INTERNATIONAL JOURNAL OF SCIENTIFIC RESEARCH AND INNOVATIVE STUDIES
ISSN: 2820-7157
www.ijsrisjournal.com

June 2024 **Volume 3** **Number 3** **Pages 06-15**

Received Date: April 09, 2024 **Accepted Date:** May 07, 2024 **Published Date:** June 01, 2024

Available Online at https://www.ijsrisjournal.com/index.php/ojsfiles/article/view/150

https://doi.org/10.5281/zenodo.11217801

Efficient Service Composition Optimization under Uncertain Environment

REMACI Zeyneb Yasmina[1]
[1]Department of Computer Science, University of Abou Bekr Belkaid Tlemcen, Algeria.
zeynebyasmina.remaci@univ-tlemcen.dz

ABSTRACT

Over the past few years, companies have been offring a variety of functionalities through Web services as part to stay competitive. Meanwhile, the customers may face challenges in choosing a service that meets their requirements due to the similarity of functionalities. Moreover, the customers' requests often emerge with a complex nature in an uncertain environment that is rarely fulfilled by an atomic service. Hence, there is a necessity for optimizing service composition based on Quality of Service (QoS) with taking into account environmental uncertainty. To attain the mentioned goal, our proposed approach adopts a local and global optimization strategies. Firstly, we adopt a heuristic based on the Entropy, Cross-Entropy, and the deviation degree for the hesitant fuzzy set to rank similar Web services. Afterwards, we introduce an improved metaheuristic called Group Leaning based Composition as a global optimization for selecting the near-optimal composition.

Key words : Deviation degree, Global Optimization, Group learning metaheuristic, Hesitant fuzzy set, Local Optimization, Service Composition, Quality of Service, Web Service.

1. INTRODUCTION

Web services becomes a pillar in providing several solutions in different vital areas, such as Electronic Health, e-government, scientific workflows, and Internet of Things environments (e.g., smart cities). This proliferation is attributed to the collaboration between the modular services that occur without requiring knowledge of the internal implementation details. The collaboration results in a combination commonly referred as the service composition. In fact, the service composition based on QoS is considered as a Single-Objective or a Multi-Objective Optimization [11]. This optimization is recognized as NP hard problem, Attracting the attention of many recent researches [6],[12],[17],[18],[19],[25]. The primary challenge is the large number of concurrent services that offer similar functionalities. On the other hand, the composition must respect the user requirement, called the global constraints. Consequently, the composition process depends on the nonfunctional features known as Quality of Service. For instance, a user can request for an online product purchase with specific requirement for cost, response time, and fidelity. This request can be a combination of multiple services such as the research of products, the currency exchange service, and the payment services. Each one can be proposed by multiple concurrent providers with different QoS. In this case, the QoS is a fundamental key to select the most relevant services, forming a composition set, and determining the pertinent composition that will satisfy the user's constraints. In the current state of the art, although various approaches and frameworks have been employed to tackle these challenges, we observe that the most of them treat QoS as static values that impact the both service selection and the resulting composition. Meanwhile, this process occurs in a fluctuating environment, influenced by factors such as involving network conditions, work-loads, etc… These factors impact on QoS throughout the process, emphasizing the inherent uncertainty and variability of QoS. For instance, the response time of

delivery service changes based on some factors like server load or network traffic.

Therefore, in this paper we aim to address three aspects. First, inspired by [30], we manage the nondeterministic QoS during the composition process by leveraging the hesitant fuzzy set. Then, we enhance the heuristic proposed by [30] by incorporating Cross-entropy for hesitant fuzzy set to minimize the search space by adding the deviation degree. Finally, we introduce the Group Leaning-based Composition (GLC) to select the service composition that respect the user's requirements. Furthermore, we utilize Global Quality Conformance (GQC) [15] (see Equation 1) as utility function to handle the nondeterministic QoS.

1.1 Contribution

Our contribution can be summarized in threefold:

1. We compute the weight of each attribute based on the Entropy. Furthermore, we rank the services, then we retrieve the TOP-K services according to Cross-Entropy. In this step, the QoS attributes are considered as hesitant fuzzy elements [26],[28].
2. We propose Group Learning-based Composition GLC) to select the near-optimal composition using the metaheuristic called GLA.
3. We perform a set of experiments that show the effectiveness and efficiency of the proposed approach.

The remainder of this paper is organized as follow: Section 2 provides an overview of related works. In Section 3, introduces notations and concepts, while Section 4 outlines the proposed framework. The conclusion is presented in Section 5.

2. STATE OF THE ART

The QoS-based service selection and/ or composition approaches has been addressed by multiple researches. Meanwhile, the uncertain environment is often overlooked and the QoS attributes are considered as static values.

2.1 Web service composition based on certain QoS

In [4], the authors handled web service composition in a multi-cloud environment through a two-step process. Firstly, they employed a cloud combiner to select an optimal combination from the available multi-cloud providers. Then, they leveraged an Intelligent Water Drops (IWD) algorithm, which considers quality of service (QoS) parameters, for the service composition.

The authors in [8] proposed an improved genetic algorithm (IGA)for web service composition optimization based on QoS and user needs. In this approach, the lexicographic optimization method (LOP) and the threshold constraint relaxation technique (TCR) are employed to eliminate the concurrent web service with lower QoS. Furthermore, the web service composition problem is transformed into a single-objective optimization problem, and an elitist genetic algorithm is adopted search global optimal solution satisfying the demand of user.

The work by [29] optimized the Web service composition by introducing an enhanced Firefly algorithm. This approach integrates the principle of selection, exchange, and mutation in the genetic algorithm into the traditional firefly algorithm, resulting in a population solution that approaches the optimal solution. The approach proposed in [22], handle the Service composition mechanism in the Internet of Things based on QoS. This approach is based on a hidden Markov model (HMM) and an ant colony optimization (ACO). Furthermore, the emission and transition matrices have been improved using the Viterbi algorithm. The work by [3] proposed a service composition approach for the multi-cloud environment. The approach is based on the ant colony optimization algorithm (ACO) leveraging the multi-pheromone mechanism to optimize the quality of service (QoS). Further, the mutation operation of the genetic algorithm is utilized to prevent the slow convergence speed and the occurrence of local optima.

A new hybrid algorithm was proposed in [13], based on the Artificial Bee Colony (ABC) and Cuckoo Search (CS). The Cuckoo Search is used to overcome the ABC's limitations such as the slow convergence problem. The work in [14] deals with Web Service Composition (WSC) problem by leveraging an improved version of the Bat Algorithm (BA) by addressing certain limitations, such as the trade-off among exploration and exploitation searching mechanisms. Firstly, a population initialization method is proposed to avoid the bad initialization that leads to premature convergence. Furthermore, an elitist method is leveraged to achieve a better convergence rate. Finally, the neighborhood method is utilized to achieve a better tradeoff between search mechanisms.

The work by [30] introduces a Multi-Criteria Decision Making (MCDM) approach that integrates Best Worst Method (BWM) to determine the weights of criteria and relative scores for alternatives. Furthermore, the authors leverage Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) to rank cloud services.

In [14], the authors proposed a hybrid approach that merges the Artificial Neural Network (ANN) with the meta-heuristic PSO algorithm for a service composition model in Cloud-edge computing. Furthermore, they introduced a formal verification method based on a labelled transition system to assess Linear Temporal Logics (LTL) formulas, aiming to enhance the proposed ANN-PSO and ensure its correctness. This approach focuses on managing three Quality of Service (QoS) attributes: response time, availability, and prices.

2.2 Web service composition based on uncertain QoS

The service composition in a multi-Cloud environment is tackled in [13]. The approach revolves around a modified PSO designed to compose the best services based on the uncertainty of the QoS attributes.

In the [24], the authors improve the artificial bee colony algorithm (ABC) by using a fuzzy distance and ranking methods to enhance the artificial bee colony algorithm diversity (FDMOABC). Subsequently, they presented a fuzzy multi-criteria decision-making method (FMCDMM) to identify the optimal composite service from the Pareto-optimal solutions generated by FDMOABC. The work by [23] leveraged an interval-based multi-objective artificial bee

colony algorithm (IM_ABC) to deal with the uncertain QoS-aware service-composition problem QWSCP. Firstly, the authors compared the services using interval-oriented dominance relationship, where the intervals represent the variation range of the QoS attributes. Then, they leveraged an interval-valued utility function to assess the uncertain quality of a composition. Finally, the authors employed an improved version of NSGA-II to derive the non-dominated service compositions. Multiple probabilistic heuristics and fuzzy formulas was leveraged in [11] to address the service selection in a fluctuating environment. Then, based on the Top-K services, a constraint programming approach is employed to select the Top-K optimal compositions of the services. Likewise, the authors in [9] proposed four ranking heuristics that perform a pairwise comparison of services and select the most pertinent ones. Then, based on the swarm-intelligence-based algorithm called discrete bat algorithm (DBA) they performed the selection of the near-optimal composition.

In [27], the authors leverage a new approach called the triangular fuzzy genetic algorithm (TGA), where the triangular fuzzy set theory (FST) is combined with genetic algorithm (GA) to solve the uncertain QWSCP. To deal with the uncertain QoS, the authors in [18] proposed two approaches based on the definition of intuitionistic fuzzy logic. The first approach characterizes the uncertainty based on the definition of the degrees of membership, non-membership and uncertainty of the intuitionistic fuzzy pairs. Meanwhile, the second approach employs the intuitionistic fuzzy estimations of the embedded devices. It is worth noting that these approaches are proposed to be applied to all types of service systems. In [1], the composition of the IoT services based on the QoS is addressed. The authors proposed a new hybrid algorithm based on the Artificial Bee Colony (ABC) for its f few control parameters and easy implementation and Bat Algorithm (BA) to avoid ABC's slow convergence limitation.

[5] proposed an approach that reduces the search space, using the clusters of web services into nonoverlapping groups based on their input(s) and output(s). Then, two heuristics are proposed, the first one employs a beam search strategy to solve the multiple QoS aware problem using scalarization and find Pareto optimal solutions. The second approach is based on the nondeterministic sorting genetic algorithm (NSGA) to solve multiple QoS aware optimization problems. In [32], the authors proposed a two-stage approach, where the first stage reduces the search space by selecting the Top-K Web services according to the Majority Judgment (MJ) heuristic. The second stage leverages the Constraint Programming (CP) heuristic to eliminate the compositions that violate the global constraints. The selected Top K compositions must optimize the Global QoS Conformance objective function [15].

3. PROBLEM STATEMENT

In the following sections, we introduce the concepts and notations leveraged to manage the web service compositions with dealing the fluctuation of QoS.

3.1 Notation

To deal with the service composition problem, we consider that the user's request represents a workflow of n tasks, each task CL_j contains a set of services S_i. These services are designated by r criteria, representing QoS. These criteria can be positive (e.g., reputation) and negative (e.g., cost). Given the uncertain nature of the environment, each attribute criterion is represented by a set of instances. The notations and concepts are detailed in Table 1.

Furthermore, in this paper, we consider multiple workflows and both positive and negative QoS attributes, including response time, reliability, availability. The positive attributes must be maximized (such as reliability and availability), while negative attributes must be minimized (such as response time). The aggregation functions are mechanism used to compute the overall QoS of the composition based on a set of functions. We leverage the aggregation functions proposed in [15] (see Table 2).

Table 1: Notations [32]

Notations	Meaning
n	The number of tasks (abstract classes).
m	The number of services per task.
r	The number of QoS attributes.
l	The number of service instances (i.e., the number of QoS realizations or the sample size).
CL_1, CL_2, ..,CL_n	The set of abstract classes, each class contains atomic Web services with the same functionality and different QoS
S_{j1}, $S_{j,..}$, S_{jm}	Represents the service i related to CL_j
QoS_{piju}	The value of the p^{th} QoS attribute related to the u^{th} instance of the service $S_i \in CL_j$
b_1, b_2, ..,b_r	The user's global constraints (i.e., the bounds that must be fulfilled by the QoS of the composition).
w_1, w_2, ..,w_r	The weight of the QoS criteria, the default value of each w_p is 1/r.
K	The size of the returned list (of compositions).

Table 2: QoS aggregation functions [15]

Function	Attribute		
	Response time	Reliability	Fidelity
Sequential	$\sum_{j=1}^{n} q(Sji)$	$\prod_{j=1}^{n} q(Sji)$	$1/n \sum_{j=1}^{n} q(Sji)$
Loop	$\sum_{j=1}^{n} q(Sji)$	$\prod_{j=1}^{n} q(Sji)$	$q(Sji)$
Parallel	$\max_{j=1}^{n} q(Sji)$	$\prod_{j=1}^{n} q(Sji)$	$1/n \sum_{j=1}^{n} q(Sji)$
Conditional	$\max_{j=1}^{n} q(Sji)$	$\min_{j=1}^{n} q(Sji)$	$\min_{j=1}^{n} q(Sji)$

3.2 Utility functions

In this work, we integrate the user's request, termed global constraints. Therefore, we utilize the utility function known Global QoS Conformance (GQC) to select the near-optimal composition. In fact, GQC is the probability that the

composite service fulfills these global constraints. Since our focus is on satisfying of r global constraints simultaneously, we calculate the product of the r probabilities associated with each constraint. Each individual probability (see Equation 2) Pr((Sw1,..,Swn),bp), denotes the chance that the composition C =(S_{w1},..,S_{wn}) has an aggregated Quality of Service (QoS_p) greater than or equal to the user's bound b_p in the case where (QoS_p) is a positive criterion). In case where two compositions ties, we rank them according the utility function (U) (see Equation 6), where a higher score indicates a superior ranking.

$$GQC\ ((S_{w1},..,S_{wn}),(b_1,..,b_r)) = \prod_{q=1}^{r}\left((S_{w1},..,S_{wn}),b_q\right) \quad (1)$$

$$\Pr\left(\left(S_{w1},..,S_{wn}\right),b_q\right) = \frac{1}{l^n} * \sum_{u_1=1}^{l} \dots \sum_{u_n=1}^{l} Step\left(Aggregate\left(S_{w_1u_1q},..,S_{w_nu_nq}\right)\right) \quad (2)$$

$$\text{Step}(\text{Aggregate}\left(\text{QoS}_{\text{ps}_{w_1u_1}},\dots,\text{QoS}_{\text{ps}_{w_nu_n}}\right),\text{b}_\text{p}) = \begin{cases} 1 \text{ if Aggregate}\left(\text{QoS}_{\text{ps}_{w_1u_1}},\dots,\text{QoS}_{\text{p}_{w_nu_n}}\right) \geq \text{b}_\text{p} \\ \text{and the criterion } p \text{ is positive} \\ 1 \text{ if Aggregate}\left(\text{QoS}_{\text{ps}_{w_1u_1}},\dots,\text{QoS}_{\text{p}_{w_nu_n}}\right) \leq \text{b}_\text{p} \\ \text{and the criterion } p \text{ is negative} \\ 0 \text{ otherwise.} \end{cases} \quad (3)$$

$$U_{positive}\ (C) = \sum_{p=1}^{r} w_p * \frac{MedianQ'_p(C) - Qmin'(p)}{Qmax'(p) - Qmin'(p)} \quad (4)$$

$$U_{negative}\ (C) = \sum_{p=1}^{r} w_p * \frac{Qmax'(p) - MedianQ'_p(C)}{Qmax'(p) - Qmin'(p)} \quad (5)$$

$$U\ (C) = U_{positive}\ (C) + U_{positive}\ (C) \quad (6)$$

Where:

$$MedianQ'_p(C) = \sum_{j=1}^{n} Median_{u\in\{1..l\}} QoS_{pS_{ji}ju} \quad (7)$$

If we deal with a positive criterion p:

$$MedianQ'_p \geq b_p \quad (8)$$

If we deal with a negative criterion p:

$$MedianQ'_p \geq b_p \quad (9)$$

4. COMPOSITION FRAMEWORK

Figure 1, presents our proposed framework that supports the user to find the best service composition. This framework is based on three modules:

1. **Class management module:** This module categorizes each service into corresponding abstract classes, ensuring regular updates for the addition of new services and the removal of unavailable components.

2. **QoS management module:** The QoS is a nondeterministic value which mean for each attribute we have a set of realized. These values are collected from the service providers and social network, then stored in a data warehouse.

3. **Optimization Module for Service Composition:** This module is used for the service composition optimization. It is based on the user requirements (Global constraints) The QoS data can be drawn from the service provider itself (e.g., cost), the social networks (e.g., ratings, reputations), and the third parties (e.g., throughput, latency). and the proposed workflow. The optimization process involves two sub-modules: the first sub-module is used to select the most relevant web service, also called local optimization based on algorithm 1; the second sub-module handles the global optimization or the selection for the near-optimal composition.

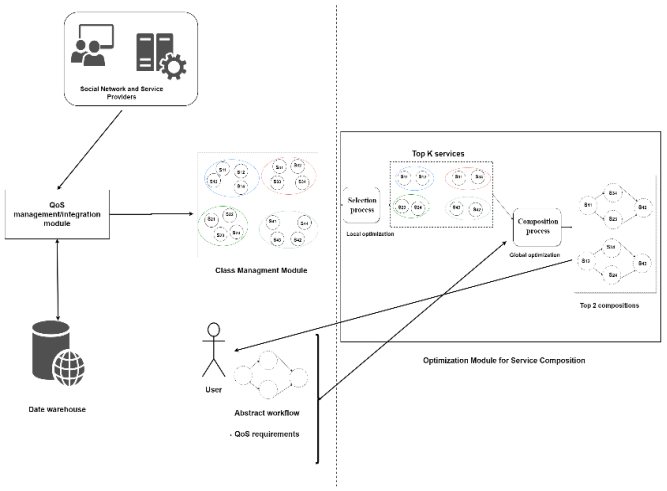

Figure 1: Proposed framework.

4.1 Hesitant Fuzzy Set (HFS) for a local optimization

Addressing real-world problems has consistently been an interesting area of research. Therefore, many theories, applications, and methods related to fuzzy sets have been proposed and applied to address these problems. Among these research efforts, [26] introduced an extension of the fuzzy sets, called Hesitant fuzzy set (HFS), designed to overcome the drawback of the traditional Fuzzy set extensions when dealing with the uncertain information. Indeed, Hesitant Fuzzy Set (HFS) denotes a set of possible values in the range [0,1] rather than of singular membership degree. Inspired by [30], we leverage a heuristic based on the hesitant fuzzy set to represent QoS realizations of Web services. In this heuristic, the closeness degree is computed to rank the services within the

same task. Furthermore, we integrate the deviation degree to address cases where the closeness degrees are equal.

In the following, we present the main concepts used in this heuristic.

Definition 1: Let X be a fixed set, a HFS on X defined in terms of a function that is when applied to X returns a subset of [0,1]. The HFS is represented mathematically by [Xia and Xu, 2011] as:

$$A = \{< x, h_A(x) | x \in X\} \quad (10)$$

The range of the set $h_A(x)$ is [0,1], denoting the possible membership degrees of the element x ∈ X with respect to the set.

Definition 2: (Chen et al. 2013a). For a HFE h , we define the deviation degree $\underset{\sigma}{\rightarrow}$of h as follows:

$$\underset{\sigma}{\rightarrow} = \left[\frac{1}{l_h}\sum_{y\in h}(y - s(h))^2\right]^{\frac{1}{2}} \quad (11)$$

Where s(h) is the score of h:

$$s(h) = {}^{1}/_{l} * \sum_{y\in h} y \quad (12)$$

4.1 QoS Normalization

Handling multiple QoS attributes within the selection process presents a challenge, as each attribute has diverse measurement units and magnitudes [16]. Consequently, it is crucial to normalize the QoS values to a uniform range during the process. We employ the normalization function outlined in Equation 4.18 to map the QoS values into a standardized range from 0 to 1.

$$N_{QoS_{ps_ijp}} = \frac{\mathrm{QoS}_{\mathrm{ps}_i ju} - Qmin(p)}{Qmax(p) - Qmin(p)} \quad (13)$$

$$Qmin(j,p) = MIN_{s_i\in CL_j\ ,u\in 1,..,l}\ (QoS_{ps_iju}) \quad (14)$$

$$Qmin(j,p) = MIN_{s_i\in CL_j\ ,u\in 1,..,l}\ (QoS_{ps_iju}) \quad (15)$$

4.2 Entropy and Cross-Entropy for Hesitant fuzzy set

Actually, the Entropy is leveraged to compute the weights of criteria. However, Cross-Entropy is applied to compute the divergence between two probability distributions. Hence, we leverage the Cross-Entropy for computing divergence between QoS realizations and best and/or worst solutions. According to [Xu and Xia, 2012], Cross-Entropy for hesitant fuzzy set is defined as follow: Let $h_{S_{j1}}(p)$ and $h_{S_{j2}}(p)$ be two HFEs of the services S_{j1} and S_{j2} respectively, then the Cross Entropy C($h_{S_{j1}}(p)$, $h_{S_{j2}}(p)$) is specified as:

$$\begin{aligned} C\left(h_{s_{j1}}(p), h_{s_{j2}}(p)\right) = {} & \frac{1}{lT_0}\sum_{u=1}^{l} \\ & \left(\frac{\left(1+tQoS_{ps_1ju}\right)ln\left(1+tQoS_{ps_1ju}\right)+\left(1+tQoS_{ps_2ju}\right)ln\left(1+tQoS_{ps_2ju}\right)}{2} - \right. \\ & \frac{2+tQoS_{ps_1ju}+tQoS_{ps_2ju}}{2} ln\left(\frac{2+tQoS_{ps_1ju}+tQoS_{ps_2ju}}{2}\right) + \\ & \frac{\left(1+t\left(1-QoS_{ps_1j(l-u+1)}\right)\right)ln\left(1+t\left(1-QoS_{ps_1j(l-u+1)}\right)\right)}{2} + \\ & \frac{\left(1+t\left(1-QoS_{ps_2j(l-u+1)}\right)\right)ln\left(1+t\left(1-QoS_{ps_2j(l-u+1)}\right)\right)}{2} - \\ & \left(\frac{2+t\left(1-QoS_{ps_1j(l-u+1)}+1-QoS_{ps_2j(l-u+1)}\right)}{2}\right. \\ & \left.\left.\ln\left(\frac{2+t\left(1-\mathrm{QoS}_{\mathrm{ps_1j(l-u+1)}}+1-\mathrm{QoS}_{\mathrm{ps_2j(l-u+1)}}\right)}{2}\right)\right)\right) \end{aligned} \quad (16)$$

Where:

$$\mathrm{T}_0 = (1+\mathrm{t})\ln(1+\mathrm{t}) - (2+\mathrm{t})(\ln(2+\mathrm{t}) - \ln(2)), \mathrm{t} \geq 0. \quad (17)$$

According to [Xu and Xia, 2012] the entropy (or disorder quantity) of a fuzzy hesitant set is defined as follows: Let h be a HFE, then entropy is specified as:

$$\begin{aligned} \mathrm{E}_{\mathrm{sjiq}}(\mathrm{h}) = {} & 1 - \frac{2}{\mathrm{lT}_0}\sum_{\mathrm{u}=1}^{\mathrm{l}} \\ & \left(\frac{\left(1+\mathrm{tQoS}_{\mathrm{ps_1ju}}\right)\ln\left(1+\mathrm{tQoS}_{\mathrm{ps_1ju}}\right)+\left(1+\mathrm{t}\left(1-\mathrm{QoS}_{\mathrm{ps_1j(l-u+1)}}\right)\right)\ln\left(1+\mathrm{t}\left(1-\mathrm{QoS}_{\mathrm{ps_1j(l-u+1)}}\right)\right)}{2}\right. \\ & \left. - \frac{2+\mathrm{tQoS}_{\mathrm{ps_1ju}}+\mathrm{t}\left(1-\mathrm{QoS}_{\mathrm{ps_1j(l-u+1)}}\right)}{2}\ln\frac{2+\mathrm{tQoS}_{\mathrm{ps_1ju}}+\mathrm{t}\left(1-\mathrm{QoS}_{\mathrm{ps_1j(l-u+1)}}\right)}{2}\right) \end{aligned} \quad (18)$$

4.1 Model of Entropy weights

In general, resolving the service selection problem necessitates an understanding of Quality of Service (QoS) and its significance in terms of QoS weights. Various approaches require users to specify their QoS preferences. Nevertheless, these methods are impractical as users often lack the necessary knowledge to assign weights to each QoS attribute. Therefore, we adopt Entropy theory [27] to determine the QoS weights. The higher the weight, the more important the QoS criterion is.

$$\mathrm{w}'_{\mathrm{p}} = \frac{1 - \mathrm{E}_{\mathrm{p}}}{\mathrm{m} - \sum_{\mathrm{p}=1}^{\mathrm{r}} \mathrm{E}_{\mathrm{p}}} \quad (19)$$

$$\mathrm{E}_{\mathrm{p}} = {}^{1}/_{m}\sum_{\mathrm{i}=1}^{\mathrm{m}} \mathrm{E}_{\mathrm{h_{ip}}}, \mathrm{p} = 1,2,\ldots,\mathrm{r}. \quad (20)$$

$$\mathrm{E}_{\mathrm{h_{ip}}} = 1 - \mathrm{C}\left(\mathrm{h_{ip}}, \mathrm{h}^{\mathrm{c}}_{\mathrm{ip}}\right) \quad (21)$$

5. PROPOSED APPROACH

Our proposed approach is based on two algorithms. First, the 'ServiceLocalOptimization' algorithm (SLO) enables us to rank the Web services. This algorithm is inspired from EntropyServiceRanking (ESR) proposed in [30]. In SLO we utilize the deviation degree during the ranking process to tackle the situations where services are equal. Then, we employ the algorithm called the Group Learning-based

Composition (GLC), depicted in algorithm 2, to select the TopK compositions.

5.1 Local optimization
The algorithm 1 (SLO) is presented below.

Algorithm 1 ServiceLocalOptimization

Input: < CL1, ..., CLn >,
h_p^+= 1 // positive ideal solution,
h_p^- = 0 // negative ideal solution

Output: RankedCL$_1$, ..., RankedCL$_n$

Begin

1: **for** p=l to r **do**
2: w_p= compute-weight() // The weight is computed according to Equation (14)
3: **end for**
4: **for** j=l to n **do**
5: **for** i=l to m **do**
6: $c^+(S_{ji}) = \sum_{p=1}^{r}(w_p * C(h_{S_{ji}}(p), h_p^+)$
7: $c^-(S_{ji}) = \sum_{p=1}^{r}(w_p * C(h_{S_{ji}}(p), h_p^-)$

8 $C(S_{ji}) = \frac{c^+(s_{ji})}{c^-(s_{ji}) + c^-(s_{ji})}$

9: **end for**
10: RankedCL$_j$=ascending-sort (CL$_j$,C (S_{ji}), $\overrightarrow{\sigma}$)
11: **end for**
12: **return** <RankedCL$_1$, ..., RankedCL$_n$>
13: **end**

- We start by leveraging Equation 19 to compute the weight $w_p$of each attribute p (line 1,3).
- We employ the Equation 16 compute the Cross-Entropy between the service $S_{ji}$and h_p^+ that represent the positive ideal solution (resp. the negative-ideal solution h_p^-) (line 4 to 5).
- We compute the closeness degree between the service S_{ji} and the ideal solution (line 8).
- We rank the services of each task based on c(S_{ji}) in ascending order. The smaller the value of c(S_{ji}), the better the rank. If two services ties in terms of c(S_{ji}) we rank them according the deviation degree see Equation 11 (line 10).
- We return the ranked classes in line 12.

5.2 Group Learning based Composition
After completing the first process and retrieving the best services and minimizing the search space, the second sub-module is executed to perform the composition process by forming the compositions according to workflow and selecting the best one based on the user requests. Therefore, we propose algorithm 2 called Group Learning-based Composition (GLC). This algorithm is based on the Group Learning Algorithm (GLA) proposed by [20].
GLA is a recent metaheuristic implemented based on the interactions of individuals within a group and the influence of group leader on group members. This algorithm mimics the impact of group leaders and managers on the ability of group members. This algorithm divides the population into a number of equal groups and selects certain individuals as the group leaders based on their fitness. Additionally, it identifies the manager, who is the individual with the best fitness in the whole population.
This population-based metaheuristic effectively addresses the limitations of multiple metaheuristics. It is characterized by its flexibility and simplicity of implementation, requiring only a few control parameters. This algorithm proposes a great exploration and exploitation and achieves a fine balance between the two, as these parameters are adjusted by the number of groups.
In [20], the researchers modeled the influence of the group manager on the group leader by the Equation 22.

$$LA = (x - y) * r. \quad (22)$$

LA represents the updated group leader after being influenced by the manager. The variable x denotes the group leader, which is the individual with the highest fitness within the group, while y refers to the manager, which is the individual with the highest fitness across the entire population. Additionally, r represents a random number generated within the range [0, 1].
Furthermore, the influence of the group leader on the population is represented mathematically by Equation 23.

$$New\_pop(i) = (LA - pop(i)) * r$$

Where $new\_Pop\ (i)$ represents the updated individual after it is affected by the LA found in Equation 22, Moreover, the external impact is represented in this algorithm by the mutation operation (see Figure 2).
In our proposed algorithm, we have enhanced Equations of GLA to deal discrete optimization problems in the following manner:

XL_g= Find_Service($\lfloor (Position(XL_g) - Position(XM)) * r \rfloor$) (24)

In this context, XL_g represents the updated service leader in the group g, influenced by the manager XM. The variable r is a random number generated within the range [0, 1]. The function Position () is employed to identify the location of service X, while the Find_Service() function is utilized to retrieve the service based on its position (see Equation 24).

New_X_{gi} = Find_Service($\lfloor (Position(XL_g) - Position(X_i)) * r \rfloor$) (25)

On the other hand, New_X_{gi} is the new service individual i of the group g after it is affected by the new service leader XL_g.

New_indiv_X_{gi}= = Find_Service($\lfloor Position(\text{New_}X_{gi}) - Position(XM)) * r \rfloor$) (26)

Where New_indiv_X_{gi} is the new service individual I within the group g after it is affected by the service manager XM.

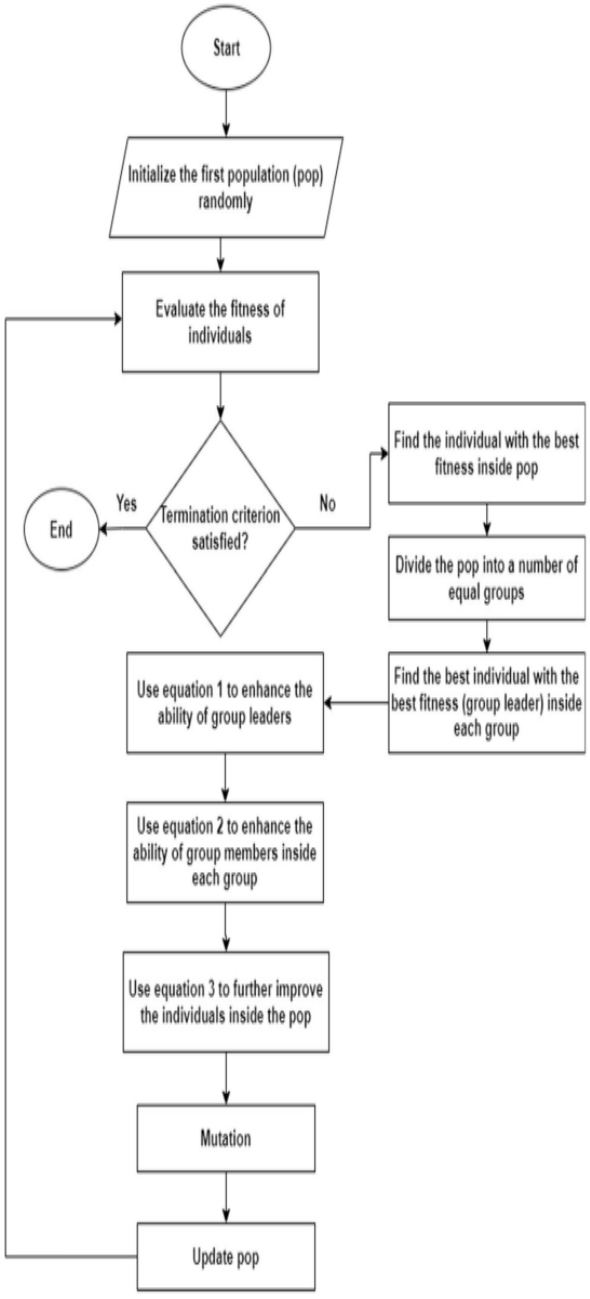

Figure 2: GLA [20]

***Algorithm** 2* GLC

Input: RankedCL1, ..., RankedCLn , k , MaxIter , PopSize, mutation_rate, group_number.
Output: BestComposition
Begin
1: **for** i=1 to PopSize **do**
2: X_i= init(< RankedCl1, ..., RankedCln >, k) // Initialize_population
3: $FitnessX_i$ = ComputeGQC(X_i);
4: **end for**
5: XM = find_manager(X_i, $FitnessX_i$)
6: XG_g = Divide_population (X_i , group_number)
7: **for** g=1 to group_number **do**
8: XL_g= finde_leader(XG_g g, $FitnessX_g$)
9: **end for**
10: **while** (t ≤ MaxIter) **do**
11: **for** g=1 to group_number **do**
12: XL_g = Update1 (XL_g , XM)
13: **for** i=1 to PopSize **do**
14: New_X_{gi} = Update2 (XL_g , X_{gi})
15: New_indiv_X_{gi} = Update3 (XM , New_X_{gi})
16: Mutation (New_indiv_X_{gi} , mutation_rate)
17: **end for**
18: **end for**
19: XM = find_manager(New_indiv_X_i, $FitnessX_i$)
20: **end while**
21: **return** XM
22: **end**

The GLC Algorithm is explained as follows:

Lines 1-4: We initialize all the population based on top-k elements (i.e., k in algorithm 1) of the ranked classes, where the population represents the possible compositions. Then, we compute the GQC for each composition.

Line 5: We designate the manager of the population.

Line 6: We divide the population into a number of

groups.

Lines 7-9: We define a group leader for each group.

Lines 10-19: This constitutes the loop of a maximum iteration count, MaxIt.

Lines 11-18: This loop iterates over different groups.

Line 12: The function Update1 update the leader, designated as XL_g , of each group by employing Equation 24.

Lines 13-17: This loop iterates over the individuals within the population.

Line 14: The function Update2 updates the individuals based on Equation 25 .

Line 15: The function Update3 adjusts the individuals based on Equation 26.

Line 16: Random change in the position of some individuals using mutation.

Line 19: We designate the manager of the population.

Line 21: return the best composition.

6. EXPERIMENTAL STUDY

To assess the performance of the proposed service composition optimization approach, in terms of Global Quality of Service (QoS) Conformance and computation time (measured in milliseconds) We conduct a series of experiments using two different types of datasets.

The first dataset utilized is a real-world dataset based on the example provided in [15], derived from a real-world QWS dataset collected by Al-Masri and Mahmoud (2007) [2]. We generate 10 QoS realizations (l) of the response time and reliability according to a Gaussian distribution. The range of each task is presented in Table 5, where μ is the mean and σ is the standard deviation. The fidelity is uniformly generated between 1 and 4. Additionally, we set the global constraints of response time, reliability, and fidelity as 2400 ms, 0.025, and 3.

It is worth mentioning that all algorithms were implemented using the Java programming language. Experiments were performed on a Windows 7 with an Intel I3 core 2.53GHz processor, 4 GB RAM.

Table 3: Configuration of Gaussian distributions [15].

Task	Response Time	Reliability
Electronic Product Finder	μ∈[78,117] σ∈[0,30]	μ∈[0.63,0.95] $\sigma=\sqrt{\mu(1-\mu)}$
SendSMS	μ∈[1046,1569] σ∈[[0,435]	μ∈[[0.45,0.80] $\sigma=\sqrt{\mu(1-\mu)}$
SmartPayment	μ∈[117,175] σ∈[0,45]	μ∈[0.60,0.90] $\sigma=\sqrt{\mu(1-\mu)}$
CreditCard Validator	μ∈[254,48] σ∈[0,105]	μ∈[0.48,0.71] $\sigma=\sqrt{\mu(1-\mu)}$
UPSTracking	μ∈[74,111] σ∈[0,30]	μ∈[0.62,0.94] $\sigma=\sqrt{\mu(1-\mu)}$

In these experiments, we varied the number of candidate web services m from 100 to 500. In addition, the number of Top services (k) to 10. We leveraged 3 QoS attributes (i.e., response time, availability, and fidelity), and we generate 10 realizations (number of instances) per QoS attribute.

Furthermore, we varied the number of populations from 100 to 500 and the number of iterations from 100 to 500.

Table 4: GQC vs. population size and number of services (m).

	Iterations =100			
		Popsize =100	Popsize =300	Popsize =500
n=5, m=100, r=3, l=10, k=10	GLC	0.93	0.92	0.92
	GWC	0.91	0.91	0.92
	IM_ABC	0.50	0.54	0.54
n=5, m=200, r=3, l=10, k=10	GLC	0.92	0.94	0.94
	GWC	0.94	0.94	0.94
	IM_ABC	0.45	0.44	0.50
n=5, m=300, r=3, l=10, k=10	GLC	0.92	0.94	0.93
	GWC	0.93	0.95	0.95
	IM_ABC	0.36	0.45	0.50

Table 5: GQC vs. number of iterations and number of services (m).

	Popsize =300			
		MaxIter =100	MaxIter =300	MaxIter =500
n=5, m=100, r=3, l=10, k=10	GLC	0.91	0.91	0.90
	GWC	0.91	0.91	0.91
	IM_ABC	0.42	0.54	0.54
n=5, m=200, r=3, l=10, k=10	GLC	0.94	0.93	0.94
	GWC	0.94	0.94	0.93
	IM_ABC	0.56	0.60	0.54
n=5, m=300, r=3, l=10, k=10	GLC	0.91	0.94	0.94
	GWC	0.92	0.93	0.95
	IM_ABC	0.45	0.58	0.56

In Table 3 and 4, we conduct a comparative analysis between our proposed approach and the GWC approach [30] and IM_ABC approach [23] according to GQC. GLC is more powerful than IM_ABC approach in terms of GQC in all experiments presented in Tables 3 and 4. However, GQC performance of GLC is almost equal to the GWC approach.

Table 6: GQC vs. number of groups (G).

	G=4	G=10	G=20
n=5,m=100,r=3,l=10, Popsize=100	0.91	0.90	0.88
n=5,m=100,r=3,l=10, Popsize=300	0.91	0.90	0.91

To explore deeper into the analysis of GWC, we investigate the influence of the number of groups, which plays a crucial role in balancing exploration and exploitation. Thus, we conduct experiments with different group sizes, setting G to 4, 10, and 20, while tracking the GQC metric. Our observations reveal that as the number of groups increases, particularly when the population size is small, the solution tends to deviate from the near-optimal solution. For instance, after 100 iterations with a population size of 100 and G set to 4, the GQC of the near-optimal solution is recorded as 0.91. However, with G set to 20, the GQC value shows a divergence, decreasing to 0.88.

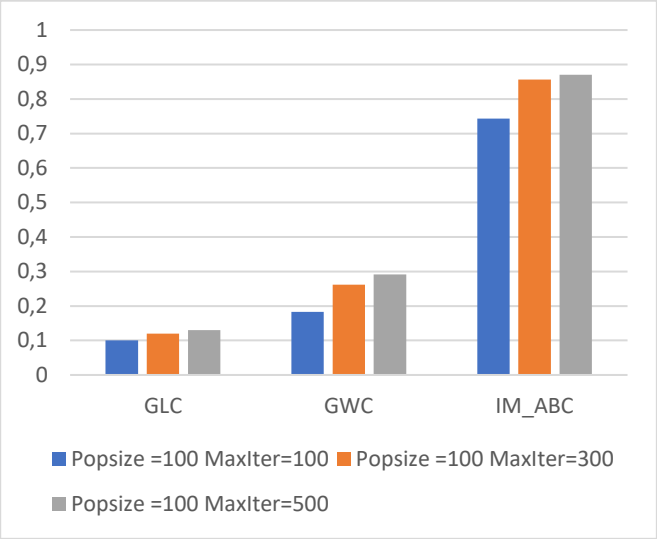

Figure 3: The effect of iterations on the CPU time.

Figure 3 displays the mean execution durations corresponding to the experiments detailed in Table 4. In this study, we assume the pre-computation of GQC. The results reveal that our proposed method exhibits a speed advantage over IM_ABC by approximately 0.7 seconds. Additionally, our approach demonstrates a marginal improvement in speed compared to GWC, with a difference of approximately 0.2 seconds. This outcome is primarily attributed to the computation of three near-optimal solutions within the GWC.

7. CONCLUSION

This paper introduces an optimization technique for web service composition under uncertain environment. Our approach consists of two main steps: firstly, employing local optimization for lowing the number of similar services by sorting and selection the most pertinent ones. Secondly, leveraging a global optimization to determine the near-optimal service composition that satisfies the user constraints. In future research, we intend to explore the correlations between both QoS user requirements and QoS of the web services. Additionally, we aim to consider the QoS of Cloud services (SAAS), such as Disc storage performance, I/O Memory performance, and data center regions into our approach for cloud composition.

REFERENCES

1. Ahanger, T. A., Dahan, F., & Tariq, U. (2023). **Hybridizing Artificial Bee Colony with Bat Algorithm for Web Service Composition**. Computer Systems Science & Engineering, 46(2).
2. Al-Masri, E., & Mahmoud, Q. H. **Discovering the best web service.** In Proceedings of the 16th international conference on World Wide Web 2007, May. pp. 1257-1258.
3. Bei, L., Wenlin, L., Xin, S., & Xibin, X. (2024). **An improved ACO based service composition algorithm in multi-cloud networks.** Journal of Cloud Computing, 13(1), 17.
4. Barkat, A., Kazar, O., & Seddiki, I. (2021). **Framework for web service composition based on QoS in the multi cloud environment.** International Journal of Information Technology, 13, 459-467.
5. Chattopadhyay, S., & Banerjee, A. (2020). **QoS-aware automatic Web service composition with multiple objectives.** ACM Transactions on the Web (TWEB), 14(3), 1-38.
6. Dahan, F. (2023). **Neighborhood Search Based Improved Bat Algorithm for Web Service Composition**. Computer Systems Science & Engineering, 45(2).
7. Dahan, F., & Alwabel, A. (2023). **Artificial Bee Colony with Cuckoo Search for Solving Service Composition.** Intelligent Automation & Soft Computing, 35(3).
8. Du, M. (2020). **An improved genetic algorithm for web service composition optimization.** World Scientific Research Journal, 6(10), 369-379.
9. Etchiali, A., Hadjila, F., & Bekkouche, A. (2023). **An intelligent bat algorithm for web service selection with QoS uncertainty**. Big Data and Cognitive Computing, 7(3), 140.
10. Ghobaei-Arani, M., & Souri, A. (2019). **LP-WSC: a linear programming approach for web service composition in geographically distributed cloud environments**. The Journal of Supercomputing, 75(5), 2603-2628.
11. Hadjila, F., Belabed, A., & Merzoug, M. (2020). **Efficient web service selection with uncertain QoS**. International Journal of Computational Science and Engineering, 21(3), 470-482.
12. Hamzei, M., Khandagh, S., & Jafari Navimipour, N. (2023). **A quality-of-service-aware service composition method in the internet of things using a multi-objective fuzzy-based hybrid algorithm**. Sensors, 23(16), 7233.
13. Haytamy, S., & Omara, F. (2020, February). **Enhanced qos-based service composition approach in multi-cloud environment.** In 2020 International Conference on Innovative Trends in Communication and Computer Engineering (ITCE) (pp. 33-38). IEEE.

14. Hosseinzadeh, M., Tho, Q. T., Ali, S., Rahmani, A. M., Souri, A., Norouzi, M., and Huynh, B. (2020). **A hybrid service selection and composition model for cloudedge computing in the internet of things**. IEEE Access, Vol. 8, pp. 85939–85949. Retrieved from https://doi.org/10.1109/ACCESS.2020.2992262 doi: 10.1109/ACCESS.2020.2992262
15. Hwang, S. Y., Hsu, C. C., & Lee, C. H. (2014**). Service selection for web services with probabilistic QoS**. IEEE transactions on services computing, 8(3), 467-480.
16. Kudermetov, R., Polska, O., Shkarupylo, V., & Shcherbak, N. (2020, September). **Normalization Techniques Comparison for QoS-based Web Services Selection by LSP Method**. In 2020 IEEE 5th International Symposium on Smart and Wireless Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS) (pp. 1-4). IEEE.
17. Liao, L., Wang, S., & Wu, J. (2023, May). **Research on web service composition selection based on QoS metrics.** In 2023 15th International Conference on Advanced Computational Intelligence (ICACI) (pp. 1-8). IEEE.
18. Poryazov, S., Andonov, V., Saranova, E., & Atanassov, K. (2022). **Two approaches to the traffic quality intuitionistic fuzzy estimation of service compositions.** Mathematics, 10(23), 4439.
19. Purohit, L., Rathore, S. S., & Kumar, S. (2023). **A QoS-Aware Clustering based Multi-layer Model for Web Service Selection.** IEEE Transactions on Services Computing.
20. Rahman, C. M. (2023). **Group learning algorithm: A new metaheuristic algorithm.** Neural Computing and Applications, 35(19), 14013-14028.
21. Remaci, Z. Y., Hadjila, F., Echialli, A., & Merzoug, M. (2021). **Dynamic Web Service Selection Based on Score Voting.** In Advances in Computing Systems and Applications: Proceedings of the 4th Conference on Computing Systems and Applications (pp. 185-195). Springer International Publishing.
22. Sefati, S., & Navimipour, N. J. (2021). **A qos-aware service composition mechanism in the internet of things using a hidden-markov-model-based optimization algorithm**. IEEE Internet of Things Journal, 8(20), 15620-15627.
23. Seghir, F., Khababa, A., & Semchedine, F. (2019). **An interval-based multi-objective artificial bee colony algorithm for solving the web service composition under uncertain QoS**. The Journal of Supercomputing, 75, 5622-5666.
24. Seghir, F. (2021). FDMOABC: **Fuzzy discrete multi-objective artificial bee colony approach for solving the non-deterministic QoS-driven web service composition problem**. Expert Systems with Applications, 167, 114413.
25. Tiwari, R. K., & Kumar, R. (2021). **G-TOPSIS: a cloud service selection framework using Gaussian TOPSIS for rank reversal problem**. The Journal of Supercomputing, 77, 523-562.
26. Torra, V. (2010). **Hesitant fuzzy sets.** International Journal of Intelligent Systems, Vol. 25, No. 6, pp. 529-539.
27. Xu, J., Guo, L., Zhang, R., Hu, H., Wang, F., & Pei, Z. (2018). **QoS-aware service composition using fuzzy set theory and genetic algorithm**. Wireless Personal Communications, 102, 1009-1028.
28. Xu, Z., & Xia, M. (2012). **Hesitant fuzzy entropy and cross-entropy and their use in multiattribute decision-making**. International journal of intelligent systems, 27(9), 799-822.
29. Xue, Y., Wang, J., & Jing, W. (2023). **An Enhanced Energy-Efficient Web Service Composition Algorithm Based on the Firefly Algorithm.** Journal of Database Management (JDM), 34(1), 1-19.
30. Yasmina, R. Z., & Fethallah, H. (2022). **Uncertain service selection using hesitant fuzzy sets and grey wolf optimisation.** International Journal of Web Engineering and Technology, 17(3), 250-277.
31. Youssef, A. E. (2020). An integrated MCDM approach for cloud service selection based on TOPSIS and BWM. IEEE Access, 8, 71851-71865.
32. Zeyneb Yasmina, R., Fethallah, H., & Fadoua, L. (2022). **Web service selection and composition based on uncertain quality of service**. Concurrency and Computation: Practice and Experience, 34(1), e6531.