A Trust Based Ranking Model for Cloud Service Providers in Cloud Computing

Alshaimaa M. Mohammed[1] and Fatma A. Omara[2]
[1] Computer Science & Math. Department, Faculty of Science, Suez Canal University, Egypt.
Shaimaa_mostafa@science.suez.edu.eg
[2] Computer Science Department, Faculty of Comp. and Information, Cairo University, Egypt
f.omara@fci-cu.edu.eg

Abstract. With the rapid growth of Cloud services, many Cloud Service Providers (CSPs) offer similar service functionalities. Hence, guaranteeing the available CSPs having a trust degree would increase the performance of the cloud environment. Therefore, selecting the trusted CSP whose services satisfy the Cloud Service Consumers' (CSC) requirements becomes a challenge. According to the work in this paper, a ranking model for CSPs has been introduced based on a combination of the trust degree for each CSP and the similarity degree between the CSPs' parameters and the CSCs' requested parameters. The proposed model consists of four phases; Filtrating, Trusting, Similarity, and Ranking. In the Filtrating phase, the existing CSPs in the system will be filtered based on their parameters. The CSPs trust values are calculated in the Trusting phase. Then, the similarity between CSCs' requirements and CSPs' services is calculated. Finally, the ranking of CSPs will be performed. To evaluate the performance of the proposed CSPs Ranking model, a comparative study has been done among the proposed CSP Ranking model and the most up to date four models using two QoS case studies and Armor data set. According to the comparative results, it is found that the proposed CSPs Ranking model outperforms the existing models with respect to execution time, time complexity and precision of the system.

Keywords: Cloud Service Provider, Cloud Service Consumers' Request, Ranking, Trust, Similarity, Fuzzy Controller, Dynamic Adaptive Particle Swarm Optimization.

1 Introduction

Nowadays, Cloud Computing is considered one of the most challenging emerging technologies [1] [2]. It provides a pool of IT computing services (i.e. CPU, Networks, Storage, and applications) that offers a range of dynamic, elastic, and on-demand services to the Consumer on the basis of usages under "pay-as-you-go" pricing model [3]. These opportunities offer many advantages such as reducing the cost of the resources' scalability, self-service, location independence and rapid deployment [4]. The main feature of the Cloud Computing is that self-service provisioning is provided, which allows the users to deploy their own sets of computing resources [5]. The core of providing the required services in the Cloud is on-demand manner, where the Cloud Service Consumers (CSCs) request specific services (e.g. computation, storage, memory, etc...) from the Cloud Service Provider (CSP). The CSP must provide the service(s) that satisfies the CSCs' requests in terms of its Quality of Service (QoS) [6]. On the other hand, there are many CSPs offer the same service(s) with different parameters [7]. Therefore, selecting the proper CSP to provide the requested service(s) has recently attracted considerable attention from the industry and academia and it is considered one of the most critical and strategic problems in the Cloud environments. From the CSCs' point of view, selecting the proper CSP is essential to assure future performance and maintain compliance with laws, policies, and rules [8].

By increasing the number of CSPs who provide similar services, ranking them becomes the most challengeable issue [9]. From the security point of view, the increasing

number of CSPs causes the cloud environment to be more competitive day by day. One of the most important factors of security in the cloud is the CSPs' trust parameter which plays a vital role to make the cloud business grows and the CSP can get more profit. To make cloud computing more attractive, CSPs' trust must be addressed where CSCs can interact with the CSPs and use their services safely [10], Therefore, to rank CSPs, the trust parameter of the CSPs must be taken into consideration as it plays a significant role for prioritizing CSPs. The CSP's trust is defined as the expectation that CSP can be relied on, predictable, and act fairly [11]. By making the CSP trustable, it helps the CSC to interact with proper CSP [12]. Therefore, most existing selection researches based on CSCs' requirements only [2], [13], [8], and other consider only the trust parameter of CSP [10]. According to the work in this paper, a CSPs Ranking model has been introduced to provide a list of the most ranking CSPs to the CSC. This model based on CSPs trust degree and the similarity of their resources with respect to the CSCs' requirements. On our point of view, the reliability of the CSPs ranking model would be increased by considering the trust degree of CSPs.

The proposed CSPs Ranking model consists of four phases; Filtrating, Trusting, Similarity, and Ranking. As the number of the available CSPs is increased, the available CSPs are filtrated in the Filtrating phase to prevent unwanted CSPs. In the Trusting phase, CSPs' trust degrees will be calculated for the accepted CSPs. In the similarity phase, the similarity value between the CSCs' requests and the parameters of the accepted CSPs will be calculated. In the Ranking phase, the CSPs ranking degree will be determined based on the CSPs' trust degrees and their similarity value.

The paper is organized as follows, related work of the CSPs ranking is introduced in Section 2. Section 3 is dedicated to illustrating the principles of the proposed CSPs' Ranking model. The performance evaluation of the proposed CSPs' Ranking model relative to the most up to date models (i.e., hypergraph computational (HGCM), Hypergraph-Binary Fruit Fly Optimization Algorithm (HBFFOA), security risk, and E-TOPSIS algorithms) is discussed in Section 4. Finally, conclusions and future work are presented in Section 5.

2 Related Work

In [14], a ranking model has been introduced to rank the CSPs based on Hypergraph Computational model (HGCM) using the Service Measurement Index (SMI) metrics of parameters [15]. The model represents the relations between the SMI metrics in the form of Hyper-edges, where the intersection is the interrelations between them. A minimum distance algorithm is used to arrange the attributes in a hierarchical order. Neighbourhood relations between the Hyper-edges have been established by fixing a threshold value using the minimum distance algorithm. The recursive using Helly property influences the order of CSPs to select the proper CSP. The time complexity of this model is determined as follows [14]:

$$\boldsymbol{O(n^2+NP)} \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots.. \boldsymbol{(1)}$$

Where n is the number of sub-attributes, N is the number of attributes, and P is the number of CSPs. The computation overhead is considered the main drawback of this model, especially, by increasing the number of CSPs.

A CSPs ranking model using checkpoint-based load balancing has been introduced in [16]. This model uses integration between checkpoint and load balancing algorithm

to increase the availability of the requests of consumers in the real-time scheduling environment [17]. Initially, the user will access the system based on the obtained ranks according to the services have been accessed earlier and the preferred value is determined by subtracting the service's ranking. Then, the priority degree for each CSP is determined by adding the preferred values for all services provided by the CSP. Finally, the CSPs are sorted based on their priority degrees in ascending order. According to this model, a simple technique is used to calculate CSPs ranks based on the opinion of the CSCs only.

A CSP ranking framework, called security risk, has been proposed to find out the secured CSPs [18]. According to this framework, a model based on a security risk assessment approach has been developed to determine the vulnerabilities and define the risks related to CSPs. The vulnerable CSPs are determined based on the stochastic process for the security risk measurement for each CSP. Then, it ranks these CSPs based on the Markov chain. After ranking the CSPs, the best CSP is selected based on the reliability and security parameters only.

An Extended TOPSIS (E-TOPSIS) model has been proposed for ranking CSPs [19]. This model is based on seven criteria; accountability, agility, assurance, financial, performance, security and privacy, and usability. It uses Minkowski distance ($\mathrm{d}(\mathrm{X},\ \mathrm{Y})$) to measure distances between the solutions using the following equation:

$$\mathrm{d}(\mathrm{X},\mathrm{Y}) = \left(\sum |X_i - Y_i|^p\right)^{1/p} \quad \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \ (\mathbf{2})$$

The model generates solutions by variation the value of p for a specific interval [l, h] and the step value of p which determines the number of iterations. For each iteration, it outputs a list of CSPs ranking. Finally, the most ranking arrangement is defined. By increasing the number of iterations, the execution time will be increased. This is considered the main drawback of this model

Recently, an approach based on the Hypergraph-Binary Fruit Fly Optimization (HBFFO) Algorithm for cloud service ranking has been presented [20]. This approach consists of three phases; Filtering, Selecting, and Ranking phases. The role of the Filtering phase is to filter CSPs based on the user's requirements using the Hyper-graph technique [14]. Then, the time-varying mapping function and Helly property have been used to identify the trustworthy CSPs in the Selecting Phase. In the Ranking phase, the HBFFO algorithm has been used to rank these CSPs based on their trustworthy, credibility and the user QoS requirements. This model provides a service selection model, in which the service selection middleware consists of many service repositories with one repository for each service type. According to the service type in the CSC's request, the CSC could forward directly to the repository of this service. The experimental analysis of this approach is done using a dataset contains two parameters only (Response Time, and Throughput). The complexity of this approach is defined using equation (3):

$$\ll \mathrm{O}(m^3);\ where\ m\ is\ the\ number\ of\ CSPs \ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \ (3)$$

According to the related work, it is found that some ranking models define CSPs ranking based on the CSCs' requests only. Other models define CSPs ranking based on their parameters only. Our proposed CSP Ranking model concerns both (i.e., CSCs requirements and CSPs parameters).

3 The proposed Cloud Service Provider (CSP) Ranking Model

The main function of the proposed Cloud Service Providers (CSP) Ranking model is that when a new request from a CSC is received, all CSPs in the cloud will be ranked based on their trust degrees and the similarity between the requests' parameters and the parameters of each CSP. Again here, the proposed CSP Ranking model consists of four phases; Filtrating, Trusting, Similarity, and Ranking.

According to our previous work, a CSP Trusting model has been developed based on SLA parameters (i.e., Reputation, Availability, Turnaround Time, data Integrity, authorization, Data Recovery, Reliability) and CSCs parameters which concern about CSPs' feedbacks such as Users' Access Frequency, and Service Success Rate (for more details see [21]). As the proposed CSP Ranking model consists of four phases; Filtrating, Trusting, Similarity, and Ranking. The main function of the Filtrating phase is to prevent unwanted CSPs to be passing to the Trusting phase. This phase has been developed using Fuzzy Controller System [22]. In the Trusting phase, the trust degree of the accepted CSPs has been determined using Dynamic Adaptive Particle Swarm Optimization (DAPSO) technique [23].

According to the work in this paper, a CSP Ranking model has been introduced and developed based on the previous developed CSP Trust model. The proposed CSP Ranking model consists of four phases; Filtrating, Trusting, Similarity, and Ranking. The framework of the proposed CSP Ranking model is illustrated in Fig. (1).

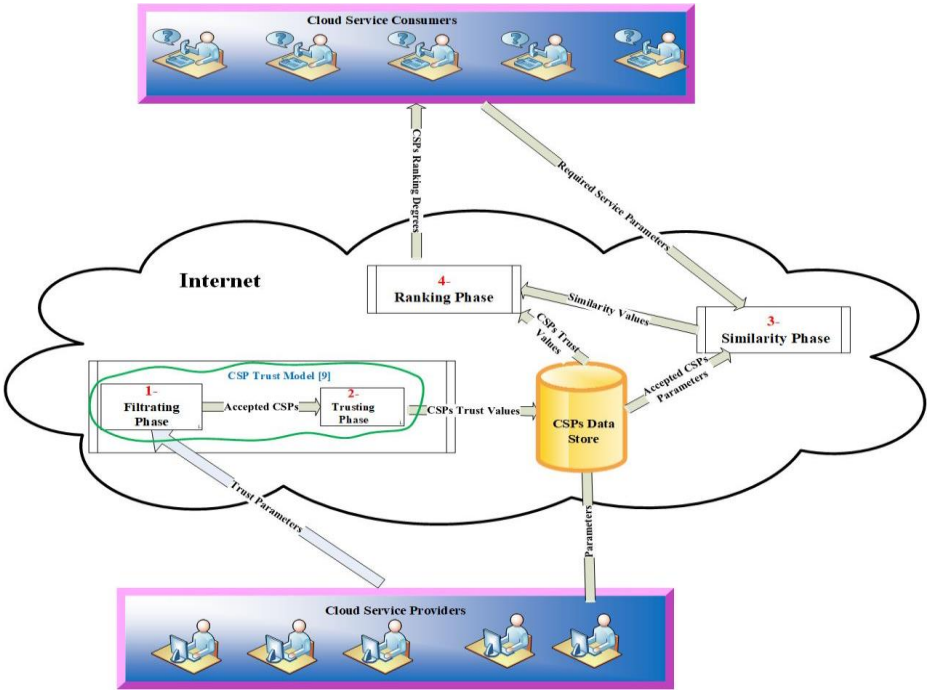

Fig. (1): The Proposed CSP Ranking Model

The Data flow of the proposed CSP Ranking model is illustrated in Fig. 2. According to Fig. 2, the data for all existing CSPs will be normalized and passed to the Filtrating phase to determine the accepted and unaccepted CSPs using Fuzzy Controller technique. The data of the accepted CSPs will be sent to the Trusting Phase to determine their trust degree using Dynamic Adaptive Particle Swarm Optimization (DAPSO) technique. When the CSC sends his request for a service, the data of the request will be normalized. The Similarity phase will define the similarity between the CSC's request and the data of the accepted CSPs using Cosine Similarity technique. Finally, the accepted CSPs will be ranked based on their trusting and similarity values.

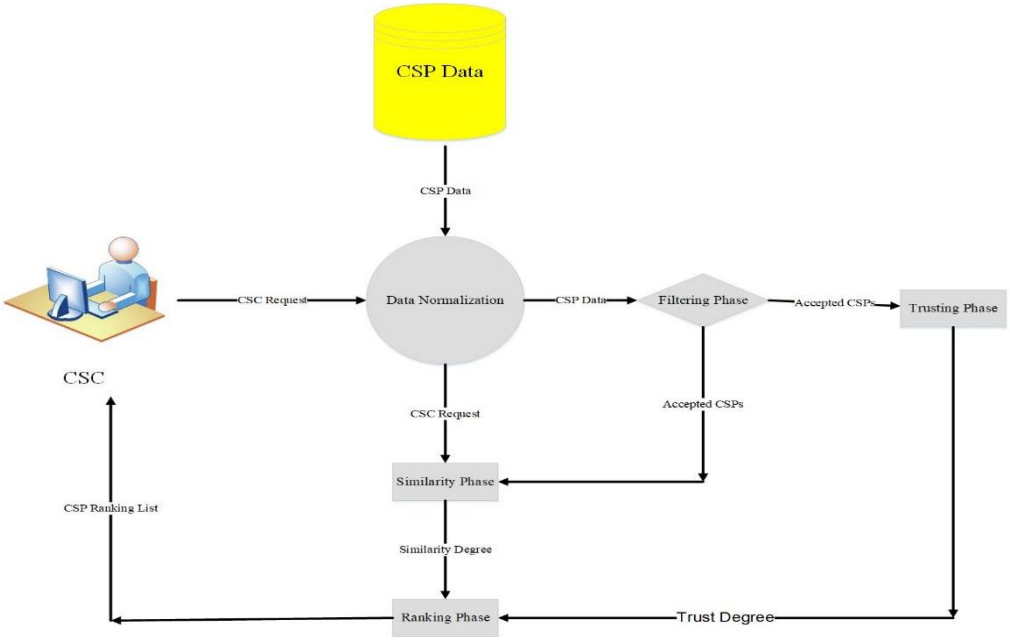

Fig. (2): Data Flow Diagram for CSP Ranking Model

3.1 Filtration phase

In this phase, a Fuzzy Controller System (FCS) is used to filter the available CSPs and define the CSPs to be considered in the trusting phase [24]. The decision makers use the FCS as an intelligent technique to support the decision-making processes [22] [25]. So, it is considered as a powerful and effective controller and predictive tool. In many clouds research work, the FCS is used as a predictive tool to predict the degree of the provider's security and trust based on IF-Then rules [26].

For the proposed model, the filtrating phase uses the FCS in to prevent unwanted providers to be involved in the Trusting, Similarity, and Ranking phases. So, the execution time is decreased, and the performance of the system is improved. Namely, for each provider, the FCS takes the values of the nine parameters (i.e., SLA and users' parameters) of this provider as input and produces an output value which indicates whether this provider will be involved in the trust metric stage or not. The output of the FCS is one of two parameters; GOOD, or POOR. The CSPs and the values of their nine

parameters could be represented as a matrix, A (n, m), where n is the number of the CSPs and m is the number of parameters to be concerned (i.e., the above nine parameters). Each column in matrix A represents the values of a specific parameter for all CSPs, and row represents the values of the nine parameters for each CSP. Therefore, each CSP with his jth parameter is presented as a (i,j), where i is the CSP identification (CSPi) and j is the value of the jth parameter:

$$A(n,m) = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{19} \\ a_{21} & a_{22} & \cdots & a_{29} \\ \vdots & \vdots & \cdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{n9} \end{bmatrix}$$

Therefore, an If-Then rule has been introduced to determine the FCS decision (i.e., GOOD or POOR). According to this rule, the decision for each CSP is determined as follows:

If-Then Rule: For each CSPi , i=1, 2… n, where n is the number of CSPs

$$\left.\begin{array}{l} \text{if } \sum_{j=1}^{9} F(P_j, OP_j) \text{ is GOOD, Then } CSP_i \text{is GOOD} \\ \text{Otherwise, } CSP_i \text{ is POOR} \end{array}\right\} \ldots\ldots\ldots\ldots\ldots (4)$$

Where $\mathbf{P_j}$ indicates the parameter j (j =1, 2… 9), and $\mathbf{op_j}$ is the logical AND/OR operator associated with parameter j. By considering AND operator only for all nine parameters, the number of the CSPs that pass to the trust metric stage decreases (i.e., discarding the important ones). In the opposite, if the OR operator is only used, the number of CSPs increases, including less important ones. Therefore, the operator of each parameter could be AND or OR depending on its features. Namely, the used operators are determined according to Equation (5):

$$op_j = \begin{cases} \boldsymbol{AND}, & \boldsymbol{if\ mean\ value\ of\ (P_j, for\ all\ CSPs) \geq mean\ value\ of\ all\ parameters} \\ \boldsymbol{OR}, & \boldsymbol{otherwise} \end{cases} \ldots\ldots\ldots (5)$$

i.e., AND operator will be associated with parameter j in If-Then rule when the mean value of this parameter for all CSPs is greater than or equal to the mean value of all nine parameters' values for all CSPs. Otherwise; OR operator will be applied.

After determining the operators, they will be substituted in Equation (4) to filter the CSPs. We must notice that nine operators will be produced using Equation (5), while Equation (4) needs only eight operators. This can be done by using the conjunction relation between the operators of two consequence parameters as follows:

$$\left.\begin{array}{l} \boldsymbol{If\ (op_i == op_j), set\ Cop_{i,j} = op_i} \\ \boldsymbol{Otherwise;\ set\ Cop_{i,j} = XOR\ op} \end{array}\right\} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (6)$$

Where Copi,j is the conjunction operator between opi and opj of the consequence parameters.

After that, the value of each element $\mathbf{a_{ij}}$ in matrix A will be ranked GOOD or POOR depending on its value according to Equation (7):

$$\mathbf{Rank}\ (a_{ij}) = \begin{cases} \mathbf{GOOD}, & \mathbf{if\ mean}(A_j) < a_{ij} \leq \mathbf{max}(A_j) \\ \mathbf{POOR}, & \mathbf{otherwise} \end{cases} \quad \ldots\ldots\ldots (7)$$

Where mean ($\mathbf{A_j}$) and max ($\mathbf{A_j}$)) are the mean and the maximum values of parameter j for all CSPs, respectively.

Finally, the results of Equations (6) and (7) for each CSP will be applied to If-Then rule (i.e., Equation (4)) to determine that this CSP will be accepted to be involved in the trust metric stage (i.e., CSP is GOOD) or not (i.e., CSP is POOR).

3.2 Trusting phase

In the Trusting phase, the trust degree of the accepted CSPs for all services they provided will be defined using the Particle Swarm Optimization technique (PSO) [27], [28], [29], [30].

The PSO is considered one of the most commonly used optimization techniques in many areas such as function optimization, artificial neural network training, fuzzy system control, and other areas [28].

By developing Trusting phase using PSO technique, $\boldsymbol{v_i^k}$ and $\boldsymbol{x_i^k}$ denote the velocity and the position of particle i in iteration k, respectively. The velocity $\boldsymbol{v_i^{k+1}}$ is computed using Equation (8) [31]:

$$\boldsymbol{v_i^{k+1}} = \boldsymbol{w_i^{k+1}} * \boldsymbol{v_i^k} + \boldsymbol{c_1} * \boldsymbol{rand_1} * \left(\boldsymbol{pbest_i^k} - \boldsymbol{x_i^k}\right) + \boldsymbol{c_2} * \boldsymbol{rand_2} * \left(\boldsymbol{gbset^k} - \boldsymbol{x_i^k}\right) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (\mathbf{8})$$

Where $\boldsymbol{w_i^{k+1}}$is the inertia weight,$\boldsymbol{c_1}$, $\boldsymbol{c_2}$ are constants, $\boldsymbol{rand_1}, \boldsymbol{rand_2}$ are random numbers between 0 and 1, pbest is the best position that each particle reached, and gbest is the best position of the group of particles.

Position $\boldsymbol{x_i^{k+1}}$ of particle i is calculated based on its velocity $\boldsymbol{v_i^{k+1}}$ and the previous position $\boldsymbol{x_i^k}$ as in Equation (9) [31].

$$\boldsymbol{x_i^{k+1}} = \boldsymbol{x_i^k} + \boldsymbol{v_i^{k+1}} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (\mathbf{9})$$

Alam [32] has claimed that the common initial value of PSO velocity is between 10% and 90% of position value, the common number of iterations is between "500" and "10000", and the common value of inertia weight is in the range from 0.4 and 0.9.

A Dynamic Adaptive Particle Swarm Optimization (DAPSO) technique is developed to determine the value of inertia weight (w_i^{k+1}) for the PSO algorithm dynamically [23]. It uses a dynamic adaptive inertia factor in the PSO algorithm to adjust its convergence rate and control the balance of global and local optima. It determines the inertia weight as in Equations (10, 11).

$$\boldsymbol{w_i^{k+1}} = \boldsymbol{w_{min}} + (\boldsymbol{w_{max}} - \boldsymbol{w_{min}})\, \boldsymbol{sin}\left(\frac{\boldsymbol{\beta_i(k)\pi}}{\mathbf{2}}\right) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (10)$$

Where, w_{max} , w_{min} are the maximum and minimum values of w (i.e., 0.4, 0.9), respectively.

$$\boldsymbol{\beta_i(k)} = \frac{\boldsymbol{f_i(k) - f_g(k)}}{\boldsymbol{f_w(k) - f_g(k)}} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (11)$$

Where, $f_i(k)$ is the fitness function of the ith particle in kth iteration, and $f_g(k)$, $f_w(k)$ are the best and worst fitness values of the swarm in the kth iteration, respectively.

The Trusting phase has been implemented using DAPSO technique. In addition, a CSPs data store has been introduced to store the services' features of the accepted CSPs and their trust degrees.

To determine the CSPs' trust degrees, DAPSO technique is used with considering nine parameters, SLA and users' parameters, (i.e., Reputation, Availability, Turnaround Time, data Integrity, authorization, Data Recovery, Reliability, Users' Access

Frequency, and Service Success Rate) for each accepted CSP as PSO initial population. The model supposed that the fitness function for each CSP is calculated as in Equation (12):

$$\mathbf{F_i = 1 - X_i; i = 1, 2, \dots 9} \quad \dots\dots\dots\dots\dots\dots\dots\dots (\mathbf{12})$$

Where Xi is the parameter value (i) for each CSP.

The pseudo code of the used DAPSO technique is depicted as follows:

Algorithm DAPSO:

i. Initialize a population of particles having a position ($X_i^1 = T_i$), and velocity ($V_i^1 = 70\% \, of \, X_i$).

// T_i ; i = 1, … ,9 are the values of the proposed parameters for each accepted provider

ii. Set parameters of PSO ($c_1 = c_2 = 2$).

iii. Set iteration $k = 1$

iv. Calculate Fitness function $F_i^k = 1 - X_i^k, \forall i$

v. Set $Pbest_i^k = X_i^k, \forall\, i$ and $Gbest^k = max(X_i)$

vi. Set $k = k + 1$

vii. Update the inertia weight as in equations 10, 11

viii. $f_g(k) = Gbest^k$, $f_w(k) = min(F_i^k)$

ix. Update velocity and position of particles

$$V_i^{k+1} = w \times V_i^k + c_1 \times rand(1) \times \left(Pbest_i^k - X_i^k\right) + c_2 \times rand(2) \times \left(Gbest^k - X_i^k\right); \forall i$$

i. $X_i^{k+1} = X_i^k + V_i^{k+1}; \forall i$

x. Evaluate Fitness function $F_i^{k+1} = 1 - X_i^{k+1}, \forall i$

xi. Update $Pbest \, \forall i$

$$if \; F_i^{k+1} \le F_i^k \; then \; Pbest_i^{k+1} = X_i^{k+1} \; else \; Pbest_i^{k+1} = Pbest_i^k$$

xii. Update $Gbest \, \forall i$

$$if \; max\left(F_i^{k+1}\right) \le max(F_i^k) \; then \; Gbest^{k+1} = max(X_i^{k+1}) \; else \; Gbest^{k+1} = Gbest^k$$

xiii. If $k \neq 100$ then go to step (vi) else go to step (xii)

xiv. Output the solution as $Gbest^{k+1}$

3.3 SIMILARITY Phase

According to the proposed CSP Ranking model, the CSC defines his requested parameters (i.e., CPU Utilization, Response Time, Cost, Availability, Usability, Flexibility, Security Management, …, etc. [33], [13]) and their requested values about these parameters. Therefore, the similarity phase has been developed to determine the similarity value between the parameters of the CSC's required service and the associated parameters for each accepted CSP that could provide the required service. To evaluate the similarity degree, the proposed model uses Cosine Similarity measure by considering the angle between two sequences of values, where a greater similarity implies a

smaller angel [34]. Therefore, as the angle between the parameters of the CSC's required service and associated published parameters of the accepted CSPs is small, the similarity between them is large (see Fig. (3)).

The Cosine Similarity for each accepted CSP is calculated using Equation (13) [34]:

$$\boldsymbol{S[i] = \cos(\alpha) = \frac{\sum_{j=1}^{m} P_j[i]\, P_j[req]}{\sqrt{\sum_{j=1}^{m}(P_j[i])^2}\ \sqrt{\sum_{j=1}^{m}(P_j[req])^2}}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (13)$$

Where, $\boldsymbol{P_j[i]}$ is the parameter value (j) of the accepted CSPi, $\boldsymbol{P_j[req]}$ is the parameter value (j) for the CSC' required service, and $\boldsymbol{j = 1, 2, \ldots, m}$ is the number of parameters in the CSC's required service.

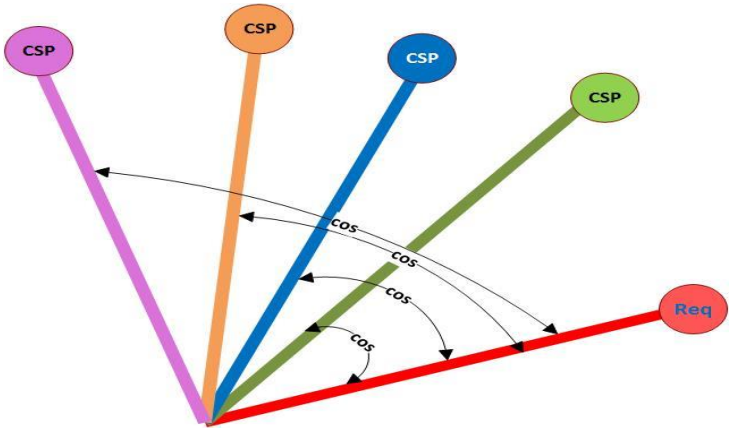

Fig. (3): Cosine Similarity between Parameters of CSC's required service and CSP's Parameters.

3.4 Ranking Phase

In this phase, the ranking value for CSPs will be determined by combining CSPs' trust degrees, and the similarity values which have been produced from the Similarity phase. The ranking value for each CSP is calculated using Equation (14).

$$\boldsymbol{Rank[CSP_i] = w_1 * T[i] + w_2 * S[i]\,, i = 1, 2, \ldots, n} \ldots\ldots \quad (14)$$

Where:

- $\boldsymbol{w_1}$, and $\boldsymbol{w_2}$ describe the weight of the trust degree and similarity value respectively. These values are defined by calculating the standard deviation of the trust degrees and similarity values for all trusted CSPs
- $\boldsymbol{T[i]}$ and $\boldsymbol{S[i]}$are the trust degree and similarity value for $\boldsymbol{CSP_i}$ respectively, and
- n is the number of accepted CSPs.

$\boldsymbol{w_1}, and\ \boldsymbol{w_2}$ are normalized as the sum of weight values equal 1. For example, suppose the standard deviation for trust degree and similarity value are $\boldsymbol{S_T = 0.1}$, $\boldsymbol{S_S = 0.3}$, respectively. These values are normalized by dividing them by their sum as $\boldsymbol{sum = 0.1 + 0.3 = 0.4}$ then $\boldsymbol{w_1 = \frac{S_T}{sum} = \frac{0.1}{0.4} = 0.25}$, $\boldsymbol{w_2 = \frac{S_S}{sum} = \frac{0.3}{0.4} = 0.75}$ where $\boldsymbol{w_1 + w_2 = 0.25 + 0.75 = 1}$

The CSP with higher ranking value (i.e. the higher trust degree and similarity value) is the most candidate one for providing the service to the CSC.

3.5 Data Normalization

Since the model is based on QoS parameters of CSPs and CSC requirement parameters, and these parameters have their different ranges and units. Therefore, it is necessary to normalize this data to a uniform range [0,1]. Suppose that $[QoS]^{CSP} \leftarrow (A_1, A_2, \dots A_n)$, represents the data provided by the CSPs, and $[QoS]^{CSC} \leftarrow (R_1, R_2, \dots R_m)$, represents the CSC request parameters. These data are normalized according to equations (15), (16) respectively:

$$\boldsymbol{norm}\,(\boldsymbol{A_i})^{\boldsymbol{CSP}} \leftarrow \frac{[\boldsymbol{A_i} - \boldsymbol{Min}(\boldsymbol{QoS})^{\boldsymbol{CSP}}]}{[\boldsymbol{Max}(\boldsymbol{QoS})^{\boldsymbol{CSP}} - \boldsymbol{Min}(\boldsymbol{QoS})^{\boldsymbol{CSP}}]} \dots\dots\dots (\mathbf{15})$$

$$\boldsymbol{norm}(\boldsymbol{R_i})^{\boldsymbol{CSC}} \leftarrow \frac{[\boldsymbol{R_i} - \boldsymbol{Min}(\boldsymbol{QoS})^{\boldsymbol{CSC}}]}{[\boldsymbol{Max}(\boldsymbol{QoS})^{\boldsymbol{CSC}} - \boldsymbol{Min}(\boldsymbol{QoS})^{\boldsymbol{CSC}}]} \dots\dots\dots (16)$$

where, $Min(QoS)^{CSP}$, $Min(QoS)^{CSC}$ are the minimum values of QoS parameters of CSPs and CSC requirement parameters, and $Max(QoS)^{CSP}$, $Max(QoS)^{CSC}$ are the maximum values of QoS parameters of CSPs and CSC requirement parameters, respectively.

4 Performance evaluation of The Proposed CSP Ranking model

The implementation environment and the time complexity of the proposed CSP Ranking model will be discussed. Then, a comparative study will be done to evaluate the performance of the proposed CSP Ranking model relative to the existing models (i.e., hypergraph computational (HGCM) [14], Hypergraph-Binary Fruit Fly Optimization Algorithm (HBFFOA) [20], security risk [18], and Extended TOPSIS (E-TOPSIS) [19]) .

4.1 The Implementation Environment

To evaluate the performance of our proposed CSP ranking model, HP ProBook computer with core i5@2.5 GHz processor, 6 GB RAM, under Windows Ten platforms are used. The experiments have been done using C# language using Microsoft Visual Studio 2010.

Armor Dataset is used to evaluate the proposed CSP Ranking model. This dataset contains the values of QoS parameters for 7334 CSPs [35]. According to Armor dataset, it is found that 6991 out of 7334 providers with no data. Therefore, only 343 providers out of 7334 will be used to evaluate the proposed model. Because the Filtering and Trusting phases of our proposed model are based on nine parameters (i.e., Reputation, Availability, Turnaround Time, data Integrity, authorization, Data Recovery, Reliability, Users' Access Frequency, and Service Success Rate), the Turn Around Time in Armor dataset is considered as the Response Time, Reputation as the Feature, Integrity as Ease of Use, Reliability as Technical Support, and Authorization as Customer Service. The parameters which do not exist in Armor dataset (i.e., Data Recovery, Service Success Rate, Users' Access Frequency) are generated randomly in the range of [0,1].

4.2 Comparative Study

This section provides the comparative study to evaluate the performance of our proposed CSP Ranking model with respect to the existing models (hypergraph computational (HGCM) [14], Hypergraph-Binary Fruit Fly Optimization Algorithm (HBFFOA) [20], security risk [18], and Extended TOPSIS (E-TOPSIS) [19]). The comparison is carried out using three steps; Validation, Time Complexity and Performance Evaluation of the proposed model.

4.2.1 Our Proposed Model Validation

To validate our proposed model, our proposed model and HGCM model have been implemented with considering the QoS case study which is presented in HGCM model [14]. This QoS case study is used in two different cases. The first case study consists of three IaaS CSPs (Amazon EC2, Windows Azure and Rackspace), and the second one consists of five IaaS CSPs (Amazon EC2, Windows Azure, Rackspace Private cloud using OpenStack, and Private cloud using Eucalyptus) (see Table (1)).

Table (1): QoS Dataset of Amazon EC2 (CSP1), Windows Azure (CSP 2), Rackspace (CSP 3), Private cloud using OpenStack (CSP 4) and Private cloud using Eucalyptus (CSP 5)

Parameter	***Case Study 1***			***Case Study 2***		***CSC Requirement***
	CSP1	***CSP2***	***CSP3***	***CSP4***	***CSP5***	
Accountability	4	8	4	5	5	4
CPU	9.6	12.8	8.8	12.8	9.6	6.4 GHZ
Memory	15	14	15	16	14	10 GB
Disk	1690	2040	630	500	400	500 GB
Availability	99.95%	99.99%	100%	99.99%	99.90%	99.9%
Cost	0.68$/hour	0.96$/hour	0.96$/hour	0.95$/hour	0.66$/hour	<1 $/hour
Data Inbound	10	10	8	10	10	100GB/Month
Data Outbound	11	15	18	11	11	200GB/Month
Storage	12	15	18	12	10	1000 GB
Service Response Time	120	780	200	180	180	120 sec
Security	4	8	4	5	5	4

With considering the first case study (i.e., three CSPs (CSP1, CSP2, CSP3, and one CSC), our proposed model produces the same results as HGCM model where the CSPs are ranked as CSP3<CSP1<CSP2 with the CSP3 is the most suitable provider. The execution time of implementing the existing HGCM model and our proposed CSP Ranking model are 1.0448, 0.6798 milliseconds respectively. Therefore, our proposed model produces the same result as the HGCM model with an execution time reduction of 34.9 %.

By considering the second case study (i.e., five CSPs (CSP1, CSP2, CSP3, CSP4, and CSP5) and one CSC), the implementation of the existing HGCM and our proposed

CSP Ranking models produce the same result with CSP3 is the most suitable provider. The execution time of implementing the existing HGCM model and the proposed CSP Ranking model is 3.363, 0.4138 milliseconds respectively. Therefore, the proposed CSP Ranking model outperforms the existing HGCM model by execution time reduction of 87.69%.

4.2.2 Time Complexity Comparison

In this section, the time complexity of our proposed model is calculated and compared with the existing models (hypergraph computational (HGCM) [14], Hypergraph-Binary Fruit Fly Optimization Algorithm (HBFFOA) [20]).

As our proposed model consists of four phases (Filtrating, Trusting, Similarity, and Ranking), we calculate its time complexity by calculating the time complexity of Trusting phase, Similarity phase, and Ranking phase).

4.2.2.1 Time Complexity of The Trusting Phase

In the Trusting phase, the trust values for all accepting CSPs will be calculated using the PSO technique. It should be noted that this phase is executed only once throughout the proposed CSP Ranking model. But, when a new CSP is included in the system or even the features of the existing CSP(s) are changed, the Trusting phase will be reactivated. Therefore, the time complexity for calculating the trust value for CSPs in the Trusting phase is determined in Equation (17):

$$\boldsymbol{O}[\boldsymbol{S}(\boldsymbol{kn}+\boldsymbol{k}+\boldsymbol{n})] \simeq \boldsymbol{O}(\boldsymbol{Skn}) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (\mathbf{17})$$

Where $\boldsymbol{S}$ is the number of the accepted CSPs, $\boldsymbol{k}$ is the number of iterations for the PSO technique, and $\boldsymbol{n}$ is the number of parameters in the trust calculation.

4.2.2.2 Time Complexity of The Similarity Phase

When a new request from CSC is sent to the system, the proposed model will pass it to the Similarity phase to calculate the similarity degree between this request and the parameters of the accepted CSPs. Therefore, the time complexity of the Similarity phase is determined using Equation (18):

$$\boldsymbol{O}(\boldsymbol{Sn}) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (18)$$

Where $\boldsymbol{n}$ is the number of parameters in the CSC's requested service and $\boldsymbol{S}$ is the number of the accepted CSPs.

4.2.2.3 Time Complexity of The Ranking phase

In the Ranking phase, the ranking value for each CSP is determined using equation (14). Therefore, the time complexity for this phase is determined using Equation (19):

$$\boldsymbol{O}(\boldsymbol{n}\log\boldsymbol{S}) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (19)$$

Where $\boldsymbol{n}$ is the number of parameters in the CSC's requested service and $\boldsymbol{S}$ is the number of the accepted CSPs.

It should be noticed that When the CSC sends his request, our proposed model retrieves CSPs trust values without recalculating it. Then, the model performs the similarity and ranking phases only.

According to the time complexity Equations 1, 3 and 19, the time complexity of the proposed CPS Ranking model is less than the time complexity of the HGCM, and the BHFFOA models [14], [20]. Also, the number of CSPs in our proposed CPS Ranking model is less than the number of CSPs in the existing HGCM and BHFFOA models because of using Filtrating phase.

In case of all CSPs are not filtered and passed to the Trusting phase, the number of CSPs in our developed CSP Ranking model will be equal to the number of CSPs in the existing HGCM, and BHFFOA models. This is considered the worst case of our developed CSP Ranking model. In this worst case, our proposed CSP ranking model still has low time complexity.

4.3 Performance Evaluation Using Armor Dataset

In this section, the comparative study of our proposed model is determined with respect to the existing models (hypergraph computational (HGCM) [14], security risk [18], and Extended TOPSIS (E-TOPSIS) [19]). This comparative study is based on defining two performance parameters; Execution Time, and Precision. Armor dataset is used to evaluate these two performance parameters by considering the number of available CSPs 20, 40… 340 respectively [35]

4.3.1 Execution Time

Fig. (4, 5) illustrate the execution time of our proposed and existing models (hypergraph computational (HGCM), security risk, and Extended TOPSIS (E-TOPSIS)) respectively. According to Fig. (4), we notice that the execution time of our proposed model and E-TOPSIS model are very close because of time scales. Therefore, Fig. (5)

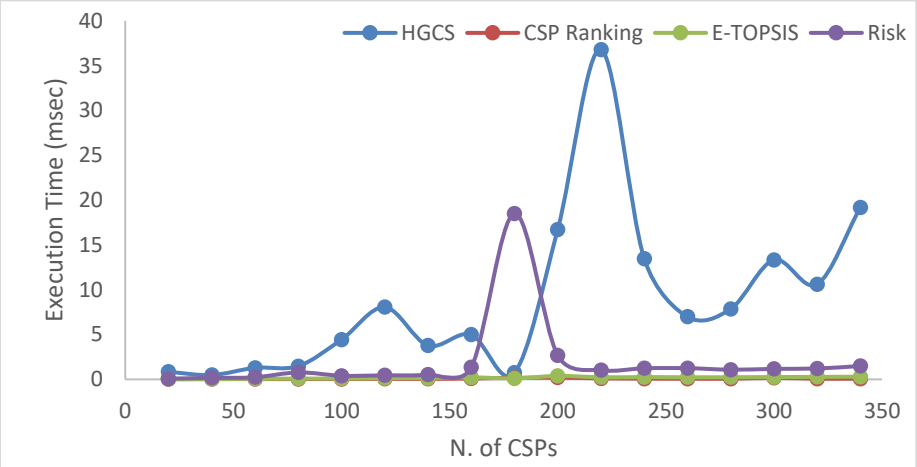

Fig. (4): Execution time for CSPs ranking and existing models

is introduced for these two models to identify the difference between them using a small range of time. Fig. (6) presents the average execution time of all models.

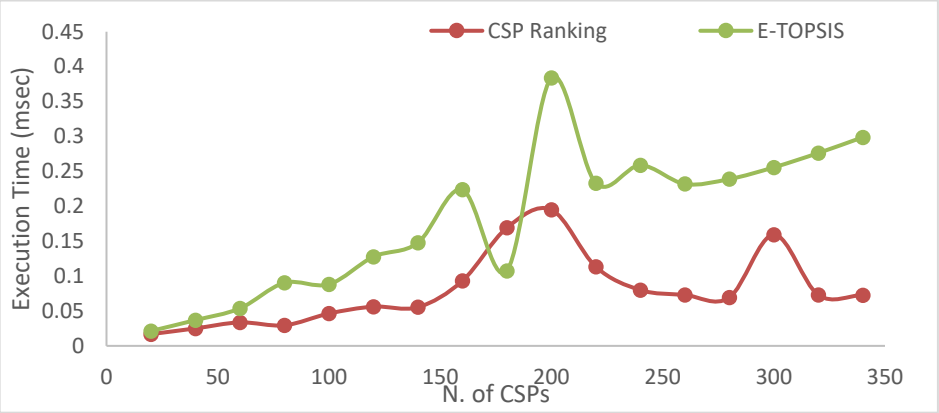

Fig. (5): Execution time for CSPs ranking and E-TOPSIS Models

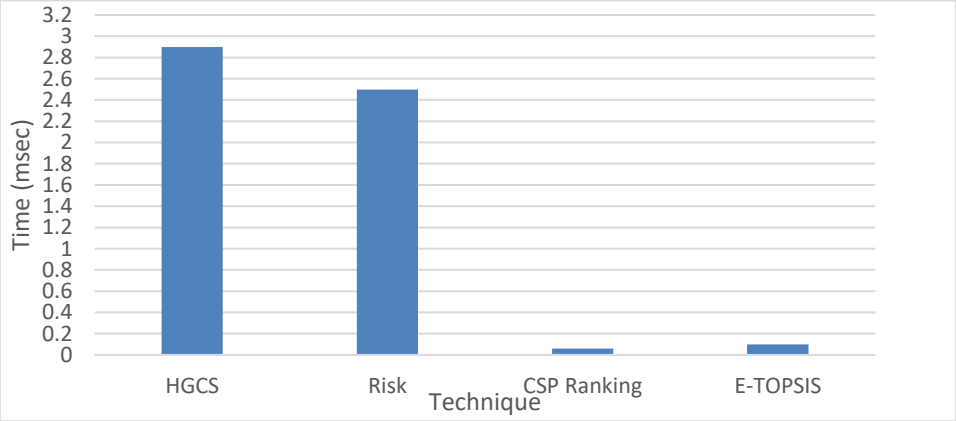

Fig. (6): Average execution time for CSPs ranking Models

According to the results in Fig. (4, 5, 6), the execution time of our proposed model is smaller than those existing models. From Fig. (6), it is noticed that the average execution time of the existing models; hypergraph computational (HGCM), security risk, and Extended TOPSIS (E-TOPSIS, and the proposed CSP Ranking models are 2.89, 2.49, 0.099, and 0.058 milliseconds respectively. So, the proposed CSP Ranking model outperforms the existing models (hypergraph computational (HGCM), security risk, and Extended TOPSIS (E-TOPSIS)) by reducing execution time by 97.99%, 97.67%, and 41.4% respectively, and the average reduction of the execution time of our proposed CSP Ranking model is 79.02% relative to the existing models.

4.3.2 Precision

Precision is defined as how experiment results are close to each other [36]. It's usually measured as the average deviation of data (see Equation (20)) [37].

$$\textbf{Precision} = \frac{\sum_{i=1}^{n} |x_i - \mu|}{n} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (20)$$

Where $\boldsymbol{\mu}$ is the mean value for data, and n is the data numbers.

We evaluate the precision of our proposed CSP Ranking model with respect to the existing models for 100 CSPs (see Fig. (7)), and the average of Precision for 340 CSPs is shown in Fig. (8).

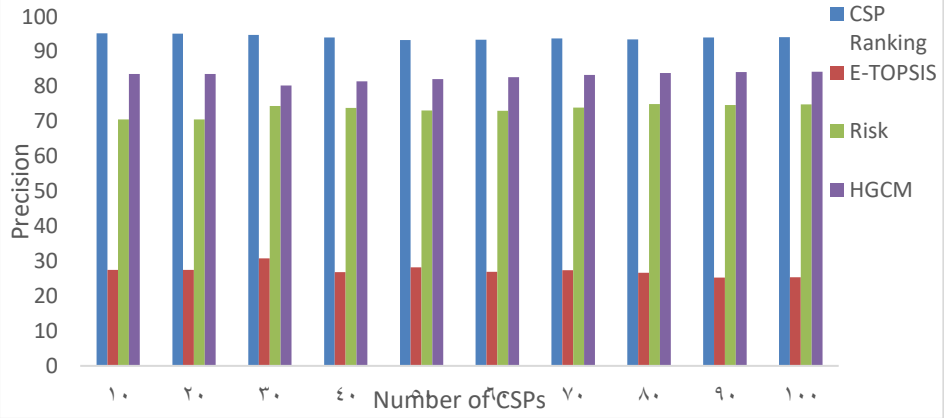

Fig. (7): Precision Evaluations for 100 CSPs

According to Fig. (7), it is found that our proposed model has higher precision values with respect to CSPs variation (i.e., 10, 20, …, 100 CSPs) relative to the existing models.

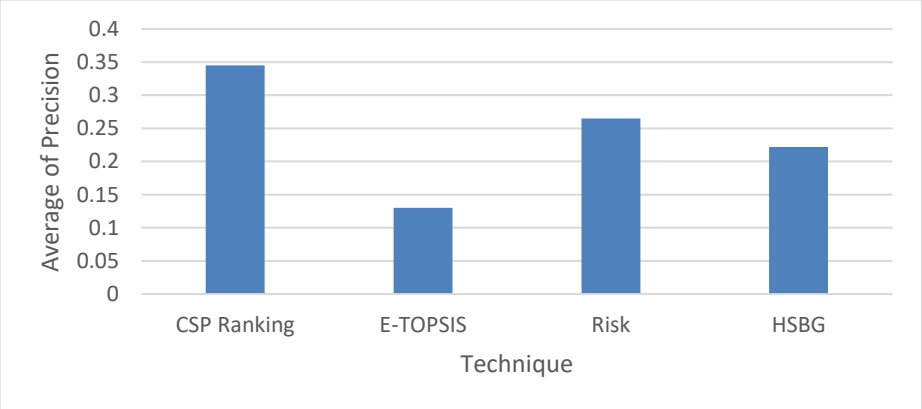

Fig. (8): Average of Precision for 340 CSPs

From Fig. (8), it is noticed that the average precision of the existing models (hypergraph computational (HGCM), security risk, and Extended TOPSIS (E-TOPSIS)) and the proposed model are 0.22, 0.26, 0.13, and 0.34 respectively. So, our proposed CSPs Ranking model outperforms the existing models (hypergraph computational (HGCM), security risk, and Extended TOPSIS (E-TOPSIS)) with respect to the average precision by 35.29%, 23.53%, and 61.76% respectively, and the average increase of the precision is 40.19%.

Generally, the proposed CSP Ranking model is considered the proper model to define CSPs' ranking degree due to the following four reasons:

1. The model has been developed in module manner using four phases (i.e., Filtrating, Trusting Similarity, and Ranking). Therefore, the model can easily be extended to include extra functions.
2. In most cases, the Filtrating and Trusting phases are activated only once.
3. The execution time of the model is decreased especially when the number of CSPs in the system has become large.
4. The precision of the model is increased by increasing the number of CSPs in the system.

5 Conclusions and Future Work

Ranking CSPs becomes one of the most important challenges in the cloud environment because it helps CSCs to select the nearest CSP that attains their requirements in suitable execution time. According to the work in this paper, a ranking model has been introduced. The proposed CSP Ranking model is based on the trust parameters of the existing CSPs, and the similarity between the parameters of CSPs, and the CSCs' requested services. The CSP Ranking model consists of four phases; Filtrating, Trusting, Similarity, and Ranking. In the Filtrating phase, the CSPs will be filtered by rejecting the unwanted CSP's based on their parameters. In the Trusting phase, the trust degrees of the accepted CSPs will be determined. The results of the first and second phases are stored in a CSPs' data store. In the Similarity phase, the similarity percentage between the parameters of the CSC's requested service and the CSPs' parameters is determined using the Cosine Similarity Measure. In the Ranking phase, the ranking values of CSPs are determined by combining CSPs' trust degrees and the similarity values between the services' features of the accepted CSPs offering the requested services and the CSC's service features to define the proper CSP that attains the CSC's requirements. Because the proposed Ranking model determines the trust value of the accepted providers only once, the search space is reduced, and the execution time is decreased. A comparative study has been done using Armor dataset among the existing models (hypergraph computational (HGCM), security risk, and Extended TOPSIS (E-TOPSIS)) and the developed CSP Ranking model to validate the performance of the proposed model. According to the comparative results, it is found that the proposed CSP's Ranking model outperforms the existing models (i.e.; HGCM, security risk, and E-TOPSIS) with respect to the execution time, the time complexity and the precision.

In the future, the proposed CSP Ranking model will be used in the services composition process to select the best providers.

References

[1] Andy Mulholland, Jon Pyke, Peter Fingar, "Enterprise Cloud Computing FAQ," TechTarget, December 2010. [Online]. Available: http://whatis.techtarget.com/definition/Enterprise-Cloud-Computing-FAQ. [Accessed April 2017].

[2] Samer Hasan, Vatsavayi Valli Kumari, "Numerical Similarity Algorithms for Cloud Service Discovery and Selection System," *International Journal of Intelligent Engineering and Systems,* vol. 10, no. 3, pp. 226-234, 2017.

[3] Rakesh Ranjan Kumar, Chiranjeev Kumar, "A Multicriteria Decision-Making Method for Cloud Service Selection and Ranking," *Advances in Computer and Computational Sciences,* vol. 2, pp. 139-147, 2018.

[4] W. Ashford, "Security in the cloud: Top nine issues in building users' trust," TechTarget, 2010. [Online]. Available: http://www.computerweekly.com/feature/Security-in-the-cloud-Top-nine-issues-in-building-users-trust. [Accessed May 2017].

[5] Ashraf Aboulnaga, Kenneth Salem, and Ahmed A. Soror, Umar Farooq Minhas, Peter Kokosielis, Sunil Kamath, "Deploying Database Appliances in the Cloud," *IEEE Data(base) Engineering Bulletin,* vol. 32, no. 1, pp. 13-20, 2009.

[6] Noha El. Attar, Wael Awad, and Fatma A. Omara, "RPOA WLB: Resource Provisioning Optimization Approach based on RPOA with Load Balance," *International Journal of Computer Applications,* vol. 105, no. 7, pp. 34-41, 2014.

[7] Gaurav Baranwal, Deo Prakash Vidyarthi , "A framework for selection of best cloud service provider using ranked voting method," in *2014 IEEE International Advance Computing Conference (IACC)* , Gurgaon, India, 2014.

[8] Michael Lang, Manuel Wiesche, Helmut Krcmar, "Criteria for Selecting Cloud Service Providers: A Delphi Study of Quality-of-Service Attributes," *International Journal of Information Systems Theories and Applications,* vol. 55, no. 6, pp. 746-758, 2018.

[9] J. Preethi, N. Sujaudeen, T. T. Mirnalinee, and P. Venugopal, "Cloud Service Ranking and Selection using Linear Programming," *International Journal of Computer Applications,* vol. 124, no. 3, pp. 39-43, 2015.

[10] Atoosa Gholami, Mostafa Ghobaei Arani, "A trust model for resource selection in cloud computing environment," in *2nd International Conference on Knowledge-Based Engineering and Innovation*, Iran, 2015.

[11] Mohamed Firdhous, Osman Ghazali, Suhaidi Hassan, "Trust Management in Cloud Computing: A Critical Review," *International Journal on Advances in ICT for Emerging Regions,* vol. 4, no. 2, pp. 24-36, 2011.

[12] Ritu, Sukhchandan Randhawab, Sushma Jain, "Trust Models in Cloud Computing: A Review," *I.J. Wireless and Microwave Technologies,* vol. 4, pp. 14-27, 2017.

[13] Azubuike Ezenwoke, Olawande Daramola, Matthew Adigun, "QoS-based ranking and selection of SaaS applications using heterogeneous similarity metrics," *Journal of Cloud Computing: Advances, Systems and Applications*, vol. 7, no. 15, pp. 1-12, 2018.

[14] Nivethitha Somu, Kannan Kirthivasan, Shankar Sriram , "A computational model for ranking cloud service providers using hypergraph based techniques," *Future Generation Computer Systems*, vol. 68, pp. 14-30, 2017.

[15] Saurabh Kumar Garg, Steve Versteeg, Rajkumar Buyya, "SMICloud: A Framework for Comparing and Ranking Cloud Services," in *2011 Fourth IEEE International Conference on Utility and Cloud Computing*, Australia, 2012.

[16] Mohammad Riyaz Belgaum, Safeeullah Soomro, Zainab Alansari, Muhammad Alam, "Cloud Service Ranking using Checkpoint based Load Balancing in Real Time Scheduling of Cloud Computing," in *International Conference on Advanced Computing and Intelligent Engineering (ICACIE2016)*, India, 2016.

[17] Dilbag Singh, Jaswinder Singh, Amit Chhabra , "Evaluating Overheads of Integrated Multilevel Checkpointing Algorithms in Cloud Computing Environment," *I.J. Computer Network and Information Security*, vol. 5, pp. 29-38, 2015.

[18] Jamal TALBI, Abdelkrim HAQIQ, "A Novel Framework for Ranking Cloud Service Providers Using Security Risk Approach," in *International Conference on Big Data Cloud and Applications*, Morocco, 2015.

[19] Constanţa Zoie RĂDULESCU, Iulia Cristina RĂDULESCU, "An Extended TOPSIS Approach for Ranking Cloud Service Providers," *Studies in Informatics and Control*, vol. 26, no. 2, pp. 183-192, 2017.

[20] Nivethitha Somu, Gauthama Raman M.R., Kannan Kirthivasan, Shankar Sriram V.S. , "A trust centric optimal service ranking approach for cloud service selection," *Future Generation Computer Systems* , vol. 86, p. 234–252, 2018.

[21] Alshaimaa M. Mohammed, Ehab Morsy, Fatma A. Omara, "TRUST MODEL FOR CLOUD SERVICE PROVIDERS USING FUZZY CONTROLLER," *International Journal of Computer Science and Information Security (IJCSIS)*, vol. 15, no. 1, pp. 373-387, 2017.

[22] V. Bezděk, "Using fuzzy logic in business," *Procedia - Social and Behavioral Sciences*, pp. 371-380, 2014.

[23] Hao Zhu, Yumei Hu, Weidong Zhu, "A dynamic adaptive particle swarmoptimization and genetic algorithm fordifferent constrained engineeringdesign optimization problems," *Advances in Nonlinear Dynamics and Vibrations on Mechanical Systems*, vol. 11, no. 3, pp. 1-27, 2019.

[24] Guanrong Chen, Trung Tat Pham, Introduction to Fuzzy Sets,Fuzzy Logic,and Fuzzy Control Systems, Florida: CRC Press LLC, 2001.

[25] A. M. D. Alvaré, FUZZY LOGIC AS AN INSTRUMENT TO PERFORM SECURITY ANALYSIS, United States: ProQuest LLC, 2015.

[26] V. Prasath, Nithya Bharathan, Neetha N.P, N. Lakshmi, M.Nathiya, "Fuzzy Logic In Cloud Computing," *International Journal of Engineering Research & Technology (IJERT),* vol. 2, no. 3, 2013.

[27] Garima Grover, Ila Chaudhary, "Implementation of Particle Swarm Optimization Algorithm in VHDL for Digital Circuits Optimization," *I.J. Information Engineering and Electronic Business,* vol. 5, pp. 16-21, 2014.

[28] Yudong Zhang, Shuihua Wang, and Genlin Ji, " A Comprehensive Survey on Particle Swarm Optimization Algorithm and Its Applications," *Mathematical Problems in Engineering,* vol. 2015, pp. 1-38, 2015.

[29] Federico Marini, Beata Walczak, "Particle Swarm Optimization (PSO). A tutorial," *Chemometrics and Intelligent Laboratory Systems,* vol. 149, pp. 153-165, 2015.

[30] S. D. Chavan, Nisha P. Adgokar, "An Overview on Particle Swarm Optimization: Basic Concepts and Modified Variants," *International Journal of Science and Research (IJSR),* vol. 4, no. 5, pp. 255-260, 2013.

[31] Bachelorarbeit, Miljenko Jakovljevic, Particle Swarm Optimization for Generating Input Data in Measurement Based Worst-Case Execution Time Analysis, Wien: Vienna University of Technology, 2011.

[32] M. N. Alama, "Particle Swarm Optimization: Algorithm and its Codes in MATLAB," *ResearchGate,* pp. 1-10, 8 March 2016.

[33] Tran Cong Hung, Nguyen Xuan Phi, "STUDY THE EFFECT OF PARAMETERS TO LOAD BALANCING IN CLOUD COMPUTING," *International Journal of Computer Networks & Communications,* vol. 8, no. 3, pp. 33-45, 2016.

[34] Jiqian Chen, Jun Ye, Shigui Du, "Vector Similarity Measures Between Refined Simplified Neutrosophic Sets and Their Multiple Attribute Decision-Making Method," *Symmetry,* vol. 9, no. 153, pp. 1-13, 2017.

[35] Talal H. Noor and Quan Z. Sheng, "Trust as a Service: A Framework for Trust Management in Cloud Environments," in *International Conference on Web Information Systems Engineering*, Berlin, Heidelberg, pp.314-321, 2011.

[36] P. Anderson, "Accuracy and Precision," SOPHIA Learning, 2013. [Online]. Available: https://www.sophia.org/tutorials/accuracy-and-precision--3. [Accessed 10 2018].

[37] Mohammad Fraiwan Al-Saleh, Adil Eltayeb Yousif, "Properties of the Standard Deviation that areRarely Mentioned in Classrooms," *AUSTRIANJOURNAL OFSTATISTICS,* vol. 38, no. 3, p. 193–202, 2009.