Reputation System Aggregation and Ageing Factor Selection using Subjective Opinions Classification

Abdelmageed Algamdi, Frans Coenen and Alexei Lisitsa
Department of Computer Science
University of Liverpool
Liverpool, UK
(A.Algamdi - f.coenen - a.lisitsa)@liverpool.ac.uk

Abstract—**This paper presents an adaptive approach to the selection of appropriate values for ageing and aggregation factors in reputation systems, based on a novel approach to subjective opinions classification. The idea is that users express the service evaluations as binomial subjective opinion which changes the system's old rating values using aggregation and ageing factors. The approach has three stages, first we use randomly generated opinions which are classified and visualized to the testers and ask the tester to estimate the resulting rating value. The second stage is to collect the estimated ratings as well as the calculated ratings from the suggested formula using aggregation and ageing with initial aggregation and ageing values. The third stage is to use the collected data from different users at different situations to adaptively change the aggregation constant and ageing factor so that we have minimum average error between the suggested formula and the tester's opinions. The same procedure can be applied on calculation of trust for various systems.**

Keywords-component; Reputation systems; Aggregation factor selection; Ageing factor selection; Subjective opinions

I. Introduction

Over the last decade, cloud computing has become an important area of research. Cloud computing offers cheap and reliable IT services to users such as storage services, complex and fast calculation and so on. Despite the many services and benefits offered over the cloud, users still have concerns about issues such as privacy, data protection and service quality. Reputation systems were designed to quantify the trust that cloud users can place in a service [1-3]. This can be calculated in various ways. The reputation system presented in [9] considers only self-assessment by providers after releasing a service and submitting required certificates through the Cloud Trust Protocol (CTP) [4,5] to the cloud audit. In previous work the authors presented an approach that allowed cloud service users to conduct assessments based on evidence provided by other cloud users that had reviewed providers' self-assessment's and modify such assessments in a way that reflected their satisfaction [6]. The assessment was done by modeling cloud user opinion as a subjective binomial opinion [12-14] and classifying this user opinion into one of six rating classes, which after that helps to decide the ratio of the aggregation constant used in the update action. The aggregation constant is the a fixed value in which we multiply it with the rating class value to get the update effect of the user assessment using the aggregation operation. The resulting trustworthiness value for a service can be also calculated by doing not only aggregation but also ageing between the current and the pervious opinions which aggregates the current opinion update effect with a ratio from the old assessments history of the same user. The problem faced is that the results are highly affected by the values of the aggregation and ageing factors. This paper thus presents a good selection for the aggregation constant and similarly the ageing factor that cause minimal error in average between the calculated update and the human --user or tester—opinion.

More specifically, in this paper, an adaptive method to estimate the aggregation constant and ageing factor enables user to first assume different values for the parameters, then calculate the update of various randomly generated opinions, asking testers for estimation based on the opinions visualization and finally using these values to provide good estimation for the parameters.

The rest of the paper is organised as follows: Section 2 presents the background information and related work. Section 3 demonstrates the proposed opinion classification method using Barycentric coordinates and Section 4 demonstrates the ways which can be used to update the trust values generated before either by aggregation or by aggregation with ageing. It also shows how we can choose the best values for the aggregation constant and the ageing factors using some experiments. Section 5 concludes this paper.

II. Background and Related Work

A. *Trust and Reputation (TR) Systems*

In cloud computing, cloud trust management systems [7-9] are responsible for calculating the trustworthiness for all the services offered over the cloud and finding the trustworthy ones. Trust and reputation (TR) systems [1-3] are example of these systems which support trust management based on service environment attributes such as security, compliance and data governance. Some of these systems use the cloud trust protocol (CTP) [4-5] as a source of information where any cloud user can request and retrieve documents and certificates about the services. After that, trust assessment techniques are used to extract the users' opinions and convert them somehow to update the trustworthiness values of the services being assessed. There are two types of assessments techniques either to do only the provider self-assessment [9] or to do the self-assessment plus a lot of users assessments which update the trustworthiness values generated from the self-assessments. The self-assessment uses the Consensus Assessments Initiative Questionnaire (CAIQ) [10] questionnaire which covers the main attributes – compliance, data governance such as in [6,9] while the user assessment uses the Smals ICT [11] for society group questionnaire generated for normal or experts clients as in [6]. The answers of the questionnaire are either Yes, No or Unknown. So, these answers are modeled as subjective binomial opinion which will be later affects the latest trust value stored for the service being assed. The questionnaire designed for users covers four main characteristics expected of the cloud service: Governance, Identity and Access Management (IAM), IT Security and Operational Security.

B. *Binomial Subjective Opinions*

The subjective opinions consider the uncertainty, the belief ownership and incomplete knowledge which is essential for the assessment methods. As the questionnaire answers are either Yes, No or Unknown, the binomial subjective opinions are the best type to deal with this situation [12-14].

A binomial opinion over a variable x is represented in subjective logic by a quadruple of real numbers $\omega_x = (b_x, d_x, u_x, a_x)$ all from the interval [0…1], subject to the constraint $b_x + d_x + u_x = 1$. They are referred to as *belief*, *disbelief*, *uncertainty* and *relative atomicity* of x, respectively. Both the user and the provider opinions are expressed as binomial opinions. These binomial opinions are calculated based on the answers of multiple choices questionnaires designed specifically to assess the service from two different views (provider and user) [12-14].

The binomial opinion can be visualized inside Barycentric Coordinates. As shown in Fig 1, the Barycentric Coordinates are simply an equal side triangle with vertices belief (b_x), disbelief (d_x) and uncertainty (u_x). The opinion is represented as a center of gravity (barycenter or geometric centroid) of locating three masses M_A, M_B and M_C at the triangle vertices. These masses are located over three axis perpendicular over the opposite triangle side of each vertex. These masses are represented b_x, d_x and u_x respectively. The base rate a_x is represented by a point in the base. The line connecting the u vertex to the point represented by a_x is called the director. The projected probability P_x of an opinion ω_x can be determined by drawing a line from the opinion point ω_x to the base and parallel to the director line [12-14].

For homogenous Barycentric coordinates, the edges are normalized in order to achieve $b_x + d_x + u_x = 1$. The projected probability can be calculated as follow, $P_x = b_x + u_x a_x$ [13].

In this paper, we assume that we have the overall subjective opinion which reflects the user opinion towards a service. The aim is to show how this opinion updates the latest trustworthy scalar value stored using aggregation and ageing operations.

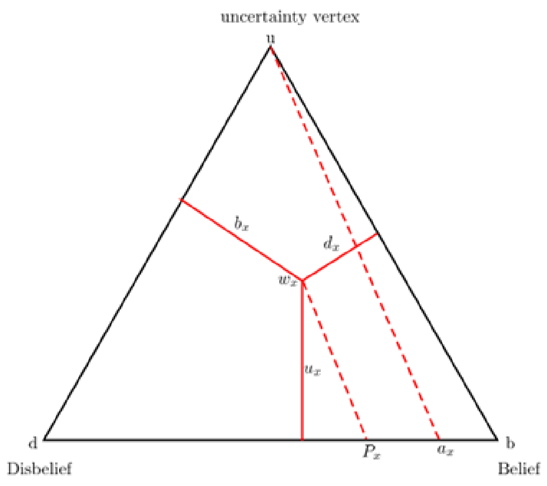

Figure 1: Binomial opinion representation inside the Barycentric coordinates.

III. The proposed opinion classification method

The basic idea of this classification method is based on the representation of the binomial opinion inside the Barycentric coordinates [13]. From the triangle shown in Fig 1, we can split the inside area into sub areas where each sub area has common ranges for b_x, d_x and u_x and can be represented as a fuzzy meanings. As shown in Fig 2, our classification approach [6], we classify any subjective opinion into one out of 6 different classes. These classes named as **very good**, **good**, **very bad**, **bad**, **un-named**, and **very uncertain** classes. These fuzzy classes can be converted later into scalar ratings like $k = 1$ for class named very good and $k = -1$ for the class named very bad. Table 1 shows the six rating classes with their classification schemes.

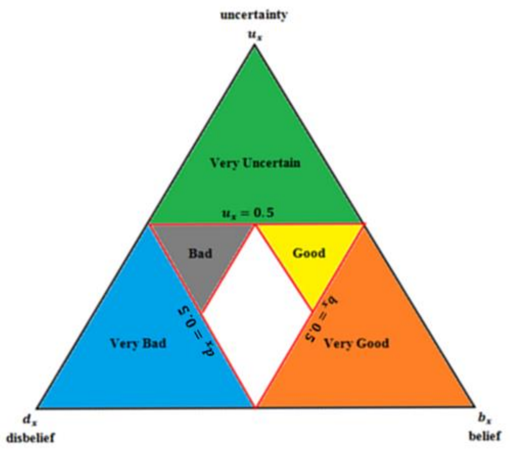

Figure 2: A binomial opinion rating classification.

Region	Belief	Disbelief	Uncertainty
Very Good Certain	$b_x \geq 0.5$	$d_x < 0.5$	$u_x < 0.5$
Good Certain	$0.25 < b_x < 0.5$	$d_x < 0.25$	$u_x < 0.5$
Very Bad Certain	$b_x < 0.5$	$d_x \geq 0.5$	$u_x < 0.5$
Bad Certain	$b_x < 0.25$	$0.25 < d_x < 0.5$	$u_x < 0.5$
Unnamed Certain	$0.25 \leq b_x < 0.5$	$0.25 \leq d_x < 0.5$	$u_x < 0.5$
Very Uncertain	---	---	$u_x \geq 0.5$

Table 1

The value of k is determined as follow and depends on the rating class for the consumer opinion:

- For very good and certain class ($k = 1$).
- For good and certain class ($k = \frac{1}{2}$).
- For very bad and certain class ($k = -1$).
- For bad and certain class ($k = -\frac{1}{2}$).
- For un-named and certain class ($k = \frac{1}{4}$ if $P_x \geq 0.5$ and $k = -\frac{1}{4}$ if $P_x < 0.5$)
- For very uncertain class ($k = 0$).

IV. UPDATING THE TRUST

The provider has the ability to do the first assessment for its service by answering the CAIQ questionnaire which produces initial scalar trust value. This can be done by collecting the provider's opinion using the approach suggested by the author in [6]. Then, the opinion will be classified into a rating class using the classifier described in Section 3 and also the class rating value k will be obtained. Finally the k value which ranges from -1 to +1 will be scaled to another scalar value from 0 to 100 respectively. The cloud users also have the ability to reassess the service after using it and submit their opinions towards the service operation in the form of questionnaire as mentioned before in section 2. The user opinion updates the latest scalar trust value by either doing aggregation only or aggregation with ageing. The update depends on which rating class the user opinion has been classified into.

A. *Update by aggregation only*

There are a lot of methods to do the aggregation. In this paper, we do aggregation of ratings by using the simple addition. This can be done by using an aggregation constant $\lambda \in [0,1]$. The aggregation has no effect on the original ratings if $\lambda = 0$ while it has the largest effect with $\lambda = 1$.

Let's define

- $r_{y,t}$ is the initial rating value (only provider) generated from the provider self-assessment for service y.
- $R^x_{y,t}$ is the old rating value (provider and user x) over time t for service y.
- $R^x_{y,(t+1)}$ represents the overall (provider and user x) new accumulated rating value after time period $t+1$ for service y.
- $R_{y,(t+1)}$ represents the overall (provider and all users) new accumulated rating value after time period $t+1$ for service y.

In order to give a permission to any user to do the assessment any number of time, our method of calculating the reputation (rating) value generated from any agent x towards service y depends not only on the current opinion rating class k_{t+1} but also on the previous one k_t. The idea behind doing another assessment is to remeasure the reputation again and produce new value instead of the generated old one. so, our method based on updating the overall reputation value with the new opinion and removing the old one for all the users that do many assessments.

Assuming that the value of previous opinion rating class for those agents that do their first assessment is $k_t = 0$. The new accumulated rating $R_{y,(t+1)}$after time period $t+1$ can be expressed as:

- For the first user assessment: $R^x_{y,(t+1)} = \lambda' + r_{y,t}$ where $0 \leq \lambda \leq 1, \lambda' = (k_{t+1} - k_t)\lambda$.
- For any user assessment except the first one: $R^x_{y,(t+1)} = \lambda' + R^x_{y,t}$ where $0 \leq \lambda \leq 1$,
- $\lambda' = (k_{t+1} - k_t)\lambda$.

The overall reputation (rating) generated from all users $x \in X$ -where X is the set of all users did the assessments- is simply generated from the average overall users' ratings as follows:

$$R_{y,(t+1)} = \frac{\sum_{x \in X} R^x_{y,(t+1)}}{|X|}$$

We assume that the overall reputation $R_{y,(t+1)}$ has lower bound of 0 and higher bound of 100. If the calculated overall reputation lies out of the boundaries we modify it to lie on the boundaries 0 if smaller and 100 if bigger.

B. *Aggregation constant estimation*

In this sub-section, the proposed method runs a lot of supervised tests in order to provide good estimation for the aggregation constant λ in the range between 0 and 1. We mean by the word supervised that we will have human testers who test whether our approach correct and we use these reviews to readjust the parameters values on our approach. The procedure is listed below:

1. Generate n- random subjective opinions $\omega_X = \{\omega_{x_1}, \omega_{x_2}, \dots, \omega_{x_n}\}$ that act as the overall opinions generated from the assessments done by the set of users $X = \{x_1, x_2, \dots, x_n\}$ assessments for a given service y.
2. Find the set of opinions' ratings $K = \{k_1, k_2, \dots, k_n\}$.
3. Visualize the set of random opinions ω_X as shown in Fig 3.
4. Choose an initial value of $\lambda \in [0,1]$.
5. Ask the human tester to give estimation to the final trust value $R^{est}_{y,(t+1)}$ after showing him the visualization picture, the value of λ and the latest trust just before the process.
6. Let the program calculate the updated trust value using the following equation.

$$R^{calc}_{y,(t+1)} = \Sigma_{\{i \leq n\}} k_i \times \lambda + R_{y,t}$$

Where,
$R_{\{y,(t+1)\}}$ is the updated trust value of the service y.
$R_{\{y,t\}}$ is the latest trust value of service y before this process.

7. Find the absolute error $e = |\, R^{calc}_{y,(t+1)} - R^{est}_{y,(t+1)}|$ and store the values (λ, e).
8. Repeat steps 1 to 7 with different values of n and λ.
9. Repeat the whole process from 1 to 8 with $m \geq 1$ testers.
   We can do this procedure for the values $n \in \{1, 5, 10, 15, 20\}$ and $\lambda \in \{0.1, 0.2, 0.4, 0.5, 0.8, 1\}$.
10. For every λ find the average absolute error
    $$e_{\lambda,\ \{avg\}} = \frac{\Sigma\, e_\lambda}{m \times n}$$
11. Select the best λ value that minimizes the average absolute error.

We've implemented software using java and Microsoft Access database (MS access) that provides a graphical user interface (GUI) that enables both the cloud providers and the cloud users to do their assessments. The login window supports three different logins: cloud provider, cloud user and tester. For the cloud provider logins, we enable the addition of new services to the system and also do the provider self-assessment by answering the CAIQ questionnaire. For cloud user logins, the user will be asked to enter his/her details and choose the service to be assessed. Then, the user questionnaire will be appeared to the user to be answered. For both provider and user assessments, we enables YES, NO, and UNKOWN answers to the MCQ questions. Then the provider/user opinion is collected using the approach in [6] and classified using the classifier in Section 3. For user assessments, the update action will be performed either by using aggregation only as described in Section 4-A or by using aggregation with ageing using the approach described later in Section 4-C. All the assessments data like the questionnaire answers, opinions and the update effects are stored in the database.

On the other hand, the tester login is designed to test the suggested assessment system by human testers. For testing, we use randomly generated answers (values) for the user questionnaire. Once we have the random answers, we use the same approach in [6] to collect the overall random user opinion and visualize it inside the Barycentric triangle shown in Figure 2. The visualization window will be outputted to the user with the rating classes shown also on the same figure and ask the tester to give a guess for the overall trust value. The tester will be informed with the latest trust as well as the aggregation constant value before giving his estimation. Note that, the tester will not be informed with the k-values for each rating class as we want to compare the suggested approach with the human feelings. The tester estimation as well as the actual calculated trust value using the approach in Section 4-A will be stored in the database. The tester repeats this procedure with different values of aggregation constant and different number of random users opinions to the same service.

For the test, the tool asks the tester to enter his details and then the testing procedure continues. For every tester, we do lot of tests with different λ and n values. The value of n represents how many random opinions will be outputted to the tester for the same service. Once we show the visualized opinions to the user and informing him the values for aggregation constant and the latest trust for the service, the tester guesses the new trust value for the situation while our program calculates the trust value from the equations shown before ad save these values on a database. This situation will be repeated over all the possible combinations of values for n and λ.

We run this test over $m = 10$ testers and the average absolute error relation is shown in Fig 4.

The horizontal axis represent the values of $\lambda \in \{0.1, 0.2, 0.4, 0.5, 0.8, 1\}$ and the vertical axis demonstrates the average absolute error $e_{\lambda,\ \{avg\}}$. We can conclude that for our situation the best value for λ that result in minimum error in average is 0.1.

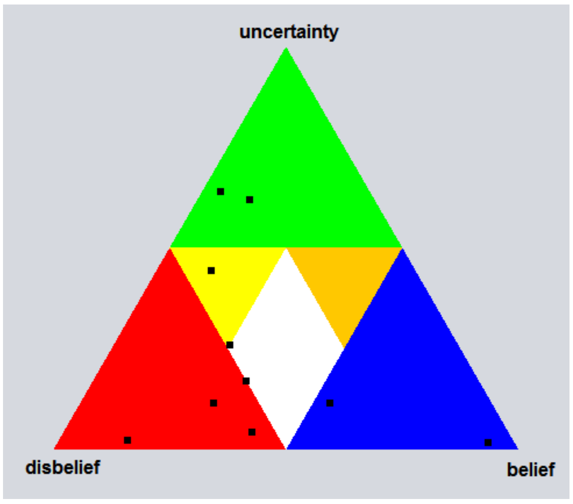

Figure 3: A snapshot of the visualization for 10 random opinions.

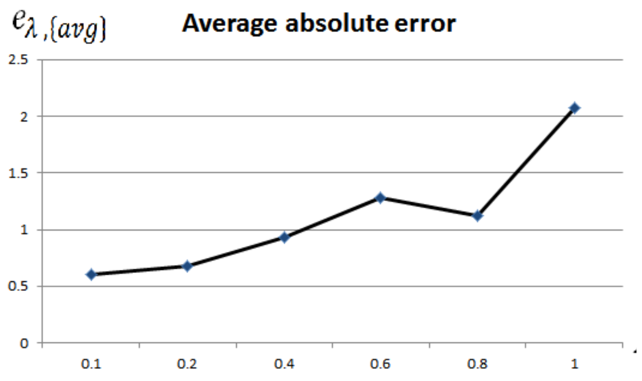

Figure 4: Relation between the aggregation constant and the average absolute error after doing 10 complete tests.

The previous way of collection users' opinions depends only on the last assessment of each user by removing all the history created before. Another way of collecting users' opinions is to do the aggregation between the last assessment outcome for each user with an aged value of the history generated by the same user. Let's define an aging factor $\Lambda \in$ **[0,1]** The value of Λ determines the history ratio of the user's opinions that contributes with the new opinion to generate the current reputation value of the user towards any service. The history is forgotten as shown in the previous method if $\Lambda = 0$ and contributes with the full ration if $\Lambda = 1$. [6]

We can define an update $\delta_{x,t}$ for every user x assessed the service. The update value for can be calculated as follows:

- For the first user assessment, there is no assessment history for the user x towards the service y. So, there is no need for doing any form of **ageing** here in the first user assessment where λ is the aggregation constant.
$$\delta_{x,t} = k_t\lambda.$$
- For any assessment except the first one, we have opinions in the past for the user x so, we should use the ageing factor now.
$$\delta_{x,t} = k_t\lambda + \Lambda\delta_{x,(t-1)}.$$
    - For decreasing the effect of the history we use $\Lambda = 0.01$ (very close to 0).
    - For increasing the contribution of the history in the calculation of the current reputation value we use $\Lambda = 0.99$ (very close to 1).

The overall reputation (rating) generated from all the users $x \in X$ towards the service y where X is the set of all users did the assessments is the sum of all the updates done by all the users X plus the initial trust r_y generated from the provider self-assessment. This is shown as follow

$$\mathbb{R}_{y,(t)} = \sum_{x \in X} \delta_x + r_y$$

We assume that the overall reputation $\mathbb{R}_{y,(t)}$ has lower bound of 0 and higher bound of 100. If the calculated overall reputation lies out of the boundaries we modify it to lie on the boundaries 0 if smaller and 100 if bigger.

D. Ageing factor estimation

In this sub-section, we also run a lot of supervised tests like the previous section but now we aim to find the best value for ageing factor results in minimum error in average. We are going to operations in order to update the trust value. The first one is the ageing with a factor of Λ which affects the latest trust value (the past) and the second operation is the aggregation with aggregation constant $\lambda = 0.1$ (as concluded from sub-section B) which represent the update effect of the current user opinion. The procedure is listed down below:

1. Assume that the aggregation constant always $\lambda = 0.1$.
2. We are going to do the following steps for the testers $\{t_1, t_2, \ldots, t_m\}$, $m \in \mathbb{N}_{\{>0\}}$.
3. We do the following steps for various ageing factor $\Lambda \in \{0.01, 0.2, 0.4, 0.6, 0.8, 0.99\}$.
4. For every tester t_i , we ask him to do n-random test trials with every single value of Λ.
5. For every trial, we ask the tester to select positive random integer l which represents how many random opinions will be generated. Each opinion represents a different time frame.
6. We visualize each opinion alone as shown in Fig. 5 and ask the tester to give estimated value about the updated trust value giving him the latest trust value (the past).
7. We let the program calculate the overall updated trust value as described in sub-section C.
8. Calculate the absolute error $e_{\{\Lambda\}}$ between the estimated value and the calculated value
9. Repeat steps from 5 to 8 for the rest ($l - 1$) trials.
10. Repeat steps from 5 to 9 with all values of Λ.
11. Repeat steps from 5 to 10 with all the other ($m - 1$) testers.
12. After finishing all the testers we calculate for every Λ the average absolute error $e_{\Lambda,\{\text{avg}\}} = \frac{\Sigma e_\Lambda}{l \times m}$.
13. Select the best ageing factor Λ with the minimum average absolute error $e_{\Lambda,\{\text{avg}\}}$.

For Fig. 6, the test procedure is that we have 10 testers. Each tester do tests for $\Lambda = \{0.01, 0.2, 0.4, 0.6, 0.8, 0.99\}$. At each value the tester is doing 10 test situations such that at trial i we generate i random opinions at different time frames. For example, the trial i=1, we generate only one opinion to a random user (this is the first time for this user to do the assessment). At i=2, we generate two opinions to the random user which means that the user had a previous assessment (his first one) plus the new assessment and so on. From Fig. 6, we can conclude that the best value for the ageing factor that could cause minimum absolute error in average is 0.01.

For our users system – cloud users assessments – we prefer low values for both aggregation constant and ageing factor. For aggregation constant, the reason is that we have a very large number of the cloud users nowadays so, if we have a large aggregation constant, the trust value could change rapidly (increasing or decreasing) after small number of assessments which is not fair as the majority of users still didn't do the assessment yet. For the ageing factor, we prefer small values because for large Λ values we give the cloud user the power to update the trust with portions bigger than his right. We can explain this as follow:

Suppose that we have aggregation constant λ and ageing factor $\Lambda = 1$. Let's consider a service y with 10 users. For every user, it is very fair to give him an update of $\pm 0.1F$ where F is the maximum trust value.

For a single user, for his first assessment the user contributes an update of $\pm\lambda$ at most. For the second assessment, the user could contribute $\pm\lambda$ for his present opinion plus another $\pm\lambda$ from his opinion in the past as the ageing ration $\Lambda = 1$. So, with only one user the trust value could reach F after n

assessments without any consideration for the other 9 users' opinions.

If $\Lambda = 0.01$, the maximum update a user can do is $\pm(\lambda + 0.02\lambda) = \pm 1.02\,\boldsymbol{\lambda}$ which is acceptable because for any user, all of his opinions towards a service contributes only with $\pm 1.02\,\boldsymbol{\lambda}$ which is very near to the setting if one user gives only one opinion for every service that has a maximum update of $\pm\boldsymbol{\lambda}$.

V. CONCLUSION

We can conclude that using the Barycentric classifier, we can get a realistic trust update for the cloud user assessments using only aggregation constant of value 0.1 without any consideration for the past where each user contributes always with his very recent opinion or if we want to consider the past we use ageing factor with small values like 0.01 in order to be fair and reliable updates. Secondly, the more partitions we uses inside the Barycentric classifiers, the more precision we get in the classification and hence on the updates**.**

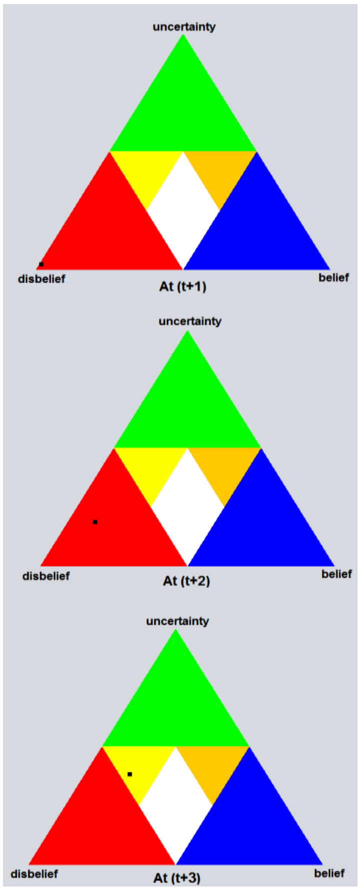

Figure 5: Three random opinions at three consecutive time slots (t+1), (t+2) and (t+3).

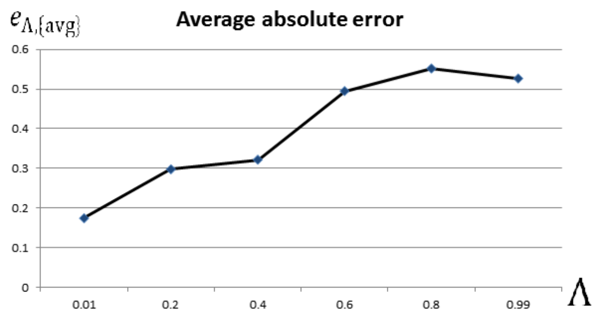

Figure 6: Relation between the ageing factor and the average absolute error after doing 10 complete tests.

REFERENCES

[1] Jøsang A, Ismail R, Boyd C, "A survey of trust and reputation systems for online service provision", Decision Support Systems 2007, 43(2),P(618 – 644).

[2] A. Jøsang, X. Luo and X. Chen, "Continuous Ratings in Discrete Bayesian Reputation Systems", The International Federation for Information Processing, vol. (263), P(151--166), 2008.

[3] A. Whitby, A. Jøsang and J. Indulska, "Filtering Out Unfair Ratings in Bayesian Reputation Systems", Autonomous Agents and Multi Agent Systems Conference, 2004.

[4] Knode, Ronald, "Digital Trust in the Cloud" August 2009. www.csc.com/security/insights/32270-digital_trust_in_the_cloud

[5] Knode, Ronald with Egan, Douglas, "Digital Trust in the Cloud: Into the Cloud with CTP – A Precis for the CloudTrust Protocol, V2.0" ,July 2010, http://www.csc.com/cloud/insights/57785-into_the_cloud_with_ctp

[6] Algamdi, A., Coenen F. and Lisitsa, A., "A trust evaluation method based on the distributed Cloud Trust Protocol (CTP) and opinion sharing", International Conference on Computer Applications & Technology (ICCAT'17), 2017.

[7] F. Doelitzscher, C. Reich, M. Knahl, A. Passfall, and N. Clarke, "An agent based business aware incident detection system for cloud environments," Journal of Cloud Computing: Advances, Systems and Applications 2012.

[8] A. Sumetanupap and T. Senivongse, "Enhancing Service Selection with a Provider Trustworthiness Model", Eighth International Joint Conference on Computer Science and Software Engineering (JCSSE), 20ll, P(281-286).

[9] S. Habib, S. Ries, M. Muhlhauser and P. Varikkattu, "Towards a Trust Management System for Cloud Computing Marketplaces: using CAIQ as a trust information source", Security and Communication Networks, 2014.

[10] CSA. "Consensus assessments initiative" https://cloudsecurityalliance.org/research/initiatives/consensus-assessments-initiative/

[11] T. Martin, "Modèle d'évaluation de sécurité cloud", Smals Research. https://www.smalsresearch.be/tools/cloud-security-model-fr/

[12] A. Jøsang and D. McAnally, "Multiplication and comultiplication of beliefs", International Journal of Approximate Reasoning 38 : P(19—51), 2005.

[13] A. Jøsang, "Book : Subjective Logic", Universitas Osloensis, 2015

[14] D. Ceolin, A. Nottamkandath, and W. Fokkink, "Subjective Logic Extensions for the Semantic Web