HMM-based Trust Model

Ehab ElSalamouny[1], Vladimiro Sassone[1], and Mogens Nielsen[2]

[1] ECS, University of Southampton, UK

[2] University of Aarhus, Denmark

Abstract. Probabilistic trust has been adopted as an approach to taking security sensitive decisions in modern global computing environments. Existing probabilistic trust frameworks either assume fixed behaviour for the principals or incorporate the notion of 'decay' as an ad hoc approach to cope with their dynamic behaviour. Using Hidden Markov Models (HMMs) for both modelling and approximating the behaviours of principals, we introduce the HMM-based trust model as a new approach to evaluating trust in systems exhibiting dynamic behaviour. This model avoids the fixed behaviour assumption which is considered the major limitation of existing Beta trust model. We show the consistency of the HMM-based trust model and contrast it against the well known Beta trust model with the decay principle in terms of the estimation precision.

1 Introduction

In modern open network systems where principals can autonomously enter and leave the environment at any time, and generally in a global computing environment, any particular principal has incomplete information about other principals currently in the same environment. In such an environment, interactions of a principal A with other principals are not assumed to be at the same level of satisfaction, or even safety, to A. One approach of taking security sensitive decisions in a global computing environment regarding interactions with principals is to adopt the notion of *probabilistic trust*, which can broadly be characterised as aiming to build probabilistic models upon which to base predictions about principals' future behaviours. Using these models, the trust of a principal A in another principal B is the probability distribution, estimated by A, over outcomes of the next interaction with B. Here the estimation process is based on the history of interactions h with the principal B. This notion of trust ensembles the trusting relationship between humans as seen by Gambetta [8].

In many existing frameworks the so-called *Beta model* [12] is adopted. This is a *static* model in the precise sense that the behaviour of any principal is assumed to be representable by a fixed probability distribution over outcomes, invariantly in time. That is each principal p is associated with a fixed real number $0 \leq \Theta_p \leq 1$ indicating the assumption that an interaction involving p yields success with probability Θ_p. Using this assumption, the Beta model for trust is based on applying Bayesian data analysis (see e.g. [20]) to the history of interactions h with a given principal p to estimate the probability Θ_p that an interaction with p yields success. In this framework the family of beta probability density functions (pdfs) is used, as a conjugate prior, together with the data h to derive a posterior beta probability density function for Θ_p. Full explanation can be found in [12, 19].

There are several examples in the literature where the Beta model is used, either implicitly or explicitly, including Jøsang and Ismail's Beta reputation system [12], the systems of Mui et al. [15] and of Buchegger [4], the Dirichlet reputation systems [11], TRAVOS [21], and the SECURE trust model [5]. Recently, the Beta model and its extension to interactions with multiple outcomes (the Dirichlet model) have been used to provide a first formal framework for the analysis and comparison of computational trust algorithms [19, 16, 13]. In practice, these systems have found space in different applications of trust, e.g., online auctioning, peer-to-peer filesharing, mobile ad-hoc routing and online multiplayer gaming.

One limitation of current Beta based probabilistic systems is that they assume a fixed probabilistic behaviour for each principal; that is for each principal, there exists a fixed probability distribution over possible outcomes of its interactions. This assumption of fixed behaviour may not be realistic in many situations, where a principal possibly changes its behaviour over time. Just consider, e.g., the example of an agent which can autonomously switch between two internal states, a normal 'on-service' mode and a 'do-not-disturb' mode. This limitation of the Beta model systems has been recognised by many researchers [12, 4, 22]. This is why several papers have used a '*decay*' principle to favour recent events over information about older ones [12]. The decay principle can be implemented in many different ways, e.g., by a using a finite 'buffer' to remember only the most recent n events, or linear and exponential decay functions, where each outcome in the given history is weighted according to the occurrence time (old outcomes are given lower weights than newer ones). Whilst decay-based techniques have proved useful in some applications, we have shown in [7] that the decay principal is useful (for the purpose of estimating the predictive probability) only when the system behaviour is highly *stable*, that is when it is very unlikely to change its behaviour.

Given the above limitations of existing probabilistic trust systems, we try to develop a more general probabilistic trust framework which is able to cover cases where a principal's behaviour is dynamic. Following the probabilistic view of the behaviour, one can represent the behaviour of a principal p at any time t by a particular state S_t which is characterised by a particular probability distribution over possible outcomes of an interaction with p. If p exhibits a dynamic behaviour, it indeed transits between different states of behaviour. This suggests using a multiple state transition system to represent the whole dynamic behaviour of a principal, where each state is defined by a probability distribution over observables. Since the definition of hidden Markov models (HMMs) coincides with this description, we elect to use HMMs for modelling and approximating the dynamic behaviour of principals.

Aiming at avoiding the assumption of fixed behaviour in Beta systems, we introduce the *HMM-based trust* as a more sophisticated trust model which is capable of capturing the natural dynamism of real computing systems. Instead of modelling the behaviour of a principal by a fixed probability distribution representing one state of behaviour, the behaviour of a principal p is approximated by a finite state HMM γ, called the *approximate behaviour model*. Then, given any sequence of outcomes of interactions with p, the approximate model γ is used to estimate the probability distribution over the potential outcomes of the next interaction with p. We call the resulting probability distribution the *estimated predictive probability distribution* of p under the approximate model γ. Following the existing notion of probabilistic trust, the estimated predictive probability distribution represents the trust in the principal p.

In order to evaluate the quality of the HMM-based trust, we contrast its estimated predictive probability distribution against the *real* predictive probability distribution which depends on the *real* behaviour of the concerned principal. For this purpose, we adopt the relative entropy measure [14, 6]. Relying on this measure we evaluate the *expected estimation error* as a measure for the quality of the trust evaluation. Note that this notion of estimation error has been used for comparison between trust algorithms in other works. See for example [16, 19].

Original contribution of the paper. In this paper we describe the basics of the HMM-based trust model. Namely, the methods of obtaining the approximate behaviour model γ for a principal p, and also estimating the probability distribution over possible outcomes of the next interaction with p using γ. We show that maximising the probability of the observations, using the Baum-Welch algorithm detailed in [2, 18], minimises the expected estimation error and therefore is a consistent method for obtaining the approximate behaviour HMM γ. For the sake of comparison to the traditional Beta trust model with a decay factor, we use Monte-Carlo methods to evaluate the expected estimation error in both cases.

Structure of the paper. The next section briefly describes the Beta trust model and the decay principle. Section 3 provides a basic and precise description for hidden markov models. Subsequently, the basic model of HMM-based trust is described in Section 4. Then it is formally shown in Section 5 that the maximum likelihood estimation, as the basis of the HMM-based trust model, is adequate in the sense that it minimises the expected relative entropy between the real and estimated predictive probability distributions. Section 6 provides an experimental comparison between the HMM-based trust model and the well known Beta trust model with decay factor. Finally we conclude our results in section 7.

2 Beta model with a decay factor

In the Beta trust model introduced by [12] an interaction with any principal yields either success s or failure f. It is also based on the assumption that any interaction with a given principal p yields success with a fixed probability θ_p. Under this assumption a sequence of ℓ outcomes $h_\ell = o_0 \cdots o_{\ell-1}$ is a sequence of Bernoulli trials, and the number of successful outcomes in h_ℓ is probabilistically distributed by a binomial distribution. The objective of the Beta trust model is then to estimate the parameter θ_p given a historical sequence of outcomes h_ℓ.

Using the Bayesian data analysis (see e.g. [20]), θ_p is seen as a random variable whose prior (initial) probability density function (pdf) is updated to a posterior pdf using given observations. With the fact that the beta pdf is a *conjugate prior* to the binomial distribution, the *posterior* pdf of θ given the sequence h is also a beta pdf. The

Beta trust model then gives an estimate for θ_p as the expected value of its posterior beta pdf. This estimate, denoted by $\mathcal{B}(\mathsf{s} \mid h)$, is related to the sequence h_ℓ as follows.

$$\mathcal{B}(\mathsf{s} \mid h_\ell) = \frac{\#_\mathsf{s}(h_\ell) + 1}{\#_\mathsf{s}(h_\ell) + \#_\mathsf{f}(h_\ell) + 2} \tag{1}$$

where $\#_\mathsf{s}(h_\ell)$ and $\#_\mathsf{f}(h_\ell)$ are the numbers of successful and unsuccessful interactions in h_ℓ respectively.

In order to cope with the cases where the behaviour of a principal is dynamic, the notion of exponential decay (or forgetting) has been incorporated in the Beta trust model [12]. The intuitive idea is to capture the most recent behaviour of the principal by favouring the recent outcomes over old ones. This is performed by associating each outcome o_i in h_ℓ with an exponential weight $r^{\ell-i-1}$, where $0 \leq r \leq 1$ is called the decay (forgetting) factor. Observe that recent outcomes are associated with higher weights than older outcomes. With the decay factor r, the Beta estimate for the distribution over $\{\mathsf{s}, \mathsf{f}\}$ is denoted by $\mathcal{B}_r(. \mid h_\ell)$, and given by the following equations.

$$\mathcal{B}_r(\mathsf{s} \mid h_\ell) = \frac{m_r(h_\ell) + 1}{m_r(h_\ell) + n_r(h_\ell) + 2} \quad , \quad \mathcal{B}_r(\mathsf{f} \mid h_\ell) = \frac{n_r(h_\ell) + 1}{m_r(h_\ell) + n_r(h_\ell) + 2} \tag{2}$$

and

$$m_r(h_\ell) = \sum_{i=0}^{\ell-1} r^{\ell-i-1} \delta_i(\mathsf{s}) \qquad n_r(h_\ell) = \sum_{i=0}^{\ell-1} r^{\ell-i-1} \delta_i(\mathsf{f}) \tag{3}$$

for

$$\delta_i(X) = \begin{cases} 1 & \text{if } o_i = X \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

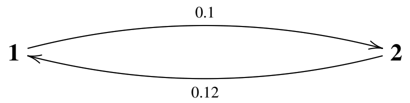
Note that incorporating the decay principle in the Beta trust model is implemented by replacing the counts $\#_\mathsf{s}(h_\ell)$ and $\#_\mathsf{f}(h_\ell)$ in Equation (1) by the sum of weights associated with the past outcomes. Although this approach has been used in many works, we have shown in [7] that it is not effective when the principal's behaviour is highly dynamic; that is when the system tends to change its state of behaviour, characterised by the exhibited probability distribution over possible outcomes. Another major limitation of this approach is that it appears hard to formally determine the optimal value for the decay factor from only observations.

3 Hidden Markov Models (HMMs)

A *Hidden Markov Model (HMM)* [1] is a well-established probabilistic model essentially based on a notion of system state. Underlying any HMM there is a Markov chain modelling (probabilistically) the system's transitions between a set of internal states. Each state in this chain is associated with a particular probability distribution over the set of possible outcomes (observations). The output of an HMM is a sequence of outcomes where each outcome is sampled according to the probability distribution of the underlying state. In the following, we denote the state of the HMM and the observation at time t by q_t and o_t respectively.

Definition 1 (hidden Markov model). A (discrete) *hidden Markov model* (HMM) is a tuple $\lambda = (Q, \pi, A, O, B)$ where Q is a finite set of *states*; π is a distribution on Q, the *initial distribution*; $A : Q \times Q \to [0,1]$ is the *state transition matrix*, with $A_{ij} = P(q_{t+1} = j \mid q_t = i)$ and $\sum_{j \in Q} A_{ij} = 1$; O is a finite set of possible *observations*; and $B : Q \times O \to [0,1]$ is the *observation probability matrix*, with $B_{ik} = P(o_t = k \mid q_t = i)$, $\sum_{k \in O} B_{ik} = 1$.

HMMs provide the computational trust community with several obvious advantages: they are widely used in scientific applications, and come equipped with efficient algorithms for computing the probabilities of events and for parameter estimation (cf. [18]), the chief problem for probabilistic trust management. It is worth noticing that



$\pi_1 = 0.5$		$\pi_2 = 0.5$
$B_{(}1, \mathsf{s}) = 0.95$	$O = \{\mathsf{s}, \mathsf{f}\}$	$B(2, \mathsf{s}) = 0.05$
$B(1, \mathsf{f}) = 0.05$		$B(2, \mathsf{f}) = 0.95$

Fig. 1. Example Hidden Markov Model.

an HMM is a generalisation of the Beta model. Indeed, in the context of computational trust, representing the behaviour of a principal p by a HMM λ_p provides a different distribution B_j over O for each possible state j of p. In particular, the states of λ_p can be seen as a collection of independent Beta models, the transitions between which are governed by the Markov chain formed by π and A, as principal p switches its internal state. According to the above definition of HMM, the probability of a sequence of outcomes $h = o_1 \, o_2 \cdots o_n$ given a HMM λ is given by the following equation.

$$P(h \mid \lambda) = \sum_{q_1, \ldots, q_n \in Q} \pi(q_1) \cdot B_{q_1 o_1} \cdot A_{q_1 q_2} \cdot B_{q_2 o_2} \cdots A_{q_{n-1} q_n} \cdot B_{q_n o_n}$$

The above probability is evaluated efficiently by an algorithm called the *forward-backward algorithm*. One instance of this algorithm, called the *forward* instance, is based on inductively (on time t) evaluating the *forward variable* $\alpha_t(j) = P(o_1 \, o_2 \cdots o_t, \ q_t = j \mid \lambda)$, that is the joint probability that the partial sequence $o_1 \, o_2 \cdots o_t$ is observed and the state at time t is j. The required probability $P(h \mid \lambda)$ is then obtained by

$$P(h \mid \lambda) = \sum_{j \in Q} \alpha_n(j)$$

Alternatively, $P(h \mid \lambda)$ can be obtained using the *backward* instance of the algorithm, where the *backward* variable $\beta_t(j) = P(o_{t+1} \, o_{t+2} \cdots o_n, \mid q_t = j, \lambda)$ is inductively (on time t) evaluated. More details on these instances of the forward-backward algorithm can be found in [18].

Another major problem of HMMs is to find the model λ which maximises the above probability of a sequence h. This problem has been addressed by Baum and his colleagues whose efforts resulted in the *Baum-Welch algorithm* [2, 18]. This algorithm iteratively estimates the parameters of an HMM λ which maximises the probability of a given sequence of outcomes h. One limitation of this algorithm is that it finds a local maxima in the model space rather than the global one.

Example 1. Figure 1 shows a two-state HMM over the observation set $\{\mathsf{s}, \mathsf{f}\}$. Both states are relatively stable. That is the probability of making both transitions $\mathbf{1} \mapsto \mathbf{2}$ and $\mathbf{2} \mapsto \mathbf{1}$ are relatively small (0.1,0.12 respectively). Also at state **1**, it is very likely to observe s (with probability 0.95), whereas at state **2** it is very likely to observe f (with probability 0.95). This HMM describes the behaviour of a stable principal whose internal state is unlikely to change.

In the area of trust, we remark that Markovian models have also been used in [10] to model the evolution of trust in the users of collaborative information systems. However, in our work, HMMs model the principal's behaviour upon which trust is computed.

4 HMM-based trust model

As described in the introduction, the HMM-based trust relies on approximating the behaviour of any given principal by a finite-state HMM γ called the approximate behaviour model. The approximate behaviour model is then used to estimate the predictive probability distribution. In order to precisely define this model, it is required to define a method for computing the approximate behaviour model γ, and also for estimating the predictive probability distribution using γ. As a general notation which will be used in these definitions we will write the probability of any random variable ζ, under a given probabilistic model $\mathcal{R}$, as $P(\zeta \mid \mathcal{R})$.

For computing γ, the maximum likelihood criterion is adopted as follows. Let $y = y_0\, y_2 \cdots y_{\ell-1}$ be an observed sequence of outcomes of interactions with a given principal, where ℓ is an arbitrary length. Let also $\mathcal{R}_n$ denote any n-state HMM. Then, using the sequence y, the n-state approximate behaviour model γ is obtained by the following equation.

$$\gamma = \operatorname*{argmax}_{\mathcal{R}_n} P(h_\ell = y \mid \mathcal{R}_n) \tag{5}$$

That is γ is the n-state HMM under which the probability of the given history y is maximised. The HMM model γ can be therefore obtained by the Baum-Welch algorithm which is described briefly in Section 3 and detailed in [2, 18].

Now we address the problem of estimating the predictive probability distribution given a particular sequence of outcomes. Let $h_\ell = o_0\, o_1 \cdots o_{\ell-1}$ be a random variable representing any sequence of observed outcomes of interaction with the principal p, where o_0 and $o_{\ell-1}$ represent respectively the least and the most recent outcomes, and ℓ is an arbitrary length. Extending this notation to future outcomes, the outcome of the next interaction with p is denoted by o_ℓ. Note that each outcome o_i is therefore a random variable representing the outcome at time i. Let also $O = \{1, 2, \ldots, \kappa\}$ be the alphabet of each single outcome. Using the n-state approximate behaviour HMM γ defined by Equation (5), the estimated predictive probability distribution given a particular sequence of outcomes w is denoted by $\mathcal{H}_{\gamma}(. \mid w)$ and defined by the following equation.

$$\mathcal{H}_{\gamma}(z \mid w) = P(o_{\ell} = z \mid h_{\ell} = w, \gamma) = \frac{P(h_{\ell} = w, o_{\ell} = z \mid \gamma)}{P(h_{\ell} = w \mid \gamma)} \tag{6}$$

where $z \in O$. The above probabilities are efficiently evaluated by the forward-backward algorithm briefly described in Section 3, and detailed in [18].

5 Consistency of maximum likelihood estimation

Like other existing probabilistic trust models, the objective of the HMM-based trust model is to estimate the predictive probability distribution for a given principal p, that is the probability of each possible outcome in the next interaction with p. Therefore it is a fundamental requirement that the approximate behaviour model γ computed for p is chosen such that the error of such an estimation is minimised.

To analyse this error, we need to model the real behaviour of the principal p. This allows expressing the *real* predictive probability distribution of p. The estimation error can be therefore evaluated as the difference between the real and estimated predictive probability distributions. In this section it is shown that the maximum likelihood criterion, defined by Equation (5) for choosing the approximate behaviour model provides a consistent method to minimise the estimation error.

5.1 Modelling the Real System

In this work we are interested in studying systems which exhibit a dynamic behaviour, that is changing their behaviour over time. We mathematically model the behaviour of the system at any time by a particular probability distribution over possible outcomes. A system p with a dynamic behaviour can be therefore modelled by a multiple state transition system where each state exhibits a particular behaviour (probability distribution). This naturally leads to choosing a generic Hidden Markov Model (HMM) λ as the real model of p's behaviour.

Here the state of a system real model λ at the time of observing o_i is denoted by the random variable q_i. Thus, given that the current underlying state is x, i.e. $q_{\ell-1} = x$, we can compute the real predictive probability distribution, denoted by $P(. \mid x, \lambda)$, that is the probability of each possible next observation, $z \in O$, using the following equation.

$$\begin{aligned} P(z \mid x, \lambda) &= P(o_{\ell} = z \mid q_{\ell-1} = x, \lambda) \\ &= \sum_{y \in Q_{\lambda}} P(q_{\ell} = y \mid q_{\ell-1} = x, \lambda)\, P(o_{\ell} = z \mid q_{\ell} = y, \lambda) \\ &= \sum_{y \in Q_{\lambda}} (A_{\lambda})_{xy}\, (B_{\lambda})_{yz} \end{aligned} \tag{7}$$

where Q_{λ}, A_{λ}, and B_{λ} are respectively the set of states, the state transition matrix, and the observation probability matrix of λ. We shall also work under the hypothesis that λ is *ergodic*. This corresponds to demanding that the Markov chain underlying λ is irreducible and aperiodic (more details on these properties can be found in [9, 17, 3]).

5.2 The estimation error

In this paper the relative entropy measure [6] is used for evaluating the difference between the real and estimated predictive probability distributions, given by Equations (7) and (6) respectively. Namely, given a sequence of outcomes $h_\ell = w$ and the current state $q_{\ell-1} = x$, this difference measure is written as follows.

$$D\left(P\left(. \mid x, \lambda\right) \parallel \mathcal{H}_\gamma\left(. \mid w\right)\right) = \sum_{z \in O} P\left(z \mid x, \lambda\right) \, \log\left(\frac{P\left(z \mid x, \lambda\right)}{\mathcal{H}_\gamma\left(z \mid w\right)}\right) \tag{8}$$

The above difference can be seen as the *estimation error* given a particular current state $q_{\ell-1}$ of λ, and the sequence of outcomes h_ℓ. Hence we define the *expected estimation error* as the expected relative entropy between the real and estimated predictive probability distributions, where the expectation is evaluated on the underlying random variables $q_{\ell-1}$ and h_ℓ. This error is denoted by $Error_\ell\left(\lambda, \mathcal{H}_\gamma\right)$. Thus,

$$Error_\ell\left(\lambda, \mathcal{H}_\gamma\right) = \mathbf{E}\left[D\left(P\left(. \mid q_{\ell-1}, \lambda\right) \parallel \mathcal{H}_\gamma\left(. \mid h_\ell\right)\right)\right] \tag{9}$$

Now we formally show that choosing the approximate behaviour model γ by maximising the likelihood of a given sufficiently long sequence y (by Equation (5)) minimises the expected estimation error.

Equation (9) can be written as follows.

$$Error_\ell\left(\lambda, \mathcal{H}_\gamma\right) = \sum_{w \in O^\ell} \sum_{x \in Q_\lambda} P\left(h_\ell = w, q_{\ell-1} = x \mid \lambda\right) \cdot \\ \cdot D\left(P\left(. \mid x, \lambda\right) \parallel \mathcal{H}_\gamma\left(. \mid w\right)\right) \tag{10}$$

Using Equation (8) we rewrite the above equation.

$$Error_\ell\left(\lambda, \mathcal{H}_\gamma\right) = \sum_{w \in O^\ell} \sum_{x \in Q_\lambda} P\left(h_\ell = w, q_{\ell-1} = x \mid \lambda\right) \cdot \\ \cdot \sum_{z \in O} P\left(z \mid x, \lambda\right) \log\left(\frac{P\left(z \mid x, \lambda\right)}{\mathcal{H}_\gamma\left(z \mid w\right)}\right) \tag{11}$$

Substituting $P\left(z \mid x, \lambda\right)$ and $\mathcal{H}_\gamma\left(z \mid w\right)$ using Equations (7) and (6) respectively, we write the above equation as follows.

$$\begin{aligned} Error_\ell\left(\lambda, \mathcal{H}_\gamma\right) &= \sum_{w \in O^\ell} \sum_{x \in Q_\lambda} P\left(h_\ell = w, q_{\ell-1} = x \mid \lambda\right) \cdot \\ &\quad \cdot \sum_{z \in O} P\left(o_\ell = z \mid q_{\ell-1} = x, \lambda\right) \log\left(\frac{P\left(o_\ell = z \mid q_{\ell-1} = x, \lambda\right)}{P\left(o_\ell = z \mid h_\ell = w, \gamma\right)}\right) \\ &= \sum_{w \in O^\ell} \sum_{x \in Q_\lambda} \sum_{z \in O} P\left(o_\ell = z \mid q_{\ell-1} = x, \lambda\right) \cdot \\ &\quad \cdot P\left(h_\ell = w, q_{\ell-1} = x \mid \lambda\right) \log\left(\frac{P\left(o_\ell = z \mid q_{\ell-1} = x, \lambda\right)}{P\left(o_\ell = z \mid h_\ell = w, \gamma\right)}\right) \end{aligned} \tag{12}$$

Since the next outcome o_ℓ depends only on the current state $q_{\ell-1}$ regardless of the history sequence h_ℓ, we have

$$P(o_\ell = z \mid q_{\ell-1} = x, \lambda) = P(o_\ell = z \mid h_\ell = w, q_{\ell-1} = x, \lambda) \tag{13}$$

Thus Equation (12) becomes

$$\begin{aligned} Error_\ell\left(\lambda, \mathcal{H}_\gamma\right) &= \sum_{w\in O^\ell} \sum_{x\in Q_\lambda} \sum_{z\in O} P(o_\ell = z \mid h_\ell = w, q_{\ell-1} = x, \lambda) \cdot \\ &\qquad \cdot P(h_\ell = w, q_{\ell-1} = x \mid \lambda) \log\left(\frac{P(o_\ell = z \mid q_{\ell-1} = x, \lambda)}{P(o_\ell = z \mid h_\ell = w, \gamma)}\right) \\ &= \sum_{w\in O^\ell} \sum_{x\in Q_\lambda} \sum_{z\in O} P(o_\ell = z, h_\ell = w, q_{\ell-1} = x \mid \lambda) \log\left(\frac{P(o_\ell = z \mid q_{\ell-1} = x, \lambda)}{P(o_\ell = z \mid h_\ell = w, \gamma)}\right) \end{aligned} \tag{14}$$

The above equation can be simplified to the following equation.

$$Error_\ell\left(\lambda, \mathcal{H}_\gamma\right) = \mathbf{E}\left[\log P(o_\ell \mid q_{\ell-1}, \lambda)\right] - \mathbf{E}\left[\log P(o_\ell \mid h_\ell, \gamma)\right] \tag{15}$$

Observe that the first term in the above equation depends only on the real behaviour model λ, while the second term depends on both the real and approximate behaviour models λ and γ. Denoting the first and second terms respectively by $C_\ell(\lambda)$ and $H_\ell(\lambda, \gamma)$, we rewrite the above equation as following.

$$Error_\ell\left(\lambda, \mathcal{H}_\gamma\right) = C_\ell(\lambda) - H_\ell(\lambda, \gamma) \tag{16}$$

Assuming that $(A_\gamma)_{ij} > 0$, that is the state transition probabilities of γ are strictly positive, it has been proved by Baum and Petrie in [1] that the following limit exists.

$$\lim_{\ell\to\infty} H_\ell(\lambda, \gamma) = H(\lambda, \gamma) \tag{17}$$

Observe also that the limit $\lim_{\ell\to\infty} C_\ell(\lambda) = C(\lambda)$ exists. This is because the ergodicity of λ implies that the distribution of the random variable $q_{\ell-1}$ converges to a stationary (fixed) distribution according to which the expectation $\mathbf{E}\left[\log P(o_\ell \mid q_{\ell-1}, \lambda)\right]$ is evaluated. The convergence of both $C_\ell(\lambda)$ and $H_\ell(\lambda, \gamma)$ implies the convergence of the estimation error (as $\ell \to \infty$) to an *asymptotic estimation error* denoted by $Error\left(\lambda, \mathcal{H}_\gamma\right)$, and expressed as follows.

$$Error\left(\lambda, \mathcal{H}_\gamma\right) = C(\lambda) - H(\lambda, \gamma) \tag{18}$$

Also, (by Theorem 3.2 in [1]) the log-probability of any observation sequence h_ℓ is related to $H(\lambda, \gamma)$ as follows.

$$\frac{1}{\ell} \log P(h_\ell \mid \gamma) \overset{a.s.}{\to} H(\lambda, \gamma) \tag{19}$$

The above equation means that the log-probability of a random sequence h_ℓ under the approximate model γ, divided by its length converges *almost surely* to $H(\lambda, \gamma)$. Here *'almost surely'* (also known as *'almost everywhere'* and *'with probability 1'*) convergence means that the probability that the function $\frac{1}{\ell} \log P(h_\ell \mid \gamma)$ converges to the above limit is 1. That is

$$P\left(\lim_{\ell \to \infty} \frac{1}{\ell} \log P(h_\ell \mid \gamma) = H(\lambda, \gamma)\right) = 1$$

Equation (19) implies that choosing an approximate model γ which maximises the probability of a sufficiently long sequence h_ℓ almost surely maximises $H(\lambda, \gamma)$, and therefore reduces the asymptotic estimation error given by Equation (18). Thus, the maximum data likelihood criterion, expressed by Equation (5) is a consistent method to obtain the approximate behaviour model, which is used to estimate the predictive probability distribution.

6 Comparison with Beta-based trust with decay principle

In this section we contrast the HMM-based trust model described above against the existing Beta trust model with exponential decay, described in [12] and Section 2 in terms of the expected estimation error. Here the estimation error is defined as the relative entropy between the real and estimated predictive probability distributions. In Section 5.2 above, we used the results obtained by Baum and Petrie in [1] to derive an expression for the expected estimation error (see Equation (16)). It appears difficult to evaluate this error analytically, or even numerically. So we use a simulation framework for HMMs to simulate the real model and adopt Monte Carlo methods to evaluate the estimation error using both HMM-based and Beta-based trust models, and therefore perform the comparison.

6.1 Evaluation of estimation error using Monte Carlo simulation

In general, any probabilistic trust model is described by an *estimating algorithm* A_σ, with a parameter σ. The estimating algorithm is fed with any observation sequence h generated by the real system λ and computes an estimated predictive probability distribution denoted by $A_\sigma(. \mid h)$. In the case of Beta trust model, the estimating algorithm is denoted by $\mathcal{B}_r$, where the parameter r is the decay factor, and the estimated predictive probability distribution $\mathcal{B}_r(. \mid h)$ is evaluated by Equations (2). In the case of HMM-based trust model, on the other hand, the estimating algorithm is denoted by $\mathcal{H}_\gamma$, where the parameter γ is an approximate behaviour HMM. Note that the parameter γ is obtained by maximising the probability of any sufficiently long sequence y generated by λ as shown in Section 4. The estimated predictive probability distribution $\mathcal{H}_\gamma(. \mid h)$ is evaluated by Equation (6).

Given a real HMM model λ, let the random variables h_ℓ denote any generated sequence of observations of length ℓ. Let also the random variable q_ℓ denote the underlying hidden state sequence. Given an estimating algorithm A_σ (e.g. $\mathcal{B}_r$ or $\mathcal{H}_\gamma$), the expected estimation error using A_σ is given by the following equation.

$$Error_\ell(\lambda, A_\sigma) = \mathbf{E}\left[D\left(P(. \mid q_\ell, \lambda) \parallel A_\sigma(. \mid h_\ell)\right)\right] \tag{20}$$

The above expected error can be approximated by the following Monte-Carlo procedure.

1. Simulate the real model λ to generate a large sample S_m of size m:

$$S_m = \{(w_1, u_1), (w_2, u_2), \ldots, (w_m, u_m)\}$$

   where w_j and u_j are respectively the observation sequence, and the underlying state sequence generated in the jth simulation run.
2. For each pair $\left(w_j, u_j\right)$,
   (a) compute both $P\left(. \mid u_j, \lambda\right)$ and $A_\sigma\,(. \mid h_\ell)$, that is the real and estimated predictive probability distributions, respectively.
   (b) Evaluate the estimation error, denoted by e_j, as

$$e_j = D\left(P\left(. \mid u_j, \lambda\right) \parallel A_\sigma\left(. \mid w_j\right)\right) \tag{21}$$

3. Approximate the required expected estimation error by evaluating the sample average.

$$Error_\ell\,(\lambda, A_\sigma) \approx \frac{1}{m}\sum_{j=1}^{m} e_j \tag{22}$$

The above approximation of the expected estimation error by the sample average is based on the law of large numbers. Note that the approximation error can be made arbitrarily small by making the sample size m sufficiently large.

6.2 Experiments

Throughout our comparison we will a 4-state real model λ with the observation alphabet $O = \{1, 2\}$, the observation probability matrix is

$$B_\lambda = \begin{bmatrix} 1.0 & 0.0 \\ 0.7 & 0.3 \\ 0.3 & 0.7 \\ 0.0 & 1.0 \end{bmatrix} \tag{23}$$

and the state transition matrix is

$$A_\lambda = \begin{bmatrix} s & \frac{1-s}{3} & \frac{1-s}{3} & \frac{1-s}{3} \\ \frac{1-s}{3} & s & \frac{1-s}{3} & \frac{1-s}{3} \\ \frac{1-s}{3} & \frac{1-s}{3} & s & \frac{1-s}{3} \\ \frac{1-s}{3} & \frac{1-s}{3} & \frac{1-s}{3} & s \end{bmatrix} \tag{24}$$

where the parameter s is called the *system stability*, which indicates the tendency of the system to staying in the same state rather than transiting to a different one.

In the following experiments, we study the effect of the system stability on both Beta estimation with a decay factor and HMM based estimation. For simplicity we confine our HMM-based trust model to use only 2-state approximate behaviour models. We also base our trust estimation on sequences of length 300. For different stability values $0 \leq s < 1$ and decay values $0 \leq r \leq 1$, we apply the Monte-Carlo procedure described above to evaluate the expected estimation error using both Beta ($\mathcal{B}_r$) and HMM ($\mathcal{H}_\gamma$) trust algorithms. Each generated sample is of size 10000.

Figure 2 shows Beta and HMM estimation errors when the system λ is unstable ($s < 0.5$). It is obvious that the minimum error value for Beta error is obtained when the decay tends to 1. The reason for this is that an unstable system is relatively unlikely to stay in the same state, and therefore unlikely to preserve the previous distribution over observations. If the estimation uses low values for the decay, then the resulting estimate for the predictive probability distribution is close to the previous distribution; this is unlikely to be the same as in the next time instant, due to instability. On the other hand, using a decay r tending to 1 favours equally all previous observations, and the resulting probability distribution is expected to be the average of the distributions exhibited by the model states. Such an average provides a better estimate for the predictive probability distribution than approximating the distribution of the most recent set of states using low decay values.

It is also obvious that the HMM estimation error is lower than Beta estimation error. The reason is that the 2-state HMM γ is a more flexible model to approximate the real HMM λ than the Beta model which is, with decay 1, equivalent to 1-state HMM model. It is worth noting that when stability is 0.25, the minimum expected beta error is 0, when the decay is 1. The HMM-estimation error is also approximately 0. In this case all elements of the transition matrix A_λ are equal and therefore, the whole behaviour can effectively be modelled by a single probability distribution over observations. This single probability distribution is perfectly approximated by taking the whole history into account using Beta model with decay 1, and also with 2-state HMM where both states are equivalent.

Figure 3 shows Beta and HMM estimations errors when the system λ is stable (stability > 0.5). Observe that both Beta with decay 1 and HMM estimation errors are increasing as the stability is higher. The reason is that, at relatively high stability, old observations become irrelevant to the current behaviour which determines the real predictive probability distribution. Hence, the estimation based on the whole history using HMM or Beta with decay 1 is worse than the estimation with the same parameters when the system is unstable, where both old and recent outcomes are relevant to the current behaviour.

Observe also in the cases of high stability that HMM based estimation is better than Beta estimation for most values of decay. However, for a particular range of decay, Beta estimation is slightly better than HMM estimation. Using any decay value in this range for Beta estimation has the effect of considering only relatively recent outcomes which characterize the current system behaviour and therefore give a better estimation for the predictive distribution. Although using any value from this specific range of decay makes Beta estimation better than HMM estimation, it appears hard to formally determine this range given only observations. When the stability is 1, the assumption of

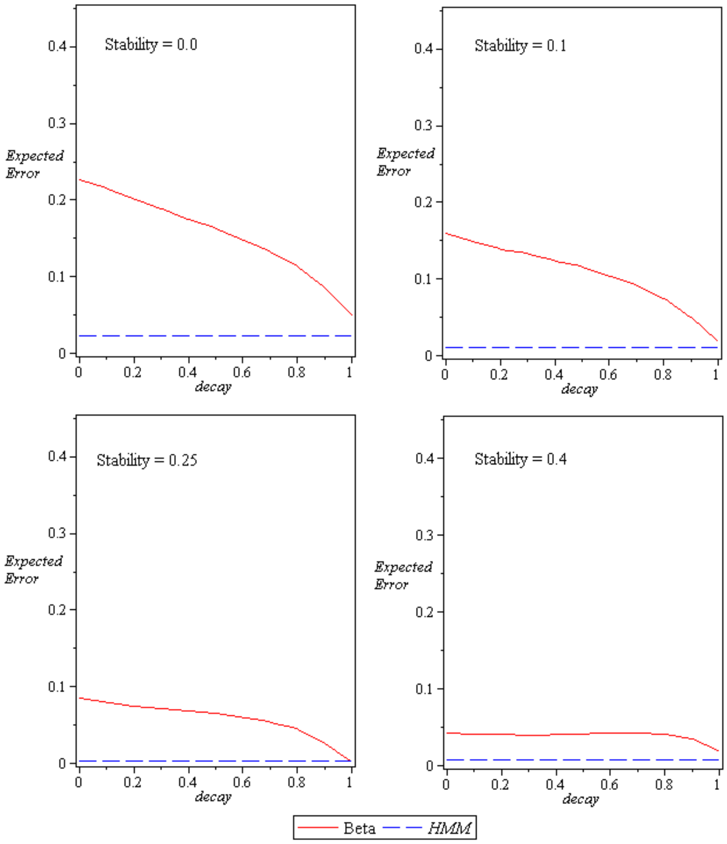


Fig. 2. Beta and HMM estimation errors versus decay factor given stability < 0.5

irreducibility is violated (see Section 5.1). In this case any sequence y of observations characterises only one single state and therefore the approximate behaviour model γ trained on y fails to approximate the whole behaviour of the real system.

7 Conclusion

In this paper we introduced the foundations for the HMM-based trust model. This model is based on approximating the behaviour of the principal by the n-states HMM γ which maximises the likelihood of the available history of observations. The approximate behaviour model γ is then used to evaluate the estimated predictive probability distribution given any sequence of observations. Modelling the real dynamic behaviour of principals by hidden Markov models, and using the results obtained by Baum and Petrie in [1], we justified the consistency of the HMM-based trust model. This justification relies on showing that maximising the likelihood of a given observation sequence minimises the relative entropy between the real and estimated predictive probability distributions.

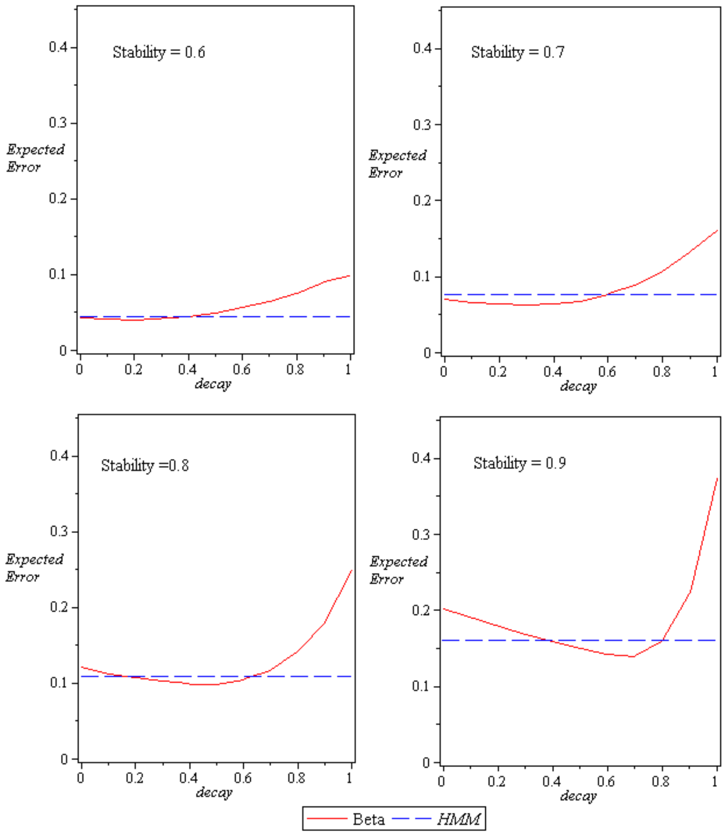


Fig. 3. Beta and HMM estimation errors versus decay factor given stabilities 0.6, 0.7, 0.8, and 0.9

To assess the estimation quality of a particular trust algorithm, we use the notion of expected estimation error that is the expected difference between the real and predictive probability distribution. Since we have no means yet to evaluate the expected estimation error expressed by Equation (18) for the HMM-based trust model using analytical or numerical methods, we use a Monte-Carlo algorithm, described in Section 6.1, for evaluating the expected estimation error.

Using an implementation of this algorithm, and adopting the relative entropy as a measure for the estimation error, we performed an experimental comparison between HMM-based trust algorithm and the Beta-based trust algorithm with an exponential decay scheme. The results of this comparison are given in Section 6.2. These results shows that HMM-based trust algorithm gives a better estimation for the predictive probability distribution when the principal behaviour is highly dynamic. When the real behaviour is more stable (less dynamic), the Beta-based algorithm with the optimal value of decay gives slightly better estimation than the HMM-based algorithm.

References

1. L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite-state Markov chains. *Annals of Mathematical Statistics*, 37(6):1554–1563, Dec 1966.
2. L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The Annals of Mathematical Statistics*, 41(1):164–171, 1970.
3. P. Brémaud. *Markov chains: Gibbs fields, Monte Carlo simulation, and queues*. Springer, 1998.
4. S. Buchegger and J.-Y. Le Boudec. A Robust Reputation System for Peer-to-Peer and Mobile Ad-hoc Networks. In *P2PEcon 2004*, 2004.
5. V. Cahill, E. Gray, J.-M. Seigneur, C. D. Jensen, Y. Chen, B. Shand, N. Dimmock, A. Twigg, J. Bacon, C. English, W. Wagealla, S. Terzis, P. Nixon, G. di Marzo Serugendo, C. Bryce, M. Carbone, K. Krukow, and M. Nielsen. Using trust for secure collaboration in uncertain environments. *IEEE Pervasive Computing*, 2(3):52–61, 2003.
6. T. M. Cover and J. A. Thomas. *Elements of Information Theory 2nd Edition*. Wiley Series in Telecommunications and Signal Processing. Wiley-Interscience, July 2006.
7. E. ElSalamouny, K. Krukow, and V. Sassone. An analysis of the exponential decay principle in probabilistic trust models. *Theoretical Computer Science*, 410(41):4067 – 4084, 2009.
8. D. Gambetta. *Can We Trust Trust?* Basil Blackwell, 1988.
9. G. Grimmet and D. Stirzaker. *Probability and Random Processes*. Oxford University Press, third edition, 2001.
10. S. Javanmardi and C. V. Lopes. Modeling trust in collaborative information systems. *International Conference on Collaborative Computing: Networking, Applications and Worksharing*, 0:299–302, 2007.
11. A. Jøsang and J. Haller. Dirichlet reputation systems. In *The Second International Conference on Availability, Reliability and Security, 2007. ARES 2007.*, pages 112–119, 2007.
12. A. Jøsang and R. Ismail. The beta reputation system. In *Proceedings from the 15th Bled Conference on Electronic Commerce, Bled*, 2002.
13. K. Krukow, M. Nielsen, and V. Sassone. Trust models in Ubiquitous Computing. *Philosophical Transactions of the Royal Society A*, 366(1881):3781–3793, 2008.
14. S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, March 1951.
15. L. Mui, M. Mohtashemi, and A. Halberstadt. A computational model of trust and reputation (for ebusinesses). In *Proceedings from 5th Annual Hawaii International Conference on System Sciences (HICSS'02)*, page 188. IEEE, 2002.
16. M. Nielsen, K. Krukow, and V. Sassone. A bayesian model for event-based trust. *Festschrift in hounour of Gordon Plotkin*, 2007. Electronic Notes in Theoretical Computer Science.
17. J. R. Norris. *Markov chains*. Cambridge University Press, 1997.
18. L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.
19. V. Sassone, K. Krukow, and M. Nielsen. Towards a formal framework for computational trust. In F. S. de Boer, M. M. Bonsangue, S. Graf, and W. P. de Roever, editors, *FMCO*, volume 4709 of *Lecture Notes in Computer Science*, pages 175–184. Springer, 2006.
20. D. S. Sivia. *Data Analysis: A Bayesian Tutorial (Oxford Science Publications)*. Oxford University Press, July 1996.
21. W. Teacy, J. Patel, N. Jennings, and M. Luck. Travos: Trust and reputation in the context of inaccurate information sources. *Autonomous Agents and Multi-Agent Systems*, 12(2):183–198, March 2006.
22. L. Xiong and L. Liu. PeerTrust: Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Transactions on knowledge and data engineering*, 16(7):843–857, 2004.