Geology: Modular Georecommendation In Gossip-Based Social Networks

Jesús Carretero[1], Florin Isaila[1], Anne-Marie Kermarrec[2], François Taïani[2,3], Juan M. Tirado[1]
jcarrete@inf.uc3m.es, fisaila@inf.uc3m.es, anne-marie.kermarrec@inria.fr, f.taiani@lancaster.ac.uk, jtirado@inf.uc3m.es
[1]Universidad Carlos III, [2]INRIA Rennes Bretagne Atlantique, [3]Lancaster University

***Abstract*—Geolocated social networks, combining traditional social networking features with geolocation information, have grown tremendously over the last few years. Yet, very few works have looked at implementing geolocated social networks in a fully distributed manner, a promising avenue to handle the growing scalability challenges of these systems. In this paper, we focus on georecommendation, and show that existing decentralized recommendation mechanisms perform in fact poorly on geodata. We propose a set of novel gossip-based mechanisms to address this problem, in a modular similarity framework called GEOLOGY. The resulting platform is lightweight, efficient, and scalable, and we demonstrate its advantages in terms of recommendation quality and communication overhead on a real dataset of 15,694 users from Foursquare, a leading geolocated social network.**

***Keywords*-distributed systems, social networks, geolocation, gossip protocols**

I. INTRODUCTION

Geolocation, the ability to precisely locate users geographically, is increasingly being exploited by social networks (e.g. Twitter, Foursquare, Gowalla) to offer new and improved services to their users. This includes place recommendations, geographic social games, and geo-based tracking. Such geolocated services have grown tremendously over the past few years: for instance, Foursquare, one of the leading geolocated social networks, recently announced 10,000,000 registered users [1] after just over 2 years of existence. This spectacular growth raises a number of engineering and scalability challenges on how to combine traditional social networking services (e.g. search, recommendation, notification), with geolocation information in architectures that are scalable, resilient, and efficient.

One promising approach that has been explored for traditional social networks, is based on fully decentralizing these services, by relying on highly scalable mechanisms such as gossip protocols [2], [3] or distributed hash tables [4]. These decentralised versions permit users to control their data privacy and to establish ad-hoc communities. They typically use a peer-to-peer model to self-organize the involved users in implicit communities and provide personalized social services such as search, query expansion, content and friends recommendations. None of these approaches have however been applied to geodata so far, and it is unclear how well they work in a geolocated context. This is because geodata are by nature different from traditional social content (friends, subscriptions, tagged items, favourite sites, etc.). In particular, the amount of information that can be inferred from a location differs from other user attributes such as subscriptions. For example, we can infer more information from a person who is subscribed to a runners magazine than from a person who frequently visits a park, as people sharing an interest in a park or in a popular place do not necessarily share the same tastes. This means that high-frequency locations (e.g. airports, parks) typically carry less semantic weight than high-frequency content (Obama, Lady GaGa, etc.). Similarly, the distribution of locations is more strongly skewed towards niche places, only visited by a few users, making it more difficult to extract meaningful information in a fully distributed setting.

To investigate these issues, we focus in this paper on georecommendation (the ability to recommend places of possible interest to a user), a key feature of most geolocated social networks. We propose a modular framework, GEOLOGY, to explore the design of distributed georecommendation services based on gossip protocols. GEOLOGY includes three mechanisms (compaction, the Adamic/Adar metric, and sampling) all aimed at alleviating the specific challenges of georecommendation in fully distributed systems. We demonstrate how combining these three mechanisms into a metric we have called GEOTROPY delivers a superior solution than other alternatives on a geolocated dataset involving 15,694 Foursquare users in the San Francisco Bay area. We show that our approach is able to recommend an average of 40% of the total possible locations of a user in 10 rounds, while incurring very reasonable network overhead.

The paper is organized as follows. We first introduce geolocated social networks, and the challenges they bring in terms of fully distributed georecommendation (Section II). We then present the GEOLOGY framework, a modular architecture for fully-decentralized recommendation systems, along with GEOTROPY, a lightweight and efficient solution to decentralized georecommendation (Section III). We present the dataset we have used for our evaluations in Section IV, and we evaluate GEOTROPY against alternative approaches in Section V. We finally discuss related work (Section VI) and conclude (Section VII).

II. BACKGROUND AND PROBLEM STATEMENT

In this paper, we look at the particular problem of georecommendation in fully distributed geolocated social networks.

Figure 1. A Foursquare notification on a Twitter timeline

We discuss both concepts below, and explain why geodata needs a special treatment.

A. Geolocated social networks

In a geolocated social network (e.g. Foursquare[1]) users publish the locations they visit using a GPS-enabled device, and use this information to interact with other users (known as friends). Locations are typically organized in *venues* (restaurant, cinemas, home), and the action of a user entering a venue is known as a *checkin*. New venues are contributed by users when they check into a location that has not been registered yet. Users can decide to leave tips about a particular venue, which are then available to anyone visiting this venue. They can publish a short message associated with each checkin; they can choose to be notified of their friend's checkins or tips; they can recommend locations to friends; earn coupons associated with commercial venues; gather badges (e.g. 'Foodie', 'Globe Trotter'). Locations are also often organized in *categories* ('restaurant', 'bar', 'cinemas'), possibly structured into a hierarchical taxonomy.

In a typical geolocated social network, users can also link their account to other social networks (Twitter, Facebook), for instance to publish their checkins and badges on their Twitter timeline, or synchronize their list of friends in both systems. For instance Figure 1 shows an example of a checkin notification on Dennis Crowley's public timeline, one of the co-founder and current CEO of Foursquare.

B. Georecommendations

A georecommendation service is a particular feature of Geolocated Social Networks that uses knowledge about a user (her *profile*, which may include her checkin history, friends, interests, current location) to recommend new locations this user might be interested in. This is a special case of recommendation, and, as such, is amenable to centralized recommendation approaches, for instance applied to link prediction and collaborative filtering [5], [6]. These centralized methods typically assume global knowledge of all users' profiles to compute recommendations, and might involve complex global parameter optimization (such as minimizing a matrix product over the system's entire dataset). They are, as such, difficult to implement efficiently in a distributed setting.

In this paper, we therefore take an opposite perspective, and consider how a lightweight fully distributed georecommendation system might be implemented. Gossip protocols and peer-to-peer architectures are natural candidates to develop this type of solution, and have been recently proposed to implement fully decentralized social networking applications [3].

[1]https://foursquare.com/

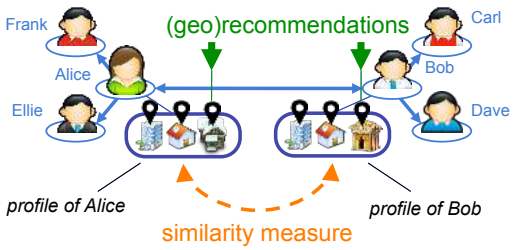

Figure 2. A typical peer-to-peer social network with implicit relationships

A typical gossip-based peer-to-peer social network works as follows: The network seeks to organize users in a distributed overlay so that similar users tend to be connected together, as shown in Figure 2. In this figure Alice and Bob have similar checkin profiles, and have therefore been selected to be neighbors of each other (using a mechanism we detail in Section III). The intuition behind this approach is that if Alice and Bob have similar profiles, then Alice might know locations of interest to Bob, and reciprocally.

C. The challenge of similarity

The success of a peer-to-peer social network hinges however crucially on the quality of the similarity measure used to detect similar users (the dashed line in Figure 2). It is in particular unclear whether measures used for traditional social content (friends, videos, music) such as cosine similarity and Jaccard coefficients [7] do actually work well on geodata. High-frequency locations (e.g. airports, parks) carry in particular less semantic weight than high-frequency content (Obama, Lady GaGa, etc.). Similarly, the distribution of locations is more strongly skewed towards niche locations, only visited by a few users, making it more difficult to extract meaningful information in a fully distributed setting.

One strategy used in centralized recommendation approaches to mitigate the influence of high-frequency items associated with a majority of users (such as airports), relies on metrics derived from information theory [7], such as information content, entropy, and the Adamic/Adar metric [8], a variation on entropy. These metrics require however some global knowledge about location distributions in the system, which might potentially considerably increase their computation cost in a fully distributed environment.

D. Contributions

In this paper we explore this particular research challenge, and investigate the inherent tension between the quality of georecommendations returned by a fully distributed social network, and the network cost incurred to generate these recommendations. To support this analysis, we propose a modular similarity framework for gossip-based recommendation systems, GEOLOGY, that combines three mechanisms (compaction, Adamic/Adar metric, and sampling) in a fully distributed setting. This framework leads us to propose one specific combination, GEOTROPY, that implements a lightweight version of the Adamic/Adar metrics adapted to the specifics of

gossip-based recommendation systems. Both the framework, and the resulting GEOTROPY solution are, to the best of our knowledge, new. We also demonstrate how current fully-decentralized recommendation systems can be expanded to deliver an advantageous trade-off between communication cost and recommendation quality.

III. APPROACH: GEOLOGY AND GEOTROPY

We first present our system model and working assumptions, before providing an overview of the architecture of GEOLOGY. Then we discus in detail its similarity framework, which leads us to present GEOTROPY, the particular variant we advocate for in this paper.

A. System model

We assume a peer-to-peer architecture similar to that of Figure 2, in which each peer (a smart phone, a tablet, a laptop) is associated with a user. Peers can connect to each other using point-to-point networking, but they only have a partial and limited view of the rest of the system. (We return to this point below, when we present how GEOLOGY works.) In the rest of the paper, we do not distinguish between peers and users, and use both terms interchangeably.

We further assume each peer collects a *checkin profile* of the checkins made by its associated user, represented in rounded boxes in Figure 2. These checkin profiles are for instance collected using a GPS receiver, or using network triangulation, with possibly additional confirmation by the user, as they are in Foursquare. For the purpose of this paper, we model the checkin profile of user u_i as a hash of key-value pairs $\vec{M}_i$ so that $\vec{M}_i[l_k]$ is the number of times u_i visited location l_k. We also note $L_i = \{l_1, ..., l_m\}$ the set of locations visited by user u_i (i.e. L_i is the set of keys of M_i).

Finally, we assume locations are organized in categories (e.g. 'restaurants', 'bars', 'cinemas'), as they are in Foursquare. The number of categories is fixed: Foursquare for instance has 355 categories. Similarly to locations, we note $\vec{C}_i$ the hash of key-value pairs such that $\vec{C}_i[c_k]$ is the number of times a location classified into category c_k was checked-in by user u_i. We also note $C_i = \{c_1, ..., c_m\}$ the set of location categories visited by user u_i. Although the classification of categories in Foursquare is hierarchical, for simplification purposes we use a flat vector representation.

B. GEOLOGY architecture

GEOLOGY uses a two-layer structure to organize users in similarity neighborhoods (Figure 3). It follows in this respect other gossip-based social networks that work along the same principles [3].

Each layer provides a peer-to-peer overlay, in which users (or peers) maintain a fixed list of neighbors (or *views*). For instance, in Figure 3, Alice is connected to Bob, Carl, and Dave in the bottom RPS (Random Peer Sampling) layer, and to Carl and Bob in the upper layer (clustering). Periodically, each peer selects a user from its view and exchanges information about its neighbors with the final goal of converging to an optimal topology of users with 'similar' checkin profiles in the top layer (*clustering*). In the GEOLOGY framework, the similarity is a modular construct, and we return to it just below.

C. Convergence

More precisely, the bottom RPS layer, allows each peer to periodically obtain a random sample of the rest of the network and thus guaranties the convergence of the second layer (clustering), while making the overall system highly resilient against churn and partitions. This is achieved by having peers exchange and shuffle their neighbors list in *periodic gossip rounds* to maximise the randomness of the RPS graph over time [9]. For reason of efficiency, each peer does not however communicate with all its neighbors in each round, but instead randomly selects one of its neighbors in its RPS view to interact with. Alice might for instance request Carl's list of RPS neighbors (which includes Ellie), and randomly decide to replace Dave by Ellie (received from Carl) in her RPS view.

The clustering layer sits on top of the RPS layers, and implements a local greedy optimisation procedure that leverages both neighbors returned by the RPS, and current neighbors from the clustering views [10], [11]. A peer (say Alice in Figure 3) will periodically update its list of similar neighbors with new neighbors found to be more 'similar' to her in the RPS layer. This guarantees convergence under stable conditions, but can be particularly slow in large systems. This mechanism therefore is complemented by a swap mechanism in the clustering layer (Figure 4), whereby two neighboring peers (here Alice and Bob) exchange their neighbors lists (so peers that are already close to them, Step 1), and seek to construct a better neighborhood based on the other peer's information (Step 2 in Figure 4).

Figure 5 shows in more detail how this second step unfolds for Alice within one gossip round. Alice uses the peers sent over to her by Bob (**b**) to construct alternative neighborhoods (**c**), and requests the checkin profiles of these peers. Alice then uses the GEOLOGY similarity framework (**d**) (described just below), to rate the different neighborhoods she could construct (**e**), and selects as her new neighborhood the one with the highest rating (**f**). She then uses the checkin profiles of her new neighborhood to select locations her neighbors have visited, but she has not, as recommendations for this round (**g**).

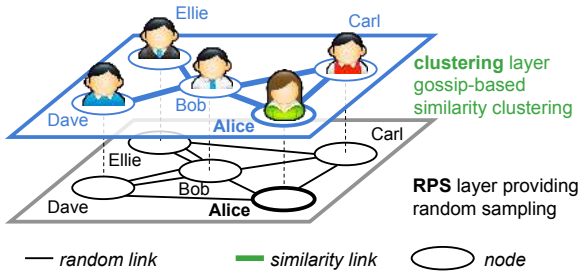

Figure 3. Gossip-based distributed clustering

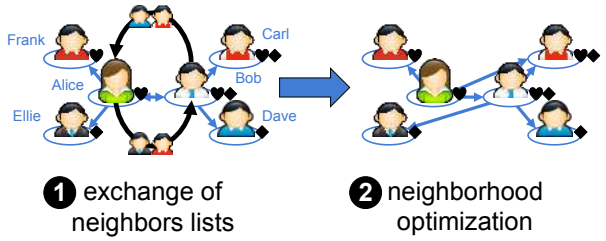

Figure 4. Clustering mechanism targetting to define an optimal neighborhood. In this example, after exchanging their profiles, Alice and Bob modify their neighbors in order to be connected with more users who share their same interests.

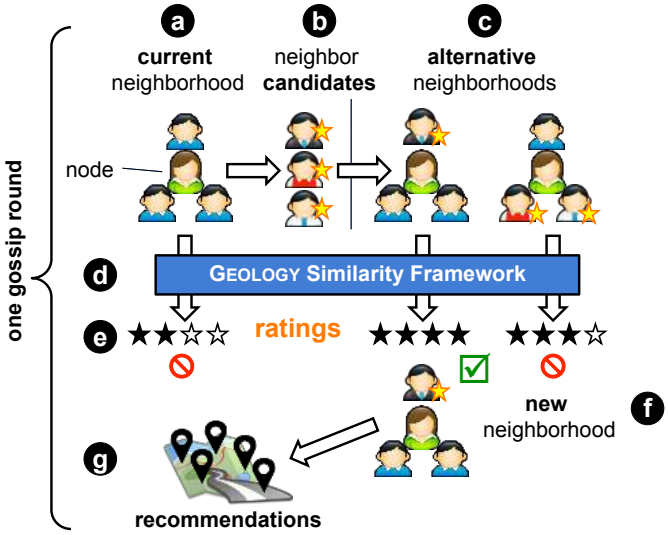

Figure 5. Gossip-based recommendations in GEOLOGY

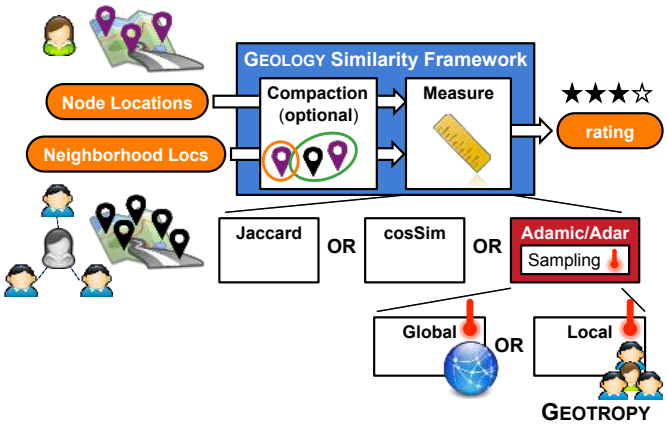

Figure 6. The GEOLOGY similarity framework

Table I
METRIC VARIANTS WITHIN THE GEOLOGY FRAMEWORK

Name	Compaction	Measure	Sampling
GEOTROPY	Yes	Adamic/Adar	local
GEOTROPY-*loc*	No	Adamic/Adar	local
g-GEOTROPY	Yes	Adamic/Adar	global
g-GEOTROPY-*loc*	No	Adamic/Adar	global
jac-catg	Yes	Jaccard	N/A
jac-loc	No	Jaccard	N/A
cos-catg	Yes	cosSim	N/A
cos-loc	No	cosSim	N/A

D. The GEOLOGY similarity framework

A crucial component of the mechanism we have just described is the similarity measure used to detect how well a peer and a neighborhood fit together. For instance a measure that favors popular locations might lend too much weight to highly visited places, and will tend to produce poor results. Conversely, a measure that requires some type of global knowledge (such as global distributions of users per location) might converge very slowly.

In GEOLOGY, the similarity measure is produced by a modular framework (the GEOLOGY *similarity framework*), shown in Figure 6. This framework is composed of three main modules: *compaction*, *measure* and *sampling*. The compaction module is optional and compacts a user's checkin profile using categories instead of locations. This reduces the amount of data that needs to be exchanged, as compaction can occur locally before a profile is sent over the network.

The measure module scores new neighborhoods according to three basic metrics: *Jaccard*, *cosinus similarity* (noted *cosSim*) [7] and *Adamic/Adar* [8] (although other metrics can be used). These metrics are explained in detail in Section III-E. For the moment, suffices to note that *Jaccard* and *cosine similarity* are *local* metrics that can directly be computed using the profiles of a user and that of her neighbors. They are therefore reasonably lightweight in terms of network cost. By contrast the *Adamic/Adar* metrics mitigates the influence of very frequent items (locations or categories—depending on whether compaction is used or not), thus addressing the potential problems presented by very popular locations (airports, parks) that are low on semantic content. Unfortunately, *Adamic/Adar* requires some global knowledge of distributions of items (locations or categories), and is therefore potentially much more expensive to implement in a fully decentralized system.

Finally, the sampling module selectively collects users profiles across the system to estimate the frequency distributions needed to compute the *Adamic/Adar* metrics. We consider two sampling methods: The *global* method uses the RPS layer to progressively accumulate knowledge regarding global distributions found in the system (either for locations or categories, depending on the status of compaction). The *local* method only uses the profiles of a peer's neighbors to compute local distributions of categories and locations. The *local* method is thus cheaper and faster (in particular in large systems), but also more approximate. One additional implication is that the ratings of different candidate neighborhoods (**c** in Figure 5) are computed using the same distribution in the same round with the *global* sampling approach, whereas the distribution used is dependent of each neighborhood with the *local* approach.

E. Resulting similarity metrics

The combination of the three modules yields 8 different similarity measures, which are listed in Table I. One combination, that we have termed GEOTROPY, is particularly interesting, as it combines a *global* metric (Adamic/Adar), with two optimizations aimed at minimizing its network cost, and improving its convergence speed: compaction (with categories), and local sampling.

The precise definitions of these 8 variants are shown in Table II. Nb denotes the list of neighbors to be evaluated by the similarity measure, and RPS a random sample of peers returned by the RPS layer.

The methods *jac-loc*, *jac-catg*, *cos-loc* and *cos-catg* are versions of the jaccard and cosine similarity [7]. GEOTROPY-*loc*, GEOTROPY , *g*-GEOTROPY-*loc* and *g*-GEOTROPY are based on the Adamic-Adar similarity measure.

For local sampling variants (GEOTROPY-*loc* and GEOTROPY) we define two key-value hash vectors that aggregate item frequencies over a user's neighborhood: $\vec{M}_{Nb}$ for locations, and $\vec{C}_{Nb}$ for categories, respectively. Their corresponding key sets are M_{Nb} and C_{Nb}. With the notation of Section III-A, each vector and each key set is computed in each round with:

$$\vec{M}_{Nb} = \sum_{k \in Nb} \vec{M}_k \quad (1) \qquad \vec{C}_{Nb} = \sum_{k \in Nb} \vec{C}_k \quad (2)$$

$$L_{Nb} = \bigcup_{k \in Nb} L_k \quad (3) \qquad C_{Nb} = \bigcup_{k \in Nb} C_k \quad (4)$$

For global sampling variants (*g*-GEOTROPY-*loc* and *g*-GEOTROPY), we define two vectors: $\vec{M}_{RPS}$ and $\vec{C}_{RPS}$. Both vectors are computed similarly to $\vec{M}_{Nb}$ and $\vec{C}_{Nb}$, but contrarily to their local variant, are accumulated between rounds.

IV. DATASET

We now present the dataset on which we evaluated the GEOLOGY framework, and its GEOTROPY variant, before moving to the evaluation section.

We crawled the profiles and geolocation information of 16,251 users in the San Francisco Bay Area between May and October 2011 from Foursquare. Foursquare does not generally provide access to the history of locations visited by users, for privacy reasons, but allows users to link their Foursquare account to their Twitter timeline. In doing so, Foursquare users can automatically publish on Twitter the sequence of locations they visit (known as *checkins* in Foursquare's vocabulary), as we have shown in Figure 1.

We used a combination of the Twitter `search`, and of the Foursquare `herenow` REST APIs [12] to find users publishing their Foursquare checkins on their Twitter timeline in the San Francisco Bay area, and used the remaining APIs of both services to obtain a trace of the locations recently visited by each users.

The 16,251 users we crawled visited 203,715 locations. To limit the scope of our experiment, and because our dataset only represents a small fraction of Foursquare's user base, we only kept locations from within the San Francisco Bay Area[2]. The selected locations represent 19% (38,175) of the total set, but they tend to be more often visited by users (curve with squares on Figure 7). This is quite understandable given our focus on San Francisco, and is more representative of the level of redundancy a geolocation service such as Foursquare would obtain in a particular region.

More precisely, Figure 7 provides an overview of the difference between the raw set of locations (black circles) and

Location-based	
jac-loc	$\sum_{j \in Nb} \lvert L_i \cap L_j \rvert / \lvert L_i \cup L_j \rvert$
cos-loc	$\sum_{j \in Nb} \frac{\vec{M}_i \vec{M}_j}{\sqrt{\vec{M}_i^2 \vec{M}_j^2}}$
GEOTROPY-*loc*	$- \sum_{l_n \in L_i \cap L_{Nb}} log \frac{\vec{M}_i[l_n]}{\vec{M}_{Nb}[l_n]}$
g-GEOTROPY-*loc*	$- \sum_{l_n \in L_i \cap L_{Nb}} log \frac{\vec{M}_k[ln]}{\vec{M}_{RPS}[l_n]}$
Category-based	
jac-catg	$\sum_{j \in Nb} \lvert C_i \cap C_j \rvert / \lvert C_i \cup C_j \rvert$
cos-catg	$\sum_{j \in Nb} \frac{\vec{C}_i \vec{C}_j}{\sqrt{\vec{C}_i^2 \vec{C}_j^2}}$
GEOTROPY	$- \sum_{c_n \in C_i \cap C_{Nb}} log \frac{\vec{C}_i[c_n]}{C_{Nb}[c_n]}$
g-GEOTROPY	$- \sum_{c_n \in C_i \cap C_{Nb}} log \frac{\vec{C}_k[c_n]}{\vec{C}_{RPS}[c_n]}$

Table II
SIMILARITY METRICS BETWEEN USER u_i AND HER NEIGHBORHOOD Nb

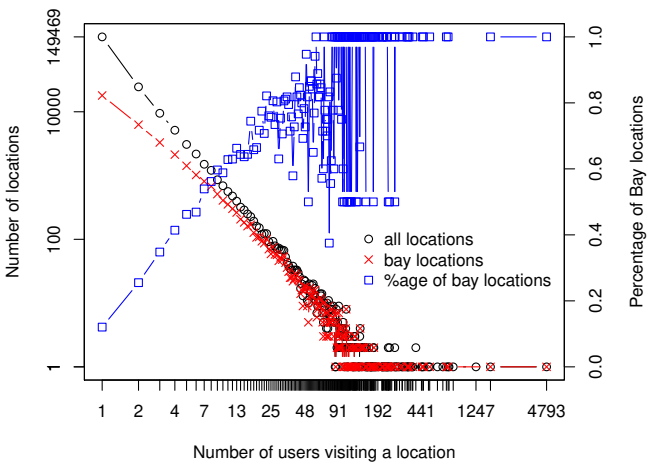

Figure 7. Popularity distribution of crawled locations. Circles represent the number of locations that are visited by a certain number of users and crosses for the Bay Area. Squares represent the value in % for the Bay Area locations

[2]Defined as the union of the 101 cities and towns belonging to the Association of Bay Area Government, http://www.abag.ca.gov/

Table III
MOST POPULAR LOCATION CATEGORIES

Name	#Users	%age Users
San Francisco Int. Airport (SFO)	4793	29.49%
AT&T Park	1644	10.12%
Oakland Int. Airport (OAK)	733	4.51%
San Jos Int. Airport (SJC)	709	4.36%
Golden Gate Bridge	516	3.18%
O.co Coliseum	458	2.82%
Mission Dolores Park	447	2.75%
AMC Loews Metreon 16	374	2.30%
San Francisco-Oakland Bay Bridge	348	2.14%
HP Pavilion	341	2.10%
California Academy of Sciences	339	2.09%
Apple Store	335	2.06%
Ferry Building	305	1.88%
Union Square Park	290	1.78%
Oracle Arena	267	1.64%

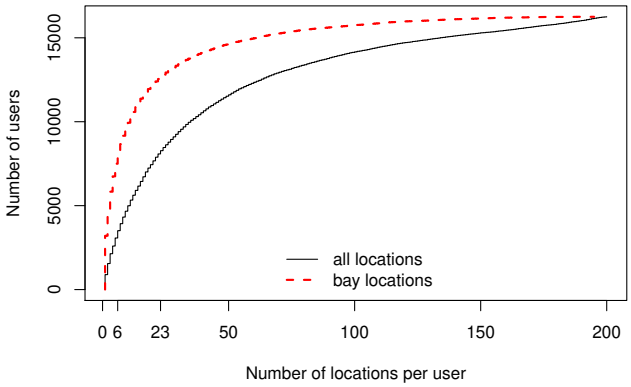

Figure 8. Cumulative distribution of the number of locations per user

Table IV
MOST POPULAR BAY CATEGORIES

Category	#Locations	%age Locations
Coffee Shops	822	4.3%
Offices	544	2.8%
Mexican Restaurants	540	2.8%
American Restaurants	510	2.6%
Grocery Stores	503	2.6%
Bars	420	2.2%
Chinese Restaurants	404	2.1%
Cafés	377	1.9%
Pizza Places	375	1.9%
Italian Restaurants	354	1.8%

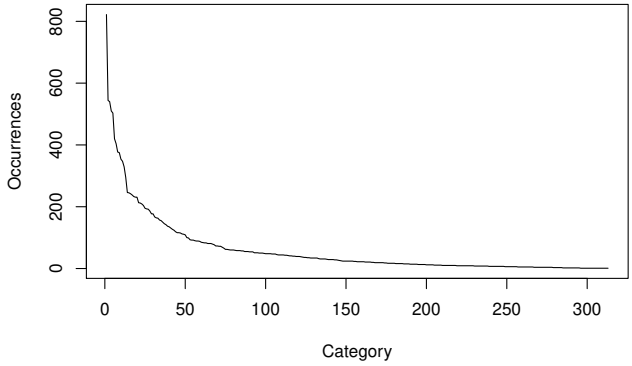

Figure 9. Number of locations per category.

Bay Area locations (red crosses) in term of user distribution on a log-log scale. The chart reads as follows: 149,469 locations were only visited by one user in our dataset (black circle on the top left), among which 18,004 locations are located in the bay (red cross below), which represents 12% of one-user locations (blue square). While global locations show a clear power law (straight line), with a long tail of locations only appearing in the timeline of a few users (73% of locations only appear in the timeline of one user), the tail of bay locations is less spread out (only 47% of bay locations only appear once). More importantly, redundant locations are primarily in the Bay Area: for instance, although bay locations only represent 19% of the total, they represent 77% of the locations visited by more than 25 users.

Table III lists the most popular Bay Area locations in term of visiting users. Quite strickingly, the percentage of users sharing a location drops particularly fast, with only the top 2 locations (SFO Airport and AT&T park), shared by more than 5% of all users. This shows that most locations are *niche* content, and that personalization is crucial to provide users with relevant location recommendations.

Removing locations outside of the Bay Area has naturally an impact on the distribution of locations per user (Figure 8). While the median number of locations per user is 23 (average 42.2) in the complete dataset, this drops to 6 (average 17.8) when only keeping bay locations.

Afterwards, in order to be able to split the location list of each user into a training and a prediction set, we removed users with only one location in their profile. Similarly, because locations only present in the list of one user cannot be predicted (since they only appear once in the dataset), we removed in a last filtering operation all locations visited by only one user, leaving us with a dataset containing 15,694 users, 19,617 locations and 785,288 checkins in the Bay Area. This subset of the original dataset contains elements that can be recommended by our framework, and provides a more *geographically homogeneous* set of users.

Finally, we analyze how the locations are classified by users into the categories provided by Foursquare. Figure 9 shows the number of locations classified per category in the final filtered dataset. From the 355 available categories, 38 categories do not contain any location and 11 only contain one. The distribution of category popularity shows a long-tail effect, with the 30 most important categories covering 50% of all locations. Table IV shows the most popular categories the locations are classified into. We highlight the fact that some categories can have similar meanings like ‘Coffee Shops’ and ‘Cafés’, although they are distinct.

V. Evaluation

This section presents the evaluation results of the metrics proposed in this work. All our experiments were performed using a simulator which has been fed with the information described in Section IV. In particular, the evaluation seeks to answer the following questions. *What is the recall of the proposed methods? How do they converge? Are they stable? What is their communication overhead? What is their sensitivity to different neighborhood sizes, dataset sizes, and location popularity?*

A. Evaluation strategy

Our dataset consists of a single snapshot. In order to simulate a recommendation process, where a peer accepts recommendations from other peers, we define two sets for every peer i: L_i^{learn} and L_i^{eval}. The first set contains the locations that will be used during the neighborhood rating and recommendation stages, the second one is the set of locations used for evaluation. In each round, each peer updates its neighborhood using the method described on Section III-E, with one of the metrics of our framework. Each peer then collects the L_k^{learn} sets of its new neighbors to build its set of recommendations for the round. A correct recommendation is one that is contained in the peer's L_i^{eval} set.

All the experiments are executed for 30 cycles. In the first cycle, the peers are organized randomly without any previous knowledge of the system. To identify the configuration of an experiment we use a tuple indicating the size of each set and the maximum number of neighbors a peer can have. For example, (80, 20, 10) indicates a configuration where L_i^{learn} contains 80% of the locations of peer i, L_i^{eval} contains the remaining 20% and each peer has 10 neighbors. For comparison purposes, when employing different metrics, the content of L_i^{learn} and L_i^{eval} sets is the same for a peer i.

B. Recommendation System

In order to assess the effectiveness of our approach we measure the proportion of locations contained in L_i^{eval} a peer can find using recommendations. We compute the incremental recall calculated as $\frac{|R_k^i \cap L_i^{eval}|}{|L_i^{eval}|}$, where R_k^i is the set of issued recommendations for peer i at cycle k.

Figure 10 shows the average incremental recall over all peers for a (80, 20, 20) configuration. The results show that all the methods using compaction return better results than those directly using locations. Particularly, GEOTROPY and *g*-GEOTROPY obtain the best results with a fast learning curve. Additionally, GEOTROPY-*loc* shows a slow increasing curve that finishes giving the third best result. In the case of *jac-catg* and *cos-catg*, the results are slightly better for *jac-catg*. This is due to the extreme popularity of certain categories that have a dominant effect on the cosine similarity measure and reduce its effectiveness. This is not the case when using Jaccard, as it only looks for shared categories, and ignores how many times each category was visited. For *jac-loc* and *cos-loc* the results are quite similar. Finally *g*-GEOTROPY-*loc* gets the worst result. This is due to the difficulty of elaborating

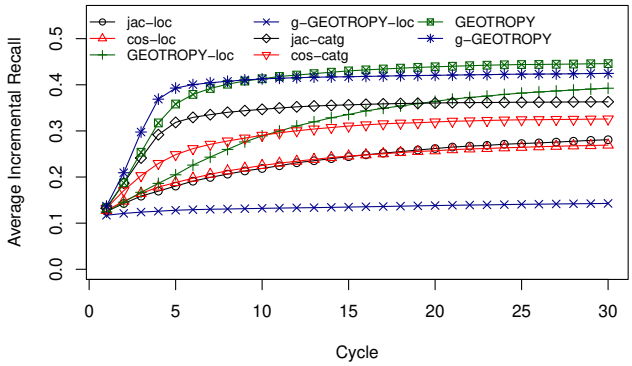

Figure 10. Average accumulative recall for configuration (80, 20, 20).

a global vision of the system based on locations from scratch. By contrast, the use of compaction reduces the number of variables to track to a small and well defined set, allowing to more rapidly build a good approximation of the system's state. For this reason, *g*-GEOTROPY performs better.

Quite interestingly. the metrics using compaction present a fast convergence, independently of the parameters employed. GEOTROPY and *g*-GEOTROPY reach a recall above 30% in the first 5 cycles. *g*-GEOTROPY stabilizes rapidly, while the recall of GEOTROPY keeps on increasing. By contrast, GEOTROPY-*loc* is the slowest approach, needing 10 cycles to obtain the same recall as methods with compaction do in 3 cycles. This fact indicates that the utilization of compaction improves the capacity of the users to select suitable neighborhoods.

C. Communication Overhead

This section evaluates the amount of information exchanged in the system for different metrics. The amount of data to be exchanged varies depending on the similarity metric. We identify four basic types of exchanged messages: $msg(L_i)$, $msg(\vec{M}_i)$, $msg(C_i)$ and $msg(\vec{C}_i)$. They are basically used to exchange locations, the occurrences of the locations, the vector of categories, and the occurrences of the categories.

Table V summarizes the size of the exchanged messages. Foursquare location identifiers are 12 bytes long. This means that exchanging only the locations is equivalent to send as many identifiers as locations are contained in L_i^{learn}. In the case of sending M_i, we need the identifiers of the locations and one byte per location representing the number of occurrences. We assume that all the occurrences can be represented with just one byte. Exchanging a vector of categories C_i only requires two bytes to represent the 355 possible categories. In order to exchange $\vec{C}_i$ we need to add a byte for every category with occurrences. Again, we assume that one byte is enough to represent the number of occurrences per category.

We assume that the peer do not use any kind of memory. Thus, they always request the data they need. We do not take

Message	Size	Method
$msg(L_i)$	$\|L_i^{eval}\| \times 12$	*jac-loc*
$msg(\vec{M}_i)$	$\|L_i^{eval}\| \times 12 + \|L_i^{eval}\|$	*cos-loc*, GEOTROPY-*loc*, *g*-GEOTROPY-*loc*
$msg(C_i)$	2	*jac-catg*
$msg(\vec{C}_i)$	$2 + \sum_{i \forall C_i \neq 0} 1$	*cos-catg*, GEOTROPY, *g*-GEOTROPY

Table V
FORMULAS EMPLOYED TO CALCULATE THE SIZE OF THE EXCHANGED MESSAGES.

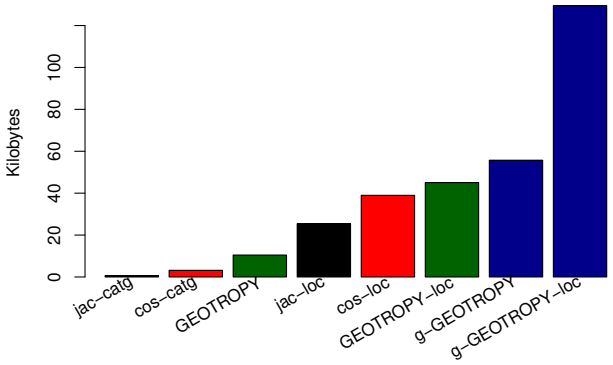

Figure 11. Average amount of data exchanged by peer per cycle for 10 neighbours and a (80%,20%) split

into account messages exchanged during the RPS gossiping nor the recommendation stage as these require the same amount of communication for all metrics.

Figure 11 shows the average number of kilobytes of exchanged data per peer and per cycle in the system for a configuration (80, 20, 10). This figure shows that the metrics that employ compaction exchange less information than the others. In particular the *jac-catg* method is the lightest. The metrics using global knowledge need additional information to be exchanged in order to build the global vision of the system. In this case, *g*-GEOTROPY generates much less traffic by using compaction.

D. Robustness

Our recommendation framework is based on the capacity of the sytem to find new peers with relevant information. This capacity depends on two factors: (i) the number of neighbors a peer can have, (ii) and the popularity of locations. Popular locations appear more often in the system, and are therefore easier to find.

Figure 12 compares the recall obtained in the last simulation cycle using different neighborhood sizes. A larger number of neighbors increases the recall for all the similarity variants. GEOTROPY and *g*-GEOTROPY remain the best two combinations in all cases. An increment in the number of neighbors is not proportional to an increase in the recall, although we

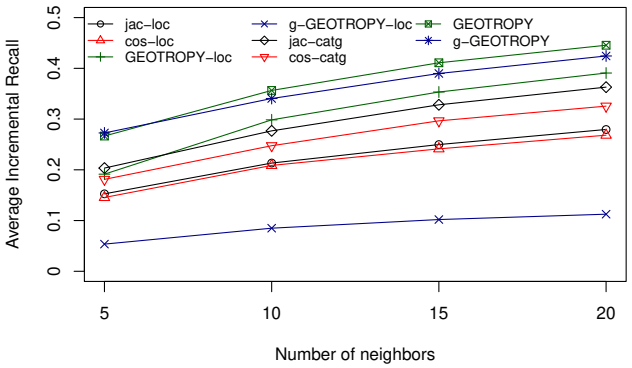

Figure 12. Recall obtained when varying neighborhood sizes (80%/20%)

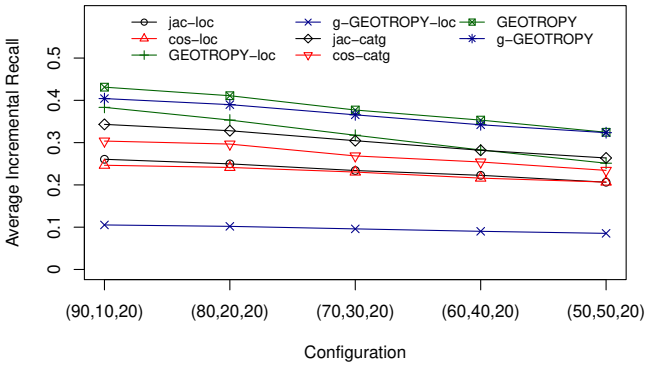

Figure 13. Recall obtained when varying the training set (20 neighbors)

multiply by four the number of neighbors, the average recall in the system only increases of 10%. This is due to the nature of the dataset and the probability of finding peers with similar locations. This shows that few neighbors are enough to provide reasonable results.

Figure 13 shows the recall obtained for different configurations using the same number of neighbors in the last simulation cycle. Reducing the size of the training set decreases the quality of the found neighborhoods and also the possibility of making good recommendations. The reduction of the size in the learning set, particularly affects the GEOTROPY and GEOTROPY-*loc* methods. However, the versions of these metrics based on global knowledge are not particularly affected. This occurs because the global knowledge provides external information, mitigating the reduction of information from the neighbors.

Another aspect to take into account is the capacity of the system to recommend locations according to their popularity. Figure 14 shows the percentage of locations in $\bigcup L_i^{eval}$ that were correctly recommended according to the number of their occurrences in $\bigcup L_i^{learn}$. The figure shows an increasing trend,

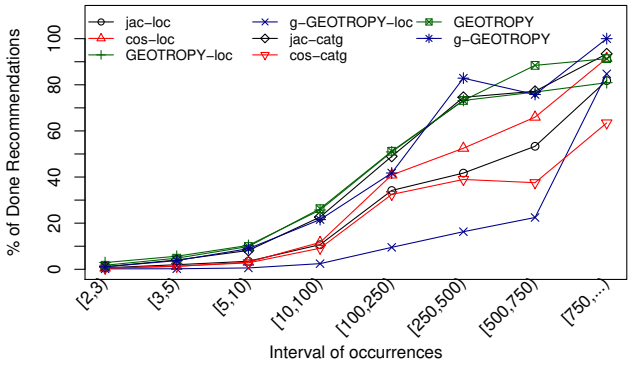

Figure 14. Percentage of recommended locations according to the interval of occurrences in $\bigcup L_i^{learn}$, for 20 neighbors and a (80%,20%) split

where popular locations are easily recommended. Using GEOTROPY it is possible to find more than 50% of the locations that appear between 100 and 250 times. The GEOTROPY method shows the best results for most of the intervals, even with the lowest numbers of occurrences.

E. Evaluation Summary

In summary our experiments show that:

- Compaction (the use of categories) is a very powerful mechanism to improve the performance of similarity metrics on fully distributed georecommendation systems. For all measures (cosine similarity, Jaccard, and Adamic/Adar), the category variants yield better results than the one directly using locations.
- The Adamic/Adar metrics can provide excellent recall results when combined with additional mechanisms (either compaction, or local sampling, or both). It is striking that the Adamic/Adar metrics provide both the two best variants in our framework (GEOTROPY and g-GEOTROPY), but also the *worst* variant, g-GEOTROPY-loc.
- The use of local sampling to approximate the frequency distributions required by the Adamic/Adar metrics does not incur any recall penalty, while considerably improving network costs. In fact the best similarity variant, GEOTROPY, uses local sampling. Local sampling also considerably improves network performance of the system, with GEOTROPY consuming 60% less bandwidth than its global variant g-GEOTROPY.
- There is a a trade-off between recall and communication overhead, but this trade-off is not as straightforward as it might be thought. The worst variant g-GEOTROPY-loc is also the most costly in terms of bandwidth. The less costly (*jac-catg*) is the fourth best variant out of 8, delivering a reasonable recall for a low network overhead.

All these findings also illustrate the benefits of modular frameworks in the design of fully-distributed georecommendation systems. That type of framework allows one to reason about the design space and potential available combinations in a principled and systematic way. They are also open to extensions, and show the way for more advanced adaptive distributed approaches to further improve results.

VI. Related Work

Several works have addressed the importance of using gossiping in fully decentralized networks. Eugster et al. [13] proposed a first approach for a gossip protocol that allows peers to have a partial vision of the entire network, showing its scalability capacities. In [9], Jelasity et al. proposed a framework to implement a RPS in a decentralized manner, demonstrating experimentally the uniformity of the partial views. In [14], authors propose a gossiping system to provide confidentiality in a decentralized system.

During the last years some works have focused on how to decentralize collaborative networks based on gossiping protocols. Bai et al. [15] addressed the problem of performing queries over a tagging system in a fully decentralized system. Their approach extracts users profiles through a gossiping protocol and looks for the most relevant items using a tag-based scoring system. This approach is based on the exploitation of users profiles and not on network proximity. In [3] Bertier et al. presented Gossple, a fully decentralized anonymous collaborative network. This network uses gossip protocol to discover new users, and an extended version of the cosine similarity to find users having relevant information. Finally, [16] proposed a fully distributed recommendation system for stream content that uses the content-rating of other neighbors to identify contents of interest.

There is a large body of research on recommendation systems. However, they are typically centralized. In [17], Han et al. proposed the decentralized implementation of a recommender system. Their approach consists on the distribution of users recommendations over a DHT. Kim et al. [18] introduced a decentralized recommendation system based on collaborative filtering. This system selects the neighborhood of a peer according to similarity metrics, and then chooses the k-top recommended items. Their evaluation, shows that they can recommend about 38% of the total set. Masa et al. [19] proposed a recommendation system for decentralized networks based on trustness. Users indicate the level of trust for a certain peer in such a way that only recommendations from trusted peers are taken into account.

The soaring popularity of location-based systems has recently raised the interest of researchers. Backstrom et al. [20] studied geolocation data from a social network and found that most friends are located close to each other. Furthermore, they proposed a centralized position-prediction system based on the position of a user's friends. Their experiments show that it is possible to predict 69% of users' positions. Ye et al. [21] proposed a centralized system to recommend tags for non-tagged locations. They leverage the past history and temporal patterns to extract knowledge in order to recommend tags. Works in [6] and [22] attempted to solve the problem of predicting the new links between users of a network with

geolocation data. Both works show the importance of using similarity measures that mitigate the effect of very popular items.

VII. Conclusions and Future Work

In this work we presented GEOLOGY, a modular georecommendation framework for fully-decentralized social networks. We demonstrated how combining compaction, the Adamic/Adar similarity metric and sampling allows a distributed peer-to-peer system to recommend geolocation data in a fully-decentralized manner. We evaluated our work using a dataset from Foursquare, a popular service where users can exchange information about geographical locations.

There are three main contributions of this paper. First, we proposed GEOLOGY, a modular framework to solve the problem of recommending geolocations in a fully-decentralized scenario. Our solution exploits the potential of gossiping protocols to define neighborhoods of users that can collaborate in order to find suitable recommendations. Second, we explored different similarity metrics to measure the semantic proximity between users. In particular, we proposed GEOTROPY, a similarity scoring technique that combines compaction, the Adamic/Adar similarity metric, and sampling. Our experiments show that GEOTROPY allows users to find on average 40% of the locations they are interested in 10 iterations, beating the results obtained by other metrics. Finally, we demonstrate that GEOTROPY reduces the communication overhead by employing the compaction module defined in our framework.

Several future research lines can be derived from this work. We will carry out a more extensive evaluation based on different datasets in order to better cover the spectrum of possible scenarios. For instance, it would be interesting to understand how our methods behave comparing different metropolitan areas, or countries. Furthermore, we plan to study the impact of definitions of categories on decentralized georecommendation systems.

Acknowledgment

Part of this work was supported by ERC Starting Grant GOSSPLE number 204742, the Spanish Ministry of Education under FPU program AP2007-03530 (Juan M. Tirado), and the Spanish Ministry of Science and Technology under the grant TIN2010-16497 "Técnicas escalables de entrada/salida en entornos distribuidos y de computación de altas prestaciones".

References

[1] Foursquare, "Foursquare announces 10M users. Available Online: http://blog.foursquare.com/2011/06/20/holysmokes10millionpeople/," 2011.

[2] V. Leroy, B. B. Cambazoglu, and F. Bonchi, "Cold start link prediction," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '10. New York, NY, USA: ACM, 2010, pp. 393–402. [Online]. Available: http://doi.acm.org/10.1145/1835804.1835855

[3] M. Bertier, D. Frey, R. Guerraoui, A.-M. Kermarrec, and V. Leroy, "The gossple anonymous social network," in *Proceedings of the ACM/IFIP/USENIX 11th International Conference on Middleware*, ser. Middleware '10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 191–211. [Online]. Available: http://dl.acm.org/citation.cfm?id=2023718.2023732

[4] J. M. Tirado, D. Higuero, F. Isaila, J. Carretero, and A. Iamnitchi, "Affinity p2p: A self-organizing content-based locality-aware collaborative peer-to-peer network," *Comput. Netw.*, vol. 54, pp. 2056–2070, August 2010.

[5] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '08. New York, NY, USA: ACM, 2008, pp. 426–434. [Online]. Available: http://doi.acm.org/10.1145/1401890.1401944

[6] S. Scellato, A. Noulas, and C. Mascolo, "Exploiting place features in link prediction on location-based social networks," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '11. New York, NY, USA: ACM, 2011, pp. 1046–1054. [Online]. Available: http://doi.acm.org/10.1145/2020408.2020575

[7] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*. New York, NY, USA: McGraw-Hill, Inc., 1986.

[8] L. A. Adamic and E. Adar, "Friends and neighbors on the web," *Social Networks*, vol. 25, no. 3, pp. 211 – 230, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0378873303000091

[9] M. Jelasity, S. Voulgaris, R. Guerraoui, A.-M. Kermarrec, and M. van Steen, "Gossip-based peer sampling," *ACM Trans. Comput. Syst.*, vol. 25, August 2007. [Online]. Available: http://doi.acm.org/10.1145/1275517.1275520

[10] M. Jelasity and O. Babaoglu, "T-man: Gossip-based overlay topology management," in *Proc. of the 3rd Int. Workshop on Engineering Self-Organising Applications*, 2005, pp. 1–15.

[11] S. Voulgaris and M. v. Steen, "Epidemic-style management of semantic overlays for content-based searching," in *Euro-Par'05*.

[12] Foursquare, "Foursquare API. Available Online: https://developer.foursquare.com/docs/," 2011.

[13] P. T. Eugster, R. Guerraoui, S. B. Handurukande, P. Kouznetsov, and A.-M. Kermarrec, "Lightweight probabilistic broadcast," *ACM Trans. Comput. Syst.*, vol. 21, no. 4, pp. 341–374, 2003.

[14] V. Schiavoni, E. Riviere, and P. Felber, "Whisper: Middleware for confidential communication in large-scale networks," in *Distributed Computing Systems (ICDCS), 2011 31st International Conference on*, june 2011, pp. 456–466.

[15] X. B. et al, "Gossiping personalized queries," in *EDBT'10*, pp. 87–98.

[16] S. Isaacman, S. Ioannidis, A. Chaintreau, and M. Martonosi, "Distributed rating prediction in user generated content streams," in *Proceedings of the fifth ACM conference on Recommender systems*, ser. RecSys '11. New York, NY, USA: ACM, 2011, pp. 69–76. [Online]. Available: http://doi.acm.org/10.1145/2043932.2043948

[17] P. Han, B. Xie, F. Yang, and R. Shen, "A scalable p2p recommender system based on distributed collaborative filtering," *Expert Systems with Applications*, vol. 27, no. 2, pp. 203 – 210, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417404000065

[18] J. K. Kim, H. K. Kim, and Y. H. Cho, "A user-oriented contents recommendation system in peer-to-peer architecture," *Expert Systems with Applications*, vol. 34, no. 1, pp. 300 – 312, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S095741740600279X

[19] P. Massa and P. Avesani, "Trust-aware recommender systems," in *Proceedings of the 2007 ACM conference on Recommender systems*, ser. RecSys '07. New York, NY, USA: ACM, 2007, pp. 17–24. [Online]. Available: http://doi.acm.org/10.1145/1297231.1297235

[20] L. Backstrom, E. Sun, and C. Marlow, "Find me if you can: improving geographical prediction with social and spatial proximity," in *Proceedings of the 19th international conference on World wide web*, ser. WWW '10. New York, NY, USA: ACM, 2010, pp. 61–70. [Online]. Available: http://doi.acm.org/10.1145/1772690.1772698

[21] M. Ye, D. Shou, W.-C. Lee, P. Yin, and K. Janowicz, "On the semantic annotation of places in location-based social networks," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '11. New York, NY, USA: ACM, 2011, pp. 520–528. [Online]. Available: http://doi.acm.org/10.1145/2020408.2020491

[22] J. Cranshaw, E. Toch, J. Hong, A. Kittur, and N. Sadeh, "Bridging the gap between physical location and online social networks," in *Proceedings of the 12th ACM international conference on Ubiquitous computing*, ser. Ubicomp '10. New York, NY, USA: ACM, 2010, pp. 119–128. [Online]. Available: http://doi.acm.org/10.1145/1864349.1864380