A Cluster-based Integrated Trust Establishment Model for Intelligent Agents

Julian Templeton
Electrical Engineering and Computer Science
University of Ottawa, Canada
jtemp005@uottawa.ca
Thomas Tran
Electrical Engineering and Computer Science
University of Ottawa, Canada
ttran@uottawa.ca

Abstract

This paper presents a cluster-based version of the Integrated Trust Establishment (ITE) model. Despite ITE's strong performance in simulated environments, the trust establishment model's robust method for dynamically updating its improvement and disimprovement rate hyperparameters can be improved to better handle trustors of varying behaviours. By modifying ITE's dynamic hyperparameter update process with a cluster-based approach, the model sees improved performance by better meeting the needs of varied trustors in an environment. This improvement is exhibited by comparing ITE with the newly proposed Cluster-Based ITE (CBITE) in simulated tests. The effects of this cluster-based approach of performing updates based on groupings of trustors is seen to be more effective than performing the updates independently. Trustees using CBITE more accurately meet the desires of varied sets of trustors than ITE.

1 Introduction

Intelligent agents are becoming increasingly robust and can be equipped with a variety of tools to help communicate with other agents and to determine the trustworthiness of these agents. Since the agents which populate a Multi-Agent System (MAS) collaborate with other agents to accomplish tasks, the ability to gauge the trustworthiness of other agents is imperative. This active research topic has driven the discovery of many different trust models, some of which are presented within surveys such as [YSL$^+$13]. The agents which reside within MASs can be referred to as a trustor or as a trustee. A trustor consumes a service from trustees whereas a trustee provides services to trustors. Agents can be both a trustor or trustee and utilize their calculated trust values of other agents to understand which trustees can be trusted when selecting interaction partners.

In: R. Falcone, J. Zhang, and D. Wang (eds.): Proceedings of the 22nd International Workshop on Trust in Agent Societies, London, UK on May 3-7, 2021, published at http://ceur-ws.org

Despite there being significant trust evaluation research, the concept of trust establishment that is presented in [Sen13] has been less explored. Within [Sen13], Sen proposes the use of trust establishment alongside trust evaluation within the trust management module which agents contain. The goal of trust establishment is to allow a trustee to be able to learn how to improve and maintain their trust with trustors in an environment. This will help trustees become viable options within a MAS and will improve the quality of agent interactions within the MAS. Although relatively unexplored, there are a number of trust establishment models that have been proposed to help trustees become more trustworthy in an environment. A recent, state-of-the-art trust establishment model is the Integrated Trust Establishment (ITE) model [AT20]. This model attempts to actively balance the amount of resources which a trustee spends during interactions with specific trustors, referred to as the Utility Gain (UG), with the trustee's trust in an environment. This is done by using robust equations with many hyperparameters and by using concepts such as the engagement of a trustor. This approach differs to an earlier model which has trustees classify the type of a trustor to determine how to calculate the UG that will be provided the trustor in an e-commerce environment [TCLK14].

ITE uses two dynamic hyperparameters, denoted by α and β, to represent the rates at which a trustee should increase or decrease the UG that is provided to a trustor. These values are dynamically updated by using the Rate of Change (RoC) of the direct feedback that is obtained from trustors for all transactions performed between the trustors and the trustee at two different time steps. The RoC values are calculated at specific time steps and provide insight into the general satisfaction of the trustors within an environment. Although this approach has provided strong simulation results, using singular RoC values to perform the updates for all trustors will inaccurately update a trustee's behaviours towards specific trustors when there are varied trustor behaviours in the environment. This occurs because the varied behaviours can disrupt the RoC that is calculated at any time step. Even if the RoC is computed for individual trustors, rather than for all trustors, these trustors may be temporarily shifting their behaviours or may be acting maliciously. By grouping trustors that display similar behaviours at a given time step and updating the α and β values for each trustor in a group based only on the behaviour changes observed within their group, the α and β variables will be more accurately updated. By replacing ITE's dynamic variable update process with this cluster-based approach, ITE's performance will be improved in varied environments. This cluster-based approach is applied to the Cluster-Based ITE (CBITE) trust establishment model that is presented in this paper. A simulation environment will be defined and used to compare ITE to CBITE in several simulated tests which will be presented and discussed. This will result in a modified ITE that helps improve performance in varied environments.

2 Background Information

Prior to the presentation of CBITE, we will outline some background information to better understand ITE's design and the expectations for the environments and agents which use ITE. ITE uses a combination of ideas from previous trust establishment models and expands upon them to improve its capabilities. Two of the core trust establishment models which ITE expands upon are the Reinforcement Learning based Trust Establishment (RLTE) model [AT15] and the Acting as a Trustee Using Implicit Feedback (ATeIF) model [AT17]. These models showcase how a trustee's behaviour can be more accurately updated for individual trustors by using a combination of explicit and implicit trustor feedback, by calculating the retention of trustors, and by predicting criteria weights for the multiple criteria of a task. Each of these concepts are used by ITE to improve its robustness of helping with trust establishment. To help exhibit ITE's dynamic variable updating mechanism at a high-level and to display how it will compare to CBITE's approach, Figure 1 presents a visual overview of how the approaches differ. As previously mentioned, ITE uses dynamically updated α and β variables to help determine the rates at which updates should be performed. Figure 1 showcases that ITE focuses on using a single RoC value to update its α and β values while the newly proposed CBITE focuses on updating the α and β values for similar trustors which have been clustered together. Performing the updates with a single RoC can cause issues since that value may inaccurately adjust the trustee's behaviour towards specific trustors.

Each of the models described in this paper are decentralized. This means that they are contained within the trustee itself, rather than central entities in the environment(s). Each model is also capable of handling tasks with multiple criteria. A task in the environment, denoted by s, consists of one or more criterion values c_i. For a trust establishment model to help a trustee update their behaviour towards trustors for tasks with multiple criteria, the model must be capable of understanding how to balance the importance of each criterion. This is more complicated than understanding single criteria tasks but is important to help the model be more applicable in domains such as e-commerce in which criteria such as the time of delivery and the quality of the item are

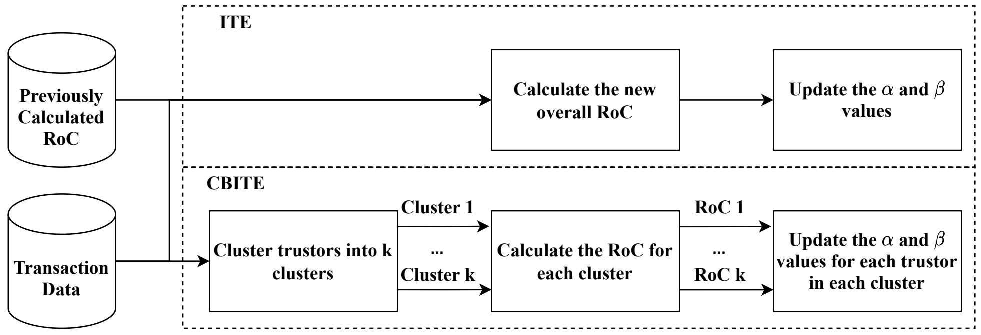

ITE
Previously Calculated RoC
Calculate the new overall RoC
Update the α and β values
CBITE
Transaction Data
Cluster trustors into k clusters
Cluster 1
...
Cluster k
Calculate the RoC for each cluster
RoC 1
...
RoC k
Update the α and β values for each trustor in each cluster

Figure 1: A high-level overview of ITE and CBITE's α and β variable updating mechanisms

represented by unique UG values.

Both ITE and our cluster-based implementation CBITE are designed to be run in singular or distributed MASs that are open and dynamic. This means that the MAS(s) allow agents to be self-interested, diverse, and deceptive while also allowing agents to freely leave and join the environment [BNS11]. Thus, it is important to test the effectiveness of these models with trustors which exhibit more varied behaviours. Although trustor attacks do not occur in the simulations performed within this paper or ITE's paper, it is also important to note that a model that is designed for open MASs may also need to help agents handle trustor attacks [SS05]. The environment used throughout this paper is also decentralized. This means that trustor communication, trust evaluation, and trust establishment will be handled by the agents themselves, rather than a centralized entity.

In the environment, intelligent agents will communicate with one another to collaborate for specific tasks. A trustor will evaluate the trustworthiness of trustees with a trust evaluation model and select zero or more trustees to interact with at each time step. After a trustee receives an interaction request for a task, the trustee proposes the UG that will be provided to the trustor for that task. This amount is computed by the trust establishment model to assist the trustee in providing accurate amounts to specific trustors. Although trustees may have limits on the amount of UG that can be provided at each time step or on the total number of trustors that can be interacted with at once, this paper assumes that trustees can serve any number of trustors at a given time step and can always provide the necessary UG. Trustors receive the proposed UG and decide whether to accept the proposed amount and perform the transaction with the trustee or to reject the UG that is provided. Accepted interactions are then performed by trustees and the trustors receive the UG for each criterion in the task. Finally, the trustor's satisfaction with the transaction will either be provided to the trustee or will be predicted by the trustee. The complete transactional data is then used by the trust establishment model to adjust how the trustee behaves towards the specific trustor.

The agent itself is expected to be capable enough of running the selected trust establishment model. One requirement from ITE is that the agent can store the transactional data from transactions that have occurred within the past H active time steps. The hyperparameters from ITE and CBITE allow the models to be tuned for specific agent capabilities, but still require some memory and computational resources. Furthermore, CBITE uses a cluster-based approach to updating ITE's dynamic improvement and disimprovement rate variables and hence requires the ability to utilize some Unsupervised Machine Learning algorithm. Any clustering algorithm can be used; thus, it is important that the agent can run a clustering algorithm that will perform well when grouping trustors based on a set of input values.

3 Cluster-Based Trust Establishment

With an understanding of the environment that is being used and the agents which reside within them, we will now present the cluster-based approach to updating ITE's dynamic variables for improved performance. Before presenting this cluster-based process, ITE's current architecture will be described. This will provide context to the effects that the cluster-based approach has on the rest of the system and flesh out CBITE's complete design. Following the description of ITE's architecture, we will present the cluster-based process and describe the issues from ITE that it aims to correct.

3.1 ITE's Architecture

Since CBITE's general design is the same as ITE's, we will first describe and present the key components from ITE. Since this paper proposes a new approach to updating the dynamic improvement and disimprovement rate variables used by ITE, rather than proposing a completely unique model, this section will provide written descriptions for certain ITE components rather than fully presenting each component of the architecture. Each formula used within this subsection and the full details of each of ITE's components can be found in Aref's paper on ITE [AT20].

As mentioned in section 2 of this paper, ITE uses the concept of trustor retention to help more accurately perform updates to trustee y's behaviour towards trustor x. By computing the retention values of individual trustors, a trustee can compute a value that is used to determine the likelihood of the trustor interacting with the trustee for a specified task s at time step t. From this, a trustee can classify whether a trustor x is engaged with the trustee, unengaged with the trustee, or neither engaged nor unengaged with the trustee. This is later used to help determine the optimal methods of updating the trustee's behaviour to improve or maintain trust with that trustor.

ITE and several previous trust establishment models, such as RLTE and ATeIF, predict how much a trustor weighs each criterion of a task through predicted relative weight (rw) values. Each trustor represents their overall expectations for a criterion of a specific task by assigning a weight to them. This weight is called a rw value, which represents the trustor's weight of the criterion divided by the trustor's demand for the criterion. Since these values represent a trustor's expectations, ITE attempts to predict these rw values continuously such that they can be used to provide the appropriate UG for each criterion. The function that is used for updating this predicted rw value stored by trustee y for trustor x for a criterion of a task s_{c_i} is displayed below.

$$rw_x^y(s_{c_i}, req) = \begin{cases} (1+\alpha) * rw_x^y(s_{c_i}, tra') & SAT_x^y(s, tra') < \Omega \ and \ rw_x^y(s_{c_i}, tra') \geq \Phi \\ (1+\beta) * rw_x^y(s_{c_i}, tra') & SAT_x^y(s, tra') \geq \theta \ and \ rw_x^y(s_{c_i}, tra') < \Phi \\ (1+\gamma * \alpha) * rw_x^y(s_{c_i}, tra') & SAT_x^y(s, tra') < \Omega \ and \ rw_x^y(s_{c_i}, tra') < \Phi \\ (1+\zeta * \beta) * rw_x^y(s_{c_i}, tra') & SAT_x^y(s, tra') \geq \theta \ and \ rw_x^y(s_{c_i}, tra') \geq \Phi \\ (1+\lambda * \alpha) * rw_x^y(s_{c_i}, tra') & else \end{cases} \tag{1}$$

where,

- $rw_x^y(s_{c_i}, req)$ is the predicted rw value assigned to trustor x by trustee y for a criterion of a task s_{c_i} for the interaction req which occurs after the previous interaction tra'
- α and β are the improvement and disimprovement variables which are dynamically updated by the model
- γ, ζ, and λ are scaling variables that help more accurately update the predicted rw in specific scenarios
- Ω is the trustee's engagement threshold and θ is the trustee's un-engagement threshold. These are used when analyzing the direct feedback that the trustor has provided from the previous interaction (denoted by $SAT_x^y(s, tra')$)
- Φ is the trustee's threshold to determine whether a predicted rw is considered to be low

Thus, using equation 1, ITE updates its predicted rw values for each criterion of a task for a specific trustor. This equation is important since the values are used by the trustee to determine how many resources should be spent to meet the expectations of the trustor. Furthermore, this equation uses the α and β variables to determine how to perform these updates accurately. Since these variables will be dynamically updated by ITE to actively manage the rates at which the predicted rw values are updated at, performing more accurate updates to these variables will help ensure that the predicted rw values are updated more accurately to account for any behaviour shifts from trustors.

When deciding the amount of UG to provide to a trustor for a criterion of a task, the trustee computes an improvement value to state how much should be given to that specific trustor for the specific criterion when added to the minimum amount of UG that can be provided. This improvement value is denoted by $Improvement_x^y(s_{c_i}, req)$, which is the improvement to be provided to x by y for s_{c_i} during the interaction req. This value is the weighted combination of an explicit improvement value and a implicit improvement value. The explicit improvement value proposes to provide a UG for a criterion based on the predicted rw values from equation 1. This is the explicit improvement since the predicted rw value uses the feedback received from a transaction to know how to adjust itself.

Although the explicit improvement is a good approximation of the amount to be provided to the trustor, implicit improvement values for each criterion are continuously updated to reflect how much UG should be added based on implicitly retrieved information. A implicit improvement value for a specific criterion uses the α and β values, the trustor's engagement, and the predicted rw value for the corresponding criterion to continuously update itself. Thus, this is another example of how the dynamic α and β variables directly affect how the trustee adjusts its behaviours towards individual trustors. The implicit improvement values that are assigned to each criterion are updated via the equation below.

$$I\_Imp_x^y(s_{c_i}, req) = \begin{cases} (1+\alpha) * I\_Imp_x^y(s_{c_i}, tra') & x \in X_{ue}^y \ and \ rw_x^y(s_{c_i}) > \Phi \\ (1+\beta) * I\_Imp_x^y(s_{c_i}, tra') & x \in X_e^y \ and \ rw_x^y(s_{c_i}) < \psi \\ I\_Imp_x^y(s_{c_i}, tra') & else \end{cases} \tag{2}$$

where,

- $I\_Imp_x^y(s_{c_i}, req)$ is the implicit feedback value assigned to trustor x by trustee y for s_{c_i}, for the interaction req which occurs after interaction tra'
- ψ is the trustee's threshold to determine whether a predicted rw is considered to be high
- X_{ue}^y is the set of all trustors unengaged with trustee y and X_e^y is the set of all trustors engaged with trustee y

Using the improvement function, ITE computes the total UG to be provided to trustor x for an interaction by adding each UG value that is to be provided for each criterion. The total UG to be provided for task s is denoted by $UG_x^y(s, req)$ and is the summation of the UG values that are provided for each criterion, which will be denoted by $ug_x^y(s_{c_i}, req)$ in equation 4. Below is the equation for computing the total UG.

$$UG_x^y(s, req) = \sum_{i=1}^{p} Improvement_x^y(s_{c_i}, req) + MinUG^y(s_{c_i}) \tag{3}$$

where,

- $UG_x^y(s, req)$ is the total UG provided by y to x for task s during interaction req
- p represents the total number of criteria within task s
- $Improvement_x^y(s_{c_i}, req) + MinUG^y(s_{c_i})$ is the UG provided for a single criterion of task s, also represented by $ug_x^y(s_{c_i}, req)$

If the proposed UG amount is accepted by the trustor, the trustee is notified and provides the UG to the trustor. After performing the service, the trustor may provide a satisfaction value to the trustee to signify the trustor's satisfaction with the transaction. This is the direct feedback that is obtained by trustees and is used by ITE to update a trustee's behaviour towards trustors. Although this value can be predicted if not received or not trusted, the trustors will always provide this value within the scope of this paper. This satisfaction value is denoted by $SAT_x^y(s, tra)$ and is calculated with the following equation.

$$SAT_x^y(s, tra) = \sum_{i=1}^{p} \frac{w_x(s_{c_i}) * ug_x^y(s_{c_i}, tra)}{d_x(s_{c_i})} \tag{4}$$

- Where $w_x(s_{c_i})$ is the trustor's weighing of s_{c_i} and $d_x(s_{c_i})$ is the trustor's demand of s_{c_i}

Given satisfaction values from trustors (equation 4), ITE uses these values to calculate the average satisfaction rate that the trustee has received for a specific task. In ITE's paper, the calculated satisfaction rate below is not explicitly stated to be computed for only satisfaction values from a single trustor or for the satisfaction values from all trustors in the environment. Thus, the equation that will be used within this paper for ITE is set such that the satisfaction rate is computed for all satisfaction values that are stored for a specific task s. This is calculated with the equation seen below.

$$\overline{SAT_x^y(s)} = \frac{\sum_{j=1}^{N_{tr}^y} SAT_x^y(s,j)}{N_{tr}^y} \tag{5}$$

where,

- N_{tr}^y is the total number of transactions that y has performed for a specific task s
- $SAT_x^y(s,j)$ is the satisfaction between trustee y and trustor x for the transaction j of task s

ITE then computes the general satisfaction rate within the environment to determine whether the average satisfaction of trustors has increased or decreased since it has been last calculated. This is the RoC of trustor satisfaction in the environment for all trustors over two different points of time.

$$g^y = \frac{\widehat{\overline{SAT_x^y}}}{\overline{SAT_x^y}} \tag{6}$$

- Where $\widehat{\overline{SAT_x^y}}$ is the satisfaction rate (equation 5) that is calculated after the previous satisfaction rate $\overline{SAT_x^y}$ is calculated

Given this RoC value, ITE then updates its α and β variables to reflect how the trustee should update its behaviours based on the general satisfaction of trustors in the environment(s). This is a robust method of performing more accurate updates to the predicted rw values from equation 1 and to the implicit improvement values from equation 2. Accurately updating these values will help to ensure a more accurate UG is provided from equation 3. Within this paper, the updates to α and β will be done after a full round of transactions between the trustors and trustee. This ensures that the changes being made will more accurately reflect the environment. Although performing the updates in small batches may help stabilize the performance, similarly to what has been proposed by Mini-Batch Stochastic Gradient Descent, this ITE implementation will perform the updates after a full time step of transactions are completed [Rud16]. Below are the two update equations for α and β.

$$\hat{\alpha} = \alpha - \ln(g^y) \tag{7}$$

$$\hat{\beta} = \beta + \ln(g^y) \tag{8}$$

Performing these updates will ensure that ITE updates trustor behaviours at a rate that reflects the general satisfaction in the environment. The issue with this approach, which is addressed by CBITE, is that these updates do not work well when in a varied environment. When more trustor behaviours exist, the RoC from equation 6 may not accurately reflect the general environment at a given time step and may incorrectly shift the values of the α and β variables. Even if the RoC is computed for individual trustors, rather than for all trustors, the same problem can exist when a trustor has a temporary shift in their behaviour that will result in the α and β variables being updated too drastically.

3.2 CBITE's Architecture

With ITE's architecture presented and the dynamic improvement and disimprovement rate variable update process presented, the cluster-based approach to improve ITE's functionality will be proposed. CBITE uses the same overall architecture, but modifies the methodology that is used for dynamically updating α and β. Thus, the changes presented by CBITE directly affect equations 5, 6, 7, and 8 from ITE. The changes made to α and β also indirectly affect the results from equations 1, 2, and 4. In CBITE, the α and β variables are stored independently within each trustor model as α_x^y and β_x^y. This allows each trustor to have the variables tuned in different ways at each time step without affecting the general methodology that is used by the trustee to adjust itself towards other trustors. For each task s, CBITE will perform the following process. To simplify the notation used, the following variables and formulas will omit the task s from their definitions, but it is important to understand that this update process is done independently for each task.

First, CBITE will require the use of an Unsupervised Machine Learning algorithm, represented by Φ_{model}, to group the trustors which have interacted with the trustee at a given time step, represented by the set $X_{interacted}$, into k distinct clusters. The Unsupervised Machine Learning algorithm can be any clustering algorithm but should be able to cluster trustors well from a selected set of inputs. Since CBITE is a decentralized model, the clustering algorithm will need to be fit with the appropriate data before the clustering is performed. In open and dynamic environments, the algorithm choice will be important, but for this paper CBITE will utilize the K-means algorithm to keep the process simple. K-means is a well known and well researched distance-based algorithm that has flaws, such as its inability to effectively deal with outliers, but is a good candidate for use in the simulations that will be performed [KM14]. Details regarding K-means can be found in Machine Learning literature such as Flach's book which describes many different Machine Learning approaches [Fla12]. When selecting a clustering approach to use for Φ_{model}, surveys such as [RCC+19] and [XW05] can provide insight into which algorithm will best suit the target environment and task due to their detailed comparisons between different algorithms. For this implementation of CBITE, the clustering algorithm will cluster trustors using the trustor satisfaction that has been obtained following a completed transaction (equation 4) and the UG which the trustee has provided to the trustor in that same completed transaction (equation 3). The collection of the k clusters will be denoted as C.

For each of the k clusters, CBITE computes the average satisfaction rate for that cluster group. This directly extends equation 5 by using both the average satisfaction from all transactions that have been performed during or before the previous time step and the average satisfaction that has been received by the trustee from the trustors that are grouped within the specified cluster at the current time step. This allows each cluster to have independent average satisfaction rates that accurately reflect the change in satisfaction from similar behaving trustors at a given time step.

$$\overline{CBSAT_{c_i}^{y}} = \frac{\sum_{q}^{|TR|} SAT_{TR_q}^{y} + \sum_{j}^{|c_i|} SAT_{x_j}^{y}}{|TR| + |c_i|} \tag{9}$$

where,

- $|c_i|$ is the number of trustors in cluster i (where cluster i contains trustors defined by x_j)
- $SAT_{x_j}^{y}$ is the SAT provided by trustor x_j to trustee y for the task that has been completed during the current time step t
- $|TR|$ is the total number of transactions for a specific task that have been completed between the trustee and trustors before the current time step
- TR_q is the q$^{\text{th}}$ transaction of a specific task that has been completed by the trustee before the current time step

For each cluster-based average satisfaction rate from equation 9, CBITE calculates the general satisfaction rate of trustors in a specific cluster. This is a direct modification to equation 6 to consider the average satisfaction change within specific clusters rather than the average satisfaction change within the entire environment. This will help ensure that any updates that are made when using the cluster-based general satisfaction rate will not consider any drastic changes in average satisfaction from dissimilar trustors. If any relatively dissimilar trustors are grouped together, most of the clustered trustors should be similar enough to ensure a representative general satisfaction rate.

$$\overline{CBg_{c_i}^{y}} = \frac{\overline{CBSAT_{c_i}^{y}}}{\overline{SAT_{x}^{y}}} \tag{10}$$

Using the cluster-based general satisfaction rate from equation 10 allows the α_x^y and β_x^y variables that are assigned to individual trustors to be updated based on the RoC obtained from a group of similar trustors at a specific time step. Since this process is only done for trustors who have completed a transaction with the trustee at the time step, this cluster-based RoC will ensure that similar trustors receive fine-tuned updates. This process ensures that the α_x^y and β_x^y values are not updated by too much if there are temporary behaviour shifts and ensures that a trustor is accurately updated based on the group that they are clustered into. The α_x^y and β_x^y values are updated as seen below only for trustors within the corresponding cluster.

$$\widehat{\alpha_x^y} = \alpha_x^y - \ln(\overline{CBg_{c_i}^y}) \tag{11}$$

$$\widehat{\beta_x^y} = \beta_x^y + \ln(\overline{CBg_{c_i}^y}) \tag{12}$$

For trustors that have not completed a transaction with the trustee at a specific time step, they will still be updated by averaging the general satisfaction rates that have been calculated for each cluster of trustors. This ensures that the trustors are still being updated based on the general trends on the environment even if the trustor has not interacted with the trustee.

$$\overline{avgCBg_c^y} = \frac{\sum_i^k \overline{CBg_{c_i}^y}}{k} \tag{13}$$

Using this averaged cluster-based general satisfaction rate, the α_x^y and β_x^y variables of all non-clustered trustors are updated to ensure that the trustee adjusts its behaviour accordingly for every trustor in the environment.

$$\widehat{\alpha_x^y} = \alpha_x^y - \ln(\overline{avgCBg_c^y}) \tag{14}$$

$$\widehat{\beta_x^y} = \beta_x^y + \ln(\overline{avgCBg_c^y}) \tag{15}$$

Thus, this cluster-based approach to updating the dynamic improvement and disimprovement rate variables allows for more fine-tuned updates to be performed towards trustors that have interacted with the trustee at a given time step. This occurs without compromising the model's ability at also updating the variables for trustors which infrequently interact with the trustee. This methodology will allow for improved performance over ITE by being able to better reflect how a trustee should update its behaviour towards individual trustors which exhibit varied behaviours in an environment. Furthermore, this approach can be optimized for a specific environment or task by selecting the appropriate algorithm to use for Φ_{model}. As an example, perhaps the CBITE implementation will decide to use one clustering algorithm for tasks with many criteria values and will use a different clustering algorithm otherwise. Each Φ_{model} can also be specifically tuned for a task which the trustee performs in the environment.

4 System Evaluation

Following the presentation of CBITE's architecture and the changes that have been made when compared to ITE, this section will describe the simulation environment that is used and will present the results obtained from comparative simulated tests between ITE and CBITE. These tests will highlight the improvements made by the cluster-based approach and analyze how each model performs in varied environments.

4.1 Simulation Definition

The simulation environment that will be used in this section is designed to be similar to the environment that has been used to evaluate ITE in its paper. It uses the concept of assigning activity levels and demand levels to trustors to allow different behaviours to be present during the simulations. There are three different activity levels that are available to be assigned to trustors which each provide the probability for the trustor to actively seek interaction partners at a given time step. The trustor can have a high activity level, a regular activity level, or a low activity level. These each represent different probabilities of the trustor choosing whether or not to be inactive for a specific time step.

Each trustor is also assigned one of three demand levels which is used to determine a range of possible values that can be used for the trustor's demand values that are assigned to the various criteria of a task. These values are then used to compute both the trustor's rw for each criterion of a task and the satisfaction that is provided to trustors (the $d_x(s_{c_i})$ value from equation 4). The demand level also indicates the probability that is used to determine whether a transaction is good or bad. Different than the trustor's satisfaction, a good transaction is a transaction in which each UG value that is provided by a trustee to a trustor for the criteria of a task meet the specified demand percentage that is contained by the trustor via their demand level. Thus, a good transaction is a transaction which satisfies the overall demand of a trustor. A bad transaction is any transaction that is not a good transaction. A trustor is assigned either a low demand level, a regular demand level, or a high demand level to increase the variety of behaviours within the environment.

CBITE will be defined in the same manner described in section 3.2 and will use the K-means clustering algorithm. Since the simulations will consist of a single task with 10 criteria values, only one K-means model will be needed for each trustee. To help produce varied trustor behaviours, each trustor within the environment is assigned with a randomly selected activity and demand level (where there is a limited amount of each activity and demand type that can be assigned to the trustors). This random assignment is based on a random seed which ensures that the same trustor configuration is used when testing each model. This will produce a variety of unique behaviours that will help with evaluating how well ITE and CBITE can perform in varied environments. Since each performed test uses different trustor to trustee ratios, the simulated environments will capture the performance of each model in varying scenarios.

The simulation environments are programmed in Python using the Mesa Multi-agent Modelling library [MK15] and use the same scheduling design that is used in ITE's Java based simulations which use the MASON library [LCRP$^+$05]. Each of the tests that are performed run exactly the same, except that the trust establishment model which is used will be different. The instantiation of a simulation uses a random seed to ensure that everything is instantiated the same way for a specific test.

Trustors typically will use a trust value to determine which trustees to interact with at a specific time step. In these simulations, any active trustors will interact with each trustee during a time step. Doing this helps the simulation capture more data regarding the interactions without requiring a large increase to the total number of agents. Due to the diverse environment, this also showcases how well the trust establishment model can satisfy the trustors without simply providing much more UG to each trustor in the environment.

To calculate trust values, each trustor uses the simple trust evaluation model that is used within ITE's paper [AT20]. Within the environment, the maximum trust is 1 and the minimum trust is 0. The total trust that is assigned to a trustee by a trustor is the weighted combination of the direct and indirect trust values for that trustee. The direct trust of a trustee for a specific trustor is equal to the average satisfaction that the trustor has computed for each transaction with the trustee (equation 4). The indirect trust that is used by a trustor for a trustee is the average of the direct trust values from all other trustors regarding that trustee. Although in a real environment this communication will not always be possible, in this simulation all agents can freely communicate and cooperate with one another. Below is the equation to retrieve the total trust of a trustee.

$$trust_x^y = direct\_trust * 0.5 + indirect\_trust * 0.5 \tag{16}$$

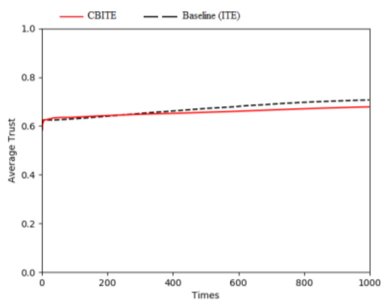
When running the simulations, the average direct trust values of all trustees from all trustors, the average UG that is provided to all trustors from each transaction, and the rate of good transactions out of all transactions will be tracked. These will provide insight into the performance of each model by determining how well the models manage the trade-off between providing more UG in exchange for increased trust and how well the models can accurately meet all the needs of trustors by analyzing the rate of good transactions. Table 1 exhibits the core hyperparameter values that are used throughout the simulation. The selected hyperparameter values are set to closely match those used within ITE's paper for a more accurate comparison. Any parameter values not exclusive to CBITE are used between both models.

Table 1: Core hyperparameter values for the simulations

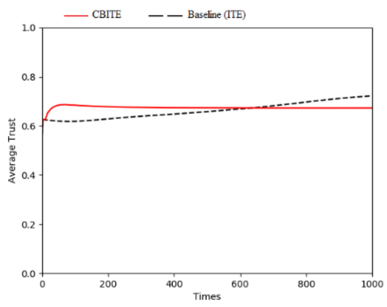
Parameter	**Assigned Value**
Number of agents (N)	100
α (initial value)	0.02
β (initial value)	-0.01
Number of clusters (k)	5
θ	0.75
λ	0.1
ζ	1
γ	1
Ω	0.5
Φ	0.25
ψ	0.5

4.2 Simulation Results

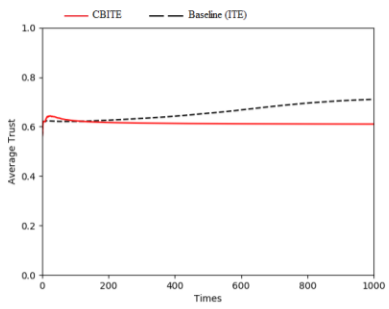
To display the performance benefits from CBITE, we will compare CBITE to ITE in three tests. Each test will use a different trustor to trustee ratio to see how well each model handles varying amounts of trustor behaviours. In the tests, each agent is only considered to be a trustor or be a trustee. In the first test, there will be 12 trustors and 88 trustees. This allows a comparative analysis of how well each model performs when there are far more trustees than trustors. This limits the amount of trustor behaviours in the environment as well. Next, the models will be compared when there are 24 trustors and 76 trustees to better understand how each model performs when there are more trustor behaviours that are introduced in the environment. Finally, the third test will involve 36 trustors and 64 trustees to introduce even more diversity into the environment while still maintaining a good number of trustees. As more trustors are introduced into the environment, the number of interactions drastically increase and will affect the tracked metrics accordingly. Figure 2 exhibits the results that have been obtained from the simulated tests.



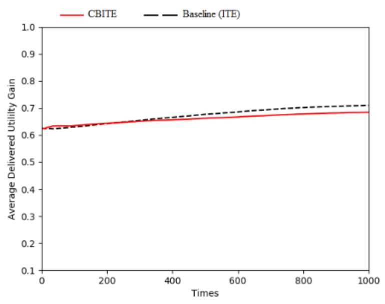
(a) Average Direct Trust: 12 Trustors, 88 Trustees



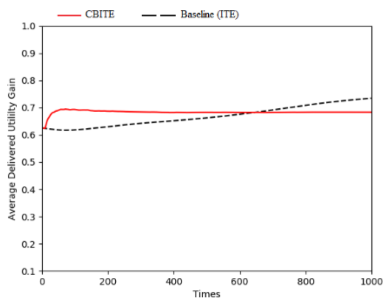
(b) Average Direct Trust: 24 Trustors, 76 Trustees



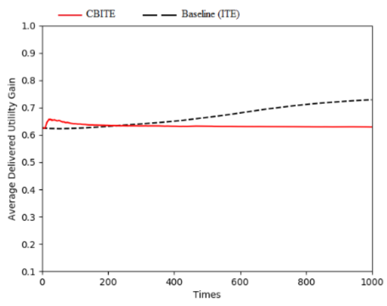
(c) Average Direct Trust: 36 Trustors, 64 Trustees



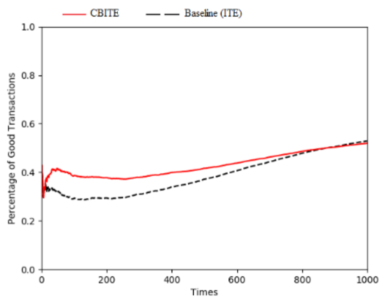
(d) Average Delivered UG: 12 Trustors, 88 Trustees



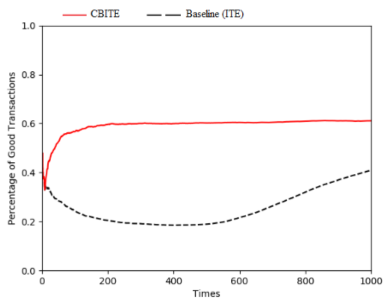
(e) Average Delivered UG: 24 Trustors, 76 Trustees



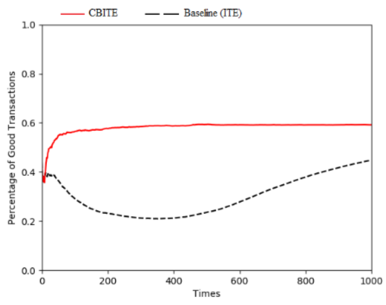
(f) Average Delivered UG: 36 Trustors, 64 Trustees



(g) Good Transaction Rate: 12 Trustors, 88 Trustees



(h) Good Transaction Rate: 24 Trustors, 76 Trustees



(i) Good Transaction Rate: 36 Trustors, 64 Trustees

Figure 2: Comparative results between ITE (black) and CBITE (red)

Figure 2 showcases that CBITE's cluster-based approach to updating the dynamic variables changes the observed performance in all metrics. It important to note that the results have a more significant difference when there are more trustors and less trustees in the environment. This makes sense given that the cluster-based approach will perform more accurate updates when there are more trustors to be clustered at each time step. Overall, CBITE receives a minor decrease in the average direct trust of trustees, but also reduces the UG that is being provided. This decrease in trust makes sense given the corresponding reduction in the UG that is provided to trustors. Despite this, the overall decrease in trust and provided UG from CBITE, which is further reduced when there are more trustors in the environment, is not too significant when compared to ITE's results.

When comparing the average direct trust and the average provided UG from the third test, see plots (c) and (f), to the first two tests, see plots (a), (b), (d), and (e), we see that CBITE is learning how to maintain a high enough trust value without providing too much UG. When analyzing ITE's performance, plots (c) and (f) demonstrate that ITE does not always attempt to balance the provided UG with the corresponding trust when working with a diverse set of trustors. Although increased trust is beneficial, the amount of UG that is spent for it can be problematic depending on the resources that are available to the trustee.

Despite the lower average direct trust and provided UG, CBITE receives a higher good transaction rate for most of all three tests. The good transaction rate also increases early on, without seeing the initial dip that is observed by ITE when there are more varied trustor behaviours in the environment (see plots (h) and (i)). Furthermore, since the good transaction rates from both models in plot (g) converge at the end, this also displays that CBITE benefits more from operating in diverse MASs. The drastic improvement to the good transaction rate in plots (h) and (i) by CBITE is indicative of CBITE's improved ability at better meeting all the needs of a trustor. Since the corresponding average provided UG is lower, see plots (e) and (f), it is clear that CBITE can better learn the expectations of trustors while providing less overall UG in the environment. Thus, trustees using CBITE are shown to better learn how to shift their behaviours towards specific trustors in varied environments when compared to ITE.

5 Discussion and Conclusion

From the simulation results that have been observed in the previous section, the cluster-based approach to updating ITE's α and β variables for each trustor results in specific performance improvements that are more prevalent as more trustor behaviours are present in the environment. Unlike ITE, the good transaction rates that are observed from CBITE in more diverse environments immediately increase and are quickly stabilized. This observation clearly indicates that the independent updates that are performed for groups of similar trustors ensures that the fine-tuned general satisfaction rate from equation 10 better represents the needs of similar trustors. Given the decreased UG that is being provided and the corresponding trust decrease, CBITE is allowing trustees to prioritize specific trustors as interaction partners.

The stabilized CBITE results in the second and third tests, as seen in plots (b), (c), (e), (f), (h), and (i) of Figure 2, showcases that trustees can learn varied behaviours in the environment quicker than ITE. Although the trustors from the simulations do not actively change their behaviours, CBITE's ability to quickly learn the needs of trustors in an environment will allow CBITE to update itself more quickly to meet shifting trustor needs when compared to ITE. Despite the observed results being less drastic in less diverse environments, CBITE still achieves comparable results with the benefit of potential large increases to performance as new agents enter the environment. This is because a trust establishment model that is designed for open and dynamic MASs should be capable of quickly learning the behaviours of a trustor to ensure that the trustee appropriately allocates UG to that trustor.

Although the simulated environment is not distributed, the same tests can be performed in simulated distributed environments with an appropriate method for agent communication. Thus, similarly to ITE, CBITE is a functional trust establishment model in both singular and distributed MASs. The tests performed in this paper provide insight into how CBITE improves upon ITE's ability to handle varied trustor behaviours. Since the tests use the same concepts that are used by ITE's paper, this ensures a fair and accurate comparison between the two. However, as more advancements are made in trust establishment model research the models can also be tested based on their performance when handling trustor attacks or when splitting the types of trust establishment and trust evaluation models that are used in the same simulated environment. This can further exhibit any improvements that can be made to CBITE to ensure that the ITE-based model is as robust as possible. Similarly, CBITE can be tested with real datasets which may contain larger sets of dissimilar trustors. Since CBITE is intended to help when there are similar trustor behaviours in an environment, performing these tests can help with determining the optimal clustering algorithms and corresponding hyperparameters to use when working in these different scenarios. Further exploration regarding the tuning of the hyperparameters that are used by ITE-based models can also lead to improved guidance on how to use the models to best target expected agent behaviours in different domains.

To conclude, the cluster-based update process that is applied to ITE provides a refinement to ITE's initial design by allowing trustees to quickly learn how to completely satisfy trustors without providing too much UG. In this paper, a trust establishment model named CBITE is presented and contains the new cluster-based improvement and disimprovement rate update module. Despite using a simplistic Unsupervised Machine Learning algorithm in the simulations, CBITE still provides improvements when adjusting a trustee's behaviour in varied environments. As trust establishment model research continues to advance, CBITE will demonstrate how trustor-based clustering can improve trust establishment model performance and will allow for further refinements to be made at optimizing the trust establishment model.

References

[AT15] A Aref and T Tran. Rlte: A reinforcement learning based trust establishment model. In *2015 IEEE Trustcom/BigDataSE/ISPA*, pages 694–701, 2015.

[AT17] A Aref and T Tran. Acting as a trustee for internet of agents in the absence of explicit feedback. In *MCETECH 2017*, pages 3–23, 05 2017.

[AT20] A Aref and T Tran. An integrated trust establishment model for the internet of agents. *Knowledge and Information Systems*, 62:79–105, 2020.

[BNS11] C Burnett, TJ Norman, and K Sycara. Trust decision-making in multi-agent systems. In *Twenty-Second International Joint Conference on Artificial Intelligence*, pages 115–120, 2011.

[Fla12] P Flach. *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Cambridge University Press, 2012.

[KM14] M Kaushik and B Mathur. Comparative study of k-means and hierarchical clustering techniques. *International Journal of Software and Hardware Research in Engineering*, 2(6):93–98, 2014.

[LCRP+05] S Luke, C Cioffi-Revilla, L Panait, K Sullivan, and G Balan. Mason: A multiagent simulation environment. *SIMULATION*, 81(7):517–527, 2005.

[MK15] D Masad and J Kazil. Mesa: An agent-based modeling framework. In *14th PYTHON in Science Conference*, pages 53–60, 2015.

[RCC+19] MZ Rodriguez, CH Comin, D Casanova, OM Bruno, DR Amancio, LdF Costa, and FA Rodrigues. Clustering algorithms: A comparative approach. *PLoS ONE*, 14, 2019.

[Rud16] S Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[Sen13] S Sen. A comprehensive approach to trust management. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-Agent Systems*, (AAMAS '13), page 797–800, Richland, SC, 2013.

[SS05] J Sabater and C Sierra. Review on computational trust and reputation models. *Artificial Intelligence Review*, 24(1):33–60, 2005.

[TCLK14] T Tran, R Cohen, E Langlois, and P Kates. Establishing trust in multiagent environments: Realizing the comprehensive trust management dream. *TRUST@ AAMAS*, 1740:35–43, 2014.

[XW05] R Xu and D Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, 2005.

[YSL+13] H Yu, Z Shen, C Leung, C Miao, and VR Lesser. A survey of multi-agent trust management systems. *IEEE Access*, 1:35–50, 2013.