Using Change Detection to Adapt to Dynamically Changing Trustees

Elham Parhizkar[†,*], Mohammad Hossein Nikravan[†], Robert C. Holte[‡], Sandra Zilles[†]
[†] Department of Computer Science, University of Regina, Regina, Canada
[‡] Department of Computing Science, University of Alberta, Edmonton, Canada

Abstract

In multi-agent systems, agents often have to rely on interactions with other agents in order to accomplish a given task. They hence need to assess the trustworthiness of other agents, which is particularly difficult if the latter change their behavior dynamically. The two common techniques to solve this problem are Hidden Markov Models (HMMs) and standard Beta Reputation Systems (BRS) equipped with a simple decay mechanism to discount older interactions. We propose instead to use *Page-Hinkley* statistics in BRS to detect and dismiss an agent whose behavior worsens. Our experimental study demonstrates that our method outperforms HMMs and, in the vast majority of tested settings, either outperforms or is on par with other typically used BRS-type methods.

Keywords: Multi-Agent Systems, Trust and Reputation Systems, Page-Hinkley Test

1. Introduction

In multi-agent systems, an agent often needs to estimate another agent's trustworthiness, i.e., the probability that an interaction with the agent is successful. A truster is an agent that tries to assess the trustworthiness of another agent, called the trustee. Trustworthiness is typically estimated based on the trustee's past behavior. Many approaches to estimate trust assume that trustees are *static*, i.e., their behavior follows a stationary model [1–3]. For example, in Beta-based models, any trustee's behavior is modeled by a fixed probability distribution over outcomes. In reality, this assumption often does not hold since trustees may change their trustworthiness over time. We call such trustees *dynamic*.

One approach to handle dynamic trustees is to use an exponential decay function, where older outcomes are assigned less weight than more recent ones [4]. So, one of the early efforts on the problem of dynamic trustees is to improve the Beta-based trust models by adopting a *forgetting (decay) factor* [1]. Although this approach shows success in specific scenarios, it is not effective when a trustee's behavior is highly dynamic [5], or when a trustee is deceptive, acting with a high trustworthiness for a certain amount of time to earn trust, and then abruptly switching to a low trustworthiness [6]. Another limitation of this technique is that it is challenging to find the optimal value for the forgetting factor [4]. In trust systems based on Hidden Markov Models (HMMs) [4, 7–9], a trustee's behavior is represented by a certain state with a particular probability distribution over possible interaction outcomes with that trustee. The trustee can change its behavior by transiting between states.

Some studies show that HMM-based trust models perform better than the Beta model with a forgetting factor [4, 6]. However, this approach faces several issues. First of all, the simulations presented in these studies [4, 6] are not comprehensive enough to evaluate the performance of the methods fully. Moe *et al.* [6] applied all the tested scenarios on a single trustee only. Also, an HMM-based trust model is more complex and has more parameters than the Beta model with a forgetting factor. This leads to challenges in estimating the parameter values [6]. In most of the existing HMM-based trust models, past interaction outcomes are used as the HMM observation sequence. However, the authors in [9] claim

[*] parhizkar@uregina.ca

that such HMM models are not effective when a trustee changes its behavior infrequently or in implicit patterns. Another major problem is *data sparsity* when there are not enough interactions to train an HMM model.

We propose a straightforward method to handle dynamic trustees by extending the Beta reputation system (BRS) with the Page-Hinkley (PH) test. The PH test is a well-known change detector in signal processing [10, 11]. To the best of our knowledge, the only existing method using suchlike statistical test for detcting dynamic trustees is RaPTaR [12], which uses the Kolmogorov-Smirnov test instead of PH.

Our method is specifically designed for cases in which it is desirable to weed out trustees whose behavior rapidly turns bad. The results of our extensive experiments demonstrate that this new method either outperforms or is on par with all other tested methods, with very few exceptions.

Especially noteworthy is the simplicity of our system. It significantly outperforms an HMM method [4] and RaPTaR [12], while at the same time being much simpler in design and implementation. This is important because it demonstrates that a complex system is not necessary to achieve state-of-the-art performance in handling dynamic trustees.

2. Related Work

Various approaches have been proposed to handle dynamic trustees [1, 4, 7–9, 13, 14]. One of the early approaches to cope with this issue is to extend the Beta reputation system (BRS) by incorporating an exponential forgetting factor [1]. The intuitive idea is to give an effective bias towards a trustee's recent behavior. This is done by scaling the parameters (α, β) of the Beta distribution with the help of a forgetting factor $0 \leq r \leq 1$, so that

$$\alpha_t = r\alpha_{t-1} + p_t, \quad \beta_t = r\beta_{t-1} + n_t, \quad \alpha_0 = \beta_0 = 1, \tag{2.1}$$

where $p_t = 1$ and $n_t = 0$ ($p_t = 0$ and $n_t = 1$, resp.) in case of a positive (negative, resp.) outcome at time step t. When $r = 1$, the weights of all outcomes are equal and when $r = 0$, only the last outcome is remembered.

Zhang and Cohen [13] handle trustees' dynamic behavior by dividing the interaction outcomes with a trustee into different elemental *time windows*. In each time window, a truster counts the numbers of positive and negative interactions. To estimate the trustworthiness value of a trustee, the numbers of positive and negative interactions in each time window are aggregated by considering the forgetting factor. A similar approach is proposed by Liu and Datta [14]. Specifically, a set of past interaction outcomes with a trustee is partitioned into sequential subsets called *transaction windows*. Then, the similarity between the most recent transaction window and earlier transaction windows is calculated. To estimate the outcome of the potential interaction, the outcomes of past similar interactions are used.

Several HMM-based approaches have been proposed to cope with dynamic behavior [4, 7, 9, 15]. Moe et al. [7] proposed a continuous-time HMM to model trust in a marketplace. In this framework, a truster keeps and updates an HMM per trustee. At each time, a trustee could be in a *trustworthy*, *neutral*, or *non-trustworthy* state. This method uses the past interaction outcomes as the observation sequence of an HMM. After learning the parameters using the Baum-Welch algorithm [16], the trustworthiness values of trustees are estimated by calculating the most probable state of each truster-trustee HMM using the forward algorithm [17]. After an interaction, the HMM belonging to the selected trustee is updated based on the interaction outcome. In [4], the behavior of a trustee is estimated by a discrete HMM η known as *approximate behavior model*. η will be trained using a sequence of interaction outcomes up to interaction t with a trustee s. Then, the probability distribution over possible outcomes of interaction $t+1$ is approximated using η. The obtained probability distribution is called the *estimated predictive probability distribution* of s.

Liu and Datta [9] proposed a trust model using a contextual information-based HMM to handle agents' dynamic behavior. Unlike most of the published HMM-based trust models, which take the history of interaction outcomes as observations of an HMM, this model considers interaction context (additional features of interactions) as observations. Contextual information of interaction is a set of features that characterizes that interaction.

Player and Griffiths proposed RaPTaR [12], a method to extend existing trust models to detect and adjust to behavior changes. RaPTaR has a learning component and a predictive component. In the learning part, it uses the Kolmogorov–Smirnov (KS) test and a modification of the Adaptive Windowing (AdWin) algorithm [18] to detect changes in trustees' behavior. Specifically, the AdWin algorithm uses a variable window size. When new data arrives, the window expands. If there is a split of the window such that the KS test identifies the two subsets come from different distributions, then a change has been detected. So, data from the oldest subset is irrelevant and should be removed from the window. Also, RaPTaR learns patterns in changes of behavior by recording the time that behavior was active and a behavior succeeded it. In the predictive component, RaPTaR produces a priori trust estimates based on the patterns learned in the learning component. A priori estimates can be used when there are no, or few, experiences with a trustee.

3. **BRS with the Page-Hinkley Test**

In this section, we introduce a new method, which is the extension of BRS with the Page-Hinkley (PH) test to handle dynamic trustees. The PH test [10, 11] is a well-known sequential analysis technique to detect changes in signal processing online. Specifically, it is designed to find changes in the mean of a Gaussian signal. However, several studies show that the PH test can be used in a wider context, e.g., detecting the change between two known probability distributions [11, 19, 20]. The original PH test is a two-sided test designed to identify both increases and decreases in the mean of a data stream. Since we are especially interested in detecting harmful changes in the behavior of trustees, i.e., when the trustworthiness values of trustees rapidly decrease, we use the one-sided version of the PH test for finding decreases.

An observation indicates the outcome of an interaction between a trustee and a truster. Observations or outcomes can be represented numerically, e.g. 0 for unsuccessful observations and 1 for successful ones. Let $h^T = o_1, \ldots, o_T$ denote the observations up to time T, and let $\bar{o}_t$ denote the mean of the observations up to time t $(1 \leq t \leq T)$. Finally, let

$$d_T = \sum_{t=1}^{T} (o_t - \bar{o}_t + \delta) \tag{3.1}$$

where the *tolerance parameter* δ indicates the magnitude of changes that are allowed.

Then, the difference between the current d_T and its maximum value so far, $D_T = \max(d_t | 1 \leq t \leq T)$, is computed:

$$PH(h^T) = D_T - d_T \tag{3.2}$$

If $PH(h^T)$ is greater than a given threshold λ, a change is detected. The value of λ depends on the acceptable false alarm rate. By increasing λ, the false alarm rate decreases, but it might lead to a delay in detecting changes.

Following the notation in [21], we denote the set of trusters by $C = \{c_i \mid 1 \leq i \leq N\}$, the set of trustees by $S = \{s_j \mid 1 \leq j \leq M\}$. The outcome of an interaction between c_i and s_j at time t, denoted as $o_t(i, j)$, is either successful $(o_t(i, j) = 1)$ or unsuccessful $(o_t(i, j) = 0)$. h^0_{ij} is the interaction history containing the sequence of interaction outcomes between c_i and s_j during the preprocessing phase (see Section 4 for details). h^T_{ij} consists of h^0_{ij} and all interaction outcomes between c_i and s_j until time step T.

Algorithm 1 The BRS-PH Algorithm for Truster c_i

1: Inputs: parameters λ and δ
2: Static $= \{s_j \mid 1 \leq j \leq M\}$
3: **for** $j = 1$ **to** M **do**
4: **if** $PH(h_{ij}^0) > \lambda$ **then** Remove s_j from Static **end if**
5: **end for**
6: **for** $t = 1$ **to** T **do**
7: **if** |Static| $= 0$ **then** Static $= \{s_j \mid 1 \leq j \leq M\}$ **end if**
8: **for** $s_j \in$ Static **do**
9: c_i estimates the trustworthiness value of s_j: $\text{brs}(p_{i,j}, n_{i,j}) = \frac{p_{i,j}+1}{p_{i,j}+n_{i,j}+2}$
10: **end for**
11: c_i picks s_j with highest $\text{brs}(p_{i,j}, n_{i,j})$ denoted by s_{j^*} for interaction.
12: c_i observes outcome $o_t(i, j^*) \in \{0, 1\}$ of interaction.
13: c_i updates the interaction history $h_{ij^*}^t$.
14: **if** $PH(h_{ij^*}^t) > \lambda$ **then** Remove s_{j^*} from Static **end if**
15: **end for**

The pseudocode of BRS with the Page-Hinkley test (BRS-PH) is given in Algorithm 1. In the beginning, it assumes that all trustees are static and puts them in a set Static of trustees considered either static or not harmful (Line 2). Then, for each trustee s_j, the PH test is applied to the full interaction history h_{ij}^0 to check if there are any harmful changes in a trustee's behavior during the preprocessing phase. If such a change is detected for a trustee, the algorithm removes that trustee from Static (Line 4). After that, in each of T rounds, the following is executed. First, it checks if Static is empty.[1] If it is, Static is re-initialized to contain all the trustees (Line 7). Then, for each trustee s_j in Static, the truster c_i estimates the trustworthiness value of s_j using the BRS formula (Line 9), where $p_{i,j}$ and $n_{i,j}$ refer to the number of positive, resp. negative, interactions that c_i has had with s_j so far. A larger value of this measure suggests higher trustworthiness of s_j for c_i.

c_i chooses the trustee s_{j^*} with the highest trustworthiness value and initiates an interaction (line 11). The outcome (positive or negative interaction with s_{j^*}) is observed (line 12) and the interaction history between c_i and s_{j^*} is updated (line 13). The updated history is checked using the PH test to see whether a harmful change has occurred in its behavior. If so, s_{j^*} is removed from Static (Line 14). This algorithm consumes time and space in $O(M)$ (and can also be implemented with a runtime in $O(\log(M))$) per iteration and truster.

4. Experimental Setup

We compared BRS-PH to BRS (BRS-Basic) [1], BRS with a forgetting factor (BRS-FF) [1], RaPTaR [12], as well as an HMM-based trust model (HMM) [4]. We ran each of the methods with every setting listed in Section 4.1.

In our simulations, interaction outcomes are either positive or negative, i.e., the outcomes are binary. All interactions are independent random events.

Our empirical study uses one truster and ten trustees. Each trustee is assigned an initial trustworthiness value sampled uniformly at random from the values $\{0.1, 0.2, \ldots, 0.9\}$. Then dynamic trustees are selected. The selection is made at random for all settings except Setting 2 (see Section 4.1 for details). The percentage of trustees that are dynamic is a parameter we vary in our experiments between 20% and 80%.

[1] This never happened in our experiments.

The trustworthiness of a static trustee in our simulations is the value it was assigned initially. Each dynamic trustee is assigned two trustworthiness values, a *good* one randomly sampled from {0.7, 0.8, 0.9}, and a *bad* one randomly sampled from {0.1, 0.2, 0.3}.

Every number in Table 1 is an average of 100 runs of the same setting. Each run in our experiment consists of a preprocessing phase followed by the main phase. BRS-Basic, BRS-FF, BRS-PH, and RaPTaR use the preprocessing to get initial estimates of the trustees' trustworthiness; HMM uses it for training. Concerning the choice of evaluation measures, most of the literature focuses on the accuracy with which a system estimates the trustees' trustworthiness, measured in terms of mean absolute error (MAE) [2, 3, 22]. This kind of evaluation ignores utility aspects such as cost. If a negative interaction is more costly than a positive interaction, MAE alone is not sufficient for assessing a system. For example, suppose there are nine trustees with a trustworthiness of 0.2 and one with a trustworthiness of 0.6. Suppose further, System A's trustworthiness estimates are 0 for the good trustee, and are perfect for the nine bad trustees, while System B's estimates are 0.4 for the good trustee and 0 for all bad ones. Then System A (MAE=0.06) would have more accurate estimates than System B (MAE=0.2), yet System B would have many more positive interactions than System A. Therefore, in addition to measuring MAE, we also apply a utility-based measure, specifically the *relative frequency of unsuccessful interactions* (RFU) [21].

In each iteration during the preprocessing phase, a trustee s_j is sampled uniformly at random. T_{pre} denotes the total number of iterations in the preprocessing phase, which is set to 100 in our simulations. Each time a trustee s_j is sampled, an interaction between the truster and s_j is simulated, using the trustworthiness value of s_j at the interaction time. During the preprocessing phase, the dynamic trustees behave exactly as they do in the main phase (as defined in Section 4.1).

After preprocessing, a run proceeds in rounds until a target number of positive interactions, denoted by T_{target} (here $T_{\text{target}} = 500$), have occurred in the main phase. Using method M to assess the trustworthiness of trustees, each round consists of four steps:

(1) The truster obtains trustworthiness estimates for all trustees from M.
(2) The truster interacts with the trustee whose trustworthiness estimate is maximal (breaking ties randomly).
(3) Based on the outcome, M updates its trustworthiness estimate of the chosen trustee.
(4) If the target number of positive interactions is not yet achieved, then the next round is initiated.

The decay factor in BRS-FF is set to 0.8. The Baum-Welch algorithm [16] is used after preprocessing and once after each subsequent interaction to learn the parameters of the HMM. The RaPTaR method has two parameters. One parameter in the KS test, $\alpha \in [0, 1]$, and the stable learning size s. We try RaPTaR with different α values and set s to 3. The PH test in BRS-PH requires setting the tolerance parameter δ and the threshold λ. We set δ to 0.05, and used a single dynamic behavior setting (Section 4.1, Setting 1: camouflage) to tune λ and then set it to 12 for *all* settings without further tuning.

4.1. **Types of Dynamic Behavior**

Our evaluation considers the following five types of dynamic behavior. In each setting except Setting 2, the dynamic trustees are initially chosen at random.

Setting 1: Camouflage. Dynamic trustees use their good trustworthiness value during the first τ rounds of a run and their bad value thereafter [22]. τ is chosen uniformly at random from $[1, T_{\text{target}})$.

Setting 2: Best-Turn-Bad. Identical to Setting 1 except that the trustees with the highest initial trustworthiness are chosen to be dynamic.

Setting 3: Good-Bad-Good. Dynamic trustees use their good trustworthiness value during the first τ_1 rounds of a run, their bad value in the next $\tau_2 - \tau_1$ rounds, and their good value thereafter. τ_1 and τ_2 are chosen uniformly at random from $[1, T_{\text{target}})$ such that $\tau_1 < \tau_2$.
Setting 4: Periodic Behavior. Dynamic trustees use their good value during the first τ_{periodic} rounds of a run, their bad value in the next τ_{periodic} rounds, their good value in the next τ_{periodic} rounds, and so on, alternating periodically [6]. So as to have at least four periods, τ_{periodic} is chosen uniformly at random from $[1, T_{\text{target}}/4]$.
Setting 5: Random Behavior. In each interaction, every dynamic trustee chooses between its good and its bad trustworthiness value randomly with equal probability [6].

During the preprocessing phase, the times at which the dynamic trustees change between good and bad values are defined in the same way except that T_{pre} is used instead of T_{target}.

5. Results and Discussion

For each setting, we report simulations with four different percentages (20%, 40%, 60%, 80%) of dynamic trustees. We ran Wilcoxon signed-rank tests including Holm-Bonferroni correction (at the 95% confidence level) for BRS-PH in comparison to each of the other methods. An entry for BRS-Basic, BRS-FF, HMM, or RaPTaR is in bold if it differs from BRS-PH's value significantly. Among these entries, an asterisk marks cases in which BRS-PH is significantly worse.

5.1. RFU

The relative frequency of unsuccessful interactions (RFU) after t interactions is the number of negative interactions among the first t interactions, divided by t. Table 1 reports RFU after 500 positive interactions have been made. In all but one case, BRS-PH either outperforms or is not significantly different from all tested methods. In Setting 3 with 80% dynamic trustees, BRS-FF outperforms BRS-PH.

BRS-PH is designed to flag a trustee as soon as its trustworthiness value decreases substantially. This explains the superiority of BRS-PH over BRS-Basic, HMM, and RaPTaR in Settings 1 and 2, where dynamic trustees start with good trustworthiness values, and later change to bad values. BRS-PH beats BRS-FF in Setting 1 (except for a tie for 40% and 80% dynamic trustees). Also, it beats BRS-FF in Setting 2 (except for a tie for 40% dynamic trustees). Intuitively, though the originally best trustees will suddenly become poor in Setting 2, BRS-FF will stick with them for a while, since their relatively recent history still makes them look better than the static trustees. BRS-PH though discards trustees that worsen substantially, which gives it an advantage in this setting.

Intuitively, BRS-PH should lose this advantage in cases when trustees that turned from good to bad later become good again. Once removed from the set R, BRS-PH will not consider these trustees again, while the competing methods still do. Setting 3 targets exactly this behavior, so we expected it to be the least likely setting for BRS-PH to beat its competitors. Two aspects should be taken into account though. First, as long as a static trustee with a high trustworthiness value exists, BRS-PH will tend to interact with that trustee and therefore not be worse in terms of RFU than the other methods. Second, the length of the bad and good intervals plays an important role. If a bad interval is long and the subsequent good interval is short, BRS-PH will not suffer much compared to the other methods. It may suffer however when a bad interval is long enough to remove a trustee s_j, but subsequently s_j is the most trustworthy trustee for a large number of iterations. In Table 1 we see that BRS-PH is never worse than BRS-Basic and HMM and is only beaten by BRS-FF for 80% dynamic trustees. Also, it is on a par with or beats RaPTaR.

Setting 4 is similar in design to Setting 3. Here, BRS-PH beats BRS-FF (except for a tie for 80% dynamic trustees). The latter aggressively follows dynamic trustees and forgets

		20%	40%	60%	80%
S1	BRS-Basic	**0.178**	**0.209**	**0.271**	**0.383**
	BRS-FF	**0.171**	0.192	**0.239**	0.325
	HMM	**0.181**	0.197	**0.254**	**0.358**
	RaPTaR($\alpha = 0.01$)	**0.188**	**0.236**	**0.288**	**0.390**
	RaPTaR($\alpha = 0.5$)	**0.196**	**0.223**	**0.274**	**0.360**
	RaPTaR($\alpha = 0.8$)	**0.192**	**0.213**	**0.259**	**0.351**
	BRS-PH	0.165	0.188	0.234	0.321
S2	BRS-Basic	**0.315**	**0.490**	**0.587**	**0.615**
	BRS-FF	**0.300**	0.425	**0.541**	**0.600**
	HMM	**0.310**	**0.466**	**0.567**	0.603
	RaPTaR($\alpha = 0.01$)	**0.321**	**0.459**	**0.549**	**0.604**
	RaPTaR($\alpha = 0.5$)	**0.336**	**0.469**	**0.577**	**0.624**
	RaPTaR($\alpha = 0.8$)	**0.322**	**0.450**	**0.568**	**0.621**
	BRS-PH	0.286	0.430	0.527	0.588
S3	BRS-Basic	0.159	0.173	0.197	0.232
	BRS-FF	0.158	0.170	0.189	**0.215***
	HMM	0.168	0.186	0.219	0.245
	RaPTaR($\alpha = 0.01$)	**0.180**	**0.192**	0.207	0.247
	RaPTaR($\alpha = 0.5$)	**0.178**	**0.186**	0.194	0.225
	RaPTaR($\alpha = 0.8$)	**0.179**	**0.188**	0.203	0.232
	BRS-PH	0.158	0.174	0.202	0.237
S4	BRS-Basic	0.167	**0.209**	0.245	0.318
	BRS-FF	**0.182**	**0.209**	**0.254**	0.322
	HMM	**0.183**	0.216	**0.256**	0.332
	RaPTaR($\alpha = 0.01$)	**0.198**	**0.224**	**0.286**	**0.363**
	RaPTaR($\alpha = 0.5$)	**0.207**	**0.230**	**0.280**	**0.344**
	RaPTaR($\alpha = 0.8$)	**0.205**	**0.230**	**0.275**	**0.338**
	BRS-PH	0.169	0.205	0.241	0.319
S5	BRS-Basic	0.167	0.186	0.239	0.306
	BRS-FF	**0.184**	**0.212**	**0.267**	**0.327**
	HMM	0.179	0.197	**0.260**	**0.318**
	RaPTaR($\alpha = 0.01$)	**0.197**	**0.241**	**0.286**	**0.350**
	RaPTaR($\alpha = 0.5$)	**0.219**	**0.238**	**0.295**	**0.353**
	RaPTaR($\alpha = 0.8$)	**0.207**	**0.236**	**0.290**	**0.347**
	BRS-PH	0.167	0.186	0.239	0.306

Table 1. Average RFU over 100 runs.

older transactions, which hurts in terms of utility when the trustees are in the bad phase. In Setting 4, this happens more often than in Setting 3; therefore, BRS-FF loses the advantage it had over BRS-PH in Setting 3 for higher percentages of dynamic trustees.

In Setting 5, a dynamic trustee with good value g and bad value b will on average behave like a static one whose trustworthiness is the mean of g and b. BRS-FF is sensitive to the frequent changes in these trustees, but their effects tend not to persist long enough for the PH test to flag them. So, BRS-PH behaves almost identically to BRS-Basic and outperforms BRS-FF and RaPTaR in Setting 5. As in all other settings, HMM cannot beat BRS-PH.

Table 2 summarizes the RFU results for the 20 tested cases (5 settings, 4 percentages). BRS-PH is declared to win (lose, resp.) if its RFU is significantly better (worse, resp.) than that of the method named in the column heading. Statistically insignificant results are considered ties. Based on this summary, BRS-PH stands out as the clear winner.

We also evaluated RFU for 30 trustees. The trends observed were the same as for 10 trustees, yet slightly more in favour of BRS-PH, and with sometimes noticeably poorer performance of HMM. It seems that HMM needs much more training data than the BRS-type methods when the number of trustees increases.

	BRS-Basic	BRS-FF	HMM	RaPTaR
Wins for BRS-PH	9	13	10	18
Ties	11	6	10	2
Losses by BRS-PH	0	1	0	0

Table 2. Summary of the RFU results.

5.2. **RFU Comparison to the Best Static Trustee**

In this section we compare the RFU for BRS-PH reported in the previous section to the average RFU for the best static trustee, which we denote RFU_{bst}. On each run, once the dynamic trustees have been chosen, the maximum trustworthiness value, v_{max}, of the remaining, static trustees is determined and $1 - v_{max}$ is used as the RFU for that run. RFU_{bst} is the average of those RFU's over the 100 runs. RFU_{bst} is a strictly theoretical value since a system cannot determine, based on a relatively small number of interactions, which of the trustees is the best static trustee.

Table 3 shows RFU_{bst} for each of our 20 scenarios (5 settings, with 4 percentages of dynamic trustees in each). The values shown for BRS-PH are copied from Table 1 for ease of reference. The row labelled "ratio" is the ratio of BRS-PH's RFU to RFU_{bst}. Ratios larger than 1.0, shown in bold, mean that BRS-PH's RFU is larger than RFU_{bst}.

It is not surprising that BRS-PH's RFU is usually larger than RFU_{bst} when there are 20 or 40% dynamic trustees. With so many static trustees, the best of them will very likely have a very high trustworthiness. Likewise, it is not surprising that BRS-PH's RFU is usually smaller than RFU_{bst} when there are 80% dynamic trustees. With so few static trustees, the best of them will very likely have a mediocre trustworthiness.

Interesting about these results is that BRS-PH's RFU is never much higher than RFU_{bst} (Setting 4-40%, is the worst), but it can be very much lower (Settings 2-80% and 3-80%).

		20%	40%	60%	80%
S1	RFU_{bst}	0.156	0.181	0.244	0.357
	BRS-PH	0.165	0.188	0.234	0.321
	ratio	**1.057**	**1.038**	0.959	0.899
S2	RFU_{bst}	0.292	0.452	0.623	0.783
	BRS-PH	0.286	0.430	0.527	0.588
	ratio	0.979	0.951	0.845	0.751
S3	RFU_{bst}	0.152	0.174	0.218	0.319
	BRS-PH	0.158	0.174	0.202	0.237
	ratio	**1.039**	1.000	0.926	0.742
S4	RFU_{bst}	0.156	0.186	0.238	0.354
	BRS-PH	0.169	0.205	0.241	0.319
	ratio	**1.083**	**1.102**	**1.012**	0.901
S5	RFU_{bst}	0.157	0.178	0.237	0.334
	BRS-PH	0.167	0.186	0.239	0.306
	ratio	**1.063**	**1.044**	**1.008**	0.916

Table 3. Average RFU for Best Static Trustee.

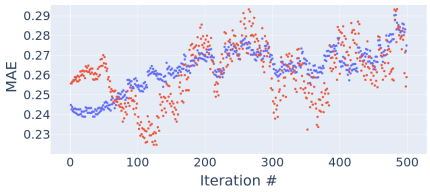
5.3. **MAE**

From an application point of view, one might prefer a utility-based evaluation measure like RFU over a measure that assesses how well an agent estimates trustworthiness values. BRS-PH is specifically designed with utility in mind; by removing a trustee s_j from its set R, BRS-PH does not even attempt any longer to estimate s_j's trustworthiness. Instead,

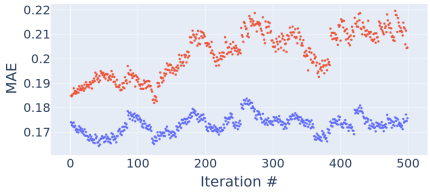
BRS-PH tries to maximize utility by avoiding the risk of initiating a negative interaction with s_j. However, since the mean absolute error (MAE) of an agent's estimate of a trustee's trustworthiness is a popular measure in the literature, we discuss it here. MAE is the mean absolute difference between the actual and the estimated trustworthiness values, where the mean is taken over all pairs of truster and trustee.

Due to BRS-PH's design, one would expect it not to do well in terms of MAE, but we will now see that it tends to outperform BRS-FF, HMM, and RaPTaR in terms of MAE, while there is no clear winner between BRS-PH and BRS-Basic.

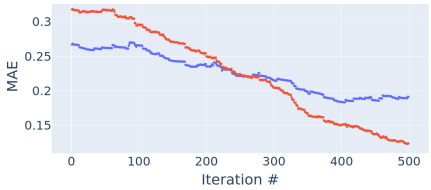
A dynamic trustee has no single *true* trustworthiness value – its trustworthiness changes over time. Hence, measuring MAE at a specific point in time (e.g., after 500 iterations) does not necessarily indicate how well a trust system estimates trustworthiness over time. For example, Figure 1a illustrates MAE at each iteration, averaged over 100 simulations of Setting 4 (80% dynamic trustees), with the iteration number on the x-axis and MAE at iteration x on the y-axis. The red curve is for BRS-FF, and the blue one for BRS-PH. Clearly, which method beats the other changes frequently across these 500 iterations.



(a) Setting 4, 80%.



(b) Setting 4, 40%.



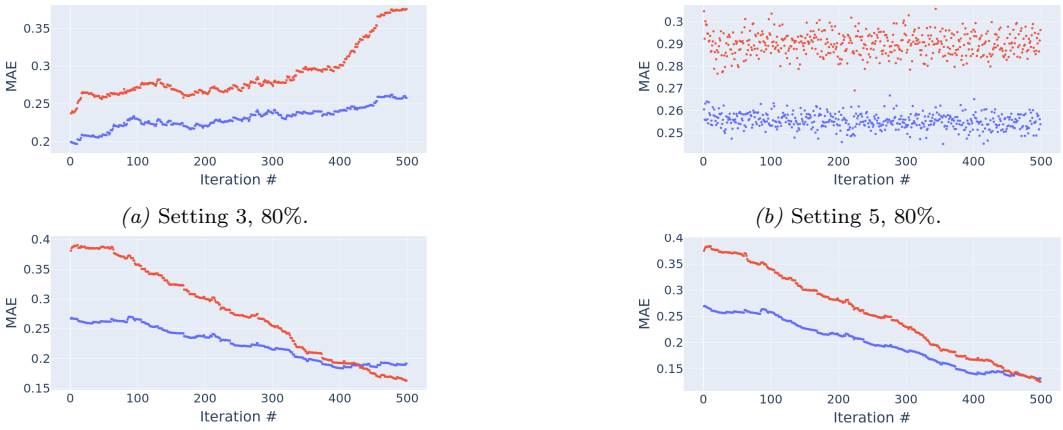
(c) Setting 1, 80%.

Figure 1. BRS-PH (blue) vs. BRS-FF (red).

Not in all cases do the MAE plots for BRS-PH and BRS-FF cross each other so often. For instance, in the 40% version of Setting 4 (Figure 1b), BRS-PH consistently yields more accurate estimates than BRS-FF. In other cases, like Setting 1-80% (Figure 1c), BRS-FF is inferior in the first 250 iterations, but consistently superior later on.

BRS-PH usually beats BRS-FF in early iterations because BRS-FF's performance heavily depends on the order in which outcomes with static trustees are observed. For example, assume in the preprocessing phase the truster has ten interactions with a static trustee whose trustworthiness value is 0.8. On average, this will result in two negative and eight positive outcomes. BRS-PH, which is identical to BRS-Basic in preprocessing as the PH test is only applied at the end of that phase, would then produce 0.750 as its estimate. However, using BRS-FF with 0.8 as the forgetting factor, the estimate would range from 0.567 (if the last two interactions are negative) to 0.799 (if the first two interactions are negative).

Since BRS-Basic and BRS-PH behave identically in preprocessing and use the same formula for selecting a trustee for interaction, their initial MAE values in the main phase are identical. They start to differ when BRS-PH flags a trustee and stops interacting with it. Removing a trustee means that BRS-PH will never update its trustworthiness estimate of that trustee again (unless all trustees are removed), which may, depending on the future behavior of that trustee, lead to better or worse estimates than those obtained by the other



(a) Setting 3, 80%.

(b) Setting 5, 80%.

(c) Setting 1, 80%.

(d) Setting 2, 80%.

Figure 2. BRS-PH (blue) vs. HMM (red).

methods. In Settings 1 and 2, the MAE curves for BRS-PH and BRS-Basic are identical for around 100 iterations; then they separate and never cross each other again. In Setting 3, the same occurs, but they separate after 350 iterations. This is not the case for Settings 4 and 5, where the curves cross each other often. For Settings 1 and 2, BRS-PH always achieved a lower MAE than BRS-Basic, but in some cases for Settings 3 and 4, BRS-Basic won.

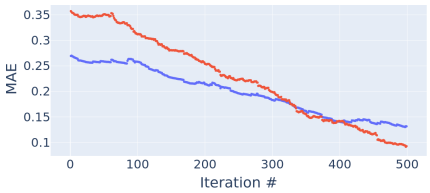
Our results show BRS-PH to be superior to HMM in terms of MAE in most cases, especially in Settings 3 and 5. Examples are given in Figures 2a and 2b, which plot the average MAE over 100 runs, with a blue curve for BRS-PH, and a red curve for HMM. In some cases for Settings 1 and 2, HMM's MAE value eventually becomes better than BRS-PH's (Figures 2c and 2d), but the curves cross relatively late.

BRS-PH also outperforms RaPTaR in terms of MAE, especially for higher values of RaPTar's α parameter and for Settings 3, 4, and 5. Note that choosing a very small α value requires a much higher confidence to flag a change. When requiring too much confidence, RaPTaR might not detect any change. Therefore, for very small values of α, the results from RaPTaR may become similar to those from BRS-Basic. In some cases for Settings 1 and 2, RaPTaR's MAE values eventually become better than BRS-PH's (Figure 3a). Also, in a few cases for Settings 3 and 4, BRS-PH's MAE values are worse than RaPTaR's in the early iterations, but become better than RaPTaR's at some point (Figure 3b). In most cases, RaPTaR's MAE values are worse than BRS-PH's from the very first iterations onwards, or after a relatively small number of iterations; an example is given in Figure 3c.

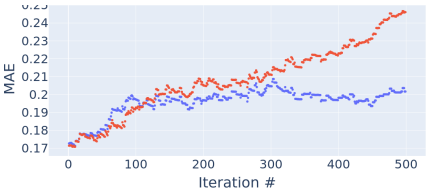
	BRS-Basic	BRS-FF	HMM	RaPTaR		
				$\alpha = 0.01$	$\alpha = 0.5$	$\alpha = 0.8$
BRS-PH Dominates	8	13	13	12	15	15
Ties	5	1	1	1	0	0
Crossing Curves (W)	0	6	6	5	4	4
Crossing Curves (B)	0	0	0	2	1	1
BRS-PH is Dominated	7	0	0	0	0	0

Table 4. Summary of the MAE curves comparing BRS-PH to BRS-Basic, BRS-FF, HMM, and RaPTaR with different α values.

Due to page limits, we cannot display MAE plots for all tested settings; Table 4 gives a summary of how BRS-PH's MAE curves compare to those of the other systems. Each row in this table refers to one of four relations between two curves. "BRS-PH Dominates" means



(a) $\alpha = 0.50$, Setting 2, 80%.



(b) $\alpha = 0.01$, Setting 1, 60%.

(c) $\alpha = 0.01$, Setting 4, 20%.

Figure 3. BRS-PH (blue) vs. RaPTaR (red).

that BRS-PH's curve is never above the curve of the other system, as, e.g., in Figure 2a. "BRS-PH is Dominated" means that BRS-PH's curve is never below that of the other system. "Crossing curves (W)" means that BRS-PH's curve is below that of the other system in the early iterations, but at some point the curves cross and BRS-PH has a worse MAE thereafter as, e.g., in Figure 1c. "Crossing curves (B)" means that BRS-PH's curve is above the other curve in the early iterations, but at some point the curves cross and BRS-PH has a better MAE thereafter as, e.g., in Figure 3b. "Ties" are when the curve for BRS-PH and the other system are entangled across all iterations as in Figure 1a. From this table, it is clear that, in our test settings, BRS-PH tends to perform better in terms of MAE than BRS-FF, HMM, and RaPTaR while there is no clear winner between BRS-PH and BRS-Basic.

5.4. **A Note on the Performance of HMM**

Previous literature claimed that HMM-based methods outperform BRS-Basic and BRS-FF. Our results do not support this claim. In fact, the HMM method from [4] tended to perform worse than BRS-FF in terms of RFU (except for Setting 5) and did not outperform it in terms of MAE either. Previous claims are in part based on an evaluation with only a single trustee, whose trustworthiness was to be estimated [6]. In such settings, the truster has no choice with whom to interact. In particular, it will not miss out on information about how another trustee would have behaved if one had chosen to interact with it. Settings with multiple trustees are evidently much harder for HMMs.

6. **Conclusion**

We proposed a new simple method for modeling dynamic trust. Specifically, we extended the Beta reputation system with the Page-Hinkley test. The results of our empirical RFU analysis show that our new method outperforms all other tested methods for handling dynamic trustees, with very few exceptions. In terms of MAE, our method tends to beat two of the three tested competitors and beats the third one as often as it loses to it.

Note that our method would not work well in a situation where one or more dynamic trustees turn bad, and even after turning bad are still the trustees with the highest trustworthiness by some noticeable margin. BRS-PH would consider these dynamic trustees as harmful and remove them, thus being left with only inferior trustees, which will cause an unnecessarily high number of negative interactions (i.e., an unnecessarily high RFU value).

This can likely be addressed by a simple modification to our algorithm, e.g., by not removing a harmful trustee s_j unless a trustee with an estimated trustworthiness not too far below that of s_j remains in R. We have not yet tested this modification.

References

[1] A. Jøsang and R. Ismail. "The Beta reputation system". In: *Proc. 15th Bled Electronic Commerce Conf.* 2002, pp. 2502–2511.

[2] W. T. L. Teacy, J. Patel, N. R. Jennings, and M. Luck. "Travos: Trust and reputation in the context of inaccurate information sources". In: *Autonomous Agents and Multi-Agent Systems* 12.2 (2006), pp. 183–198.

[3] K. Regan, P. Poupart, and R. Cohen. "Bayesian reputation modeling in e-marketplaces sensitive to subjectivity, deception and change". In: *Proc. 21th AAAI Conf. on Artificial Intelligence*. 2006, pp. 1206–1212.

[4] E. ElSalamouny, V. Sassone, and M. Nielsen. "HMM-based trust model". In: *Intl. Workshop on Formal Aspects in Security and Trust*. 2009, pp. 21–35.

[5] E. ElSalamouny, K. T. Krukow, and V. Sassone. "An analysis of the exponential decay principle in probabilistic trust models". In: *Theoret. Comput. Sci.* 41 (2009), pp. 4067–4084.

[6] M. E. G. Moe, B. E. Helvik, and S. J. Knapskog. "Comparison of the Beta and the Hidden Markov Models of trust in dynamic environments". In: *IFIP Intl. Conf. on Trust Management*. 2009, pp. 283–297.

[7] M. E. G. Moe, M. Tavakolifard, and S. J. Knapskog. "Learning trust in dynamic multiagent environments using HMMs". In: *Proc. 13th Nordic Workshop on Secure IT Systems*. 2008.

[8] G. Vogiatzis, I. MacGillivray, and M. Chli. "A probabilistic model for trust and reputation". In: *Proc. 9th Intl. Conf. on Autonomous Agents and Multiagent Systems*. 2010, pp. 225–232.

[9] X. Liu and A. Datta. "Modeling context aware dynamic trust using Hidden Markov Model". In: *Proc. 26th AAAI Conf. on Artificial Intelligence*. 2012.

[10] E. S. Page. "Continuous inspection schemes". In: *Biometrika* 41.1/2 (1954), pp. 100–115.

[11] D. V. Hinkley. "Inference about the change-point from cumulative sum tests". In: *Biometrika* 58.3 (1971), pp. 509–523.

[12] C. Player and N. Griffiths. "Improving trust and reputation assessment with dynamic behaviour". In: *The Knowledge Engineering Review* 35 (2020).

[13] J. Zhang and R. Cohen. "Evaluating the trustworthiness of advice about seller agents in e-marketplaces: A personalized approach". In: *Electronic Commerce Research and Applications* 7.3 (2008), pp. 330–340.

[14] X. Liu and A. Datta. "A trust prediction approach capturing agents' dynamic behavior". In: *Proc. 22th Intl. Joint Conf. on Artificial Intelligence*. 2011.

[15] X. Zheng, Y. Wang, and M. A. Orgun. "Modeling the dynamic trust of online service providers using HMM". In: *Proc. 20th IEEE Intl. Conf. on Web Services*. 2013, pp. 459–466.

[16] L. E. Baum, T. Petrie, G. Soules, and N. Weiss. "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains". In: *The Annals of Mathematical Statistics* 41.1 (1970), pp. 164–171.

[17] L. R. Rabiner. "A tutorial on Hidden Markov Models and selected applications in speech recognition". In: *Proc. IEEE* 77.2 (1989), pp. 257–286.

[18] A. Bifet and R. Gavalda. "Learning from time-changing data with adaptive windowing". In: *Proc. Intl. Conf. on Data Mining*. SIAM. 2007, pp. 443–448.

[19] G. Lorden. "Procedures for reacting to a change in distribution". In: *The Annals of Mathematical Statistics* 42.6 (1971), pp. 1897–1908.

[20] M. Basseville. "Detecting changes in signals and systems—a survey". In: *Automatica* 24.3 (1988), pp. 309–326.

[21] E. Parhizkar, M. H. Nikravan, and S. Zilles. "Indirect Trust is Simple to Establish." In: *Proc. 28th Intl. Joint Conf. on Artificial Intelligence*. 2019, pp. 3216–3222.

[22] S. Jiang, J. Zhang, and Y.-S. Ong. "An evolutionary model for constructing robust trust networks." In: *Proc. 12th Intl. Conf. on Autonomous Agents and Multiagent Systems*. 2013, pp. 813–820.