TRULLO - local trust bootstrapping for ubiquitous devices

Daniele Quercia, Stephen Hailes, Licia Capra
Department of Computer Science, University College London, London, WC1E 6BT, UK.
{D.Quercia, S.Hailes, L.Capra}@cs.ucl.ac.uk

***Abstract*—Portable devices have become sufficiently powerful that it is easy to create, disseminate, and access digital content using them. The volume of such content is growing rapidly and, from the perspective of each user, relevance and quality are key. In the absence of a central authority, distributed trust models may be intended as a mechanism for filtering information by keeping track of who disseminates relevant content and who does not. This mechanism is capable of doing so across various contexts (content categories). However, a problem with such an approach is that, as the number of content categories increases, so does the sparseness of the trust relation; thus the setting of initial trust values in the absence of direct experience becomes problematic. The most sophisticated of the current solutions employ pre-defined context ontologies, using which initial trust in a given context is set based on that already held in similar contexts. However, universally accepted (and time invariant) ontologies are rarely found in practice. For this reason, we propose a mechanism called TRULLO (TRUst bootstrapping by Latently Lifting cOntext) that assigns initial trust values by exploiting statistical properties of the ratings of its user's past experiences. We evaluate the effectiveness of TRULLO by simulating its use on ubiquitous devices in an informal antique market setting. We also evaluate the computational cost of a J2ME implementation of TRULLO on a standard mobile phone.**

I. Introduction

Using their portable devices, users may produce, disseminate, and access digital content (e.g., news, photos, videos). In so doing, users may be engaged in everything from urban planning to creative expression [1], [2].

Digital content may span several categories (e.g., news may be about 'council politics' or about 'local events'), and only part of it is of high quality (defined subjectively - as interesting, relevant, accurate, etc.). To benefit from a wide variety of content, users should be able preferentially to select that fraction of content that is in their categories of interest and that is of high quality. Since the calculation of quality is subjective, and since a centralized approach may not scale with growing number of users, users should do so in a decentralized setting. As a consequence, each user may run a distributed trust model [3], [4] - a software agent that keeps track of who has disseminated quality content and who has not. Given that there are a myriad of (portable) devices participating in the exchange of content in a broad variety of categories, one problem arises. Before trust models can be used effectively, a critical mass of reputation information must be collected; the requisite quantity of which dramatically increases with the number of producers and of content categories (contexts[1]). That problem may be alleviated by effectively setting *initial* trust values.

Current research literature suggests that user A may assign its initial trust in user B:
(1) as a constant representing A's initial disposition to trust [6];
(2) based on other users' recommendations about B [5];
(3) close to A's trust in B in a similar and known context [7].

We propose a solution complementary to those existing approaches and will then discuss which approach to use in which case (Section V). In so doing, we make the following main contributions:

- Designing a novel mechanism, named TRULLO (TRUst bootstrapping by Latently Lifting cOntext), that, in contrast to existing approaches (Section VI), neither fixes arbitrary initial trust values, nor relies on collecting recommendations, nor assumes a universally accepted ontology. Instead, it statistically analyzes the ratings of its user's past experiences to bootstrap unknown trust values. Section III-A introduces the key ideas on how TRULLO bootstraps trust and Section III-B then proposes an algorithm implementing those ideas.
- Evaluating the effectiveness of TRULLO at predicting trust (Section IV-A) by simulating its use on ubiquitous devices in informal antique markets (described in the next section).
- Evaluating the computational cost of a J2ME implementation of TRULLO on a Nokia 3230 mobile (Section IV-B).

II. Running Example

Throughout the paper, we consider an application of portable devices in informal antique markets. Given that those markets are huge, one major problem is that visitors cannot see prices of everything in which they are interested. To solve that problem, stall holders and visitors could use their ubiquitous devices, such as their mobile phones and PDAs. Stall holders may, for example, disseminate ads of the items they are selling through their portable devices, and visitors may collect those ads upon which they may compare prices and consequently decide which stalls to visit. Of course, visitors would have time to read only a subset of the ads. As such, they would wish to receive ads that are in the categories (contexts) of

[1] In the literature of trust management, content categories are called contexts (e.g., see [5])

their interest (e.g., ads about cabinets, chairs, and desks) and that come from *trustworthy* sellers (e.g., sellers who are known to inflate prices, or to spam visitors with irrelevant ads, may be deemed untrustworthy).

We consider that a visitor selects trustworthy sellers as follows:

(1) The visitor receives ads from sellers on her portable device.

(2) Ads are rated. The visitor does not have explicitly to rate ads: monitoring the time the visitor spent reading an ad, or being physically located in front of the stall that is the subject of an ad, might serve as surrogates for rating.

(3) On input of ads' ratings, the trust model updates its trust values for sellers and arranges those values across contexts. By context, we simply mean a textual description of an ad's category (e.g., 'Roman coins').

(4) Based on the resulting trust values across contexts, an instance of TRULLO running on the device extracts statistically relevant information (trust features) through a technique called Singular Value Decomposition [8] (to which we will refer as SVD and 'the decomposition' interchangeably). By combing those features, TRULLO then bootstraps unknown trust values. The device then select trustworthy sellers and consequently shows its user only those ads she would find relevant.

This work focuses on how TRULLO extracts trust features upon which it then bootstraps unknown trust values. Before describing how TRULLO does so, we spell out our assumptions:

- A mechanism to distinguish one context from another is given. This mechanism does not need to describe the relationships among all possible contexts (as does an ontology tree); rather, it simply needs to distinguish contexts. An example of such a mechanism is a directed graph, whose nodes are contexts and whose edges are change conditions [9], [10]. A qualitative context description (e.g., 'Roman coins') may be then mapped into a node of the graph by considering that change conditions specify whether certain keywords are present in the description. This is a particular example of how to distinguish contexts, but each device that runs TRULLO is free to choose any other. This is an advantage of TRULLO's bootstrapping compared to bootstrapping from recommendations: while the latter requires that all recommenders and recipients of recommendations differentiate contexts in the same way, TRULLO simply expects each device to have *its own way* of doing so.
- TRULLO is given a way to distinguish one seller from another. That is, it must be able to *uniquely* identify sellers (e.g., by binding together a unique public key with each seller). If identities are not unique, TRULLO may suffer from sybil attacks [11], in which a malicious seller takes on multiple identities and pretends to be multiple, distinct sellers. In the absence of a central authority, TRULLO may statistically guarantee seller-identity bindings by using a mechanism similar to SybilGuard [12].

$$D= \begin{array}{cc} & \begin{array}{cccc} & & \text{contexts} \rightarrow & \\ c_w & c_x & c_y & c_z \end{array} \\ \text{sellers} \downarrow \begin{array}{c} B \\ C \\ D \\ E \end{array} & \begin{pmatrix} \mathbf{0.67} & \underline{0.71} & \mathbf{0.49} & \mathbf{0.97} \\ 0.37 & 0.92 & 0.8 & 0.82 \\ 0.3 & 0.9 & 0.21 & 0.42 \\ 0.23 & 0.98 & 0.24 & 0.32 \end{pmatrix} \end{array}$$

$$E= \begin{array}{cc} & \text{features} \rightarrow \\ \text{sellers} \downarrow & \begin{pmatrix} 0.5629 & -0.5690 & 0.5701 & -0.1853 \\ 0.5958 & -0.2133 & -0.7673 & 0.1043 \\ 0.4071 & 0.4534 & 0.2904 & 0.7378 \\ 0.4030 & 0.6520 & 0.0446 & -0.6406 \end{pmatrix} \end{array}$$

$$F= \begin{array}{cc} & \begin{array}{cccc} & & \text{features} \rightarrow & \\ f_1 & f_2 & f_3 & f_4 \end{array} \\ \text{features} \downarrow & \begin{pmatrix} \mathbf{2.4972} & 0 & 0 & 0 \\ 0 & \mathbf{0.6136} & 0 & 0 \\ 0 & 0 & \mathbf{0.3330} & 0 \\ 0 & 0 & 0 & 0.0169 \end{pmatrix} \end{array}$$

$$G^T= \begin{array}{cc} & \begin{array}{cccc} & & \text{contexts} \rightarrow & \\ c_w & c_x & c_y & c_z \end{array} \\ \text{features} \downarrow \begin{array}{c} f_1 \\ f_2 \\ f_3 \\ f_4 \end{array} & \begin{pmatrix} \mathbf{0.3253} & -\underline{0.2838} & \mathbf{0.5869} & \mathbf{-0.6849} \\ \mathbf{0.6844} & \underline{0.7282} & \mathbf{0.0118} & \mathbf{0.0334} \\ \mathbf{0.3743} & -\underline{0.3222} & \mathbf{-0.7892} & \mathbf{-0.3650} \\ 0.5344 & -0.5341 & 0.1803 & 0.6297 \end{pmatrix} \end{array}$$

Fig. 1. Singular value decomposition (SVD). On a visitor's device, TRULLO gathers known trust values in a seller-by-context matrix D, upon which it applies the decomposition. Based on the three resulting matrices, it then reconstructs the trust value to be bootstrapped. The resulting matrices are: seller-by-feature matrix E (trust in sellers across features), diagonal feature matrix F (feature contribution to trust assessment), and transposed context-by-feature matrix G^T (feature relevance across contexts). To ease illustration in Section III-B, some elements are in bold or underlined.

III. TRULLO Bootstrapping Model

Based on those assumptions, to bootstrap unknown trust values, TRULLO applies the decomposition. We consider the rationale for using the decomposition first (Section III-A), and then look at how we will apply it to the problem at hand (Section III-B).

A. Singular Value Decomposition

Consider a buyer (Alice) and its (n_s x n_c) trust value matrix D in which n_s rows represent the sellers from whom Alice has previously received ads in n_c contexts. For example, each element D_{iy} represents Alice's trust value in Bob (the i^{th} seller) for sending ads about 'Roman coins' (the y^{th} context). Applying SVD on the matrix D means decomposing it into three other matrices (see Figure 1):

$$D_{n_s \times n_c} = E_{n_s \times n_f} \cdot F_{n_f \times n_f} \cdot G^T_{n_f \times n_c},$$

where:

- Each E's i^{th} row represents a set of Alice's trust values in the i^{th} seller across *features* - for example, E_{ik} represents Alice's trust value in Bob (the i^{th} seller) in the k^{th} feature.
- F's diagonal contains all features' contributions to trust assessment arranged in *descending* order[2] - for example, F_{kk} expresses the extent to which the k^{th} feature impacts on Alice's assessment of trust.
- Each G^T's k^{th} row reflects the relevances of the k^{th} feature across contexts - for example, G^T_{ky} represents the

[2]By definition, the number of feature is $n_f = min\{n_s, n_c\}$ [8]; however, in most applications $n_s > n_c$ and thus $n_f = n_c$.

relevance of the k^{th} feature in the y^{th} context 'Roman coins'.

Assume, for the sake of argument, that we do not know the trust value for the i^{th} seller in the y^{th} context. From the three decomposed matrices, we may initially estimate that value as the *weighted* combination of trust values in the i^{th} seller across *all* features. For each k^{th} feature, the weighting factors are two: the feature's relevance in the y^{th} context (G^T_{ky}), and the feature's influence on trust assessments (F_{kk}). For example, Alice's initial trust value in Bob for sending ads of 'Roman coins' equals the *weighted* sum of Alice's trust values in Bob in all features. Each feature is weighted according to its relevance in the context 'Roman coins' and to its general influence when assessing trust. Importantly, we do not need to explicitly define features: from trust values, the decomposition extracts statistically relevant information (features) and assigns, to each of the features, a number corresponding to its statistical relevance. TRULLO then latently (no context ontology required) bootstraps trust.

B. Bootstrapping Steps of TRULLO

Knowing how the decomposition can be applied to reconstruct unknown trust values, we are now ready to devise the bootstrapping steps of TRULLO. To do so, we consider the case in which TRULLO running on A's device has to bootstrap a trust value t for B in context c_x ('Roman coins'). To bootstrap t, TRULLO carries out the following steps:

Step 1. Determine the contexts in which A has previously received ads from B. For example, consider a situation in which those contexts are c_w, c_y, and c_z.

Step 2. Determine the sellers from whom A has received ads in the previously identified contexts plus the bootstrapping one (i.e., in c_w, c_x, c_y, and c_z). For example, those sellers may be C, D, and E. Overall, TRULLO considers n_s sellers in n_c contexts. In this example, $n_s = n_c = 4$; in general, $n_s \neq n_c$. In the extreme case of TRULLO having limited information (i.e., either n_s or n_c equals 1), TRULLO bootstraps t depending on the following cases:

1) $n_s = 1$ and $n_c = 1$ (A has received no ad - TRULLO has just been installed on A). TRULLO has no information, but has at least one default recommender: its user. So TRULLO bootstraps t according to its user's risk attitude, which existing mechanisms (e.g., [13]) may elicit.
2) $n_s = 1$ and $n_c > 1$ (A has received ads only from B and has done so in n_c contexts). TRULLO has to formulate a hypothesis of "how trustworthy B is" in a new context. It does so by setting t as the median of its known trust values in B. We choose the median instead of, for example, the mean because the median gives less weight to outliers [14].
3) $n_s > 1$ and $n_c = 1$ (A has received ads from n_s sellers in the bootstrapping context, but not from B). TRULLO bootstraps t based on social investigations into how humans set initial trust in the real world. In the most-cited model of generalized trust by Hardin [15], initial trust is based on trusting disposition, which, in turn, is based on accumulated experiences *within* a particular context. In this vein, TRULLO bootstraps t as the median of the trust values for the sellers known in the bootstrapping context, which may be interpreted as the 'typical behavior' for sellers in that context.

In the above cases (very limited information), a simple median may appear to be a reasonable choice. However, in the presence of enough information ($n_s > 1$ and $n_c > 1$), Section IV-A3 on "Simulation Results" demonstrates that the results obtained by extracting features are by far more accurate than simply using the median (even for small n_s and n_c). The next steps detail how TRULLO extract those features.

Step 3. Populate a matrix $D_{n_s \times n_c}$. In our example, this matrix contains Alice's trust values in the four sellers in all contexts. D is, however, incomplete - A's trust for B in c_x is missing. As SVD does not compute on incomplete matrices, we insert the row's average to fill the gap. Given that the row's average is an arbitrary value, step 6 will not use its decomposition, as Troyanskaya *et al.* [16] suggested. For example, in Figure 1, TRULLO populates D and assigns the row's average 0.71 to the the value to be bootstrapped (underlined).

Step 4. Apply SVD on D thus obtaining $D = E \cdot F \cdot G^T$ (as Figure 1 shows).

Step 5. From the three resulting matrices, extract the elements F_{kk} and G^T_{kj}, $\forall k \in [1, m]$ and $j \in [1, n_c]$, where: F_{kk} is the k^{th} feature's influence on trust assessment; G^T_{kj} is the k^{th} feature's relevance in the j^{th} context; and m is the number of relevant features[3]. In Figure 1, being $m = 3$, TRULLO extracts the elements in bold from F and G^T, and also those underlined from the latter.

Step 6. For each l^{th} context in which A has received ads from B (in our case, $l \in \{w, y, z\}$), regress A's trust in B (in the i^{th} seller) in the l^{th} context against the relevances in that context of all m features: $D_{il} = b_1 F_{11} G^T_{1l} + \ldots + b_m F_{mm} G^T_{ml}$. As we neither regress D_{ix} nor consider any feature in the bootstrapping context (G^T's column corresponding to c_x), the row's average in step 3 has little influence in the regression. That regression then results in m correlation coefficients $\{b_1, \ldots, b_m\}$. For example, in Figure 1, TRULLO regresses the

[3]The number of features that the decomposition extracts from a matrix ($n_s \times n_c$) is $min\{n_s, n_c\}$. The number of contexts A knows is $(n_c - 1)$. Therefore, the number m of relevant features is $min\{n_s, n_c, (n_c - 1)\}$, i.e., $min\{n_s, (n_c - 1)\}$.

elements in bold of D against those of F and G^T, and obtains the following correlation coefficients: $\{0.5886, -0.4578, 0.5734\}$.

Step 7. Having the correlation coefficients, the elements in bold of F, and those underlined of G^T, finally compute $t = D_{ix} = b_1 F_{11} G^T_{1x} + \ldots + b_m F_{mm} G^T_{mx}$. In words, Alice's trust in B in context c_x equals the *weighted* combination of Alice's trust values in B across features $(b_1, \ldots, b_k, \ldots, b_m)$. The weighting factors for each k^{th} feature are the feature's influence on trust assessment (F_{kk}) and the feature's relevance in the x^{th} context (G^T_{kx}). In the example of Figure 1, TRULLO would set t to 0.8037.

That concludes the description of the bootstrapping steps. We now turn to evaluating TRULLO.

IV. Evaluation

We are concerned with effectiveness of TRULLO at predicting trust in novel contexts (Section IV-A) and the added computational cost it entails on a mobile phone (Section IV-B).

A. *Effectiveness of TRULLO*

To quantify bootstrapping effectiveness, we need an appropriate metric. To choose this metric, consider the visitors' goal in bootstrapping trust: accurately to predict sellers' trustworthiness in new contexts, thereby relying only on those trustworthy. In meeting this goal, we are concerned with one measure of effectiveness: *the bootstrapping visitor's utility*, which increases whenever the visitor gets the desired resource (e.g., quality ads) and decreases otherwise.

1) Simulation Setup: In advance of carrying out controlled experiments in a real antique market, it is necessary for us to simulate the system of visitors and sellers. In doing this, however, we seek to ground as much of the necessary behavioral modeling on reality as possible. We refer both to the antique sections of eBay and Amazon and to a study of emerging behavior in electronic bidding by Yang *et al.* [17], specifically to model the distributions of visitors' interests and the production of sellers' ads across contexts. Naturally, there are likely to be changes in a real ubiquitous deployment, but our modeling affords a reasonable expectation of reality. In particular, given the inherent property of publishing ads (known as "preferential attachment" and discussed later), we expect that visitors and sellers distribute across contexts according to a Zipf-like distribution (few contexts have most sellers and visitors, and most contexts, being specialized, have few).

While we shall give the full details next, at a high level, the system we simulate behaves as follows:

- We define a set of possible contexts.
- For each seller in the simulation, a subset of the contexts is allocated to that seller so that the overall distribution of sellers across contexts is Zipf.

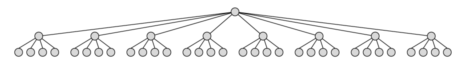

Fig. 2. Ontology of reference. We evaluate TRULLO by simulating its use on ubiquitous devices in antique markets. The simulation models the number of contexts and their relationships with the above ontology. This reflects a typical ontology of the antique sections on eBay and Amazon.

- Likewise, for each visitor, a subset of the contexts are allocated to that visitor so that the overall distribution of visitors is Zipf.
- Each seller is deemed to be an expert in one context, but it also publishes ads in its other contexts, with an increasing likelihood of inaccuracy as the contexts of its ads "move away" from its area of expertise. This models sellers being not equally expert in all contexts.
- Visitors rate the ads based on their accuracy and consequently update their trust in sellers[4].
- In their contexts of interest, visitors select new sellers by bootstrapping trust with TRULLO, and calculate the utility of the ads provided by the selected sellers.
- To measure TRULLO's effectiveness, we compare the utility values so obtained against those obtained by other literature techniques.

Based on these points (detailed next), we will run several simulations (Section IV-A2) and report the results (Section IV-A3).

Defining contexts. In general, when users define which contexts exist and the relationships among them, they define ontologies. For example, they may consider 'antique coins', 'Roman coins', and 'Greek coins' as possible contexts, and may then define relationships among those contexts by logically arranging them in a tree [7] or a graph [9], in which 'antique coins' (parent context) originates 'Roman coins' and 'Greek coins' (child contexts). In real-life, each seller or visitor has its own implicit ontology (representing what it knows) upon which it accordingly acts (sends or rates ads).

Although we focus on evaluating the effectiveness of TRULLO at predicting trust *without* an ontology, it is necessary for us to craft one for the purposes of simulation. This ontology is used to model the characteristics of antique markets (i.e., number of contexts, and how visitors and sellers distribute across them), but *not* to bootstrap trust. Looking at both Amazon and eBay, antique ontologies are flat, i.e., they have few levels and most of the nodes lie at lower levels. We hence consider an ontology of 40 contexts: 1 root, 8 children, each of which has 4 grandchildren (Figure 2).

Contexts in which sellers send ads and in which visitors show interest.

Real world observation. We have ordered the 42 lowest categories in the eBay antique section by the number of items on sale and have then plotted the result in Figure 3: a Zipf distribution best approximates the result; this is in line with

[4]Based on the ads that sellers disseminate in a context, visitors subjectively decide whether sellers are expert in that context.

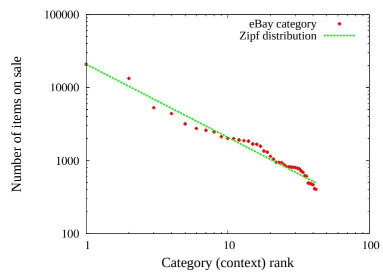

Fig. 3. Popularity of antique categories in eBay. The categories (contexts) are ranked by the number of antiques on sale. The resulting popularity follows a Zipf distribution.

literature expectation. Zipf-like distributions are rooted in the dynamics of sending ads: the more ads a seller sends up to a certain moment, the more likely it is that the seller will send other ads in the future. This, which is a form of preferential attachment, is known to lead to power-law distributions, as shown in economics and complex networks [18]. Moreover, Yang *et al.* [17] show that most of the emerging behaviors in the bidding process on both eBay and auction.co.kr (eBay's Korean partner) also follow a power-law distribution.

Model. We assign to each context in the ontology of reference a number of visitors and sellers such that the popularity of each context (ranked by the number of visitors and sellers) follows a Zipf distribution. While there is good reason to suppose that the distributions of sellers to contexts, and visitors to contexts, are all Zipf-like (power-law with parameter *close* to unity), there is no real reason to suppose that they follow the 'strict' Zipf distribution (power-law with unitary parameter). However, slightly changing such a parameter (1 ± 0.2) demonstrated little effect on our simulation results; we thus report results for a unitary parameter. Overall, the consequence of our modeling is that most visitors and sellers are associated with a few (popular) contexts and that most ads will therefore be created for those contexts. We then consider that each seller is expert in one of the contexts with which it is associated (this context being chosen uniformly at random). Being expert in a context allows sellers to be able to produce highly-relevant ads in that context. However, to account for real-life unpredictability in the behavior of sellers, we allow sellers to randomly change their expertise with a probability p (see Section IV-A3).

Visitors rate ads.

Consideration. Visitors are unaware of the area of expertise of each seller. They simply obtain ads purporting to be within a particular context but, in reality, being of variable quality depending on the distance between the area of expertise of the seller and the context of the ad. Intuitively, the closer an ad to its seller's expertise, the more accurate and thus well-received the ad. This accounts for sellers not being equally expert in all contexts.

Model. Whenever it receives an ad, a visitor rates it as follows: (ad's relevance)= $w - dist(c_a, c_s)$, where c_a is the ad's context, c_s is the seller's context of expertise, and w is the maximum contextual distance (in our ontology $w = 4$). In words, the closer the ad's context to the seller's expertise, the higher the ad's relevance. In particular, as the distance between the two contexts increases, the relevance proportionally decreases from a maximum w (c_a and c_s are the same) to a minimum 0 (c_a and c_s are farthest). We compute the distance between c_a and c_s as the minimum number of edges between the two contexts. Of course, other measures of contextual distance might be defined. Indeed, we have also used a more complex distance function, whereby siblings of lower levels are considered closer than siblings at higher levels. This did not lead to statistically significant results.

Visitors update their trust. Upon rating seller B's ads, A's trust model updates its trust in B. To model this, it was necessary to integrate TRULLO with an available distributed trust model, and we elected to use B-trust [19], our Bayesian trust model for ubiquitous devices. The advantage of this model is that it formally updates trust according to Bayes' theorem on input of *discrete* ratings (not necessarily binary), and, having being designed for ubiquitous devices, it does so relying on a small data structure. For further details of B-trust, please refer to [19].

2) Simulation Execution: We divide the simulation execution into two phases: visitors build initial knowledge upon which they then bootstrap trust.

Build initial knowledge. Initially, sellers send ads of their items, and interested visitors receive them. To simulate this, in each round, each seller sends one ad in each of the contexts c_i with which it is associated. Visitors who are associated with c_i receive the ad and the process of rating and trust update proceeds as described above. Over the set of all sellers, most ads are produced in few contexts (those with most sellers) and few ads in most contexts (those with few sellers), simply as a consequence of allocating sellers to contexts according to a Zipf distribution.

Trust bootstrapping. After the initial phase, whenever visitors wish to view ads in a given context, their devices have to choose the best seller(s) in that context. To model this, each visitor selects one new seller in each of the contexts c_j to which it has been assigned. To do so, for each context c_j, the visitor:

1. Bootstrap trust in c_j for all known sellers other than those we have already assigned to c_j;
2. Select the seller with the highest bootstrapped value;
3. Update its utility: utility_sum= utility_sum + $\frac{1}{w}(w - dist(c_j, c_s))$. In words, the closer c_j to the seller's expertise (c_s), the higher the visitor's utility increase. To obtain an average utility within $[0, 1]$, we have normalized the contribution to the sum (multiplying it by $\frac{1}{w}$).

Two standard bootstrapping methods. In the first of those three steps, each visitor must bootstrap her trust. To do so, literature includes two approaches: initial disposition bootstrapping [20] and recommendation-based bootstrapping [5]. We hence consider that each visitor bootstraps trust with three methods and records a sum of utility for each of them. Thus,

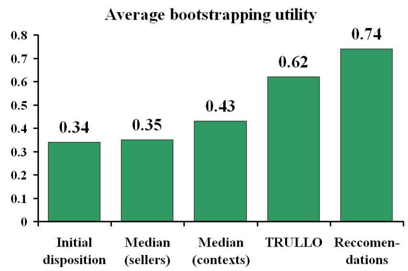

Fig. 4. Average bootstrapping utility for TRULLO, two standard bootstrapping methods, and the median of trust values across sellers and across contexts. We average all utilities across all visitors, and each utility lies within $[0, 1]$. Recommendation-based bootstrapping relies on the ideal case of truthful recommenders (no fake recommendations) among which there is no ontology misalignment.

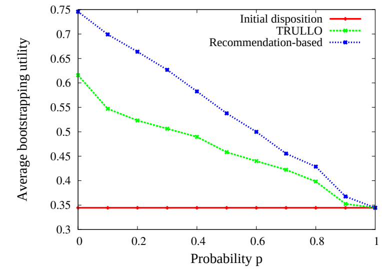

Fig. 5. Average bootstrapping utility for TRULLO and two standard bootstrapping methods as a function of the probability p of sellers randomly changing their expertise.

in addition to TRULLO, we consider:

- *Initial disposition bootstrapping.* The visitor sets the initial trust to be a fixed value in a range $[0, 1]$ that reflects her disposition to trust. For example, that value might be 0.2 if the she is pessimist, or 0.8 if she is optimist. In our simulation, a visitor setting the same initial trust (whatever that is) for all sellers is equivalent of a visitor randomly choosing a seller, thus making initial disposition bootstrapping the worst-case scenario.
- *Recommendation-based bootstrapping.* The visitor bootstraps trust in seller B in context c_x by collecting recommendations from other visitors. We consider that recommenders send their actual trust values in B (i.e., they are wholly reliable) in context c_x (with no ontology misalignment). Under such assumptions, and given that sellers perform consistently over time, recommendation-based bootstrapping is the ideal case. To see why, consider that a recommendation about B is a record of how B performed. That record is objective in the sense that the recommender does not introduce any distortion (no ontology misalignment or fake recommendations). As B performs consistently over time, the recommendation is a predictor of how B will perform at all future points. Naturally, this is unrealistic but it serves as a yardstick (best case scenario) to evaluate TRULLO.

3) Simulation Results: We report results of the simulation execution first, and then closely analyze how some simulated factors might have affected them.

TRULLO compared to two standard bootstrapping methods. We compare TRULLO to both initial disposition bootstrapping and recommendation-based bootstrapping. We simulate a typical antique market that consists of $N_s = 100$ sellers and of $N_v = 1000$ visitors. We run the first part of the simulation twice. As we will see, after two updates, trust values converge (i.e., their confidence is maximum) because sellers performs consistently over time. We then run the second part and average the utilities for each bootstrapping methods across all visitors. Figure 4 shows that TRULLO's average utility (0.62) is much closer to (the ideal) recommendation-based bootstrapping's (0.74) than to (the baseline) initial disposition bootstrapping's (0.34). Even if it relies on truthful recommendations with no ontology misalignments, recommendation-based bootstrapping does not reach the maximum average utility of 1 because some contexts have no specialized sellers (selecting which the resulting utility would be 1).

TRULLO compared to a simple median. One may now ask whether using a simple median instead of TRULLO would yield similar results. To see whether this is the case or not, consider a visitor bootstrapping its trust in seller S. We distinguish two cases:

1) If the visitor knows k_s sellers in the bootstrapping context, it may set its initial trust as the median of its trust values in those sellers (median across sellers).
2) If the visitor knows S across k_c contexts, it may bootstrap trust as the median of its trust values in S across those contexts (median across contexts).

Figure 4 shows that TRULLO's average utility is much higher than that of the median (computed across either sellers or contexts). In particular, either median performs slightly better than initial disposition. Still, for any number of contexts/sellers, either median's utility is less than 0.45.

Factors affecting the effectiveness of TRULLO. Having these preliminary results, we now see how some simulated factors might have affected TRULLO's bootstrapping utility:

- *Confidence in the trust values upon which visitors bootstrap.* By decreasing the number of rounds of the first part of the simulation, visitors would receive fewer ads from sellers and thus run fewer trust updates, and they would then bootstrap trust upon more uncertain trust values. By doing so, TRULLO's average utility, however, does not significantly change because sellers perform consistently over time, and hence trust values in them converge just after receiving two ads from any of them in a given context. So, to capture real-life unpredictability in the behavior of sellers, we now allow sellers to send ads whose relevance reflects either their actual expertise with probability $(1 - p)$ or a randomly chosen expertise with probability p. Figure 5 shows that as p increases, the average utilities for TRULLO and recommendation-based bootstrapping decrease, as expected. However, even for

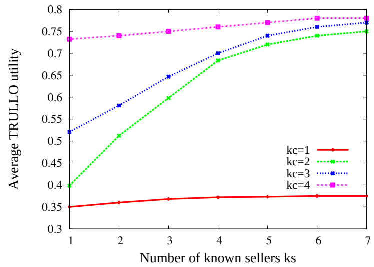

Fig. 6. Average bootstrapping utility for TRULLO versus its input size. Each utility refers to TRULLO bootstrapping trust in a context in which k_s sellers are known in other k_c contexts. We average across all visitors.

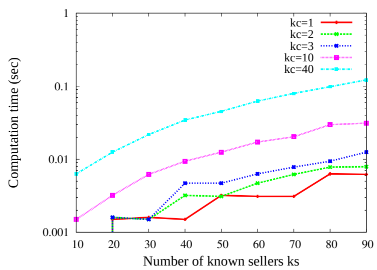

Fig. 7. Visitor's device performance (seconds) versus TRULLO input size, as means of 10 runs.

high values of p, both bootstrapping methods perform better than initial disposition. Also, their utilities decrease linearly. The reason is that for each 'misplaced' ad (i.e., ad that a seller sends from a context other than that of her expertise), the utility decreases by a certain amount on average. The number of 'misplaced' ads increases as p increases and, thus, the average utility *proportionally* decreases.

- *Size of the input matrix.* The size of TRULLO's input matrix depends on how sellers distribute across contexts. As the input size changes, bootstrapping accuracy might change, and that might affect the utilities of visitors using TRULLO. To see how, during the simulation execution, whenever TRULLO bootstrapped trust, we kept track of its utility and of the corresponding input size. We then averaged all utilities corresponding to the same size. Figure 6 shows that TRULLO performs better than initial disposition bootstrapping even on input of one single trust value. As one might expect, as the number of contexts and the number of sellers upon which TRULLO bootstraps increase, so does its average utility.

B. Computational Cost of TRULLO

Given that TRULLO effectively bootstraps trust, it is now worth checking whether it is usable on a mobile phone. To this end, we implemented TRULLO in J2ME and ran it on a Nokia 3230 mobile phone (whose features include: Symbian operating system 7.0, 32 MB of memory, 32-bit RISC CPU). Figure 7 shows TRULLO performance, given as the mean of 10 runs, for varying input $(k_s \times k_c)$ matrix sizes. We minimized background activities by shutting down all applications other than TRULLO. The computational overhead is very low. That is because TRULLO's input matrix only contains the ratings of a single user's past experiences. For example, to bootstrap a value in a context in which 20 sellers are known in 10 contexts (maximum input in the previous experiments), TRULLO takes just 3.2 milliseconds.

V. Discussion

Based on those evaluation results, we now discuss various open questions.

Correlation of how sellers perform across contexts. One inherent property of TRULLO design is that one can extract statistical features from trust values. Unfortunately, we do not have any trust set from real ubiquitous computing applications that corroborates this property. However, we do have Internet web sites reporting user ratings across contexts. Take hostels.com: it reports customer ratings of hostels across contexts, namely character, security, location, staff, fun, and cleanliness. By sampling parts of these trust ratings and applying the singular value decomposition, we learned that they do correlate across contexts - for example, trust ratings about 'character' and 'staff' roughly share the same statistical features. That, however, does not guarantee that this would be the case in all ubiquitous computing applications. For this reason, our simulation setup has made no explicit assumption on whether trust correlates across contexts, and it has shown how some simulated factors have affected the results.

Choosing the right bootstrapping method for a ubiquitous computing application. The choice of the right bootstrapping method is application-dependent. More concretely, consider the following aspects that are usually critical in ubiquitous computing:

- *Device computational cost.* If the computational cost must approach zero and bootstrapping accuracy does not matter, then initial disposition bootstrapping may be a fair choice. Otherwise, one might use TRULLO, which runs on a standard mobile at modest computational cost (Section IV-B).
- *Device communication overhead.* In a fully distributed setting, asking for recommendations might considerably increase data traffic among devices. To avoid that, one may use TRULLO since it is more effective than initial disposition bootstrapping and solely relies on local information (no device communication required).
- *Threats.* Modeling hostile environments is an important research question, on which we have not focused but now ponder briefly. Consider environments that may be deemed hostile because of either ontology misalignments among users, or presence of fake recommendations, or users behaving very differently across contexts. At the presence of the first two problems (which are indeed very likely), recommendation-based bootstrapping may suffer as it relies on third party information that, in this case, would be made unreliable by ontology misalignments and fake recommendations. In such a situation, one may prefer TRULLO as it relies only on local information.

However, if sellers perform very differently across logically close/related contexts, then TRULLO may turn to be a poor choice as it would not extract any statistical property from sellers' behavior. Thus, at the presence of all three problems, one has to resort to initial disposition bootstrapping.

VI. Related Work

Literature includes three main ways to set initial trust. *First*, most of the reputation models in P2P networks [21], [22] and in social networks [23], [24] effectively bootstrap trust using recommendations, but they do so in a single context. It would be no trivial, yet interesting to extend these approaches using, for example, collaborative filtering techniques [25]. However, this extension needs further research for coping with fake recommendations and ontology misalignments among recommenders, and for scaling in a fully distributed setting of portable devices.

The *second* proposition consists in assigning fixed trust bootstrapping values. Two examples include: Perich *et al.* [20], who define trust values based on the trustor's initial dispositions (pessimistic, optimistic, and undecided); and Buchegger *et al.* [6], who set the initial trust to a uniform distribution (as does our recent work [19]). Those choices apply only to specific problem domains (e.g., to packet forwarding).

The *third* proposition consists in setting the initial trust value for B in context c_x close to the trust values we already have about B in contexts *similar* to c_x. This may efficiently bootstrap trust, but, on the other hand, needs a measure of contextual distance to find out which are the contexts similar to c_x. Two recent types of approach might fill the gap: the first [26], [7] defines similarity between any two contexts in an ontology as the distance between the two corresponding nodes; the second type [27] draws context similarity based on a direct graph of contexts (a less-constrained structure than a tree) whose weights have to be, however, manually set by device users. The researchers who proposed the first type of approach have acknowledged that the idea of a universally accepted context ontology hardly belongs to reality; those of the second concede that their solution has to be automated to be usable. TRULLO automates the bootstrapping process in that it decomposes the ratings of its user's past experiences and consequently determines unknown trust values without user intervention.

VII. Conclusion

We have shown that TRULLO effectively bootstraps trust by integrating it with B-trust (an existing trust model for ubiquitous devices) and by simulating its use on ubiquitous devices in informal antique markets; in particular, TRULLO performs close to how exchanging recommendations would do in an ideal (though unrealistic) world, one in which recommenders are wholly truthful and, furthermore, share the same ontology. Our J2ME implementation of TRULLO does not impact the usability of a Nokia 3230 mobile phone. We are currently designing controlled experiments to be carried out in a large scale deployment.

Acknowledgments: We thank Damon Wischik, Vladimir Dyo, and Ilias Leontiadis for their contributions, and Microsoft Research Cambridge for its financial support.

References

[1] J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava, "Participatory sensing," in *Proc. of ACM WSW*, 2006.

[2] H. Rheingold, *Smart Mobs: The Next Social Revolution*. Perseus Books Group, 2002.

[3] L. Kagal, T. Finin, and A. Joshi, "Trust-Based Security in Pervasive Computing Environments," *IEEE Computer*, December 2001.

[4] V. Cahill, E. Gray, J.-M. Seigneur, C. Jensen, Y. Chen, B. Shand, N. Dimmock, A. Twigg, J. Bacon, C. English, W. Wagealla, S. Terzis, P. Nixon, G. Serugendo, C. Bryce, M. Carbone, K. Krukow, and M. Nielsen, "Using Trust for Secure Collaboration in Uncertain Environments," *IEEE Pervasive Computing Mobile and Ubiquitous Computing*, August 2003.

[5] A. Abdul-Rahman and S. Hailes, "Supporting Trust in Virtual Communities," in *Proc. of HICSS*, 2000.

[6] S. Buchegger and J.-Y. L. Boudec, "A robust reputation system for P2P and mobile ad-hoc networks," in *Proc. of P2PECON*, 2004.

[7] J. Liu and V. Issarny, "Enhanced Reputation Mechanism for Mobile Ad Hoc Networks," in *Proc. of iTrust*. LNCS, 2004.

[8] G. Forsythe, M. Malcom, and C. Moler, *Computer Methods for Mathematical Computations*. Prentice Hall, 1977.

[9] J. Coutaz, J. Crowley, S. Dobson, and D. Garlan, "Context is key," *Communications of the ACM*, March 2005.

[10] J. Crowley, J. Coutaz, G. Rey, and P. Reignier, "Perceptual Components for Context Aware Computing," in *Proc. of Ubicomp*. LNCS, 2002.

[11] J. R. Douceur, "The Sybil Attack," in *Proc. of IPTPS*, 2002.

[12] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman, "SybilGuard: defending against sybil attacks via social networks," in *Proc. of ACM SIGCOMM*, 2006.

[13] B. Shand, N. Dimmock, and J. Bacon, "Trust for ubiquitous, transparent collaboration," *Wireless Networks*, March 2004.

[14] A. Serjantov and R. Anderson, "On dealing with adversaries fairly," in *Proc. of WEIS*, 2004.

[15] R. Hardin, "The street-level epistemology of trust," *Politics and Society*, December 1993.

[16] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. Altman, "Missing value estimation methods for DNA microarrays," *Bioinformatics*, vol. 17, no. 6, 2001.

[17] I. Yang, H. Jeong, B. Kahng, and A. L. Barabasi, "Emerging behavior in electronic bidding," *Physical Review*, vol. 68, no. 1, 2003.

[18] A.-L. Barabasi, *Linked: How Everything Is Connected to Everything Else and What It Means*. Penguin, 2003.

[19] D. Quercia, S. Hailes, and L. Capra, "B-trust: Bayesian Trust Framework for Pervasive Computing," in *Proceedings of iTrust*. LNCS, 2006.

[20] F. Perich, J. Undercoffer, L. Kagal, A. Joshi, T. Finin, and Y. Yesha, "In Reputation We Believe: Query Processing in Mobile Ad-Hoc Networks," in *Proc. of the IEEE Mobiquitous*, 2004.

[21] K. Aberer and Z. Despotovic, "Managing trust in a peer-2-peer information system," in *Proc. of ACM CIKM*, 2001.

[22] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The Eigentrust algorithm for reputation management in P2P networks," in *Proc. of ACM WWW*, 2003.

[23] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins, "Propagation of trust and distrust," in *Proc. of ACM WWW*, 2004.

[24] J. Sabater and C. Sierra, "Reputation and social network analysis in multi-agent systems," in *Proc. of the ACM AAMAS*, 2002.

[25] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl, "Grouplens: Applying collaborative filtering to Usenet news," *Communications of the ACM*, 1997.

[26] L. Capra, "Reasoning about Trust Groups to Coordinate Mobile Ad-Hoc Systems," in *Proc. of IEEE SECOVAL*, 2005.

[27] M. Kinateder and K. Rothermel, "Architecture and Algorithms for a Distributed Reputation System," in *Proc. of iTrust*. LNCS, 2003.