Trust-Based, Privacy-Preserving Context Aggregation and Sharing in Mobile Ubiquitous Computing

Michael Xing and Christine Julien

The Center for Advanced Research in Software Engineering
The University of Texas, Austin, TX, USA
c.julien@mail.utexas.edu

Abstract. In ubiquitous computing environments, we are surrounded by significant amounts of context information about our individual situations and the situations we share with others around us. Along with the widespread emergence of ubiquitous computing and the availability of context information comes threats to personal privacy that result from sharing information about ourselves with others in the vicinity. We define an individual's context to be a potentially private piece of information. Given the individual context of multiple participants, one can compute an *aggregate* context that represents a shared state while at the same time preserves individual participants' privacy. In this paper, we describe three approaches to computing an aggregate measure of a group's context while maintaining a balance between the desire to share information and the desire to retain control over private information. Our approaches allow dynamic tuning of information release according to *trust levels* of the participants within communication range. By evaluating our approaches through simulation, we show that sharing aggregate context can significantly increase the rate at which a group of co-located users learns an aggregate measure of their shared context. Further, our approaches can accomplish high quality context sharing even in situations with low levels of trust, assuming the availability of a small number of highly trustworthy partners.

1 Introduction

Ubiquitous computing allows users to share information about their personal situations directly with one another, enabling users to collaboratively construct aggregate views of their shared local situations, or *context*. Constructing these aggregate views requires sharing potentially highly sensitive and personal information, which in turn relies on users' trust in one another. Imagine a mobile app that can communicate with other nearby mobile devices and retrieve the names of the apps that other mobile devices' users are using. This could be useful from a social connectivity perspective; we could learn what other apps that other people in a similar social situation are using at a given time. For example, at a sporting event, we could determine what other apps nearby spectators are using to augment their experiences, for example to check scores or view replays. However, the app poses a significant threat to the privacy of the mobile device users. Our approach strives to address the tension between this privacy constraint and the incentives for exchanging context information among nearby mobile devices.

This paper explores practical mechanisms to enable ubiquitous computing users to construct aggregate views of their shared context while retaining control over the dissemination of their private data. In our target environment, participants with smart mobile devices (e.g., smart phones) collect and share information with one another directly

(i.e., across peer-to-peer links) without the support of an infrastructure. Such an environment is becoming increasingly commonplace as smaller, wearable devices are becoming mainstream: mobile phones record users' locations using GPS and other localization technologies; Google Glass can take pictures, record videos, recognize a user's voice, and capture myriad context information about a user[1]; Nuubo, a wireless cardiac monitoring platform, can transmit physiological parameters to a user's doctors[2]; Sony's smart watch can connect to Android phones and display received texts, emails, and notifications[3]. These devices possess considerable computing power and can communicate through direct wireless channels. Direct interaction among nearby users enables new forms of data sharing but also presents a challenge in enabling users to control the release of their potentially private information. Only by sharing information, however, do participants reap many of the benefits of the information-rich environment.

We assume an established "trust network" in the ubiquitous computing environment. Specifically, for a given participant, this provides a trust value for every other participant in the network. Work exists in establishing flexible trust networks in mobile ad hoc environments [6, 10, 14, 17, 20]. Our context aggregation and sharing mechanisms utilize trust values computed by such a trust network to determine the amount of private information to release to other nearby users. Returning to our previous example, at a sporting event, a spectator is likely to be seated nearby a group of friends or family with whom he has a high degree of trust. The spectator is perfectly willing to share private information with these trusted friends; aggregating together the context of a group of trusted friends can *obfuscate* each individual's private information, enabling the aggregate to be shared with acquaintances with a somewhat lower level of trust. Using this novel combination of trust and aggregation, our privacy preserving context distribution mechanism reduces the risk of privacy leakage. Our approaches allow users to gradually reveal their information in aggregate, to both protect the individuals privacy and to converge to a collective (correct) aggregate of context information.

Our motivation stems from users' needs to be able to feel safe to collect and share context information in settings that lack centralized trusted authorities. Limited work exists on privacy in ubiquitous computing, but most approaches require elaborate, centralized infrastructure; we review these methods in Section 2. Our approaches target completely infrastructure-less environments and rely on direct, peer-to-peer wireless interactions among users' devices. We make the following concrete contributions: (*i*) we define three aggregation schemes that explicitly trade individual privacy for the degree of data sharing in mobile ubiquitous computing environments; (*ii*) we tune the amount or nature of sharing to established measures of trust; and (*iii*) we evaluate our aggregation approaches under different deployment scenarios and trust networks. We measure our approaches' abilities to converge to a correct assessment of the shared context in a short amount of time. Our approaches can significantly speed up the rate at which the entire group learns an aggregate measure of their shared context. Further, our approaches achieve a high quality of context aggregation, even with low levels of trust among participants, as long as there are a small number of highly trusted collaborators.

[1] `http://www.google.com/glass/start`
[2] `http://www.nuubo.com`
[3] `http://sonymobile.com/us/products/accessories/smartwatch/specifications`

2 Related Work

Our basic goal is similar to that of *differential privacy* in statistical databases: it should be possible to accurately query a database while maintaining the privacy of individuals whose data is represented in that database [5]. Specifically, queries should release information about the *population* represented in the database without releasing information about any individual that is not generally publicly available. Techniques from differential privacy motivate our goals, but they assume that information about the population is collected in a single (secured) central database.

One of an individual's most sensitive pieces of data is the individual's location; many techniques exist to protect the privacy of individuals' locations. Most approaches somehow augment the location data, for example protecting sensitive location trajectories in a centralized database by inserting realistic fake trajectories [19], by perturbing location trajectories by "crossing paths" of multiple users [9], or by adding uncertainty to objects' locations in moving object databases [1]. These solutions are specific to location data, and the focus is often on attempting to maintain a high fidelity (correctness) of responses to queries about locations while preserving privacy.

Significant recent efforts have focused on privacy and on its interplay with crowdsensing specifically and with mobile distributed sensing more generally. In the former scenarios, a query issuer requests information that is sensed by mobile participants, potentially aggregated, and returned to the querier. Ensuring participation requires ensuring privacy, most often with respect to the location of the user whose device does the sensing [4, 15, 16]. Other approaches take advantage of the additive properties of desired aggregates and use *data slicing* [25] or cryptographic techniques [13] to compute complete aggregates for independent sets of data providers. These approaches either assume resilient communication (e.g., no slices of data can be lost) or an ultimate backend (centralized) server. In contrast, we aim for a purely distributed approach in which all of the users desire the aggregate of context information shared among themselves and not mediated by a service provider that sits between the querier and the tasked mobile sensing devices. We also explore the novel use of *trust* in influencing the release of private information in mobile and ubiquitous computing environments.

Other approaches have attempted to preserve privacy for data types beyond location by introducing noisy data in participatory sensing [7]; this work's motivation is quite close to our own, where individual users compute aggregates (fusions) over locally available data, but this related work does not incorporate trust (instead relying on random perturbations). The approach circumvents the fundamental limitations of perturbation for privacy by taking advantage of properties of the targeted time series data.

Our motivation (and approach) is also similar to secure multi-party computation [26], in which participants share information to jointly compute some function (e.g., an aggregate) over their individual data without explicitly releasing their (potentially private) individual information. This technique has been applied to distributed data mining [3], to computing a sum of private data while relying on data slicing [24], and even to collaborative filtering in peer-to-peer networks [2]. While the approach is decentralized, it requires a high degree of controlled coordination among participants that is not possible in purely ad hoc environments. Further, because it is based on cryptographic primitives, the computational complexity is not reasonable for mobile devices

or common tasks [18]. We take advantage of the fact that coordinating parties in mobile and ubiquitous computing situations may not be *completely* distrustful of each other, and we leverage this trust to reduce the cost of achieving acceptable levels of privacy.

Existing work that combines trust and privacy generally focuses on *trading* privacy for trust, i.e., revealing private information to others to earn a more substantial level of trust [23], and on incentivizing this tradeoff [22]. We look at trust and privacy from a different perspective, presupposing a framework for establishing trust in other individuals that allows graduated release of private information based on established trust levels. Establishing trust among collaborating parties has been well studied in both completely distributed mobile ad hoc networks and in pervasive computing, and several approaches exist that we can rely on to establish trust values between individuals [6, 11, 20, 21]. For the remainder of this paper, we assume such a mechanism is in place and that, in using such a mechanism, we can rely on a *trust table* that is available to each individual on the local device. The trust table maps an another individual's identifier to a *trust level*, which our algorithms will use in determining how to share data.

3 Trust-Based Sharing of Context

Our operational model is one in which a group of participants make independent decisions about sharing context, without the aid of any infrastructure. The goal of the participants, in general, is to learn some aggregate measure of the entire group's context (e.g., the apps in use by other nearby spectators at a sporting event, the average grade of a group of students on an exam, an average of a health indicator for a group at a fitness club, or the bounding box of the locations of contributors to a participatory sensing application). When a participant i encounters a participant j, i must decide what information to share, where the options range from sharing i's individual context data (which results in the largest loss of privacy) to sharing an aggregate that combines i's data with some other participants' context values. This *partial aggregate* that each participant computes is that participant's working estimate of the target global aggregate. We assume that the only way for participants to exchange information is to encounter each other and make that exchange directly, i.e., through a peer-to-peer connection. Our approach assumes the aggregate functions can be computed incrementally (e.g., a sum, average, minimum, maximum, union, bounding box of locations, etc.) and that individuals' context values do not change. Along with each aggregate, we maintain a list of contributors to the aggregate to prevent including a participant multiple times.

The novelty of our approach lies in the following key observations. First, we do not commonly find ourselves in situations in which we have absolutely no trust in any other participants. Second, mutually trusting participants can work together to aggregate their information to obfuscate their individual context, increasing their individual levels of privacy. Third, sharing aggregate measures of context contributes positively to an entire group learning a (near) correct value for the aggregate of the entire group. While Alice may be willing to reveal her individual exam grade to her best friend, Bob, (and Bob may be willing to do the same), she may feel more self-conscious about releasing it to Cindy, who she does not know (or trust) as well. However, once she and Bob have exchanged their individual context information, they can aggregate (e.g., average) their scores and give the average to Cindy, sacrificing less of their individual privacy. Of

course, an average of two grades provides only a small degree of added privacy; an average of 50 grades provide much more. Therefore, sharing aggregate context values depends not only on the trust values associated with the recipients, but also, at least indirectly, on the size of the aggregate (i.e., the number of values aggregated).

We assume that each participant (e.g., device, application, or user, depending on the application) maintains its own *trust table*, that holds a *trust value* for every other participant. It is not required that trust values are mutual (i.e., participants i and j need not have the same level of trust in each other). Trust values can be based on reputations, can be learned, can change over time, and can even be context-dependent [21]; these concerns are outside the scope of this paper. Instead, we rely on the availability of this trust information to determine when to share potentially private context information. Concretely, we assume that, when a participant is about to share private context information, the participant can query its local trust table to determine the level of trust associated with the potential recipient of the data. Based on the level of trust, the participant can determine whether to share information and what specifically to share (e.g., individual context data or an aggregate of multiple individuals' context data).

We assume trust values are on an unbounded continuous scale; a value of 0 indicates complete trust, and larger values indicate lower trust. For convenience, we assume that the trust table values correspond to aggregate sizes; if participant i has a trust value of x for participant j, then i is willing to share its information with j as long as it is contained in an aggregate with size greater than x. Practically, this representation of trust values requires a processing step to convert trust values computed from a scheme such as [21]. Fig. 1 shows a small example that demonstrates some of these concepts; in the figure, Alice is willing to share her individual context information with Bob, who can then combine it and share it with Cindy, who is less trusted by both Alice and Bob. We next describe four schemes that determine how to share context information, given the availabile trust information.

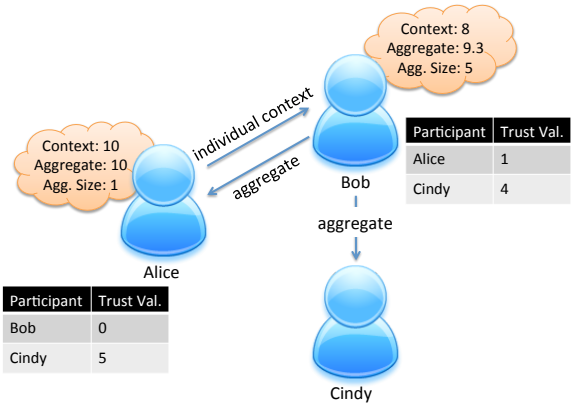


Fig. 1. An Example of Trust-Influenced Context Sharing; "Aggregate" in this case is the average value; "Agg. Size" designates the number of participants represented in the aggregate

Scheme 1: Individual Context Only. The first scheme is a baseline; in the first scheme, participants only ever share individual context, and they only share that context with other participants that they trust completely (i.e., for which the trust value is 0). When participant i encounters participant j, i determines whether j is completely trusted. If not, i does nothing. If j is completely trusted, participant i sends participant j its individual context information. Upon receiving this information, j incorporates i's information into an incrementally computed aggregate (that includes j's own context

information as well as any other pieces of information that j has received from other participants). In this scheme, as average trust decreases, context information moves more slowly, and the aggregate that any participant can feasibly compute is just the average of context values from other participants that completely trust the participant.

Scheme 2: Aggregate Context Only. In the second scheme, participants incrementally compute aggregates and share only those aggregates with other participants they encounter. Aggregates can be computed incrementally by adding in additional participants' context if they are not already represented in the aggregate or by merging two aggregates if their contributor lists are disjoint. When participant i encounters participant j, to determine whether to share the computed aggregate, i retrieves j's trust value from the trust table. If the *size* of the aggregate is larger than the trust value, then i sends the aggregate to j. If not, i sends nothing to j. This scheme is a generalization of Scheme 1, as an aggregate of size one is simply the individual context information of participant i. On the receiving side, things are a bit more complicated when receiving an aggregate than when receiving a piece of individual context. Because the recipient may already store a partially computed aggregate, the recipient must determine what to do with the new aggregate. In general, if there is an intersection in the contributors to the received aggregate and the stored aggregate, the recipient can only keep one of the aggregates[4]. In Scheme 2, we keep the originally stored aggregate. If there is not an intersection in the contributors to the received and locally stored aggregates, the recipient merges the aggregates, generating an even larger aggregate.

Scheme 3: Smart Aggregate Context. This third scheme differs from the second only in that instead of the recipient keeping the original aggregate, it keeps the larger of the two aggregates (i.e., the larger of the received aggregate and the previously stored one). Intuitively, this scheme should perform better with respect to the computation of the correct aggregate value; as we will see in Section 4, this is not always the case.

Scheme 4: Mixed Information. The fourth scheme mixes aspects of the above approaches. When participant i encounters j, i still uses its trust value for j to determine what to send. However, in addition to sending the aggregate if the aggregate size is larger than j's trust value, if j's trust value is 0 (i.e., i completely trusts j), i also sends its individual context. Upon reception, j behaves like the second scheme unless there is an intersection between i's transmitted aggregate and j's stored aggregate. If there is no intersection, j just merges the two. If there *is* an intersection, j instead simply merges i's information into j's stored aggregate. At first glance, these seems to be an obvious addition, but this fourth scheme does come with the disadvantage of exchanging extra information, which comes at an increased cost of communication; in Section 4, we investigate whether this effective doubling of the overhead achieves better results.

4 Experimental Results

To compare and contrast our schemes for context sharing subject to privacy constraints, we implemented the schemes in our Grapevine context framework [8]. Grapevine piggybacks context information (whether individual context or aggregate information) on data packets transmitted in the course of other network (application) traffic.

[4] Some aggregation functions are *duplicate insensitive*, and any aggregates can be merged. We assume this is not the case, and address the issue of duplicate sensitive aggregates instead.

We implemented our approaches in the ONE network simulator [12]. Each node was assigned a context value and given the task of attempting to find the average of all of the context values in the network. We assume context values are static; we discuss handling dynamic context values in Section 5. We measure the *percent error* of each node's estimate of the global aggregate. The overheads of our approaches are low (see [8] for a presentation of the overhead of piggybacking context information in Grapevine). The approaches all generate the same amount of extra data except for the Mixed Information scheme, which generates twice the number of piggybacked bits.

We first look at simple networks in which a participant can have just one of two levels of trust in another participant: high (complete trust) or low (complete distrust). Our second set of experiments extends the complexity of the trust distributions in the simple network. Our final experiments explore more realistic deployments. We plot the average error in the computed aggregate over time; we stopped the simulations when the aggregate had stabilized. Table 1 gives the evaluation settings.

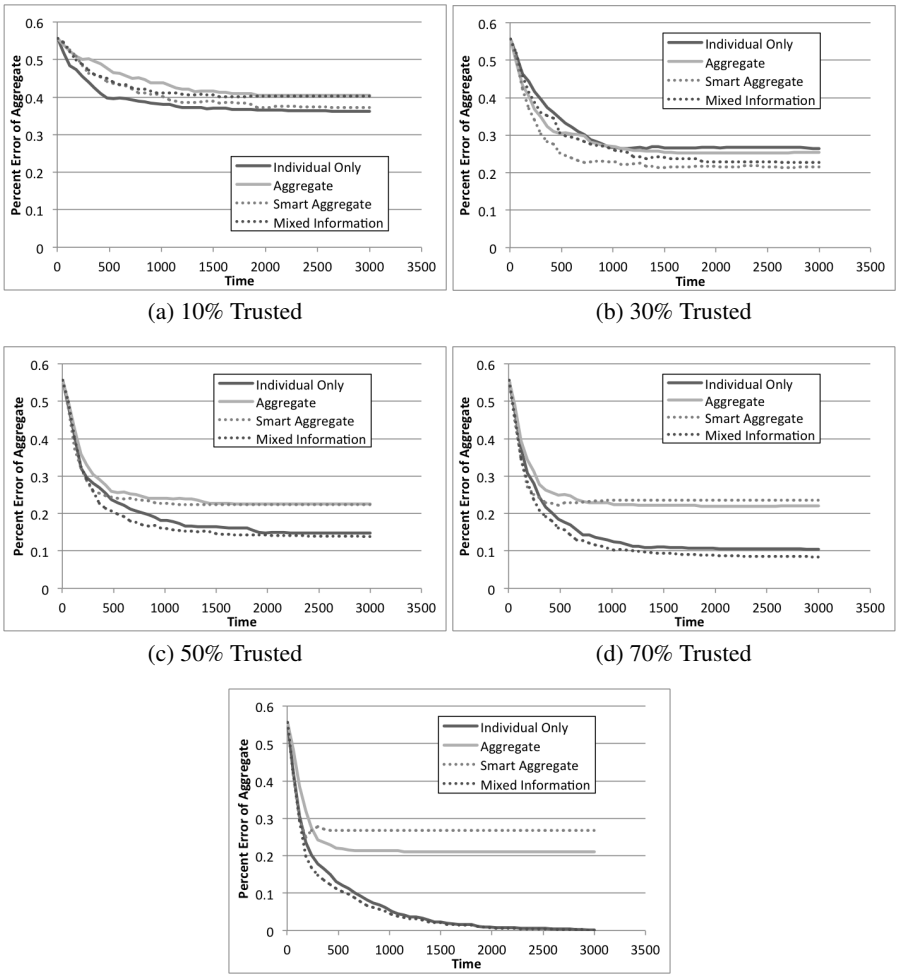
Table 1. Simulation Settings

Setting	**Value**
Transmit speed	250 KB/second
Transmit range	30 meters
Movement	Random waypoint
Speed	$U(3, 4)$
World size	300m $\times$ 300m

Binary Trust. In the first stage of our experiments, we used a small network and highly control trust values to benchmark the behavior of our four schemes. These networks consisted of 10 mobile participants. Grapevine does not generate its own traffic; instead it piggybacks context (either the individual context value or a computed aggregate) on top of these application-level packets. Each participant generated a new packet for some other randomly selected participant, on average, every five seconds. In general, this relatively high traffic load is beneficial to context sharing since context information can spread more quickly. We provide results for situations when each participant trusted 100%, 70%, 50%, 30%, or just 10% of the other participants.

Starting from the bottom of Fig. 2, we see that when the levels of trust are high, nothing significantly outperforms just sharing individual data. Because participants are able to directly collect the data that goes into the aggregate, the aggregate's error is very low. As we move to the mid-range of trust values, the quality of the four schemes comes together. At 30% trustworthy participants, we start to see that when the trustworthiness of the participants falls, the aggregate schemes show the potential to outperform the individual scheme. Finally, for the situation with very low trust, the quality of the aggregate falls off precipitously, indicating that, when a participant trusts very few others, it is difficult to share aggregate information with any quality.

Mixed Trust. In our second experiments, we explored these last two points in more depth, attempting to identify trust distributions in which the aggregate schemes excel (and thereby push the envelope of protecting privacy in the face of untrustworthiness) and attempting to identify just how low we can push the trustworthiness of participants and still achieve a reasonably low error rate in the aggregate. We stay with our simple 10 participant network, but we explore the trust distributions shown in Fig. 3, assigning a fraction of the participants to be highly trustworthy (with whom a participant will share individual data), a fraction to be of medium trustworthiness (with whom a participant will share a medium sized aggregate; in this example, an aggregate of size 5 or larger), and a fraction to be completely untrustworthy.



(a) 10% Trusted

(b) 30% Trusted

(c) 50% Trusted

(d) 70% Trusted

(e) 100% Trusted

Fig. 2. Average quality of computed aggregate with two trust values: high and low

When there is a very low level of trust among the participants (Distribution 5 in Fig. 3 and the corresponding results in Fig. 4(d)), the quality of the aggregate remains quite low. Further, when the trust levels are relatively high (Distribution 2 in Fig. 3 and the corresponding results in Fig. 4(a)), directly sharing individual information remains the best option. Where the trust levels are more mixed (Fig. 4(b) and (c)):, we see potential for aggregation to improve information quality while adhering to participants' privacy requirements.

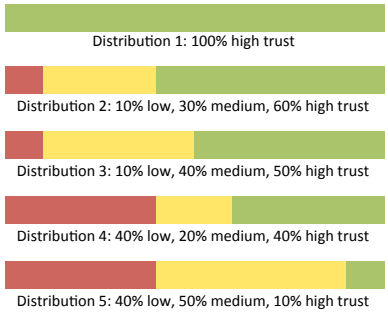
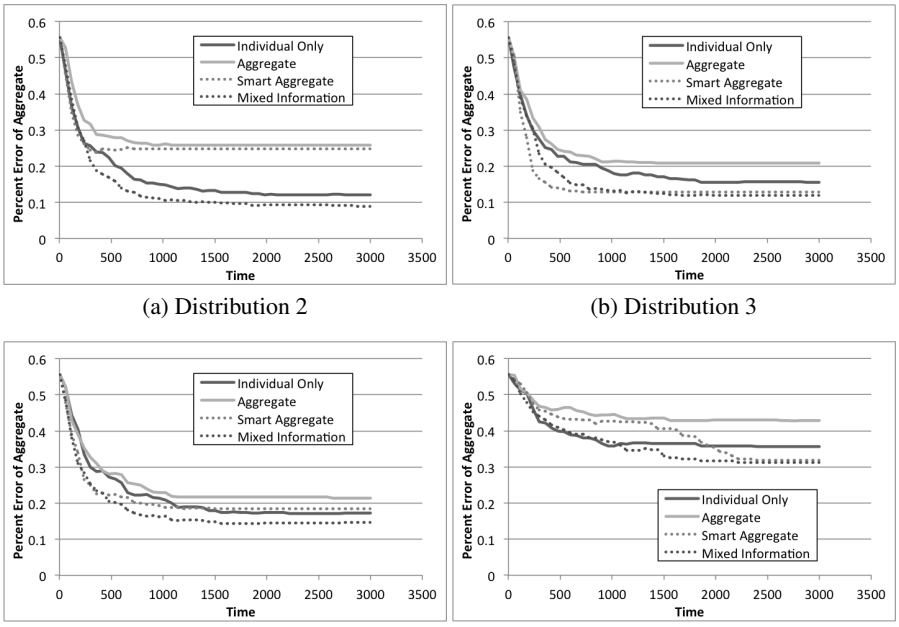


Fig. 3. Trust Distributions

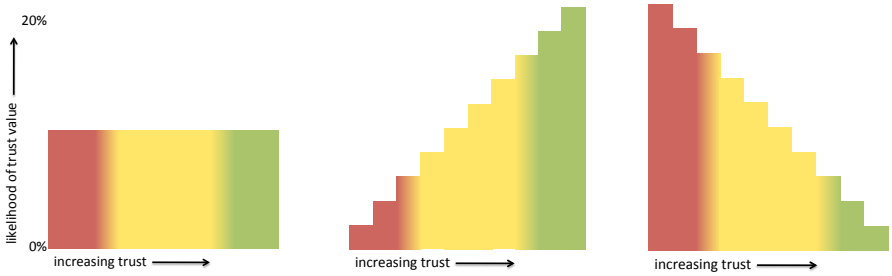


(a) Distribution 2 (b) Distribution 3

(c) Distribution 4 (d) Distribution 5

Fig. 4. Average quality of computed aggregate with three trust values in distributions from Fig. 3

Random Trust. For our last experiments, we evaluate how our schemes would perform "in the wild." We increase the size of the network to 50 mobile participants. We assign a participant's trust value for another according to three different trust distributions, shown in Fig. 5: *Random*, in which the trust is chosen equiprobably from 10 possible trust values ranging from completely untrustworthy to completely trustworthy, *More Trusted*, in which the choice is weighted toward the more trustworthy values, and *Less Trusted*, in which the choice is weighted toward the less trustworthy values.

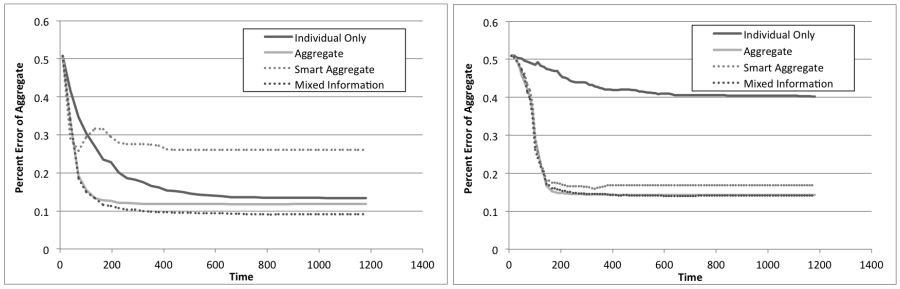


(a) Random (b) More Trusted (c) Less Trusted

Fig. 5. Random Trust Distributions

We highlight two key findings that demonstrate the benefits of using trust to control the release of private information in dynamic networks. Fig. 6 shows that ***sharing aggregate information can significantly help distribute context information, especially in situations of relatively low trust, assuming a handful of highly trusted participants in the network***. In the *More Trusted* case (Fig. 6(a)), the benefits of sharing aggregates

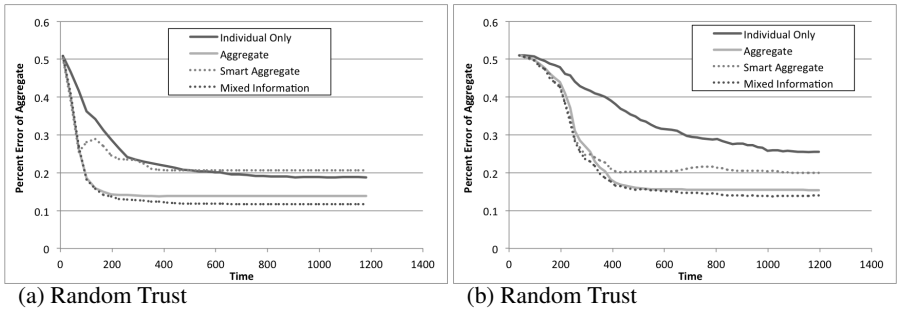
is marginal compared to sharing individual context. Even in this scenario, however, sharing aggregates leads to a quicker assessment of the aggregate value (i.e., the curves for the Aggregate and Mixed Information schemes lie to the left of the the Individual scheme). More strikingly, Fig. 6(b) shows that, for the *Less Trusted* distribution, all three aggregate schemes drastically outperform the Individual scheme, both in terms of the speed of assessing the aggregate and in the quality of the computed aggregate. In the Individual scheme, there are simply not enough direct contacts with highly trustworthy individuals to compute an accurate aggregate from only individual information.



(a) Random trust weighted to high trust (b) Random trust weighted to low trust

Fig. 6. Average quality of computed aggregate with weighted randomly assigned trust values

In Fig. 7, we use the *Random* trust distribution from Fig. 5; Fig. 7(a) uses the same traffic generation rate as before: on average, each participant generates a new application-level packet every 5 seconds. In Fig. 7(b), on average, each participant generates a new application-level packet only every 50 seconds. Understanding the behavior of our schemes under these lower traffic conditions is important since reducing network overhead is essential in these dynamic networks that rely almost exclusively on battery operated devices and wireless links. Fig. 7 shows that, ***in situations when fewer opportunities are available for piggybacking context, sharing aggregate information results in much more rapid and higher quality computation of the global aggregate***.



(a) Random Trust
(1 packet per participant every 5 seconds)

(b) Random Trust
(1 packet per participant every 50 seconds)

Fig. 7. Average quality of computed aggregate with randomly assigned trust for different traffic

Figs. 6(b) and 7(b) highlight another key benefit of Grapevine. In many ubiquitous computing scenarios, users find themselves in situations where they will choose not to share any of their context information because of the potential sacrifice of their privacy. By enabling users to share their context information within aggregates instead of only

individually, Grapevine enables a much higher degree of context sharing and learning, improving the experiences of all participants in the ubiquitous computing application.

5 Conclusions and Future Work

We explored using trust to influence how private context is shared in dynamic mobile, ubiquitous computing applications. By incrementally computing aggregate measures of context and basing how aggregates are shared on their size relative to the trustworthiness of the recipient, our context sharing schemes control the release of private information. Both the degree of trustworthiness and the desired quality of aggregate information are application-dependent; the results in this paper give a foundational understanding that application designers could use in making tradeoffs for their implementations.

For convenience, we used a linear correlation between trust values and aggregate sizes to demonstrate the relationships between decreasing trust and increasing aggregate sizes. While this gives important insights, the relationship between trust and the size of the shared aggregate may not be linear. Studying alternative (i.e., non-linear) relationships and the ability of application developers to tune them is future work. Further, when a participant shares individual context, the recipient has complete control of that information and could potentially share the individual data directly. This must be accounted for by conservatively assigning trust; this is why the results in Fig. 6(b) are so important: even in scenarios weighted towards lower trust, having a small number of highly trustworthy partners is sufficient for bootstrapping context sharing.

Further, our approach "leaks" the identity information of the participants in the aggregate measures. This information may be obfuscated (i.e., revealing an anonymous but unique identifier may arguably release less personal information), but nonetheless, there is potential to tie the identifier back to the identity of the contributor. Future work will investigate how to further protect this identity information. Our approach computes an aggregate for a single snapshot of participants' context values. Other applications may need to allow participants to change their context values and have those updates reflected in the computed aggregate. Updating context values contained in an aggregate is non-trivial and is the focus of our ongoing work.

Figs. 6(a) and 7(a) show that the Smart Aggregate scheme often performs *worse* than even the Aggregate scheme. It turns out that receiving a larger number of smaller aggregates results in a higher *information diversity*, making it more likely that the recipient can merge aggregates. This points to another possible scheme, one in which a participant keeps and shares multiple smaller aggregates, sending a recipient aggregates only as large as required by the trust values. This also has potential benefits for updating context values, as updating within a smaller aggregate is likely to be easier.

In summary, this work is the first of its kind to use trust to obfuscate private context in mobile ubiquitous computing environments. This is feasible for applications that compute aggregates of shared local context and can tolerate a small error in that computation. Our schemes are particularly suited to cases where the application traffic is low and there is generally low trust mixed with a handful of highly trustworthy partners.

Acknowledgements

This work was supported in part by the NSF under grant CNS-1218232. Any findings, conclusions, or recommendations are those of the authors.

References

1. O. Abul, F. Bonchi, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *Proc. of ICDE*, pages 376–385, 2008.
2. D. Bickson, D. Dolev, G. Besman, and B. Pinkas. Peer-to-peer secure multi-party numerical computation. In *Proc. of P2P*, pages 257–266, 2008.
3. C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Zhu. Tools for privacy preserving distributed data mining. *ACM SIGKDD Explorations Newsletter*, 4(2):28–34, 2002.
4. C. Cornelius, A. Kapadia, D. Kotz, D. Peebles, M. Shin, and N. Triandopoulos. Anonysense: Privacy-aware people-centric sensing. In *Proc. of MobiSys*, pages 211–224, 2008.
5. C. Dwork. Differential privacy. In *Proc. of ICALP*, pages 1–12, 2006.
6. L. Eschenauer, V. Gligor, and J. Baras. On trust establishment in mobile ad-hoc networks. In *Proc. of SPW*, pages 47–66, 2004.
7. R. Ganti, N. Pham, Y.-E. Tsai, and T. Abdelzaher. PoolView: Stream privacy for grassroots participatory sensing. In *Proc. of SenSys*, pages 281–294, 2008.
8. E. Grim, C.-L. Fok, and C. Julien. Grapevine: Efficient situational awareness in pervasive computing environments. In *Proc. of PerCom WiP*, pages 475–478, 2012.
9. B. Hoh and M. Gruteser. Protecting location privacy through path confusion. In *Proc. of SECURECOMM*, pages 194–205, 2005.
10. T. Jiang and J. Baras. Ant-based adaptive trust evidence distribution in MANET. In *Proc. of ICDCS Workshops*, pages 588–593, 2004.
11. L. Kagal, T. Finin, and A. Joshi. Trust-based security in pervasive computing environments. *IEEE Computer*, 34(12):154–157, 2001.
12. A. Keränen, J. Ott, and T. Kärkkäinen. The ONE simulator for DTN protocol evaluation. In *Proc. of SIMUTools*, 2009.
13. K. Kursawe, G. Danezis, and M. Kohlweiss. Privacy-friendly aggregation for the smart-grid. In *Proc. of PETS*, pages 175–191, 2011.
14. J. Li, R. Li, and J. Kato. Future trust management framework for mobile ad hoc networks. *IEEE Communications*, 46(4):108–114, 2008.
15. Q. Li and G. Cao. Efficient and privacy-preserving data aggregation in mobile sensing. In *Proc. of ICNP*, pages 1–10, 2012.
16. Q. Li and G. Cao. Providing privacy-aware incentives for mobile sensing. In *Proc. of PerCom*, pages 76–84, 2013.
17. Z. Liu, A. Joy, and R. Thompson. A dynamic trust model for mobile ad hoc networks. In *Proc. of FTDCS*, pages 80–85, 2004.
18. C. Orlandi. Is multiparty computation any good in practice? In *Proc. of ICASSP*, pages 5848–5851, 2011.
19. N. Pelekis, A. Gkoulalas-Divanis, M. Vodas, D. Kopanaki, and Y. Theodoridis. Privacy-aware querying over sensitive trajectory data. In *Proc. of CIKM*, pages 895–904, 2011.
20. A. Pirzada and C. McDonald. Establishing trust in pure ad-hoc networks. In *Proc. of ACSW*, pages 47–54, 2004.
21. D. Quercia, S. Hailes, and L. Capra. B-Trust: Bayesian trust framework for pervasive computing. In *Proc. of iTrust*, pages 298–312, 2006.
22. M. Raya, R. Shokri, and J.-P. Hubaux. On the tradeoff between trust and privacy in wireless ad hoc networks. In *Proc. of WiSec*, pages 75–80, 2010.
23. J.-M. Seigneur and C. Jensen. Trading privacy for trust. In *Proc. of iTrust*, 2004.
24. R. Sheikh, B. Kumar, and D. Mishra. Privacy-preserving k-secure sum protocol. *Int'l. J. of Computer Science and Information Security*, 6(2):184–188, 2009.
25. J. Shi, R. Zhang, Y. Liu, and Y. Zhang. PriSense: Privacy-preserving data aggregation in people-centric urban sensing systems. In *Proc. of INFOCOM*, 2010.
26. A. Yao. Protocols for secure computations. In *Proc. of FOCS*, pages 160–164, 1982.