A Self-Organising Directory and Matching Service for Opportunistic Social Networking

Sonia Ben Mokhtar
LIRIS-CNRS
7, Batiment. Blaise Pascal
69100 Villeurbanne
France

Afra J. Mashhadi and Licia Capra
Dep. of Comp. Science,
University College London
Gower Street
London WC1E 6BT, UK

Liam McNamara
Computer Laboratory,University of Cambridge
15 JJ Thomson Avenue
Cambridge CB3 0FD, UK

ABSTRACT

Web-based social networking services enable people who share interests to find each other and collaborate in online and offline social activities. Thanks to the widespread popularity of networked handheld devices, constantly carried by members of the public throughout their daily activities, new possibilities of collaboration and interaction are emerging, among people who are not only *socially* close to each other, but also in *physical* proximity. However, a challenge arises as to how to enable people with similar interests to find each other, so to fulfill their social activities anywhere and anytime, while constantly moving around. In this paper, we present ADESSO, a semi-distributed directory and matching service that supports *opportunistic social networking* in delay tolerant networks. ADESSO consists of a set of *self-organising brokers*, automatically elected based on their mobility patterns. Users offload their requests to perform social activities onto brokers upon encounters; brokers then collaborate, by means of either request exchanges or broker fusion, in order to match activities in a way that satisfies users' social preferences. Preliminary performance evaluation, conducted using real human mobility traces and social networks, shows that ADESSO generates matches that highly satisfy users' preferences, entailing only a small overhead.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; C.2.4 [**Distributed Systems**]: Distributed Applications

General Terms

Algorithms

Keywords

Mobile Social Networking, Human DTNs

1. INTRODUCTION

SNS'10, April 13, 2010, Paris, France.

Web-based social networking services (e.g., MySpace, Last.fm, Facebook, Flickr, Twitter, etc.) enable people who share interests (e.g., music tastes or hobbies) to come together in global online communities, and participate in the collective production and sharing of information (e.g., pictures, Twitter updates, music profiles, etc.). They can also be used to arrange real world activities, as different as the recently organised parties[1] and demonstrations during the 2009 civil unrest in Iran[2].

Thanks to the widespread popularity of handheld networked devices (e.g., iPhone, gPhone and Blackberry), constantly carried by members of the public throughout their daily activities, new possibilities of social interactions are emerging, among people who are not only *socially* close to each other, but who also have a connection through *physical proximity*. Consider, for example, employees wishing to arrange out-of-work sport activities, immigrants living in the same neighbourhood willing to meet to help each other, or single people willing to organise a date with someone who often visits the same places (and is not simply living in the same metropolitan city). Community building and localised dating services are examples of *opportunistic social networking*, whereby both social and physical proximity are exploited to provide new forms of situated social interactions [5].

In this paper, we are interested in providing support for these kinds of interactions in human delay tolerant networks (also called pocket switched networks [13]), that is, enabling social activities between people who share interests and who have been physically close to each other (either directly or indirectly through third parties). To realise this goal, a user must first specify her interest in performing an '*activity*', along with her '*social preferences*' (i.e., some qualities of the users with whom she would like to perform such an activity), as part of what we call a '*user task*'. Such tasks must also define a time-to-live (TTL), before which the user expects to receive one or more recommendations about whom they can perform the activity with. While existing related research efforts in human DTNs have been designed to *spread* information in the network (e.g., [16, 19]), in our scenario the goal is to collate tasks together instead, so to be able to perform socially good matches. Hence, directory-based approaches, developed to support service discovery, offer a more suitable interaction paradigm; however, as we highlight in Section 2, available solutions are not directly applicable to human delay tolerant networks (DTNs).

In this paper, we present *ADESSO*, a semi-distributed self-organising directory and matching middleware service that supports opportunistic social networking in human DTNs (Section 3). Specifi-

[1] http://news.bbc.co.uk/1/hi/wales/north_east/7861733.stm
[2] http://www.rsf.org/News-and-information-fall-victim.html

cally, ADESSO supports the semantic specification of users' tasks (Section 3.1). In order to enable mobile users to publish their tasks in the network, ADESSO dynamically elects brokers based on nodes' mobility patterns (i.e., nodes' popularity) (Section 3.2). Popular nodes are those that are most likely to meet the largest number of other nodes in the network, and thus to gather the highest number of tasks. In order to maximise the chances of performing good matches (i.e., matches of tasks from users who have expressed reciprocal and high preferences in their social networks), brokers interact with each other when they meet, performing collaborative matching on tasks they carry (*broker collaboration*). Furthermore, ADESSO dynamically de-elects bad quality brokers (e.g., brokers that are no longer encountering large numbers of nodes) by having popular brokers claim stewardship of tasks from less popular ones (*broker fusion*) (Section 3.3). Finally, ADESSO performs semantic and social-aware task matching (Section 3.4) and notifies interested users by means of a delay-tolerant networking protocol (Section 3.5). We have conducted an extensive performance evaluation of ADESSO, using real human mobility traces and social networks (Section 4). Our preliminary evaluation confirms that ADESSO generates matches that highly satisfy users' social preferences, while entailing only a small communication overhead. The summary of our contributions and future work are finally discussed in Section 5.

2. RELATED WORK

Two streams of research are closely related to the work presented in this paper: semi-distributed solutions to the problem of service discovery in mobile settings, and mobility-aware protocols for content dissemination in delay-tolerant settings.

Semi-Distributed Communication Middleware. The problem of finding and matching users' tasks in human DTNs is very similar to the problem of finding services in pervasive settings. A variety of centralised directory services (e.g., Salutation [23], Jini [3]) have been proposed to find services in mobile settings; however, they all rely on the constant availability of static directories embedded in a fixed backbone. To overcome this limiting assumption, fully distributed protocols (e.g., SLP [12], UPnP [2]) have been proposed instead, but they do not scale well to dynamic settings, as they rely on broadcasting of service advertisements (i.e., our users' tasks). *Semi-distributed* solutions have beed shown to be a good compromise. Ariadne [22], for example, uses a protocol to dynamically elect *brokers* when needed: nodes acting as directories periodically advertise their presence (up to a given number of hops); a timeout expires when a node has not heard from a directory for a given period of time. At that point, the node initiates a directory-election protocol, broadcasting an election message in the network (again up to a given number of hops), and ending up by selecting as broker the node that "covers" the largest number of other nodes. The major limitation of this protocol is that it does not consider node mobility. A node's coverage is a very dynamic parameter: because of mobility, coverage may take very different values from the time an election is launched, to the time a service request needs to be answered (e.g., a node may be selected as broker and then disappear from the network soon after). This causes the costly election mechanism to be frequently re-launched. Furthermore, Ariadne relies on a synchronous publication/election strategy, under the assumption that nodes are connected to other nodes in the network most of the time. However, we have analysed the Bluetooth colocation traces of real human mobility ([11, 24]) and seen that the opposite holds, that is, a node's neighbourhood is often empty or very sparse, requiring the middleware to support a delay-tolerant task publication and broker election mechanism instead.

Mobility-Aware Communication Middleware. In the area of delay-tolerant networking, various routing protocols (e.g, [16, 19]) have been proposed that reason about node mobility patterns to decide how best to route content, making sure that relevant content reaches interested nodes, while minimising network overhead. These routing protocols have been used, for example, to port the publish-subscribe interaction paradigm to the mobile setting: in [10] and [4], each node acts as a broker, periodically broadcasting its interests to the one-hop away neighbours; adjacent brokers who receive these beacons store the information, along with a timestamp that enables them to maintain fresh information about local interests. A routing framework is then offered that is capable of disseminating content end-to-end, using metrics of social interactions. These routing protocols and frameworks assume that humans follow fairly regular movement patterns in their daily life, so that these can be learned over time; moreover, they assume that like-minded people are more likely to be colocated, or in close spatial proximity, than those who share no interests. Such assumptions have been confirmed by a number of studies on human behaviour [20, 14]. These approaches cannot be directly used to enable local social networking, as the interaction paradigm is fundamentally different: while in publish-subscribe and DTN settings content is *spread* in the network in a one-to-many fashion, we aim to *gather* users' tasks together, thus increasing the chances of finding quality matches between users who are directly or *indirectly* connected in the partially learned social network. However, a similar mobility-aware reasoning can be exploited to dynamically elect brokers that, based on their past movement and colocation patterns, are expected to serve well as matching catalysts in the near future.

In the next section, we thus propose a *semi-distributed* directory and matching middleware service, whereby publication and matching of tasks is achieved by means of *dynamically elected* and *self-organising brokers*, based on nodes' historical movement and colocation patterns.

3. ADESSO: A SEMI-DISTRIBUTED, SELF-ORGANISING DIRECTORY AND MATCHING SERVICE

In order to enable the kind of technologically-mediated social interactivity described in Section 1, we need a middleware that supports: (1) the semantic specification of user tasks and associated social preferences (Section 3.1); (2) their publication in the network (Section 3.2); (3) the further dissemination of the injected information in the network, in order to increase the chances of finding a good match (Section 3.3); (4) the matching of user tasks, in a way that satisfies users' social preferences (Section 3.4); and (5) the notification of computed matches back to the interested users (Section 3.5).

3.1 Task Specification

In terms of specification, we use a semantic model extending the Friend-Of-A-Friend (FOAF [1]) ontology defined in [6] to enable mobile users to specify a *user task* as a combination of four elements. An *activity* they would like to perform, a set of *social preferences* related to such an activity, a number k stating how many *recommendations* the user would like to receive from the system when matching that activity, and, finally, an *expiry time* (reffered to as TTL in the remaining of the paper) before which a match should be returned. To avoid ambiguity, activities are semantically defined by referring to existing ontologies (e.g., SUMO), taxonomies (e.g., Wordnet) or Web pages (e.g., Wikipedia). Social preferences are expressed as a specialization of the directed

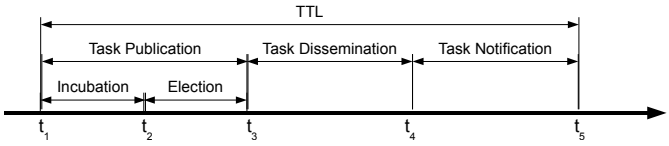

Figure 1: Task Lifecycle

weighted *'foaf:knows'* relation, indicating how much the user enjoys performing the stated activity with its known social relations. A lightweight semantic matchmaker [8], specifically developed to run efficiently on portable devices, is then employed to match activities.

Once the task has been specified, it is ready to be injected into the network. In order to understand how task publication, dissemination, matching and notification work, the full task lifecycle will be illustrated first. As shown in Fig. 1, task publication (Section 3.2) starts when a new task is created (time t_1); the task needs to be handed over to a broker that will then mediate its matching with other users' tasks. However, at the time of publication t_1, the publisher may not be within reach of any broker; it thus keeps the task locally for a maximum period of time called *Incubation* period (duration $Inc = t_2 - t_1$), during which the publisher tries to hand off the task to any broker it meets. If a broker has not been met by time t_2, a broker will then be dynamically elected in what we term the *Election* period (duration $Elect = t_3 - t_2$). By the end of the publication phase, a broker has received the newly created task and added it to the pool of tasks it is in charge of. During the task dissemination phase (Section 3.3), the broker will then collaborate with any broker it meets, trying to create quality matches for the tasks it carries (duration $Dissem = t_4 - t_3$). Finally, once a task has been matched at time t_4 (Section 3.4), the notification phase starts (Section 3.5), during which the broker will send recommendations back to the users involved in the performed match (duration $Notif = t_5 - t_4$). The overall lifecycle of a task must not last longer than the task's TTL, that is, $TTL = t_5 - t_1$. If a task cannot be matched before the end of its TTL, the task expires and the user who issued it is notified.

3.2 Task Publication

During the Task Publication phase, a newly created task is handed over from a publisher to a broker, with potential broker election required, in case no broker is encountered. Rather than attempting to carry out these functionalities instantaneously, ADESSO deals with them in a more delay-tolerant fashion: task injection is attempted for an incubation period $t_2 - t_1$, during which the publisher monitors its colocated nodes and hands off its task as soon as a broker is encountered. If no broker is encountered during this period, then broker election is attempted within an election period, following the strategy discussed below.

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 [18]. Specifically, 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 node A starts logging more detailed information about their encounters (i.e., day of the week/time of the day encounters occur). Based on the collected logs, A can now compute what we call a *regularity weight*, that is, the number of times A has met B in a given *regularity interval* (e.g., Monday 9am-10am), over a certain *observation period* (e.g., last four weeks). This information can thus be used by each node to *predict* how likely they are to encounter what familiar stranger nodes in the near future.

Furthermore, each node logs how many different devices it encounters during a given time interval ΔT. This value is continuously updated and locally fed in input to a Kalman filter [15], in order to predict how many devices this node is likely to meet in a forthcoming period ΔT. We call this predicted value *node popularity* $Pop_{\Delta T}(N)$; for example, $Pop_{24h}(N) = 6$ means that node N expects to encounter 6 different nodes in the next 24 hours. Nodes advertise their predicted popularity continuously, together with their presence. Such information is used to estimate the *quality* of a node to act as broker: intuitively, the higher a node's popularity, the higher the chances of it gathering many tasks from colocated devices, the higher the probability of performing quality matches.

When a publisher N initiates a broker election mechanism, the previously described regularity logs are used to estimate which nodes, among its familiar strangers, are likely to be encountered before the end of the election period. These nodes' regularity values are then used, together with the nodes' estimated popularity, to elect a broker. More formally, N computes, for each familiar stranger X, a utility function:

$$U_{Elect}(N, X) = RW_{Elect}(N, X) * Pop(X)$$

where $RW_{Elect}(N, X)$ is the maximum regularity weight recorded between N and X, for any timeslot ts_i within the $Elect$ period, that is, $RW_{Elect}(N, X) = \max(RW_{ts_i}(N, X))), \forall ts_i \subset Elect$; $Pop(X)$ is the latest predicted popularity value recorded by N for X. The node with the highest utility is then elected.

Once a broker A has been (pre)elected, the publisher node N waits until the two encounter: at that point, A is notified it is now a broker, and tasks are off-loaded from N to A. However, while moving, should N encounter a node C with better predicted popularity than A, then N would elect C as broker instead, and offload its current tasks to it. Furthermore, should N meet an existing broker after A has been pre-elected but before the two have actually met, the election process would be invalidated and N would submit its tasks to the just met broker. Finally, should the end of the election period be reached without N actually meeting A, N would self-elect itself as broker.

3.3 Task Dissemination

During the Task Dissemination phase, elected brokers cooperate upon encountering each other, in order to enhance the quality of social-based matching. ADESSO realises this cooperation by means of two mechanisms: *Broker Fusion* and *Broker Collaboration*.

Broker Fusion - When two brokers meet, if one (say A) is of much lower quality than the other (say B), then the lower quality one transfers all its unmatched tasks to the other, and ceases to act as broker. Superiority among brokers is determined by their popularity: upon encounters, A and B exchange their latest predicted popularity; if the difference is greater than a parametric threshold, the best broker takes over. For instance, if A has current predicted popularity $Pop_{\Delta T}(A) = 75$, B has current predicted popularity $Pop_{\Delta T}(B) = 155$, and the tolerated difference is 25% of the most popular one, than B takes the load off A and A ceases to act as broker, as $Pop_{\Delta T}(A) < Pop_{\Delta T}(B) * (1 - 0.25)$. Broker fusion is an essential mechanism to self-maintain the directory and matching service provided by ADESSO: in fact, poor quality nodes who have self-elected themselves as brokers (because of unavailability of any other broker in a given time period) as well as brokers whose popularity has decreased over time (because of changes in their mobility

patterns) are automatically removed from their brokers' duties. In so doing, only a minimal number of brokers are active at any given time, thus avoiding tasks being spread all over the network.

Broker fusion allows filtering out bad brokers. However, when two popular brokers meet, fusing them may not be the best action, as we may de-elect a broker that was relied upon by many other nodes, and that will thus have to be re-elected soon after. In these cases, ADESSO opts for broker collaboration instead.

Broker Collaboration - When two brokers of comparable quality encounter each other, the one with the smallest number of (currently unmatched) tasks (say A) transfers such tasks to the other broker (say B). B then performs a matching for the whole set of tasks, generating a list of pairs (i.e., matched tasks) that maximises the overall utility. Each task whose best match belonged to the other broker, is validated and ready to be notified. If a task T requires more than 1 recommendation (i.e., $k > 1$), ADESSO returns the k best matching tasks for T. Unmatched tasks, as well as tasks whose best match was already managed by the same broker, return to such broker instead, in the hope of finding better matching tasks later. ADESSO performs broker collaboration and fusion only once brokers are within reach (1 hop) of each other; this is to avoid leaving the system in an inconsistent state, due to task lists being lost while traveling multi-hop from one broker to another. Also, broker fusion and collaboration require whole task lists to be exchanged (and processed), causing both communication and computation overhead; we thus limit this overhead to situations where brokers are indeed physically in reach. Moreover, as ADESSO brokers are the most popular nodes in the network, hence it is likely they will meet during their tasks lifecycle, thus being able to perform quality matches without using expensive multi-hop communication.

3.4 Task Matching

The utility that users experience from local social networking depends on the ability of brokers to collate tasks from socially-related users, and to compute matches that satisfy users' social preferences. The lack of a global vision of the social network, which is due to its inherent distribution (each user has knowledge of its social links only), prevents centralised reasoning on user social preferences, as could be done in Web 2.0 social networking services. Instead, individual pieces of this network, encoded within tasks, are gathered by brokers in order to build a larger view of the social network. Algorithms for propagation of social preferences, in order to guess missing links from existing ones, could be further deployed within ADESSO to let brokers operate on denser networks as demonstrated in [7].

Based on the view of the social network that each broker has crawled, task matching can then be carried out using a variety of algorithms. In this paper we consider the *LocalSatisfaction* algorithm as defined in [6]. Specifically, let us consider, for simplicity of presentation, that each task has to be matched by one other task only, i.e., $k = 1$ (e.g., the user is interested in being recommended one single user with whom to perform an activity). Given a pair of matched tasks (T_1, T_2), submitted by users U_1 and U_2 respectively, we define $utility(T_1, T2)$ as the weight of the directed edge from U_1 to U_2 in the social network, if such edge exists, and 0 otherwise. When a task T has to be matched, LocalSatisfaction scans the full list of advertised and yet unexpired tasks, and returns the one delivering the best utility from the requesting user perspective only.

Note that in [6] the authors relied on a simplified middleware, whereby a statically chosen set of nodes was acting as brokers: task publication could only occur when within reach of a broker, and task dissemination simply consisted of brokers comparing tasks when within reach of each other. Such a solution is obviously quite limiting in real DTN settings, as nodes who rarely encounter the pre-selected brokers will suffer from unmatched (or poorly matched) tasks; moreover, it is not clear how many and which brokers to select. The ADESSO semi-distributed, self-organising directory and matching service presented in this paper overcomes these limitations by dynamically electing brokers when needed.

3.5 Task Notification

Tasks that have been matched, as well as those that have expired, are notified back to the involved users during Task Notification. Contrary to task matching, that requires an information gathering protocol, task notification requires information to be routed towards specific nodes in the network (i.e., unicast messages from a broker to nodes involved in a match). To accomplish this, we rely in this paper on existing routing protocols: for example, if a match has to be urgently notified, epidemic-style protocols can be used; however, if delays are tolerable and sparing resources is more important, then a variety of delay-tolerant routing protocols ([18, 19, 16]) can be used within ADESSO instead. An investigation of the most suitable approach to be adopted within ADESSO will be investigated as part or our future work.

4. EVALUATION

This section presents the performance evaluation of the ADESSO middleware. We begin with a description of the simulation setup we used and the metrics we have been collecting. We then list the experiments we have conducted and analyse the results obtained. A custom-built event simulator was developed in Java 5.0.

4.1 Simulation Setup

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 From the same trace set, which includes phone calls and text messages exchanged between users, we have extracted a social network whereby a link between user A and 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.

Publication Events Distribution The rate of task publication is expected to vary across users and across timeslots (e.g., day/night, week-days/week-ends). To model task publication as realistically as possible, we have used real text message traces, from the same MIT trace subset, as an analogy for the publication of user tasks. During the three months of experiment, more than 3000 messages were generated.

4.2 Metrics

In order to evaluate ADESSO, we collected three metrics:

Matching: The first metric we use is to count how many tasks are successfully *matched*, so that users effectively get recommendations for their requested activity. However, percentage of matches is in itself a poor metric, as we are not particularly interested in matches between strangers, but rather in matches of high social

value (i.e., among users who enjoy carrying out a given activity together).

Satisfaction: We thus use the *satisfaction* metric to quantify how well the system performed in matching user tasks, with respect to their users' weighted social network. The optimal result for a user (U_1) submitting task (T_1), would be if it were matched with the currently active task whose owner (U_2) is ranked highest by U_1. Although U_2 may inhabit a separate partition of the network from U_1, we use this upper bound as it gives an indication of U_1's potential best utility (called 'best matching hope' hereafter). The ratio between the utility that T_1 actually does generate, and the 'best matching hope' U_1 wanted to achieve, is defined as the *satisfaction*. Note that for measuring the generated satisfaction of each matched task, we only consider the first task recommended by the system (out of the k recommendations that may be requested by the user).

Overhead: We have measured the communication overhead incurred by ADESSO both in terms of the number of messages generated in the system due to broker election, fusion and collaborations, and in terms of the the traffic generated by those messages. Both the number of messages and the traffic generated have an impact on mobile nodes' resource consumption (e.g., battery).

For the estimation of the traffic generated, tasks are assumed to have an average size of 1 kB (we created a basic task specification, with an activity that refers to an existing ontology, that has three social preferences, and a TTL, and the actual size of such file was around 500 bytes, we have thus used 1kB as a plausible approximation of richer user task descriptions). Note that in the evaluation of both the number of messages and the traffic generated, only the overhead entailed by broker nodes is considered; information exchanged by non-broker nodes such as the exchange of node's popularity is ignored as it can be integrated in the node's HELLO protocol. Finally, we have measured the maximum required buffer size on elected brokers used to store user tasks.

4.3 Benchmark

We compare the performance of ADESSO with the following protocols:

Random, Independent Brokers: In this scenario, brokers are elected by mobile nodes but without reasoning on nodes' mobility patterns. Furthermore, elected brokers neither fuse nor collaborate. This scenario allows us to assess the benefits of reasoning on nodes' popularity for electing quality brokers as well as the benefits of broker collaboration and fusion in finding good recommendations for the users.

Epidemic Request Propagation and Matching: In this scenario, all the nodes in the network act as brokers. Upon encountering each other, brokers exchange a copy of all their unmatched tasks, leading to an epidemic flooding propagation of user tasks in the network. When a task is about to expire it is matched locally in all the brokers that have a copy of it, and all the corresponding recommendations are sent back to the requesting user. We expect in the scenario a very high satisfaction, as each task will be matched with its 'best matching hope' if the two tasks are in reach of each other (n-hops away). However, we also expect a very high overhead both in terms of traffic generated, number of events and required buffer size, as tasks are being exponentially duplicated in the network.

4.4 ADESSO Matching, Satisfaction and Overhead

We compare in this experiment the matching, satisfaction and overhead generated by ADESSO with respect to the protocols implemented in our benchmark, i.e., Epidemic Task Propagation and Matching, and Random Independent Brokers.

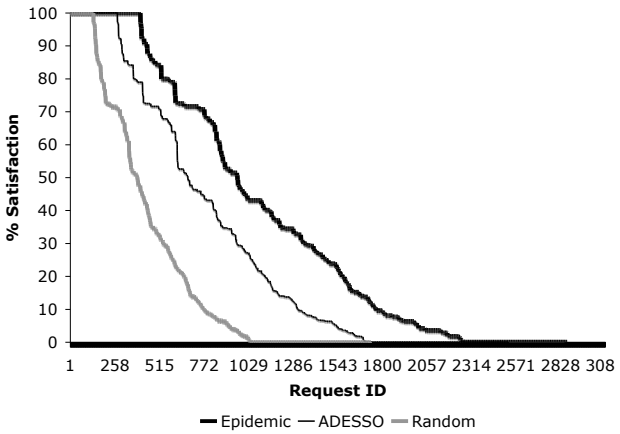

Figure 2: ADESSO: Distribution of Satisfaction

ADESSO Matching: As expected, Epidemic has the highest percentage of matched requests, i.e., 93%, followed by ADESSO with 81%, then Random Brokers (56%).

ADESSO Satisfaction: We have measured the distribution of the satisfaction generated by the various algorithms of our benchmark and results are depicted in Fig. 2, where the X axis represents task identifiers and the Y axis contains the generated satisfaction for that user task. ADESSO clearly outperforms the random independent brokers. Finally, as expected, Epidemic-based task propagation and matching generates a better satisfaction than ADESSO. By aggregating the values of the generated satisfaction for all the matched tasks, Epidemic-based matching performs 26% better than ADESSO. However, the overhead of the Epidemic-based task propagation and matching is significantly higher than ADESSO's generated overhead as shown in Table 1 and analysed below.

ADESSO Overhead: Table 1 compares the overhead, in terms of number of messages, traffic generated and buffer size, as required by ADESSO and Epidemic. We do not consider the random independent brokers as those brokers do not interact with each other and thus do not send any messages and do not generate any traffic. As shown, Epidemic generates a traffic of 3 orders of magnitude higher than ADESSO with a massive traffic of more than 6.6 GB during the three months experiment, which makes it unfeasible for use in real mobile settings. ADESSO further requires 43% less memory than Epidemic to store user tasks.

Overall, results show that ADESSO is a valuable choice for the realisation of the Local Social Networking vision as it generates a fairly high satisfaction while entailing a very low overhead.

	No. Messages	Traffic (MB)	Buffer Size (kB)
ADESSO	682	2.5	132
Epidemic	114020	6686	234

Table 1: ADESSO Overhead

5. CONCLUSION

While social networking Web sites enable virtual interactions between socially related people (i.e., Global Social Networking), the widespread adoption of networked handheld devices will foster physical interactions between them (i.e., Local Social Networking). In order to enable the Local Social Networking vision, we presented ADESSO, a self-organising directory and matching middleware service that exploits radio connectivity to find out people that are colocated and share similar interests. ADESSO supports

semantic task specification, as well as task publication, dissemination, matching and notification by means of self-organising brokers. Brokers are dynamically elected based on their mobility patterns and their popularity; upon meeting each other, they may either collaborate or fuse, thus dynamically re-organising themselves and eventually de-electing bad quality brokers. Our preliminary performance evaluation, conducted using real human mobility traces and social networks, confirmed that ADESSO generates matches that highly satisfy users social preferences entailing only a small overhead.

As part of our ongoing work, we are conducting experiments using different mobility traces (i.e., 36 users in the city of Cambridge, UK [24], 500 cabs in the city of San Fransisco (USA) [21]) and social networks. We also aim at investigating the most appropriate mobility-aware routing protocol for task notification within ADESSO. Our future work includes the integration of incentive mechanisms in ADESSO (such as [17]), to ensure that elected brokers will behave effectively and will be rewarded for doing so. Finally, we are investigating the integration of privacy-aware matching algorithms as performed by privacy-aware service discovery protocols, where untrusted brokers can carry out 'blind' matching of sensitive information [9].

6. REFERENCES

[1] The FOAF project. http://www.foaf-project.org/.

[2] Universal Plug and Play. http://www.upnp.org/, 1998.

[3] K. Arnold, B. O'Sullivan, R.W. Scheifler, J. Waldo, and A. Wollrath. *The Jini Specification*. Addison-Wesley, 1999.

[4] Roberto Baldoni, Roberto Beraldi, Leonardo Querzoni, Gianpaolo Cugola, and Matteo Migliavacca. Content-based Routing in Highly Dynamic Mobile ad hoc Networks. *International Journal of Pervasive Computing and Communications*, 1(1):277 – 288, December 2005.

[5] Russell Beale. Supporting Social Interaction with Smart Phones. *IEEE Pervasive Computing*, 4:35–41, April 2005.

[6] Sonia Ben Mokhtar and Licia Capra. From Pervasive To Social Computing: Algorithms and Deployments. In *ACM International Conference on Pervasive Services*, London, UK, July 2009.

[7] Sonia Ben Mokhtar, Liam McNamara, and Licia Capra. A middleware service for pervasive social networking. In *M-MPAC workshop of the ACM/IFIP/USENIX 10th International Middleware Conference*, December 2009.

[8] Sonia Ben Mokhtar, Davy Preuveneers, Nikolaos Georgantas, Valérie Issarny, and Yolande Berbers. EASY: Efficient SemAntic Service DiscoverY in Pervasive Computing Environments with QoS and Context Support. *Journal of System and Software*, 81(5), May 2008.

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

[10] Paolo Costa, Cecilia Mascolo, Mirco Musolesi, and Gian Pietro Picco. Socially-aware Routing for Publish-Subscribe in Delay-Tolerant Mobile Ad Hoc Networks. *IEEE Journal On Selected Areas In Communications (JSAC)*, 26(5):748–760, June 2008.

[11] Nathan Eagle and Alex (Sandy) Pentland. CRAWDAD data set mit/reality (v. 2005-07-01). Downloaded from http://crawdad.cs.dartmouth.edu/mit/reality, July 2005.

[12] E. Guttman, C. Perkins, J. Veizades, and M. Day. Service Location Protocol, Version 2. Technical Report RFC 2608, IETF, June 1999.

[13] Pan Hui, Augustin Chaintreau, Richard Gass, James Scott, Jon Crowcroft, , and Christophe Diot. Pocket Switched Networking: Challenges, Feasibility and Implementation Issues. *Second IFIP Workshop on Autonomic Communications*, October 2005.

[14] Pan Hui, Jon Crowcroft, and Eiko Yoneki. Bubble Rap: Social-based Forwarding in Delay Tolerant Networks. In *MobiHoc '08: Proceedings of the 9th ACM International Symposium on Mobile ad hoc Networking and Computing*, pages 241–250, New York, NY, USA, May 2008. ACM.

[15] Andrew H. Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, 1970.

[16] Anders Lindgren, Avri Doria, and Olov Schelén. Probabilistic Routing in Intermittently Connected Networks. *SIGMOBILE Mobile Computing Communications Review*, 7(3):19–20, July 2003.

[17] Jinshan Liu and Valérie Issarny. An Incentive Compatible Reputation Mechanism for Ubiquitous Computing Environments. *International Journal of Information Security*, 6(5):297–311, September 2007.

[18] Afra Mashhadi, Sonia Ben Mokhtar, and Licia Capra. Habit: Leveraging Human Mobility and Social Network for Efficient Content Dissemination in MANETs. In *WOWMOM'09: Proceedings of 10th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, June 2009.

[19] Mirco Musolesi, Stephen Hailes, and Cecilia Mascolo. Adaptive Routing for Intermittently Connected Mobile Ad Hoc Networks. In *Proceedings of the 6th IEEE International Symposium on World of Wireless Mobile and Multimedia Networks*, pages 183–189, Washington, DC, USA, June 2005. IEEE Computer Society.

[20] Erik Nordström, Christophe Diot, Richard Gass, and Per Gunningberg. Experiences from Measuring Human Mobility using Bluetooth Inquiring Devices. In *MobiEval '07: Proceedings of the 1st International Workshop on System Evaluation for Mobile Platforms*, pages 15–20, New York, NY, USA, June 2007. ACM.

[21] Michal Piorkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser. CRAWDAD data set epfl/mobility (v. 2009-02-24). Downloaded from http://crawdad.cs.dartmouth.edu/epfl/mobility, February 2009.

[22] Francoise Sailhan and Valérie Issarny. Scalable Service Discovery for MANET. In *PERCOM '05: Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications*, pages 235–244, Washington, DC, USA, March 2005. IEEE Computer Society.

[23] Salutation Consortium. Salutation. http://www.salutation.org/, 1999.

[24] James Scott, Richard Gass, Jon Crowcroft, Pan Hui, Christophe Diot, and Augustin Chaintreau. CRAWDAD data set cambridge/haggle (v. 2006-09-15). Downloaded from http://crawdad.cs.dartmouth.edu/cambridge/haggle, September 2006.