quad (1)$$

and $|V^k_i \cap V^j_{i+1}| > \frac{|C^k_i|}{2}$ and $|V^l_i \cap V^j_{i+1}| > \frac{|C^l_i|}{2}$. This condition will only hold if there exist edges between V^k_i and V^l_i in timestamp $i + 1$. Intuitively, it implies that new interactions have been created between nodes which previously were part of different clusters. This caused κ% [3] of nodes in the two original clusters to join the new cluster. Note that, in an ideal or complete merge, with $\kappa = 100$, all nodes in the two original clusters are found in the same cluster in the next timestamp. The two original clusters are completely lost in this scenario. Figure 2 shows an example of a complete merge event at t=3. The dotted lines represent the newly created edges. All the nodes now belong to a single cluster (C^1_3).

3) κ-Split: A single cluster C^j_i is marked as split if κ% of nodes from this cluster are present in 2 different clusters in the next timestamp. The essential condition is that:

$Split(C^j_i, \kappa) = 1$ iff $\exists C^k_{i+1}, C^l_{i+1}$ such that

$$\frac{|(V^k_{i+1} \cup V^l_{i+1}) \cap V^j_i|}{Max(|V^k_{i+1} \cup V^l_{i+1}|, |V^j_i|)} > \kappa\% \quad (2)$$

and $|V^k_{i+1} \cap V^j_i| > \frac{|C^k_{i+1}|}{2}$, $|V^l_{i+1} \cap V^j_i| > \frac{|C^l_{i+1}|}{2}$.

Intuitively, a split signifies that the interactions between certain nodes are broken and not carried over to the current timestamp, causing the nodes to part ways and join different clusters. Also note that a broken edge, by itself, does not necessarily indicate a split event, as there may be other interactions existing between vertices in the cluster (similar to the notion of k-connectivity). Time t=4 in Figure 2 shows a split event when a cluster gets completely split into three smaller clusters.

4) Form:A new cluster C^k_{i+1} is said to have been formed if none of the nodes in the cluster were grouped together at the previous time interval i.e. no 2 nodes in V^k_{i+1} existed in the same cluster at time period i.

$Form(C^k_{i+1}) = 1$ iff $\exists$ no C^j_i such that $V^k_{i+1} \cap V^j_i > 1$

Intuitively, a form indicates the creation of a new community or new collaboration. Figure 2 at time t=5 shows a form event when two new nodes appear and a new cluster is formed.

5) Dissolve: A single cluster C^k_i is said to have dissolved if none of the vertices in the cluster are in the same cluster in the next timestamp i.e. no two entities in the original cluster have an interaction between them in the current time interval.

$Dissolve(C^k_i) = 1$ iff $\exists$ no C^j_{i+1} such that $V^k_i \cap V^j_{i+1} > 1$

Intuitively, a dissolve indicates the lack of contact or interactions between a group of nodes in a particular time period. This might signify the breakup of a community or a workgroup. Figure 2 at time t=6 shows a dissolve event when there are no longer interactions between the three nodes in Cluster C^1_5 resulting in a breakup of the cluster into 3 clus-

[3]We used a κ value of 50 in our experiments.

ters - C_6^1, C_6^2 and C_6^3.

Events involving individuals: We wish to analyze not only the evolution of communities but the influence of the behavior of individuals on communities. In this regard, we introduce four basic transformations involving individuals over snapshots.

1) Appear: A node is said to appear when it occurs in C_i^j but was not present in any cluster in the earlier timestamp.
$Appear(v,i) = 1$ iff $v \notin V_{i-1}$ and $v \in V_i$
This simple event indicates the introduction of a person (new or returning) to a network. In Figure 2, at time $t = 5$ two new nodes appear in the network.

2) Disappear: A node is said to disappear when it was found in a cluster C_{i-1}^j but is not present in any cluster in the timestamp i.
$Disappear(v,i) = 1$ iff $v \in V_{i-1}$ and $v \notin V_i$
This indicates the departure of a person from a network. In Figure 2, at time $t = 6$ two nodes of cluster C_5^4 disappear from the network.

3) Join: A node is said to join cluster C_i^j if it exists in the cluster at timestamp i. This may be due to an *Appear* event or due to a leave event from a different cluster. Note that in case, the cluster C_i^j must be sufficiently similar to a cluster C_{i-1}^k

$Join(v, C_i^j) = 1$ iff $\exists C_i^j$ and C_{i-1}^k such that $C_{i-1}^k \cap C_i^j > \frac{|C_{i-1}^k|}{2}$ and $v \notin V_{i-1}^k$ and $v \in V_i^j$

The cluster similarity condition ensures that C_i^j is not a newly formed cluster. This condition differentiates a *Join* event from a *Form* event. Nodes forming a new cluster will not be considered to be *Join* events since there will be no cluster C_{i-1}^k in the previous timestamp with similarity $> \frac{|C_{i-1}^k|}{2}$ with the newly formed cluster.

4) Leave: A node is said to leave cluster C_{i-1}^k if it no longer is present in a cluster with most of the nodes in V_{i-1}^k. A node that leaves a cluster may leave the network as a *Disappear* event or may join a different cluster. In a collaboration network, a *Leave* event might correspond to a student graduating and leaving a group.

$Leave(v, C_i^j) = 1$ iff $\exists C_i^j$ and C_{i-1}^k such that $C_{i-1}^k \cap C_i^j > \frac{|C_{i-1}^k|}{2}$ and $v \in V_{i-1}^k$ and $v \notin V_i^j$

The similarity constraint between the two clusters is used to maintain cluster correspondence. Note that if the original cluster dissolves, the nodes in the cluster are not said to participate in a *Leave* event. This is due to the fact that there will no longer be a cluster with similarity $> \frac{|C_{i-1}^k|}{2}$ with the dissolved cluster C_{i-1}^k.

5.0.1 Algorithms for Event Extraction:

We leverage the use of efficient bit matrix operations to compute the events between snapshots. First, for each temporal snapshot, we construct a binary $k_i \times n$ matrix T_i where k_i is the number of clusters at timestamp i and n is the number of nodes. We then compare the matrices of successive snapshots to find events between them [4]. Let $T_i(x,:)$ and $T_i(:,y)$ correspond to the x^{th} row and y^{th} column vector of matrix T_i respectively. To compute all the events between two snapshots, we perform a set of binary operations (AND and OR) on the corresponding matrices. The linear operations performed to identify each event are presented below. Let $|x|_1$ represent the L^1-*norm* of a binary vector x.

$$|x|_1 = \sum_{i=1}^{|x|} x_i \tag{3}$$

We can compute the events as :

Dissolve(T_i,T_{i+1}) = $\{x|1 \le x \le k_i, \arg\max_{1 \le y \le k_{i+1}} (|AND(T_i(x,:), T_{i+1}(y,:))|_1 \le 1\}$

Form(T_i,T_{i+1}) = Dissolve(T_{i+1},T_i)

Merge(T_i,T_{i+1},κ) = $\{< x,y,z > |1 \le x \le k_i,$
$1 \le y \le k_i, x \ne y, 1 \le z \le k_{i+1},$
$|AND(OR(T_i(x,:), T_i(y,:)), T_{i+1}(z,:))|_1 \ge \kappa,$
$|AND(T_i(x,:), T_{i+1}(z,:))|_1 \ge \frac{|T_i(x,:)|_1}{2},$
$|AND(T_i(y,:), T_{i+1}(z,:))|_1 \ge \frac{|T_i(y,:)|_1}{2}\}$

Split(T_i,T_{i+1},κ)= Merge(T_{i+1},T_i,κ)

Continue(T_i,T_{i+1})= $\{< x,y > |1 \le x \le k_i, 1 \le y \le k_{i+1},$
$OR(T_i(x,:), T_{i+1}(y,:)) == AND(T_i(x,:), T_{i+1}(y,:))\}$

Appear(T_i,T_{i+1}) = $\{v|1 \le v \le |V|,$
$|T_i(:,v)|_1 == 0, |T_{i+1}(:,v)|_1 == 1\}$

Disappear(T_i,T_{i+1}) = $\{v|1 \le v \le |V|,$
$|T_i(:,v)|_1 == 1, |T_{i+1}(:,v)|_1 == 0\}$

Join(T_i,T_{i+1}) = $\{< y,v > |1 \le y \le k_{i+1}, 1 \le v \le |V|, T_{i+1}(y,v) == 1,$
$\exists x, 1 \le x \le k_i$ s.t. $|AND(T_i(x,:), T_{i+1}(y,:))|_1 > \frac{|T_i(x,:)|_1}{2},$
$T_i(x,v) == 0\}$

Leave(T_i,T_{i+1})= $\{< x,v > |1 \le x \le k_i, 1 \le v \le |V|, T_i(x,v) == 1,$
$\exists y, 1 \le y \le k_{i+1}$ s.t. $|AND(T_i(x,:), T_{i+1}(y,:))|_1 > \frac{|T_i(x,:)|_1}{2},$
$T_{i+1}(y,v) == 0\}$

$T_i(x,y)$ represents the value in the x^{th} row and y^{th} column of T_i. The construction of the matrices and the operations to find the events are all linear in time complexity(O(n)), assuming that $k_i << n$ and $k_{i+1} << n$. The timing results for event detection for various values of n are shown in Table 1. The number of clusters, k_i and k_{i+1} are 50 in each case. The advantage of using the bit matrix operations is that they enable us to leverage GPU [14] and multi-core [17] architectures quite efficiently. Note that, since we are computing events for two timestamps at a time, the whole event detection process can be trivially parallelized.

Number of nodes	Time (secs)
1000	0.006507
10000	0.0721
100000	0.714485
1000000	7.987603
2000000	10.435108
4000000	20.8885

Table 1: Timing Results

6. BEHAVIORAL ANALYSIS

[4]If the number of nodes changes between the timestamps, we will increase the length of the matrices to reflect the largest of the two

Next, we use the critical events to study important behavioral patterns in evolving graphs. Most of the research on evolving networks [3, 10] have focused solely on analyzing community behavior. In this section, we begin by presenting some interesting results on community behavior and then move on to study the *behavior of nodes in the network* and their influence on others. Finally, we incorporate some simple semantic information and study the information flow among nodes and communities.

6.1 Community Behavioral Analysis

The analysis of interaction graphs over time enables us to infer the evolution of group behavior over time. By analyzing the community-based events obtained, we observed several interesting merge and split events in the DBLP dataset, that afforded insight into interesting relationships between group collaborations as well as the evolution of topics.

6.1.1 Group Merge:

In the DBLP dataset, a group merge corresponds to a collaboration between members of two or more groups from the previous time period. This suggests that the resultant merger represents a confluence of ideas or topics. Note that, more than two clusters can merge together, but our algorithm will discover this event as a set of two-way merge events. For instance, let us consider a cluster merge event that occurred in the 2005-2006 time interval. Our algorithm identified two groups (one from Germany and one from Italy) who independently published articles in different conferences in 2005.

Cluster 1 in 2005

AAAI 2005: **Niels Landwehr, Kristian Kersting, Luc De Raedt:** *nFOIL: Integrating Nave Bayes and FOIL*

AAAI 2005: **Luc De Raedt, Kristian Kersting, Sunna Torge:** *Towards Learning Stochastic Logic Programs from Proof-Banks.*

Cluster 2 in 2005

ICML 2005 : **Sauro Menchetti, Fabrizio Costa, Paolo Frasconi:** *Weighted Decomposition Kernels.*

IJCAI 2005 : **Andrea Passerini and Paolo Frasconi:** *P. Kernels on Prolog Ground Terms.*

Merged Cluster in 2006

ILP 2006 : **Niels Landwehr, Andrea Passerini, Luc De Raedt, Paolo Frasconi:** *kFOIL: Learning Simple Relational Kernels*

From the merge event, we can hypothesize that Niels Landwehr and Luc De Raedt, who were working on Inductive Logic in 2005 are collaborating on Passerini and Frasconi who worked separately on kernels and the resultant paper is a combination of these ideas.

Indeed, in the abstract of the 2006 paper, the authors describe the paper as *"A novel and simple combination of inductive logic programming with kernel methods is presented. The kFOIL algorithm integrates the well-known inductive logic programming system FOIL with kernel methods."*

One relatively simple conclusion we could make from our observations is that the propensity of a merger between clusters seems to be dependent on two main factors - the *proximity or sociability of the authors* and the *similarity of the topics* of the papers involved. We discuss more on these two factors in the next two subsections.

6.1.2 Group Split:

Next, let us consider a split event that occurred in the same time period. Our algorithm found a cluster consisting of papers on structure extraction from HTML and unstructured documents.

Cluster in 1998:

FODO 1998: **Seung Jin Lim, Yiu-Kai Ng:** *Constructing Hierarchical Information Structures of Sub-Page Level HTML Documents*

ER 1998: **David W. Embley, Douglas M. Campbell, Y. S. Jiang, Stephen W. Liddle, Yiu-Kai Ng, Dallan Quass, Randy D. Smith:** *A Conceptual-Modeling Approach to Extracting Data from the Web.*

IDEAS 1998: **Aparna Seetharaman, Yiu-Kai Ng:** *A Model-Forest Based Horizontal Fragmentation Approach for Disjunctive Deductive Databases*

CIKM 1998: **David W. Embley, Douglas M. Campbell, Randy D. Smith, Stephen W. Liddle:** *Ontology-Based Extraction and Structuring of Information from Data-Rich Unstructured Documents*

In the next year (1999), this cluster splits into two different clusters. While Seung Jin Lim, Yiu-Kai Ng and David W. Embley continue working on extracting information from Web Documents, Stephen W. Liddle, Douglas M. Campbell, Chad Crawford specialized on Business Reports.

Cluster 1 in 1999

CIKM 1999: **Seung Jin Lim, Yiu-Kai Ng:** *An Automated Approach for Retrieving Hierarchical Data from HTML Tables.*

DASFAA 1999: **Seung Jin Lim, Yiu-Kai Ng:** *WebView: A Tool for Retrieving Internal Structures and Extracting Information from HTML Documents*

SIGMOD 1999: **David W. Embley, Y. S. Jiang, Yiu-Kai Ng:** *Record-Boundary Discovery in Web Documents.*

Cluster 2 in 1999

CIKM 1999: **Stephen W. Liddle, Douglas M. Campbell, Chad Crawford:** *Automatically Extracting Structure and Data from Business Reports.*

An important reason for split events is the divergence of topics, as we can observe in the above example.

We find that, by examining the merge and split events, we can gain insight into interesting relationships between group collaborations as well as the evolution of topics, which we will examine in more detail in the last subsection.

6.2 Movement-based Analysis

Movement for individuals is defined using the basic events - Join and Leave. We use these basic events to identify more complex behavior. In particular, we are interested in capturing the behavioral tendencies of individuals that contribute to the evolution of the graph. We wish to use these behavioral patterns to perform reasoning and predict future trends of the graph. We define four behavioral measures that can be *incrementally computed* at each time interval using the events discovered in the current interval.

6.2.1 Stability Index

The Stability index measures the tendency of a node to have interactions with the same nodes over a period of time. A node is highly stable if it belongs to a very stable cluster, one that does not change much over time. Let $cl_i(x)$ represent the cluster that node x belongs to in the i^{th} time interval. The Stability Index (SI) for node x over T times-

Disease	Treat-ment	Age	Sex
diabetes +/- renal impairment	Drug	62	M
diabetes +/- renal impairment	Drug	59	M
hepatic impairment	Drug	56	M
diabetic neuropathy	Drug	66	F
diabetes +/- renal impairment	Drug	60	M
diabetes +/- renal impairment	Drug	62	F
diabetes +/- renal impairment	Drug	70	F
diabetes +/- renal impairment	Drug	66	M
diabetes +/- renal impairment	Drug	55	M
diabetes +/- renal impairment	Drug	50	M
diabetes +/- renal impairment	Drug	49	M
hepatic impairment	Drug	50	M
diabetic neuropathy	Drug	69	M
diabetes mellitus (type 2 niddm)	Drug	52	M
hepatic impairment	Drug	48	M
hepatic impairment	Drug	48	M
diabetes +/- renal impairment	Drug	49	M
hepatic impairment	Drug	49	M
diabetes mellitus (type 2 niddm)	Placebo	56	M

Table 2: Low Stability Index - Clinical Trials Data

tamps is measured incrementally as:

$$SI(x,T) = \sum_{i=1}^{T} \frac{|cl_i(x)|}{\sum_{j=1}^{V_i}(Leave(j, cl_i(x)) + Join(j, cl_i(x)))} \quad (4)$$

Stability for the Clinical Trials Dataset: In the case of the clinical trials data, nodes in a cluster correspond to individuals having similar observations. When a node has a low Stability index score, it indicates that the observations of that particular patient fluctuate appreciably. This causes the node to jump from one cluster to another repeatedly. This behavior represents an anomaly and can indicate possible side-effects of the drug being administered. Note that the dataset contains two groups of people, one group on the placebo and the other group on the drug with a distribution of 40:60. If the people with very low Stability index (outliers in this case) happen to be people on the drug, there is a reasonable indication that there may be a hepatoxic effect from the drug intake, whereas if it is uniform over both sets of people, it would indicate there are no noticeable side-effects.

Accordingly, we computed the Stability index for all the nodes in the clinical trial data. On examination, we found 19 nodes having very low Stability index scores (below a threshold). This indicates that these nodes move between clusters in almost every time interval that they are active. This suggests a significantly unstable behavior exhibited by these nodes. The 19 patients are shown in Table 2.

In this particular application unstable nodes (patients) are a cause for concern since that may be an indication of toxicity. Drilling down on the nineteen most unstable nodes we find that only one of them is on the placebo. In fact drilling down even further it is observed that out of the top 200 most unstable patients, eighty percent were on the drug. This indeed was very suspicious given the original distribution (40:60) and points to potential toxicity. **As it turns out, according to domain experts, this drug was discontinued for toxicity two years after this study was conducted.** We should also note that when we applied the same procedure on a clinical study where the drug in question was considered "safe" and met with FDA approval we did not find such a pattern of behavior.

6.2.2 Sociability Index

For the DBLP data, we define a related measure, called the Sociability Index. The Sociability Index is a measure of the number of different interactions that a node participates in. This behavior can be captured by the number of *Join* and *Leave* events that this node is involved in. Let $cl_i(x)$ be the cluster that node x belongs to at time i. Then, the Sociability Index is defined as:

$$SoI(x) = \frac{\sum_{i=1}^{T}(Join(x, cl_{i+1}(x)) + Leave(x, cl_i(x)))}{|\text{Activity(x)}|} \quad (5)$$

where $Activity(x) = \sum_{i=1}^{T}(x \in V_i)$ indicates the number of intervals that node x is active. Similar to the stability index, this is computed incrementally. The measure gives high scores to nodes that are involved in interactions with **different** groups.

Note that this measure does not represent the degree, which is a factor of the number of interactions a node is involved in. A case in point was a node that had a degree of 80, but a sociability of close to 0. When we examined the clusters the node belonged to, we found that the node interacted with the same nodes over several timestamps.

Application of Sociability for Link Prediction:

Problem Definition: The goal in link prediction [20] is to use past interaction information to predict future links between nodes. Since, in this paper we are analyzing the evolution of clusters, our goal is to predict future co-occurrences of nodes in clusters.

In the case of the DBLP collaboration network, two nodes are clustered together if they work on related papers or belong to the same work-group, as we have seen. We use the behavioral patterns we have discovered to predict the likelihood of authors being clustered together in the future.

For prediction, we employ the Sociability index which, as we described before, gives the likelihood of an author being involved in different collaborations. If an author has a high Sociability Index score, the chances of him/her joining a new cluster in the future is very high. We compute the Sociability Index scores for authors using Equation 5. We use a degree threshold to prune these authors. [5] We find all authors who have high Sociability index scores (> 0.75) and degree higher than the threshold and who have not been clustered together in the past. We then predict future cluster co-occurrences between them.

The seminal paper on link prediction [20] provided an empirical analysis of several techniques for link prediction. We adopt the same scenario and split our DBLP snapshots into two parts. We use the clusterings for the first 5 years (1997-2001) to predict new cluster co-occurrences for the next 5 years. Note that we are only considering new links between authors. Hence we consider only authors that have not been clustered together previously.

Similar to the evaluation performed by Liben-Nowell and Kleinberg [20], we use as our baseline a random predictor that randomly predicts pairs of authors who have not been clustered together before, and report the accuracy of all the methods methods relative to the random predictor. To perform comparisons, we implement three other approaches that were shown to perform well by the authors above:

Common Neighbor-based: This approach [23, 20] gives high similarity scores to nodes that have a large number of

[5]The threshold value we used was 50 papers

Predictor	Accuracy
Random Predictor Probability	0.14%
Sociability Index	**275**
Common Neighbors	25
Adamic-Adar	46
Jaccard Coefficient	23

Table 3: Cluster Link Prediction Accuracy. Accuracy score specifies the factor improvement over the random predictor. This method of evaluation is consistent with the one performed by Liben-Nowell and Kleinberg [20].

neighbors in common. This measure is based on the notion that if two authors have a large number of common neighbors and have not yet collaborated, there is a good chance that they will, in the future. It is given by:

$$Score(a,b) = |\gamma(a) \cap \gamma(b)| \tag{6}$$

where $\gamma(a)$ represents the neighbors of node a.

Adamic-Adar: This measure, originally proposed by Adamic and Adar [18] in relation to similarity between web pages, weights a common neighbor based on its importance. It is defined as:

$$Score(a,b) = \sum_{c \in (\gamma(a) \cap \gamma(b))} \frac{1}{log(|\gamma(c)|} \tag{7}$$

Nodes that have fewer neighbors are deemed more important than nodes with high degrees.

Jaccard coefficient: This measure, a popularly used similarity metric, computes the probability of two nodes having a common neighbor.

$$Score(a,b) = \frac{|\gamma(a) \cap \gamma(b)|}{|\gamma(a) \cup \gamma(b)|} \tag{8}$$

We used all the algorithms to predict cluster links for the last 5 years (2002-2006). We only considered pairs of authors who have not been clustered together in any of the 5 earlier snapshot graphs. The accuracy was computed as a factor of the random predictor [20], which was found to give a correct result with probability 0.14%. The results are shown in Table 3. We find that the *Sociability Index-based method performs the best overall*, outperforming other approaches appreciably with a large ratio of correct predictions (275). This result suggests that behavioral patterns of evolving graphs can be used to predict future behavior.

6.2.3 Popularity Index

The Popularity index is a measure defined for a cluster or community at a particular time interval. The Popularity Index of a cluster at time interval $[i, i+1]$ is a measure of the number of nodes that are attracted to it during that interval. It is defined as:

$$PI(C_i^j) = (\sum_{x=1}^{V_i} Join(x, C_i^j)) - (\sum_{x=1}^{V_i} Leave(x, C_i^j)) \tag{9}$$

This measure is based on the transformation a cluster undergoes over the course of a time interval. If a cluster does not dissolve in $[i, i+1]$ and a large number of nodes join the cluster and few leave it, then the cluster will have a high Popularity Index score. Note, that the Popularity index is an influence measure defined for a cluster. Also note, that this measure does not simply reflect the size of a cluster. If a cluster is very large, it does not indicate that it is attracting new nodes to it.

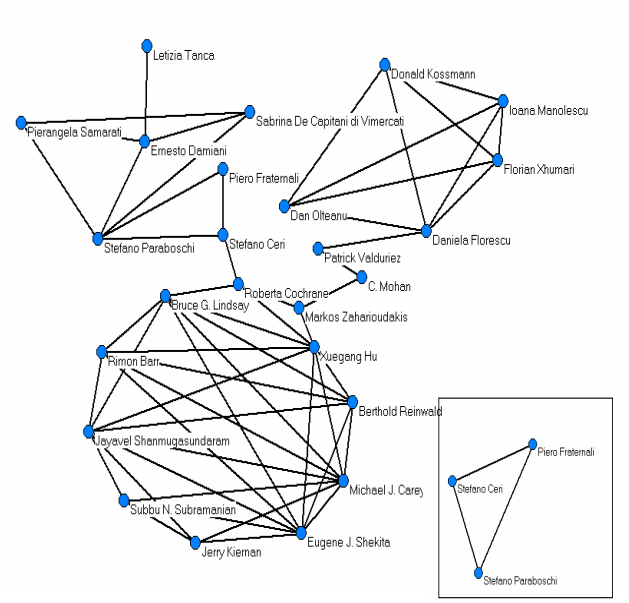

Figure 3: Illustration of certain authors belonging to a very popular cluster (1999-2000 time period). Original cluster (3 authors) shown in small box. In the large graph, we show the connections among 25 authors from the new cluster who published XML-related papers in that time-frame.

In the DBLP dataset, the popularity index can be used to find topics of interest for a particular year. For instance, if a large number of nodes join a cluster at a particular time point and a high percentage of them are working on a specific topic, it indicates a *buzz around that topic* for that year. On the other hand, if a large number of authors leave a cluster, and there are not many new nodes joining it, it indicates a loss of interest in a particular topic.

To find hot topics, we computed the popularity index scores for each cluster, and identified the most popular clusters, at each timestamp. We then examined the clusters that had high popularity scores to see if a large percentage of the authors in them were working on a particular topic.

We will now present an interesting result we obtained for the time span 1999-2000. In 1999, three authors Stefano Ceri, Piero Fraternali and Stefano Paraboschi formed a cluster. They were involved in a few papers on XML and web applications. In the next year (2000), these three authors were involved in a large number of collaborations, resulting in around 50 joins to their cluster. When we examined the topics of the papers that resulted, we found that 30 of these authors published papers related to XML. Since there were no papers on XML before 1999, this was a new and hot topic at that point. Since then there have been large number of papers on XML. Figure 3 shows the original 3 person cluster as well as the authors from the new cluster who were involved in XML related work in that particular time interval.

6.2.4 Influence Index

The influence index of a node is a measure of the influence this node has on others. Note that the influence that we are considering, in this case, is with regard to cluster evolution. We would like to find nodes that influence other nodes into participating in critical events. This behavior is measured

Author	Influence Index
H. V. Jagadish	290.125
Hongjun Lu	268.5
Jiawei Han	266.625
Philip S. Yu	251.66
Rajeev Rastogi	246.85
Beng Chin Ooi	237
Tok Wang Ling	220.428
Heikki Mannila	206.5
Wenfei Fan	200.142
Qiang Yang	199
Johannes Gehrke	179.85
Christos Faloutsos	167.85
Rakesh Agrawal	157.875
Edward Y. Chang	153
Guy M. Lohman	131.375
Dennis Shasha	129.29
Jennifer Widom	128.375
Hamid Pirahesh	127.625
Michael J. Franklin	121.5
Hector Garcia-Molina	118.625

Table 4: Top 20 Influence Index Values - DBLP Data

for a node x, over all timestamps, by considering all other nodes that leave or join a cluster when x does. If a large number of nodes leave or join a cluster with high frequency when a certain node x does, it suggests that node x has a certain positive influence on the movement of the others. Let $Companions(x)$ represent all nodes over all timestamps that join or leave clusters with node x. The Influence for node x is given by:

$$Inf(x) = \frac{|\text{Companions(x)}|}{|\text{Moves(x)}|} \tag{10}$$

Here $Moves(x)$ represents the number of *Join* and *Leave* events x participates in. [6] Note that, this definition by itself, does not measure influence, since nodes that interact and move along with highly influential nodes will have high Influence score values as well. Hence, to eliminate these *follower* nodes, additional pruning constraints are needed.
Let $Max\_Int(x)$ denote the node with which node x has the maximum number of interactions. Let $Deg(x)$ denote the number of neighbors of node x.
$Influence\ Index(x) = Inf(x)$ unless any of the following hold :

- $Inf(Max\_Int(x)) > Inf(x)$
- $Deg(Max\_Int(x)) > Deg(x)$

If any of the two conditions hold, $Influence\ Index(x) = 0$. The additional constraints are imposed in order to ensure that we find the most influential nodes in the datasets.

We computed the Influence Index scores for nodes in the DBLP dataset. The top 20 authors are shown in Table 4. We further illustrate the use of the Influence index in the next section.

6.3 Incorporating semantic content

To quantify the relevance of author pairs in the DBLP dataset, we make use of semantic information from their papers. To construct a knowledge base, we identify a set of unique keywords composed of frequently used technical terms in the papers from the conferences in our corpus. We then group related keywords together to form keyword-sets,

[6]To compute the Influence index efficiently, we incrementally update Companions() and Deg() for all nodes. The number of *Join* and *Leave* events (Moves()) are used in the Sociability case also and are stored incrementally as well.

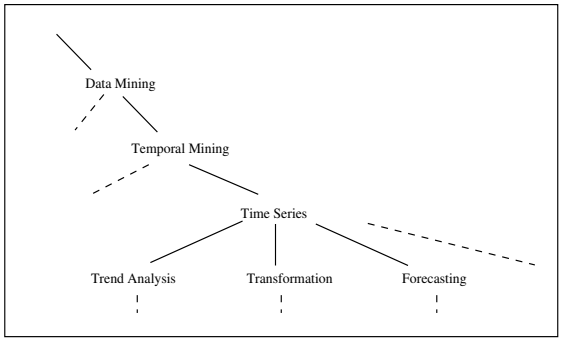

Figure 4: A sample subgraph of the keyword DAG hierarchy

$k = \{w_1, ..., w_n\}$, where each w_i is a keyword. Each paper is thus labeled with a set of related keywords. An example of keyword-set is $\{'WWW', 'Web', 'Internet'\}$. Similarly each author is associated with a paper-set, P, consisting of the union of the keyword-sets from all the papers that she/he co-authored in a particular time period. $P = \{k_1, ..., k_p\}$ where each k_i is a keyword-set. Thus every node (i.e. author) in our co-authorship graph is associated with keyword-sets based on the corresponding papers she/he has authored. However, the relationship between two authors cannot be inferred by merely comparing their paper-sets since different keywords are associated with different semantic meanings.

To capture this, we construct an ontology in the form of a hierarchy or a DAG where each node represents a keyword-set. [7] Nodes at higher levels in the hierarchy represent keyword-sets that are more general, while nodes closer to the leaves represent more specific keywords. A node a has a child b if b is a keyword-set that represents a more specific term related to a. An example hierarchy is shown in Figure 4.

Using this hierarchy, we define the notion of semantic similarity [21, 12] in this context. To begin with, the Information Content (IC) of a keyword-set, using Resnik's definition [26], is given as:

$$IC(k_i) = -ln\frac{F(k_i)}{F(root)} \tag{11}$$

where k_i represents a keyword-set and $F(k_i)$ is the frequency of encountering that particular keyword-set over all the paper titles in the corpus. Here, $F(root)$ is the frequency of the root term. Note that frequency count of a keyword-set includes the frequency counts of all subsumed keyword-sets in an is-a hierarchy. Accordingly, the root of our hierarchy includes the frequency counts of every other keyword in the ontology. Note that terms with smaller frequency counts will therefore have higher information content values(i.e. more informative).

Using the above definition, the Semantic Similarity (SS) between two keyword-sets can be computed as follows:

$$SS(k_i, k_j) = IC(lcs(k_i, k_j)) \tag{12}$$

where $lcs(k_i, k_j)$ refers to the lowest common subsumer of keyword-sets k_i and k_j.

The Semantic Similarity between two authors,A_a and A_b, is a function of the keyword-sets that they have in common.

[7]The keyword ontology was manually constructed by the authors for this work.

We define it as the weighted sum of the Information Content of the keyword-sets associated with both of them. It is represented as follows:

$$Similarity(A_a, A_b) = \sum_{k_i \in K_a and k_i \in K_b} (IC(k_i)) \quad (13)$$

where K_a and K_b are the paper-sets that annotate authors A_a and A_b respectively. If two authors have identical paper-sets consisting of very specific topics (high Information Content), then they will have the highest semantic similarity.

Next, we illustrate how semantic similarity can be used to reason about community-based critical events, using examples from the DBLP dataset.

6.3.1 Group Merge:

The key intuition here is that the probability of a merge event depends on the Semantic Similarity between two clusters. For instance, if two clusters are comprised of papers on highly related topics, it stands to reason that there is a high likelihood of a merge between them.

Let us consider two clusters C_i^a and C_i^b consisting of k^a and k^b paper-sets respectively. A simple way to compute the semantic similarity between two clusters is based on the information content of their keyword-set pairs, as shown below:

$$Inter\_SS(C_i^a, C_i^b) = \frac{\sum_{k^a}^{m=1} \sum_{k^b}^{n=1} SS(m,n)}{k^a * k^b} \quad (14)$$

To define the semantic similarity between two clusters, one can also employ a mutual information measure [30]. Mutual information is a measure of the amount of statistical information shared between two distributions. We can compute the mutual information between two clusters based on the keyword-sets of all the authors belonging to those clusters. This would be applicable when there is history data that can be used to estimate the probabilities. Given probabilities of keyword-sets m and n occurring in a cluster as p^m and p^n respectively, and their co-occurrence probability p^{mn}, we can define the Semantic Mutual Information (SMI) between the two clusters C_i^a and C_j^b as:

$$SMI(C_i^a, C_i^b) = \sum_{m=1}^{k^a} \sum_{n=1}^{k^b} SS(m,n) * p^{mn} * \log_{k^a * k^b} \frac{p^{mn}}{p^m * p^n}$$

Note that each keyword-set pair is weighted by the information content of their most informative common ancestor in the keyword ontology (i.e. Lowest Common Subsumer).

The keyword information can be used to analyze the effect of topics on community based events. If two clusters in snapshot S_i - C_i^a and C_i^b, are merging in time T_i into cluster C_{i+1}^c, the inter-cluster similarity of C_i^a and C_i^b and their similarity to the new cluster formed, C_{i+1}^c can be indicative of the evolution of a topic by combining two similar sub-topics.

We illustrate this with an example from the DBLP dataset. In 1999, we found two clusters with high semantic similarity scores (shown in the first two columns of Table 5), due to the common keywords - "constraint", "query", "aggregate/aggregation". We found that these two clusters merged in the following year (2000) giving a single cluster with the following papers. All three clusters are shown in Table 5. This example illustrates the relationship between semantic similarity between clusters and the $Merge$ event. This example illustrates the relationship between semantic similarity between clusters and the $Merge$ event.

6.3.2 Group Split:

As we saw earlier, an important factor causing a *Split* event is topic divergence. The probability of topic divergence is inversely proportional to the semantic similarity between topics in a cluster. We define the intra-cluster semantic similarity $Intra\_SS$ as the average of the semantic similarity scores between the keyword-sets in the cluster. The semantic similarity within a cluster is given by:

$$Intra\_SS(C_i^a) = \frac{\sum_{k^a}^{m=1} \sum_{k^a}^{n=m+1} SS(m,n)}{\frac{k^a * (k^a - 1)}{2}} \quad (16)$$

where k^a, as before, represents the total number of keyword-sets in cluster C_i^a. If the intra-cluster semantic similarity is small, then it indicates that the cluster is likely to split in the next few timestamps. For instance, in 2001 we found a cluster with relatively low intra-cluster semantic similarity. This cluster contained very disparate keyword-sets {{web,data extraction} and {spatio-temporal, information system}}. The papers in this cluster are shown in the first column of Table 6. In 2002, this cluster split into two different clusters, shown in the columns 2 and 3 in Table 6. Thus, the semantic similarity within clusters can be useful to characterize and predict future *Split* events.

6.3.3 Group Continue:

A continue event is useful to study the evolution of a topic. Since the authors belonging to the cluster do not change, a continue event can provide information about how ideas evolve. The clusters that correspond to a continue event will tend to have a reasonably high semantic similarity score. We present some examples of the papers corresponding to continue events in Table 7. The first column in the table represents a paper from the cluster at time stamp i and the second column denotes the most similar paper from the continuing cluster at $i + 1$.

7. DIFFUSION MODEL FOR EVOLVING NETWORKS

We use the behavioral patterns discussed in the previous section to define a diffusion model for evolving networks. Diffusion models have been studied for complex networks [1, 9] and specifically in the context of influence maximization [15, 16] where the task is to identify key start nodes that can be used to effectively propagate information through the network. The information can be either an idea or an innovation that propagates through the network over time. In this regard, Kempe et al [15, 16] discuss two models for the spread of influence through social networks. We examine this scenario from an evolving perspective, where the nodes and edges of the network are transient.

Let us consider an idea or innovation that arrives into the network at timestamp a. We define four states for nodes in the evolving network - *active*, *inactive*, *contagious* and *isolated*. These states are not mutually exclusive, as we will see later. At the beginning of the diffusion process, at time a, all nodes in the network are *inactive*. The diffusion model begins with a set of nodes that are activated (provided the information) at the first timestamp. These

Cluster 1	Cluster 2	Merged Cluster
1) Querying Aggregate Data 2) On the Orthographic Dimension of Constraint Databases 3) A Performance Evaluation of Spatial Join Processing Strategies	Exact and Approximate Aggregation in Constraint Query	1) On the Content of Materialized Aggregate Views. 2) Automatic Aggregation Using Explicit Metadata 3) Reachability and Connectivity Queries in Constraint Databases

Table 5: Group Merge Event - Column 1 and 2 show the papers from the original clusters. Column 3 shows the papers from the merged cluster.

Cluster	Split Cluster 1	Split Cluster 2
1) Web Site Evaluation: Methodology and Case Study 2) RoadRunner: Towards Automatic Data Extraction from Large Web Sites. 3) SIT-IN: a Real-Life Spatio-Temporal Information System.	Spatio-temporal Information Systems in a Statistical Context.	RoadRunner: automatic data extraction from data-intensive web sites.

Table 6: Group Split Event - Column 1 shows the papers from the original cluster. Columns 2 and 3 show the papers from the split clusters.

active nodes will be *contagious* briefly, in that, in the next timestamp they can activate other nodes they interact with, passing on the information they received. Subsequently, the newly *contagious* nodes proceed to attempt to activate their *inactive* neighbors. The process continues, with the information propagating through the network until at time T there are $\sigma(T)$ active nodes in the network. In earlier work, the effect of a *contagious* node has been limited to one timestamp, which means that an *active* node can attempt to activate its neighbors only once. However this does not capture the fact that the network topology can change, with the neighbors of nodes changing over time. After a *contagious* node has activated some of its neighbors, new nodes might come in contact with it in subsequent time instances. In this regard, we relax this constraint allowing a node to remain *contagious* when confronted with new neighbors. A node can thus attempt to activate each unique neighbor once.
When a node is surrounded by *contagious* nodes, it's propensity to get activated is given by an activation function.

Definition: The activation function for a node v, $Ac_v()$ is a non-negative function that maps the weights associated with the neighbors of v, $wt(x,v)\ \forall x = neighbor(v)$ to either 0 or 1.

We describe two Activation functions,Max and Sum, for a node v as

$$Ac_v^{max}(u_1,u_2,...,u_m) = (\arg\max_{1\le i\le m}(wt_v(u_i)) \ge \theta_v \quad (17)$$

$$Ac_v^{sum}(u_1,u_2,...,u_m) = \sum_{1\le i\le m} wt_v(u_i) \ge \theta_v \quad (18)$$

Here, θ_v denotes the activation threshold for node v. The weights on the edges represent the likelihood of that particular interaction leading to an activation. If the edge between two nodes has a high weight, it indicates that if one of the nodes gets activated, the chance of it activating the other is high. In our case, we define the weights for an interaction based on the Sociability Index values of the nodes involved, since Sociability can best capture the aforementioned property. If a node is highly sociable, it has a high propensity of passing on information to other nodes it interacts with. Hence, for each interaction of node x with a neighbor, y, the weight of the interaction is given by

$$wt_x(y) = SoI(y) \quad (19)$$

Similarly $wt_y(x) = SoI(x)$. Note that since we are dealing with diffusion over time, the SoI(x) represents the cumulative value defined in (5) until the current time point. The Sociability values thus can change over time.

The set of nodes activated in a given time interval i due to the initial node x and the cardinality of this set are given by $R_x(i)$ and $\sigma_x(i)$ respectively. The total set and number of nodes activated due to x after T timestamps of the diffusion process are given as

$$R_x(T) = \cup_{i=1}^{T} R_x(i) \quad (20)$$

$$\sigma_x(T) = \sum_{i=1}^{T} \sigma_x(i) \quad (21)$$

It is also important to consider the effect of deleted nodes and edges. When a node is not participating in any interaction in the current timestamp it is said to be *isolated*. An *isolated* node cannot influence any other nodes since it has no interactions.

Claim 7.1. *An active node can be isolated.*

PROOF. As we mentioned earlier, the topology of the network can change at every timestamp. Hence, a node that has just become active can be separated from its neighbors due to the deletion of edges. The node will then remain isolated until a new interaction is formed with it.

An example of this scenario is shown in Figure 5. Node A begins the diffusion process activating node B. B is contagious at time i and activates node C. However at the next timestamp, C no longer interacts with B, D and E. Although it is active and contagious, it is isolated at this time instant. In the future, if it interacts with other nodes, it can attempt to activate them once.

Influence Maximization: Influence Maximization is an important problem for diffusion models and has practical applications in viral marketing and epidemiology. The challenge is to find an initial set of active nodes that can influ-

Cluster 1	Cluster 2
Object Recognition Using Appearance-Based Parts and Relations	Hierarchical Organization of Appearance-Based Parts and Relations for Object Recognition
Mining Insurance Data at Swiss Life	A Data Mining Support Environment and its Application on Insurance Data
M-tree: An Efficient Access Method for Similarity Search in Metric Spaces	Processing Complex Similarity Queries with Distance-Based Access Methods
Optimizing Queries in Distributed and Composable Mediators	Distributed View Expansion in Composable Mediators
Scaling up Dynamic Time Warping to Massive Dataset	Scaling up dynamic time warping for datamining applications

Table 7: Continue Events - Column 1 shows a paper from a cluster that is part of a Continue event. Column 2 shows the paper from the cluster in the next timestamp.

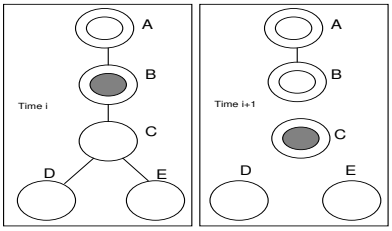

Figure 5: Isolation of active nodes. The double circles indicate active nodes. The grey inner circle represents contagious nodes. Nodes D and E are inactive.

ence the most number of inactive nodes over the duration of the diffusion.

Problem Definition: Given a graph G that evolves over T timestamps and a diffusion model, the task is to find the set of k initial nodes S to maximize $R_S(T)$ where $R_S(T) = \cup_{x \in S} R_x(T)$

Kempe *et al* [15, 16] discuss a greedy algorithm for finding the initial set that maximizes the influence. They find the start nodes that maximize $\sigma(T)$, where $\sigma(T) = \sum_{x \in S} \sigma_x(T)$. To find $\sigma_x(T)$ for all nodes x, they simulate the diffusion process over the network. However, in our case, the network is dynamic with edges and nodes getting added or deleted. At a particular timestamp i, it is unclear how the network is going to change at time $i+1$. Hence, simulating the diffusion on the static graph will not work. Considering high-degree nodes to start the diffusion process has been examined in social network research [32]. However, using the degree to determine the initial nodes may not be a good option [15], since it is possible for nodes of high degree to be clustered, which limits their range. Instead, we advocate the use of the Influence Index we defined in the previous section for this purpose. The Influence Index is an incremental measure which considers the behavior of the nodes over the previous timestamps and chooses nodes that have the highest degree of influence over other nodes. Also, *by pruning followers of influential nodes*, we are ensuring that the nodes with high influence index are *not likely to be clustered*.

Empirical Evaluation: We conducted an experiment to evaluate the performance of the Influence index-based initialization. To compare, we employed an approach based on accumulated degree, where we picked nodes that had the highest degree, over the preceding timestamps, to be the start nodes. As a baseline, we implemented a random approach where the initial nodes are chosen at random. We constructed a graph using a subset of nodes from the DBLP collaboration network. We considered the interactions from 1997-2001 to compute sociability, degree and influence scores. We then assumed the introduction of a new idea at 2002 and then tracked its diffusion through the network over the next 4 timestamps (till 2006). We used an active set size, k, of 5 and both the Sum and Max activation functions. We performed the experiments 100 times, choosing random activation thresholds for the nodes from [0,1]. The results are shown in Table 8. Our results suggest that the Influence index can be useful in this regard. It succeeds in *activating 61% and 81% of the nodes* in the network in 4 timestamps for the Max and Sum Activation functions respectively, clearly outperforming the other approaches.

Method	Activated nodes (%)	Activated nodes (%)
	Max Activation	Sum Activation
Random	16.67	20.39
Accumulated Degree	51.9	65.33
Influence	61.12	81

Table 8: Diffusion Results

8. CONCLUSION AND FUTURE WORK

In this paper, we have presented an event-based framework for characterizing the evolution of dynamic interaction graphs. The framework is based on the use of certain critical events that facilitate our ability to compute and reason about novel behavior-oriented measures, which can offer new and interesting insights for the characterization of dynamic behavior of such interaction graphs. We have presented a diffusion model for evolving networks and have shown the use of behavioral patterns for influence maximization. We have demonstrated the efficacy of our framework in characterizing and reasoning on two different datasets - the DBLP dataset and a clinical trials dataset. The application of the behavioral patterns we obtained to a cluster link prediction scenario provided favorable results, with the Sociability Index producing a large number of accurate predictions.

A key next step for us is to extend our framework for reasoning about events over time. In this context we propose to adapt and evaluate the use of Allen's interval algebra [2]. We would also like to improve the performance of Semantic Similarity by extending our ontology, which currently comprises only of keywords from the titles of papers. We would also like to extend our framework to reason and infer other behavioral patterns, as well as include other types of interaction graphs.

9. ACKNOWLEDGEMENTS

This work is supported in part by the DOE Early Career Principal Investigator Award No. DE-FG02-04ER25611 and NSF CAREER Grant IIS-0347662. The authors would like to thank Sameep Mehta for his useful comments and suggestions.

10. REFERENCES

[1] F. Alkemade and C. Castaldi. Strategies for the diffusion of innovations on social networks. *Computational Economics*, 25(1-2), 2005.

[2] J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[3] L. Backstrom, D. P. Huttenlocher, and J. M. Kleinberg. Group formation in large social networks: membership, growth, and evolution. *SIGKDD*, 2006.

[4] A.-L. Barabasi and E. Bonabeau. Scale-free networks. *Scientific American*, 288:60–69, 2003.

[5] A.-L. Barabasi, H. Jeong, R. Ravasz, Z. Nda, T. Vicsek, and A. Schubert. On the topology of the scientific collaboration networks. *Physica A*, 311:590–614, 2002.

[6] D. Chakrabarti, R. Kumar, and A. Tomkins. Evolutionary clustering. *SIGKDD*, 2006.

[7] A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Physical Review E*, 70, 2004.

[8] F. M. Couto, M. J. Silva, and P. Coutinho. Semantic similarity over the gene ontology: family correlation and selecting disjunctive ancestors. *Proc. ACM CIKM Intl. Conf. on Information and Knowledge Management*, pages 343–344, 2005.

[9] R. Cowan and N. Jonard. Network structure and the diffusion of knowledge. *Journal of Economic Dynamics and Control*, 28:1557–1575, 2004.

[10] T. Falkowski, J. Bartelheimer, and M. Spiliopoulou. Mining and visualizing the evolution of subgroups in social networks. *IEEE/WIC/ACM International Conference on Web Intelligence*, 0, 2006.

[11] G. W. Flake, S. R. Lawrence, C. L. Giles, and F. M. Coetzee. Self-organization and identification of web communities. *IEEE Computer*, 36:66–71, 2002.

[12] P. Ganesan, H. Garcia-Molina, and J. Widom. Exploiting hierarchical domain structure to compute similarity. *ACM Trans. Inf. Syst*, 21(1), 2003.

[13] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proc. National Academy of Sciences of the United States of America*, 99(12):7821–7826, 2002.

[14] N. K. Govindaraju, M. Lin, and D. Manocha. General purpose computations using graphics processors. *Ninth Annual Workshop on High Performance Embedded Computing*, 2005.

[15] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. *SIGKDD*, 2003.

[16] D. Kempe, J. Kleinberg, and E. Tardos. Influential nodes in a diffusion model for social networks. *Proc. Intl. Colloquium on Automata, Languages and Programming (ICALP)*, 2005.

[17] J. Kurzak and J. Dongarra. Implementing linear algebra routines on multicore processors with pipelining and a lookahead,. *PARA*, 2006.

[18] A. A. Lada, , and E. Adar. Friends and neighbors on the web. *Social Networks*, 25(3):211–230, July 2003.

[19] J. Leskovec, J. M. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. *SIGKDD*, 2005.

[20] D. Liben-Nowell and J. M. Kleinberg. The link prediction problem for social networks. *Proc. ACM CIKM Intl. Conf. on Information and Knowledge Management*, 2003.

[21] D. Lin. An information-theoretic definition of similarity. *Proc. 15th Intl. Conf. Machine Learning*, July 1998.

[22] J. Lin, M. Vlachos, E. Keogh, and D. Gunopulos. Iterative incremental clustering of time series. *EDBT*, pages 106–122, 2004.

[23] M. Newman. Clustering and preferential attachment in growing networks. *Phys. Rev. E*, 64, 2001.

[24] M. E. J. Newman. Modularity and community structure in networks. *Proc. National Academy of Sciences of the United States of America*, 103(23):8577–8582, 2006.

[25] J. Reichardt and S. Bornholdt. Detecting fuzzy community structures in complex networks with a potts model. *Physical Review Letters*, 93, 2004.

[26] P. Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artifical Intelligence Research*, 11:95–130, 1999.

[27] R. Richardson, A. F. Smeaton, and J. Murphy. Using WordNet as a knowledge base for measuring semantic similarity between words. Technical Report CA-1294, Dublin, Ireland, 1994.

[28] R. Samtaney, D. Silver, N. Zabusky, and J. Cao. Visualizing features and tracking their evolution. *IEEE Computer*, 27(7):20–27, 1994.

[29] M. Spiliopoulou, I. Ntoutsi, Y. Theodoridis, and R. Schult/. Monic - modeling and monitoring cluster transitions. *SIGKDD*, 2006.

[30] A. Strehl and J. Ghosh. Cluster ensembles a knowledge reuse framework for combining multiple partitions. *Journal on Machine Learning Research*, 3:583–617, December 2002.

[31] S. van Dongen. A cluster algorithm for graphs. Technical Report INS-R0010, National Research Institute for Mathematics and Computer Science in the Netherlands, Amsterdam, 2000.

[32] S. Wasserman and K. Faust. Social network analysis. *Cambridge University Press*, 1994.

[33] H. Yang, S. Parthasarathy, and S. Mehta. Mining spatial object patterns in scientific data. *Proc. 9th Intl. Joint Conf. on Artificial Intelligence*, 2005.