PHILOSOPHICAL TRANSACTIONS OF THE ROYAL SOCIETY A

Phil. Trans. R. Soc. A (2008) **366**, 3781–3793
doi:10.1098/rsta.2008.0134
Published online 4 August 2008

Trust models in ubiquitous computing

By Karl Krukow[1], Mogens Nielsen[2] and Vladimiro Sassone[3,*]

[1]*Trifork, Margrethepladsen 4, 8000 Århus C, Denmark*
[2]*University of Aarhus, Nordre Ringgade 1, 8000 Åarhus C, Denmark*
[3]*School of Electronics and Computer Science, University of Southampton, Southampton, UK*

We recapture some of the arguments for trust-based technologies in ubiquitous computing, followed by a brief survey of some of the models of trust that have been introduced in this respect. Based on this, we argue for the need of more formal and foundational trust models.

Keywords: trust and reputation management; probabilistic modelling; science for ubiquitous computing

1. Introduction

It is a well-established fact that ubiquitous computing (UbiComp) raises a whole new set of challenges with respect to fundamental security aspects such as secrecy and privacy (e.g. the UbiComp Grand Challenge as formulated in Sloman (2006)). Many new features of UbiComp (virtual anonymity, scalability, mobility, autonomy, ubiquity, incomplete information, global connectivity, etc.) will affect our notion of security requirements. As an example, a mobile computing entity may find itself in a hostile environment, disconnected from its preferred security infrastructure, e.g. its usual certification authorities, and the autonomy requirement means that even in this scenario, it must be able to assign privileges to other UbiComp entities; privileges that are meaningful based on usually incomplete information the assigning entity has about the assigned entity.

One particular attempt addressing these challenges builds on the intuitively appealing idea of building a security framework for UbiComp resembling the web of trust, which we all use in our daily communication with, for example, institutions, companies and other human beings. This has led to a growing research field sometimes referred to as computational trust.

To be more specific, computational trust builds on *abstractions* inspired by the human concept of trust, which aims at supporting decision making by computational agents in the presence of unknown, uncontrollable and possibly harmful entities and in contexts where the lack of reliable information makes classical techniques useless.

* Author for correspondence (vs@ecs.soton.ac.uk).

One contribution of 19 to a Discussion Meeting Issue ‘From computers to ubiquitous computing, by 2020’.

As expected of an ineffable idea deeply linked with human emotions and experience, trust appears in several very different forms, from description and specification languages to middleware, from social networks and management of credential to human–computer interaction. These rely in different degrees on a variety of underpinning mathematical theories, including, e.g. logics, game theory, semantics, algorithmics, statistics and probability theory.

Computational trust was originally introduced as an alternative to traditional security technologies in trust management (Blaze *et al.* 1999*b*). This approach still provides an important class of applications where, for example, access to information or resources is based on a *provider's* trust in the *requesters*. However, within UbiComp computational trust deals not merely with access control, but more generally with decision making by computational agents in the presence of unknown, uncontrollable and possibly harmful entities. This is the case for, for example, the autonomous selection by a *requester* of (apparently similar) services based on its trust in particular *providers*. Such decisions may also affect security: interaction often entails exposing personal data, as well as requiring resources such as time, computation, battery and storage.

Within computational trust, several applications have been constructed with truly impressive experimental performance. However, we are not yet in a position where we understand why, when and how a particular approach is applicable, as expressed in Sabater & Sierra (2005). Such questions are typically formulated in terms of underlying models. Many models for computational trust have been proposed, but it is hard to identify one model (or even a few) *accepted widely* by the research community. Unfortunately, the lack of widely accepted formal models leads to a lack of clarity about the exact objectives of proposed systems; as Samuel Karlin was quoted to have said in a tribute lecture to honour R. A. Ficher: 'The purpose of models is not to fit the data but to sharpen the questions'.

Our position is first of all that within computational trust we need to sharpen *our* questions, in the sense that within *any* approach to computational trust, it should be possible to ask and to answer formally questions on the behaviour of systems. This applies, of course, to any piece of software, and a wealth of literature exists under headings such as software *specification*, *analysis* and *verification*. However, most of this builds on a simple *correctness* approach, in the sense that questions above are formulated as yes/no questions. This makes perfect sense for many software systems, but it is our position that in the setting of computational trust, we need to develop new formal frameworks for a more general notion of correctness, which allows us, for example, (i) to express and to argue *how well* a particular system behaves under various assumptions about the environments (i.e. in which application scenarios does the system do well?) and (ii) to express and argue how *robust* a particular system is with respect to changes in the environment.

In this paper, we first give a brief survey of some of the systems and models for computational trust, which have been studied in the literature, and which we find particularly relevant for a discussion of our positions above (based on Krukow 2006).

We then illustrate some preliminary attempts towards a new approach to correctness as introduced above. This is presented within formal probabilistic models, in which we sketch ideas towards a theoretically well-founded technique for comparing probabilistic systems in various different environments.

2. Some systems and models

It is not our intention to survey the entire collection of works on trust in computer science; this is too comprehensive, and excellent surveys already exist (e.g. Grandison & Sloman 2000; Ramchurn *et al.* 2004; Sabater & Sierra 2005; Jøsang *et al.* 2006). Also, the PhD thesis of Abdul-Rahman (2005) contains a vast survey (mostly) of the human notion of trust in computer science, including insights from social sciences.

So, here we only focus on a few of the systems and models which we find particularly relevant for our discussion on computational trust within UbiComp. However, we should mention that there is a whole different strand of research on trust distinct from the more technical notions on which we focus, which has resulted in the term 'trust' being overloaded within computer science. This other strand deals with a computational formalization of the *human notion of trust*, i.e. trust as a sociological, psychological and philosophical concept. However, the human concept of trust is elusive and its many facets make it hard to define formally (Marsh 1994; Cahill *et al.* 2003). We believe that to live up to the UbiComp challenge, it is necessary that the two concepts be merged in a 'unified' theory of trust that combines the strengths of both notions. To be more precise, our ideal would be to combine the *rigour* of traditional trust management with the *dynamics* and *flexibility* of the human notion.

(*a*) *Credential-based computational trust*

(i) *Influential systems*

Blaze *et al.* (1996) developed the traditional notion of trust management, and also developed the first prototype trust management system, *PolicyMaker*. For a good overview of PolicyMaker and traditional trust management, see Blaze *et al.* (1999*b*). The most general form of proof of compliance (POC) in PolicyMaker is undecidable and several natural restrictions are NP hard (Blaze *et al.* 1998). PolicyMaker considers a version of the POC problem, which requires that all assertions are *monotonic* (assertions are fully programmable functions that are part of credentials). This leads to a restricted notion of POC, which is decidable in polynomial time (Blaze *et al.* 1998). *KeyNote* (Blaze *et al.* 1999*a*,*c*), the successor of PolicyMaker, restricts the language of assertions to a simple domain-specific language so that resource usage is proportional to program size (Blaze *et al.* 1999*b*). KeyNote is less general than PolicyMaker but has simpler syntax and semantics, and requires less computational power. Architectural trade-offs between PolicyMaker and KeyNote are considered by Blaze *et al.* (1999*c*). A number of applications using PolicyMaker and KeyNote have also been developed (Blaze *et al.* 2001, 2002).

Delegation Logic (*DL*; Li *et al.* 1999, 2000, 2003) and its monotonic version D1LP is a language for trust management of credentials, policies and requests; it extends the logic programming language Datalog with expressive delegation constructs (Li *et al.* 2003). D1LP is implemented by translation in to ordinary logic programs, which enables use of existing logic programming technology, e.g. Prolog. An important feature of DL and D1LP is that the notion of POC is well founded in the roots of logic. D1LP supports the concepts of authenticated attributes and decentralized

attribute authority. *Decentralized attributes* allow an entity to assert that another has a certain attribute (i.e. satisfies a certain property, e.g. 'being a university student'). D1LP is declarative, expressive and tractable (compliance checking is polynomial in the size of credentials, policies and requests).

The *role-based trust* (*RT* management) framework is a family of languages for policies and credentials, which combines the strengths of role-based access control and trust management (Li & Mitchell 2002, 2003*b*). The RT framework consists of the languages together with an engine which works by translating credentials into Datalog rules, similar to the DL languages. This enables POC checking in polynomial time. Apart from supporting role-based features, Li *et al.* argue that RT is more convenient than D1LP although both support attribute-based access control. RT supports concepts of intersection roles, manifold roles and delegation of role activation; these enhance the expressive power, compared with other frameworks, e.g. D1LP (Li & Mitchell 2002). Furthermore, RT supports distributed credentials and distributed credential discovery (Li & Mitchell 2003*b*). RT is monotonic and it has been argued that non-monotonicity requires complete information, which is unrealistic in distributed systems (Li & Mitchell 2002; Li *et al.* 2003).

Other examples include SPKI/SDSI (Ellison *et al.* 1999; Clarke *et al.* 2001; Li & Mitchell 2003*c*), SD3 (Jim 2001) and Binder (DeTreville 2002); the latter two of which also are based on Datalog. Czenko *et al.* (2005) present a version of *RT* with non-monotonic features. Appel & Felten (1999) use a logic-based approach to authentication. Their logic is higher order and undecidable. In analogy with proof-carrying code, the requester must submit a proof that the request should be allowed; the proof is then efficiently *checkable* in the framework.

(ii) *Fundamental models*

The traditional trust management approach was born from an engineering perspective: important concepts were identified and prototype systems were built. Since then the notion of POC (and trust management in general) has developed and is now better founded on existing theory. We consider just two formal models that give well-founded notions of POC.

Li & Mitchell (2003*a*) proposed *Constraint Datalog*, an extension of Datalog, as a promising operational foundation for trust management. A number of existing trust management systems are 'equivalent' to a subset of Datalog. Li & Mitchell (2003*a*) argued that standard Datalog 'is not sufficiently expressive for fine-grained control of structured resources'. They show how Datalog can remain tractable when extended with the so-called linearly decomposable unary constraint domains, and that permissions associated with structured resources are expressible within this framework.

Weeks (2001) presents a mathematical framework for trust management systems based on the existence of *least fixed points* of monotonic functions on complete lattices. This gives a general semantic model for the trust management systems and the notion of POC; further, it is shown that the general framework can be instantiated to obtain a number of existing systems, e.g. KeyNote and SPKI.

A great advantage of this approach is that existing theory of fixed points and algorithms for fixed point computation can be used in trust management engines. Weeks (2001) shows also how this generality can even lead to more efficient algorithms for compliance checking.

(*b*) *Experience-based computational trust*

We use the term 'experience based' to cover the systems and models where an entity's trust in another is based, in part, on past behaviour or evaluations of past behaviour (similar to the human notion of trust). This also covers many so-called *reputation systems* or *reputation-based* trust management systems, which are often used in peer-to-peer (P2P) and eCommerce applications.

The amount of literature on experience-based trust models, including reputation systems, has quickly grown very extensive. In our view, these systems are based on a few fundamental principles, and even fewer fundamental models. Hence, we do not claim to be complete: we focus on the fundamental models, and only *selected examples* of systems deploying those models. Also, in the following we make a number of simplifications, but stay general enough to capture most of the principles of existing experience-based systems.

Experience-based trust is based on a set of principals $\mathcal{P}$ interacting. Principal p records its interactions with other principals, so that at each point in time, t, it has a set, an *interaction history* $\mathrm{Hist}^p(t)$, consisting of a representation of its timed interactions in the past with other principals. We write $\mathrm{Hist}^p_q(t)$ for the q-projection, i.e. p's timed interactions with principal q.

At time t, the sets $(\mathrm{Hist}^p(t)|p\in\mathcal{P})$ constitute the *direct data of an experience-based system at time t*. When p needs to make a decision at time t, e.g. about a principal q, principal p does this based on information about the direct data of the system at time t. Usually, this information is incomplete: while p (often) knows $\mathrm{Hist}^p(t)$, the sets $\mathrm{Hist}^r(t)$ for $r\neq p$ may not be known exactly. This may be due to several reasons: p may only have $\mathrm{Hist}^r(t')$ for some $t'<t$; when asked about $\mathrm{Hist}^r(t')$, r may lie; principal p may not be able to obtain any information about $\mathrm{Hist}^r(t')$; principal p may only see some abstracted version $\mathrm{Abs}(\mathrm{Hist}^r(t'))$ of r's direct data; and any combinations of the above, etc.

Most experience-based systems work on some abstracted version of the direct data, denoted $\mathrm{AbsHist}^p(t)$. A common example of an abstraction is the following. At time t, principal p is interested in information about principal q. Each past interaction is evaluated as either 'positive' or 'negative', and time is ignored; hence, $\mathrm{Hist}^p(t)$ is abstracted to a set of pairs (one for each other principal q) consisting of the number of 'positive' interactions and the number of 'negative' interactions (with q). p may obtain information about other principals' interactions with q, and combine the information obtained into a single pair by adding up the total number of 'negative' interactions, and similarly adding up the total number of 'positive' interactions. This example system is much like the eBay system (www.ebay.com).

In the following, we consider two types of systems: non-probabilistic and probabilistic.

(i) *Non-probabilistic approaches*

We use the term *concrete reputation systems* for experience-based systems where the sets $\mathrm{Hist}^p(t)$ undergo little or no abstraction (Krukow *et al.* 2005). Shmatikov & Talcott (2005) define a concrete reputation system, which is centred around a notion of 'licences'. A licence formalizes restrictions and

obligations, i.e. what principals *must* do and what they *cannot* do, expressed in terms of certain programmable Boolean functions, taking interaction histories and other relevant data as input.

Licences are used when principals want to access resources owned by other principals. The provider of resources p specifies, for each resource r an access method, which takes as input a licence l, a requester q, and an interaction history $\mathrm{Hist}_q^p(t)$ and outputs a Boolean telling if the agent q can access r with licence l given the interaction history. The resource owner can specify this method using any computable function, but typically the owner would check if the licence permits this use and if it has expired, etc. This gives a reputation system where decisions are made based on exact criteria on past histories. Hence, if one can reason about the licence functions and the access methods, then it is possible to reason about the security guarantees provided by the system.

Kamvar *et al.* (2003) present a reputation system for P2P systems, called *EigenTrust* (also known as EigenRep), based on the existence of stationary distributions for Markov chains. In comparison with the framework of Shmatikov & Talcott, EigenTrust abstracts away more information: each interaction is evaluated as either 'satisfactory' or 'unsatisfactory', and time is ignored; each principal p computes a value for other principals q as $s_{pq} = \mathrm{sat}(p,q) - \mathrm{unsat}(p,q)$ (i.e. the number of interactions between p and q, where p rated q's performance as satisfactory minus the number of unsatisfactory ones); and finally, these values are then normalized for each peer p (by comparison with other peers' performance): $c_{pq} = \max(s_{pq}, 0)/\sum_q \max(s_{pq}, 0)$. The normalized values define the abstracted histories $\mathrm{AbsHist}_q^p(t)$, which give rise to a Markov chain (given by $[c_{pq}]$) that has a stationary distribution $(t_q)_{q\in\mathcal{P}}$. This distribution is computed using an iterative synchronous algorithm; the value t_q then represents the 'global score' of q (uniformly for all p).[1]

One problem with EigenTrust is that no meaningful semantic interpretation of the value t_q exists (only 'the larger the better'). Furthermore, temporal aspects are ignored, and information is thrown away with the normalization. In other words, it is hard to do formal reasoning about the system.

It should also be mentioned that Stephen Marsh was among the first to formalize a computational human notion of trust in computer science in his PhD dissertation (Marsh 1994). Abdul-Rahman & Hailes (2000) and Abdul-Rahman (2005) were was also among the first to consider a simplified practical model similar to Marsh's. Xiong & Liu (2004) present PeerTrust featuring a complex trust metric, but which only has an intuitive justification. In our opinion, these systems exhibit the same problems as EigenTrust: rigorous reasoning about the past behaviour seems impossible given only the abstracted information.

(ii) *Probabilistic approaches*

The probabilistic systems work by assuming a particular probabilistic model, say λ, for the behaviour of principals. The goal is to predict the behaviour of principals in future interactions, given their behaviour in past interactions and

[1] This works in a manner similar to Google's Pagerank.

the model λ, i.e. using the standard notation $P(\cdot|\cdot)$ for a conditional probability, the goal is to compute a probability $P(\text{'next'}|\text{'past'}, \lambda)$. The abstractions, i.e. AbsHist(t), are then chosen to be as efficient as possible, while preserving as much information as relevant with respect to the model.

In the following, we shall illustrate our probabilistic trust-based systems using a simple, but still representative, example of a principal model λ, in which each interaction is observed as being either 'honest' or 'dishonest'. Furthermore, for each principal q, there is a *fixed* probability $\theta_q \in [0,1]$ of q acting honestly in any interaction. Note that this assumes that q is always honest with probability θ_q independently of any other information we might have (e.g. the time, the past, interactions with other principals, etc.). The parameters, θ_q, are unknown and the goal is to estimate them.

Despotovic & Aberer (2004, 2006) propose a probabilistic system and an estimation algorithm based on *maximum likelihood*. In our simple illustrative model, the algorithm uses a maximum-likelihood procedure, which seeks to find a θ_q, which maximizes the likelihood expression based on past interactions.

In Despotovic & Aberer (2004, 2006), peers can report to other peers on past behaviour (and they are allowed to lie in their reports), and the authors also present an approach based on normal distributions instead of the fixed θ_q's. Similarly, the maximum-likelihood techniques are used to estimate the parameters of the normal distribution.

Jøsang & Ismail (2002) and Mui *et al.* (2002) were among the first (independently) to develop reputation systems based on a Bayesian probabilistic approach with *beta priors* (for further developments, see Buchegger & Le Boudec 2004; Teacy *et al.* 2005). In our simple illustrative probabilistic model λ considered above, i.e. with fixed probability $\theta_q \in [0,1]$ of principal q acting honestly in any interaction, the main idea is to represent the current estimate of the θ_q's by probability density functions (pdfs) defined on the interval [0,1], more specifically by pdfs from the family Beta(α, β), where the two parameters $\alpha > 0$ and $\beta > 0$ select a specific beta distribution from the family. As an example, Beta(1, 1) represents the uniform pdf. A central observation and idea is that the beta distributions provide a so-called *family of conjugate prior distributions*. Furthermore, applying Bayes' theorem provides an algorithmically very simple way of computing the posterior pdf from a prior Beta(α, β) and subsequent observations of h honest and d dishonest interactions with q: Beta($\alpha + h$, $\beta + d$).

Jøsang & Ismail (2002), Mui *et al.* (2002), Buchegger & Le Boudec (2004) and Teacy *et al.* (2005) all present systems based on the beta model. Technically, all the systems work by maintaining for each principal q the two parameters (α, β) of the current pdf representation of θ_q, and then estimating θ_q with the expected value of Beta(α, β): $\alpha/(\alpha + \beta)$.

However, the systems (except for Mui *et al.* (2002) and Teacy *et al.* (2005)) deviate from the simple model above in the following sense: the parameters (α, β) are adjusted as time passes, for example, Jøsang uses exponential decay, where α and β are multiplied by a constant (between 0 and 1) each time parameters are updated (or a fixed time limit is exceeded). The intuition is that somehow information about more recent interactions should be considered more important than information about older interactions.

Several models are based on a notion of 'belief theory' that is related to probability theory: Yu & Singh (2002) developed a distributed reputation system, and Jøsang (2001) developed the subjective logic of opinions. Indeed, the subjective logic is closely linked to the probabilistic beta model (Jøsang 2001).

Finally, there are a number of 'economic' reputation system models based on the theory of games, for example, a 'reputation effect' occurs in rational strategies when modelling interaction as a finitely repeated Prisoner's Dilemma game (Kreps & Wilson 1982; Wilson 1985). For a good overview of this area, see the work of Dellarocas (2003, 2004). The notion of (computational) mechanism design is also relevant for this area. For mechanism design, see Papadimitriou (2001) and Feigenbaum & Shenker (2002); with respect to trust and mechanism design, see Dash *et al.* (2004).

3. Towards formal computational trust

The main conclusion on our survey above is that although quite a few attempts towards a formal foundation for computational trust exist, we are still a long way from a framework allowing us (i) to express and to argue *how well* a particular system behaves under various assumptions about the environments (i.e. in which application scenarios does the system do well?) and (ii) to express and argue how *robust* a particular system is with respect to changes in the environment. In this section, we illustrate a few preliminary attempts towards such a framework based on ideas from Krukow & Nielsen (2007), Nielsen *et al.* (2007) and Sassone *et al.* (2007).

We focus here on probabilistic computational trust as described above. Consider, for example, the maximum-likelihood algorithm of Aberer & Despotovic. Is the algorithm correct? The traditional notion of correctness would require a proof that the algorithm provided satisfies its specification in the traditional sense; in this case, that it actually computes the maximum likelihood. While this is certainly necessary, what we are more interested in is in which sense maximum likelihood is 'the correct' algorithm to apply, i.e. is the specification 'correct'? In this view, an algorithm can be more or less appropriate depending on how well it approximates θ in the particular chosen probabilistic model. One can, of course, argue for the usefulness of the algorithm based on experiments within particular applications, but in the following, we propose an alternative and formal approach in which to express a new notion of 'correctness' addressing (i) and (ii) above.

More concretely, we shall propose a generic measure to 'measure' specific probabilistic trust-based systems in a particular environment (i.e. 'a set of representative and common conditions'). The measure, which is based on the so-called Kullback–Leibler divergence, is a measure of how well an algorithm approximates the 'true' probabilistic behaviour of principals.

Consider a probabilistic model of principal behaviour, say λ. We consider only the behaviour of a single fixed principal p, and we consider only algorithms that attempt to solve the following problem. Suppose we are given an interaction history $\boldsymbol{X}$ obtained by interacting n times with principal p. Suppose also that there are m possible outcomes $(y_1, \ldots, y_m)$ for each interaction. The goal of a probabilistic trust-based algorithm, say $\mathcal{A}$, is to approximate a distribution on

the outcomes $(y_1, \ldots, y_m)$ given this history $\boldsymbol{X}$. That is, $\mathcal{A}$ satisfies

$$\mathcal{A}(y_i|\boldsymbol{X}) \in [0,1] \quad (\text{for all } i), \qquad \sum_{i=1}^{m} \mathcal{A}(y_i|\boldsymbol{X}) = 1.$$

We assume that the probabilistic model, λ, defines the following probabilities: $P(y_i|\boldsymbol{X},\lambda)$, i.e. the probability of 'y_i in the next interaction given a past history of $\boldsymbol{X}$' and $P(\boldsymbol{X}|\lambda)$, i.e. the '*a priori* probability of observing sequence $\boldsymbol{X}$ in the model'.

Now, $(P(y_i|\boldsymbol{X},\lambda)|i=1,2,\ldots,m)$ defines the true distribution on outcomes for the next interaction (according to the model); by contrast, $(\mathcal{A}(y_i|\boldsymbol{X})|i=1,2,\ldots,m)$ attempts to approximate this distribution. And clearly, the question of *how well* $\mathcal{A}$ performs in environments conforming with λ now boils down to how close $(\mathcal{A}(y_i|\boldsymbol{X})|i=1,2,\ldots,m)$ is to $(P(y_i|\boldsymbol{X},\lambda)|i=1,2,\ldots,m)$. Probability theory provides ways of formalizing this, e.g. the Kullback–Leibler divergence (Kullback & Leibler 1951), which is closely related to Shannon entropy, is a measure of the distance from a true distribution to an approximation of that distribution. The Kullback–Leibler divergence from distribution $\hat{p} = (p_1, p_2, \ldots, p_m)$ to distribution $\hat{q} = (q_1, q_2, \ldots, q_m)$ on a finite set of m outcomes, is given by (any log base could be used)

$$D_{\mathrm{KL}}(\hat{p}||\hat{q}) = \sum_{i=1}^{m} p_i \log_2\left(\frac{p_i}{q_i}\right).$$

Now we have a measure of the 'quality' of an output, how can this be generalized to a measure of the quality of an algorithm $\mathcal{A}$ relative to λ? For each n let $\boldsymbol{O}^n$ denote the set of interaction histories of length n. Let us define, for each n, the *nth expected Kullback–Leibler divergence from* λ *to* $\mathcal{A}$:

$$D^n_{\mathrm{KL}}(\lambda||\mathcal{A}) \stackrel{(\mathrm{def})}{=} \sum_{\boldsymbol{X}\in\boldsymbol{O}^n} P(\boldsymbol{X}|\lambda) D_{\mathrm{KL}}(P(\cdot|\boldsymbol{X},\lambda)||\mathcal{A}(\cdot|\boldsymbol{X})).$$

Note that for each input sequence $\boldsymbol{X}\in\boldsymbol{O}^n$ to the algorithm, we evaluate its performance as $D_{\mathrm{KL}}(P(\cdot|\boldsymbol{X},\lambda)||\mathcal{A}(\cdot|\boldsymbol{X}))$; however, we accept that some algorithms may perform poorly on very unlikely training sequences, $\boldsymbol{X}$. Hence, we weigh the penalty on input $\boldsymbol{X}$, i.e. $D_{\mathrm{KL}}(P(\cdot|\boldsymbol{X},\lambda)||\mathcal{A}(\cdot|\boldsymbol{X}))$, with the intrinsic probability of sequence $\boldsymbol{X}$; that is, we compute the *expected* Kullback–Leibler divergence.

Note further that we now have a well-founded framework for addressing question (i) above: we can use the expected Kullback–Leibler divergence as a measure of *how well* a particular algorithm performs relative to a particular model (i.e. to a range of principal environments).

This framework has been applied by Nielsen *et al.* (2007) to ask and to answer formally, for example, the question of comparing the performances of the maximum-likelihood algorithm of Despotovic & Aberer (2004) and the beta-based algorithm of Mui *et al.* (2002) relative to the simple λ model introduced above, i.e. in estimating the fixed probability θ of honest behaviour of a particular principal.

Nielsen *et al.* (2007) also illustrate how the approach addresses question (ii) above. The two algorithms are generalized to a continuum of algorithms, and it is shown that among this continuum of algorithms, the beta-based algorithm of Mui *et al.* (2002) is optimal for precisely $\theta = 1/2 \pm 1/\sqrt{12}$. And it follows from the

analysis, that the expected Kullback–Leibler divergence is a continuous function of θ, and hence that we can formalize a notion of *robustness* of the quality of the beta-based algorithm of Mui *et al.* (2002) around the particular θ value above.

4. Concluding remarks

In this paper, we have been discussing the role of trust in UbiComp. We believe that trust could be an important ingredient in meeting the UbiComp Grand Challenge, but as stated in Sloman (2006): 'A discipline of trust will only be effective if it is rigorously defined'. We have argued for the need of models as a foundation for asking and answering questions on the performance of systems in computational trust, and we have introduced a few preliminary ideas towards this ambitious goal (for simple probabilistic models).

In doing so, we have followed the view of Samuel Karlin: that the purpose of models is maybe not to fit the data, but rather to sharpen the questions. But good models must do both, and clearly the probabilistic models we have been advocating here need a lot of further improvements in order to be more realistic.

For example, the beta model of principal behaviour (which we consider to be state-of-the-art) assumes that for each principal p there is a single fixed parameter θ_p so at each interaction, *independently of anything else we know*, there is probability θ_p for a 'good' outcome and probability $1-\theta_p$ for a 'bad' outcome. For *some* applications, one might argue that this is unrealistic, for example, (i) the parameter θ_p is fixed, independent of time and (ii) p's behaviour when interacting with us is likely to depend on our behaviour when interacting with p.

As mentioned above, some beta-based reputation systems attempt to deal with the first problem by introducing notions of 'decaying'. The idea is that information about old interactions should weigh less than information about new ones; however, this represents a departure from the probabilistic beta model, where all interactions 'weigh the same'. Since a new model is *not* introduced, i.e. to formalize this preference towards newer information, it is not clear what the exact benefits of forgetting factors are, and more generally when and why to choose between, for example, exponential decay as opposed to say linear decay? Nielsen *et al.* (2007) propose preliminary ideas following the spirit of this paper are introduced, formally modelling the dynamic behaviour of a principal by a hidden Markov model.

The notion of context is also relevant for computational trust models, as has been recognized by many. Given a single-context model, one can obtain a multi-context model by instantiating the single-context model in each context. However, as Sierra & Sabater (2005) argue, this is too naive: the goal of a true multi-context model is not just to *model* multiple contexts, but to provide the basis for transferring information from one context to another related context. To the best of our knowledge, there are no techniques dealing *formally* with this problem within the field of trust and reputation.

References

Abdul-Rahman, A. 2005 A framework for decentralised trust reasoning. PhD thesis, Department of Computer Science, University College London, UK.

Abdul-Rahman, A. & Hailes, S. 2000 Supporting trust in virtual communities. In *Proc. 33rd Ann. Hawaii Int. Conf. on System Sciences*, vol. 9. Washington, DC: IEEE.

Appel, A. W. & Felten, E. W. 1999 Proof-carrying authentication. In *Proc. 6th ACM Conf. on Computer and Communications Security* (*CCS'99*), pp. 52–62. New York, NY: ACM Press.

Blaze, M., Feigenbaum, J. & Lacy, J. 1996 Decentralized trust management. In *Proc. 17th Symposium on Security and Privacy*, pp. 164–173. Los Alamitos, CA: IEEE Computer Society Press.

Blaze, M., Feigenbaum, J. & Strauss, M. 1998 Compliance checking in the PolicyMaker trust management system. In *Proc. Financial Cryptography: 2nd Int. Conf.* (*FC'98*). Lecture Notes in Computer Science, no. 1465, pp. 254–274. Berlin, Germany: Springer.

Blaze, M., Feigenbaum, J., Ionnidis, J. & Keromytis, A. D. 1999*a* The KeyNote trust management system, version 2, RFC-2704. See ftp://ftp.rfc-editor.org/in-notes/rfc2704.txt.

Blaze, M., Feigenbaum, J., Ionnidis, J. & Keromytis, A. D. 1999*b* The role of trust management in distributed systems security. In *Secure internet programming: security issues for mobile and distributed objects* (eds J. Vitek & C. D. Jensen). Lecture Notes in Computer Science, no. 1603, pp. 185–210. Berlin, Germany: Springer.

Blaze, M., Feigenbaum, J. & Keromytis, A. D. 1999*c* KeyNote: trust management for public-key infrastructures. In *Proc. Security Protocols: 6th Int. Workshop*. Lecture Notes in Computer Science, no. 1550, pp. 59–63. Berlin, Germany: Springer.

Blaze, M., Ionnidis, J. & Keromytis, A. D. 2001 Trust management for IPsec. In *Network and Distributed System Security Symposium* (*NDSS'01*) *Conference Proceedings*. See http://www.isoc.org/isoc/conferences/ndss/01/.

Blaze, M., Ionnidis, J. & Keromytis, A. D. 2002 Offline micropayments without trusted hardware. In *Financial Cryptography, 5th Int. Conf.* (*FC'01*) (ed. P. F. Syverson). Lecture Notes in Computer Science, no. 2339, pp. 21–40. Berlin, Germany: Springer.

Buchegger, S. & Le Boudec, J.-Y. 2004 A robust reputation system for peer-to-peer and mobile *ad-hoc* networks. In *Proc. 2nd Workshop on the Economics of Peer-to-Peer Systems*.

Cahill, V. *et al.* 2003 Using trust for secure collaboration in uncertain environments. *IEEE Pervasive Comput.* **2**, 52–61. (doi:10.1109/MPRV.2003.1228527)

Clarke, D., Elien, J.-E., Ellison, C., Fredette, M., Morcos, A. & Rivest, R. L. 2001 Certificate chain discovery in SPKI/SDSI. *J. Comput. Secur.* **9**, 285–322.

Czenko, M., Tran, H., Doumen, J., Etalle, S., Hartel, P. & den Hartog, J. 2005 Nonmonotonic trust management for P2P applications. In *1st Int. Workshop on Security and Trust Management* (*STM*). Electronic Notes in Theoretical Computer Science, vol. 157, pp. 101–116. Amsterdam, The Netherlands: Elsevier.

Dash, R. K., Ramchurn, S. D. & Jennings, N. R. 2004 Trust-based mechanism design. In *Proc. 3rd Int. Conf. on Autonomous Agents and Multi-Agent Systems* (*AAMAS'04*), pp. 748–755. New York, NY: ACM Press.

Dellarocas, C. 2003 The digitization of word of mouth: promise and challenges of online feedback mechanisms. *Manage. Sci.* **49**, 1407–1424. (doi:10.1287/mnsc.49.10.1407.17308)

Dellarocas, C. 2004 Sanctioning reputation mechanisms in online trading environments with moral hazard. Working paper. See http://ccs.mit.edu/dell.

Despotovic, Z. & Aberer, K. 2004 A probabilistic approach to predict peers' performance in P2P networks. In *Proc. 8th Int. Workshop on Cooperative Information Agents* (*CIA 2004*). Lecture Notes in Computer Science, no. 3191, pp. 62–76. Berlin, Germany: Springer.

Despotovic, Z. & Aberer, K. 2006 P2P reputation management: probabilistic estimation vs. social networks. *Comput. Netw.* **60**, 485–500. (doi:10.1016/j.comnet.2005.07.003)

DeTreville, J. 2002 Binder, a logic-based security language. In *Proc. 2002 IEEE Symposium on Security and Privacy* (*S&P 2002*), pp. 105–113. Washington, DC: IEEE Computer Society Press.

Ellison, C., Frantz, B., Lampson, B., Rivest, R., Thomas, B. & Ylonen, T. 1999 SPKI certificate theory, RFC 2693. See ftp://ftp.rfc-editor.org/in-notes/rfc2693.txt.

Feigenbaum, J. & Shenker, S. 2002 Distributed algorithmic mechanism design: recent results and future directions. In *Proc. 6th Int. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications* (*Dial-M'02*), pp. 1–13. New York, NY: ACM Press.

Grandison, T. & Sloman, M. 2000 A survey of trust in internet applications. *IEEE Commun. Surv. Tutorials* **3**, 2–16.

Jim, T. 2001 SD3: a trust management system with certified evaluation. In *Proc. 2001 IEEE Symposium on Security and Privacy*, pp. 106–115. Oakland, CA: IEEE Computer Society Press.

Jøsang, A. 2001 A logic for uncertain probabilities. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* **9**, 279–311.

Jøsang, A. & Ismail, R. 2002 The beta reputation system. In *Proc. 15th Bled Conf. on Electronic Commerce*.

Jøsang, A., Ismail, R. & Boyd, C. 2006 A survey of trust and reputation systems for online service provision. *Decis. Support Syst.* **43**, 618–644. (doi:10.1016/j.dss.2005.05.019)

Kamvar, S. D., Schlosser, M. T. & Garcia-Molina, H. 2003 The eigentrust algorithm for reputation management in P2P networks. In *Proc. 12th Int. Conf. on World Wide Web*, pp. 640–651. New York, NY: ACM Press.

Kreps, D. & Wilson, R. 1982 Reputation and imperfect information. *J. Econ. Theory* **27**, 253–279. (doi:10.1016/0022-0531(82)90030-8)

Krukow, K. 2006 Towards a theory of trust for the global ubiquitous computer. PhD thesis, University of Aarhus, Denmark. See http://www.brics.dk/~krukow.

Krukow, K. & Nielsen, M. 2007 From simulations to theorems: a position paper on research in the field of computational trust. In *Proc. Formal Aspects in Security and Trust, 2006*. Lecture Notes in Computer Science, no. 4691, pp. 96–111. Berlin, Germany: Springer.

Krukow, K., Nielsen, M. & Sassone, V. 2005 A formal framework for concrete reputation-systems with applications to history-based access control. In *Proc. 12th ACM Conf. on Computer and Communications Security* (*CCS'05*), pp. 260–269. New York, NY: ACM Press.

Kullback, S. & Leibler, R. A. 1951 On information and sufficiency. *Ann. Math. Statist.* **22**, 79–86. (doi:10.1214/aoms/1177729694)

Li, N. & Mitchell, J. 2002 Design of a role-based trust-management framework. In *Proc. 2002 IEEE Symposium on Security and Privacy* (*S&P 2002*), pp. 114–130. Washington, DC: IEEE Computer Society Press.

Li, N. & Mitchell, J. 2003*a* Datalog with constraints: a foundation for trust-management languages. In *Proc. 5th Int. Symposium on Practical Aspects of Declarative Languages* (*PADL 2003*). Lecture Notes in Computer Science, no. 2562, pp. 58–73. Berlin, Germany: Springer.

Li, N. & Mitchell, J. 2003*b* A role-based trust-management framework. In *Proc. DARPA Information Survivability Conf. and Exposition* (*DISCEX III*), pp. 201–213. Washington, DC: IEEE Computer Society Press.

Li, N. & Mitchell, J. 2003*c* Understanding SPKI/SDSI using first-order logic. In *Proc. 16th IEEE Computer Security Foundations Workshop* (*CSFW'03*), pp. 89–103. Los Alamitos, CA: IEEE Computer Society Press.

Li, N., Feigenbaum, J. & Grosof, B. N. 1999 A logic-based knowledge representation for authorization with delegation. In *Proc. 9th Computer Security Foundations Workshop* (*CSFW'99*), pp. 162–174. Washington, DC: IEEE Computer Society.

Li, N., Grosof, B. N. & Feigenbaum, J. 2000 A practically implementable and tractable delegation logic. In *Proc. 2000 IEEE Symposium on Security and Privacy* (*S&P 2000*), pp. 27–42. Washington, DC: IEEE Computer Society.

Li, N., Grosof, B. N. & Feigenbaum, J. 2003 Delegation logic: a logic-based approach to distributed authorization. *ACM Trans. Inform. Syst. Secur.* (*TISSEC*) **6**, 128–171. (doi:10.1145/605434.605438)

Marsh, S. P. 1994 Formalising trust as a computational concept. PhD thesis, Department of Computer Science and Mathematics, University of Stirling.

Mui, L., Motashemi, M. & Halberstadt, A. 2002 A computational model of trust and reputation for ebusinesses. In *Proc. 5th Annual Hawaii Int. Conf. on System Sciences* (*HICSS'02*), p. 188. IEEE.

Nielsen, M., Krukow, K. & Sassone, V. 2007 A Bayesian model for event-based trust. *Electronic Notes Theor. Comput. Sci.* **172**, 499–521. (doi:10.1016/j.entcs.2007.02.017)

Papadimitriou, C. H. 2001 Algorithms, games and the internet. In *Proc. 33rd Annual ACM Symposium on Theory of Computing* (*STOC'01*), pp. 749–753. New York, NY: ACM Press.

Ramchurn, S. D., Huyhn, D. & Jennings, N. R. 2004 Trust in multi-agent systems. *Knowl. Eng. Rev.* **19**, 1–25. (doi:10.1017/S0269888904000116)

Sabater, J. & Sierra, C. 2005 Review on computational trust and reputation models. *Artif. Intell. Rev.* **24**, 33–60. (doi:10.1007/s10462-004-0041-5)

Sassone, V., Krukow, K. & Nielsen, M. 2007 Towards a formal framework for computational trust. In *Proc. 5th Int. Symposium on Formal Methods for Components and Objects*. Lecture Notes in Computer Science, no. 2562, pp. 175–184. Berlin, Germany: Springer.

Shmatikov, V. & Talcott, C. 2005 Reputation-based trust management. *J. Comput. Secur.* **13**, 167–190.

Sloman, M. 2006 Ubiquitous computing grand challenge. See http://www-dse.doc.ic.ac.uk/Projects/UbiNet/GC.

Teacy, W. T. L., Patel, J., Jennings, N. R. & Luck, M. 2005 Coping with inaccurate reputation sources: experimental analysis of a probabilistic trust model. In *Proc. 4th Int. Joint Conf. on Autonomous Agents and Multiagent Systems*, pp. 997–1004. New York, NY: ACM Press.

Weeks, S. 2001 Understanding trust management systems. In *Proc. 2001 IEEE Symposium on Security and Privacy*, pp. 94–106. Los Alamitos, CA: IEEE Computer Society Press.

Wilson, R. 1985 Reputations in games and markets. In *Game-theoretic models of bargaining*, pp. 27–62. Cambridge, UK: Cambridge University Press.

Xiong, L. & Liu, L. 2004 PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Trans. Knowl. Data Eng.* **16**, 843–857. (doi:10.1109/TKDE.2004.1318566)

Yu, B. & Singh, M. P. 2002 An evidential model of distributed reputation management. In *Proc. 1st Int. Joint Conf. on Autonomous Agents and Multiagent Systems*, pp. 294–301. New York, NY: ACM Press.