International Journal on Cloud Computing: Services and Architecture (IJCCSA)
Vol. 12, No. 1, February 2022

SELECTING TRUSTWORTHY CLIENTS IN THE CLOUD

Imen Bouabdallah[1] and Hakima Mellah[2]

[1]Department of Computer Science, USTHB, Bab ezzouar, Algeria
[2]Information and multimedia system Department, CERIST, Ben Aknoun, Algeria

ABSTRACT

With the recent increase demand for cloud services, handling clients' needs is getting increasingly challenging. Responding to all requesting clients with no exception could lead to security breaches, and since it is the provider's responsibility to secure not only the offered cloud services but also the data, it is important to ensure clients reliability. Although filtering clients in the cloud is not so common, it is required to assure cloud safety.

In this paper, we made use of multi agent system interaction mechanisms to handle the cloud interactions for the providers' side, and of Particle Swarm Optimization to select the and determine the accurate trust weight and therefore to filtrate the clients by determining malicious and untrustworthy clients. The selection depends on previous knowledge and overall rating of trusted peers. The conducted experiments proves that the combination between MAS, PSO and trust outputs relevant results with different number of peers, the solution is able to converge to the best result after small number of iterations. To explore more space and scenarios, synthetic data was generated to improve the model's precision .

KEYWORDS

Cloud, Multi agent system, trust, interaction, Particle Swarm Optimization.

1. INTRODUCTION

Cloud computing is a distributed network that guarantees access to a large amount of data and IT infrastructure in an open environment at several levels (software, hardware...). The use of cloud computing has been highly increased since the start of the Covid-19 pandemic and the augmentation of remote work [1] [2], leading to an upsurge in number of both cloud clients and cloud providers.

The high number of users in the cloud can cause the emergence of malicious entities. Identifying malicious users between the large numbers all at once would be difficult due to the insufficient amount of information regarding the users' behavior. The non-detection of such entities can produce security problems and restrictions in cloud use [3].

To overcome the security flaws in the cloud multiple paradigms have been introduced to improve the protection levels or to reinforce security. Multi agent system (MAS), one of the many paradigms, is a distributed structure [4] that has been employed to solve multiple complex problems such us detecting security flaws in multiple fields, as in the cloud.

MAS is made up of multiple autonomous agents existing in a shared environment. These agents are capable of interacting and exchanging data and resources. They are distributed all over the

system's network and capable of remaining connected through their interactions mechanism. Their intelligence allows them to autonomously take actions in order to solve their assigned tasks and evolve in the system.

Since distribution is common in Multi Agent System (MAS) and the cloud, and both paradigms consist of interconnected agents (in MAS) or users (in the cloud), combining them would draw the strong aspects of both.

Yet, because the cloud is a large and open environment [5] (multiple entities exist and can leave or enter the network randomly) and diversity of users, the combination still presents some flaws that were the reason for the lack of early research in this area.

Because of MAS's strong features to solve complex task, and to respond and overcome the cloud's flaws it has gain popularity in the field. Multi Agent System features include autonomy, perception and awareness, reactivity, interaction, cooperation [5], and many more depending on the implementation.

This work is a prelude for an adaptive interaction framework, the proposed model uses interaction along with cooperation to share knowledge among agents to determine trustworthiness of cloud consumers.

In this paper, Multi Agent System interaction mechanisms are used along with Particle Swarm Optimization (PSO) in the cloud to determine the trustworthy clients and eliminate the malicious ones. The following section discusses some of the recent related works, section 3 presents a background to: define MAS and the cloud, introduce trust in the cloud and present the main actors of the system; section 4 discusses the proposed PSO-Trust model and section 6 presents the evaluation and experimentation results.

2. RELATED WORKS

In the following we'll expose a selective overview of the previous related works, regarding the use of Multi Agent Systems in the cloud and the implementation of trust for security.

2.1. Multi Agent System Across the Cloud

MASs have successfully solved many problems in the cloud, and it is even more promising when combined with other promising techniques. E.g., enhancing security by detecting intrusions [6], decision-making, service discovery [7], service reservation [8] …

Multi Agent System is implemented in the cloud to project the actors in the cloud as agents of the system, the agents would be responsible for executing the commands of the actors and negotiating instead of them.

Service discovery was studied in [9]; the authors used a hybridization of MAS and web services. They presented sets of agents that act on behalf of cloud consumers and cloud providers to generate and agree to the contract, the service discovery agent acts on behalf of the clients using semantic ontology to select the best-fitted service provider.

Despite the diverse works joining MAS with the cloud and the various features of MAS, the majority of works focus on the concept of agents' autonomy, but still neglect a critical aspect that can generate effective results: cooperation.

In this work, cooperation between agents is considered through knowledge sharing among acquaintance agents to provide more accurate data to the system. We make use of the interaction mechanisms of MAS to exchange relevant and up to date data.

2.2. Exploring Trust in the Cloud

Trust management in the cloud has been increasingly studied to improve both security and intrusion detection, and is employed for service discovery and recommendation. In the cloud, privacy is a major concern for all users and providers. Most of the works considering trust in the cloud, employ it on the client side ([10] [11] [12]) to help in choosing a service provider.

Using users' feedback and behaviour, authors in [11] proposed an Evidence Trust model to select the trusted Cloud Service Provider (CSP). The final assigned trust value is computed through cumulating feedback and behavioural trust and would determine the trustworthiness of a CSP. To detect false feedback, they compared the previously submitted feedback from user with the current one. In this case, if the client is always giving a false feedback it will pass undetected.

Li, et al in [13] used trust along reputation in an Internet Of Things (IOT) cloud-based to determine the trustworthiness of a cloud service, by using feedback rating of costumers and evaluation of security metrics.

Despite the multiple studies around the concept of trust in the clous, the works on guaranteeing only trusted clients in the network are very insignificant and the literature is hardly expressive; but since it's the provider's responsibility to secure the cloud, it is highly important to check clients' background before granting the service and experiencing the security problems (identity theft, data breach…), which this paper discusses.

For the reason that if a provider in the cloud is under attack or experiencing security flaws, that would certainly influence clients' data and moreover provider's reputation, which can hardly be changed resulting in losing its users.

Mehraj and Banday discus in [14] an application of trust in cloud deployment using Zero-Trust engine where, instead of trust being handled by a role agent, they use the engine to determine trustworthy entities. As for end users, they rely on authentication to validate users' identity and therefore trustworthiness, which may not always be the source of user's credibility.

A dynamic multi-dimensional trust evaluation method was proposed in [15] to determine trustworthiness of cloud providers and cloud consumers. In this research, the trustworthiness is considered to define the honest behaviour of the consumer, and since it does not interact only with providers but also with: brokers, auditors and SLA agents, compliance information is collected from all entities to calculate trust.

2.3. Implementing Trust using PSO

Bio-inspired algorithms follow real life principals and are evolutionary aiming to improve the quality of problem solving. They derive from the behaviour of large groups but still can be applied to small data and provide precise and accurate data [16].

Particle swarm optimization is based on social behaviour [17] and thus can make use of shared knowledge between entities of the system. It generates precise results while requiring only small sized data (compared to other algorithms).

[18] discuss introducing trust for group decision making. The individuals in a group can be influenced by others to change their opinions and decisions. In the proposed model, individuals' opinions (or expert evaluations as illustrated in the paper example [18]) represent the particles of PSO that are employed to determine their optimum. Authors argue, by comparing the proposed model to another model without trust implementation, that the drawbacks can be avoided using trust and making use of group intelligence.

In [19] PSO was introduced to optimize Neural Network (NN) parameters and use NN to determine costumers trust rate in the cloud, Although the experiments showed that the results were promising, the NN require large data and need training over multiple times to really outputs precise data. Unlike that, PSO-Trust model presented in this paper requires only small swarm population to converge to the best solution.

3. BACKGROUND

Multi agent systems and the cloud are the main technologies involved in this paper, the following section presents a brief review of these paradigms.

3.1. Multi Agent System

Multi agent system (MAS) [20] consists of multiple interacting intelligent agents that are capable of handling tasks and cooperating to solve complex problems. Each agent receives a different task and outputs a different action, that would eventually be collected to solve a complex problem representing the global goal of the system.

Intelligent Agents are autonomous and situated in an environment [21], they are capable of perceiving their surroundings, making and choosing their own decisions, and taking actions without assistance.

Agents are endowed of multiple features: autonomy, intelligence, interactivity, interaction… [21][5]. The latter is a mean of communication and cooperation; by interacting, agents would share knowledge or resources and would cooperate to accomplish complex tasks. Interaction's type changes with the goals and tasks. Negotiation, a type of interaction, is usually set up to settle over an agreement or to reach a compromise to outcome conflicts. Agents' autonomy drew much attention to them in research area, and are therefore applied in various domain.

3.2. The Cloud

Cloud computing [22] is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources. The users in the cloud are distributed over the network yet interconnected and able to access the needed resources at any time remotely.

These users can be classified, NIST [22] presents an architecture of five main roles in the cloud. In our opinion, and while implementing MAS along the cloud, the roles could be brought to two main classes: users and helper agents. In this work, we only consider users; however, the presence or absence of helper agents would not affect the model.

As for users, two actors are identified: cloud clients (consumers): the users asking for services, and cloud service providers: the actors offering the services at a given price with some defined criteria.

Many security flaws are present at this level effecting clients' privacy and data, and providers database and reputation [3]. To encounter some of the difficulties, multi agent systems (MAS) are introduced to make use of agents' intelligence for handling complex tasks. Agents would negotiate for the provider and study the opponent before and during the iteration.

Interactions and data exchange would affect security and therefore user satisfaction (which can be gained by ensuring the protection of client's privacy). Hence, we introduced trust as a security weight.

3.2.1. Trust Definition in the Cloud

In literature, trust is defined as the firm belief in the reliability of the received information or also as "the vesting of confidence in persons or abstract systems, made on the basis of a leap of faith which brackets ignorance or lack of information" [23]. Between agent, trust could be defined based on multiple criteria that differentiate based upon the context which might lead to having different definition; in some works, it is based on reputation from other agents [24], reliability of the agent, respect of the given information, insurance of privacy…

Trust and security are constantly associated, mostly because trust engender a feeling of safety that is earned, acquired or build on strong knowledge [25]. Trusting an entity implies asserting the credibility of information it shares and the identity it presents and thus guarantying its access to data or platforms, which is major security concern.

3.2.2. Actors in the Cloud-Based MAS

In order to clarify the subsequent sections, we start here by defining the main actors of the cloud-based system.

The cloud is a meeting environment for providers and clients (Figure 1), where the providers try to satisfy the clients need and expectations for services' quality.

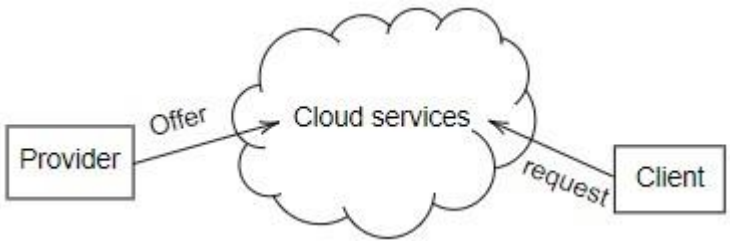

Figure 1 – Main actors in the cloud

The clients (consumers) in the cloud would ask for services (software, database, collaborative environment…), and the providers would be responsible for delivering the best services at the best desirable cost and quality:

1. A cloud provider: delivers cloud computing services and solutions, responsible for making a service available for consumers [22],
2. A cloud consumer: could be individual or organization, using cloud services and resources, and maintain a business relationship with the provider. The consumer itself could represent a cloud provider.

In a cloud-based MAS, agents take actions for the cloud actors; client are represented by a client agent and providers by service agents (to handle the large number of requests, multiple service agents are assigned to a single cloud provider (Figure 2)):

- Client Agent (CA): carries cloud consumer request and handle the search and negotiation until service delivery,
- Service Agent (SA): represents the cloud service provider, it handles the consumers requests, it can also act as a cloud consumer to ask for complimentary services from other providers.

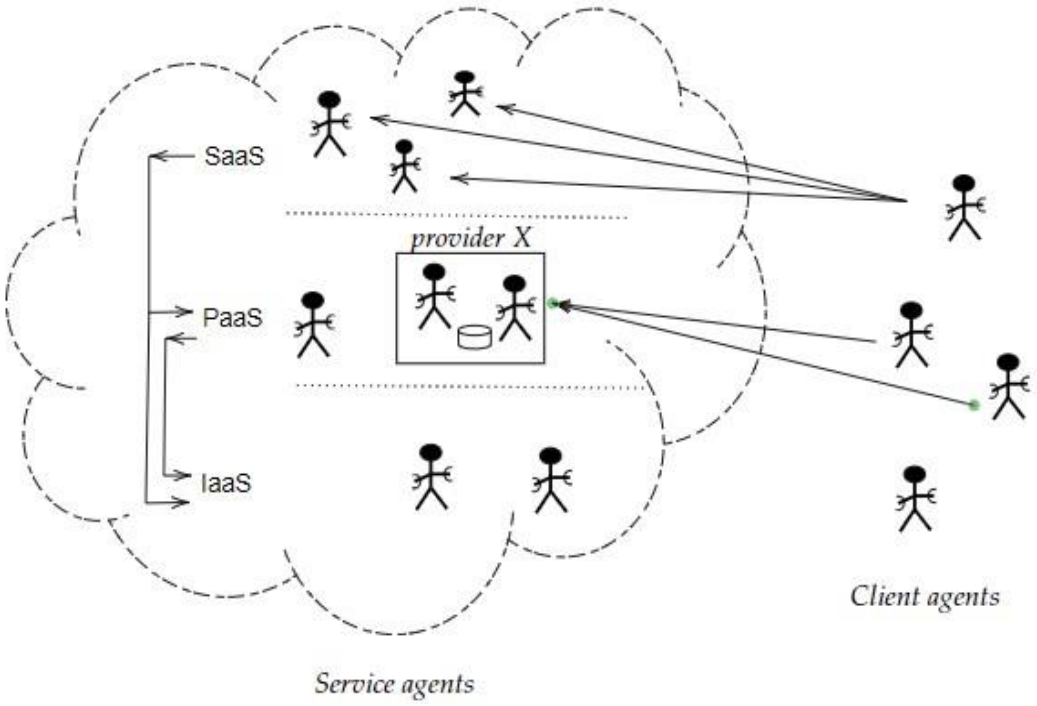

Figure 2 - MAS cloud agents

Agents conduct interactions instead of cloud actors to save time and effort because of agent's autonomy they are able to complete their assigned tasks without assistance.

Trust is defined for a service agent (SA: agent that guarantees services access) of a client if it: respects the agreements, does not violate the cloud resources term of use and privacy, does not barging for so long that it causes famine (if the system does not have a timeout limit). On the other hand, a client could be less trusted if it has a bad reputation regarding the previous discussed points.

4. PSO-TRUST MODEL

In the following section, we present the proposed model PSO-Trust starting by defining PSO and presenting the model's flowchart.

4.1. Particle Swarm Optimization

Particle Swarm Optimization (PSO) [16] is a search algorithm [26], inspired by the social behaviour of bird flocks. Each individual in PSO represents a potential solution and is called a particle. The whole population is referred to as a swarm.

PSO is an Artificial Intelligence (AI) technique that aims to find the approximate solution (global best) to maximization or minimization problems by running the algorithm multiple times until the whole population converges or the maximum number of iterations is exceeded.

Each particle possesses a position, and a velocity that represents the speed and direction by which the particle moves at each iteration.

At each iteration, velocity and position are updated through the following equations:

$$v(t+1) = v(t) + c_1 r_1(t)\left(p^{best}(t) - x(t)\right) + c_2 r_2(t)\left(g^{best}(t) - x(t)\right) \quad (1)$$

$$x(t+1) = x(t) + v(t+1) \quad (2)$$

Where:

- t : time/iteration
- v(t) : velocity at time stamp t
- x(t) : position
- c_1 ,c_2 : constants for nostalgia and envy (resp.), also called trust parameters
- r_1,r_2 : random vectors $\in$[0,1]
- p^{best}, g^{best}: personnel and global best (resp.)

When a particle moves to a new position, it is affected by three components: previous position, social component and cognitive component.

The whole population's (swarm) size effects the convergence of the algorithm for: the smaller the size the slower to converge, and the bigger the size the more space to explore.

4.2. Model Flowchart

The following model (Figure 3) casts a presentation of the proposed PSO-Trust model.

Upon receiving a service request, and instead of start negotiation on the spot, the SA would first start by checking the client's trustworthiness by either retrieving its trust weight from the database (if the agent has had previous interactions with the client), or acquire it from its acquaintances (Figure 3).

If none of the acquaintances possess information about the client, its trust weight is set to zero (as a neutral value). Otherwise, the PSO model is used to determine clients' trust weight, and is judged upon the resulting value whether to be trusted or not.

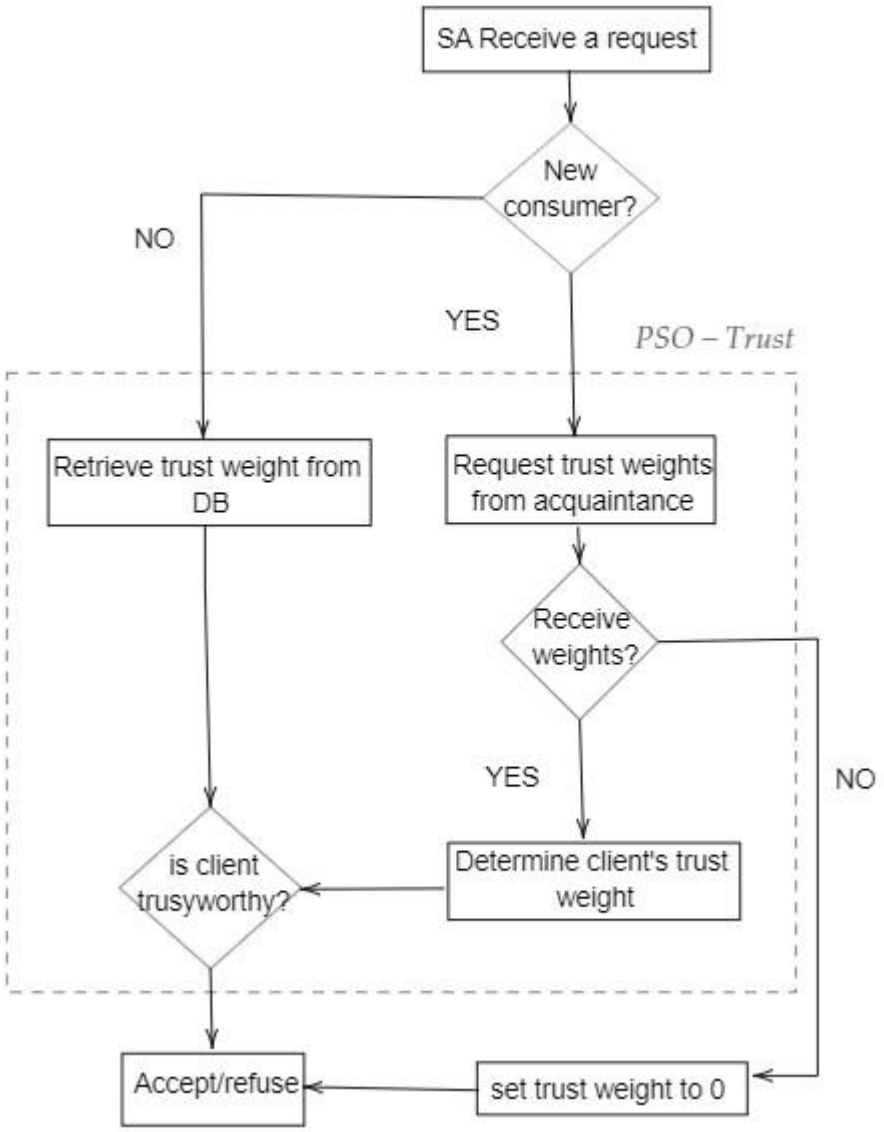

Figure 3 - Steps to evaluating client's trustworthiness

After setting a trust value for the client an either accepting or declining to interact with the client, the trust value is stored in the database for next usages (either by the provider or its acquaintances).

Trust values fall between [-1, 1] instead of [0, 1] since clients could be untrustworthy. (-1) denotes an untrustworthy agent and (1) denotes a trusty one. The median value (zero) is mostly used for agents with no history of interaction whatsoever.

4.3. Model Presentation

In the cloud-based MAS, service agents (SA) are responsible for negotiating with the client (CA) and offering the needed services. When a SA is contacted by a new client C_x, to ensure security and assign a trust weight for the new costumer, the SA would inquire its acquaintances to send their trust weights of the client C_x.

If present, the agent would receive several different weights and it would be hard to pick one from the large choice.

To overcome that, Particle Swarm Optimization is used to select and find the most appropriate weight solution in a 2D plan, where we consider particles as a tuple ($T_{a(x)}$, $R_{TA(x)}$) where:

- $T_{A(x)}$: The trust weight of the agent A(x),
- and $R_{TA(x)}$: The trust value received from agent A(x).

The fitness function for each particle is determined using Euclidian distance to calculate the distance between each particle and the global best so far.

$$Fintess\ function = \ e = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (3)$$

Where: (x_1, y_1) are the particle's position and (x_2, y_2) the global best position. The personnel best of the particles is updated if the error is minimal.

The algorithm outputs the global best of the population by generating the mean of all particles.

$$Global\ Best = \sum_{1}^{size} \frac{ParcticlePosition(i)}{size} \quad (4)$$

But as such Equation (4) will be assigning the same importance to all trust weight values, nevertheless the ones coming from credible acquaintances should have more impact in the equation. In order to respond to that the equation is changed so that the particle position is multiplied by a variant as an impact factor (Θ) depending on the trust values:

$$Global\ Best = \frac{1}{size} \sum_{1}^{size} \Theta * ParcticlePosition(i) \quad (5)$$

Since the population positions are between [-1,1] (to correspond to the trust weights), and so that the generated solution can be used immediately, the results of equation (5) are normalized in $[-1, 1]$.

$$Global\ Best' = 2 * \frac{GlobalBest - \min x}{\max x - \min x} - 1 \quad (6)$$

With: x = Particle Position.

5. EVALUATION

Before preforming the experiments, and to have a realistic view of the proposed model in a cloud environment we started by implementing the framework in Netlogo [27] simulation software.

Netlogo is a MAS modelling environment for simulating social and complex phenomenal in a world allowing the user to observe its state through time. It is mostly used by research to project the ideas into an interactive environment.

Because the cloud is open, we configure the number of agents in the population as variable, and could be increased or decreased by the user (increases even up to 150).

To get a clear view of the simulated world we chose a small population number for the

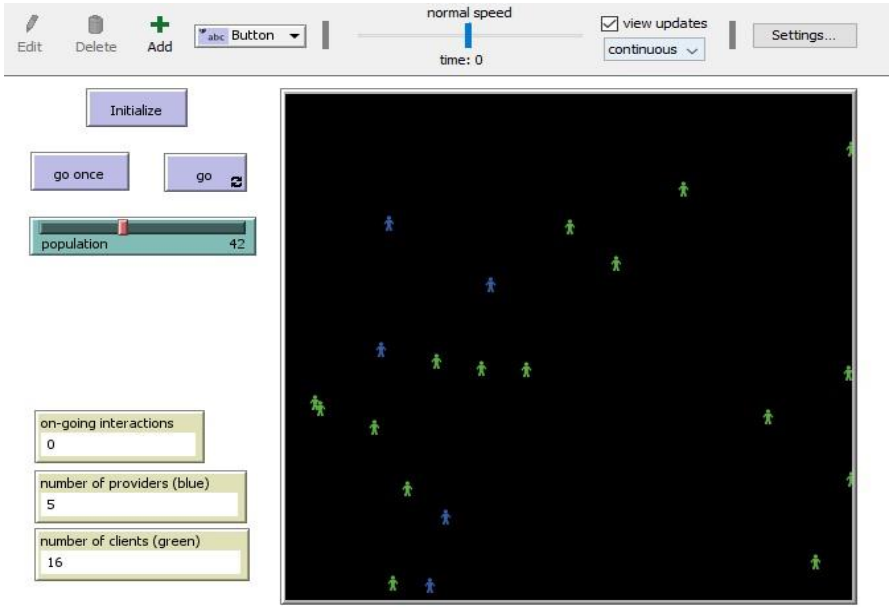

Figure 4 - The Netlogo world after initialization illustrated experiment.

At initialization (Figure 4), a random number of clients (in green) and providers (in blue) are present in the network. The “go” function allow the system to simulate the cloud environment interactions.

After running the program for some time (Figure 5), it can be noticed how some providers express the need to contact their acquaintances after receiving requests from consumers, while

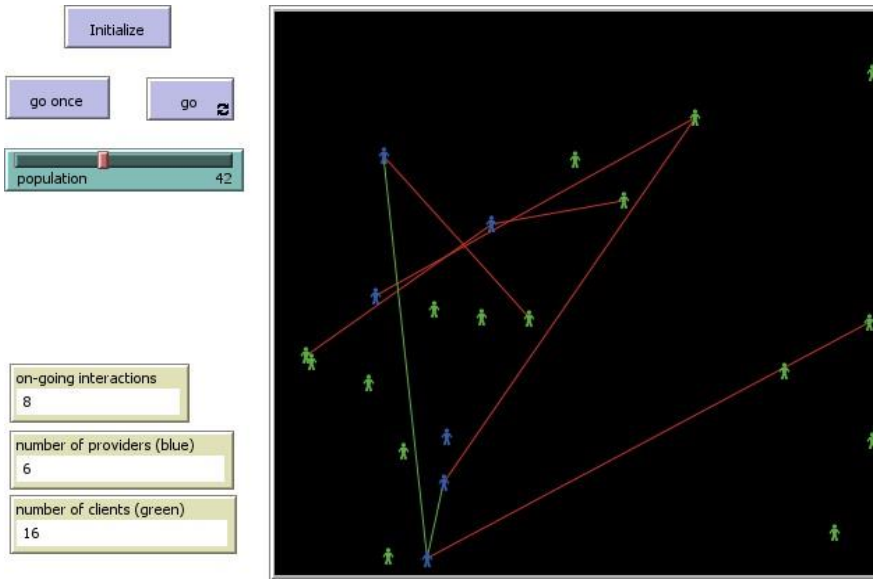

Figure 5 - Ongoing interactions in the cloud

others do not. This is due to the presence or absence of data of the contacting clients in the providers' knowledge database (if the provider holds already previous information of the client, it could not contact its acquaintances and proceed with the data it holds).

The number of agents in the cloud can change during runtime due to the openness of the cloud (agents can enter or leave the environment randomly), but just as the maximum number of agents in the population is reached no more agent can enter until others leave (since the population number is previously set by the user).

To experiment our work and evaluate the performance of the PSO-Trust model, we develop a framework using Python and deployed it in Jupiter notebook under Google's cloud platform.
Python is an interpreted programming language, offering a huge number of libraries, frameworks, and even plotting library (matplotlib) to ease creating animated and interactive visualisations. In the training, and due to the lack of clients' data (for the purpose of confidentiality), we used a random dataset to evaluate the model.

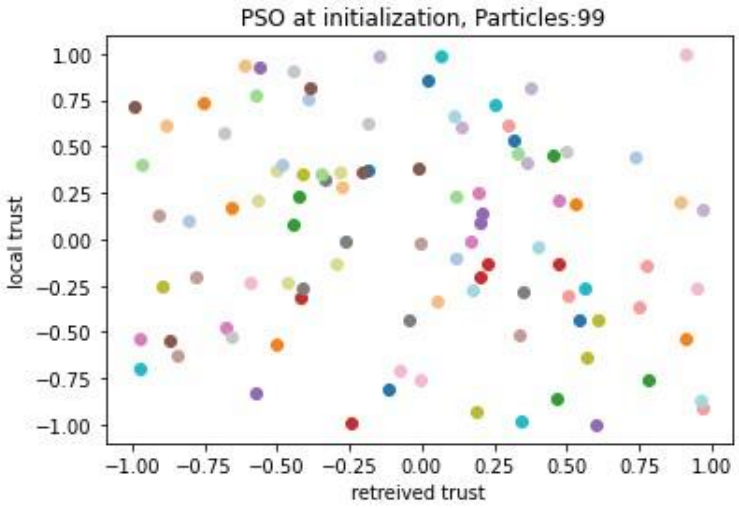

Figure 6 - Particles' distribution with large population

The population size may differentiate depending of the received trust weights and the number of agent's acquaintance. To test the algorithm efficiency, we used different population sizes in the experiments (between 5 and 100).

To illustrate experiment's results, we start by exploiting large population; the initial high sized population present in Figure 6 (where each particle is represented by a different colour each to ease observation) is able converges after a minimal number of iterations

Before conducting the experiments, we stated by configuring the parameters for the model's equations. c_1 and c_2 represent (resp.) how much confidence the particle has in itself (the local position), and how much confidence it has in its neighbourhood. Hence, since we need the particles to be more attracted to the global than personal best so that cooperation can emerge and so that the model converges fast, envy value is set slightly bigger than nostalgia ($c2 >> c1$).
The algorithm runs for a several numbers of iterations until the global best is reached or the error rate is minimum.

In Figure 7, after several iteration (five to be precise) particles have almost fully converge to the global best (represented by the black square).

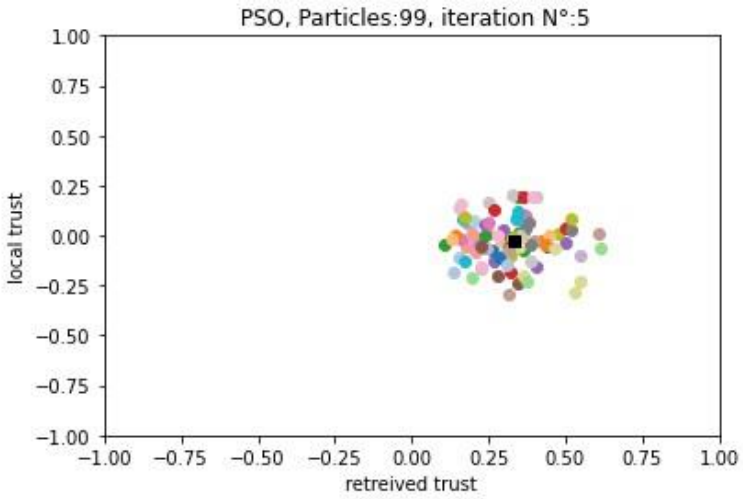

Figure 7 - Convergence after five iterations.

Experiments also showed that even with a small swarm size (Figure 8). The algorithm also succeeded with the small swarm size in reaching the best trust weight value in Figure 9 (the black triangle in figure 9 represents the global best position).

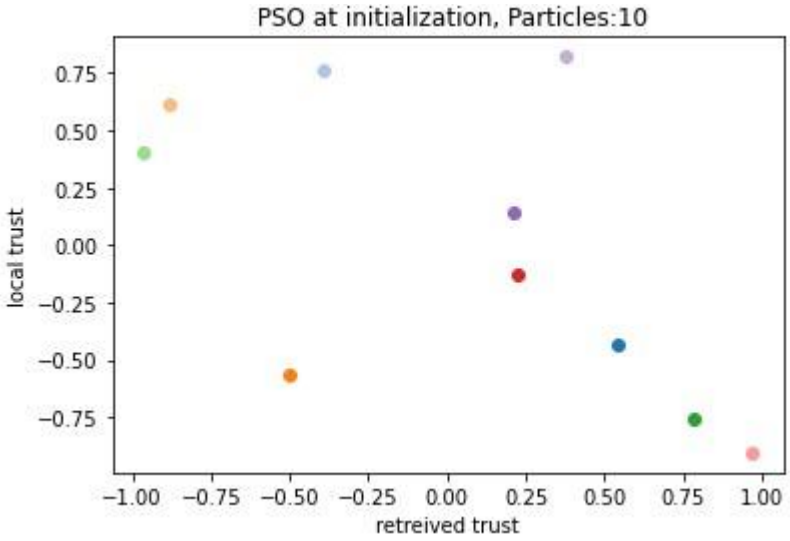

Figure 8 - Small swarm size.

Therefore, even if the acquaintance offering information are numbered, the algorithm can still provide an accurate solution.

To explore more space, we used synthetic data [28] a fake data that mimics real data by generating large data from small scale base. The generated data can simulate complex and not yet encountered conditions which allows us to enforce the model's accuracy.

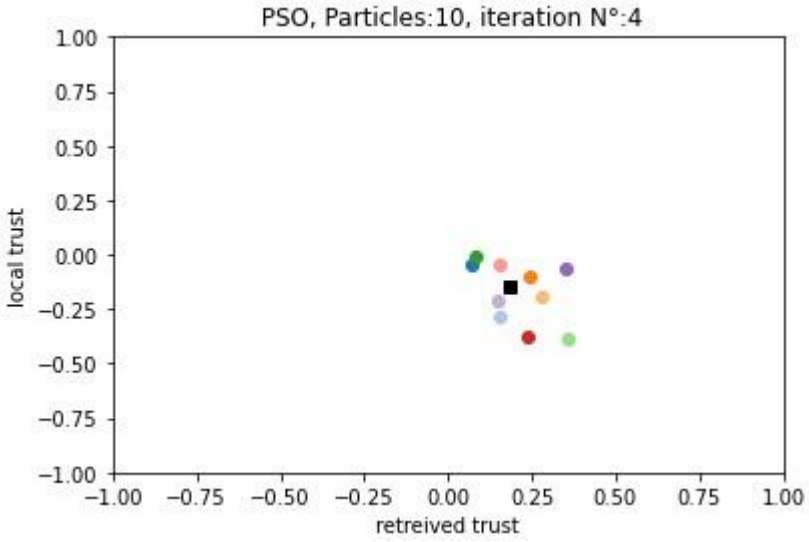

Figure 9 - Swarm population near convergence

In some cases, synthetic data can be more effective to enhance model training because it exposes it to random scenarios and it can be either fully (when data is missing) or partially synthetic. Also, it could be adjusted to fit the needs.

To generate synthetic data, Python offers a large choice of libraries and packages for generating this type of data such as: Pydbgen [29], Scikit-Learn [30], Faker, etc, or Frameworks like PyTorch using Generative Adversarial Networks (GAN) and implementing Neural Networks to train the model and generate new samples of data (Figure 10).

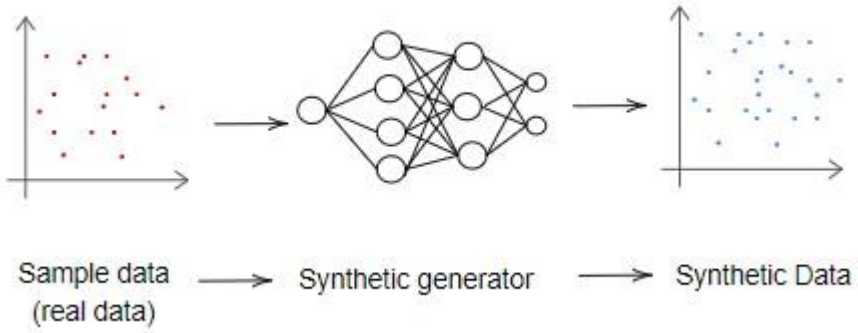

Figure 10 - Synthetic data generation

After many executions we noticed that, compared to large data, when handling small population the model takes longer to converge; in order to overcome that and achieve better results we introduced error rate as a stop criterion to allow the model to stop only after global optimum is reached and avoid early convergence.

To tune this constraint and control the learning process, we run the program multiple times with different values to compare the outcomes. As a result, after considering the changes of the error value throughout the experiments while changing the population size and to fit the model we made it variable so that it changes with respect to the size.

Because providers refuse to share their data (due to security concerns), we could not find suitable data as to compare our proposed solution with other models and the generated data would not fit with the model's description.

6. CONCLUSIONS AND FUTURE WORKS

Since it is the providers' responsibility to keep the data and clients secure, trust in the cloud should not be only considered by clients. Malicious users can attack the cloud and effect the data it stores, therefore, effect many clients and risk their privacy.

To secure the network and identify the clients that are worthy of trust, we proposed a PSO-Trust model where we make use of the interaction mechanisms of MAS to share reliable data. Making use of synthetic data will allow the model to get more accurate and precise results.

One of the difficulties we faced is the lack of data, for this we generated a database through deep learning from the small sized data to represent feedback from providers and we aim next to generate clients' history. For the next step, we would extend our work to construct an opponent interaction model. The current work would represent a pre-interaction phase to the interaction model and adapt the ongoing interactions based on the previous shared knowledge.

REFERENCES

[1] "En pleine pandémie, Google enregistre des profits records grâce à la publicité," [Online]. Available: https://www.france24.com/fr/amériques/20210428-en-pleine-pandémie-googleenregistre-des-profits-records-grâce-à-la-publicité. [Accessed 02 06 2021].

[2] "Wall Street : l'heure de la pause malgré Amazon," 2021. [Online]. Available: https://www.boursedirect.fr/fr/actualites/categorie/marche-us/wall-street-l-heure-de-la-pausemalgre-amazon-boursier. [Accessed 2 June 2021].

[3] Subramanian, Nalini & Jeyaraj, Andrews. (2018). Recent security challenges in cloud computing. Computers & Electrical Engineering. 71. 28-42. 0.1016/j.compeleceng.2018.06.006.

[4] C. Psaltis, "The True Meaning of 'Open Cloud' ", 2021 [Online]. Available: https://thenewstack.io/the-true-meaning-of-open-cloud/

[5] Ferber, Jacques. (2001). Multi-Agent System: An Introduction to Distributed Artificial Intelligence. J. Artificial Societies and Social Simulation. 4.

[6] O. Achbarou, M. Kiram, S. Elbouanani and O. Bourkoukou, "A New Distributed Intrusion Detection System Based on Multi-Agent System for Cloud Environment," International Journal of Communication Networks and Information Security, vol. 10, pp. 526 - 533, 2018.

[7] K. M. Sim, "Agent-Based Cloud Computing," in IEEE Transactions on Services Computing, vol. 5, no. 4, pp. 564-577, Fourth Quarter 2012, doi: 10.1109/TSC.2011.52.

[8] S. Son and K. Sim, "A Price- and-Time-Slot-Negotiation Mechanism for Cloud Service Reservations," IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics: a publication of the IEEE Systems, Man, and Cybernetics Society, vol. 42, pp. 13-28, 2011.

[9] Parhi, Manoranjan & Pattanayak, Binod & Patra, Manas. (2015). A Multi-agent-Based Framework for Cloud Service Description and Discovery Using Ontology. 10.1007/978-81-3222012-1_35..

[10] A. M. Mohammed, E. I. Morsy and F. A. Omara, "Trust model for cloud service consumers," 2018 International Conference on Innovative Trends in Computer Engineering (ITCE), 2018, pp. 122-129, doi: 10.1109/ITCE.2018.8316610.

[11] Mujawar, Tabassum & Bhajantri, Lokesh. (2020). Behavior and Feedback based Trust Computation in Cloud Environment. Journal of King Saud University - Computer and Information Sciences. 10.1016/j.jksuci.2020.12.003.

[12] C. Mao, R. Lin, C. Xu and Q. He, "Towards a Trust Prediction Framework for Cloud Services Based on PSO-Driven Neural Network," IEEE Access, pp. 2187-2199, 2017.

[13] X. Li, Q. Wang, X. Lan, X. Chen, N. Zhang and D. Chen, "Enhancing Cloud-Based IoT Security

Through Trustworthy Cloud Service: An Integration of Security and Reputation Approach," IEEE Access, vol. 7, pp. 9368-9383, 2019.
[14] S. Mehraj and M. T. Banday, "Establishing a Zero Trust Strategy in Cloud Computing Environment," 2020 International Conference on Computer Communication and Informatics (ICCCI), 2020, pp. 1-6, doi: 10.1109/ICCCI48352.2020.9104214.
[15] Challagidad, Praveen & Birje, Mahantesh. (2020). Multi-dimensional Dynamic Trust Evaluation Scheme for Cloud Environment. Computers & Security. 91. 101722. 10.1016/j.cose.2020.101722.
[16] Mukhopadhyay, Mayukh. (2014). A brief survey on bio inspired optimization algorithms for molecular docking. International Journal of Advances in Engineering & Technology. 7. 868-878. 10.7323/ijaet/v7_iss3.
[17] A. Engelbrecht, "Particle Swarm Optimization," in Computational Intelligence: An Introduction, John Wiley & Sons, Ltd, 2007.
[18] Zhou, Xiaoyang & Ji, Feipeng & Wang, Liqin & Ma, Yanfang & Fujita, Hamido. (2020). Particle swarm optimization for trust relationship based social network group decision making under a probabilistic linguistic environment. Knowledge-Based Systems. 200. 105999. 10.1016/j.knosys.2020.105999.
[19] C. Mao, R. Lin, C. Xu and Q. He, "Towards a Trust Prediction Framework for Cloud Services Based on PSO-Driven Neural Network," IEEE Access, vol. 5, 2017.
[20] G. Weiss, Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence, Cambridge, MA, USA, 1999.
[21] M. Wooldridge and N. R. Jennings, "Intelligent agents: theory and practice," The Knowledge Engineering Review, vol. 10, no. 2, p. 115–152, 1995.
[22] P. M. Mell and T. Grance, "SP 800-145. The NIST Definition of Cloud Computing," National Institute of Standards & Technology, Gaithersburg, MD, USA, 2011.
[23] M. Guido, "TRUST: REASON, ROUTINE, REFLEXIVITY," in SCARR Conference on Risk & Rationalities, Queens' College Cambridge, 2007.
[24] J. Granatyr, V. Botelho, O. R. Lessing, E. E. Scalabrin, J.-P. Barthès and F. Enembreck, "Trust and Reputation Models for Multiagent Systems," ACM Comput. Surv., vol. 48, no. 2, 2015.
[25] Lamsal, Pradip. (2002). Understanding Trust and Security.
[26] K. . J. and . E. R., "Particle swarm optimization," in Proceedings of ICNN'95 - International Conference on Neural Networks, 1995.
[27] U. Wilensky, "NetLogo," Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL, 1999.
[28] N. Laskowski, 'What is synthetic data? - Definition from WhatIs.com', *SearchCIO*, Feb. 2018. https://searchcio.techtarget.com/definition/synthetic-data (accessed Nov. 10, 2021).
[29] S. Tirthajyoti, 'pydbgen: Random dataframe and database table generator', *Github*. https://github.com/tirthajyoti/pydbgen (accessed Jan. 11, 2022).
[30] F. Pedregosa *et al.*, 'Scikit-learn: Machine Learning in Python', *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.