FriendSensing: Recommending Friends Using Mobile Phones

Daniele Quercia[§‡] Licia Capra[‡]
[§]MIT SENSEable City Laboratory, Cambridge, USA
[‡]Dept. of Computer Science, University College London, UK
quercia@mit.edu, l.capra@cs.ucl.ac.uk

ABSTRACT

Social-networking sites, such as Facebook, require members to manually find and confirm their friends. Finding friends is tedious for some and may be made less so by automating the process. We propose to do so by means of a framework that we call *FriendSensing*. Using short-range technologies (e.g., Bluetooth) on their mobile phones, social-networking users "sense" and keep track of other phones in their proximity. Proximity records are then processed using a variety of algorithms that are based on social network theories of geographical proximity and of link prediction. This processing can be performed either on the social-networking website, after records have been uploaded, or locally on the user's mobile phone, so that privacy-conscious individuals do not have to disclose their proximity data to the social-networking website. The result is a personalized and automatically generated list of people the user may know. We evaluate the extent to which FriendSensing helps users find people they know, and we do so against real mobility and social network data.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*distributed applications*;; H.3.3 [**Online Information Services**]: Web-based services.

General Terms

Algorithms

Keywords

Social matching systems, recommender systems, Web 2.0

1. INTRODUCTION

Social networking services (e.g., MySpace, Facebook, Last.fm, etc.) enable people who share interests to come together in online communities, where they can interact in a variety of ways. The effectiveness of these services comes from the social connections their members make. However, finding and confirming friends on social-networking websites is a tedious and time-consuming task, so much so that Facebook has recently launched the tool "People You May Know". This tool recommends "friends of friends": If A knows B and B knows C, then Facebook tells A: "You May Know C". This way of recommending friends is purely based on social proximity, and is thus unlikely to be of value to new members, whose direct social relations have yet to be uncovered.

Alternative ways of recommending friends have been recently proposed and scrutinized (Section 2). Those ways are based on either social-networking profiles (e.g., they recommend people with shared interests) or audio recordings from collar devices (e.g., they recommend people with whom one has had lengthy face-to-face contacts). The former requires users to create fairly detailed profiles, and is thus nonetheless tedious than eliciting friends in the first place; to lighten the process, profiles are increasingly being defined via user-defined freely-chosen tags, with a negative impact on the accuracy of algorithms used to automatically compute profile similarity [9]. The latter has had so far very limited applicability, as it requires the usage of invasive technology (i.e., collar devices) for data collection.

However, a less invasive and more widely available form of data collection exist, which has not been explored by researchers yet, that is, to simply have mobile phones keep their Bluetooth on to track other phones in proximity. Since people usually carry their mobile phones [17], this way of collecting data is appealing for its simplicity and consequently begs an important research question: can proximity data from Bluetooth be used to recommend friends?

We demonstrate that the answer is 'Yes', and we do so by making two main contributions:

- A framework called *FriendSensing* that automatically recommends friends by logging and analysing colocation data. More precisely, using short-range connections (e.g., Bluetooth), mobile phones "sense" and record which other mobile devices are in proximity. FriendSensing then processes those records and suggest to users people they may know. It does so by using social network theories of "geographical proximity" and of "link prediction" (Section 3).

- An evaluation of the effectiveness of FriendSensing on real mobility and social network data from the Reality Mining Project [6] (Section 4).

.

2. EXISTING SOLUTIONS

To automatically discover and recommend friends, existing approaches mainly vary depending on what information they process: social-networking profiles, emails, or data from portable devices.

Social-networking profiles. Chen *et al.* [10] proposed four algorithms for suggesting people on Beehive (IBM internal social-networking website). The algorithms are different combinations of two basic ideas. The first idea is to match people by common interests - to match, for example, those who blog on similar topics or share the same role within IBM. The second is to match people by social connections - to match those who are in "social proximity" of each other by, for example, connecting friends-of-friends. The two ideas match people by the content of their profiles, and the researchers conceded that their ways of matching people are preliminary and should be improved further. More promising ways have been then proposed. For example, Ferne [21] has suggested the use of recommender systems. Those systems traditionally process user ratings (e.g., product reviews) to recommend new products such as movies or albums, but they could be easily adapted for recommending people. Toward that adaptation, Terveen and McDonald took the first step in 2005 [22] - they reviewed social science literature and then proposed a specific research agenda for systems that recommend people (which they called "social matching systems").

Emails. Karagiannis and Vojnovic gathered the emails exchanged by more than 100,000 employees of their company's research labs [23]. They represented their data as a graph whose nodes are employees and whose links are email exchanges. Then, to recommend new email addresses for contact lists, they connected "friends-of-friends" relationships.

Portable device data. Wyatt *et al.* [5] built a framework with which collar devices capture audio readings and automatically suggest to their users who they may know. Using their audio sensors, collar devices record face-to-face conversations and, based on conversation length, they infer who is likely to befriend whom. The inference is made possible by knowing global properties (e.g., clustering coefficients) of the users' social network. Under this assumption, the promise is that one could *accurately* reconstruct the whole social network.

There has been no work on recommending friends that exploits more readily-available information (such as proximity data from mobile phones) and that requires no a priori knowledge about a user and its social network.

3. OUR PROPOSAL: FriendSensing

To enable (new) members of social networking websites automatically discover their friends, we have designed the FriendSensing framework. FriendSensing automatically creates personalised recommendations of people a user may know, by means of the following two steps (Figure 1):

1. Logging Encounters - using short-range radio technologies ready available on almost all modern mobile phones (e.g., Bluetooth), each user transparently records encounters with colocated people. More precisely, each phone A keeps track of how many times it has met another phone B and how much time it has spent with B. Indeed, mobile phones have been recently demonstrated to be a cheap and viable way of sensing and logging colocation information [19]. We make here the assumption that a mobile phone is a *personal* device, and that it is not shared among people. Moreover, we assume it is possible to link devices (e.g., phone's Bluetooth ID number) to users' identities in social-networking websites. Note that the Cityware project has already convinced a considerable number of Facebook users to register their phone's Bluetooth ID number on their profiles [13].

2. Recommending Friends - colocation records are processed to elicit relevant encounters and to arrange them into a weighted social network (Section 3.1); this network is then traversed to compute personalised lists of people each user may know (Section 3.2). Note that FriendSensing does not prescribe *where* the processing of proximity records, and the navigation of the inferred social network, should occur: both can be performed by the social networking website, after these records have been uploaded, or by the mobile device itself, if such records are considered sensitive and should thus be maintained private. We now present algorithms for proximity processing and for network navigation in general terms, and defer a discussion about the implications of different architectural deployments to the evaluation (Section 4).

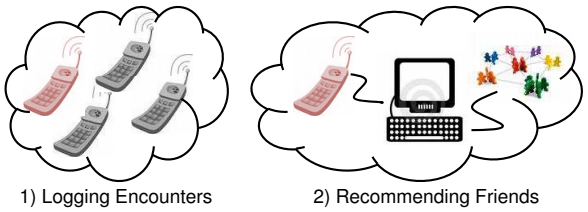


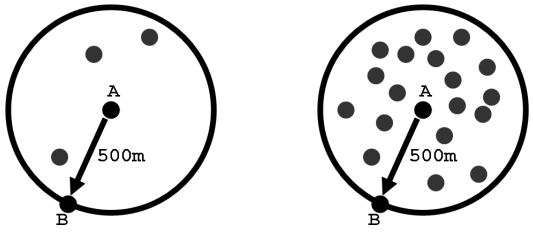
Figure 1: FriendSensing Framework

3.1 Processing Encounters

Once colocation logs have been collected, FriendSensing must filter out irrelevant encounters from *relevant* ones; that is, for each user A, it must identify which of A's encounters are likely to be A's friends. FriendSensing does so by computing the probabilities of A befriending other individuals (A's friendship probabilities) from proximity data.

Researchers have already suggested ways of computing these probabilities from geographical proximity. However, geographical information is not widely available on mobile phones; should localisation technology like GPS become a commodity, it would still fail to capture indoor encounters (e.g., at home, in the office, on the tube, in the pub, etc.). We thus first review approaches to compute A's friendship probabilities from geographical proximity, and then adapt their formulae to our case of "mobile phone proximity".

Geographical Proximity. Researchers modeled the probability of two individuals being friends based on the intuition that friendship probability increases with geographic prox-



Countryside (a) Central London (b)

Figure 2: Friendship Probability changes from (a) the countryside to (b) central London.

imity - the closer two individuals are, the likelier they are to befriends. A case in point is Kleinberg [11, 12] who modeled the probability of A and B being friends as:

$$p(A \rightarrow B) \propto \frac{1}{dist(A,B)^r} \quad (1)$$

This tells us that the probability of being friends with a person at a distance d decays as d^{-r} for some power of r. The best choice for the exponent r has been found to be 2, and this is best because the resulting network is navigable and small-world [11]. By navigable we mean that, by only knowing her own social contacts, a person is able to reach any other person in the network, and she does so in a limited number of steps.

When this model was first proposed it was unclear how accurate it would be in reality. That is why, four years later, Liben-Nowell *et al.* [16] collected a half million profiles on the blogging site LiveJournal, on which people reported their US locations and lists of friends. The researchers then applied formula (1) to those profiles, observed a *loose* fit with the data, and concluded that the absolute value of geographic distance alone is insufficient to model friendship.

This finding comes as no surprise if one considers that two individuals at the same distance may find themselves in areas with different population *densities*. To see how, consider that A and B live 500 meter apart. At the same very distance, A and B would likely be either next-door neighbors in the countryside (Figure 2(a)) or complete strangers in central London (Figure 2(b)).

This suggests that one also needs to consider population density. Libel-Nowell *et al.* [16] did so in a simple way - they replaced the absolute distance $dist(A,B)$ with a ranked distance:

$$p(A \rightarrow B) \propto \frac{1}{rankDist_A(B)+1} \quad (2)$$

where the denominator is A's rank of B, which is the number of people who are closer to A than B is, and it is expressed as:

$$rankDist_A(B) = |\{C : dist(A,C) < dist(A,B)\}| + 1.$$

According to expression (2), the probability of A befriending B depends on the number of people within distance $dist(A,B)$ and not on $dist(A,B)$ itself, which accounts for population density. That is because, geographically, the more dense the population between A and B, the lower B ranks. So B ranks far higher in the countryside (it ranks 4^{th} in Figure 2(a)) than it would do in central London (it ranks 19^{th} in Figure 2(b)). Consequently, at the same distance, B is more likely to befriend A in the countryside than in central London.

Using this definition, the researchers fitted the LiveJournal data optimally. Interestingly, they estimated that 66% of LiveJournal friendships form through *geographic* process: that is, geography partly predicts friendship, even in a virtual community such as LiveJournal. This result is rather promising - it suggests that, by analyzing proximity, we may well discover a considerable number of friends.

Mobile Phone Proximity. As previously argued, we do not record geographic distances but rather keep track of: how many times a user A has met (e.g., it has been within Bluetooth range of) user B (frequency $freq(A,B)$), and how much time it has spent with B (duration $dur(A,B)$). So we now need to express the friendship probability as a function of frequency or duration. A plausible way of doing so is by considering that the probability of A befriending B increases with $freq(A,B)$ and with $dur(A,B)$ respectively. That is:

$$p(A \rightarrow B) \propto freq(A,B)^r \quad (3)$$

$$p(A \rightarrow B) \propto dur(A,B)^r \quad (4)$$

However, as with geographical information, we cannot consider frequency or duration alone to compute friendship probabilities, because both of them are non-uniformly distributed. Indeed, individuals do have skewed mobility patterns; this has been shown not only for college students [6] (against whose movements we will run our evaluation), but also for conference attendees [2], and for hundreds of thousands of mobile users [7]. As a consequence, frequency or duration alone bear little meaning. To see why, consider two individuals who have met four times. Those four times entail completely different meanings in different situations - for instance, they would reflect either *random encounters* (if those individuals are jet setters and go out a lot) or *strong friendship* (if they are homebodies and rarely go out). Rather than using absolute frequency and duration values, we have thus taken their rank. From frequency, the friendship probability becomes:

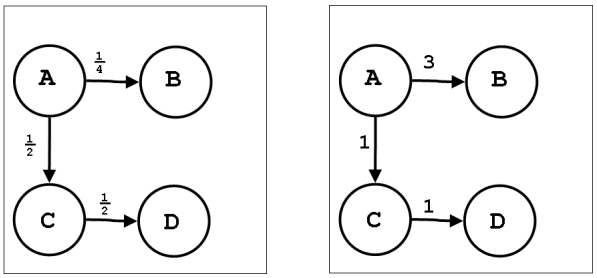
$$p(A \rightarrow B) \propto \frac{1}{rankFreq_A(B)+1} \quad (5)$$

where:

$$rankFreq_A(B) = |\{C : freq(A,C) > freq(A,B)\}| + 1.$$

Consequently, the probability of A befriending B depends on the number of people who have met A more frequently than B has done - it does not depend on $freq(A,B)$ itself but it depends on how A ranks B. That is, if A and B met four times, the friendship probability $p(A \rightarrow B)$ does not only depend on number four but also on how active A is. If A goes around a lot, then B would rank lower than if A often stays home.

Similarly, by replacing frequency with duration, the friendship probability becomes:



(a) (b)

Figure 3: Networks of Encounters. Link weights are (a) friendship probabilities or (b) ranks.

$$p(A \rightarrow B) \propto \frac{1}{rankDur_A(B) + 1} \quad (6)$$

where:

$$rankDur_A(B) = |\{C : dur(A, C) > dur(A, B)\}| + 1.$$

Again, the probability of A befriending B depends not on $freq(A, B)$ itself but on the number of people who have met A for longer than B has done. For instance, if A and B have spent two hours co-located, then the friendship probability $p(A \rightarrow B)$ does not depend on $2h$ itself but on how active A is (i.e., how many people A has been co-located with for more than $2h$).

Social Network of Encounters. From the proximity logs, the above friendship probabilities can be computed and used to infer a weighted social network of encounters: each mobile device is represented as a node, and a link is added between any pair of individuals who have met at least twice (this is to remove encounters caused by chance). Each link $A \rightarrow B$ is then weighted using either the row probability of A befriending B (computed using either formula 5 or 6 - Figure 3 (a)), or deriving, from such probabilities, the ranking of B from A's perspective (Figure 3 (b)). We we explain when to opt for probabilities and when for ranks next.

3.2 Computing Recommendation Lists

Once the network of encounters has been computed, FriendSensing processes it to compute personalised lists of people each user may know, that is, to predict which of A's encounters are likely to be A's friends. In the literature of social networks, this problem is called "link prediction" and different methods have been proposed in recent years to tackle it [15]. These methods assigns a $score(A, B)$ to a pair of nodes (A, B) following one of two possible strategies:

Shortest Path - The score between a pair of nodes A and B is the weighted length of the shortest path between them [20]. The intuition behind it is that social networks are "small worlds" (individuals are connected by short chains [20]) and, as such, if there are short paths between A and B, then A and B are likely to befriend each other. The shortest path algorithm accepts weights on the network that represent capacity constraints - in our case, weights that reflect how unlikely it is for two nodes to befriend each other. Rankings reflect just that (the higher $rankDur_A(B)$ or $rankFreq_A(B)$, the less likely A befriends B), and should thus be associated to links in the social network of encounters. The path length is then weighted in the sense that it is the sum of the weights along the shortest path. In the network of encounters shown in Figure 3 (b), A's rank of B is 3, A's rank of C is 1, and C's rank of D is 1. By computing *shortest path* from user A's perspective, the following scores are thus computed: $score(A, B) = 3$, $score(A, C) = 1$, and $score(A, D) = 2$. Such scores are then used to create A's personalised friends' recommendation list $\{C(1^{\text{st}}), D(2^{\text{nd}}), B(3^{\text{rd}})\}$.

Markov Chain Algorithms - For this class of algorithms, the score between a pair of nodes A and B is computed as the fraction of time spent at B by a random walk in the network originating in A [24]. In this network, weights reflect connection strength between pairs of nodes. In our social network of encounters, we thus use friendship probabilities $prob(A \rightarrow B)$ (computed using Formulae 5 or 6) as links' weights, as shown in Figure 3(a). After starting at node A (which is called prior node), the walk may unfold in different ways depending on which of the following three algorithms is deployed:

- *PageRank with prior.* At each node, the walk either iteratively moves through one of the node's outgoing links (whose weights are transition probabilities) or jumps back to the prior node A [8].
- *K-MarkovChain.* It is similar to *PageRank with prior*. The difference is that the walk has now fixed length K [24].
- *HITS with prior.* At each node, the walk either moves through one of the node's *incoming* or outgoing links or jumps back to the prior A [24].

Algorithmically, to compute scores, the three algorithms all convert the network in a first-order Markov chain. Once scores for a walk originating in A have been computed, they are then used to build A's recommendation list.

The shortest path is the simplest algorithm among the four and yet it has been shown to work best on a number of social networks [4, 24]. In our evaluation, we will confirm this literature finding once again (Section 4).

3.3 Summary of FriendSensing Strategies

The FriendSensing framework thus offers eight strategies for recommending friends, derived from combining a strategy for processing proximity data into friendship probabilities (either *frequency* or *duration*), with a link-prediction algorithms (*shortest path*, *PageRank*, *KMarkovChain* or *HITS*).

4. EVALUATION

4.1 Simulation Setup

The goal of FriendSensing is to recommend to its users people they may know. To ascertain the effectiveness of FriendSensing at meeting this goal, our evaluation ought to

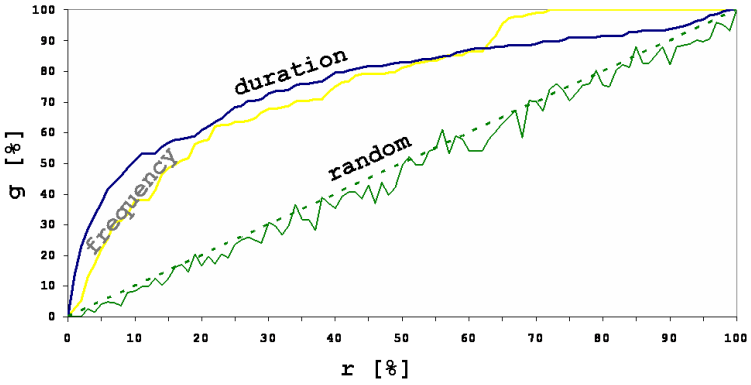


Figure 4: Fraction g of predicted ties versus fraction r of recommended community. Three strategies considered: *random*, *frequency*, and *duration*.

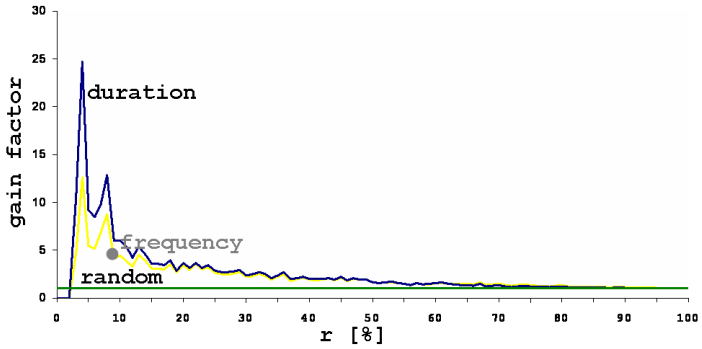


Figure 5: Gain factor versus fraction r of recommended community. Three strategies considered: *random*, *frequency*, and *duration*.

answer the question of how effectively FriendSensing suggests people one knows.

To do so, we set up a simulation driven by real data. Ideally, we should do so by using empirical evidence about how people move (mobility traces) *and* about, among those people, who befriends whom (social network). The problem is that mobility traces do not come with corresponding social networks - one usually has the mobility traces of some people and the social network of others. The only exception is the Reality Mining project at MIT [6]. The MIT traces contain colocation information from 96 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 (5 minute) Bluetooth device discoveries. Note that, while focusing on these mobility traces, we expect the results obtained to equally hold in other human mobility scenarios; in fact, as existing analysis demonstrates [3], such traces share many unifying features (e.g., node inter-contact time, formation of cliques, etc.) with other mobility traces (e.g., Cambridge and Dartmouth traces [1]). Beside providing 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 between user A and user B is created if A sent a text message or made a phone call to B.

In our simulations, we used the MIT mobility traces to log encounters; using these logs, we ran FriendSensing and computed friends' recommendations. We then compared these recommendations with the MIT actual social network (largest connected component) and computed the fraction of the social network's ties correctly predicted by FriendSensing. We refer to this fraction as "*good* recommendations" g, and we study how g varies while we increase the percentage r of community members recommended to each user from 0 to 100%.

4.2 Results

In order to study the effect of the colocation processing strategy separately from the link prediction strategy, we performed two sets of experiments.

(1) Frequency vs. Duration. In the first set of experiments, we aimed to compare the effectiveness of *frequency* as a colocation processing strategy, as opposed to *duration* (both described in Section 3.1). We did so by disabling any link propagation strategy, and using the ranking produced by the frequency / duration colocation processing strategies on each node to build recommendations' lists instead. This is equivalent to running FriendSensing on people's mobile

[1]http://crawdad.cs.dartmouth.edu/

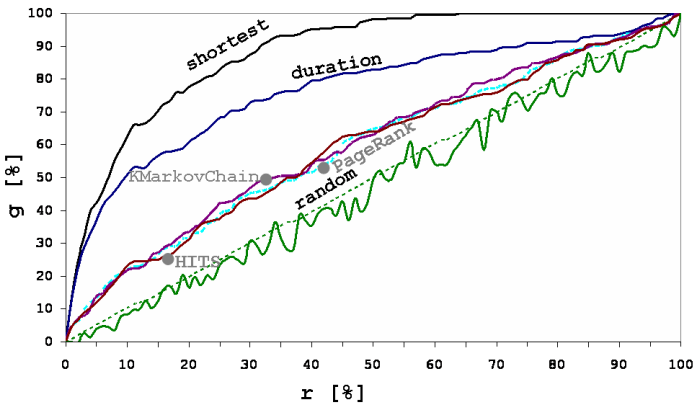


Figure 6: Fraction g of predicted ties versus fraction r of recommended community. Six strategies considered: *random*, *duration*, *shortest path*, *PageRank*, *HITS*, and *KMarkovChain*.

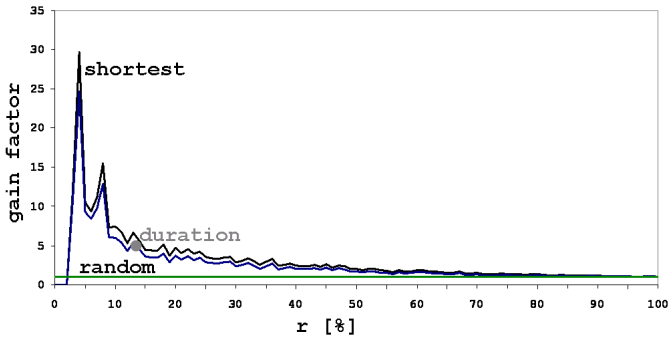


Figure 7: Gain factor versus fraction r of recommended community. The gain is over *random*, and it is for two strategies: *shortest path* and *duration*

devices, without reporting they proximity logs to the social networking website (where the full FriendSensing approach, including link propagation, could be executed).

Figure 4 plots g (good recommendations) versus r (recommended community) for these strategies with respect to a *random* selection of people to recommend. For the random strategy, g increases linearly with r - the random strategy fluctuates around a straight line (dashed in the figure). That is because the more community members are recommended, the likelier to get some of them right. At the extreme of $r = 100\%$ (the whole community has been recommended to each user), g reaches 100% (for all strategies). As for the two remaining strategies, they both perform significantly better than random. Note that *duration* discovers friends faster than *frequency*. Interestingly, after each user has been recommended 60% of the community, *duration* slows down, and *frequency* takes on and is able to discover few other friends.

To see now which strategy performs better over another, we compare *frequency* and *duration* against the random one. We do so by defining the *gain factor* over *random* as:

$$gain_{strategy} = \frac{g_{strategy}}{g_{random}} \quad (7)$$

where $g_{strategy}$ is the fraction of good recommendations for $strategy = duration \mid frequency$ and g_{random} is that for *random*. A gain factor of one means the strategy performs no better than random (no gain). A factor of two means that the strategy performs twice as better as random.

Figure 5 shows that *duration* gains more than *frequency* - especially so for the first 20% of community members recommended. As one expects, frequency and duration die off up to a point where both of them flatten toward *random* (no gain). That is because, after recommending most friends, any strategy has left only few friends to recommend, and those friends are hard to predict.

Duration and "Link Prediction". The second set of experiments aimed to compare the different link prediction strategies presented in Section 3.2 (i.e., *shortest path*, *PageRank*, *HITS*, and *KMarkovChain*) on the quality of recommendations instead. We did experiments whereby these strategies where executed on a social network of encounters built using *duration* information and *frequency* information. Since results obtained with *duration* were consistently better than those obtained with *frequency*, we report results for the former case only. This setup corresponds to scenarios where mobile users have reported their (processed) colocation information to the social networking website, so that link prediction can be performed.

Figure 6 plots g versus r for all the four strategies. We also plot the results obtained with our baseline *random* strategy, as well as when using *duration* without propagation, to high-

light what privacy-conscious users would miss by not uploading their colocation information on the social networking website.

As shown, *PageRank*, *HITS*, and *KMarkovChain* perform equally and only show small differences due to confidence on the results. Those results are similar and come from the common use of Markov chains by the three algorithms. Also, one would be better off using only *duration* rather than combining it with those three algorithms. That is not necessarily bad news as it suggests that, by relying only on her own proximity information, a user both gets quality recommendations and, while doing so, she retains control of her own data. In line with the literature, *shortest path* performs best. Indeed, Figure 7 shows that it gains more than *duration*, and it does so consistently. That is because, unlike *duration*, *shortest path* is able to suggest to a user A also those friends who belong to the A's social circle but have not been met by A yet.

5. DISCUSSION

Privacy Concerns. One of the problems of existing friends-of-friends approaches is that they expose sensitive information. To see why, consider that A has two lovers B and C. Those approaches would readily match B and C - they would say to lover B: "You may know C". A for some reason may feel uncomfortable about it. This is true not only of parallel daters such as A but also of dutiful citizens whose conscientiousness makes their spouses proud. Simply because of privacy concerns [14], those dutiful citizens may feel uncomfortable uploading their contact to social-networking websites. However, they can still run FriendSensing on their phones if they are willing to resort to the *duration* strategy. As shown experimentally in the previous section, this strategy produces quality recommendations and relies only on proximity information collected by the device on which FriendSensing runs - it only relies on its user's private information. So users have control over what data they are willing to disclose and, as a pleasant by-product, they also eliminate their switching costs from one social-networking website to another; that is because they keep their own data not on social-networking websites but on their mobile phones instead. Also, individuals can still use the best recommendation strategy (*shortest path*) and suffer from little privacy exposure; they can do so by using security techniques that verify social ties while exposing minimal information about those ties [18].

Not Only Proximity Data. To run their experiments, "mobile computing" researchers need real data, and they often need to know how people move (mobility traces) and, among those people, who befriends whom (social network). Since researchers have mobility traces but do not usually have the corresponding social networks, for years now, they have been calling for ways of inferring social networks from mobility traces. From mobility, FriendSensing infers potential friends. To go from inferring friends to accurately inferring social networks, FriendSensing still needs to be refined. One promising way of doing so is to consider non-geographic information. Indeed, research has shown that friendship does not only depend on geographic factors, but it also depends on whether individuals have similar occupation, cultural backgrounds, or roles within a company [1]. Therefore, it is promising for FriendSensing to reason not only on proximity information but also on non-geographic information. FriendSensing may do so by adapting existing work by Adamic and Adar [1] or, more recently, work by Clauset *et al.* [4].

6. CONCLUSION

FriendSensing automates the process of finding friends on social-networking websites. Using their mobile phones, FriendSensing users profit from a set of recommendation strategies grounded in the literature of social networks. Using real mobility and social network data, we have validated that a strategy that keeps track of how much time people spend co-located (*duration*) works better than a strategy that simply keeps track of how may times people meet each other (*frequency*). Plus, we have also demonstrated that, by arranging duration data in a network and by then running shortest path on this network, one is able to effectively rank encounters who happen to be friends.

The effectiveness of FriendSensing strategies may depend on the type of mobile community. To test whether this is true and the extent to which it is so, one should gather mobility patterns and social networks of communities other than Reality Mining's. Also, to test whether FriendSensing users would actually find discovering friends less tedious, one should run a user study. One such study may have social-networking members register their phones (MAC addresses) on their profiles (as already successfully attempted by the Cityware project [13]); those profiles will then feature "people you may know" widgets fed by FriendSensing.

7. REFERENCES

[1] L. A. Adamic and E. Adar. How to search a social network. *Social Networks*, 27(3):187–203, 2005.

[2] A. Chaintreau, P. Hui, C. Diot, R. Gass, and J. Scott. Impact of Human Mobility on Opportunistic Forwarding Algorithms. *IEEE Transactions on Mobile Computing*, 6(6):606–620, 2007.

[3] A. Chaintreau, P. Hui, C. Diot, R. Gass, and J. Scott. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Transactions on Mobile Computing*, 6(6):606–620, 2007. Fellow-Crowcroft„ Jon.

[4] A. Clauset, C. Moore, and M. E. J. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453:98–101, 2008.

[5] Danny Wyatt and Tanzeem Choudhury and Jeff Bilmes. Learning Hidden Curved Exponential Random Graph Models to Infer Face-to-Face Interaction Networks from Situated Speech Data. In *Proc. of the* 23^{rd} *Conference on Artificial Intelligence (AAAI)*, Chicago, July 2008.

[6] N. Eagle and A. S. Pentland. Reality mining: sensing complex social systems. *Personal Ubiquitous Computing*, 10(4):255–268, 2006.

[7] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi. Understanding individual human mobility patterns. *Nature*, 453(7196):779–782, June 2008.

[8] T. H. Haveliwala. Topic-Sensitive PageRank: A Context-Sensitive Ranking Algorithm for Web Search. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 15(4):784–796, 2003.

[9] A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme. *Information Retrieval in Folksonomies: Search and Ranking*, pages 411–426. 2006.

[10] Jilin Chen and Werner Geyer and Casey Dugan and Michael Muller and Ido Guy. "Make New Friends, but Keep the Old": Recommending People on Social Networking Sites. In *Proc. of ACM Conference on Human Factors in Computing Systems (CHI)*, Boston, April 2009.

[11] Jon Kleinberg. The small-world phenomenon: An algorithmic perspective. In *Proc. of the* 32^{nd} *ACM Symposium on Theory of Computing (STOC)*, pages 163–170, Oregon, May 2000.

[12] J. Kleinberg. The convergence of social and technological networks. *Communications of the ACM*, 51(11):66–72, 2008.

[13] V. Kostakosand and E. O'Neill. Cityware: Urban computing to bridge online and real-world social networks. *Handbook of Research on Urban Informatics: The Practice and Promise of the Real-Time City*, pages 195–204, 2008.

[14] N. Lathia, S. Hailes, and L. Capra. Private Distributed Collaborative Filtering using Estimated Concordance Measures. In *Proceedings of ACM Recommender Systems (RecSys)*, Minneapolis, October 2007.

[15] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of American Society for Information Science and Technology*, 58(7):1019–1031, 2007.

[16] D. Liben-Nowell, J. Novak, R. Kumar, P. Raghavan, and A. Tomkins. Geographic Routing in Social Networks. *Journal of the National Academy of Sciences*, 102(33):11623–11628, 2005.

[17] P. J. Ludford, D. Frankowski, K. Reily, K. Wilms, and L. Terveen. Because i carry my cell phone anyway: functional location-based reminder applications. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*, pages 889–898, Montréal, Canada, 2006.

[18] Michael J. Freedman and Antonio Nicolosi. Efficient Private Techniques for Verifying Social Proximity. In *Proceedings of the* 6^{th} *International Workshop on Peer-to-Peer Systems (IPTPS)*, Bellevue, February 2007.

[19] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell. Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, pages 337–350, New York, NY, USA, 2008. ACM.

[20] M. E. J. Newman. The Structure and Function of Complex Networks. *SIAM Review*, 45(2):167–256, 2003.

[21] Peter Ferne. Collaborative Filtering and Social Capital. In *Proc. of W3C Workshop on the Future of Social Networking (MSNWS)*, Barcelona, November 2008.

[22] L. Terveen and D. W. McDonald. Social matching: A framework and research agenda. *ACM Transactions Computer-Human Interactions*, 12(3):401–434, 2005.

[23] Thomas Karagiannis and Milan Vojnovic. Behavioral Profiles for Advanced Email Features. In *Proc. of* 18^{th} *International World Wide Web Conference (WWW)*, Madrid, April 2009.

[24] S. White and P. Smyth. Algorithms for estimating relative importance in networks. In *Proc. of the* 9^{th} *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 266–275, Washington, August 2003.