Static and Dynamic Structural Correlations in Graphs

Jian Wu, Ziyu Guan, Qing Zhang, Ambuj Singh, Xifeng Yan

Abstract—Real-life graphs not only contain nodes and edges, but also have events taking place, e.g., product sales in social networks. Among different events, some exhibit strong correlations with the network structure, while others do not. Such structural correlations will shed light on viral influence existing in the corresponding network. Unfortunately, the traditional association mining concept is not applicable in graphs since it only works on homogeneous datasets like transactions and baskets.

We propose a novel measure for assessing such structural correlations in heterogeneous graph datasets with events. The measure applies hitting time to aggregate the proximity among nodes that have the same event. In order to calculate the correlation scores for many events in a large network, we develop a scalable framework, called gScore, using sampling and approximation. By comparing to the situation where events are randomly distributed in the same network, our method is able to discover events that are highly correlated with the graph structure. We test gScore's effectiveness by synthetic events on the DBLP co-author network and report interesting correlation results in a social network extracted from TaoBao.com, the largest online shopping network in China. Scalability of gScore is tested on the Twitter network. Since an event is essentially a temporal phenomenon, we also propose a dynamic measure which reveals structural correlations at specific time steps and can be used for discovering detailed evolutionary patterns.

Index Terms—Graph, structural correlation, hitting time.

1 Introduction

The rise of the Web, social networks, and bioinformatics has presented scientists with numerous graphs, each consisting of millions of nodes and edges. Hidden in these large datasets are the answers to important questions in networking, sociology, business, and biology. These graphs not only have topological structures, but also contain events/activities that occurred on their nodes. For example, an eBay customer could sell or bid on a product. A Facebook user could play a Zynga game with friends. This complex combination raises new research problems in graph data analysis [26], [21], [10].

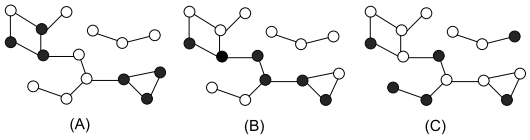


Fig. 1. Structural Correlation

Among different events taking place in a network, some exhibit strong correlations with the network structure, while others do not. Such structural correlations might shed light on viral influence existing in the corresponding network, which is the key to many research problems in product marketing [9], online advertisement [5], and recommendation [15]. Fig. 1 shows the distribution of three different events over the same graph. We can easily instantiate Fig. 1 into different application scenarios. They could be three different products bought by members in a social network, or three different intrusion alerts raised by computers in a computer network. In terms of products, dark nodes in Fig. 1(A), 1(B) and 1(C) represent the members who purchased the products A, B and C, respectively. Intuitively, Fig. 1 shows that in this network, people who bought product A (or B) are closer to each other. On the contrary, black nodes for C seem to be randomly distributed. In this scenario, the network would be suitable for promoting A and B and we can promote A and B to people who have not bought them. While it is hard to derive deterministic relationships between sales and the network structure, it is possible to study how the sales is correlated to the structure. In fact, one can raise several interesting questions related to the structure and events distributed over the structure:

1) In regard to the sales of A and B, which one is more related to the underlying social network?
2) Given two networks G and G' for the same group of users, e.g., their email network and Facebook network, is the sales of A more related to G than G'?
3) If we have snapshots of network G during different periods, can we measure how product A was dispersed over the network over time? Was it purchased by a small community at the beginning?

In order to answer the above questions, we need to address the following research problems: (1) *How to*

- *J. Wu is with the College of Computer Science, Zhejiang University, Hangzhou, CN 310027. E-mail: wujian2000@zju.edu.cn*
- *Z. Guan, A. Singh and X. Yan are with the Department of Computer Science, University of California, Santa Barbara, CA 93106. E-mail: ziyuguan@cs.ucsb.edu*
- *Q. Zhang is with TaoBao.com, Hangzhou, CN 310099. E-mail: yunzheng@taobao.com*

Digital Object Indentifier 10.1109/TKDE.2012.133

define and measure the correlation between the graph structure and events? (2) *How to compute the measure efficiently in large graphs, if we want to rank all events according to the measure*?

Unfortunately, the classic association mining concept is not applicable in this setting since it only works on homogeneous datasets like transactions and baskets [1], [27]. In this paper, we propose a novel measure to assess structural correlations in a graph. The measure aggregates the proximity among nodes on which the same event has occurred, using a proximity function such as hitting time [19]. We develop an efficient computation framework, gScore (**G**raph **S**tructural **Cor**relation **E**stimation), to quickly calculate correlation scores in large scale networks. By estimating the deviation from the expected correlation score of a random situation, our method is able to discover the events of nodes that are highly correlated with the graph structure.

Our contributions. We propose a novel concept, structural correlation, to measure how an event is distributed in a graph and address a key research problem in analyzing the relation between structures and contents. While many studies have demonstrated that social links could significantly influence the behavior of human beings [7], [16], [9], we suspect that such influence should be further scrutinized for more fine-grained knowledge: *in which kind of social links (e.g., phone networks, email networks, employee networks, etc.) and for which kind of behaviors (e.g.,shopping, hobby, interest, and opinion) social influence is observed, and how strong the influence is*. In this study, we quantify the correlation between link structures and human behaviors, and make different behaviors' correlations comparable using statistical significance. We discover that the correlation actually fluctuates dramatically with regard to link types, event types, and time, implying a need to further examine the cause of the fluctuation. Note that in this work, we are not going to perform a causality study of correlations [3]. Our problem can be viewed as a step before the causality problem since the latter assumes the existence of correlations.

We systematically introduce a framework to define and measure structural correlations in graphs. The principle is to aggregate the proximity among nodes which have the same event and compare the aggregated proximity to the situations where the event is randomly distributed in the graph. This framework can integrate various graph proximity measures [8] such as hitting time [19], personalized PageRank [23] and Katz [14].

We take hitting time as an example and propose a modified version named *Decayed Hitting Time* (**DHT**) to better and faster calculate structural correlation. We develop scalable algorithms using sampling and approximation techniques to calculate DHT for individual nodes and the average DHT for all the nodes which share the same event. We investigate the expectation and variance of the correlation when an event is randomly distributed over a graph and develop several approximation techniques. These techniques can help us quickly estimate the deviation from random cases, thus making online computation of structural correlations possible. We demonstrate gScore's effectiveness by constructing synthetic events on the DBLP co-author network. We also report interesting correlated and uncorrelated products discovered from TaoBao.com, the largest online shopping network in China. Scalability of gScore is tested on the Twitter network.

Since an event is essentially a temporal phenomenon, we also propose a dynamic measure which can be used to estimate the fine-grained structural correlation of an event at different time steps. The dynamic measure computes the DHT from the currently "infected" node to previously "infected" ones, and then estimates the significance of the obtained DHT. It could be helpful for setting up fine-grained promotion strategies for products in an online shopping network. We compare the dynamic measure with repeatedly applying the original correlation measure on the evolving event node set on DBLP and TaoBao and demonstrate its effectiveness.

2 Problem Formulation

An attributed graph $G = (V, E)$ has an event set Q. The event set of a node v is written as $Q(v) \subseteq Q$. In this work, we consider undirected and unweighted graphs. Nevertheless, the proposed measure and algorithms could be generalized to weighted and/or directed graphs.

Suppose an event q (e.g. purchasing a specific product) is taking place in G. Each node v can take two values in terms of q: $f_q(v) = 1$ if $q \in Q(v)$; otherwise, $f_q(v) = 0$. Let $m = \sum_v f_q(v)$ denote the number of nodes where q occurred. Let $n = |V|$ be the number of nodes in G. We could formulate the following two research problems:

Problem 1: Determine whether there is a correlation between q and the graph structure of G. If not, q is just randomly distributed in G.

Its associated ranking problem is as follows:

Problem 2: Given a set of different events $Q = \{q_i\}$ on G, rank $\{q_i\}$ according to their correlation strength with respect to the graph structure of G.

To address these problems, we need a measure that captures the distribution of an event in a graph, and then assess the significance of the observed measure score. A simple measure could be to assess the probability that a node's neighbors have q given that the node has q. However, this 1-neighborhood event fraction measure could not well capture the distribution of an event in a graph since it only considers direct neighbors. We will show this drawback in experiments.

3 Structural Correlations

Intuitively, if the m event nodes of q are close to one another, then the correlation is high. Otherwise, it will be more similar to a random situation where q is randomly distributed in G. Therefore, we propose using the average proximity between one node in those m nodes

and the remaining $m-1$ nodes to assess the **structural correlation** between q and G,

$$\rho(V_q) = \frac{\sum_{v \in V_q} s(v, V_q \setminus \{v\})}{|V_q|}, \quad (1)$$

where V_q is the set of m nodes on which q occurred and $s(v, V_q \setminus \{v\})$ is the closeness of the remaining nodes to v. $s(\cdot)$ can be any graph proximity measure that measures the proximity between a node and a set of nodes in a graph topology notion. We could rewrite $s(\cdot)$ as the sum of the contribution of each node in $V_q \setminus \{v\}$,

$$s(v, V_q \setminus \{v\}) = \sum_{u \in V_q \setminus \{v\}} s_q(v \rightsquigarrow u), \quad (2)$$

where $s_q(v \rightsquigarrow u)$ is the contribution of u to $s(v, V_q \setminus \{v\})$. We can divide proximity measures into two categories: pairwise and holistic. Pairwise measures are defined on node pairs, e.g. shortest distance and personalized PageRank, while holistic measures explicitly estimate the proximity of a node to a set of nodes. The set of nodes is reached if any node in the set is reached. One example is hitting time: starting from node v, it calculates the expected number of steps needed to reach one node in $V_q \setminus \{v\}$. For pairwise measures, $s_q(v \rightsquigarrow u)$ is invariant to q, while for holistic measures $s_q(v \rightsquigarrow u)$ depends on the distribution of q in G. For example, in Fig. 2 suppose we want to measure the proximity of v to other black nodes. If we use shortest distance, u's contribution is 4. If hitting time is adopted, u does not even influence the measure score, since v always hits other black nodes in one step. Hitting time is less influenced by remote black nodes and anomalies, and more focused on graph local areas. This is desirable since we may have correlation patterns on the graph which are distant to one another, as shown in Fig. 2. Holistic proximity measures could help detect correlations better. In this work, we will focus on hitting time. Nevertheless, our framework is applicable to other proximity measures too.

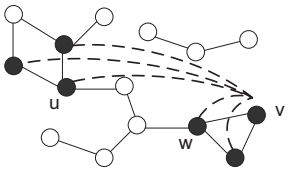


Fig. 2. Measuring Structural Correlations

Having the measure ρ, we should also consider how the ρ value is distributed for random cases, i.e. there is no correlation. Let $\rho(\widehat{V}_m)$ denote the correlation score for a randomly selected set $\widehat{V}_m$ of m nodes where $m = |\widehat{V}_m| = |V_q|$. As m increases, there will be an increasing chance for m randomly selected nodes to be close to one another (in the extreme case $m = n$, nodes are obviously very close). Thus, we should estimate the significance (denoted by $\tilde{\rho}(V_q)$) of $\rho(V_q)$ compared to $\rho(\widehat{V}_m)$. Since it is hard to obtain the distribution of $\rho(\widehat{V}_m)$, we propose to estimate the expectation and variance of $\rho(\widehat{V}_m)$ (denoted by $E[\rho(\widehat{V}_m)]$ and $Var[\rho(\widehat{V}_m)]$, respectively), and then estimate $\tilde{\rho}(V_q)$. More details are presented in Section 5.2. We refer to this framework as gScore (**G**raph **S**tructural **Cor**relation **E**stimation).

3.1 Random Walk and Hitting Time

A *random walk* is a Markov chain. If in the t-th step we are at node v_t, we move to a neighbor of v_t with probability $\frac{1}{d(v_t)}$, where $d(v_t)$ is the degree of node v_t. Let $\mathbf{A}$ denote the adjacency matrix of G. A_{ij} equals 1 if there is an edge between node v_i and v_j, and 0 otherwise. We use $\mathbf{P} = [p_{ij}]_{n \times n}$ to denote the transition probability matrix of the random walk. We have $\mathbf{P} = \mathbf{D}^{-1}\mathbf{A}$ where $\mathbf{D}$ is a diagonal matrix with $D_{ii} = d(v_i)$. We use $\Pr(pa_{1 \rightsquigarrow l}) = p_{12}p_{23}\dots p_{(l-1)l}$ to denote the probability that a random walk takes path $v_1, v_2, \dots, v_l$, starting from v_1 (also called the probability of that path).

Let B be a subset of V. The hitting time (also called access time) [19] from a node v_i to B is defined as the expected number of steps before a node $v_j \in B$ is visited in a random walk starting from v_i. Let $h(v_i, B)$ denote the hitting time from v_i to B and x_t denote the position of the random walk at time step t. By definition, we have

$$h(v_i, B) = \sum_{t=1}^{\infty} t \Pr(T_B = t | x_0 = v_i), \quad (3)$$

where $\Pr(T_B = t | x_0 = v_i)$ is the probability that the random walk starting from v_i first hits a node in B after t steps. Hitting time is a weighted average path length where each considered path is weighted by its probability. One can easily derive [19]

$$h(v_i, B) = \sum_{v_k \in V} p_{ik} h(v_k, B) + 1. \quad (4)$$

Eq. (4) expresses a one step look-ahead property of hitting time. The expected time to reach a node in B from v_i is equivalent to one step plus the mean of hitting times of v_i's neighbors to a node in B. By the definition of hitting time (i.e. Eq. (3)), we have $h(v_i, B) = 0$ for $v_i \in B$. From the above analysis, we obtain a linear system for computing hitting time from all nodes to B:

$$\begin{cases} h(v_i, B) = 0 & v_i \in B \\ h(v_i, B) = \sum_{v_k \in V} p_{ik} h(v_k, B) + 1 & v_i \notin B \end{cases}. \quad (5)$$

It can be shown that this linear system has a unique solution [19]. Hitting time can be used to measure a notion of asymmetrical proximity among nodes in a graph with respect to graph topology, with applications such as Web query suggestion [20], recommendation [6] and link prediction [22]. However, hitting time has one drawback: its value range is $[1, +\infty)$. That is, if there is no path between two nodes u and v, the hitting time from v to u is $+\infty$. This also results in infinite value of $\rho(V_q)$ (Eq. (1)). In this work, we propose decayed hitting time, which inverses the range of $[1, +\infty)$ to $[0, 1]$. The meaning of hitting time also changes slightly.

TABLE 1
Common Notations in this paper.

n	Number of nodes in graph G
$\mathbf{P}$	Transition probability matrix of G
q	The event
V_q	The set of nodes having q
m	Size of V_q
$\widehat{V}_m$	A random set of m nodes
$\tilde{h}(v, B)$	Decayed hitting time from node v to node set B
$\rho(V_q)/\tilde{\rho}(V_q)$	Correlation score/significance score of q
$E[\rho(\widehat{V}_m)]/Var[\rho(\widehat{V}_m)]$	Expectation/Variance of the correlation of a randomly selected set of m nodes

3.2 Decayed Hitting Time

Decayed Hitting Time (DHT) is defined as follows

$$\tilde{h}(v_i, B) = \sum_{t=1}^{\infty} e^{-(t-1)} \Pr(T_B = t | x_0 = v_i). \quad (6)$$

For DHT, when nodes in B are close to v_i, $\tilde{h}(v_i, B)$ will be high. The reason for the substitution of the exponentially decaying term for the number of steps is twofold. First, by doing this we avoid the infinite measure score problem mentioned above. Second, in our problem setting, we want to emphasize the importance of neighbors and discount the importance of longer range weak relationships. Although hitting time is shown to be empirically effective in practical applications [6], [22], solving a linear system for a large scale graph can be computationally expensive (i.e. $O(n^3)$). To address this issue, Sarkar *et al.* proposed *truncated hitting time* (THT) [24] and developed sampling techniques to approximate hitting time [25]. However, a drawback of THT is that it sets an arbitrary upper bound T (the truncated path length) for the actual hitting time. For THT, although long paths have small probabilities, they have high weights, i.e. their lengths. Therefore, the contribution of these long paths to the hitting time measure may not be negligible. In an extreme case where v_i cannot reach any node in B, THT will return T, while the true hitting time is $+\infty$! In DHT, longer paths not only have lower probabilities, but also have lower weights, i.e. $e^{-(t-1)}$. Hence, we can properly bound the contribution of long paths. This facilitates approximating DHT. We summarize the common notations used throughout this paper in Table 1.

4 Computing Decayed Hitting Time

We develop techniques for efficiently estimating DHT. In Section 4.1 we propose an iterative matrix-vector multiplication algorithm for estimating DHT. Lemma 1 and Theorem 1 give bounds for real DHT after t iterations. In Section 4.2 we present a sampling algorithm for estimating DHT and develop its bounds in Theorem 2 and 3. Finally Section 4.3 analyzes the complexity of the two approximation algorithms.

4.1 Iterative Approximation

From the definition in Eq. (6), we know if we can obtain $\Pr(T_B = t | x_0 = v_i), t = 1, 2, \ldots$, then we can compute $\tilde{h}(v_i, B)$. Let $\Pr(T_{\bar{B},v_j} = t | x_0 = v_i)$ be the probability that the random walk starting from v_i hits v_j after t steps without visiting any node in B. We have

$$\Pr(T_B = t | x_0 = v_i) = \sum_{v_j \in B} \Pr(T_{\bar{B},v_j} = t | x_0 = v_i).$$

The problem becomes how to compute $\Pr(T_{\bar{B},v_j} = t | x_0 = v_i)$. In particular, we have

$$\Pr(T_{\bar{B},v_j} = t | x_0 = v_i) = \sum_{v_k \notin B} \Pr(T_{\bar{B},v_k} = t-1 | x_0 = v_i) p_{kj},$$

which implies that one can first get to $v_k \notin B$ using $t-1$ steps (without visiting any $v_j \in B$), and then move from v_k to v_j with probability p_{kj}. It takes the sum over all possible v_k's. Therefore, we can derive the following iterative computation method: let $\mathbf{P}_B$ be a modification of $\mathbf{P}$ where rows corresponding to the nodes in B are set to zeros. Let $\mathbf{u}_t$ be a $n \times 1$ vector containing $\Pr(T_{\bar{B},v_k} = t | x_0 = v_i)$ as its k-th element. $\mathbf{u}_0$ is the vector with i-th element set to 1 and all other elements to 0. One can easily verify $\mathbf{u}_t = \left(\mathbf{P}_B^T\right)^t \mathbf{u}_0$. In fact, $\mathbf{P}_B$ and $\mathbf{u}_0$ define the corresponding random walk model for computing DHT from v_i to B. Let $\mathbf{z}_B$ be the vector with elements corresponding to nodes in B set to 1 and all other elements to 0. We can rewrite Eq. (6) as

$$\tilde{h}(v_i, B) \quad = \quad e^0 \mathbf{z}_B^T \mathbf{u}_1 + e^{-1} \mathbf{z}_B^T \mathbf{u}_2 + \ldots$$

We can iteratively compute $\mathbf{u}_1, \mathbf{u}_2, \ldots$ and accumulate elements corresponding to nodes in B from these vectors (multiplied by $e^0, e^{-1}, \ldots$ respectively). If we stop after a number of iterations, it results in an estimate of the actual DHT. In the remaining part of this section, we derive bounds for DHT from such an estimate using Lemma 1 and Theorem 1. We use $d_B(v_k)$ to denote the number of neighbors of v_k which are in B and $\lambda_{kB} = d_B(v_k)/d(v_k)$ to denote the corresponding fraction.

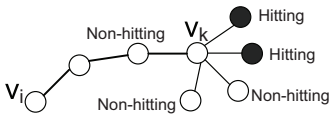


Fig. 3. Bounds for One Path

Lemma 1: Let $pa_{i \rightsquigarrow k}^l$ be a length-l path from v_i to v_k which has not yet hit any node in B. Let $\Pr(pa_{i \rightsquigarrow k}^l)$ be the probability that a random walk takes this path. We define subpaths of $pa_{i \rightsquigarrow k}^l$ as length-$(l+1)$ paths sharing $pa_{i \rightsquigarrow k}^l$ as a prefix. The contribution of $pa_{i \rightsquigarrow k}^l$ to $\tilde{h}(v_i, B)$ is upper bounded by $\lambda_{kB} \Pr(pa_{i \rightsquigarrow k}^l) e^{-l} + (1 - \lambda_{kB}) \Pr(pa_{i \rightsquigarrow k}^l) e^{-(l+1)}$ and lower bounded by $\lambda_{kB} \Pr(pa_{i \rightsquigarrow k}^l) e^{-l}$.

Proof: Since $pa_{i \rightsquigarrow k}^l$ has not hit any node in B, the probability $\Pr(pa_{i \rightsquigarrow k}^l)$ will be further distributed to subpaths of $pa_{i \rightsquigarrow k}^l$. By querying the neighbors of v_k, we

know the probability $\lambda_{kB}\Pr(pa^l_{i\rightsquigarrow k})$ will be distributed on length-$(l+1)$ hitting subpaths which have certain contribution $\lambda_{kB}\Pr(pa^l_{i\rightsquigarrow k})e^{-l}$. The remaining probability $(1-\lambda_{kB})\Pr(pa^l_{i\rightsquigarrow k})$ will be distributed on length-$(l+1)$ non-hitting subpaths, as illustrated in Fig. 3. The contribution of this part is lower bounded by 0 and upper bounded by $(1-\lambda_{kB})\Pr(pa^l_{i\rightsquigarrow k})e^{-(l+1)}$ (i.e. all subpaths of those length-$(l+1)$ non-hitting subpaths of $pa^l_{i\rightsquigarrow k}$ hit targets). Combining the two parts, the conclusion follows. □

Theorem 1: Let $\tilde{h}_t(v_i,B) = e^0\mathbf{z}_B^T\mathbf{u}_1 + \cdots + e^{-(t-1)}\mathbf{z}_B^T\mathbf{u}_t$ be the estimate of $\tilde{h}(v_i,B)$ after $\mathbf{u}_t$ is computed. Then $\tilde{h}(v_i,B)$ can be bounded as follows

$$\tilde{h}_t(v_i,B) + \sum_{v_k\notin B}\Pr(T_{\bar{B},v_k}=t|x_0=v_i)\lambda_{kB}e^{-t} \le \tilde{h}(v_i,B)$$
$$\le \tilde{h}_t(v_i,B) + \sum_{v_k\notin B}\Pr(T_{\bar{B},v_k}=t|x_0=v_i)\Big(\lambda_{kB}e^{-t}$$
$$+ (1-\lambda_{kB})e^{-(t+1)}\Big).$$

Proof: We can view the random walk process defined by computing $\tilde{h}(v_i,B)$ as distributing probability 1 onto all paths which start from v_i and end once a node $v_j\in B$ is hit. We use paths_{iB} to denote the set of all such paths. Examining the definitions of $\mathbf{u}_t$ and $\mathbf{z}_B$, we can find $\tilde{h}_t(v_i,B)$ is the contribution of all paths in paths_{iB} whose lengths are less than or equal to t. We use $\mathrm{paths}_{iB}^{1\rightarrow t}$ to denote the set of these paths. Paths in $\mathrm{paths}_{iB}^{1\rightarrow t}$ have certain contribution to $\tilde{h}(v_i,B)$. In the meantime, let $\mathrm{paths}_{i\bar{B}}^{t}$ denote the set of all length-t paths starting from v_i which have not yet hit any node in B. The probabilities accounted for by paths in $\mathrm{paths}_{iB}^{1\rightarrow t}$ and $\mathrm{paths}_{i\bar{B}}^{t}$ are $\Pr(\mathrm{paths}_{iB}^{1\rightarrow t}) = \sum_{v_j\in B}\sum_{l=1}^{t}\Pr(T_{\bar{B},v_j}=l|x_0=v_i)$ and $\Pr(\mathrm{paths}_{i\bar{B}}^{t}) = \sum_{v_k\notin B}\Pr(T_{\bar{B},v_k}=t|x_0=v_i)$, respectively. By examining the iterative computation of $\mathbf{u}_t$, one can verify that $\Pr(\mathrm{paths}_{iB}^{1\rightarrow t}) + \Pr(\mathrm{paths}_{i\bar{B}}^{t}) = 1$. It means $\Pr(\mathrm{paths}_{i\bar{B}}^{t})$ is the remaining probability that has not been contributed to $\tilde{h}(v_i,B)$. According to Lemma 1, we can derive lower and upper bounds of the contribution for each $pa\in\mathrm{paths}_{i\bar{B}}^{t}$. Aggregating those bounds and adding the certain contribution of $\mathrm{paths}_{iB}^{1\rightarrow t}$, the conclusion follows. □

4.2 A Sampling Algorithm for $\tilde{h}(v_i,B)$

We propose a standard Monte Carlo sampling method for estimating $\tilde{h}(v_i,B)$. A straightforward sampling scheme is as follows: we run c independent random walk simulations from v_i and in each random walk we stop when a node in B is encountered. Suppose these random walks' path lengths are $l_1,\ldots,l_c$. Then we use the average $\overline{\tilde{h}(v_i,B)} = \sum_{j=1}^{c}e^{-(l_j-1)}/c$ as the estimate of $\tilde{h}(v_i,B)$. However, this scheme is not a wise choice due to the following two reasons: 1) if we cannot reach any node in B from v_i, the random walk will never stop; 2) for a large scale graph, if we do not impose a maximum number of steps that a random walk can take, the sampling algorithm will be time consuming. In fact, since we adopt an exponentially damping factor (i.e. $e^{-(t-1)}$), the contribution of long paths are negligible.

With the above concerns, we adopt a variant sampling scheme: we run c independent random walk simulations from v_i and in each random walk we stop when a node in B is visited or a maximum number of s steps is reached. We provide bounds for $\overline{\tilde{h}(v_i,B)}$ by this sampling scheme in Theorem 2.

Theorem 2: Consider a simulation of c independent random walks from node v_i with a maximum number of s steps for each random walk. Suppose out of c runs, c_h random walks hit a node in B and the corresponding path lengths are $l_1,\ldots,l_{c_h}$. Let $\bar{c}=c-c_h$ be the number of random walks which reach s steps and do not hit any node in B. Then the sample mean $\overline{\tilde{h}(v_i,B)}$ in the sampling scheme without the constraint of maximum number of steps can be bounded as follows

$$\frac{\sum_{j=1}^{c_h}e^{-(l_j-1)}}{c} \le \overline{\tilde{h}(v_i,B)} \le \frac{\sum_{j=1}^{c_h}e^{-(l_j-1)}+e^{-s}\bar{c}}{c}.$$

Proof: In the sampling scheme with the constraint of maximum number of steps, only c_h random walks have certain contribution to $\overline{\tilde{h}(v_i,B)}$. Hence, we turn to the bounds for the contribution of the remaining $\bar{c}$ random walks which do not hit any node in B. For each of these $\bar{c}$ random walks, the contribution to $\overline{\tilde{h}(v_i,B)}$ is upper bounded by e^{-s} (i.e. hitting a node in B at $(s+1)$-th step). Aggregating those $\bar{c}$ random walks, we have $\overline{\tilde{h}(v_i,B)} \le (\sum_{j=1}^{c_h}e^{-(l_j-1)}+e^{-s}\bar{c})/c$. A lower bound for the contribution of those $\bar{c}$ random walks is 0. This leads to $\sum_{j=1}^{c_h}e^{-(l_j-1)}/c \le \overline{\tilde{h}(v_i,B)}$. □

We use $\tilde{h}'_{iB}$ and $\tilde{h}''_{iB}$ to represent the above lower and upper bounds for $\overline{\tilde{h}(v_i,B)}$, respectively. The following theorem provides the lower bound for the sample size c in order to obtain an ϵ-correct answer for $\tilde{h}(v_i,B)$ with respect to $[\tilde{h}'_{iB},\tilde{h}''_{iB}]$ with probability $1-\delta$.

Theorem 3: Suppose we simulate c independent random walks for estimating $\tilde{h}(v_i,B)$ and impose a maximum number of s steps for each random walk. Then for any $\epsilon>0$ and $\delta>0$, in order to obtain $\Pr(\tilde{h}'_{iB}-\epsilon \le \tilde{h}(v_i,B) \le \tilde{h}''_{iB}+\epsilon) \ge 1-\delta$, c should be at least $\frac{1}{2\epsilon^2}\ln(\frac{2}{\delta})$.

Proof: From the sampling scheme without the constraint of maximum number of steps, we have $\overline{\tilde{h}(v_i,B)} = \sum_{j=1}^{c}e^{-(l_j-1)}/c$ where l_j is the path length of the j-th random walk. It is obvious that $E(\overline{\tilde{h}(v_i,B)}) = \tilde{h}(v_i,B)$. Since random walks are independent and $0\le e^{-(l_j-1)}\le 1$ for $j=1,\ldots,c$, according to Hoeffding's inequality [13] we have $\Pr(|\overline{\tilde{h}(v_i,B)}-\tilde{h}(v_i,B)|\le\epsilon)\ge 1-2e^{-2c\epsilon^2}$. From Theorem 2 we know $\tilde{h}'_{iB} \le \overline{\tilde{h}(v_i,B)} \le \tilde{h}''_{iB}$. Therefore, we have

$$-\epsilon \le \tilde{h}(v_i,B) - \overline{\tilde{h}(v_i,B)} \le \epsilon$$
$$\Leftrightarrow\quad \tilde{h}'_{iB}-\epsilon \le \tilde{h}(v_i,B) \le \tilde{h}''_{iB}+\epsilon$$

We can further obtain

$$\begin{aligned} & \Pr(\tilde{h}'_{iB} - \epsilon \leq \tilde{h}(v_i, B) \leq \tilde{h}''_{iB} + \epsilon) \\ = \; & \Pr(|\overline{\tilde{h}(v_i, B)} - \tilde{h}(v_i, B)| \leq \epsilon) \geq 1 - 2e^{-2c\epsilon^2}. \end{aligned}$$

Setting $1 - 2e^{-2c\epsilon^2} \geq 1 - \delta$ gives us $c \geq \frac{1}{2\epsilon^2}\ln(\frac{2}{\delta})$. □

4.3 Complexity

Hereafter, we use *Iterative-alg* and *Sampling-alg* to denote the iterative algorithm and sampling algorithm developed above, respectively. Suppose we use adjacency lists to store graphs and matrices. The space complexity of the two algorithms is $O(|E|)$. The major time-consuming parts of Iterative-alg are the iterative matrix-vector multiplication and the construction of $\mathbf{P}_B$. The corresponding time complexity is $O(t|E|)$, where t is the number of iterations. For Sampling-alg, the major time cost in each random walk step is the membership judgement of the current node v to B. We can either sort B and use binary search, or build an index array for B. The corresponding time costs are $O(cs\log|B| + |B|\log|B|)$ and $O(cs + |V|)$, respectively.

5 Assessing Structural Correlations

In this section, we first propose a sampling method for estimating $\rho(V_q)$ and develop the lower bound for the sample size in order to get ϵ'-correct answers (Section 5.1). Then we describe our methodology for assessing the significance of the observed ρ score, i.e. $\tilde{\rho}(V_q)$.

5.1 Estimating $\rho(V_q)$

To compute $\rho(V_q)$, we need to compute $\tilde{h}(v_i, V_q \setminus \{v_i\})$ for all $v_i \in V_q$. However, for large scale graphs, V_q may also have a large size, posing a challenge for efficient computation of $\rho(V_q)$. Although these $\tilde{h}$'s are dependent on each other, they form a finite population. We can still use sampling techniques to efficiently estimate $\rho(V_q)$ by applying Hoeffding's inequality for finite populations [13]. Specifically, we randomly select c' nodes from V_q, denoted by $v_1, \ldots, v_{c'}$, to estimate their DHTs to the remaining nodes and take the average $\overline{\rho(V_q)}$ as an estimate for $\rho(V_q)$. Here we can use either Iterative-alg or Sampling-alg for estimating each $\tilde{h}(v_i, V_q \setminus \{v_i\})$. If Iterative-alg is used, from Theorem 1 we obtain bounds for each $\tilde{h}(v_i, V_q \setminus \{v_i\})$ in the sample set. Aggregating those bounds, we can get bounds for $\overline{\rho(V_q)}$. Following the same manner for the proof of Theorem 3 and applying Hoeffding's inequality for finite populations [13], we can obtain the lower bound for c' in order to obtain an ϵ'-correct answer. We omit the details due to space limitation. When Sampling-alg is used, we provide the lower bound for c' in the following theorem.

Theorem 4: Suppose we randomly select c' nodes from V_q to estimate their DHTs to the remaining nodes and take the average $\overline{\rho(V_q)}$ as an estimate of $\rho(V_q)$. For the sake of clarity, let $B_i = V_q \setminus \{v_i\}$. Suppose we have used Sampling-alg to obtain an ϵ-correct answer for each $\tilde{h}(v_i, B_i)$ $(i = 1, \ldots, c')$ with respect to $[\tilde{h}'_{iB_i}, \tilde{h}''_{iB_i}]$. Then for any $\epsilon' > 0$ and $\delta' > 0$, in order to obtain

$$\Pr\left(\frac{\sum_{i=1}^{c'} \tilde{h}'_{iB_i}}{c'} - \epsilon - \epsilon' \leq \rho(V_q) \leq \frac{\sum_{i=1}^{c'} \tilde{h}''_{iB_i}}{c'} + \epsilon + \epsilon'\right) \geq 1 - \delta',$$

c' should satisfy $(1-\delta)^{c'}(1 - 2e^{-2c'\epsilon'^2}) \geq 1 - \delta'$.

Proof: From the conditions we have

$$\Pr(\tilde{h}'_{iB_i} - \epsilon \leq \tilde{h}(v_i, B_i) \leq \tilde{h}''_{iB_i} + \epsilon) \geq 1 - \delta, \quad i = 1, \ldots, c'.$$

Notice $\overline{\rho(V_q)} = \sum_{i=1}^{c'} \tilde{h}(v_i, B_i)/c'$. Since $\tilde{h}$'s are estimated independently, multiplying those probability inequalities together we obtain

$$\Pr\left(\frac{\sum_{i=1}^{c'} \tilde{h}'_{iB_i}}{c'} - \epsilon \leq \overline{\rho(V_q)} \leq \frac{\sum_{i=1}^{c'} \tilde{h}''_{iB_i}}{c'} + \epsilon\right) \geq (1-\delta)^{c'}.$$

Since $0 \leq \tilde{h}(v_i, B_i) \leq 1$ for $i = 1, \ldots, c'$, according to Hoeffding's inequality for finite populations [13] we know $\Pr(|\overline{\rho(V_q)} - \rho(V_q)| \leq \epsilon') \geq 1 - 2e^{-2c'\epsilon'^2}$. Since the underlying estimation of $\overline{\rho(V_q)}$ is independent from Hoeffding bounds, we have

$$\begin{aligned} \Pr\Big(\frac{\sum_{i=1}^{c'} \tilde{h}'_{iB_i}}{c'} - \epsilon - \epsilon' \leq \rho(V_q) \leq \frac{\sum_{i=1}^{c'} \tilde{h}''_{iB_i}}{c'} + \epsilon + \epsilon'\Big) \\ \geq (1-\delta)^{c'}(1 - 2e^{-2c'\epsilon'^2}). \end{aligned}$$

Setting $(1-\delta)^{c'}(1 - 2e^{-2c'\epsilon'^2}) \geq 1 - \delta'$, we get the inequality c' should satisfy. Note δ should be large enough so that $(1-\delta)^{c'}(1 - 2e^{-2c'\epsilon'^2})$ can go beyond $1 - \delta'$ as c' increases. □

5.2 Estimating the Significance of $\rho(V_q)$

After obtaining the estimate of $\rho(V_q)$, we need to measure the deviation of $\rho(V_q)$ from the expected ρ value of $\widehat{V}_m$ (i.e. a set of randomly selected m nodes from the graph), in order to distinguish structural correlations from random results. In particular, we have

$$E[\rho(\widehat{V}_m)] = \frac{\sum_{V_m \subseteq V} \rho(V_m)}{C_n^m}, \tag{7}$$

where V_m is any set of m nodes. The ideal solution is to obtain the distribution of $\rho(\widehat{V}_m)$ and use the ratio between the number of node sets with size m whose ρ values are greater than or equal to $\rho(V_q)$ and C_n^m as the significance score for q. However, for a large scale graph it is very hard to get the distribution since C_n^m is very large. Here we propose an approximation method. Notice $\rho(\widehat{V}_m)$ is defined as the average of $\tilde{h}(v_i, \widehat{V}_m \setminus \{v_i\})$ where $v_i \in \widehat{V}_m$. If we assume these $\tilde{h}$'s are independent, according to Central Limit Theorem, $\rho(\widehat{V}_m)$ can be approximated by a normal distribution, where $Var[\rho(\widehat{V}_m)] = Var[\tilde{h}(v_i, \widehat{V}_m \setminus \{v_i\})]/m$. If we obtain $E[\rho(\widehat{V}_m)]$ and $Var[\rho(\widehat{V}_m)]$, we can calculate the **adjusted structural correlation** $\tilde{\rho}$ for q as follows

$$\tilde{\rho}(V_q) = \frac{\rho(V_q) - E[\rho(\widehat{V}_m)]}{\sqrt{Var[\rho(\widehat{V}_m)]}}. \tag{8}$$

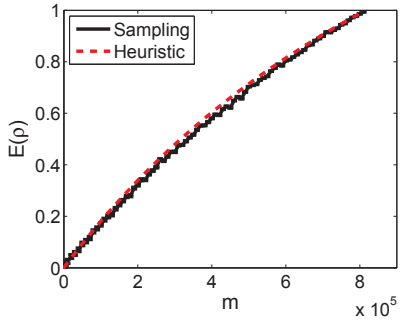


Fig. 4. Comparison of sampling and geometric distribution heuristic for estimating $E[\rho(\widehat{V}_m)]$.

This idea is similar to using z-scores to assess the significance of data mining results [11]. Eq. (8) can be used to derive the significance of q for a hypothesis that q is not randomly distributed over G. The independence assumption should work well as long as the dependence between those $\tilde{h}$'s is weak. This could be true because DHT focuses on local areas of the graph. Each black node only has high dependence on nearby black nodes. In the remaining part of this section, we provide efficient methods for estimating $E[\rho(\widehat{V}_m)]$ and $Var[\rho(\widehat{V}_m)]$.

We propose two methods to efficiently estimate $E[\rho(\widehat{V}_m)]$. The first one is a sampling method. Eq. (7) suggests sampling ρ values for different sets of m nodes. However, computing ρ is costly since m could be large. Using the sampling method proposed in Section 5.1 to estimate ρ introduces another layer of sampling and would introduce more estimation error. Therefore, we propose to sample $\tilde{h}$ directly. Specifically, we have

$$E[\rho(\widehat{V}_m)] = \frac{\sum_{V_m} \frac{\sum_{v_i \in V_m} \tilde{h}(v_i, V_m \setminus \{v_i\})}{m}}{C_n^m} = \frac{\sum_{v_i, V_{m-1}} \tilde{h}(v_i, V_{m-1})}{C_n^1 C_{n-1}^{m-1}}$$

where $v_i \notin V_{m-1}$. It means $E[\rho(\widehat{V}_m)]$ is equal to the expected DHT from a random v_i to a random V_{m-1} which does not contain v_i. Thus, we can directly sample (v_i, V_{m-1}) pairs and take the average DHT among those pairs as an estimate of $E[\rho(\widehat{V}_m)]$. Given that we have already obtained ϵ-correct $\tilde{h}$'s by Theorem 3, we can derive a very similar sample size lower bound in the same manner for Theorem 4's proof, by applying Hoeffding's inequality [13]. We omit the details due to space limitation. For a fixed graph, we can pre-compute $E[\rho(\widehat{V}_m)]$ for a number of different m values and employ interpolation to estimate $E[\rho(\widehat{V}_m)]$ for arbitrary m.

Alternatively, we can derive an approximation method for $E[\rho(\widehat{V}_m)]$ by a *geometric distribution*. This approximation method is empirical. A geometric distribution is a probability distribution of the number t of Bernoulli trials needed to get one success. When we randomly generate $\widehat{V}_m$, each node of G has probability $\frac{m}{n}$ to be chosen. In the following discussion, we assume each node of G is chosen independently with probability $\frac{m}{n}$. With this relaxation, $|\widehat{V}_m|$ becomes a binomial random variable with m as its expected value. Consider we start from a node $v_i \in \widehat{V}_m$ to hit the remaining nodes in $\widehat{V}_m$. Let $p = \frac{m-1}{n-1}$ be the probability of each node other than v_i being in $\widehat{V}_m$. The probability that we first hit (i.e. stop) a target node after one step is $\sum_j p_{ij} p = p$. The probability that we stop after two steps is $\sum_{j,k} p_{ij} p_{jk} (1-p) p = (1-p)p$. We do not consider cases where the surfer comes back to v_i in this approximation. This forms a geometric distribution where the probability that we "succeed" after t steps is $(1-p)^{t-1}p$. By the definition of DHT (i.e. Eq. (6)), $\tilde{h}(v_i, \widehat{V}_m \setminus \{v_i\})$ is actually the expectation of $e^{-(t-1)}$ under the geometric distribution described above:

$$\tilde{h}(v_i, \widehat{V}_m \setminus \{v_i\}) = \sum_{t=1}^{\infty} e^{-(t-1)} (1-p)^{t-1} p = \frac{p}{1 - e^{-1}(1-p)}. \quad (9)$$

Since v_i is an arbitrary node in $\widehat{V}_m$, we have

$$\rho(\widehat{V}_m) = \frac{p}{1 - e^{-1}(1-p)}. \quad (10)$$

Since we assume each node of G is chosen independently, the obtained $\rho(\widehat{V}_m)$ is an approximation of $E[\rho(\widehat{V}_m)]$. In case the graph contains 0-degree nodes, we just need to multiply Eq. (10) by the probability that a randomly selected node is not a 0-degree node. We empirically compare this heuristic approximation method with the sampling method on the DBLP co-author network. The results are shown in Fig. 4. Regarding the sampling method, we sample 1500 (v_i, V_{m-1}) pairs for each m and use Sampling-alg to estimate DHT. The error bars on the curve of the sampling method represent lower and upper bounds for the estimates of $E[\rho(\widehat{V}_m)]$. We can see that results obtained by sampling roughly fit the curve of the heuristic method. Therefore, we can either use sampling method and interpolation or the heuristic method to estimate $E[\rho(\widehat{V}_m)]$. In our experiments we employ the heuristic method.

Regarding $Var[\rho(\widehat{V}_m)]$, we also propose a sampling method. Directly estimating $Var[\rho(\widehat{V}_m)]$ by sample variance again requires computing ρ for each sampled V_m and is time consuming since m could be large. Recall that we assume $\tilde{h}(v_i, \widehat{V}_m \setminus \{v_i\})$'s in the numerator of the definition of $\rho(\widehat{V}_m)$ are independent. We approximate $Var[\rho(\widehat{V}_m)]$ by $Var[\tilde{h}(v_i, \widehat{V}_m \setminus \{v_i\})]/m$. For a given m, we just sample (v_i, V_{m-1}) pairs and take the sample variance of the corresponding DHTs divided by m as an estimate of $Var[\rho(\widehat{V}_m)]$. Again, pre-computation and interpolation can be used here to estimate $Var[\rho(\widehat{V}_m)]$ for arbitrary m.

We summarize the gScore framework in Algorithm 1. Step 2 and 4 can use either Iterative-alg or Sampling-alg to estimate DHT.

6 Dynamically Measuring Correlations

The ρ measure in Eq. (1) captures an event's overall degree of correlation. However, an event is essentially

Algorithm 1: The gScore framework

Offline Phase
1 Choose a set of m values.
2 For each m value, sample (v_i, V_{m-1}) pairs to estimate $Var[\tilde{h}(v_i, V_{m-1})]/m$ as the variance estimate.
Online Phase
3 Randomly choose a sample of c nodes from V_q.
4 For each of c nodes, estimate its DHT to the remaining nodes in V_q.
5 Compute sample mean $\overline{\rho(V_q)} = \sum_{i=1}^{c} \tilde{h}(v_i, V_q \setminus \{v_i\})/c$.
6 Estimate $E[\rho(\widehat{V}_m)]$ by Eq. (10).
7 Estimate $Var[\rho(\widehat{V}_m)]$ by interpolating the pre-computed variances.
8 Compute approximate z-score by Eq. (8).

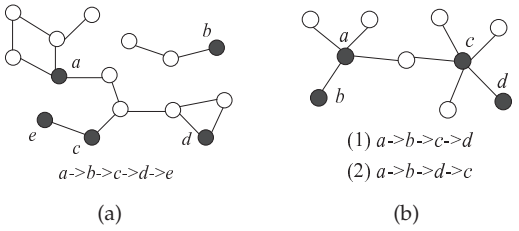


Fig. 5. Two examples illustrating the drawbacks of ρ for dynamically measuring structural correlations.

a temporal phenomenon and people may want to know the detailed evolutionary patterns of events. For example, some products may exhibit high correlations in specific periods of a year[1], which could be helpful for setting up fine-grained promotion strategies for those products. In this section, we develop a measure to capture the dynamic aspect of events.

6.1 A Dynamic Measure

We reformulate the event set as $V_q = \{v_1, v_2, \ldots, v_t\}$ where the node subscripts represent time steps. The idea is that at time step l, we compute the proximity of node v_l to the previous nodes $v_1, \ldots, v_{l-1}$. Specifically, the measure is defined as

$$\rho_l^D(V_q) = \tilde{h}(v_l, \{v_1, \ldots, v_{l-1}\}). \tag{11}$$

In this way, the correlation scores form a time series where we have a correlation measurement after each node turns "black".

We can also capture the time factor by a straightforward method which makes use of ρ in Eq. (1): for each time step l we take $\{v_1, \ldots, v_l\}$ as an event set and compute the ρ score. However, this method cannot well capture the correlation change in each time step. To illustrate this point, we show two examples in Fig. 5. The node sequences for events are shown below the two graphs. In Fig. 5(a), before node e turns black the event was randomly occurring on the graph. Node e is probably influenced by node c since c is the only neighbor of e. The DHT from e to $\{a, b, c, d\}$ is 1, while the averaged DHT among the five nodes is still near those of random cases. Regarding Fig. 5(b), let us consider two different sequences which are different at the last step. If we compute the averaged DHT at the last step, the two sequences will have the same correlation score. However, sequence (2) should have a lower correlation score than sequence (1) regarding the last step, since c is a well connected node and is less likely to be influenced by d. ρ_l^D can reflect this difference.

1. For instance, we find game cards tend to exhibit high correlations in Spring Festival, the most important holiday in China, since students have more time to play and it is easier to invite their friends to play the game.

6.2 From Measure to Significance

For the dynamic measure, we also need to assess the significance of the observed score. The null hypothesis is "node v_{l+1} at time step $l+1$ is just randomly selected from $V \setminus \{v_1, \ldots, v_l\}$". Since the finite population of this distribution is not prohibitively large (the size is equal to the number of nodes not in $\{v_1, \ldots, v_l\}$), we directly estimate the p-value of the observed score.

We develop two methods for this task. The first method is based on matrix-vector multiplications. Let $B_l = \{v_1, \ldots, v_l\}$ contain the nodes of event q up to the l-th step. Let $\mathbf{P}_{B_l}$ be a modification of the transition probability matrix $\mathbf{P}$ where rows corresponding to nodes in B_l are set to zeros, and $\Pr(T_{\bar{B}_l, v_j} = t | x_0 = v_i)$ be the probability that the random walk starting from v_i hits v_j after t steps without visiting any node in B_l. We can expand $\Pr(T_{\bar{B}_l, v_j} = t | x_0 = v_i)$ as $\sum_{v_k \notin B_l} \Pr(T_{\bar{B}_l, v_k} = t-1 | x_0 = v_i) p_{kj}$. It is easy to verify that the (j, i)-element of $(\mathbf{P}_{B_l}^T)^t$ is equal to $\Pr(T_{\bar{B}_l, v_j} = t | x_0 = v_i)$. Therefore, we can compute the estimates of DHTs from all nodes not in B_l to B_l by the following equation:

$$\tilde{\mathbf{h}}^T = e^0 \mathbf{z}_{B_l}^T \mathbf{P}_{B_l}^T + e^{-1} \mathbf{z}_{B_l}^T (\mathbf{P}_{B_l}^T)^2 + \cdots + e^{-(t-1)} \mathbf{z}_{B_l}^T (\mathbf{P}_{B_l}^T)^t, \tag{12}$$

where $\mathbf{z}_{B_l}$ has the same definition as $\mathbf{z}_B$ in Section 4.1 and $\tilde{\mathbf{h}}$ is a vector containing the estimates of DHTs. Eq. (12) indicates that we can iteratively compute $\mathbf{z}_{B_l}^T (\mathbf{P}_{B_l}^T)^t$ to obtain the distribution of the decayed hitting time from a randomly selected node $v \notin B_l$ to B_l. Given the distribution $\{\tilde{h}_1, \ldots, \tilde{h}_{n-|B_l|}\}$, the p-value of the observed score $\tilde{h}(v_{l+1}, B_l)$ in the $(l+1)$-th step is

$$\frac{1}{n - |B_l|} |\{i | \tilde{h}_i \geq \tilde{h}(v_{l+1}, B_l)\}| \tag{13}$$

The second method is a sampling method: we simply sample k nodes from $V \setminus B_l$ and estimate their DHTs to B_l. These k DHTs form an empirical distribution for the decayed hitting time from a randomly selected node $v \notin B_l$ to B_l. Supposing that we independently sample nodes and that k is large enough[2] so that the sampled DHTs give a good approximation of the real distribution, the empirical p-value of $\tilde{h}(v_{l+1}, B_l)$ is estimated similarly as in Eq. (13).

2. In practice, we find usually $k = 10000$ is enough to give a good approximation.

Compared to the sampling method, the matrix-vector multiplication method gives us a more accurate distribution. However, the matrix-vector method has a higher computational cost, $O(t|E|)$ in our case, where t is the number of iterations. When the graph is large, the sampling method could be more efficient since it only estimates DHTs for a sample of nodes. In experiments we adopt the sampling method for significance estimation.

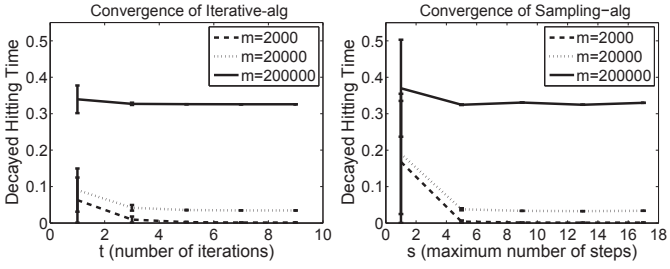
Smoothing In practice we find correlated black nodes and uncorrelated black nodes tend to appear alternately, making the evolution curve fluctuate drastically. To better facilitate finding useful correlation evolutionary patterns, we propose to use smoothing techniques for drawing the evolution curve. Specifically, we put a sliding window of size r over $\{v_1, v_2, \ldots, v_t\}$ and compute the average $(\sum_{i=(l-\frac{r-1}{2})}^{l+\frac{r-1}{2}} \rho_i^D(V_q))/r$ as the score in the l-th step (we truncate the range of index i so that i is within $[1, t]$). Two schemes for deciding the window size r can be adopted, according to specific applications: (1) *equal depth*, where every window contains the same number of nodes; (2) *equal time*, where windows span time periods of the same length. Intuitively, if $\{v_1, v_2, \ldots, v_t\}$ are evenly distributed along the time line, we can use either equal depth or equal time. There is no big difference. If there are many big gaps in the time line distribution, it would be better to use equal time since event nodes separated by time gaps could exhibit different correlation patterns.

7 Empirical Studies

This section presents experimental results on three real world datasets: DBLP, TaoBao and Twitter, as well as synthetic events. We first investigate the performance of two DHT approximation algorithms proposed in Section 4. We then verify gScore's effectiveness on synthetic events. In Section 7.4 we report interesting correlated and uncorrelated products discovered from the TaoBao network. Section 7.5 focuses on dynamic evolution of structural correlation. Finally, we analyze the scalability of gScore with the Twitter network. All experiments are run on a PC with Intel Core i7 CPU and 12GB memory. The source code of gScore can be downloaded at "http://www.cs.ucsb.edu/∼xyan/software/gScore.html".

7.1 Datasets

DBLP The DBLP snapshot was downloaded on Oct. 5th, 2010 (http://www.informatik.uni-trier.de/∼ley/db). Its paper records were parsed to obtain the co-author social graph. Keywords in paper titles are treated as events associated with nodes (authors) on the graph. The first time an author used a keyword was also recorded. It contains 815,940 nodes, 2,857,960 edges and 171,614 events.



(a) (b)

Fig. 6. Exploring the convergence of (a) Iterative-alg, and (b) Sampling-alg.

TaoBao The TaoBao dataset was obtained from China's most famous customer-to-customer shopping Website named TaoBao (http://www.taobao.com). By the end of 2009, TaoBao has about 170 million users and 1 billion products. We extracted users from three cities (Beijing, Shanghai and Hangzhou) with their product purchase history, and constructed the friend social graph among them. It consists of 794,001 nodes, 1,370,284 edges. We selected 100 typical products from TaoBao to show the effectiveness of our measure.

Twitter The Twitter dataset has about 40 million nodes and 1.4 billion edges (http://twitter.com). We do not have events for this dataset. It is mainly used to test the scalability of gScore.

7.2 Performance of DHT Approximation

We investigate the convergence and running time of the two DHT approximation algorithms: Iterative-alg and Sampling-alg. Iterative-alg has one parameter (number of iterations t) and Sampling-alg has two parameters (maximum number of steps s and number of random walks c). For Iterative-alg, we investigate its converging speed with respect to t. For Sampling-alg, we find when $c > 600$, increasing c hardly improves the obtained bounds. Thus, we set $c = 600$ and investigate the converging speed of Sampling-alg with respect to s. The results are shown in Fig. 6 with various m values (the number of nodes that have the same event). For each m value, we randomly select a node v and a set B of $m-1$ nodes and apply the two algorithms to estimate $\tilde{h}(v, B)$. This process is repeated 50 times and the averaged results are reported. As shown in Fig. 6, both algorithms converge quickly after about 5 iterations. Note that Iterative-alg gives lower and upper bounds for $\tilde{h}$, while Sampling-alg gives bounds for an estimate of $\tilde{h}$, i.e. $\overline{\tilde{h}}$. Comparing Fig. 6(a) and Fig. 6(b), one can find that the two algorithms converge to roughly the same values. It means empirically Sampling-alg provides a good estimation of $\tilde{h}$.

The running time of Iterative-alg and Sampling-alg for estimating one DHT under different m values is shown in Fig. 7. For Iterative-alg, we report the running time for $t = 1$ and $t = 9$ and for Sampling-alg, $s = 1$ and

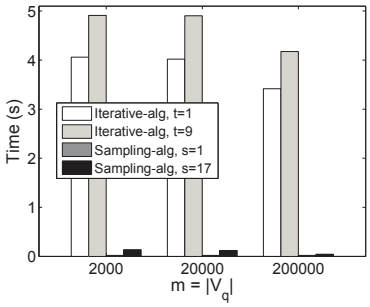


Fig. 7. Comparison of Iterative-alg and Sampling-alg with respect to the time used to estimate one DHT.

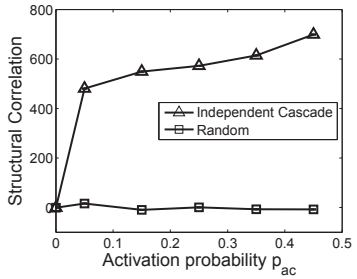
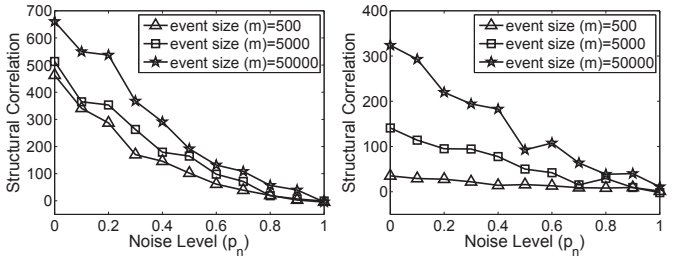


Fig. 8. Applying gScore on synthetic events.



(a) DHT (b) Shortest distance

Fig. 9. Comparison of DHT and pairwise shortest distance as the proximity measure by adding noises into the cascade model.

$s = 17$. It shows that Sampling-alg is much faster than Iterative-alg. Note that regarding Iterative-alg, the time cost of "s=9" is not 9 times of that of "s=1". This is because not only matrix-vector multiplication but also the construction of $\mathbf{P}_B$ account for time cost. In fact, Iterative-alg runs even faster when m increases: less rows of $\mathbf{P}$ are needed to construct the desired matrix. Since Sampling-alg is much faster than Iterative-alg and also provides reasonable estimates for DHTs, for the following experiments we employ Sampling-alg to estimate DHT. gScore also refers to Sampling-alg. Hereafter, we set $s = 12$ and $c = 600$.

7.3 Effectiveness on Synthetic Events

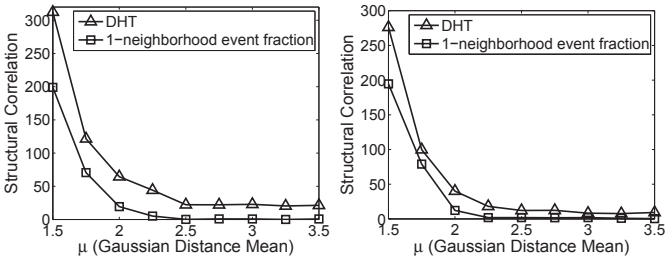
To evaluate the effectiveness of our measure, we generate synthetic events on the DBLP graph using the cascade model for influence spread [16]: at first a random set of 100 nodes is chosen as the initial V_q; then in each iteration nodes joining V_q in the last iteration can activate each currently inactive neighbor with probability p_{ac}; we stop when $|V_q| > 10000$. p_{ac} can be regarded as representing the level of participation in an event. Intuitively, higher p_{ac} would lead to higher correlation. *For all the following experiments, we report the significance estimates as the measure of structural correlation, i.e. $\tilde{\rho}$ in Eq. (8). $\tilde{\rho}$ can be regarded as approximate z-scores. Higher scores mean higher (more significant) correlations, while a score close to 0 indicates that there is no correlation.*

The results are shown in Fig. 8. "Random" means we expand the initial 100 random nodes with randomly selected nodes from the remaining nodes in order to match the corresponding event sizes of cascade model. We can see as p_{ac} increases, the curve of cascade model goes up, while that of "Random" remains around 0.

We further test the performance of gScore by adding noises to the above cascade model. p_{ac} is set to 0.2. Specifically, we break the correlation structure by relocating each black node to a random node in the remaining graph with probability p_n (noise level). $p_n = 1$ means all black nodes are randomly redistributed. We report results for different event sizes (m), i.e. spread levels.

gScore is applicable to other graph proximity measures. Here we also instantiate gScore with pairwise shortest distance for comparison. In this case, Eq. (1) becomes the average shortest distance among all pairs of black nodes. For large scale graphs, computing shortest distances for all pairs of black nodes is usually very costly. Pre-computing and storing pairwise shortest distances for the whole graph is not practical either. Hence, we sample black node pairs to estimate the correlation measure. By applying Hoeffding's inequality for finite populations [13], we can easily derive a lower bound for the sample size in order to get ϵ-correct answers. The significance estimation methodology in Section 5 is also applicable. The expectation of the correlation measure for $\widehat{V}_m$) is the average shortest path length of the graph. Its variance can be approximated by the variance of shortest path length divided by the event size m. We use sampling to estimate mean and variance. We use the reciprocal of shortest distances to avoid infinite distances when no path exists between two nodes.

We show results in Fig. 9. For a good proximity measure, the correlation significance should decrease smoothly, as the noise level increases. As we see, the curves of DHT gradually decrease with increasing noises and get around 0 when $p_n = 1$, indicating DHT can well capture structural correlations, while the curves of shortest distance are not stable and fluctuate a bit when increasing noises. The reason should be that pairwise shortest distance is affected by long distances among black nodes, as mentioned in Section 3. The relocation of one node will affect all remaining $m - 1$ nodes equally and the independent assumption in normal approximation and variance estimation may not work very well. In Fig. 9(b), we find the correlation scores for $m = 500$ is



(a) 2-hop neighborhood (b) 3-hop neighborhood

Fig. 10. Comparison of DHT and 1-neighborhood event fraction as the proximity measure by generating more general correlations in local neighborhoods.

TABLE 2
Structural correlation for top five correlated products in category "Laptops and tablets" in TaoBao.

#	Product	Bounds for $\tilde{\rho}$	ρ ($\times 10^{-2}$)	$\lvert V_q \rvert$
1	ThinkPad T400	$[554.43, 554.47]$	$[6.2396, 6.2400]$	47
2	Apple iPad	$[227.56, 227.57]$	$[6.7979, 6.7984]$	698
3	ThinkPad X200	$[91.39, 91.42]$	$[1.0799, 1.0802]$	60
4	Toshiba L600	$[20.36, 20.41]$	$[0.2009, 0.2014]$	31
5	ThinkPad T410	$[-1.13, -1.09]$	$[0.0004, 0.0009]$	72

much lower than that for $m = 50000$. This is also due to long distances. Recall that the cascade model chooses initial black nodes randomly, which means different threads of influence spread could be distant from one another. When m is small, long distances could show high impact.

We also compare DHT with the 1-neighborhood event fraction measure described in Section 2. We find the 1-neighborhood measure performs as well as DHT with the cascade model. This is because the cascade model always generates correlation patterns in 1-neighborhood. However, more general correlation patterns can occur in a graph, e.g. products can be attractive to specific communities in a graph, but purchases may not always occur among direct neighbors. We use a new model to generate more general correlation patterns: we first randomly select 500 nodes as seed black nodes; then $a\%$ nodes (minimum 1) in each seed node's k-hop neighborhood are painted black. Their distances to the seed node are distributed as gaussian with mean μ and variance 0.25. Distances out of range are reset to the nearest value within range. We explore $k = 2, 3$ and set $a = 0.1$. μ controls the average distance to seed nodes. The results are shown in Fig. 10. As μ increases, the curves of the 1-neighborhood measure drop to around 0 (no correlation), while those of DHT stay around 22 (2-hop) and 9 (3-hop). This means DHT can detect more general correlations, while the 1-neighborhood measure cannot. If the user only considers correlations in 1-hop neighborhoods, 1-neighborhood event fraction is preferred since it is more efficient.

TABLE 3
Structural correlation for top five correlated products in category "Other" in TaoBao.

#	Product	Bounds for $\tilde{\rho}$	$\lvert V_q \rvert$
1	Mamy Poko baby diapers	$[238.50, 238.51]$	4892
2	Beingmate Infant milk powder	$[227.71, 227.72]$	163
3	EVE game cards	$[198.56, 198.58]$	374
4	Mabinogi game cards	$[189.56, 189.58]$	446
5	Gerber cookies	$[149.51, 149.52]$	1491

7.4 Real Event Structural Correlation

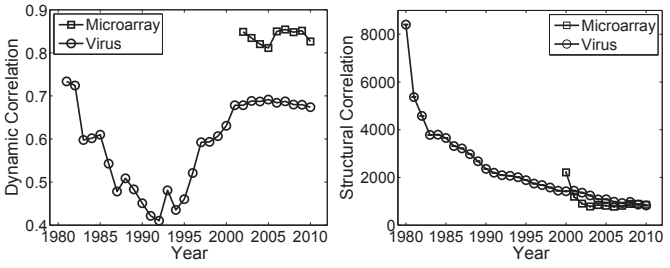
We apply gScore on real events occurring on graphs and report interesting highly correlated events and uncorrelated events. Using Eq. (8), we obtain an estimate (lower and upper bounds) of $\tilde{\rho}$ for each event. A ranked list of events can be generated according to these bounds. If the bounds of two events overlap, we increase sample numbers and the maximum steps to break a tie. For this experiment, we omit the results for DBLP keywords due to space limitation. Readers can refer to [12] for details. We group products from TaoBao into two categories: *Laptops and tablets* and *Other* and show top five products for each case. Before presenting the results, we would like to emphasize that our correlation findings are just for the specific social networks involved in this study.

TABLE 4
Structural correlation for the five most uncorrelated products in category "Other" in TaoBao.

#	Product	Bounds for $\tilde{\rho}$	$\lvert V_q \rvert$
1	Tiffany rings	[2.71,2.72]	1092
2	Jack&Jones suits	$[-0.48, -0.46]$	311
3	Ray-Ban sunglasses	$[-0.78, -0.77]$	4958
4	Swarovski anklets	$[-0.88, -0.84]$	72
5	Jack&Jones shirts	$[-3.28, -3.27]$	1606

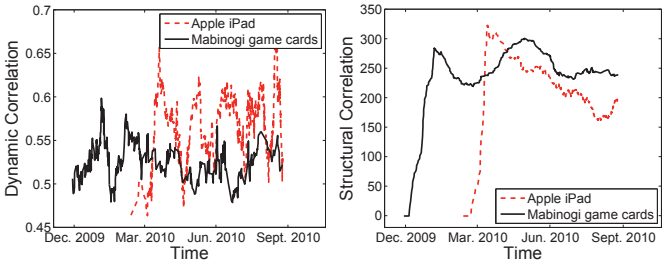
Table 2 shows the ranked lists for top five products in "Laptops and tablets". We also show ρ values in Table 2. ThinkPad and Apple products usually have high correlation with the underlying network, indicating there are fan communities for these brands. An interesting exception is ThinkPad T410, which is a new version of Thinkpad T400. In comparison with T400, its correlation score is very close to that of random cases. The reason may be that people in the fan community already bought T400 and they would not further buy a new version for T400 since they are quite similar and not cheap.

The ranked list for top five products from category "Other" is shown in Table 3. Here "EVE" and "Mabinogi" are two online games and players in China must buy *game cards* to obtain gaming time. We find products for infants, like diapers and powder tend to be correlated with the network. This indicates people tend to follow friends' recommendations when choosing this kind of products. Game card are also highly correlated with the network. Intuitively, playing with friends is an important attractive feature of online games.



(a) Dynamic measure (b) Static measure

Fig. 11. Correlation Evolution of two keywords in DBLP: (a) the dynamic measure, and (b) $\tilde{\rho}$ with evolving V_q.



(a) Dynamic measure (b) Static measure

Fig. 12. Correlation Evolution of two products in TaoBao: (a) the dynamic measure, and (b) $\tilde{\rho}$ with evolving V_q.

Finally, we show the $\tilde{\rho}$ scores for the five most uncorrelated products from category "Other" in Table 4. These products' scores are very close to those of random cases (some scores deviate a little from random cases due to estimation errors in variance). This indicates that for clothing and accessories, people usually follow their own preferences.

7.5 Structural Correlation Evolution

We apply the dynamic correlation measure ρ_l^D proposed in Section 6 on selected keywords and products from DBLP and TaoBao. We also compare ρ_l^D with the static measure ρ computed on the event occurrences before a time point. We demonstrate that ρ cannot reflect dynamic correlation well.

First let us focus on the DBLP dataset. Since the time unit is year in DBLP dataset, we do not know the specific ordering of nodes on which the event occurred in the same year. Thus, we treat the nodes obtaining the event in the same year as a whole and compute the DHT from each of those nodes to nodes obtaining the event in previous years. Then the significance scores (Eq. (13)) are computed and averaged to represent the correlation score for that year. The evolution curves for two keywords, "Microarray" and "Virus", are shown in Fig. 11. Fig. 11(a) shows the results of the proposed dynamic correlation measure while Fig. 11(b) shows those obtained by repeatedly computing $\tilde{\rho}$ on evolving V_q. We show (1 − p-value) for the dynamic measure for easy comparison. We can see that the evolution curves generated by $\tilde{\rho}$ decline first and then stay relatively stable at the end, while the curves generated by the dynamic measure have more complicated patterns. This is because $\tilde{\rho}$ computes the averaged proximity among all nodes before a time step and it is hard to capture the correlation at a specific time step. The dynamic measure has a more natural definition for estimating the structural correlation for a specific time step. Taking "Virus" as an example, one can see from Fig. 11(a) that the correlation first decreases and then increases. The decrease represents the topic dispersion phenomenon: at the beginning, several researchers proposed the topic; then more and more researchers started to pursue this topic, making the keyword less correlated with the social network. The increase means there are many collaborations between newly emerged nodes and previous nodes for the keyword. This may indicate the flourishing of this topic: professors engaged in the topic hire many new students to pursue that topic.

The evolution curves for two products in TaoBao are shown in Fig. 12. Since we have the specific time information for each product purchase, we show the correlation evolution with respect to the node sequence. Regarding smoothing, we adopt equal depth with window size 51 (we do not smooth the curves for DBLP since each year's score is already an average). As is shown in Fig. 12, the curves generated by the dynamic measure exhibit more complicated patterns than those generated by $\tilde{\rho}$. For example, the dynamic correlation of iPad fluctuates several times, which could be helpful for setting up a fine-grained promotion strategy in terms of time, while regarding its static correlation there are no apparent correlation peaks after the one at the very beginning. It is difficult to discover fine-grained patterns of correlation evolution by the static measure. In conclusion, the proposed dynamic measure is useful for investigating step-by-step correlation evolution which cannot be well described by the static measure.

7.6 Scalability of Sampling-alg

Finally, we investigate the scalability of Sampling-alg when the graph size n increases. The Twitter graph is used to perform this experiment. We extract subgraphs with different sizes (i.e. n) and for each n, different values of m are tested. Results are averaged over 50 sampled DHTs. Fig. 13 shows that Sampling-alg is scalable and only needs 0.17s to estimate one DHT on a graph with 10 million nodes. Although the time cost of Sampling-alg is linear in n, it only involves creating an index array of size n in memory. Regarding ρ, the estimation time is only 8.5s on a graph with 10 million nodes if we set the number of samples $c' = 50$. Note that this can also be regarded as the time used for computing one adjusted correlation $\tilde{\rho}$ since $E(\rho)$ and $Var(\rho)$ can be obtained from pre-computed results. Intuitively, when n is fixed and m increases, the running time should decrease since it is easier to hit a target node (most random walks do not need to reach the maximum steps,

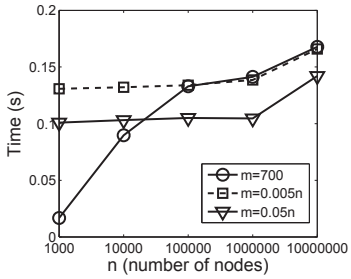


Fig. 13. Running times of Sampling-alg for estimating one DHT when varying the graph size.

s). This is the reason that the curve of $m = 0.05n$ is below that of $m = 0.005n$. Since we only store the adjacency list, the memory cost is linear in the number of edges in the graph. We do not show the curve here due to space limitation.

8 Related Work

Our work is related to attributed graph mining. Recently, researchers have incorporated node attributes in graph mining. For example, Ester *et al.* found attribute data can be used to improve clustering performance [10]. Zhou *et al.* constructed clustering algorithms based on structural and attribute similarities [28]. Silva *et al.* [26] proposed a novel structural correlation pattern mining problem which aims to find pairs (S, V) in which S is a frequent attribute set and V is a dense subgraph where each node contains all the attributes in S. The key difference between this problem and ours is that we study the correlation between an attribute and the entire network structure globally. In [21], the authors examined how to find dense subgraphs where nodes had homogeneous attributes. Again, they also focused on local patterns but not global structural correlations studied in this work. In bioinformatics, researchers have studied how to remove gene correlation of gene ontology graphs in order to better assess gene ontology term significance [2]. However, they consider a notion of correlation defined by reachability, while our structural correlation is based on proximity. Moreover, their goal is correlation elimination, rather than detection.

Our work is also related to research in social influence. In social networks, behaviors of two people tend to be related if they are friends. Anagnostopoulos *et al.* [3] studied the problem of distinguishing social influence from other sources of correlation using time series of people behaviors. La Fond and Neville [18] presented a randomization technique for distinguishing social influence and homophily for temporal network data. Our work is different from theirs in two aspects. First, these studies assume the existence of correlation (direct neighborhoods), while we try to determine if there is a correlation. Second, they are concerned with direct friendships, while our structural correlation is defined in a more general graph proximity notion. For example, in [3] the parameter α in the logistic function is used to indicate the degree of correlation. If a correlated event occurs in local areas of the graph but has not got a chance to occur on two adjacent nodes, their model cannot even learn α since α is multiplied by 0. gScore can cope with this situation since it considers graph proximity, rather than direct neighbors. The structural correlation problem addressed in this paper can be treated as a step before the above correlation causality (e.g. influence vs. homophily) analysis problem, since the latter problem usually assumes the existence of correlations and the goal is to find out whether it is due to influence or not.

There are many graph proximity measures proposed in the literature. Here we name a few. *Common neighbors* and *Jaccard's coefficient* are two measures based on node neighborhood [22]. Common neighbors computes the number of common neighbors of two nodes. Jaccard's coefficient is defined as the number of common neighbors divided by the number of all distinct neighbors of two nodes. *Katz* [14] defined a measure which sums over all paths between two nodes, exponentially damped by their length to make short paths more important. *Hitting time* [19] and *personalized PageRank* [23] are random walk based graph proximity measures. Our structural correlation framework can adopt any graph proximity measure. We use hitting time in this work since it is a more holistic measure.

The randomization and sampling techniques used in this work have been studied extensively. Gionis *et al.* used swap randomization to assess data mining results [11], while [17] provided a rigorous bound for identifying statistically significant frequent itemsets. We successfully extended related techniques to the graph domain and showed that these techniques with appropriate modifications are scalable in large-scale graphs. Random walk simulation has been used to estimate PageRank [4]. Their truncation length is a random variable depending on the teleportation parameter, while in our case we use fixed truncation length since our target is hitting time.

9 Conclusions

In this paper, we studied the problem of measuring how strongly an event that took place in a graph is correlated to the graph structure. A novel measure, called structural correlation, was introduced to assess this correlation. It can also be used to derive statistical significance to test if an event is randomly distributed over a graph or not. We proposed using hitting time to instantiate our framework and derived a set of sampling and approximation algorithms so that the correlation score can be estimated very quickly in large-scale graphs. By comparing the score with the situation where the event is randomly distributed in the same network, our method is able to discover the events of nodes that are highly correlated with the graph structure. Our method is scalable and successfully applied to the co-author DBLP network and a social network extracted from TaoBao.com, the largest

online shopping network in China, with many exciting discoveries. We also proposed a dynamic measure which revealed structural correlations at specific time steps and can be used to discover detailed evolutionary patterns.

REFERENCES

[1] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *SIGMOD*, pages 207–216, 1993.

[2] A. Alexa, J. Rahnenführer, and T. Lengauer. Improved scoring of functional groups from gene expression data by decorrelating go graph structure. *Bioinformatics*, 22(13):1600–1607, 2006.

[3] A. Anagnostopoulos, R. Kumar, and M. Mahdian. Influence and correlation in social networks. In *SIGKDD*, pages 7–15, 2008.

[4] B. Bahmani, A. Chowdhury, and A. Goel. Fast incremental and personalized pagerank. *PVLDB*, 4(3):173–184, 2010.

[5] H. Bao and E. Y. Chang. Adheat: an influence-based diffusion model for propagating hints to match ads. In *WWW*, pages 71–80, 2010.

[6] M. Brand. A random walks perspective on maximizing satisfaction and profit. In *SDM*, 2005.

[7] J. J. Brown and P. H. Reingen. Social ties and word-of-mouth referral behavior. *J. of Consumer Research*, 14(3):350–362, 1987.

[8] P. Chebotarev and E. V. Shamis. On proximity measures for graph vertices. *Automation and remote control*, 59(10):1443–1459, 1998.

[9] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *SIGKDD*, 2010.

[10] M. Ester, R. Ge, B. J. Gao, Z. Hu, and B. Ben-Moshe. Joint cluster analysis of attribute data and relationship data: the connected k-center problem. In *SDM*, pages 25–46, 2006.

[11] A. Gionis, H. Mannila, T. Mielikäinen, and P. Tsaparas. Assessing data mining results via swap randomization. In *SIGKDD*, pages 167–176, 2006.

[12] Z. Guan, J. Wu, Q. Zhang, A. Singh, and X. Yan. Assessing and ranking structural correlations in graphs. In *SIGMOD*, pages 937–948, 2011.

[13] W. Hoeffding. Probability inequalities for sums of bounded random variables. *J. of the American Statistical Association*, 58(301):13–30, 1963.

[14] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

[15] H. Kautz, B. Selman, and M. Shah. Referral web: combining social networks and collaborative filtering. *Communications of the ACM*, 40(3):63–65, 1997.

[16] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *SIGKDD*, pages 137–146. ACM, 2003.

[17] A. Kirsch, M. Mitzenmacher, A. Pietracaprina, G. Pucci, E. Upfal, and F. Vandin. An efficient rigorous approach for identifying statistically significant frequent itemsets. In *PODS*, 2009.

[18] T. La Fond and J. Neville. Randomization tests for distinguishing social influence and homophily effects. In *WWW*, pages 601–610, 2010.

[19] L. Lovász. Random walks on graphs: A survey. *Bolyai Soc. Math. Studies*, 2(2):1–46, 1993.

[20] Q. Mei, D. Zhou, and K. Church. Query suggestion using hitting time. In *CIKM*, 2008.

[21] F. Moser, R. Colak, A. Rafiey, and M. Ester. Mining cohesive patterns from graphs with feature vectors. In *SDM*, 2009.

[22] D. L. Nowell and J. Kleinberg. The link prediction problem for social networks. In *CIKM*, pages 556–559, 2003.

[23] L. Page, S. Brin, R. Motwani, and T. Winograd. The Pagerank citation algorithm: bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[24] P. Sarkar and A. Moore. A tractable approach to finding closest truncated-commute-time neighbors in large graphs. In *UAI*, 2007.

[25] P. Sarkar, A. W. Moore, and A. Prakash. Fast incremental proximity search in large graphs. In *ICML*, pages 896–903, 2008.

[26] A. Silva, W. Meira Jr, and M. J. Zaki. Structural correlation pattern mining for large graphs. In *Proc. of the 8th Workshop on Mining and Learning with Graphs*, pages 119–126, 2010.

[27] R. Srikant and R. Agrawal. Mining generalized association rules. *Future Generation Computer Systems*, 13(2-3):161–180, 1995.

[28] Y. Zhou, H. Cheng, and J. Yu. Graph clustering based on structural/attribute similarities. *PVLDB*, 2(1):718–729, 2009.

Jian Wu received his PhD degree in computer science from Zhejiang University, China in 2004, where he serves as an Associate Professor of Computer Science. His research interests include cloud computing and data mining. He received the Second Grade Prize from the National Science Progress Award in China.

Ziyu Guan received the BS and PhD degrees in Computer Science from Zhejiang University, China, in 2004 and 2010, respectively. He is currently an assistant project scientist in the Information Network Academic Research Center (INARC) in the computer science department of University of California at Santa Barbara. His research interests include attributed graph mining and search, machine learning, expertise modeling and retrieval, and recommender systems.

Qing Zhang is with Taobao.com which is China's online auction leader at the time. He is the founder of Alibaba's data warehouse and successfully built Taobao data platform system. He was in charge of the major upgrading of Taobao data platform from Oracle, Oracle RAC 4 nodes, Oracle RAC 20 nodes to the current Aerial Ladders which consists of 2000 computers running a hadoop system.

Ambuj K. Singh received the BTech degree from the Indian Institute of Technology and the PhD degree from the University of Texas at Austin. He is currently a professor of computer science at the University of California at Santa Barbara. He has written more than 130 technical papers in the areas of distributed computing, databases, and bioinformatics. He has graduated more than 25 students in his career. His current research interests include querying and mining, especially of biological data.

Xifeng Yan is an assistant professor at the University of California at Santa Barbara. He holds the Venkatesh Narayanamurti Chair in Computer Science. He received his Ph.D. degree in Computer Science from the University of Illinois at Urbana-Champaign in 2006. He was a research staff member at the IBM T. J. Watson Research Center between 2006 and 2008. He has been working on modeling, managing, and mining large-scale graphs in bioinformatics, social networks, information networks, and computer systems. He received NSF CAREER Award.