A Middleware Service for Pervasive Social Networking

Sonia Ben Mokhtar
Dept. of Computer Science
University College London
London WC1E 6BT, UK
s.benmokhtar@cs.ucl.ac.uk

Liam McNamara
Dept. of Computer Science
University College London
London WC1E 6BT, UK
l.mcnamara@cs.ucl.ac.uk

Licia Capra
Dept. of Computer Science
University College London
London WC1E 6BT, UK
l.capra@cs.ucl.ac.uk

ABSTRACT

Today's online social networking applications (e.g., Facebook, Twitter, Last.Fm) allow users that are *socially* close to each other (i.e., users with shared interests) to participate in the collective production and sharing of information (i.e., *virtual* interactions). *Pervasive Social Networking* (PSN) is a new vision that aims to complement virtual interactions with *physical* ones, by enabling users who are both *socially* and *physically* related to find each other and perform activities of common interest. To enable this vision, both users' social networks *and* mobility patterns must be reasoned upon. In this paper, we present a social networking middleware service that dynamically combines both social and physical proximity relations between mobile users to accurately recommend them people with whom to perform activities of common interest. At the heart of this service is a social network propagation component that infers users' relations both within the same (intra) and across (inter) users' activities. We evaluate the impact of various middleware deployment strategies on the ability of the social network propagation component to find related users, and analyse the advantages and shortcomings of each of them.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*Distributed Applications*

General Terms

Algorithms

Keywords

Social Networks, Pervasive Computing, Middleware

1. INTRODUCTION

Pervasive Social Networking (PSN) [1] (also called Mobile Social Networking) is a novel computing paradigm deriving from the convergence of Pervasive Computing and Web 2.0 social networking services (e.g., Facebook, MySpace, Twitter). While in the traditional Pervasive Computing vision [13] mobile users seamlessly discover and interact with services offered by their surrounding physical environment (e.g., display screens, multimedia servers), in PSN users may further discover people, among their direct (e.g., friends) or indirect (e.g., friends of friends) social relations, who share similar interests (e.g., playing tennis, going to the theatre). Unlike Web 2.0 services that connect people who are related, by exploring their social relations, PSN focuses on the discovery of people who are both *socially* and, crucially, *physically* proximate.

In a typical PSN application, users would describe their interests in performing *activities* along with their *social preferences*, indicating with whom, among their direct social relations, they enjoy performing such activities. They would pass this information to the PSN application hosted on their mobile device (e.g., iPhone, Blackberry, Android-powered mobile phones), and expect in return accurate recommendations about users with similar interests, to whom they are directly or indirectly connected in the social network, and who are physically proximate. Examples of PSN applications are Google Latitude[1], iPhone BreadCrumbs[2], and Here I Am[3]. At present, these applications work under the assumptions that: (1) mobile users have continuous access to online services from their mobile devices, to where they upload and share their location (e.g., GPS coordinates); (2) the application has complete knowledge of the users' social networks, so that propagation algorithms can accurately compute friends-of-friends relations to guess missing links in the social graph (similar to the Facebook 'People You May Know' service).

We believe these assumptions to be limitative with respect to the PSN vision. For example, location information may be very imprecise, if not completely unavailable, especially in metropolitan cities, so that person-to-person co-location cannot be inferred from GPS coordinates. Also, the scalability of current solutions is dubious, if the number of people who used these PSN applications were to increase up to the number of Web 2.0 users. Finally, privacy-conscious users may be reluctant to disclose both their social network and their location to a single central service. To enable the full PSN vision, we argue that current solutions should be complemented with semi-distributed ones, to increase avail-

[1]http://www.google.co.uk/latitude/
[2]http://www.apptism.com/apps/breadcrumbs/
[3]http://www.android.com/market/#app=hereiam

ability, scalability, and to provide users with better control of their own information, by means of privacy-aware semi-distributed discovery protocols as those presented in [3]. The effectiveness of PSN applications will then rely on the ability of the semi-distributed system to dynamically gather information about users' social preferences and co-location information, so to be able to compute accurate recommendations from partial social network knowledge, while also enforcing users' privacy.

In this paper, we present a social computing middleware service that enables the semi-distributed PSN vision. Each mobile device logs regular encounters with other mobile devices, detected by means of Bluetooth radio connectivity, as shown in [8]. Our middleware, hosted on a set of nodes called *brokers*, dynamically gathers these logs, together with users' interests and associated social networks, during periods of colocation. Brokers exchange the collected information between each other, and employ social network propagation algorithms to infer missing social links. Brokers finally compute a similarity measure, combining both social and physical proximity information between users, and use that measure to provide accurate recommendations to users. As for privacy, we do not investigate it further in this paper, and refer the reader to solutions for privacy enforcement in distributed settings surveyed in [2].

The remainder of the paper is structured as follows: in Section 2 we review the state-of-the-art in social network propagation algorithms. Section 3 presents our middleware architecture and social networking model, within which such algorithms are being deployed. We illustrate different middleware deployment strategies, and evaluate their impact on both social network propagation and incurred overhead in Section 4. Finally, Section 5 concludes the paper and presents our future directions of research.

2. BACKGROUND

Similar to Web 2.0 applications, social network propagation is at the heart of the PSN vision, as it enables people who regularly encounter each other to discover new social relations they may enjoy. Social network propagation can be performed either within a single activity network (*intra*-activity), or across networks, each dealing with a different activity (*inter*-activity).

2.1 Intra-Activity Social Network Propagation

Imagine having a social network where nodes are tennis players and edges represent the "enjoy-playing-tennis-with" relation. Intra-activity social network propagation tries to answer the question: if A enjoys playing tennis with B, and B with C, what can we say about A enjoying playing tennis with C? A very similar problem has been investigated by the trust research community [7], and a variety of algorithms have been proposed to accurately infer trust relationships within a given web-of-trust (WoT). For example, [6] proposes the computation of the maximal flow function from a source node A to a sink node C in the WoT, thus guaranteeing that the inferred trust for C does not exceed the weight of any edge in the path from A to C. In [4], the authors propose an adaptation of the PageRank algorithm [9], called Personalised PageRank (PPR), whereby a random walk is computed over a graph of users, rather than over a graph of web pages, with the walk starting at the source node A, thus obtaining a personalised walk (i.e., trust propagation from A's viewpoint). In [10], a lightweight graph-based semi-supervised learning technique is described, which propagates trust by first quantifying how similar relations are; relations are deemed similar if they either have the same source (e.g., $A \rightarrow X$ and $A \rightarrow Y$) or the same sink (e.g., $X \rightarrow C$ and $Y \rightarrow C$). These algorithms have been developed with different goals in mind: [6] provides resilience against Sybil attacks from malicious users, PPR aims to add subjectivity to the propagation process, while [10] aims to minimise computational costs.

2.2 Inter-Activity Social Network Propagation

Social relations are very activity-dependent. For example, while there might be an edge from A to B in the tennis player network described above, there may not be one in a badminton player network. However, some activities have strong similarities with each other, so that a question arises as to whether one can propagate social relations across different domains. A similar problem has started to be investigated once again within the trust research community. In [11], for example, the authors propose an algorithm called TRULLO that exploits singular value decomposition to automatically infer activity similarity, consequently performing trust propagation across similar activities.

The accuracy of all the above algorithms has been evaluated in web-based settings, where the social networks are centralised and fully known. However, in our PSN vision, only a partial view of each social network is known at any given time, thus raising the question of whether such techniques can indeed successfully propagate under partial knowledge, and to what extent. In the reminder of the paper, we focus our attention on intra-activity propagation using the Personalised PageRank algorithm. In particular, we adopt this algorithm within the middleware service we present in the next section, and investigate the impact that different architectural deployments have on its propagation power. It is our plan to evaluate the other techniques as well in our future work, to assess what intra- and inter- propagation algorithms to embed in our PSN middleware.

3. SOCIAL NETWORKING MIDDLEWARE

In this section, we first present the social networking and physical proximity models we use in order to quantify how related two users are in the social network and in the physical proximity network separately (Section 3.1). We then illustrate how our PSN middleware service (Section 3.2) combines these values into a users' similarity measure that will be used to recommend people to each other. Finally, we discuss various deployment options of our middleware service (Section 3.3), before turning to evaluating them in the next section.

3.1 Social and Physical Proximity Models

As presented in [1], a pervasive social network can be represented with a quaternary relation that allows the definition of activity-specific social links. The relation *(Person1, Person2, Activity, PreferenceValue)* specifies that *Person1* likes performing the activity *Activity* with *Person2* with a social

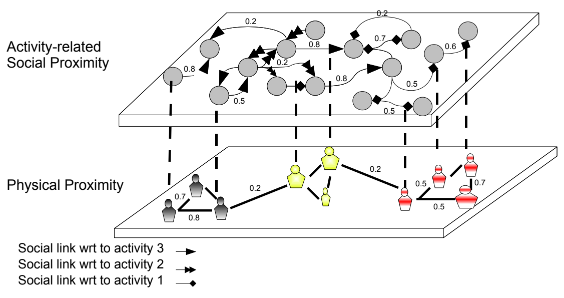
Activity-related Social Proximity
Physical Proximity
Social link wrt to activity 3
Social link wrt to activity 2
Social link wrt to activity 1
0.2
0.8
0.7
0.6
0.5
0.7
0.8
0.2
0.5
0.5
0.7

Figure 1: Social and Physical Proximity Networks

preference value *PreferenceValue* (which can be assumed to vary in the range $[0, 1]$). This allows the specification of a multi-activity social network as exemplified in Figure 1. In this figure, the overall social network is depicted in the upper layer: nodes represent users, and directed edges exist between users who have enjoyed performing an activity together. More precisely, each edge is annotated with two pieces of information: the activity that is of interest to the two users, and a preference value specifying how much the user from which the edge departs 'enjoys' performing the related activity with the sink user. Users are expected to declare their direct social preferences, as commonly done in Web 2.0 social networking services; on top of these explicitly made connections, further social links are inferred using the propagation algorithms described in Section 2.

In addition to the multi-activity social network, each user's device keeps track of the other users she regularly encounters in order to build its physical proximity network (lower layer in Figure 1). Specifically, each node monitors, throughout its lifetime, its contacts with other nodes in the network, in a way similar to that described by the Habit protocol [8]. Encounters are first simply logged ("node A encounters node B at time t"); if encounters with B become frequent, then B becomes a familiar stranger to A and it is added to the list of the k most regular nodes with A (k is a parameter of our middleware service, to be tuned based on how much resources we want to devote to tracking familiar strangers). A physical proximity value is further associated to each node's familiar stranger based on how *regularly* those nodes encounter each other. More precisely, based on information about when A encounters her familiar strangers (e.g., day of the week/time of the day encounters occur), A can compute a regularity weight, that is, the number of times A has met B in a given *regularity interval* (e.g., Monday 10-11am), over a certain *observation period* (e.g., last four weeks). The *physical proximity value* from A to B is then computed as the number of time slots for which A has a regularity weight greater than a threshold value, divided by the number of timeslots available in a week (e.g., 168 if hourly timeslots are considered). For instance, if A has recorded the following regularity values for B: Monday 10am-11am: 0.75; Tuesday 4pm-5pm: 0.6 and Thursday 11am-12am: 0.45, the physical proximity value between A and B for the incoming week will be 2/168, assuming a regularity threshold of 0.5.

3.2 Middleware Service

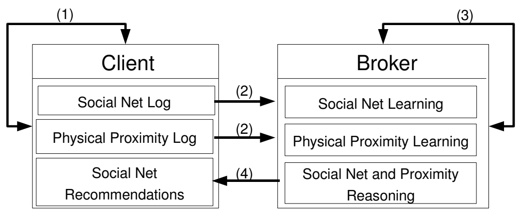
(1)
(3)
Client
Broker
Social Net Log
Physical Proximity Log
Social Net Recommendations
Social Net Learning
Physical Proximity Learning
Social Net and Proximity Reasoning
(2)
(2)
(4)

Figure 2: Middleware Components and Interactions

The architecture of our proposed Social Networking Middleware Service is depicted in Figure 2. As shown, our middleware logically distinguishes between two types of nodes, i.e., *Brokers* and *Clients*. We assume there exists one client per user (i.e., its personal mobile device). Clients maintain updated information about what activities their users have performed, with whom, and with what social preference ('Social Net Log' functionality). We assume such feedback is entered by the user via the PSN application interface running on her personal device. Throughout their lifetime, clients also maintain updated logs of their top k familiar strangers ('Physical Proximity Log' functionality and interaction (1) in Figure 2), as described in Section 3.1. On the other hand, brokers are the nodes responsible for gathering (and propagating) the social network and proximity information, in order to compute recommendations for the mobile users they serve. Specifically, upon encountering a broker, clients offload their social preferences and physical proximity information (interactions (2)). This information is collected by the 'Social Net Learning' and 'Physical Proximity Learning' functionalities respectively. When two brokers meet, they further exchange this information (interaction (3)) and locally propagate the social links using the newly acquired knowledge. Finally, brokers reason over both the social and physical proximity networks to compute recommendations for clients ('Social Net and Proximity Reasoning' functionality) and further notify them upon encounter (interaction (4) in the figure).

Recommendations are computed and ranked using the following utility function:

$$\begin{aligned} Utility(A, B, act) &= \alpha * SocialProximity(A, B, act) + \\ &\quad \beta * PhysicalProximity(A, B) \end{aligned}$$

where $SocialProximity(A, B, act)$ corresponds to the weight on the direct or propagated social link between A and B with respect to the activity act, $PhysicalProximity(A, B)$ is the physical proximity value computed as described in Section 3.1, and α and β are two parameters that can be adjusted to prioritise the social proximity and the physical proximity differently, as required by specific PSN applications.

3.3 Middleware Deployment

Similar to Web-based social networking services, the effectiveness of PSN applications will depend on the number of users they reach, and consequently on the connectivity of the social network they build. The architectural deployment of PSN middleware thus plays a crucial role. In this paper, we investigate three deployment strategies for our middleware:

Fully Centralised Deployment. In the fully centralised deployment scenario, clients have a near-permanent connection to an infrastructure. A centralised server, acting as a broker, stores clients' social preferences and physical proximity records, and performs social network propagation on them. Each time a client wants to perform a specific activity, she connects to the infrastructure (as done by current PSN applications) and gets a set of fresh recommendations about other clients that share similar interests to her, among her physically and socially related users.

Semi-Distributed, Mobile Overlay. In this deployment scenario, a set of mobile nodes are dynamically elected to act as brokers. As described in Section 1, this scenario is a suitable complement/alternative to the fully centralised one, in order to increase availability and scalability. Furthermore, privacy-enforcement mechanisms [3] can be more easily applied in semi-distributed settings. Brokers can be elected based on different criteria: for example, network coverage, mobility pattern, available resources, etc. In this paper, we use a simple election process based on node popularity (i.e., the devices that meet the highest number of nodes within a time period are elected to act as brokers); more advanced and dynamic election techniques, such as [12] augmented with reasoning on node's trust properties, will be investigated in the future. Crucially, as brokers are mobile, we cannot assume the existence of a permanent connection to/among them; rather, clients communicate their social and proximity network information to brokers during periods of colocation, and brokers exchange such information when in reach of each other. Brokers perform social network propagation based on the partial view of the network they have collected, and notify clients of their computed recommendations when encountered.

Fully Distributed Deployment. In this scenario, all the nodes in the network act as brokers. Clients share their social preferences and physical proximity logs with all nodes they encounter (as they are themselves brokers), and locally compute propagations based on the information they have gathered.

4. PERFORMANCE EVALUATION

In this section, we present an evaluation of our middleware service conducted by means of simulation. We start by describing the simulation setup (Section 4.1), followed by three experiments evaluating the effect of the broker election and deployment strategies on the social network propagation algorithm (Section 4.2 and 4.3, respectively) and the overhead due to the exchange of social network and physical proximity information between brokers (Section 4.4).

4.1 Simulation Setup

In order to evaluate our social networking middleware service, we needed a dataset comprising two pieces of information: (1) human mobility traces, and (2) a weighted users' social network. The MIT Reality Mining project's data [5] has the unique characteristic of offering *both* these aspects at once; we thus focus on these traces, as described below.

Mobility Traces. In terms of human movement, the MIT traces contain colocation information from 100 subjects (staff and students) at the MIT campus over the course of the 2004-2005 academic year, to whom Bluetooth-enabled Nokia 6600 phones were given; colocation information was collected via frequent (every 5 minute) Bluetooth device discoveries. To make the dataset more manageable, we have extracted three months of colocation data, corresponding to September-October-November 2004.

Social Network. Beside providing a unique dataset in terms of length and breadth of mobility traces, the MIT dataset also implicitly includes information about the users' social network. In fact, it logs both the text messages sent, and the phone calls made by each phone in the study. Using this information, we have extracted a social network whereby a link from user A to user B is created if A sent a text message or made a phone call to B; these links are also weighted, depending on the intensity of the activity between the two users. This implicit social network extraction allowed us to tie real social behaviour with the actual users movement. We have further focused our evaluation on one social network propagation algorithm, i.e., the Personalised Page Rank (PPR) algorithm [4]. The evaluation of the effect of the architectural deployment on other social network propagation algorithms will be studied in future work.

Rather than assuming that users are all equally interested in propagating their social relations across colocated devices, we wanted to model the fact that users have different social behaviours, with some being aggressive social networkers, while others being less so. We have thus assigned clients with a probability of exchanging their social network and physical proximity information with brokers, derived from the MIT real phone call traces. In so doing, we reflect differences across users, as well as across timeslots (e.g., day/night, week-days/week-ends). During the three months of experiment, more than 9000 phone calls were issued. In all our experiments, interests are assumed to be related to a single activity.

Metrics. In evaluating the effect of our middleware deployment strategies on the social network propagation, we have been interested in measuring: *(1) Degree of Propagation* (DoP), that is, the number of missing links at a specific point in time (called *checkpoints* hereafter), with respect to the maximum number of links that can be learnt by a centralised server when running the propagation algorithm on the whole social network. The DoP is averaged among nodes acting as brokers. Over time this metric shows how fast brokers can distributively learn and propagate over the users' social networks, and how fast they can adapt to social network updates; *(2) Overhead*, measured as the communication overhead incurred by our middleware service in terms of the traffic generated by the exchange of the users' social preferences and regularity logs between brokers.

4.2 Effect of the Election Strategy on the DoP

The first experiment aimed at comparing different broker election strategies, namely, 'popular' broker election and

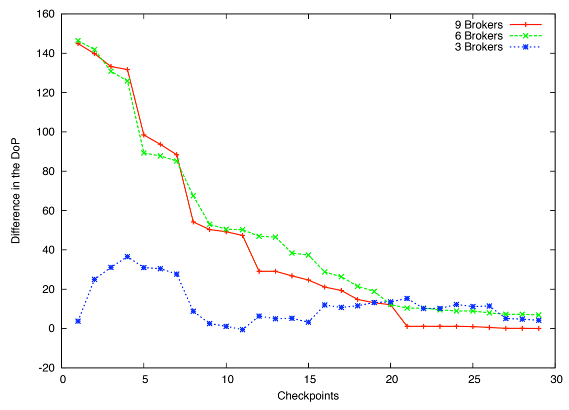

Figure 3: Difference in the DoP between Popular and Random Brokers

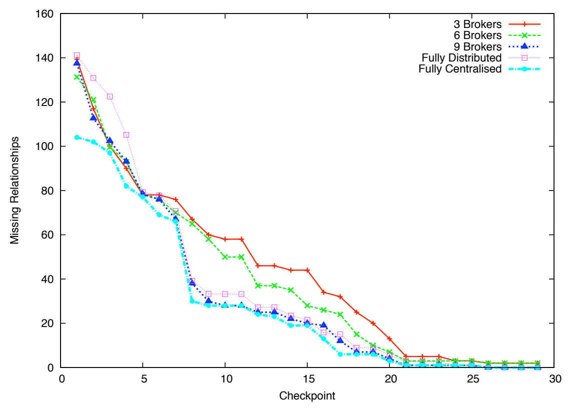

Figure 4: Effect of the Deployment Strategy on the Propagation

'random' broker election, in the semi-distributed deployment scenario, in order to quantify their impact onto the degree of propagation brokers are capable of. In order to compare the two strategies, we have measured the *difference* between the averaged DoP performed by 3, 6 and 9 popular brokers and the averaged DoP performed by 3, 6 and 9 random brokers. Propagations were computed every three days during the three months experiment. For each configuration of the random brokers (i.e., 3, 6 and 9 brokers), we have performed 3 runs where different brokers have been selected. The results correspond to the mean of those 3 runs. Results, depicted in Figure 3, show that the 3 random brokers are performing similarly to the 3 popular brokers as the difference line fluctuates around zero. This indicates that more brokers might be required to perform effective social network propagation. Instead, there is a clear distinction between the 6 and 9 popular brokers and the 6 and 9 random brokers (respectively) as difference lines become close to zero around checkpoint 20 (i.e., at the end of the second month). This indicates that with 6 and 9 brokers popular brokers, social network propagation and dissemination can be performed much faster than with random brokers as the former meet more often to exchange their knowledge on the social network.

4.3 Effect of the Deployment Strategy on the DoP

The second experiment aimed at comparing different broker deployment strategies, namely, 'fully-centralised', 'semi-distributed' and 'distributed', in order to quantify their impact onto the degree of propagation brokers are capable of. For the semi-distributed deployment scenario, we have considered three different configurations, i.e., with 3, 6 and 9 popular brokers. The standard deviation for each measure of the DoP is also measured.

Results, depicted in Figure 4, show that increasing the number of brokers increases the DoP over time, i.e., new social links can be discovered faster and thus new social network updates are expected to be integrated faster as well. The number of brokers to deploy, however depends on the network configuration and nodes' mobility patterns.

In the context of the MIT mobility traces, 9 popular brokers are sufficient to yield the best results[4](as it is indeed already comparable to the centralised deployment). Note that the fully distributed approach has similar results to the semi-distributed one; however, its overhead is much higher, as shown in the next experiment.

4.4 Overhead due to the SN Propagation

Finally, we have measured the communication overhead, in terms of the traffic generated between brokers when exchanging clients' social preferences and regularity logs. For estimating the overhead due to the exchange of regularity logs we have considered that each node sends his top 10 proximity links (i.e., k=10). For the social logs instead, we have counted the real mount of data exchanged between brokers (no limit in the number of social links). As expected, the results, summarised in Table 1, show that overhead increases with the number of nodes acting as brokers. However, as the DoP stabilises with a fairly small number of appropriately chosen brokers (as demonstrated in the previous experiment), the semi-distributed middleware deployment does represent a valuable alternative to centralised approaches. Note that the estimated overhead due to the exchange of proximity logs is about four times higher than the overhead due to the exchange of social logs (e.g., for 9 brokers, 22KB of proximity logs, and 6KB of social logs are exchanged by each of the 9 brokers on average per day). A number of solutions could be investigated to reduce that overhead (e.g., compressing data, sending updates of the logs instead of the whole logs). Finally, the overhead of exchanging proximity logs could be avoided if clients perform the proximity-based filtering of recommendations locally. However, this may generate another type of overhead, i.e., computational overhead on the client side, in addition to extra-traffic due to the sending of a higher number of recommendations.

4.5 Discussion

From the previous experimental results we have learnt that semi-distributed mobile overlays constitute a good compro-

[4]We have also performed the experiment with 12 popular brokers and results were similar to those of 9 brokers.

	3 Brok.	6 Brok.	9 Brok.	Fully Dist.
Social Logs	1 MB	5 MB	11 MB	191 MB
Prox. Logs	3.5 MB	19.8 MB	45.8MB	753 MB

Table 1: Communication Overhead

mise between the degree of propagation brokers can perform over time and the overhead they incur. Furthermore, a small number of brokers (which depends on the network configuration and nodes' mobility patterns) is enough for performing dynamic social network propagation. Finally, appropriately choosing brokers is crucial for the realisation of the PSN vision. In this context, nodes' popularity seems to be a good criterion that might be complemented with nodes' trust properties in order to increase the confidence of clients in disclosing their social and physical proximity information.

5. CONCLUSIONS AND FUTURE WORK

In the last few years, we have witnessed two major phenomena: the widespread adoption of networked handheld devices, and the vast popularity of social networking Web sites. We believe it will not be long before these two phenomena will converge toward the so-called Pervasive Social Networking (PSN) vision, thus enabling users to realise their social activities with their co-located friends and relations anywhere and anytime. Contrary to online social networking applications, where users' social networks are centralised and highly available for those applications to reason upon, in PSN the social network is inherently distributed, as each user holds such information on their personal device. Furthermore, social links will be more and more related to specific activities, which makes reasoning over them even more complex.

In this paper, we have presented a middleware service that helps realising the PSN vision by enabling the dynamic learning and propagation of distributed social networks. Our middleware service logically distinguishes between clients (i.e., nodes that have activity-related social preferences and request recommendations of other nodes who are both physically and socially related to them), and brokers (i.e., nodes that gather those social preferences, carry out social network propagation, and provide recommendations to users).

Thus far, we have focused on one intra-activity propagation algorithm (i.e., PPR) and evaluated the effect of various deployment strategies on its ability to opportunistically learn new links between clients from existing ones. Our evaluation, performed using real human mobility traces and social networks, shows that semi-distributed mobile overlays are capable of performing social network propagations almost as accurate as centralised approaches, while entailing only a small overhead.

Our ongoing work focuses on the integration and evaluation of different intra- and inter- activity propagation algorithms within our middleware service, to assess which suits the PSN scenario best. Our future work will investigate two main directions: the integration of privacy-preserving techniques, as presented in the context of service discovery [3], within our semi-distributed middleware deployment, and trust-aware broker election mechanisms, thus electing those nodes who are not only the most popular, but also those having the highest reputation in the network.

6. REFERENCES

[1] S. Ben Mokhtar and L. Capra. From Pervasive To Social Computing: Algorithms and Deployments. In *ACM International Conference on Pervasive Services*, 2009.

[2] R. Speicys Cardoso and V. Issarny. Architecting pervasive computing systems for privacy : A survey. In *Proc. of the Working IEEE/IFIP Conference on Software Architecture (WICSA)*, 2007.

[3] R. Speicys Cardoso, P-G. Raverdy, and V. Issarny. A Privacy-Aware Service Discovery Middleware for Pervasive Environments. In *Proc. of Joint iTrust and PST Conferences on Privacy, Trust Management and Security*, 2007.

[4] M. Dell'Amico and L. Capra. SOFIA: Social Filtering for Robust Recommendations. In *Proc. of Joint iTrust and PST Conferences on Privacy, Trust management and Security*, 2008.

[5] N. Eagle and A. Pentland. CRAWDAD Dataset MIT/Reality (v. 2005-07-01). http://crawdad.cs.dartmouth.edu/mit/reality.

[6] M. Feldman, K. Lai, I. Stoica, and J. Chuang. Robust incentive techniques for peer-to-peer networks. In *Proc. of the 5th ACM Conference on Electronic Commerce*, 2004.

[7] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proc. of the 13th ACM International Conference on World Wide Web*, 2004.

[8] A. J. Mashhadi, S. Ben Mokhtar, and L. Capra. Habit: Leveraging Human Mobility and Social Network for Efficient Content Dissemination in MANETs. In *Proc. of 10th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, 2009.

[9] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford University, 1999.

[10] D. Quercia, S. Hailes, and L. Capra. Lightweight Distributed Trust Propagation. In *Proc. of the 7th IEEE International Conference on Data Mining*, 2007.

[11] D. Quercia, S. Hailes, and L. Capra. Trullo - local trust bootstrapping for ubiquitous devices. In *Proc. of 4th IEEE International Conference on Mobile and Ubiquitous Systems: Networking&Services*, 2007.

[12] F. Sailhan and V. Issarny. Scalable Service Discovery for MANET. In *Proc. of the 3rd IEEE International Conference on Pervasive Computing and Communications (PerCom)* 2005.

[13] M. Weiser. The Computer for the 21st Century. *ACM SIGMOBILE Mobile Computing and Communications Review*, 1999.