In Reputation We Believe: Query Processing in Mobile Ad-Hoc Networks*

Filip Perich Jeffrey Undercoffer Lalana Kagal

Anupam Joshi Timothy Finin Yelena Yesha

University of Maryland, Baltimore County
1000 Hilltop Circle, Baltimore, Maryland 21250
{fperic1, junder2, lkagal1, joshi, finin, yeyesha}@cs.umbc.edu

Abstract

Research on data management in mobile ad-hoc networks focuses on discovering sources and acquiring information. Mobile devices assume answers to be correct and do not verify the veracity of the information or the providers. This assumption is suitable for most client-server environments; however, peer-to-peer environments lack the intrinsic stability of "anchored" sources. In mobile ad-hoc networks, sources may provide faulty information, which can lead to incorrect conclusions. Consequently, devices need a mechanism to evaluate the integrity of their peers and the accuracy of peer provided information. To address this problem we propose a query processing model that relies on distributed trust and belief. Each device maintains and shares beliefs regarding the degree of trust it has for its peers – where trust is determined by experience and reputation. Additionally, each device associates a value indicating its belief in the accuracy of the information it holds. This knowledge is used by devices to determine the reliability of query responses. We implement our model in GloMoSim and provide experimental results for different combinations of trust and accuracy algorithms.

1. Introduction

Hand-held computing devices are becoming increasingly common and are used for a variety of tasks – ranging from booking airline tickets to finding directions to a local movie theater. These devices mostly adhere to the traditional client-server model in which they act as clients and access non-transient information on trusted servers. With the advent of ad-hoc computing technologies like Bluetooth, we believe that the scenario will soon morph into a peer-to-peer model where devices will communicate and collaborate with each other to produce contextual knowledge. In this model the wired infrastructure may not always be accessible, forcing a greater reliance on peer provided information. We can imagine one device asking another to recommend a good Chinese restaurant that is nearby or inquiring about the closest library.

Current research in wireless ad-hoc networks focuses on discovering sources and acquiring needed information, tacitly assuming that every obtained answer is correct. In a client server model, the server is "anchored" and a client can verify through several authentication and integrity schemes that the information came from the server, forcing accountability. In dynamic environments there are no such anchored or accountable servers. Furthermore, there are no schemes to protect a device from malicious peers deliberately providing incorrect answers to queries.

Recent query processing frameworks [5, 12] suggest that for a mobile device to provide a required information to its user, the device needs to pro-actively query its dynamic environment. Pro-activeness is a necessary feature for mobile ad-hoc networks as one cannot guarantee that a required information source is reachable when a user asks her question. A device should instead be able to anticipate future user questions via the use of profiles and pro-actively cache relevant answers obtained from its peers. The device must, however, determine and only cache trusted, reliable answers. Most mobile devices are not capable of such reasoning at this point. They also lack a "common sense" that we often employ

*This work was supported in part by NSF awards IIS 9875433 and 0209001, and DARPA contract F30602-00-2-0591

to decide the reliability of a source and information the source provides.

To fully realize the potential of the mobile ad-hoc paradigm we need a mechanism for mobile devices to evaluate autonomously peers and peer information.

Along with enabling devices to estimate their trust in other devices and the accuracy of the information they provide, we believe that the mechanism should also enable devices to detect and distinguish between *malicious peers*, which purposely provide incorrect information, *ignorant peers*, which are unable to guarantee a reliability level of provided information, and *uncooperative peers*, which have reliable information but refuse to make it available to other devices. This mechanism would implicitly create an *incentive model*, in which all devices must provide only reliable information and provide this information often, otherwise they risk losing the ability to communicate with other devices in the environment. This is because devices could stop interacting and forwarding traffic for their malicious and uncooperative peers.

As a step toward this mechanism, we define a query processing model based on *distributed belief* and *trust management*, where devices calculate trust in their peers and accuracy belief of obtained information. To mitigate the effects of malicious and "ill-informed" devices, our initial model categorizes peers as *reliable* and *unreliable*. In our model, the accuracy of an answer is a function of the trustworthiness of the information source and its belief in the accuracy of its answer. Devices assign trust to an information source based upon past experience and from the recommendations of those devices that it trusts.

We have simulated several belief and trust management schemes within the GloMoSim [14] simulation environment. We illustrate the usefulness of our model by way of experimental results, based upon a mobile ad-hoc network consisting of 50 devices where each device generates 100 different queries. We introduce untrustworthy devices into the network and are able to answer questions like: (i) When reputation and trust are at issue, what percentage of queries are rejected and hence are considered to be "unanswered"; (ii) What is the convergence period for the first or last device to determine which devices are honest and which are not; and (iii) How does the introduction of new devices into the environment influence data gathering in pervasive computing environments?

2. Related Work

Most recently data management community has been advocating the use of *profiles*, especially when dealing with pervasive systems and stream data, in order to precache answers for future user queries. In a seminal work, Cherniak *et al* [5] explore the use of profiles in the area of client/server based data recharging for mobile devices. As an alternative, Perich *et al* [12] propose a profile in terms of *beliefs*, *desires*, and *intentions*. These systems, however, do not address reliability of cached information in terms of accuracy of the answer. Consequently, devices could be caching wrong answers from unreliable devices.

The management of trust and belief has been well researched within the domain of software agents and artificial intelligence, including the Semantic Web. It has been, however, largely ignored by the rest of the research community due to its vast computing time and other resource constraints. Although the notion of provenance is quickly gaining popularity within the database community [4].

Trust and belief management models can be divided into two categories: mathematical and logical.

Jonker *et al* [10] propose a mathematical model for capturing trust in multi-agent systems. Their model consists of beliefs, and trust is a function of the values of these beliefs. The trust function is based on initial trust, experiences, and trust dynamics. The types of trust dynamics determine how past experiences affect the newly computed trust value. Richardson *et al* [13] present a mechanism for calculating the trustworthiness of users on the Semantic Web by developing a "web of trust" based on web algorithms like Google's PageRank [8]. In this approach, every user maintains trust values for a small set of users and uses the belief values of these users and her trust in the users to calculate her own beliefs. Abdul-Rahman *et al* [1] define a formal trust model based on trust and recommendations. Users store trust values for other users and ask trusted users for recommendations when dealing with unknown users. However, once a trust value is calculated it is not updated.

There is also a significant amount of work on developing logical trust models. Blaze *et al* [3] define trust management as creating policies and assigning credentials. They use a PolicyMaker engine for checking if users' credentials conform to policies before granting them access. Keynote [2] is designed along the same lines as PolicyMaker, however, it has been designed to be simpler, to provide more support for PKI, and to allow policies and credentials to be transported over insecure communication channels. Referee is a similar trust management system that is designed to facilitate security decisions for the web [6]. Kagal *et al* [11] also describe a policy based infrastructure for security and trust management in multi-agent systems and the Semantic Web. In this system, every entity has a policy that re-

flects its current binary trust values and exchanges them with other entities via speech acts.

For our approach, we chose to develop a mathematical model. First a mathematical model requires fewer computing resources than logical models, which require reasoning engines and a certificate verification. Additionally, unlike logical models that only describe conditions when devices are "trusted" enough to access a certain information, mathematical models can also be employed to represent answer accuracy and to handle situations when more than one answer is provided for a given query. Our work differs from other mathematical models in that we propose several trust learning schemes based on experience and recommendations, allow information sources to specify their trust in the information being provided, and use both kinds of values to compute belief. Most other schemes either provide trust learning algorithms based on experience or on recommendations but do not combine the two. They also ignore the believed accuracy of the information source, whereas we use it as a factor for rating the trustworthiness of a source.

3. Distributed Belief Model

Peer devices[1] are the information sources in pervasive computing environments. Because these environments are dynamic and their topologies change frequently, a mobile device needs a mechanism to evaluate the integrity of its peers and the accuracy of the information they offer. The role of this mechanism is twofold: It mitigates the danger presented by using faulty information for some computation. Next, it protects the environment from denial-of-service type attacks, where malicious or ignorant devices flood the environment with erroneous data. Such a mechanism must evaluate the integrity of a device and be able to at least categorize devices as *reliable* and *unreliable*.

A successful mechanism for evaluating device integrity and information accuracy must address the inherent limitations of mobile ad-hoc networks and of mobile devices. Due to power, memory and computation constraints, the mechanism should not assume that each device can maintain information about all devices in the environment and the information they can provide. The mechanism also should not assume that each device is reachable by any other device in the environment at any time due to the underlying network constraints and the mobility of devices. Additionally, devices may often be unable to connect to a wired infrastructure and thus the model should not assume that devices can off-load and share their states through wired nodes.

[1] We use the terms *peer* and *peer devices* interchangeably.

To address the problem, we propose a query-processing model that relies on a distributed belief. Our model does not rely on any wired infrastructure nor does it assume connectivity among all devices. Our model also does not assume that each device can maintain belief information about every other device or information the other devices can provide. Instead, our model makes only the following two assumptions:

Every device is able to assign an accuracy degree to any information the device provides to its peers. The accuracy degree represents the device's belief about the correctness of the information, which can range from *distrust* to *undecided* to *trust* value.

Every device maintains trust degrees about a subset of devices in the environment representing how much a device trusts the other devices for providing accurate answers to queries. A device, when asked, can provide its recommendation for some other device in question. Similar to accuracy degree, the recommendation can range from *distrust* to *undecided* to *trust* value.

In the remainder of this section, we define the necessary terminology of beliefs and belief functions that our model will operates over, and our model's protocol.

Definition 1 $A_D(i)$ *represents an* **accuracy degree** *in range from* -1.0 *to* 1.0 *suggested by device* D *for information* i *answering a given question.*

$A_D(i) = -1.0$ implies that device D absolutely disbelieves information i answers a given question. $A_D(i) = 1.0$ implies the device D is absolutely positive i answers the question, and $A_D(i) = 0$ means the device is undecided how i is answering the question.

Definition 2 $T_D(A)$ *is a* **trust degree**, *also in range from* -1.0 *(absolute distrust) to* 1.0 *(absolute trust), representing how much* D *believes a device* A *to be reliable and to provide accurate answers.*

In order to maintain a belief knowledge, each device must compute an initial trust for devices it has never interacted with before. Each device may also use this function to compute initial trust for devices it has interacted with in the past but for which it no longer keeps its trust degrees due to its limited memory resources.

Definition 3 α_D *represents an* **initial trust function** *a device* D *uses to compute an initial trust for new device* A*.* D *performs this function whenever it interacts with a device for which it does not have any prior information.*

We distinguish between *pessimistic*, *optimistic* and *undecided* initial trust functions. Applying the pessimistic function, a device does not trust any new peer by assigning it a trust degree of -1.0. In the *optimistic*

case, each new device is always trusted and is assigned a trust degree of 1.0 while in last case the initial trust degree is 0.

Definition 4 Δ_D *represents a* **trust learning function** *a device D employs to compute its future trust for a device A based on the current trust degree and past experiences.*

Once a querying device determines the final answer to its query, it uses this function to update its trust beliefs in other devices based on the devices' responses.

Each device uses the initial trust and trust learning functions to build and maintain its belief about other devices in its environment. Subsequently, when a device needs to answer a query for which it does not already have a cached answer, the device should only use answers from its trusted peers. We define trusted peers as:

Definition 5 *A device A is a* **trusted peer** *for device D if and only if $T_D(A) \geq \tau$ where τ is D's trust threshold.*

When a device receives a response from its peer, it should be able to use its belief in the peer's answer accuracy to weight the information the peer returns. Additionally, when a device queries its peers for a trust degree recommendation of some other peer, the device should also be able to pro-weight the returned recommendations. From belief network research concepts, we assume that all devices in the environment are equal *experts* in providing answers to queries as well as trust recommendation. Consequently, in our model, a device does not maintain trust degree per the combination of question domain and device but simply per device only:

Definition 6 $\otimes$ *represents a* **trust-weighting function** *a device D uses to pro-weight a suggested accuracy degree $A_X(i)$ of answer i provided by device X with its trust degree in device X, $T_D(X)$.*

In our model, we overcome the problem of unreliable information and unreliable information sources by requiring that each device ask at least n sources the same question. This, however, introduces two additional problems. First, the device must be able to combine all received responses and their suggested degrees of accuracy. Second, the device must be able to decide, if possible, on only one final answer, it believes is correct. To address these problems we define the following two abstract functions:

Definition 7 $\bigoplus$ *represents an* **accuracy-merging function** *a device D uses to combine weighted accuracy degrees for same answers from multiple peers.*

Definition 8 Λ *represents a* **final-answer function** *a device D uses to determine the final answer it believes answers its initial question based on all answers, suggested accuracy degrees, and its trust in the responding peers.*

3.1. Query Processing

We first describe the query processing steps from the perspective of the querying device and then from the perspective of the responding peer devices.

In our model, a querying device first collects responses from peers and then computes their trust degrees. We chose this approach because a device that first computes trust degrees and queries only its trusted peers may, in some situations, not be able to query any device. More importantly, the querying device will never interact with other, not yet trusted devices. Consequently, the querying device cannot learn and adjust its trust degree about other peers in its vicinity it does not already trust, thus limiting the set of potential information sources.

3.1.1. Information Source Discovery

In our model, when a device needs to obtain an answer for a query, it first attempts to discover which of its peers may have the necessary answer. The device does so by evaluating its cache of advertisements received from its peers and by broadcasting a source discovery request messages to its peers up to n-hop away. While n is a tunable parameter, we only used $n = 2$ in order to prevent message flooding. The discovery message $discovery(ID_Q, Q, nonce)$ consists of the device's identity ID_Q, the question Q, and a nonce for differentiating it from other discovery messages sent by this device. A device may sent out the discovery messages more than once based on the responses it receives from its peers.

3.1.2. Information Advertisement

When a cooperating, non-malicious peer receives a source discovery, it checks its cache to find an answer matching the question. If the peer has a cached answer, it will respond by sending an advertisement message $advertisement(ID_S, Q, nonce)$. ID_S is the identifier of the device, *e.g.* its MAC address and Q is the question the peer can answer. Optionally, a device may proactively broadcast bulk advertisements at random intervals.

3.1.3. Querying Peers

After receiving responses from its peers, the querying device evaluates all advertisements in its cache in order to determine possible sources for its query. If a device is unable to discover a sufficient number of information sources that could provide answer to its question, the device simply broadcasts the question to all peers in its vicinity, again up to n hops away. If, however, the device is able to collect some information sources, the device sends a query to only those peers by constructing one query message $query(ID_Q, ID_S, Q, nonce)$ at a time.

3.1.4. Collecting Answers

When a cooperating, non-malicious peer receives a query message, and has a matching answer, it will respond with a message in the form of $answer(ID_S, A, accuracy, nonce)$. The message includes the identity of the answering peer together with the device's proposed answer and the accuracy degree of the answer. The querying device temporarily caches the response while waiting for additional answers from other peers.

3.1.5. Recommendation Request

When a device receives an answer from its peer ID_S, it calculates the peer's *trustworthiness* by looking up its trust beliefs. If the device is unable to determine if the answering peer is a *trusted peer*, the device may initiate a recommendation session for the answering peer.

In our model, the device can proceed in two ways: The device can either ask only those devices who it believes are its *trusted peers* or the device can ask anyone in n-hop distance for recommendations about the answering device. The querying device ID_Q does so by sending out a recommendation request message $recommendation_{request}(ID_Q, ID_R, ID_S)$ to some remote peer ID_R with the identity of the answering peer.

3.1.6. Recommendation Response

When a cooperating, non-malicious peer receives a recommendation request, it looks up its trust beliefs to determine if the querying device ID_Q is one of its *trusted peers*. If this is the case, the device responds with its trust degree in ID_S by sending a recommendation responses message $recommendation_{response}(ID_R, ID_S, T_R(S))$. The querying device then stores the recommendation response in its cache as additional information for detecting the current and future trust degree of the answering device.

3.1.7. Calculating Final Answer

Once a device receives responses from all peers it queried or once a pre-set session time period ends, the querying device attempts to decide on the final, correct answer. A device determines the final answer only if it received an answer from at least one trusted peer. For every different answer value it received, the device computes a combined accuracy degree for the particular value based on the suggested accuracy degrees of the information sources and their related trust degrees. Formally, given a set S of answer tuples (i, D, $A_D(i)$), where i is the suggested answer and $A_D(i)$ is the suggested accuracy by device D, the combined accuracy degree is:

$$A_{combined}(i) = \bigoplus_{S} \Big(A_D(i) \otimes T_Q(D) \Big) \qquad (1)$$

If all trusted devices provide the same answer to the original query, the querying device will simply use that answer as the final value if its combined accuracy degree is above a certain threshold τ, similar to a threshold concept for a *trusted peer*. In some cases, and a primary point of this exercise, the querying device may, however, receive multiple conflicting answers. To address this problem, the querying device can apply two different techniques:

The querying device may accept an answer only if precisely one of the suggested answers has a combined accuracy level above τ. We refer to this technique as **only-one** answer. This is a pessimistic approach as the querying device will not cache an answer, if there is more than one answer above the threshold or if there are no answers above the threshold. At the same time, this approach will limit the amount of uncertain or distrusted data kept in the cache.

Alternatively, a device may employ a more optimistic technique that considers the possibility of a question having multiple valid answers, *e.g.* the list of Chinese restaurants in a given location. In this case, the device will choose the answer with the highest combined accuracy degree above τ. If there are multiple answers with the highest accuracy degree, the device will randomly choose one. We refer to this approach as the **highest-one** answer.

3.1.8. Updating Trust Belief

If the querying device is able to determine its final answer, it uses the answer to evaluate interaction experiences it had with the answering peers. The experiences can range from *negative* to *undecided* to *positive*. We list the experience types in Table 1.

When a querying devices has either a positive or negative experience with any answering device, it should update its trust degree for that device for future interactions. In our model, we adapt two types of trust learning functions as classified in [10]. The first category of trust learning functions employs all history information for predicting a future trust degree of a device. The other category employs only the current experience and the current trust degree. The querying device then either increases or decreases the trust degree using slow, fast or exponential steps.

3.1.9. Answering Peers

Thus far, we have detailed the steps from the point of view of the querying device. We now describe the steps taken by answering peers.

In order for a cooperating, non-malicious device to answer a query asked by its peer, the device must first have a matching answer in its cache or the device must be able to produce the answer using some other means.

The device will return an answer only if the querying peer is one of its *trusted peers*. The device evaluates its beliefs to determine if its local trust degree of the querying peer is above the trust threshold τ. Otherwise, the device first initiates a recommendation session, described above, and computes a combined recommended trust degree. Given a set R of recommendation tuples (Q, D, $T_D(Q)$), where Q is the querying peer, D is the recommending peer with a suggested trust degree $T_D(Q)$, the combined recommendation trust degree is computed by the answering device A as:

$$R_{combined}(Q) = \bigoplus_{R} \Big(T_D(Q) \otimes T_A(Q) \Big) \qquad (2)$$

The device then sends back an answer to the querying device only when $T_A(Q) > \tau$ or $R_{combined}(Q) > \tau$. In other words, for a device to return an answer, either the device explicitly trusts the querying peer or its trusted peers recommended the device to do so.

	$A_D(i) < 0$	$A_D(i) \geq 0$
$i = a$	negative experience	positive experience
$i \neq a$	undecided	negative experience

Table 1. Querying device interaction experience with a device D based on a final trusted answer a when D suggested i as the answer with accuracy degree $A_D(i)$.

Spatial Dimensions	150x150 m^2
Simulation Period	50 min
Simulation KB	800 data/answers, 800 questions
Mobile Devices	50
Mobility Pattern	Random Way-Point, 5 s waiting period, speed 1-5 ms^{-1}
Routing Protocol	AODV
Flooding Range	2 hops
Tx Range	$25m$
Tx Throughput	$2Mbps$
Device's Cache Size	250 kB, 50% of simulation KB
Cache Replacement	Trusted-Based LRU, victim is the least used among least accurate data
Device's Initial KB	100 questions, 100 answers not matching initial questions
Device's Initial Trust	3-5 other devices

Table 2. Simulation Environment

4. Experimental Results

We have implemented the belief-driven query processing framework as part of the GloMoSim simulator [14]. GloMoSim is a scalable simulation environment for wireless and wired networks with support for mobility. It is designed using a discrete event simulation capability provided by Parsec. Table 2 summarizes our simulation environment. In order to allow devices employing pessimistic trust approaches, where a trust in other devices can only decrease, we randomly assigned every device to have an initial high trust above τ for 3 to 5 other devices in the environment. While this is not necessary for optimistic or undecided approaches, we used the same initial assignments in order to be able to compare them.

4.1. Experimental Parameters

To study the performance of a distributed-belief query processing model, we have defined different approaches for each function from Section 3. In our querying simulator we have parametrized 30 different aspects of the model; however, for this paper we are interested in only the parameters listed in Table 3. To simplify the experiments, we assume that a *dishonest* device is always dishonest. That is a dishonest device in our environment never provides a correct answer because it is malicious or not willing to cooperate. On the other hand an honest

Dishonesty Level	Ratio of malicious nodes
Trust Threshold τ	0.75
Initial Trust:	
Positive α_+	initial trust $T = 1.0$
Negative α_-	$T = -1.0$
Undecided α_0	$T = 0$
Trust Learning:	
Blindly positive Δ_{++}	$T_Q(S) = 1.0$ after 4 / 10 positive experiences
Blindly negative Δ_{--}	$T_Q(S) = -1.0$ after 4 / 10
Fast positive Δ_{f+}	$T_Q(S) += (+0.5 \mid -0.2)$
Fast negative Δ_{f-}	$T_Q(S) += (+0.2 \mid -0.5)$
Balanced fast Δ_f	$T_Q(S) += (+0.5 \mid -0.5)$
Balanced slow Δ_s	$T_Q(S) += (+0.2 \mid -0.2)$
Exponential Δ_{exp}	$T_Q(S) = \frac{1}{2} T_Q(S) + \frac{1}{2}$
Trust Weighting:	
$\otimes_x$	$x * y$
$\otimes_{cos}$	$y \leq 0$: 0 $y > 0$: $x\left(1 - cos\left(\frac{y\pi}{2}\right)\right)$
Accuracy Merging:	
Minimum $\bigoplus_{min}$	least suggested accuracy
Maximum $\bigoplus_{max}$	highest suggested accuracy
Average $\bigoplus_{avg}$	average accuracy
Final Answer:	
HO	highest-one
Distrust Convergence	Time for honest devices to discover all dishonest devices

Table 3. Experimental Parameters

device is either a reliable device or at least an ignorant one.

4.2. Answer Accuracy vs. Trust Functions

We measured how different initial trust and trust learning strategies improve the number of correctly answered queries. We varied the *dishonesty level* from 0% to 100%. When the *dishonesty level* was 0% every device attempted to provide only accurate information, based on its beliefs. With an increasing level of dishonesty there were proportionally more devices that were always malicious. Due to the size constraints of this paper, we report only on the results for an initially optimistic trust function α_+. The results are shown in Figure 1.

We note that the combination of α_+ and Δ_{++} represented a scenario where a device did not use its trust and accuracy beliefs for query processing. Since Δ_{++} can only increase a trust degree and by default all unknown devices are trusted, a querying device could never change the trust degree for its peers, *i.e.* decrease the degrees. Therefore, this combination served as a baseline for comparison with other trust learning functions.

Δ_s and Δ_{f+} learning functions had similar performances. This was also true for Δ_f and Δ_{f-} functions. This was due to the fact that in the case of initial optimistic trust function all unknown devices were by default trusted. For honest devices the trust did not have to decrease while for dishonest devices the trust decreased using same steps for each pair of functions. Δ_{exp} learning function also behaved similarly because it used even larger steps to decrease a trust for any dishonest device in the environment.

We do not show results for undecided and pessimistic initial trust functions. Since α_+ assigns an absolute trust to any unknown peer, the querying device was able to obtain always more answers, correct or incorrect, than when using α_- or α_0. This holds even for less reliable environments. In fact when the trust learning function employs small steps to decrease a trust degree of a peer, the querying device was able to obtain answer for as many question as when everyone in the vicinity was honest. This was, however, at a cost of an increased number of wrong answers the device accepted as correct ones. While the trust learning functions with a high trust decreasing step were able to somewhat reflect the dishonesty of the environment by accepting less answers, they still had several times worser performance than when the initial trust function was not optimistic.

We found that the initial trust value a device assigned to an unknown peer greatly affected the final trust value for that peer. The final trust degree in turn affected the accuracy of results the device obtained for its queries. The higher the initial trust value was the more often a device was able to obtain an answer to its query. However, the device frequently accepted incorrect answers as it was unable to correctly estimate the real trust value for malicious devices. On the other hand, devices using more *pessimistic* or *undecided* initial approaches accepted fewer number of suggested answers, sometimes including incorrect responses. However, the ratio of correct versus incorrect answers was significantly higher for the latter approaches.

4.3. Distrust Convergence vs. Dishonesty Level

We also measured the effects of the dishonesty level on the distrust convergence period. The convergence period represented the time from the beginning of the simulation until all honest devices in the environment were able to detect unreliable sources. Figure 2 depicts the distrust convergence periods for environments with

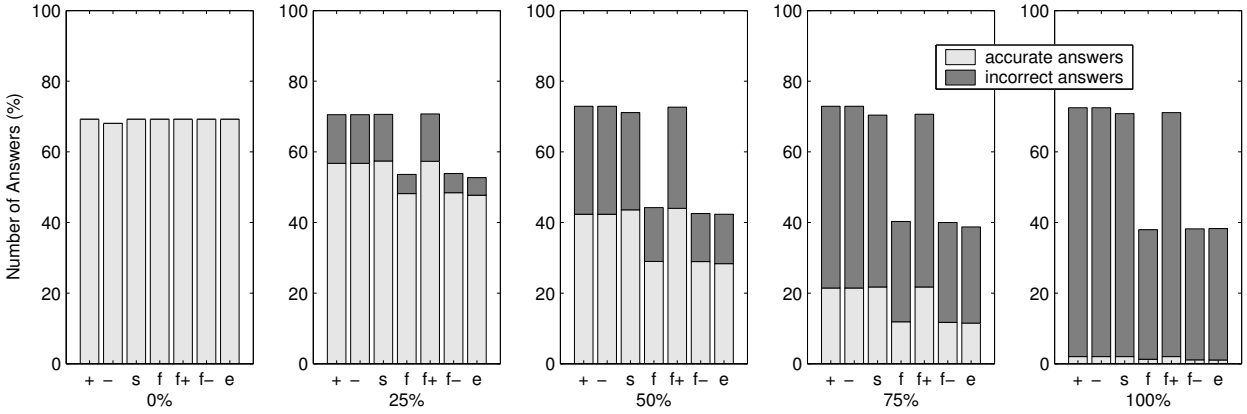

Figure 1. The effects of trust learning functions with an initial optimistic trust for environments with varying level of dishonesty. The results are shown for Δ_{++}, Δ_{--}, Δ_s, Δ_f, Δ_{f+}, Δ_{f-}, and Δ_{exp} learning functions.

25%, 50% and 100% dishonesty levels. The graphs display values for the same combination of initial trust and trust learning functions from Experiment 4.2.

We found that the trust convergence period was highly dependent on the dishonesty level. Since a device was unable to obtain reliable recommendations and answers in highly dishonest environments, a device was unable to judge the honesty of its peers and for non-optimistic approach this resulted in a final undecided trust.

The *blindly negative* trust learning function had the most visible effect on updating a trust belief. Since this function could only decrease the amount of device's trust in its peer, the function always caused the least trust among all possible combinations in every simulation run. For an *undecided* initial trust, this heuristic was quickly approaching an absolute distrust of -1.0 for dishonest devices in most environments. We note that for environments where all devices were dishonest, this heuristics stabilized at 0, *i.e.* at undecided trust degree, as it could not generate enough positive or negative experiences. A similar rule applied to other combinations. In general, the less evidence a device could obtain for or against its peer, the less likely the device was able to make a decision about the peer's dishonesty.

5. Conclusions and Future Work

As the mobile ad-hoc network paradigm shifts from the traditional client-server approach to a peer-to-peer model, devices need to be able to determine how much they can trust their peers and the information the peers provide. To address the issue, we proposed a query processing model for these pervasive computing environments based on *distributed belief* and *trust management*.

To mitigate the effects of malicious and "ill-informed" devices, our model categorizes peer devices as *reliable* (trusted) and *unreliable*. Using our model devices compute trust in their peers and the accuracy degree of offered information. The accuracy of information is a function of the trustworthiness of the information source and its belief in the accuracy of its data. Devices assign trust to an information source based upon past experience and from the recommendations of those devices that it trusts. We have shown the effects of different heuristics for estimating device trust and accuracy of information in environments with varying level of dishonesty through a simulation.

For our work we have assumed that the integrity of messages sent in the wireless network can be guaranteed. However, for our future work we will investigate the possibility of integrating our model with an intrusion detection and monitoring scheme, such as [9], that allows the detection of unreliable devices that tamper with packets they agree to forward. We also propose to investigate the notion of estimating trust based on domains [7] instead of treating all informations as part of one domain and treating all devices as equivalent domain experts.

References

[1] A. Abdul-Rahman and S. Hailes. A Distributed Trust Model. In *Workshop on New Security Paradigms*, 2003.

[2] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis. The KeyNote Trust Management System Version. Internet RFC 2704, 1999.

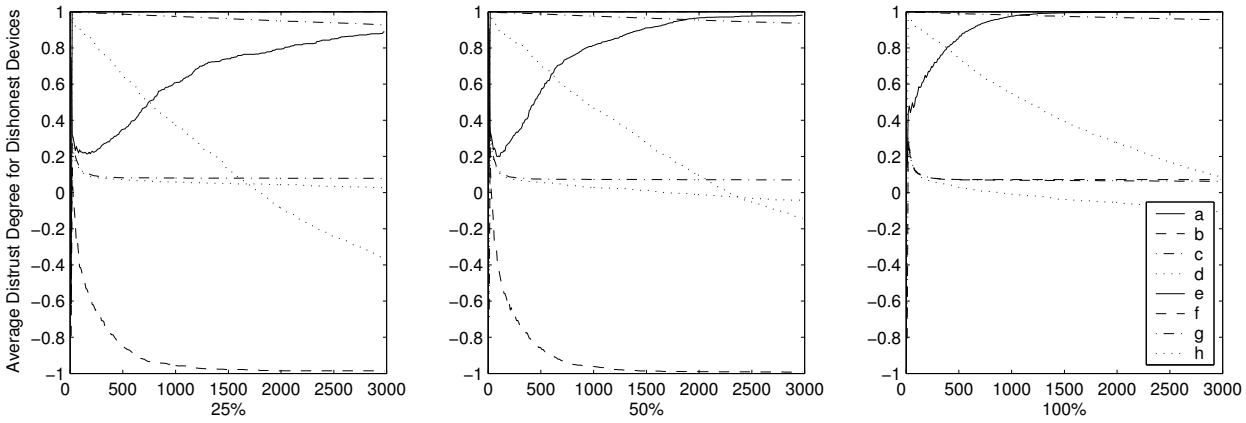

Figure 2. Average distrust convergence period in seconds for varying levels of dishonesty. The results are for (a) Δ_{++}, (b) Δ_{--}, (c) Δ_s, and (d) Δ_f with α_+ and for the same functions using α_0 for results (e–h), respectively.

[3] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis. The Role of Trust Management in Distributed Systems. *Secure Internet Programming*, 1999.

[4] P. Buneman, S. Khanna, and W.-C. Tan. Why and Where: A Characterization of Data Provenance. In *International Conference on Database Theory*, 2001.

[5] M. Cherniak, M. Franklin, and S. Zdonik. Expressing User Profiles for Data Recharging. *IEEE Personal Communications*, July 2001.

[6] Y.-H. Chu, J. Feigenbaum, B. LaMacchia, P. Resnick, and M. Strauss. REFEREE: Trust management for Web Applications. *Computer Networks and ISDN Systems*, 1997.

[7] L. Ding, L. Zhou, and T. Finin. Trust Based Knowledge Outsourcing for Semantic Web Agents. In *IEEE/WIC International Conference on Web Intelligence (WI 2003)*, 2003.

[8] Google. PageRank. `http://www.google.com/technology`.

[9] R. Janakiraman, M. Waldvogel, and Q. Zhang. Indra: A Peer-to-Peer Approach to Network Intrusion Detection and Prevention. In *Proceedings of IEEE WETICE 2003*, June 2003.

[10] C. M. Jonker and J. Treur. Formal Analysis of Models for the Dynamics of Trust Based on Experiences. In *MAAMAW-99*, 1999.

[11] L. Kagal, T. Finin, and A. Joshi. A Policy Based Approach to Security for the Semantic Web. In *ISWC2003*, 2003.

[12] F. Perich, A. Joshi, T. Finin, and Y. Yesha. On Data Management in Pervasive Computing Environments. *IEEE Transactions on Knowledge and Data Engineering*, May 2004. Accepted for publication.

[13] M. Richardson, R. Agrawal, and P. Domingos. Trust Management for the Semantic Web. In *2nd International Semantic Web Conference (ISWC2003)*, 2003.

[14] X. Zeng, R. Bagrodia, and M. Gerla. GloMoSim: A Library for Parallel Simulation of Large-Scale Wireless Networks. In *Workshop on Parallel and Distributed Simulation*, 1998.