Evaluating Geo-Social Influence in Location-Based Social Networks

Chao Zhang Lidan Shou Ke Chen Gang Chen Yijun Bei
College of Computer Science
Zhejiang University, China
{chaozhang, should, chenk, cg, byj}@zju.edu.cn

ABSTRACT

The emerging location-based social network (LBSN) services not only allow people to maintain cyber links with their friends, but also enable them to share the events happening on them at different locations. The geo-social correlations among event participants make it possible to quantify mutual user influence for various events. Such a quantification of influence could benefit a wide spectrum of real-life applications such as targeted advertising and viral marketing.

In this paper, we perform an in-depth analysis of the geo-social correlations among LBSN users at event level, based on which we address two problems: *user influence evaluation* and *influential events discovery*. To capture the geo-social closeness between LBSN users, we propose a unified influence metric. This metric combines a novel social proximity measure named *penalized hitting time*, with a geographical weight function modeled by *power law distribution*. We propose two approximate algorithms, namely *global iteration* (GI) and *dynamic neighborhood expansion* (DNE), to efficiently evaluate user influence with tight theoretical error bounds. We then adopt the sampling technique and the threshold algorithm to support efficient retrieval of top-K influential events. Extensive experiments on both real-life and synthetic LBSN data sets confirm that the proposed algorithms are effective, efficient, and scalable.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Data Mining

General Terms

Algorithms, Experimentation

Keywords

Social Network, Information Extraction, Structural Analysis

1. INTRODUCTION

With the popularity of Web 2.0 technology and the proliferation of GPS-equipped mobile terminals, recent years have witnessed enormous growth in location-based social network (LBSN) services. These services, such as Foursquare, Facebook Places, and Google Latitude, not only allow one to maintain cyber links with other users, but also enable one to share one's events/activities happening at certain locations in various forms. To give a few examples, a Foursquare user may check-in at a newly opened golf court, and an Apple fan may post a geo-tagged tweet when she is shopping at an Apple store. Such events are being created and shared everywhere and every second in LBSNs. Foursquare, as a representative LBSN, has hit a whopping 2 billion check-ins since its foundation in 2009, and millions of new check-ins are still being added every day [1].

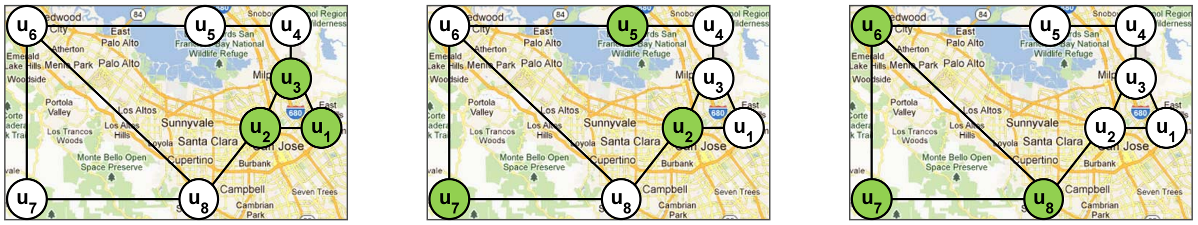
Among the numerous LBSN events, some exhibit strong geographical and social correlations among their participants, while others do not. Figure 1 shows the respective participants of three events in an illustrative LBSN, where each circle represents a user and its color indicates if he/she has participated in that event. We can observe that (1) the *iPhone* users (the green ones in Figure 1(a)) are socially connected and geographically close to each other; (2) the *KFC* users are faraway from each other both socially and geographically; (3) the *golf* users are socially close but geographically faraway.

Such different geo-social correlations motivate us to study the *mutual influence* between social relation and geographical distance for LBSN events. In Figure 1, if an iPhone user, say u_2, happens to find a new product at an Apple store in her vicinity, her posted information is likely to spread out across the community and drive nearby Apple fans (u_1 and u_3) to the same store. On the contrary, if a KFC or golf user creates a related post, the others may be less influenced, either because the information cannot reach them or the location of the attraction is too far.

The above description leads to the following observations. On one hand, the social links facilitate the propagation of user-generated information in the network. Therefore, events happening in one's close ties are likely to cause participation [3, 22]. On the other hand, the geographical distance plays a role (either strong or weak) in determining the users' physical activities. Intuitively, people tend to visit nearby places rather than distant ones in practice [12, 8, 24]. However, the impact of distance on the tendency of participation is undefined. To date, no previous study has addressed the aforementioned *mutual influence*.

In this paper, we perform an in-depth analysis of the geographical and social correlations among LBSN users for different events, and attempt to seek the answers to the following two interesting questions:

1. **User Influence Evaluation:** Given a set of LBSN users attributed with a common event, how to quantify the geo-social influence of one user to the others?



(a) iPhone (b) KFC (c) golf

Figure 1: User states of three LBSN events (The green circles denote the participants of each event).

2. **Influential Events Discovery:** Given a set of events, how to find influential events, i.e., the events for which one user exerts strong influence to the others?

Not difficult to imagine, the evaluation of geo-social influence can benefit a wide spectrum of real-life applications. Let us look at one example, an Apple store in New York wants to promote the sales of a newly released product (e.g., iPhone 5). By analyzing the mutual geo-social influences among iPhone users, it can simply target the local influential users and send coupons to them. If these users post relevant information on-line, many other iPhone fans are likely to be attracted to the store for that new product. Compared with existing social influence analysis methods [22, 11, 4], the most distinguishing characteristic in the geo-social influence evaluation process is the need to consider geographical distances among users. If a user u in New York has many social friends, but most of them live in Los Angeles, then u should not be regarded as *influential*: few of u's friends will be affected by u's post, because the store near u is too far for them. As another example, marketers can promote products relevant to influential events, because the users of such events are typically close to each other in both geographical space and social space, thus the sales are expected to be elevated virally due to the intrinsic geo-social correlations.

Although a number of studies have analyzed the mutual user influence in social networks [22, 2, 3, 11, 4], none of the proposed techniques are capable of quantifying geo-social influence for events. The reasons are threefold: First, these studies typically perform social influence analysis in a global setting without considering the events associated with each user. Second, while the information generated by one LBSN user may reach another via different social paths, the influence analysis in the previous studies is usually confined to direct social links. Third, these studies fail to take into account the geographical distance between two users. In practice, however, people are more likely to visit nearby places rather than distant ones, indicating a clear need to explore the impact posed by geographical distance.

Our Contributions. In this paper, we provide a unified user influence metric which tightly combines social proximity and geographical mobility features of LBSN users. On the social side, we propose a modified version of the hitting time measure, named *penalized hitting time* (PHT), to quantify the social proximity between LBSN users. Hitting time is a random-walk-based graph proximity measure which has been shown to be effective for link prediction [17], query suggestion [18], graph clustering [7], and so on. However, it is sensitive to long paths and tends to benefit popular entities [17, 19, 20]. Our PHT measure intrinsically avoids this drawback, owing to a nice property that the path weight is exponentially penalized by path length and thus short paths are given more priority. On the geographical side, we explore the mobility patterns of the users based on a real-life LBSN data set. The statistical results show that physical distance does play an essential role in determining the mobility behaviors of a user, and the geographical influence with regard to distance can be well modeled by *power law distribution*.

The computation of PHT is a challenging task, due to the fact that PHT takes account of all social paths between two LBSN users. As we will see, directly computing the PHT between two users takes $O(n^3)$ time (n is the total number of LBSN users), which is intolerable for large LBSNs. In view of this problem, we propose two approximate algorithms, namely the *global iteration* (GI) and *dynamic neighborhood expansion* (DNE) algorithms. Both algorithms work efficiently when computing PHT, and meanwhile ensure tight theoretical error bounds. In particular, the DNE algorithm can compute PHT in constant time regardless of LBSN size.

Relying on the user influence metric, we measure the influence of an event by aggregating the influences of event users, and investigate two specific aggregate functions, namely MAX and AVERAGE. Not surprisingly, the discovery of influential events is not trivial either, especially when the number of event users and the number of LBSN events are both large. We employ the *sampling technique* to avoid computing geo-social influence for each user when estimating event score, and adopt the *threshold algorithm* to efficiently retrieve top-K influential events.

We have conducted extensive experiments on both real-life and synthetic LBSN data sets. The experimental results confirm the effectiveness and efficiency of our proposed algorithms. Specifically, for the user influence evaluation problem, GI and DNE outperform existing solutions in running time by a factor up to an order of magnitude; for the influential events discovery problem, the sampling and threshold techniques work effectively for discovering top-K influential events, with many interesting findings.

Organization. The rest of this paper is organized as follows. We review related work in Section 2 and formally define our problems in Section 3. In Section 4 and 5, we present algorithms for evaluating user influence and discovering influential events, respectively. We study the empirical performance of the proposed algorithms in Section 6, and finally conclude the paper in Section 7.

2. RELATED WORK

Generally, existing approaches relevant to our work fall into three categories: social influence analysis, LBSN mining, and graph proximity computation.

Social Influence Analysis. So far, many studies have investigated the influence of a single node in social networks. In a number of papers, it is believed that influence is relevant to the structural roles of the target nodes and their topological interactions with the underlying social network, and thus social influences can be derived with link analysis, e.g., PageRank [6] and HITS [16]. Many other studies attempted to estimate social influence by exploring historical user data such as blogs [2] and tweets [4]. All these studies analyzed social influence in a global setting without considering node attributes. In contrast, we compute social influence at event level and provide a finely grained capture of influence strength. Tang *et al.* [23] also found social influences could vary greatly across different topics and performed topical influence computation using probabilistic model. Our social influence analysis significantly differs from their work in that we associate each node with a set of discrete events based on ground-truth LBSN data, rather than a probabilistic latent topic distribution.

The interaction between social influence and structural correlation has also been extensively studied in literature. Specifically, Anagnostopoulos *et al.* [3] proposed to distinguish influence from other correlation factors based on temporal analysis of Flicker user behaviors. La Fond *et al.* [11] performed randomization tests for distinguishing influence and homophily in temporal social networks. Guan *et al.* [13] studied the problem of assessing structural correlations on event-level granularity. The focuses of these studies are to determine the existence of social correlation or identify influence hidden behind correlation, whereas we attempt to quantify the strength of influence.

LBSN Mining. Recently, with the rise of on-line LBSN services, researchers have paid much attention to mining LBSN. To name a few, Cho *et al.* [8] modeled user location as a dynamic Gaussian mixture and employed a generative approach to postulate the mobility pattern of an individual, Scellato *et al.* [21] exploited geographical features to address the link prediction problem, and Ye *et al.* [24] incorporated distance information into traditional collaborative filtering framework for friend recommendation. These studies demonstrated that geographical distance could serve as a useful source for various mining tasks. Unfortunately, none of them attempted to fuse physical proximity with cyber connection to evaluate influences among LBSN users.

Graph Proximity Computation. *Hitting time* and its variants [19, 20] have been proposed and successfully used for link prediction [17], product recommendation [5], query suggestion [18], graph clustering [7], etc. These studies exploited the fact that hitting time captures the holistic feature of the underlying network and is quite robust to noise. However, it is also demonstrated that hitting time is sensitive to long paths and tends to benefit popular entities [17, 19, 20]. Our PHT measure, instead, has a nice property that the path weight is exponentially dampened by the path length, and thus intrinsically avoids this drawback. It is worth mentioning that although the *decayed hitting time* measure proposed by Guan *et al.* [13] also has this property, it can be viewed as a special case of PHT wherein the dampening parameter is fixed at e^{-1}.

A number of other graph proximity measures have also been proposed. In specific, *Jaccard's coefficient* is defined as the number of the common neighbors of two given vertices divided by the number of their distinct neighbors. The limitation of this measure is that it only considers the direct neighborhood of a vertex. In contrast, *Katz* [15] avoids this problem by summing over the collection of paths between two vertices, and the weight of a path decays exponentially with its length so that small paths are given more weight. *Personalized PageRank* [14] is another random-walk-based graph proximity, which biases the probability distribution of PageRank towards a set of pre-specified graph vertices. Although we adopt the PHT measure in this work, the proposed framework can be extended to compute social influence under the above proximity measures.

3. PRELIMINARIES

In this section, we provide some preliminaries for evaluating geo-social influence in LBSN. In what follows, we introduce the concept of *penalized hitting time* in Section 3.1, conduct geographical influence analysis in Section 3.2, and formulate our problems in Section 3.3. Table 1 lists the notations used in the rest of this paper.

Table 1: Notations used in the paper.

	Description
G	A location-based social network.
V_G	The vertex set of G.
E_G	The edge set of G.
S_e	The set of users on which event e has happened.
l_u	The location of an LBSN user $u \in V_G$.

3.1 Penalized Hitting Time

Given an event e, let u and v be two LBSN users in S_e, we analyze the social influence of u to v from a random walk perspective. A random walk in G is defined as follows: a user starts from vertex $x_0 \in V_G$ and randomly moves to its neighbors, if at step t the user is at vertex $x_t = i$, then in the next step she moves to i's neighbor j with probability proportional to the weight w_{ij}, namely:

$$p_{ij} = Pr(x_{t+1} = j | x_t = i) = \frac{w_{ij}}{d_i}$$

where $d_i = \sum_{j=1}^{n} w_{ij}$ is the degree of vertex i. Clearly, the sequence $\{x_t\}$ forms a Markov chain. We use $\mathbf{P} = [p_{ij}]_{n\times n}$ to denote the transition probability matrix of this Markov chain, so that $p_{ij} = \frac{w_{ij}}{d_i}$ if $(i, j) \in E_G$ and zero otherwise.

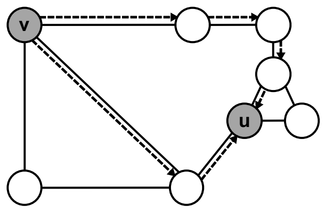


Figure 2: Random walk in the social network.

Now consider a user performing a random walk from v, the user may hit u along various paths, as shown in Figure 2. Let L be the path along which v hits u for the first time, satisfying $|L| = h$. We assign a weight τ^h to L where $\tau (0 < \tau < 1)$ is an attenuation parameter. In other words, the weight of L is exponentially dampened by its length, and thus short paths are made more important. The rationale behind is that people are more likely to be affected by their friends, rather than people socially faraway. Hence, two graph vertices are considered *socially close* if there are many short paths connecting them. Based on this observation, we define the *penalized hitting time* (PHT) as the expected path weight of the random walk that starts from v and hits u for the first time.

Definition 1. *Let u and v be two LBSN users in V_G, given an attenuation parameter $\tau \in (0, 1)$, the penalized hitting time from v to u is*

$$s_u(v) = \sum_{h=1}^{\infty} \tau^h Pr(H_u = h | x_0 = v)$$

where $Pr(H_u = h|x_0 = v)$ is the probability that the random walk starting from v first hits u after h steps.

3.2 Geographical Mobility Pattern Analysis

To analyze the geographical influence posed by physical distance, we have crawled a real LBSN data set from Gowalla[1] during a three-month period. We find that the users' mobility patterns are significantly influenced by geographical distance. Specifically, a user is more likely to visit a POI if that POI is not faraway from him. To better understand this, we calculate the distances between all pairs of two consecutive check-ins and plot a histogram, as shown in Figure 3. Not difficult to observe, the log-scale check-in probability is approximately linear to the log-scale geographical distance. The geographical influence can therefore be suitably modeled as a power law distribution w.r.t. geographical distance.

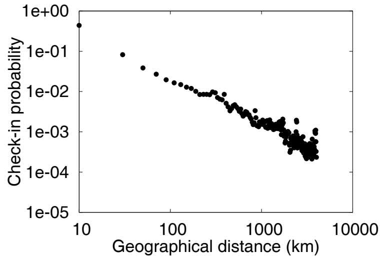


Figure 3: Check-in probability *vs.* geographical distance.

DEFINITION 2. *Given two users u and v, let $||l_u - l_v||$ be the geographical distance between u and v, we define the geographical influence of u to v as:*

$$g_u(v) = \begin{cases} \alpha \cdot ||l_u - l_v||^\beta & \text{for } ||l_u - l_v|| > \alpha^{-1/\beta} \\ 1 & \text{for } ||l_u - l_v|| \le \alpha^{-1/\beta} \end{cases} \quad (1)$$

where α and β are the parameters of a power law distribution.

Remark. The geographical influence $g_u(v)$ could be interpreted in the following way: u and v are two users attributed with a common event (e.g., playing golf), if u creates a post related to a golf court near him and v knows this via the LBSN, given the distance of u and v, how likely would v also pay a visit to the golf court?

3.3 Problem Definition

The PHT measure captures the social proximity between LBSN users and the power law distribution models their physical interactions. We combine them to derive a unified geo-social influence metric.

DEFINITION 3. *Let u and v be two LBSN users attributed with a common event, the geo-social influence of u to v is given by:*

$$f_u(v) = s_u(v) \cdot g_u(v)$$

Relying on the unified influence metric, we are now ready to formulate the *user influence evaluation* and *influential events discovery* problems.

PROBLEM 1. ***(User Influence Evaluation)*** *Given an event e and its user set S_e, evaluate the geo-social influence of user $u \in S_e$ to all the other users in S_e:*

$$f_u(S_e) = \sum_{v \in S_e - \{u\}} f_u(v)$$

[1]http://en.wikipedia.org/wiki/Gowalla

PROBLEM 2. ***(Influential Events Discovery)*** *Given an event collection C, retrieve K events from C with the largest event scores. The score of an event e is defined as the aggregate user influence of the users in S_e:*

$$score(e) = \text{AGG}_{u \in S_e} f_u(S_e)$$

where AGG *can be the aggregate function* MAX *or* AVERAGE.

4. USER INFLUENCE EVALUATION

In this section, we discuss how to derive the influence of one user. Below, we first provide the GI and DNE algorithms for computing PHT in Section 4.1, and then investigate the computation of geographical influence in Section 4.2.

4.1 Computing Penalized Hitting Time

4.1.1 The GI Algorithm

Given two users u and v, the PHT $s_u(v)$ is the expected path weight of the random walk that starts from v and first hits u. Naturally, we could compute $s_u(v)$ in a one step look-ahead fashion: consuming one step to move to v's neighbors and then summing up their PHTs. Accordingly, we have:

$$\begin{aligned} s_u(v) &= \sum_{h=1}^{\infty} \tau^h Pr(H_u = h | x_0 = v) \\ &= \sum_{h=1}^{\infty} \tau^h \sum_{w \in V_G} p_{vw} Pr(H_u = h-1 | x_0 = w) \\ &= \tau \sum_{w \in V_G} p_{vw} s_u(w) \end{aligned}$$

Hence, the following recurrence relation holds for any user $v \in V_G$.

$$s_u(v) = \begin{cases} \tau \sum_{w \in V_G} p_{vw} s_u(w) & \text{for } v \neq u \\ 1 & \text{for } v = u \end{cases} \quad (2)$$

Let $\mathbf{P}_u$ be a modification of the original transition matrix $\mathbf{P}$ where the entries in the row corresponding to u are set to all zeros; $\mathbf{s}_u$ be a $n \times 1$ vector ($n = |V_G|$) where $\mathbf{s}_u(v)$ is the PHT from v to u; and $\mathbf{c}_u$ be a $n \times 1$ vector with the element corresponding to u set to 1, and all other elements to 0. Then the recurrence relation leads to a linear system consisting of n equations:

$$\mathbf{s}_u = \tau \mathbf{P_u} \mathbf{s}_u + \mathbf{c}_u \quad (3)$$

Now, the problem of computing the PHT of every user $v \in V_G$ to u is reduced to solving Equation 3 to obtain the vector $\mathbf{s}_u$. Unfortunately, the time complexity of solving the linear system is $O(n^3)$, which is prohibitive for large LBSNs containing millions of users.

```
Algorithm 1: GlobalIteration($\mathbf{P}_u, \mathbf{c}_u, \tau, k$)
1 Initialize $\mathbf{s}_u^{(0)} = \mathbf{c}_u$
2 for $i = 1$ to $k$ do
3     $\mathbf{s}_u^{(i)} = \tau \mathbf{P_u} \mathbf{s}_u^{(i-1)} + \mathbf{c}_u$
4 return $\mathbf{s}_u^{(k)}$
```

To overcome this problem, we propose the *global iteration* (GI) algorithm to obtain approximate PHTs. The idea is to assign an arbitrary PHT value for each user in V_G, and then iterate over the linear system for a few times. As shown in Algorithm 1, we choose

$\mathbf{c}_u$ as an initial PHT vector. Then, we substitute $\mathbf{s}_u^{(i)}$ into Equation 3 and get the next-round PHT vector. After iterating for k rounds, we output $\mathbf{s}_u^{(k)}$ as the resulting PHT vector. In the following, we prove the GI algorithm will converge to the exact value of $\mathbf{s}_u$ if k is large enough, and give an analytical error bound of $\mathbf{s}_u^{(k)}$.

THEOREM 1. *The iterative process will converge to the exact value of* $\mathbf{s}_u$ *if* $k \to \infty$.

PROOF. Let $\mathbf{T} = \tau \mathbf{P_u}$, the l_∞ natural norm of $\mathbf{T}$ is

$$||\mathbf{T}||_\infty = \max_{1 \le i \le n} \sum_j |\mathbf{T}_{ij}| = \tau.$$

Then the spectral radius of $\mathbf{T}$ satisfies $\rho(\mathbf{T}) \le ||\mathbf{T}||_\infty < 1$. With Equation 3, we have the k-th round value of $\mathbf{s}_u^{(k)}$:

$$\begin{aligned}
\mathbf{s}_u^{(k)} &= \mathbf{T}\mathbf{s}^{(k-1)} + \mathbf{c}_u \\
&= \mathbf{T}(\mathbf{T}\mathbf{s}^{(k-2)} + \mathbf{c}_u) + \mathbf{c}_u \\
&\dots \\
&= \mathbf{T}^k \mathbf{s}_u^{(0)} + (\mathbf{T}^{k-1} + \dots + \mathbf{T} + \mathbf{I})\mathbf{c}_u
\end{aligned}$$

Since $\rho(\mathbf{T}) < 1$, the matrix $\mathbf{T}$ is convergent and $\lim_{k \to \infty} \mathbf{T}^k \mathbf{s}_u^{(0)} = \mathbf{0}$, which ensures that:

$$\lim_{k \to \infty} \mathbf{s}_u^{(k)} = (\sum_{i=0}^{\infty} \mathbf{T}^i)\mathbf{c}_u = (\mathbf{I} - \mathbf{T})^{-1}\mathbf{c}_u$$

Hence, the sequence $\{\mathbf{s}_u^{(k)}\}$ converges to $(\mathbf{I} - \mathbf{T})^{-1}\mathbf{c}_u$, which is exactly the solution of Equation 3. $\square$

THEOREM 2. *Let* $\mathbf{s}_u^{(i)}$ *be the i-th round estimation of* $\mathbf{s}_u$. $\forall v \in V_G, |\mathbf{s}_u(v) - \mathbf{s}_u^{(k)}(v)| \le \frac{\tau^k}{1-\tau} \max_{w \in V_G} |\mathbf{s}_u^{(1)}(w) - \mathbf{s}_u^{(0)}(w)|$.

PROOF. $\forall k \ge 1$, we have $\mathbf{s}_u^{(k+1)} - \mathbf{s}_u^{(k)} = \mathbf{T}(\mathbf{s}_u^{(k)} - \mathbf{s}_u^{(k-1)})$. Let $||\cdot||$ be the l_∞ natural norm, then:

$$||\mathbf{s}_u^{(k+1)} - \mathbf{s}_u^{(k)}|| \le ||\mathbf{T}|| \cdot ||\mathbf{s}_u^{(k)} - \mathbf{s}_u^{(k-1)}|| \dots \le ||\mathbf{T}||^k \cdot ||\mathbf{s}_u^{(1)} - \mathbf{s}_u^{(0)}||.$$

Thus, $\forall m > k$, we have:

$$\begin{aligned}
||\mathbf{s}_u^{(m)} - \mathbf{s}_u^{(k)}|| &= ||\sum_{i=k}^{m-1} (\mathbf{s}_u^{(i+1)} - \mathbf{s}_u^{(i)})|| \\
&\le \sum_{i=k}^{m-1} ||(\mathbf{s}_u^{(i+1)} - \mathbf{s}_u^{(i)})|| \\
&\le ||\mathbf{T}||^k \cdot ||\mathbf{s}_u^{(1)} - \mathbf{s}_u^{(0)}|| \cdot (\sum_{i=0}^{m-k-1} ||\mathbf{T}||^i) \\
&= ||\mathbf{T}||^k \cdot ||\mathbf{s}_u^{(1)} - \mathbf{s}_u^{(0)}|| \cdot \frac{1 - ||\mathbf{T}||^{m-k}}{1 - ||\mathbf{T}||}
\end{aligned}$$

We know that $\lim_{m \to \infty} \mathbf{s}_u^{(m)} = \mathbf{s}_u$ and $||\mathbf{T}|| = \tau$, thus:

$$||\mathbf{s}_u - \mathbf{s}_u^{(k)}|| = \lim_{m \to \infty} ||\mathbf{s}_u^{(m)} - \mathbf{s}_u^{(k)}|| \le \frac{\tau^k}{1-\tau} \cdot ||\mathbf{s}_u^{(1)} - \mathbf{s}_u^{(0)}||$$

which completes the proof. $\square$

4.1.2 *The DNE Algorithm*

The GI algorithm reduces the complexity of PHT computation from $O(|V_G|^3)$ to $O(k|E_G|)$. However, the performance of GI is still not the best one can hope for. Indeed, for LBSNs involving millions of users, a single matrix-vector multiplication could be quite expensive, making it really necessary to design a more efficient and scalable solution. Below, we propose another, more efficient algorithm called *dynamic neighborhood expansion* (DNE), which can compute PHT in constant time with a tight theoretical error bound.

As stated above, the major problem of GI is that it needs to perform matrix-vector multiplications over the entire LBSN. We argue that the PHTs of the users socially faraway from u are actually negligible as the weight of a path is exponentially dampened by its length. Based on this observation, DNE performs PHT computation in two phases: (1) In the first phase named ***expansion-update***, DNE starts from u and incrementally expands u's neighborhood to incorporate users that are close to u, and obtains their PHTs using the recurrence relation (Equation 2). Such a process continues until the cardinality of u's neighborhood is large enough to ensure a small approximation error. (2) In the second phase named ***refinement***, DNE performs a few iterations (Equation 2) over the neighborhood vertices to refine their PHTs. The PHTs of vertices outside u's neighborhood are set to zeros and do not need to be computed at all.

Let N_u be the set of vertices that have been incorporated into u's neighborhood, and $B_u \subseteq N_u$ be the set of boundary vertices ("boundary" means the vertex has at least one neighbor not in N_u). A fundamental issue in the first phase is: *which vertex in* B_u *should be chosen to expand in each round?* We adopt the best-first strategy and choose the vertex with the largest PHT in B_u. The rationale behind is twofold: First, for a vertex with large PHT, its neighbors are also likely to have large PHTs (Equation 2). The best-first manner thus gives priority to vertices with high PHTs and inclines to ignore unimportant vertices. Second, soon we will see, the approximation error of DNE is determined by the vertex with the largest PHT in B_u. By eliminating it, DNE can produce a better approximation after each expansion.

Algorithm 2: DNE($\mathbf{P}_u, \mathbf{c}_u, \tau, m, k$)

```
1  Initialize $N_u = \{u\}, B_u = \{u\}, \widehat{\mathbf{s}}_u^{(0)} = \mathbf{c}_u$
2  while $|N_u| < m$ and $B_u \neq \emptyset$ do
       // Expansion
3      Remove $w$ from $B_u$ where $\widehat{\mathbf{s}}_u^{(0)}(w) = \max_{v \in B_u} \widehat{\mathbf{s}}_u^{(0)}(v)$
4      foreach in-neighbor $v$ of $w$ do
5          if $w \notin N_u$ then
6              Add $w$ into $N_u$
7      foreach in-neighbor $v$ of $w$ do
8          if $w$ has any in-neighbor not in $N_u$ then
9              Add $w$ into $B_u$
       // Update
10     foreach $v \in N_u$ do
11         $\widehat{\mathbf{s}}_u^{(0)}(v) = \tau \sum_{w \in V_G} p_{vw} \widehat{\mathbf{s}}_u^{(0)}(w) + \mathbf{c}_u(v)$
   // Refinement
12 for $i = 1$ to $k$ do
13     foreach $v \in N_u$ do
14         $\widehat{\mathbf{s}}_u^{(i)}(v) = \tau \sum_{w \in V_G} p_{vw} \widehat{\mathbf{s}}_u^{(i-1)}(w) + \mathbf{c}_u(v)$
15 return $\widehat{\mathbf{s}}_u^{(k)}$
```

Algorithm 2 presents the details of the DNE algorithm. As shown, we initialize N_u with $\{u\}$ and incrementally expand N_u until its cardinality reaches a pre-defined number m. Each expansion is directed by the best-first strategy such that the vertex with largest PHT in B_u is selected and its neighbors are added into N_u. In the refinement phase, we iterate over the vertices in N_u for k times and finally output $\widehat{\mathbf{s}}_u^{(k)}$. Let $\widehat{\mathbf{s}}_u$ be the ideal PHT vector by ignoring all vertices not in N_u, it is worth mentioning that the output of DNE is actually the k-th round estimation[2] of $\widehat{\mathbf{s}}_u$.

Figure 4 gives an example of the DNE algorithm with $\tau = 0.6$. We use dark circles to denote the vertices that have been incorporated into N_u. Among them, the ringed ones denote the vertices that are in B_u. To compute the PHTs of all vertices to u, the DNE algorithm executes the following steps: (1) Initialization: $s_u(u) = 1$. (2) Expand u and update: $s_u(u) = 1, s_u(v_1) = 0.2, s_u(v_2) = 0.6, s_u(v_3) = 0.3, s_u(v_4) = 0.15$. (3) Expand v_3 and update: $s_u(u) = 1, s_u(v_1) = 0.2, s_u(v_2) = 0.6, s_u(v_3) = 0.3, s_u(v_4) = 0.15, s_u(v_5) = 0.18$. (4) Expand v_1 and update: $s_u(u) = 1, s_u(v_1) = 0.2, s_u(v_2) = 0.6, s_u(v_3) = 0.354, s_u(v_4) = 0.15, s_u(v_5) = 0.18, s_u(v_6) = 0.06, s_u(v_7) = 0.06$. (5) Refine PHTs for 10 times: $s_u(u) = 1, s_u(v_1) = 0.227, s_u(v_2) = 0.6, s_u(v_3) = 0.366, s_u(v_4) = 0.15, s_u(v_5) = 0.219, s_u(v_6) = 0.068, s_u(v_7) = 0.068$.

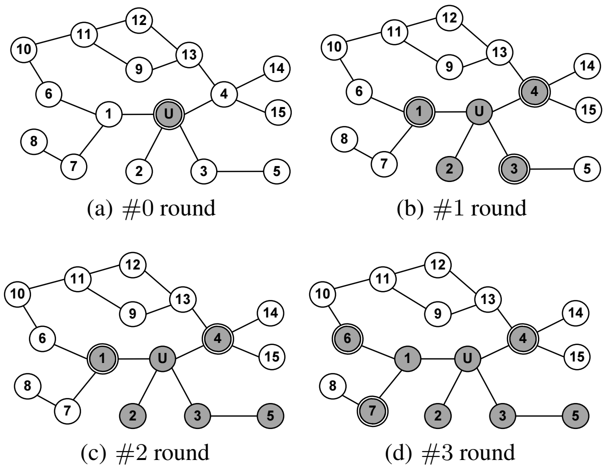

(a) #0 round (b) #1 round (c) #2 round (d) #3 round

Figure 4: An example of the DNE algorithm.

Suppose the average degree of each vertex in V_G is δ. Then the complexity of the *expansion-update* phase is $O(\delta^2 + 2\delta^2 + \cdots + \frac{m}{\delta}\delta^2) = O(m^2)$, and that of the *refinement* phase is $O(mk\delta)$. The total complexity of the DNE algorithm is thus $O(m^2 + mk\delta)$. While it saves vast amounts of computation cost, the question is: *how well does DNE approximate the PHTs of all vertices?* Below, we provide a theoretical error bound for DNE.

LEMMA 1. *$\forall v \in V_G$, it is ensured $\widehat{\mathbf{s}}_u(v) \le \mathbf{s}_u(v)$.*

PROOF. Let $\mathbf{P'}_u$ be a modification of $\mathbf{P}_u$ where the row entries corresponding to vertices not in N_u are set to all zeros, then we have $\widehat{\mathbf{s}}_u = \tau\mathbf{P'}_u\widehat{\mathbf{s}}_u + \mathbf{c}_u$. Let $\mathbf{T'} = \tau\mathbf{P'_u}$, then its spectral radius satisfies $\rho(\mathbf{T'}) \le ||\mathbf{T'}||_\infty < 1$. Following the same manner as in the proof of Theorem 1, we obtain:

$$\begin{cases} \widehat{\mathbf{s}}_u &= \lim\limits_{k\to\infty}(\mathbf{T'}^k + \cdots + \mathbf{T'} + \mathbf{I})\mathbf{c}_u \\ \mathbf{s}_u &= \lim\limits_{k\to\infty}(\mathbf{T}^k + \cdots + \mathbf{T} + \mathbf{I})\mathbf{c}_u \end{cases}$$

$\forall k \ge 0$, we have $\mathbf{T'}^k\mathbf{c}_u \le \mathbf{T}^k\mathbf{c}_u$. Hence, $\forall v \in V_G$, it is ensured that $\widehat{\mathbf{s}}_u(v) \le \mathbf{s}_u(v)$. ☐

[2]And $\widehat{\mathbf{s}}_u^{(k)} = \widehat{\mathbf{s}}_u$ when $k \to \infty$.

LEMMA 2. *Let $\mathcal{M} = \max\limits_{v\in B_u} \mathbf{s}_u(v)$, and O_u be the set of vertices not in N_u, then $\forall v \in O_u$, $\mathbf{s}_u(v) - \widehat{\mathbf{s}}_u(v) \le \tau\mathcal{M}$.*

PROOF. Since $\forall v \in O_u, \widehat{\mathbf{s}}_u(v) = 0$, it suffices to prove $\mathbf{s}_u(v) \le \tau\mathcal{M}$. We construct a matrix $\mathbf{P}_B$ for vertices in $O_u \cup B_u$, where the rows of vertices in O_u stay the same as in $\mathbf{P}$, and the rows of vertices in B_u are set to all zeros. In addition, we construct a column vector $\mathbf{c}_B$ where the entries of vertices in O_u are set to zeros, and the entries of vertices in B_u are set to their accurate PHTs.

As the PHTs of all vertices in O_u are gained from the vertices in B_u, the accurate PHTs of vertices in $O_u \cup B_u$ satisfy $\mathbf{s} = \tau\mathbf{P}_B\mathbf{s} + \mathbf{c}_B$. Again, the solution of this linear system can be obtained with the iterative technique, namely $\mathbf{s}^{(k)} = \tau\mathbf{P}_B\mathbf{s}^{(k-1)} + \mathbf{c}_B$ and $\mathbf{s} = \lim\limits_{k\to\infty}\mathbf{s}^{(k)}$.

Let $\mathbf{s}^{(0)} = \mathbf{c}_B$, then $\forall v \in O_u \cup B_u, \mathbf{s}^{(0)}(v) \le \mathcal{M}$. Moreover, if $\forall v \in O_u \cup B_u, \mathbf{s}^{(k)}(v) \le \mathcal{M}$, it is ensured $\forall v \in O_u, \mathbf{s}^{(k+1)}(v) \le \tau\mathcal{M}$. Hence, $\forall v \in O_u$, we have $\mathbf{s}_u(v) = \lim\limits_{k\to\infty}\mathbf{s}^{(k)}(v) \le \tau\mathcal{M}$, thus completing the proof. ☐

LEMMA 3. *Let $\mathcal{M} = \max\limits_{v\in B_u}\mathbf{s}_u(v)$, then $\forall v \in N_u$, $\mathbf{s}_u(v) - \widehat{\mathbf{s}}_u(v) \le \tau^2\mathcal{M}$.*

PROOF. Let $\Delta\mathbf{s}_u$ be a $n \times 1$ vector where $\Delta\mathbf{s}_u(v) = \mathbf{s}_u(v) - \widehat{\mathbf{s}}_u(v)$. We construct another $n \times 1$ vector $\Delta\mathbf{c}$, where the entries of vertices in O_u are set to their accurate PHTs and the entries of vertices in N_u are set to zeros. Since the PHT errors for all vertices in N_u are all caused by vertices in O_u, $\Delta\mathbf{s}_u$ satisfies $\Delta\mathbf{s}_u = \tau\mathbf{P'}_u\Delta\mathbf{s}_u + \Delta\mathbf{c}$.

The solution of this linear system is given by the iterative representation $\Delta\mathbf{s}_u = \lim\limits_{k\to\infty}\Delta\mathbf{s}_u^{(k)}$. We set $\Delta\mathbf{s}_u^{(0)} = \Delta\mathbf{c}$, then $\forall v \in V_G, \Delta\mathbf{s}_u^{(0)}(v) \le \tau\mathcal{M}$. In addition, if $\forall v \in V_G, \Delta\mathbf{s}_u^{(k)}(v) \le \tau\mathcal{M}$, then $\forall v \in N_u, \Delta\mathbf{s}_u^{(k+1)}(v) \le \tau^2\mathcal{M}$. We thus have $\forall v \in N_u, \Delta\mathbf{s}_u(v) = \lim\limits_{k\to\infty}\Delta\mathbf{s}_u^{(k)}(v) \le \tau^2\mathcal{M}$. ☐

THEOREM 3. *Let $\widehat{\mathcal{M}}^{(k)} = \max\limits_{v\in B_u}\widehat{\mathbf{s}}_u^{(k)}(v)$, and that $\lambda = \frac{\tau^k}{1-\tau}\cdot\max\limits_{v\in V_G}(\widehat{\mathbf{s}}_u^{(1)}(v) - \widehat{\mathbf{s}}_u^{(0)}(v))$. Given any vertex v, we have:*

$$\mathbf{s}_u(v) - \widehat{\mathbf{s}}_u^{(k)}(v) \le \begin{cases} \frac{\tau}{1-\tau^2}(\widehat{\mathcal{M}}^{(k)} + \lambda) & \textit{for } v \notin N_u \\ \frac{\tau^2}{1-\tau^2}(\widehat{\mathcal{M}}^{(k)} + \lambda) + \lambda & \textit{for } v \in N_u \end{cases}$$

PROOF. Let $w \in B_u$ be the vertex corresponding to $\mathcal{M} = \max\limits_{v\in B_u}\mathbf{s}_u(v)$; and $\widehat{\mathcal{M}} = \max\limits_{v\in B_u}\widehat{\mathbf{s}}_u(v)$. By Lemma 3, we know $\mathcal{M} - \widehat{s}_u(w) \le \tau^2\mathcal{M}$, which can be transformed to

$$\mathcal{M} \le \frac{1}{1-\tau^2}\widehat{s}_u(w) \le \frac{1}{1-\tau^2}\widehat{\mathcal{M}}.$$

Meanwhile, following the same manner as in the proof of Theorem 2, for any vertex in V_G, we have

$$\widehat{\mathbf{s}}_u(v) - \widehat{\mathbf{s}}_u^{(k)}(v) \le \lambda \tag{4}$$

(1)With Lemma 2, we obtain $\forall v \notin N_u$, $\mathbf{s}_u(v) - \widehat{\mathbf{s}}_u^{(k)}(v) = \mathbf{s}_u(v) - \widehat{\mathbf{s}}_u(v) \le \tau\mathcal{M} \le \frac{\tau}{1-\tau^2}\widehat{\mathcal{M}}$. According to Equation 4, it is ensured $\widehat{\mathcal{M}} - \widehat{\mathcal{M}}^{(k)} \le \lambda$. Thus, $\mathbf{s}_u(v) - \widehat{\mathbf{s}}_u^{(k)}(v) \le \frac{\tau}{1-\tau^2}(\widehat{\mathcal{M}}^{(k)} + \lambda)$. (2)With Lemma 3, we have $\forall v \in N_u$, $\mathbf{s}_u(v) - \widehat{\mathbf{s}}_u(v) \le \tau^2\mathcal{M} \le \frac{\tau^2}{1-\tau^2}(\widehat{\mathcal{M}}^{(k)} + \lambda)$. Substituting Equation 4 in, we have $\mathbf{s}_u(v) - \widehat{\mathbf{s}}_u^{(k)}(v) \le \frac{\tau^2}{1-\tau^2}(\widehat{\mathcal{M}}^{(k)} + \lambda) + \lambda$. ☐

4.2 Computing Geographic Influence

The computation of geographical influence is quite straightforward so long as the parameters α and β in Equation 1 are known. In this section, we use *ridge regression* to estimate α and β. To achieve this, we first need to transform Equation 1 into log-log scale:

$$\log g_u(v) = \log \alpha + \beta \log ||l_u - l_v||$$

Let $y = \log g_u(v)$, $x = \log ||l_u - l_v||$, $\omega_0 = \log \alpha$, and $\omega_1 = \beta$, then the above equation becomes $y = \omega_0 + \omega_1 x$. We employ the least square error as the loss function, namely:

$$E(\omega) = \frac{1}{2}\sum_{i=1}^{n}(y - t_i)^2 + \frac{\lambda}{2}||\omega||^2$$

Here, n is the number of histogram points, t_i is the ground truth probability of point i , and λ is the regularization term to avoid overfitting. The optimal values of ω_0 and ω_1 can be obtained as $opt\{\omega_0, \omega_1\} = \arg_{\omega_0,\omega_1} \min E(\omega)$. Accordingly, α and β can be derived as $\alpha = 10^{\omega_0}$ and $\beta = \omega_1$.

5. INFLUENTIAL EVENTS DISCOVERY

Up to now, we have restricted our discussion to the *user influence evaluation* problem. In this section, we address the *influential events discovery* problem. Recall that the score of an event e is defined as the aggregate (MAX or AVERAGE) user influence of the users in S_e. To retrieve top-K influential events from an event collection C, the straightforward way is as follows: First, we compute the influence of each user in S_e, and derive their aggregate influence as e's score. Then, by computing the score of each event $e_i \in C$, we can find out K events with the largest scores. However, such a straightforward solution may be inefficient when the cardinalities of S_e and C are both high. In view of this problem, we exploit the sampling technique and the threshold algorithm to speed up the discovery of top-K events.

When computing the score of one event, we adopt the standard Monte Carlo sampling scheme. To be more specific, given an event e, we randomly pick c users from S_e and compute the influences of these c users. Then, we obtain their aggregate influence as an estimate of $score(e)$. Below, we provide the lower bound of c to obtain an ϵ-correct answer for $score(e)$ when the aggregate function AVERAGE is used.

THEOREM 4. *Suppose we randomly select c users from S_e to estimate the average influence score of e, to ensure $Pr(|score(e) - \widehat{score(e)}| \leq \epsilon) \geq 1 - \delta$, the number of samples c must satisfy $c \geq \frac{1}{2\epsilon^2} \ln \frac{2}{\delta}$.*

PROOF. According to Hoeffding's inequality, we have

$$Pr(|score(e) - \widehat{score(e)}| \leq \epsilon) \geq 1 - 2e^{-2c\epsilon^2}.$$

Setting $1 - 2e^{-2c\epsilon^2} \geq 1 - \delta$, we get $c \geq \frac{1}{2\epsilon^2} \ln \frac{2}{\delta}$. $\square$

To avoid computing influence score for every event in C when retrieving top-K influential events, we adopt the threshold algorithm [10]. The key observation is that, the geo-social influence of one LBSN user to another must not exceed τ. Thus, for an event e, its score will never be larger than $\tau(|S_e| - 1)$. Based on this observation, Algorithm 3 gives the details of the threshold algorithm. As shown, we first build a list L of all LBSN events, sorted in the descending order of cardinality. Then we initialize an empty result set A with a fixed size k, and keep track of the least ranking score in A as *threshold*. Next, we scan L sequentially and progressively update A. The tricky part is that L is unnecessary to be scanned entirely. For the event e currently being processed, if $\tau(|S_e| - 1)$ is below $threshold$, it is ensured the score of any following event will be below $threshold$ as well. Thus, further scanning will not generate top-K results any more and the algorithm safely terminates.

Algorithm 3: getTopKEvents(G, K)

```
1  Build a list $L$ of all events in $G$
2  Sort $L$ in the descending order of event cardinality
3  Initialize an empty set $A$ of a fixed size $K$
4  foreach event $e$ in $L$ do
5      $threshold$ = the least influence score in $A$
6      if $|A| = k$ and $\tau(|S_e| - 1) \leq threshold$ then
7          break
8      Compute $score(e)$ with sampling technique
9      if $|A| < k$ or $score(e) > threshold$ then
10         Update $A$ with $e$
11 return $A$
```

6. EXPERIMENTS

In this section, we empirically evaluate the effectiveness and efficiency of the proposed algorithms. First in Section 6.1, we demonstrate the effectiveness of our geo-social influence evaluation framework with a toy example. Then we examine the performance of the proposed algorithms on a real-life LBSN data set in Section 6.2. Finally, we study the scalability of the algorithms with synthetic data sets in Section 6.3. All algorithms were implemented in JAVA and the experiments were conducted on an Intel Core 2 Duo 2.93Ghz PC with 4GB memory.

6.1 A Toy Example

We first apply our geo-social influence computation framework to the running examples described in Figure 1, and see if the proposed measures can indeed effectively compute user influence and discover influential LBSN events. We run the GI algorithm with the attenuation parameter $\tau = 0.5$, and the iteration number $k = 20$. Meanwhile, we compute geographical influence with the parameters $\alpha = 6.0, \beta = -1.45$ (obtained from Figure 3). Table 2 reports the social and geographical influence of each user, as well as the scores of the three events *iPhone, KFC* and *Golf.* Recall that, given an event e and its user set S_e, there are two versions of event score, depending on different aggregate functions (MAX or AVERAGE). We denote by **M-score** the maximum user influence of the users in S_e, by **A-score** the average user influence of the users in S_e.

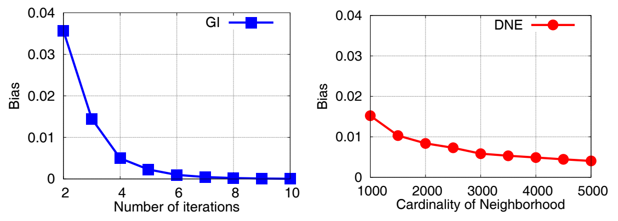
We can see, user u_2 has the largest geo-social influence to the other users. This agrees with our intuition that u_2 can be easily hit by u_1 and u_3, and since their geographical distances are small, u_1 and u_3 are likely to visit the Apple store near u_2. Among the three events, *iPhone* has the largest score since its users are close both geographically and socially. In contrast, *KFC* has low social and geographical scores as its users scatter randomly in the LBSN. *Golf* has high social influence but relatively low geographical influence, due to the fact that although its users are socially close, they are distant to each other geographically.

6.2 Performance Study

In this subsection, we evaluate the performance of our proposed algorithms based on a large real data set obtained from Gowalla. We crawled this data set during a three-month period between

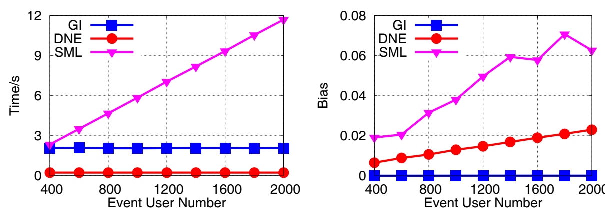
Table 2: Geo-social influences of iPhone, KFC, and Golf users.

Event	User Social Influence	User Geographic Influence	User Influence	M-Score	A-Score
iPhone	$s_{u_1}(u_2) = 0.208$, $s_{u_1}(u_3) = 0.211$ $s_{u_2}(u_1) = 0.307$, $s_{u_2}(u_3) = 0.229$ $s_{u_3}(u_1) = 0.306$, $s_{u_3}(u_2) = 0.225$	$g_{u_1}(u_2) = 0.915$, $g_{u_1}(u_3) = 0.915$ $g_{u_2}(u_1) = 0.915$, $g_{u_2}(u_3) = 0.915$ $g_{u_3}(u_1) = 0.915$, $g_{u_3}(u_2) = 0.915$	$f_{u_1} = 0.383$ $f_{u_2} = 0.490$ $f_{u_3} = 0.486$	0.490	0.453
KFC	$s_{u_2}(u_5) = 0.027$, $s_{u_2}(v_7) = 0.057$ $s_{u_5}(u_2) = 0.018$, $s_{u_5}(u_7) = 0.057$ $s_{u_7}(u_2) = 0.038$, $s_{u_7}(u_5) = 0.057$	$g_{u_2}(u_5) = 0.445$, $g_{u_2}(u_7) = 0.162$ $g_{u_5}(u_2) = 0.445$, $g_{u_5}(u_7) = 0.162$ $g_{u_7}(u_2) = 0.162$, $g_{u_7}(u_5) = 0.162$	$f_{u_2} = 0.021$ $f_{u_5} = 0.017$ $f_{u_7} = 0.015$	0.021	0.017
Golf	$s_{u_6}(u_7) = 0.306$, $s_{u_6}(u_8) = 0.225$ $s_{u_7}(u_6) = 0.211$, $s_{u_7}(u_8) = 0.208$ $s_{u_8}(u_6) = 0.229$, $s_{u_8}(u_7) = 0.307$	$g_{u_6}(u_7) = 0.255$, $g_{u_6}(u_8) = 0.162$ $g_{u_7}(u_6) = 0.255$, $g_{u_7}(u_8) = 0.255$ $g_{u_8}(u_6) = 0.255$, $g_{u_8}(u_7) = 0.162$	$f_{u_6} = 0.114$ $f_{u_7} = 0.107$ $f_{u_8} = 0.108$	0.114	0.109



(a) GI convergence (b) DNE convergence

Figure 5: The convergence of GI and DNE ($\tau = 0.5$).



(a) Time cost vs. $|S_e|$ (b) Bias vs. $|S_e|$

Figure 6: Performance *vs.* $|S_e|$ ($\tau = 0.5$).

September 2011 and December 2011. There are 329839 users with 1679245 undirected social links among them, those users have altogether visited 2973453 distinct POIs. We consider each POI as an event and the users who have checked-in at the POI as the participants of that event. The location where a user most recently checked-in is regarded as his location.

6.2.1 Performance of PHT Computation

We study the performance of the GI and DNE algorithms for computing PHT under two performance metrics, namely **time** and **bias**. Given an event e and a user $u \in S_e$, **time** is the total elapsed time of computing the PHTs of all other users in S_e to u, and **bias** is the sum of absolute PHT approximation errors. For comparison, we also implemented the Simulation algorithm (abbreviated as SML) [13], which is essentially a sampling based algorithm for computing *decayed hitting time*. SML can be slightly modified in the following way to compute PHT: Given an event e and a user $u \in S_e$, to compute the PHTs of all other users in S_e to u, we independently run c random walk simulations from each $v \in S_e - \{u\}$. Each random walk stops when u is hit or a maximum number of steps s is reached. Then, we derive the average path weight of the c simulations as v's PHT.

Convergence of GI and DNE. We first check the convergence of the GI and DNE algorithms. GI has one parameter (the number of iterations k) and DNE has two parameters (the size of neighborhood m and the number of local iterations k). For GI, we investigate its convergence w.r.t k. For DNE, we find that the bias does not vary much when the number of iterations is larger than 5. Thus, we set $k = 5$ and investigate its convergence w.r.t m.

We randomly choose a popular event e ($|S_e| > 1000$) from the Gowalla data set along with an arbitrary user $u \in S_e$, and then compute the PHTs of other users in S_e to u. Such a process is repeated 100 times and the average results are reported. As shown in Figure 5, both algorithms converge rapidly when the respective parameters k and m increase. Moreover, the bias of DNE is only slightly larger than GI, even when the cardinality of neighborhood is as small as 1000. It suggests that DNE empirically provides a good estimation of PHT.

Performance *vs.* Event Size. In this set of experiments, we compare the performance of the GI, DNE, and SML algorithms for events of different sizes. We set $k = 10$ for GI; and $m = 1000,\ k = 5$ for DNE. Such parameter settings are adequate to ensure GI and DNE provide good estimations of PHTs (Figure 5). For SML, larger c and s values lead to tighter estimations of PHTs, and smaller c and s values contribute to better efficiency. We set $c = 1000$ and $s = 20$, as we find that this setting makes SML reach a good balance between estimation tightness and running time.

Figure 6 reports the results when event size varies from 400 to 2000. As shown, the running time of SML grows linearly while that of GI and DNE stays constant, yet SML still causes much larger bias than GI and DNE. This is expected. Although appealing for its simplicity, SML suffers from two shortcomings: (1) SML needs to perform c random walk simulations for each user $v \in S_e - \{u\}$, indicating that the time cost will grow rapidly as $|S_e|$ increases. In contrast, GI and DNE both compute the PHTs of all users at one time and thus are $|S_e|$-independent. (2) SML is a non-deterministic approximate algorithm and its bias may fluctuate randomly. GI and DNE, on the contrary, are deterministic algorithms with tight theoretical error bounds.

Comparing the performance of GI and DNE, we can see DNE is much more efficient than GI, but it has larger bias than GI. This is because DNE excludes vertices that are too faraway from the target vertex, and thus sacrifices a little precision for speed. Fortunately, since the PHT of each excluded vertex is always small (Lemma 2), the total bias of DNE is still acceptable even when the event size is quite large (0.023@2000).

6.2.2 Influential Events Discovery

We proceed to evaluate the influence scores for different events in the Gowalla data set. Since DNE is much faster than GI and also provides quite tight estimations for PHTs, we employ the DNE algorithm to compute PHT, with the parameters $m = 1000,\ k = 10$. Using the *ridge regression* technique as described in Section 4.2, we obtain the parameters of the power law distribution for computing geographical influence: $\alpha = 6.0, \beta = -1.25$.

Again, given an event e, we denote by **M-score** the maximum user influence of the users in S_e, and by **A-score** the average user influence of the users in S_e.

Event Scores of Different Categories. The 2973453 Gowalla events distribute in 416 different categories. In this set of experiments, we look into how the M-score and A-score vary across different categories. For this purpose, we choose 10 representative categories and compute the M-score and A-score for every event in each category, with the attenuation parameter $\tau = 0.8$. Table 3 presents the largest M-score and A-score in each category.

Table 3: Largest M-scores and A-scores for 10 representative categories in Gowalla ($\tau = 0.8$).

Category Name	M-Score	A-Score
library	21.782	2.783
bar	12.374	2.344
theater	14.590	2.038
resort	11.015	1.776
historic landmark	10.258	1.645
apple store	15.376	1.521
asian food	8.621	1.225
apparel	7.370	0.742
airport terminal	2.512	0.181
hotel	1.738	0.093

We can see, the M-score and A-score are high for the categories *library*, *bar*, and *theater*. This indicates that the visitors of a typical POI in these categories are close to each other both geographically and socially. For example, the POI reaching the largest M-score 21.782 in the category *library* is the *M.D. Anderson Library* in the University of Houston, which has 97 visitors. Our manual investigation suggests that these visitors are mostly students from the University of Houston, each one having many social links with the others. Not difficult to imagine, if the *M.D. Anderson Library* launches a new service and employs one of the 97 student to post this message on-line, the information can quickly reach the other students and may drive them to experience it. Likewise, the high scores of the categories *bar* and *theater* imply the users form social communities for some hobby (drinking or watching drama), and meanwhile live congregationally.

In contrast, the M-score and A-score are low for the categories *hotel* and *airport terminal*. This is because the social relations among hotel and airport visitors are usually weak, resulting in low social correlations. Moreover, people visiting a hotel mostly do not live in the city where the hotel locates, and it is the same for people visiting an airport.

For the rest five categories, the M-score and A-score of the categories *apparel* and *food* are lower than the *apple store* category. This means people are more likely to follow their friends' preferences when they are purchasing a digital product than purchasing clothes or food. The scores of the categories *resort* and *historic landmark* are quite high. Interestingly, we find the events reaching largest M-scores are usually hot places with thousands of visitors, whereas the events reaching largest A-scores are some local places with only a few visitors that are highly correlated. Take the category *resort* as an example, the event corresponding to the largest M-score 10.258 is the *Disney's Grand Floridian Resort*, which has 739 visitors from both Florida and other states. In contrast, the event with the largest A-score is the *Bauwagen Estenfeld*, which is a local resorting club located in Germany, having only 11 visitors living in Estenfeld.

Top-K Events in Each Category. In this set of experiments, we zoom in to retrieve top-K influential events in one category. In the following, we report the results for the categories *apparel* and *asian food*.

Table 4 and 5 are the top-five influential events in category *apparel* with the largest M-scores and A-scores, respectively. From Table 4, we find that the events with large M-scores are usually famous brands (e.g., Zara) having a significant number of consumers. One can also observe that hot brands with large M-scores do not necessarily have large A-scores. It is because there exist many consumers who have weak connections with the other consumers of those hot brands, the low influences of such consumers drastically pull down the A-scores. In contrast, the events with large A-scores are mostly some brands not so famous, but possessing a set of consumers that are highly correlated. Take *Bergner's* as a concrete example, it is a department store offering mid-line to higher end merchandise. The 15 customers of *Bergner's* all live in central Illinois, and there are as many as 68 social links among them.

Another interesting finding is that, the retrieved events are mostly stores selling women's apparel. This implies that females are more likely to discuss with their friends and follow their choices, leading to the formation of fan communities for some brands. In comparison, the purchasing behaviors of males do not exhibit so much aggregation.

Table 4: Top-5 events with the largest M-scores in category "apparel".

Event	M-Score	A-Score	$\lvert S_e \rvert$
ZARA Central World	7.37	0.227	107
Bloomingdale's	7.11	0.054	198
Louis Vuitton	6.72	0.190	61
JCPenney	6.38	0.181	227
Jack Jones	6.37	0.067	524

Table 5: Top-5 events with the largest A-scores in category "apparel".

Event	A-Score	M-Score	$\lvert S_e \rvert$
Bergner's	0.742	3.17	15
American Eagle Outfitting	0.717	2.51	14
Deja Vu Underground	0.708	2.37	12
Hot Topic Women's Apparel	0.679	2.78	11
Sweet & Tender	0.654	2.02	13

Table 6 and 7 present the top-five events in category *asian food*, along with the POI locations. Somewhat surprisingly, few of the top POIs are located in Asia, especially for the POIs with large A-scores. The reason may be that people visiting Asian restaurants in America or Europe are usually Asian immigrants or Asian students studying abroad. It is highly possible that they have strong social relations with each other, forming fan communities for the restaurants serving hometown food. Our manual investigation verifies this phenomenon. Take the restaurant *China Tiger* as an example. We find its consumers are mostly Chinese students studying in Helsinki, they live quite close to each other and there are a significant number of social links in the ensemble.

6.3 Scalability Study

To examine the scalability of the proposed algorithms, we exploit the publicly available data set LiveJournal[3], a free on-line community whose members are highly active. The LiveJournal data set has 4847571 users and 68993773 directed social links. We do not have user locations or events for this data set, so we only use it to check the scalability of the GI and DNE algorithms. Specifically, we extract synthetic subgraphs with different graph sizes (in terms of vertex number) from LiveJournal. Meanwhile, we randomly select 100 target users. Then we run the GI and DNE algorithms on each subgraph to compute PHTs to those 100 users and report the average running time. Figure 7 is the result. As shown, while

[3]http://snap.stanford.edu/data/soc-LiveJournal1.html

Table 6: Top-5 events with the largest M-scores in category "asian food".

Event	M-Score	$\|S_e\|$	Location
Shokudo	8.62	122	Honolulu, US
Pho Lien Hoa	8.44	94	Oklahoma, US
Bedok 85 Food Centre	8.41	148	Singapore
Kyoodai Restaurant	8.13	67	Phuket, Thailand
Goma Tei	7.87	95	Honolulu, US

Table 7: Top-5 events with the largest A-scores in category "asian food".

Event	A-Score	$\|S_e\|$	Location
Saigon Bay	1.225	11	Stockton, US
China Tiger	1.204	14	Helsinki, Finland
King Wah	1.178	17	Daly City, US
Ravintola Annapurna	1.155	12	Helsingfors, Finland
ABC Cafe Bakery	1.147	11	San Francisco, US

the running time of GI grows with graph size, that of DNE remains constant. This is because DNE only iterates over vertices in the neighborhood of a target vertex, and thus works more efficiently than GI for large LBSNs.

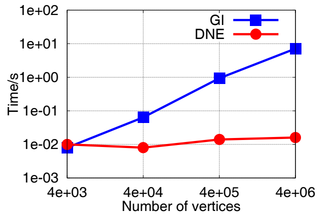


Figure 7: The scalability of GI and DNE.

7. CONCLUSION

This paper provided an in-depth study of geo-social influence in location-based social networks. We proposed a unified metric to quantify mutual user influence on event-level granularity. This metric combined a novel social proximity measure named *penalized hitting time*, with a geographical weight function modeled by *power law distribution*. Based on this metric, we addressed the *user influence evaluation* and *influential events discovery* problems. Specifically, we proposed two approximation algorithms, namely GI and DNE, to efficiently compute user influence; and we adopted the *sampling technique* and the *threshold algorithm* to retrieve top-K influential events. Our experimental results demonstrated that the proposed algorithms are effective, efficient, and scalable.

There are some potential future directions of this work. In particular, besides the power-law distribution, it is promising to consider other methods [9] for modeling the geographical mobility patterns of users. Moreover, it is also interesting to explore the performance of different combinations of geographic influence and social influence in addition to their product.

8. ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation of China (Grant No. 60970124, 60903038, and 61170034).

9. REFERENCES

[1] http://goo.gl/Y39Gm.

[2] N. Agarwal, H. Liu, L. Tang, and P. S. Yu. Identifying the influential bloggers in a community. In *WSDM*, pages 207–218, 2008.

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

[4] E. Bakshy, J. M. Hofman, W. A. Mason, and D. J. Watts. Everyone's an influencer: quantifying influence on twitter. In *WSDM*, pages 65–74, 2011.

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

[6] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998.

[7] M. Chen, J. Liu, and X. Tang. Clustering via random walk hitting time on directed graphs. In *AAAI*, pages 616–621, 2008.

[8] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: user movement in location-based social networks. In *KDD*, pages 1082–1090, 2011.

[9] A. Clauset, C. R. Shalizi, and M. E. J. Newman. Power-law distributions in empirical data. *SIAM Review*, 51(4):661–703, 2009.

[10] R. Fagin, A. Lotem, and M. Naor. Optimal aggregation algorithms for middleware. In *PODS*, 2001.

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

[12] M. C. González, C. A. H. R., and A.-L. Barabási. Understanding individual human mobility patterns. *CoRR*, abs/0806.1256, 2008.

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

[14] G. Jeh and J. Widom. Scaling personalized web search. In *WWW*, pages 271–279, 2003.

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

[16] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

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

[18] Q. Mei, D. Zhou, and K. W. Church. Query suggestion using hitting time. In *CIKM*, pages 469–478, 2008.

[19] P. Sarkar and A. W. Moore. A tractable approach to finding closest truncated-commute-time neighbors in large graphs. In *UAI*, pages 335–343, 2007.

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

[21] S. Scellato, A. Noulas, and C. Mascolo. Exploiting place features in link prediction on location-based social networks. In *KDD*, pages 1046–1054, 2011.

[22] P. Singla and M. Richardson. Yes, there is a correlation: - from social networks to personal behavior on the web. In *WWW*, pages 655–664, 2008.

[23] J. Tang, J. Sun, C. Wang, and Z. Yang. Social influence analysis in large-scale networks. In *KDD*, pages 807–816, 2009.

[24] M. Ye, P. Yin, W.-C. Lee, and D. L. Lee. Exploiting geographical influence for collaborative point-of-interest recommendation. In *SIGIR*, pages 325–334, 2011.