Journal of Theoretical and Applied Information Technology
15th October 2022. Vol.100. No 19
© 2022 Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

QOS AWARE WEB SERVICES COMPOSITION PROBLEM IN MULTI-CLOUD ENVIRONMENT USING HYBRID OPTIMIZATION ALGORITHM

G. AMIRTHAYOGAM[1*], DR. C. ANBU ANANTH[2], P. ELANGO[3]
[1*]Research Scholar, Annamalai University, Department of Computer Science and Engineering, Chidambaram, Tamilnadu, India, Email: amir.yogam@gmail.com
[2]Associate Professor, Annamalai University, Department of Computer Science and Engineering, Chidambaram, Tamilnadu, India, Email: anbu_ananth2006@yahoo.com
[3]Assistant Professor, Perunthalaivar Kamarajar Institute of Engineering and Technology, Department of Information Technology, Karaikal, Puducherry, India,

Email: elanalin74@gmail.com

ABSTRACT

Information technology is building communications networks for a company, safeguarding data and information, and creating and administering databases. Currently, cloud computing provides a single set of physical resources for providing multiple information technology (IT) services to a large user base with varying needs. With the advent of multi-cloud computing, abundant web services are published by several providers to their worldwide users. Web service composition technology attracted a lot of attention for the sake of reduction in software development costs. In a multi-cloud environment (MCE), each atomic web service published by any cloud provider with the same functionality has a different price and quality of service (QoS). Service discovery and composition are the key challenges for web services development. The challenges in the composition of services distributed in multi-cloud environments include increased cost and a reduction in its speed due to the increasing number of services, providers, and clouds. Consequently, to overcome these challenges, the QoS-aware multi-cloud web service composition is presented in this work. The research work initially proposed a Deep Neural Network (DNN) aim of the study is to recommend the web service composition. The proposed method recommends the matched services to the user based on the user's need. Subsequently, a Hybrid Firefly and Bee Colony Optimization Algorithm is introduced for the Multi-cloud service composition problem of NP-Hard in a multi-cloud environment. The proposed optimization algorithm reduces the number of cloud providers to provide the best services. Additionally, the security of multi-cloud web service composition is important, to provide this security, a Fuzzy Generalized Rough Set Theory is presented in this paper. Accordingly, this fuzzy rough set theory eliminates insecure services. Subsequently, the proposed work is implemented using Python software. The performance metrics are throughput, response time, availability, feasibility, efficiency, etc. The proposed method is compared with the existing BOTV-PSO, MLTS-MCSC, IWD, and GNN-QSC. Subsequently, compared to these existing methods, the proposed method is 2% more than the existing methods for execution time, for response time, the proposed method is approximately 3% higher than the existing methods. The proposed method is 5% higher than the existing methods for availability and 4% higher for throughput. The proposed work performs the best service composition to the user and in future determines the minimum and maximum allowable used clouds to minimize the communication costs in a multi-cloud environment respectively.

Keywords: *Web Services Composition, Service Recommendation, Deep Neural Network, Hybrid Algorithm, Fuzzy Generalized Rough Set Theory, QoS Metrics.*

1. INTRODUCTION

Cloud-based services provide information technology (IT) as a service over the Internet or dedicated network, with delivery on demand, and payment based on usage. Cloud-based services range from full applications and development platforms to servers, storage, and virtual desktops. Over recent years, the design and development of software systems have been undergoing extraordinary changes driven by the fast advancement of the Web [1]. Web services have been created to support the automated use of Web applications and the evolution of the Web. Subsequently, this provides high-level abstractions for organising applications in large-scale environments. Additionally, this is because of essential features Web services hold, such as being loosely coupled, reusable, and configurable [2]. Service-Oriented Computing (SOC) is a major paradigm that provides a set of standards and concepts for Web services, such as Web Services Description Language, Business Process Execution Language, and Simple Object Access Protocol [3]. SOC is a concept that assembles software applications to create a network of services to achieve rapid, low-cost, and cross-organisational distributed business processes. The building blocks of SOA, Web services are widely supported by service providers such as Amazon Web Services [4]. Existing services often cannot satisfy the diversified needs of users. Web Service Composition (WSC) is an application of SOA, which creates new services via integrating existing services to accomplish advanced functionalities [5].

QoS Aware Web Service Composition

The process of choosing specific services from numerous abstract tasks and integrating them into big granular services is known as WSC [6]. According to particular business principles, the web composition service is to connect services organically from the standpoint of the business process and to achieve specific goals each abstract service in service composition collaborates is described in [7]. Individual Web services have functional attributes specified by their inputs and outputs, which require a set of inputs and produce a set of outputs after the execution. Consequently, the functional correctness of a composite service should be satisfied when combining different web services. While many web services deliver the same functionality, non-functional Quality of Service (QoS) attributes, such as response time and cost (including communication time and cost), become discriminating factors. QoS requirements must be considered explicitly for an effective composition and are clearly defined in the Service Level Agreement (SLA) [8]. SLA can be described as a declarative contract between users and service providers. In recent years, with the continuous development of cloud computing, more and more web services are published on the web. Individual services can no longer meet the needs of users, and more and more users are composing Web services to meet various business processes to achieve more complex functions [9]. WSC'S powerful service function can help enterprises to gain more benefits and to save money. Consequently, the research in [10] focused on the proper service composition in the selection of a large number of web services on the Internet.

Web Service Composition in Multi-Cloud Environment

Web service is a modular and self-described application that is published based on a set of standards such as SOAP, WSDL, and UDDI [11]. Service composition problems can be resolved by selecting a set of web services in such a way that their combination meets the functional and non-functional requirements of the user [12]. With the advent and rapid development of cloud computing, more clouds can carry out the existing tasks in the cloud with different functions, and this cloud environment is a natural choice for providing various types of resources as a service [13]. To meet the user's needs, cloud-based systems are usually designed by calling up several providers. The service composition in cloud environments allows for the integration of various cloud resources into a set of integrated services for providing cloud-based solutions that meet certain qualitative criteria. Most of the service composition methods that have been proposed for cloud computing consider all the composite services in one cloud, rather than searching for services from the various available clouds [14]. Organizations often distribute their services using cloud providers to ensure the availability and quality of the provided services, and also to reduce the risk of data loss. In addition, service composition in multi-cloud environments poses many issues such as the cost of communications within the cloud, increased fiscal costs, and security issues. Subsequently, challenging tasks include reducing the number of participating clouds and the number of providers due to the limitations of the services [15]. IT services refer to the application of business and technical expertise to enable organizations in the creation, management and optimization of or access to information and business processes. To ensure the efficiency and security of work, a multi-cloud service composition environment is presented. However, to overcome the Web Service Composition problem of NP-Hard in a multi-cloud environment, a hybrid Firefly and Bee Colony Optimization Algorithm is presented in this work. Consequently, the current research work seeks to find the best possible service composition in multi-cloud environments. The rest of the paper is demonstrated as follows, section 2 depicted the literature survey of the study, and section 3 portrays the problem definition and motivation of the research. Section 4 illustrates the proposed research methodology, section 5 reveals the experimentation and result discussion of the work, and section 6 discloses the conclusion of the research work.

2. LITERATURE REVIEW

The literature survey of the work is based on the study of the web service recommendation process with different algorithms. Subsequently, this work presented the DNN method for web service composition recommendation.

Mirsaeid Hosseini Shirvani *et al* [16] overcome the combinatorial problem by introducing the bi-objective time-varying particle swarm optimisation (BOTV-PSO) technique. A proper balance of exploration and exploitation is established by the parameters that are adjusted based on the elapsed time. A single objective genetic algorithm (SOGA) that only optimises the cost function and ignores CSR, and a multi-objective simulated annealing technique were used (MOSA), and several scenarios are defined and compared the algorithm with the multi-objective GA-based (MOGA) optimiser, to demonstrate the effectiveness of the algorithm. The BOTV-PSO outperformed previous techniques in terms of convergence, diversity, scalability, fitness, and performance, based on the experimental results.

In a multi-cloud environment, a framework for service composition is proposed by Abdelbasset Barkat *et al* [17], these services are composed based on two factors to create a composing service that meets the user's request: the first one for each service, a set of QoS (quality of service) criteria, and the second is the number of cloud bases involved in the composition process. In multi-cloud environments, a hybrid formal verification approach is proposed by Alireza Souri *et al* [18] for assessing service composition, then a final service composition with a high level of QoS is achieved by reducing the number of cloud providers. In a multi-cloud context, this approach uses behavioural modelling to investigate the process of composition, user requests, and service selection. A Pi-Calculus-based process algebra methods and Multi-Labelled Transition Systems (MLTS)-based model checking for monitoring the non-functional properties and functional specifications as the QoS standards, the suggested technique allows for the analysis of service composition. For multi-cloud service composition, the suggested method meets the functional requirements.

In a multi-Cloud context, Samar Haytamy *et al* [19] developed an upgraded QoS-based Service Composition Approach to precisely compose the best Cloud providers to contract with them for composing the desired services to reduce the Cloud consumer cost function. The best services based on the uncertainty of QoS attributes were composed using a modified Particle Swarm Optimization (PSO) in this article. Using a real QoS dataset, the proposed approach has been implemented. In comparison to existing models, the proposed approach has attained a high degree of optimality with low time complexity, according to the comparative results.

A security-aware multi-cloud service composition approach based on fuzzy Formal Concept Analysis (FCA) and rough set theory (RS) is proposed by Fatma Lahmar *et al* [20], which are the two mathematically strong techniques. The paper introduced the fuzzy relations of fuzzy FCA and the approximation of RS to ensure a high level of security for the hosting clouds and selected services. Fuzzy FCA will ensure that the composing services are chosen from a small number of clouds because of its bottom-up parsing technique, which lowers inter-cloud communication costs. The approach's effectiveness and performance were demonstrated by the experimental results.

M. Heidari *et al* [21] suggested the Skyline service algorithm composing services in multi-cloud environments, which examined all the clouds during the service composition process. The proposed method can provide an applicable composition service to the user with the lowest communication cost by considering the number of clouds and by using fewer providers. Additionally, there are two steps involved in the Skyline algorithm. First, by considering the communication time and the number of providers the optimum cloud composition is chosen from among all feasible providers. Subsequently, to construct all of the possible compositions in a multi-cloud environment, the Skyline algorithm is used in the second stage. Shorter communication durations and fewer clouds between the clouds are the parameters that are chosen. Accordingly, it can be said that Skyline makes it possible to select a suitable composition of user-requested services in a multi-cloud environment.

Huayi Yin *et al* [22] proposed a method for executing atomic services in a composite request using the MCE and an energy-aware multiple targets service composition method. The network differs between composite requests and clouds or even between clouds. In order to construct all atomic services, a composite request requires multiple clouds to work together and the cloud can only deliver a limited number of services. "All-Search" is denoted by exploring all feasible mappings between service request blocks (a collection of requests to several atomic services) and clouds that this technique obtains scheduling. In order to achieve multiple targets, a heuristics algorithm that can achieve "Itersplit" is proposed to reduce the complexity of All-Search.

Remaci Zeyneb Yasmina *et al* [23] devised a majority judgment-based algorithm for narrowing the search space. In order to select the Top K compositions that meet the QoS global criteria, it uses constraint programming search. The approach's outstanding performance is demonstrated by the experimental results. In practice, end-users look for web service compositions that best fulfil their QoS needs (i.e., QoS global constraints). the task of selecting optimal compositions was completed, even though the number of services is constantly expanding and their corresponding QoS is intrinsically unknown (due to environmental variables).

FatehSeghir *et al* [24] suggested the multi-objective quality of service (QoS)-driven WSC problem (MOQWSCP) aims to find the best combinations of atomic web services (i.e. composite service). In order to solve the defined FMOQWSCP, a fuzzy discrete multi-objective artificial bee colony (FDMOABC) technique is proposed, which includes a fuzzy ranking mechanism to deal with solution sorting and to control and maintain the diversity of FDMOABC's solutions, a new fuzzy distance measure is used. Moreover, among the Pareto-optimal solutions generated by FDMOABC, and to find the best composite services, a fuzzy multi-criteria decision-making approach (FMCDMM) is presented.

Chen Wang *et al* [25] anticipated a memetic Estimation of the Distribution Algorithm based approach, namely MEEDA, to tackle this problem. Utilizing a variety of neighbourhood structures, MEEDA explored four domain-dependent local search strategies that look for effective composite services. Aside from that, an efficient local search approach is presented, which combines a uniform fitness distribution scheme for picking acceptable solutions with stochastic local search operators for effectively and efficiently exploiting neighbours, to greatly lower the computing time of MEEDA. Create a more complex, expanded version of the service composition benchmark dataset to better illustrate MEEDA's scalability and effectiveness.

From the literature studies, the composition of Web services can define applications that increase in complexity through progressively aggregating new components at a high level of abstraction. With the rapid growth in the number of available Web services, has become virtually impossible for a human user to analyze all these services and generate the composition plan manually. This raises the need for automation of web services composition. The need for automatic composition of services is justified by the ubiquitous Internet which forces companies to abandon the legacy business models and outdated systems and organize the virtual enterprise. In this work, the deep neural network method recommends the matched services. Subsequently, to overcome the Web Service Composition (WSC) problem of NP-Hard in a multi-cloud environment, a hybrid optimization algorithm is presented.

3. RESEARCH PROBLEM DEFINITION AND MOTIVATION

The number and diversity of Web-accessible services have been increasing rapidly, and are expected to keep growing shortly. While so many services can provide more choices for SOA-based applications, which brings about a heavy burden on modelling and verification of service composition that plays a very important role in SOA methodology. With the rapid growth of the number of web services and the complexity of composition logic, the construction, adaptation, and evolution of such compositions become more and more difficult, time-consuming, error-prone, or even sometimes impractical. Meanwhile, it becomes a big data challenge problem, how to discover a service composition satisfying a special request efficiently from the huge number of services. An important goal of web-service composition is to achieve maximum flexibility in adapting dynamically to an environment in which the available services (whether simple or composed services) are constantly changing in terms of availability, load balancing or application. Ensure efficiency and security are an important problem in web service recommendation and the Web Service Composition contains the problem of NP-Hard in a multi-cloud environment.

In order to better meet users' complex requirements in the face of the vast number of cloud services available and with the advent of cloud computing, the virtualized environment's constraints, in terms of QoS, interoperability, security policies, resource availability, and so on. The new techniques now span multiple clouds, because it has been proven that composing services from several clouds is more beneficial than relying on services from a single cloud. Despite the advantages of multi-cloud environments, there are always some security risks that most threaten the cloud consumers' data, which makes the identification of suitable services a challenging task. Due to the growing number of clouds, services, and providers, composition services distributed in multi-cloud systems face greater costs and a speed reduction. In this work, the efficiency and security of the work are provided by the multi-cloud web service composition process. However, to overcome the WSC problem of NP-Hard in a multi-cloud environment, a hybrid optimization is suggested. Consequently, the number of providers and participating clouds must be minimised to overcome these obstacles.

4. PROPOSED RESEARCH METHODOLOGY

Web Service Composition (WSC) becomes a more important technology in the domain of web service with the increase in the number of web services. The WSC provides an effective solution to complex web services. The term "cloud computing" has recently become popular in the search community, indicating the importance scientists place on this field of research. Cloud computing is a new computing model that provides shared resources and data based on a service delivery model, where everything from infrastructures, platforms, and software are given to the user like a set of services. In order to fulfil their requests, the users of cloud platforms interact with these services; users may require multiple services to complete a single request because requests become more complicated.

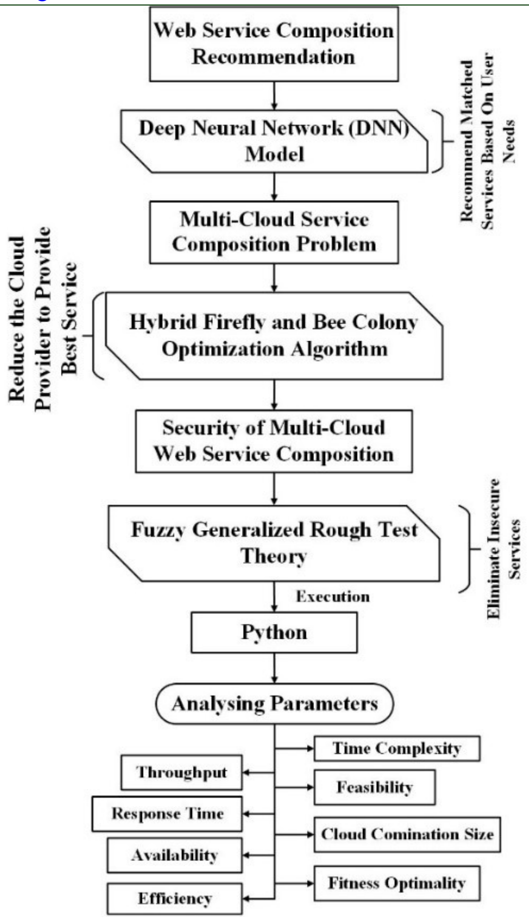
Web Service Composition Recommendation
Deep Neural Network (DNN) Model
Recommend Matched Services Based On User Needs
Multi-Cloud Service Composition Problem
Reduce the Cloud Provider to Provide Best Service
Hybrid Firefly and Bee Colony Optimization Algorithm
Security of Multi-Cloud Web Service Composition
Fuzzy Generalized Rough Test Theory
Eliminate Insecure Services
Execution
Python
Analysing Parameters
Throughput
Response Time
Availability
Efficiency
Time Complexity
Feasibility
Cloud Comination Size
Fitness Optimality

Figure 1: Flow Diagram of the Proposed Work

The process of gathering a set of services to satisfy a user request is called service composition. Subsequently, this research work entitled QOS criteria (Response Time, Throughput, availability) of web service composition in the multi-cloud environment developed two approaches for minimizing the number of cloud bases involved in the process of service composition. Figure 1 illustrates the flow diagram of the proposed work.

4.1 Web Service Composition (WSC) Recommendation

Users gradually participate in the creation of web content with the rapid development of Web 2.0. Nevertheless, it is becoming more and more difficult to meet users' complex needs with a single service. Consequently, users begin to combine different services for the generation of their service composition. In order to produce a unified application, service composition is a collection of services arranged in a logical order. Due to the lack of domain and interface knowledge, it is difficult for users to create suitable service processes for their business needs. Accordingly, the research work proposed a Deep Neural Network (DNN) model for the service composition recommendation method.

Assisting in the service composition process, the DNN algorithm is used to suggest matched services to the user. The Bayesian classifier is used to model their interests, and other service compositions that satisfy their interests are recommended to the user when the user completes a service composition, considering the diversity of user interests. Moreover, the search for services does not satisfy user functional and non-functional requirements. In this section, service compositions are first sent to DNN for training. The DNN, on the other hand, is developed to meet the customer's weight desire through deep learning training set (DLTS) training so that it may be used for personalised suggestions. The following sections describe the detailed training process of DNN.

4.1.1 Deep Neural Network

The structure of the DNN is depicted in figure 2, it mainly consists of five components: (1) characterization pre-processing layer; (2) embedding layer; (3) input layer; (4) hidden layer and (5) output layer. The characterisation pre-processing layer and embedding layer in DNN are designed to lower the training process's computing complexity. The DNN uses a double hidden layer structure to learn the mapping relation between the composite service and the associated weight due to the difficulty of fitting the preference weights.

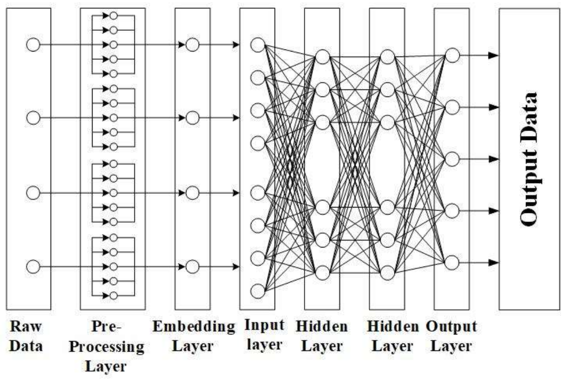

Figure 2: Structure of DNN

Figure 2 portrays the structure of DNN, and the training process of DNN can be described as follows:

Step 1: Data Pre-Processing

In DNN, the feature vectors of each input training sample are obtained through characterization pre-processing. Embedding processing is to convert the multiple feature vectors into a dense eigenvector. To ensure that the training is not under-fitted, this makes the input training data more useful and compact.

Step 2: Update the weight of the hidden layer

DNN learns the relationship between input and output data by iteratively adjusting the weight of hidden layer nodes using the Adam approach throughout the training phase. Accordingly, to personalised preferences (QoS criteria weights), the neural network structure is constantly learning to transfer user preferences (composite services).

Step 3: Decision Boundary Fitting

DNN constantly fits the decision boundary of the labelled training data as the number of training iterations increases. The fitted decision boundary allows DNN to classify composite services when training is completed.

Characterization Pre-Processing: The raw input training data is optimized for characterization pre-processing to guarantee the accuracy of DNN. For each service vv^i in input training data SP^i, the characteristics (QNV) are calculated. Finally, a feature vector $cv\left(vv^i\right)$ is formulated in equation (1).

$$cv\left(vv^i\right)=\left(R_c\left(vv^i\right),R_t\left(vv^i\right),R_a\left(vv^i\right),R_r\left(vv^i\right),R_f\left(vv^i\right)\right) \quad (1)$$

Embedding: The principle of embedding technology is to represent each input data with a dense feature vector rather than the original sparse vector. The operating performance of DNN can be improved by incorporating sparse feature vectors as input. Pre-process each service vv^i for each raw input data set SP^i using equation (1), and create a pre-processed matrix using equation (2).

$$PM\left(SP^i\right)=\begin{bmatrix} R_c\left(vv^1\right) & R_c\left(vv^2\right) & \dots & R_c\left(vv^i\right) & \dots & R_c\left(vv^{mt}\right) \\ R_t\left(vv^1\right) & R_t\left(vv^2\right) & \dots & R_t\left(vv^i\right) & \dots & R_t\left(vv^{mt}\right) \\ R_a\left(vv^1\right) & R_a\left(vv^2\right) & \dots & R_a\left(vv^i\right) & \dots & R_a\left(vv^{mt}\right) \\ R_r\left(vv^1\right) & R_r\left(vv^2\right) & \dots & R_r\left(vv^i\right) & \dots & R_r\left(vv^{mt}\right) \\ R_f\left(vv^1\right) & R_f\left(vv^2\right) & \dots & R_f\left(vv^i\right) & \dots & R_f\left(vv^{mt}\right) \end{bmatrix} \quad (2)$$

From the above equation to generate a new dense feature vector, the $PM\left(SP^i\right)$ is dimensionally reduced:

$$DV\left(SP^i\right)=\left(cv\left(vv^1\right),cv\left(vv^2\right),\dots,cv\left(vv^i\right),\dots,cv\left(vv^{mt}\right)\right)_{1\times 5mt} \quad (3)$$

Where $DV\left(SP^i\right)$ represents the embedded dense feature vector SP^i and it is the input vector for the DNN's input layer.

Training Parameter of DNN

In the preparation for training, mean squared error (MSE) is employed as the loss function of DNN. The formula of MSE is shown in equation (4).

$$MSE=\frac{1}{Mm}\sum_{i=1}^{m}\left\|\hat{y}^{(i)}-y^{(i)}\right\|^{2} \tag{4}$$

Where m is the quantity of training samples, $\hat{y}^{(i)}$ is the i th actual output of the training set, $y^{(i)}$ is the i th expected output of the training set.

In the DNN training process, the Adam approach is used to conduct error backpropagation. The Adam method's calculating formula is as follows:

$$g_t=\nabla_\theta L\left(f\left(x^{(i)};\theta\right),y^{(i)}\right) \tag{5}$$

$$s_t=\rho_1 s_{t-1}+\left(1-\rho_1\right)g_t \tag{6}$$

$$r_t=\rho_2 r_{t-1}+\left(1-\rho_2\right)g_t \tag{7}$$

Where, s_t is the first-moment estimation of g_t, r_t is the second-moment estimation of g_t, ρ_2 is defaulted to 0.9, g_t is the gradient of MSE, ρ_1 is defaulted to 0.999, and L is the loss function MSE.

Since, s_t and r_t are initialized as 0 vectors, the deviation of s_t and r_t need to be corrected. The annotation formula is shown as follows:

$$\hat{s}_t=\frac{s_t}{1-\rho_1^t} \tag{8}$$

$$\hat{r}_t=\frac{r_t}{1-\rho_2^t} \tag{9}$$

Finally, the updated formula of Adam is shown in equation (10).

$$\theta_{t+1}=\theta_t-\eta\frac{\hat{r}_1}{\sqrt{\hat{r}_2+\sigma}} \tag{10}$$

Where, η indicates the learning rate, σ is a small constant with a default value of 10^{-8}. The RNN receives the service component when users pick a service component. Subsequently, it computes the prediction services using the weights, then sends the top n predictions to the recommendation list and posts them to users. The matched services are suggested to the user by DNN, and to represent their interests, a Bayesian classifier is then used, which can be shown below.

4.1.2 Service Composition Recommendation Based on Naive Bayes

The information obtained is used to further decrease the service components selected by users, and to extract user interests based on the reduced service component set, the Naive Bayes classifier is then used. Finally, based on their preferences, similar service combinations are suggested to the user. Bayesian can quickly and efficiently identify the user's interest according to several service components clicked by the user, and those with similar interest in the user template library can directly match the common components clicked by the user.

Information Gain: User interests are determined based on the service composition; it is performed after the completion of service composition. The information gain algorithm is used to lower the service component set to lessen the interference of noncritical service components. Subsequently, it is possible to determine the gain value $IG(s)$ of each service component in the service composition. The service components are sorted by the gain value $IG(s)$, and the first n service components are regarded as the reduced service component set. The process of information gain is depicted in table 1.

Table 1: Process of Information Gain

1) The entropy of each service component SC_j in the service composition SC is calculated, which is $H\left(SC_j \mid SC\right)$.
2) The entropy without this service component SC_j in the service composition SC is calculated, which is $H\left(SC_j \mid \overline{SC}\right)$.
3) The classification gain value of this service component, which is, $IG\left(SC_j\right)$ is the difference between the entropy $H\left(SC_j \mid SC\right)$ and the entropy $H\left(SC_j \mid \overline{SC}\right)$, as stated in the following formula:
$IG\left(SC_j\right)=H\left(SC_j \mid SC\right)-H\left(SC_j \mid \overline{SC}\right)$ (11)
$IG\left(SC_j\right)=\sum_{i=1}^{n}P\left(c_i\right).\log_2 P\left(c_i\right)+\sum_{i=1}^{n}P\left(c_i \mid SC_j\right)\log_2 P\left(c_i \mid SC_j\right)+\sum_{i=1}^{n}P\left(c_i \mid \overline{SC_j}\right)\log_2 P\left(c_i \mid \overline{SC_j}\right)$ (12)
$P\left(c_i \mid SC_j\right)=\frac{n\left(SC_j \mid c_i\right)}{n\left(c_i\right)}$ (13)
(14)

The probability of the service component SC_j belonging to interest c_i is represented by $P(c_i \mid SC_j)$. $P(c_i \mid \overline{SC_j})$ indicates the probability that the service component SC_j does not belong to interest c_i. $n(SC_j \mid c_i)$ means the number of service compositions including SC_j in interest c_i. $n(\overline{SC_j} \mid c_i)$ means the number of service compositions excluding SC_j in interest c_i. $n(c_i)$ is the number of service compositions in interest c_i. $P(c_i)$ represents the proportion of services compositions belonging to interest c_i in all services compositions.
4) The first n service components are combined to form a reduced service component set and the service components are sorted by categorization gain value.

User Interest Modeling: The Naive Bayes classifier is used to determine the user interests based on the reduced service component set. The process of user interest modelling is specified in table 2.

Table 2: User Interest Modeling Process

1) The probability of the reduced service component set belonging to each interest category is calculated by the Naive Bayes classifier, as discussed in the information gain model, which is $P(c_i \mid SC)$. According to the Bayesian formula, $P(c_i \mid SC) = P(c_i \mid SC_1, SC_2, \ldots, SC_n) \alpha P(S$. Assuming that SC_1, $SC_2, \ldots,$ SC_n is independent, $P(SC_1, SC_2, \ldots, SC_n \mid c_i) = \prod_{j=1}^{n} P(SC_j \mid c_i)$. Consequently, as shown in formula (15), SC represents the sequence of the reduced service components $(SC_1, SC_2, \ldots, SC_n)$. $P(c_i \mid SC) = P(c_i \mid SC_1, SC_2, \ldots, SC_n) \alpha P(SC_1, SC_2, \ldots, SC_n \mid c_i) P(c_i)$ (15) $P(SC_1, SC_2, \ldots, SC_n \mid c_i) = \prod_{j=1}^{n} P(SC_j \mid c_i)$ (16)
2) According to formula (15), $P(c_i \mid SC) \alpha P(c_i) \prod_{j=1}^{n} P(SC_j \mid c_i)$. Subsequently, this paper selects the interest category with the highest probability as the user interest; therefore, formula (17) is feasible. $\arg\max P(c_i \mid SC) \alpha \arg\max \left[P(c_i) \prod_{j=1}^{n} P(SC_j \mid c_i) \right]$ (17)

The Naive Bayes classifier is used to determine the user interests based on the reduced service component set. Based on the similarities from high to low, the distance between different service compositions, and the service compositions are recommended to the user the N-gram distance is used for the computer.

4.2 Multi-Cloud Service Composition Problem

In today's world, cloud providers display their services in a variety of contexts using a variety of functional and non-functional features. A major problem in multi-cloud service composition is finding and selecting an appropriate atomic service from a pool of activated services. In the service composition problem, reducing the number of cloud providers is crucial, since it influences total cost, energy consumption, and response time. Consequently, this research work proposed a Hybrid Firefly and Bee Colony Optimization Algorithm to assess the service composition in multi-cloud environments decreasing the number of cloud providers to compose the best services based on uncertain QoS attributes. Subsequently, it considers the composition in both application level and user's preference in the cloud environment.

4.2.1 HFF-ABCOA

The FF is a population-based metaheuristic inspired by flashing fireflies' communication behaviour. Based on the idealised properties of flashing fireflies, its mathematical model has been proposed as follows: (1) All fireflies are unisex, (2) the attractiveness of a firefly is proportional to its brightness or light intensity, and (3) the landscape of the fitness function determined the brightness of a firefly.

Accordingly, based on the foraging behaviour of natural honey bee swarms, the ABC algorithm is inspired and it is a metaheuristic technique. Employed bees, observer bees, and scouts are the three types of bees are considered in a colony of artificial bees in the ABC. One-half of the population of artificial bees are employed bees, while the other half consists of onlookers and scouts.

The quantity of bees engaged is proportional to the number of food sources available (possible solutions). Employed bees are bees that are currently pursuing a food source and, in the meantime, with onlookers, these bees share their knowledge of food sources. From those discovered by employed bees, onlooker bees select high-quality food sources and then seek the foods in the immediate vicinity. Scout bees are employed bees who forsake their unpromising food sources in quest of fresh ones.

In addition, the diversity measure is used to help in the switch criteria. Consequently, the proposed HFF-ABCOA has three main components, the global search optimizer, the local search algorithm and the switch criteria. The details of each component and implementation steps of the HFF-ABCOA are presented as follows.

Global Search Optimizer: The global searcher in the proposed hybrid algorithm is a variation of FF that employs a random attraction model. In this FF variation, each solution is compared to a randomly chosen solution from the population's pool of promising solutions. Assume that the objective function values of all WP individuals in the population are sorted. Consequently, the first firefly x_1 is the best and the last WP firefly is the worst. Each solution x_i selects one solution from the set $set\left\{x_1, x_2, \ldots, x_{i-1}\right\}$ in the suggested random attraction model. Consequently, for each parameter value $x_{i,k}$, the update is determined by

$$x_{i,k}^{t+1} = x_{i,k}^{t} + \beta.\left(x_{j,k}^{t} - x_{i,k}^{t}\right) + \alpha\, S_k.\left(rand_i - \frac{1}{2}\right) \quad (18)$$

Where length scale is denoted as S_k for the kth variable, is a uniform random number in the range $(0,1)$ is represented as $rand_i$, t represents the generation number, $j \in \{1, 2, \ldots, i-1\}$ and $k = 1, 2, \ldots, D$. Subsequently, this simulation corroborated previous findings that if the randomization parameter α is connected to the magnitude of each variable, the FF search process can be improved. The scaling parameters S_k are calculated by

$$S_k = \left|u_k - l_k\right| \quad (19)$$

Where, l_k and u_k are the lower and upper bound of the parameter $x_{i,k}$. In the basic FF, the average number of updates of solution in each generation is $\frac{(WP-1)}{2}$. On the other hand, in the proposed FF variant, each solution can be updated at most once in each generation. The number of solution updates is much lower in the proposed FF variant than in the basic FF.

Local Search Algorithm: ABC performance mainly depends on its search equation given by equation (20). According to equation (20), the new candidate solution is created by moving the old solution to a randomly selected individual, and the search direction is completely random. Subsequently, the equation (20) is random enough for exploration and consequently can provide solutions with plenty of diversity and far from the actual solutions. In the employed phase, every solution x_i is updated by

$$v_{i,m} = x_{i,m} + \varphi.\left(x_{i,m} - x_{l,m}\right), \quad i = 1, 2, \ldots, WP/2 \quad (20)$$

Where, m is a randomly chosen parameter index, φ is a uniform random number in the range $(-1,1)$ and x_l represents the other solution selected randomly from the population. The update process is finalized after greedy selection is applied between x_i and v_i.

Combining search strategies that have different abilities so that they can complement each other during the search process can achieve better optimization results than a single search strategy as in the basic ABC. Subsequently, an ensemble of multiple solution search strategies for ABC is developed to perform a local search. Two search equations coexist throughout the search process in the proposed MABC and compete to develop better new solutions. The first is a basic ABC search technique is given in equation (20). The second search strategy employed in the MABC to generate a new candidate solution v_i by using the solution x_i is described by

$$v_{i,k} = x_{i,m} + \varphi_i.\left(x_{i,m} - x_{l,m}\right) \quad (21)$$

Where, m is a randomly chosen parameter index, φ_i is a random number in the range $(-1,1)$, x_l represents the other solution selected randomly from the population and $k = 1, 2, \ldots D$.

Table 3: Algorithm for Dynamic Regulation for Search Strategy

```
if $f(v_i) < f(x_i)$ then
    $x_i = v_i$;
    {solution $x_i$ is updated and its assigned
search strategy $S_i$ is kept for further search}
else
    if $S_i = S_1$ then
        $S_i = S_2$;
    else
        $S_i = S_1$;
    end if
    {solution $x_i$ is kept and its assigned search
strategy is replaced for further search}
    end if
```

In order to determine how to assign these search strategies to solutions from the population, an encoding method is used. Let denote the search strategy given by equation (20) as S_1 and the search strategy given by equation (21) as S_2. Each solution x_i is randomly assigned a search strategy, S_i, from the set $\{S_1, S_2\}$, at the beginning of the search. In the course of the search process, the value S_i is changed according to the quality of the new candidate solution v_i. If the candidate v_i has a lower objective function value than its parent x_i, it indicates that the current search equation is appropriate for the search. In that case, the current strategy is kept for further search and the parent solution is replaced with the candidate solution. Otherwise, it means that the current strategy cannot enhance the quality of the solution and it is replaced. Also, in that case, the parent solution is kept for the next generation.

Consequently, the search strategy algorithm is presented in table 3. The input of the algorithm includes the parent solution x_i and its assigned search strategy S_i, the candidate solution v_i and the objective function f. The output of the Algorithm is the solution x_i and its assigned search strategy S_i which will be used in the next generation.

Switch Criteria: Differences among individuals of a population are a prerequisite for exploration, but too much diversity in each phase of the search may lead to inefficient search. Population diversity is usually high at the beginning of a search process, and it decreases as the population moves towards the global optimum.

In the implementation of the HFF-ABCOA, it is important to know when to switch from the FF to MABC. For this purpose, the FFMABC incorporates the diversity metric as equation (22) to measure the population diversity.

$$Diversity = \frac{1}{WP}\sum_{i=1}^{SP}\sqrt{\frac{1}{D}\sum_{k=1}^{D}\left(x_{i,k} \quad x'_k\right)^2} \qquad (22)$$

Where, SP is a number of solutions in the population, D is the dimension of the problem and x'_k is the central position of the whole population. In order to determine when to switch the search to MABC, the population diversity during each generation is measured. Consequently, it indicates that the solution quality does not improve sufficiently quickly. In order to aid the proposed technique in avoiding stagnation, the search is moved to the MABC at that point.

4.3 Security of Multi-Cloud Web Service Composition

The service composition suffers from several limitations of security issues such as a lack of native support for encryption, decryption and authorization. In this work, a generalized framework is proposed for a secure multi-cloud service composition approach using Fuzzy Generalized Rough Test Theory. To guarantee a high-security level of the hosting clouds and the selected services, this work exploits the fuzzy relations with the rough set. To handle data with continuous qualities and find discrepancies in services, fuzzy-rough set theory can be combined with rough set theory. By excluding ineligible clouds and insecure services, this fuzzy-rough set approach will help minimise the search space.

In this work, rough set theory (RS) and fuzzy formal concept analysis (Fuzzy FCA) are used, both of which have been effectively employed in a variety of domains. Given the imprecision and uncertainty of security levels (i.e., the degree of satisfaction with security policies) not only at the cloud zone level but also at the service instance level, Fuzzy FCA is used. The principle of rough sets is used to approximate a user's request when the required services cannot be located or when the user's request is too complex.

4.3.1 Generalized Fuzzy

Generalized Formal Concept Analysis is an effective mathematical approach for data analysis, knowledge discovery, and information retrieval. Fuzzy is an extension of Formal Concept Analysis, where, the truth degree is a valued representation of the object/attribute relations provided by Fuzzy FCA. Instead of using binary relations, i.e., whether the object has the attribute or not, a membership value in [0, 1] is associated with each couple (object, attribute).

In this model, fuzzy FCA is used to explicitly represent the multi-cloud environment. Accordingly, it can be accomplished by excluding untrustworthy clouds and services (with low-security levels) and clustering clouds/services with similar security rules into tiny clusters. The most secure clouds will then be extracted using fuzzy FCA, and based on the user's QoS and security requirements, the safe services will be constructed.

4.3.2 Rough Set Theory

The rough set is a branch of set theory that is closer to fuzzy theory. In order to find specific information and perfect knowledge, this mathematical technique is used to gather objects with similar properties. In rough set theory, the investigated items are a finite and non-empty universe U. An equivalency relation $R \subseteq U \times U$ divides the universe U into two distinct sets via a partitioning method represented by U/R. In some cases, it would be impossible to precisely describe a set from the universe U using R. Consequently, the rough set provides two operations called lower approximation $\underline{QP}$ and upper approximation $\overline{QP}$, defined as follows:

$$\underline{QP} = \bigcup \{ X \in U/R \mid X \subseteq Q_P \} (23)$$

$$\overline{QP} = \bigcap \{ X \in U/R \mid Q_P \subseteq X \} (24)$$

Where, R is the family of definable subsets of characteristics retrieved from the universe U, as well as the set of concepts existent in the fuzzy context. While searching for appropriate things in a large area, its main goal is to deal with ambiguous and imprecise information.

4.3.3 Fuzzy Generalized Rough Test Theory

The generalized fuzzy is used to represent the multi-cloud environment and to exclude untrusted clouds/services in this work. Subsequently, in some cases, the user request may not be found in any of the existing formal concepts. Consequently, an approximate result must be returned to satisfy the user's security requirements. Consequently, it will be supplemented with a rough set theory when the fuzzy FCA cannot approach the requisite collection of policies.

In order to deal with uncertainty and poor knowledge, the combination of fuzzy FCA and rough set has garnered a lot of attention as two theories of fuzzy sets. The rough set's family of definable sets $\sigma(U/E)$ is replaced by a collection of fuzzy ideas that exist in a formal context. The concepts' Extents and Intents, replace the definable sets of objects and characteristics. When a group of objects or qualities does not form a concept, using the lower and upper approximations is a solution to identify the best or closest concepts that approach them. The motivation behind using fuzzy FCA is transforming the insecure MCE into a secure search space, by filtering the clouds which do not satisfy the required security policies. Consequently, fuzzy FCA is used to indicate how much each cloud can satisfy the user's security criteria using truth degrees. The MCE is modelled using a fuzzy formal context known as K MC, which represents the interactions between clouds and security regulations to achieve this purpose.

The multi-cloud fuzzy context can be thought of as a collection of fuzzy relationships, such as the following: $((C_1, P_2), 0.7)$; $((C_2, P_3), 1)$; $((C_1, P_4), 1)$; $((C_3, P_5), 0.9)$. Consequently, declare that the cloud C_1 completely respects the security policy P_4 because the object C_1 has the attribute P_4 with a membership property set to 1. When the memberships of the clouds C_2 C_3 are compared to the security policy, it can be seen that the cloud C_2 meets the security policy P_4 better. The latter may not be entirely satisfied by the cloud, even if a cloud C adopts a specific policy P. A security threshold is considered, which is often determined by security specialists, to preserve only trusted clouds. Subsequently, this will aid in the elimination of clouds whose memberships are less than a certain threshold. Using the fuzzy relation as an example, the membership value 0.5 associated (C_5, P_5) shows that if the user requires the security policy, C_5 it will be removed.

5. EXPERIMENTATION AND RESULT DISCUSSION

The proposed model uses a service composition dataset and service process call records from the ProgrammableWeb website to conduct experiments. The 20035 users' records are included in the service process call records. Subsequently, the QoS evaluation uses a real benchmark dataset called WSDream dataset using response time, availability, and throughput. The throughput (in bits per second) and response time (in seconds) of each web service are varied in this dataset based on the calls made by different users. Accordingly, this simulates the behaviour of the real web services and the changes in the QoS values during the service execution engine running.

Table 4: System Configuration for Simulation

Simulation System Configuration	
Python Jupiter	Version 3.8.0
Operation System	Ubuntu
Memory Capacity	4GB DDR3
Processor	Intel Core i5 @ 3.5GHz
Simulation Time	200 seconds

The proposed method is implemented using Python software. Subsequently, the simulation configuration of the proposed work is presented in table 4. The proposed method can be simulated and evaluated using the Python Jupiter Version 3.8.0. The operating system of the proposed work is Ubuntu, their memory capacity is 4GB DDR3, their processor is Intel Core i5 @ 3.5GHz and simulation time is 200 seconds, respectively.

Table 5: Performance Values of Proposed Method

Parameters	Values
Execution Time	64.48865294456482 s
Throughput	91.91169008883757
Efficiency	85.5181316424005
Feasibility	84.45221708169524
Recall	93.08449260399618
Precision	90.33969880499986

Table 5 illustrates the performance values of the proposed method, it demonstrates the execution time, throughput, efficiency, feasibility, recall, and precision. The execution time of the proposed method is 64.489 s, throughput is 91.912, and efficiency is 85.518. Subsequently, the feasibility of the proposed work is 84.452, recall is 93.084, and precision of the proposed work is 90.34, respectively. Consequently, the performance graph of the proposed method is represented in the following sub-section.

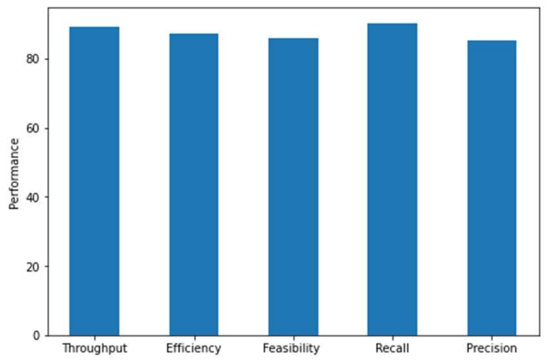

Figure 3: Performance Graph of Proposed Method

Figure 3 depicted the performance graph of the proposed method, it includes throughput, efficiency, feasibility, recall, and precision. The throughput of the proposed work is 91.912, and the efficiency of the proposed work is 85.518. The feasibility of the proposed work is 84.452, the recall of the research is 93.084, and the precision of the proposed work is 90.34, respectively.

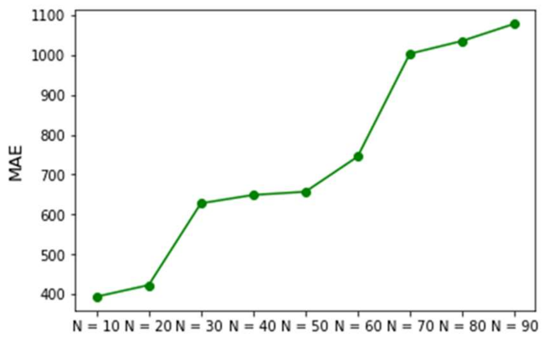

Figure 4: Mean Absolute Error of the Proposed Work

The mean absolute error of the proposed work is revealed in figure 4. The MAE of the proposed work is gradually increased when the number of services increases. The MAE reaches up to 1070 when the number of services is 90. The MAE is 375 when the number of services is 10 and the MAE is gradually increased to 1070, respectively.

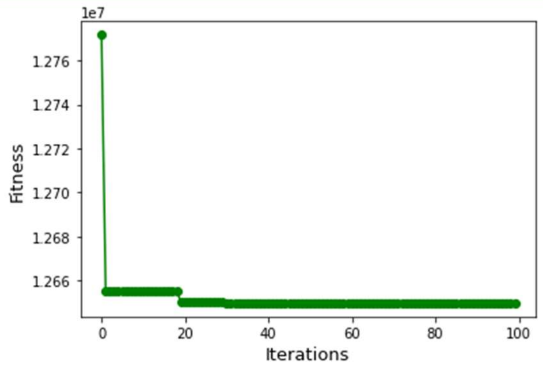

Figure 5: Fitness Function of the Proposed Method

Figure 5 illustrates the HFF-ABCOA fitness function value in all scenarios. In this work, the number of iterations increases, and the fitness value decreases. Based on the increasing number of clouds, the volume of search space grows; this is the reason that the amount of fitness value decreases in the last scenarios.

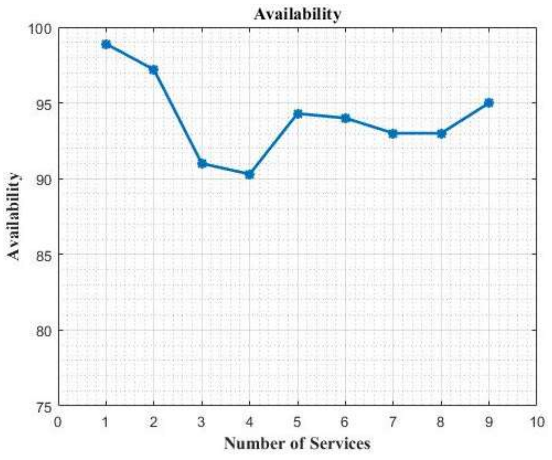

Figure 6: Availability of the Proposed Method

Figure 6 portrays the availability of the proposed method, it demonstrated the different number of services. When the number of services increases, the availability decreases, furthermore, the availability of the proposed method is 98.97%. The availability of the proposed method is reached 95% when the number of services increases.

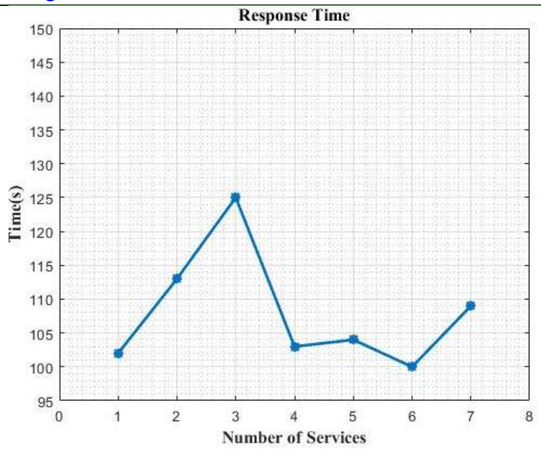

Figure 7: Response Time of the Proposed Method

Figure 7 discloses the response time of the proposed method; it demonstrates that the response time reaches 101.5 s. The proposed method has several services, when the number of services increases the response time of the proposed method also varies. When the number of services is 7, the response time of the proposed work is 104 s.

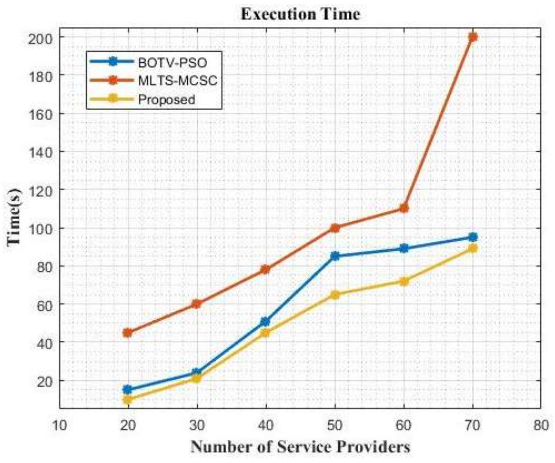

Figure 8: Comparison Graph of Execution Time

Figure 8 reveals the comparison graph of execution time. The proposed method is compared with the existing method like Bi-Objective Time-Varying Particle Swarm Optimisation (BOTV-PSO) Algorithm [16], and Multi-Labelled Transition Systems Multi-Cloud Service Composition (MLTS-MCSC) [18]. While compared to these existing methods, the proposed method has less value. Consequently, the proposed method produces the best performance than the other methods.

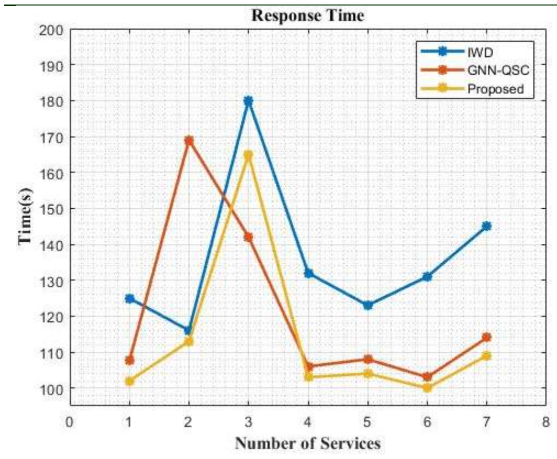

Figure 9: Comparison Graph for Response Time

Figure 9 portrays the comparison graph for response time. The proposed method is compared with the existing Intelligent Water Drops (IWD) algorithm [17] and Genetic Algorithms and Neural Networks for QoS-aware IoT Services Composition (GNN-QSC) [26] methods. The graph revealed the proposed method has a less response time, furthermore, the proposed method performs better than the other two methods.

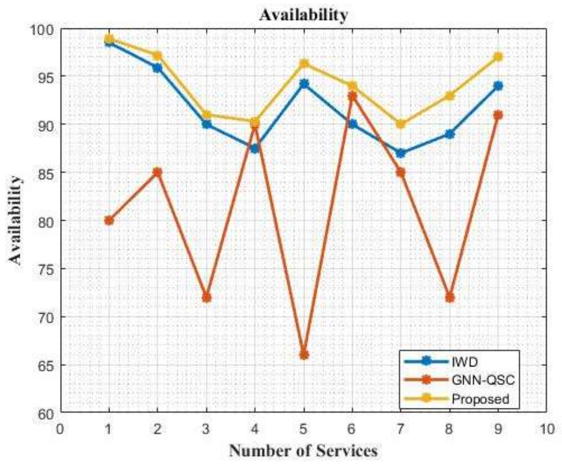

Figure 10: Comparison Graph for Availability

The availability of the proposed method is compared with the existing IWD and GNN-QSC methods, which are depicted in figure 10. The figure demonstrated that the availability of the proposed method is higher than the other existing methods. Consequently, this figure depicted that the proposed method has a higher performance than the other existing methods.

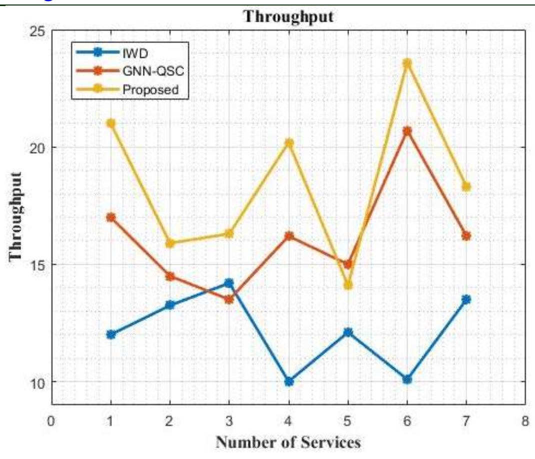

Figure 11: Comparison Graph for Throughput

The comparison graph for throughput is depicted in figure 11. The proposed method is compared with the existing IWD and GNN-QSC methods. The figure depicted that the proposed method produces higher performance than the other existing method because the throughput of the proposed method is a higher value than the other methods.

6. CONCLUSION

Cloud computing is an elastic service provisioning model that enables on-demand network access to a shared pool of computing resources. The popularity of cloud computing has increased in recent years. Each cybersecurity attack on security tenets can make a business financial loss or even fail. Today, users and enterprises are increasingly using the cloud to access software resources in the form of web services. Subsequently, web service composition is an important concept in recent years, however, a deep neural network is proposed in this work for web service composition recommendation. Based on the user's needs, the proposed method recommends the matched services. Accordingly, a hybrid firefly and bee colony optimization algorithm are presented in this article for multi-cloud composition problems, it provides the best service by reducing the cloud providers. Subsequently, a Fuzzy generalized rough set theory is introduced for the security of multi-cloud web service composition, and it eliminates the insecure services. Consequently, the proposed method is implemented in Python software.

- ➢ The performance metrics of the proposed method are throughput, response time, availability, feasibility, efficiency, etc.

- ➢ The performance of the proposed method is compared with the existing BOTV-PSO, MLTS-MCSC, IWD, and GNN-QSC methods.
- ➢ The execution time of the proposed method is 0.5% higher than the existing BOTV-PSO method and 3% higher than the MLTS-MCSC.
- ➢ Subsequently, the proposed method is 3% higher than the existing methods for response time, for availability, the proposed method is 5% higher than the existing methods and for throughput, the proposed method is 4% higher than the other methods.

Subsequently, this work performs best and user-interested web service composition recommendation. Accordingly, cloud services are offered as self-contained components that provide IT solutions for consumer requirements via the Internet; they are typically delivered by a Web service interface. They interact between them through the Internet. Consequently, it allows these objects to offer their functionalities in the form of services. Consequently, in future work, communication costs are minimized based on identifying affinity web services for co-hosting, as well as determining the minimum and maximum allowable used clouds. Additionally, in future, plan to quantify cloud reliability in terms of user business processes, respectively.

REFERENCES:

[1] G. Rodríguez, C. Mateos, and S. Misra, "Exploring web service QoS estimation for web service composition," in Communications in Computer and Information Science, *Cham: Springer International Publishing*, 2020, pp. 171–184.

[2] C. Hu, X. Wu, and B. Li, "A framework for trustworthy web service composition and optimization," *IEEE Access*, vol. 8, pp. 73508–73522, 2020

[3] S. Chattopadhyay and A. Banerjee, "QoS-aware automatic web service composition with multiple objectives," *ACM trans. web*, vol. 14, no. 3, pp. 1–38, 2020.

[4] Z. Wang, B. Cheng, W. Zhang, and J. Chen, "Q-graphplan: QoS-aware automaticservice composition with the extended planning graph," *IEEE Access*, vol. 8, pp. 8314–8323, 2020.

[5] Y. Song, Y. Wang, and D. Jin, "A Bayesian approach based on Bayes minimum risk decision for reliability assessment of web service composition," *Future internet*, vol. 12, no. 12, p. 221, 2020.

[6] D. Yu, L. Zhang, C. Liu, R. Zhou, and D. Xu, "Automatic Web service composition driven by keyword query," *World Wide Web*, vol. 23, no. 3, pp. 1665–1692, 2020.

[7] U. Arul and S. Prakash, "Toward automatic web service composition based on multilevel workflow orchestration and semantic web service discovery," *Int. J. Bus. Inf. Syst.*, vol. 34, no. 1, p. 128, 2020.

[8] M. Du, "An improved genetic algorithm for web service composition optimization," *World Scientific Research Journal*, no. 6, pp. 369–379, 2020.

[9] C. Wang, H. Ma, G. Chen, S. Hartmann, and J. Branke, "Robustness estimation and optimisation for semantic web service composition with stochastic service failures," *IEEE trans. emerg. top. comput. intell.*, vol. 6, no. 1, pp. 77–92, 2022.

[10] N. Kashyap, A. C. Kumari, and R. Chhikara, "Multi-objective Optimization using NSGA II for service composition in IoT," *Procedia Comput. Sci.*, vol. 167, pp. 1928–1933, 2020.

[11] B. Pang, F. Hao, D. S. Park, and C. D. Maio, "A multi-criteria multi-cloud service composition in mobile edge computing," *Sustainability*, vol. 12, no. 18, 2020.

[12] A. S. B. Priya and R. S. Bhuvaneswaran, "Retraction Note to: Cloud service recommendation system based on clustering trust measures in multi-cloud environment," *J. Ambient Intell. Humaniz. Comput.,* 2022.

[13] L. Wang, Z. Yang, and X. Song, "SHAMC: A Secure and highly available database system in multi-cloud environment," *Future Gener. Comput. Syst.,* vol. 105, pp. 873–883, 2020.

[14] A. Ramamurthy, S. Saurabh, M. Gharote, and S. Lodha, "Selection of cloud service providers for hosting web applications in a multi-cloud environment," *in 2020 IEEE International Conference on Services Computing (SCC)*, 2020.

[15] N. Xie, W. Tan, X. Zheng, L. Zhao, L. Huang, and Y. Sun, "An efficient two-phase approach for reliable collaboration-aware service composition in cloud manufacturing," *J. Ind. Inf. Integr.,* vol. 23, no. 100211, p. 100211, 2021.

[16] M. Hosseini Shirvani, "Bi-objective web service composition problem in multi-cloud environment: a bi-objective time-varying particle swarm optimisation algorithm," *J. Exp.*

Theor. Artif. Intell., vol. 33, no. 2, pp. 179–202, 2021.

[17] A. Barkat, O. Kazar, and I. Seddiki, "Framework for web service composition based on QoS in the multi cloud environment," *Int. J. Inf. Technol.*, vol. 13, no. 2, pp. 459–467, 2021.

[18] A. Souri, A. M. Rahmani, N. J. Navimipour, and R. Rezaei, "A hybrid formal verification approach for QoS-aware multi-cloud service composition," *Cluster Comput.,* vol. 23, no. 4, pp. 2453–2470, 2020.

[19] S. Haytamy and F. Omara, "Enhanced QoS-based service composition approach in multi-cloud environment," *in 2020 International Conference on Innovative Trends in Communication and Computer Engineering (ITCE)*, 2020.

[20] F. Lahmar and H. Mezni, "Security-aware multi-cloud service composition by exploiting rough sets and fuzzy FCA," *Soft Comput.,* vol. 25, no. 7, pp. 5173–5197, 2021.

[21] M. Heidari and S. Emadi, "Services Composition in Multi-Cloud Environments using the Skyline Service Algorithm," *International Journal of Engineering,* vol. 34, no. 1, pp. 56–65, 2021.

[22] H. Yin and Y. Hao, "An energy-aware multi-target service composition method in a multi-cloud environment," *IEEE Access*, vol. 8, pp. 196567–196577, 2020.

[23] R. Zeyneb Yasmina, H. Fethallah, and L. Fadoua, "Web service selection and composition based on uncertain quality of service," *Concurr. Comput.,* vol. 34, no. 1, 2022.

[24] F. Seghir, "FDMOABC: Fuzzy Discrete Multi-Objective Artificial Bee Colony approach for solving the non-deterministic QoS-driven web service composition problem," *Expert Syst. Appl.,* vol. 167, no. 114413, p. 114413, 2021.

[25] C. Wang, H. Ma, G. Chen, and S. Hartmann, "Memetic EDA-Based Approaches to QoS-Aware Fully-Automated Semantic Web Service Composition," *IEEE Transactions on Evolutionary Computation*, 2021.

[26] R. Boucetti, O. Hioual, and S. M. Hemam, "An approach based on genetic algorithms and neural networks for QoS-aware IoT services composition," *J. King Saud Univ. - Comput. Inf. Sci.*, 2022.