Open Access ISSN: 3007-1437 (Online), ISSN: 3007-1429 (Print)

Alkadhim Journal for Computer Science (KJCS)

Journal Homepage: https://alkadhum-col.edu.iq/JKCEAS

Chicken Swarm Optimization based QoS Web Service Composition in Cloud Computing Environment

[1]Dina Riadh Alshibani*

[1]Mustansiriyah University, College of Science, Computer Science Department, Baghdad, Iraq

Article information

Article history:
Received: January, 30, 2026
Accepted: March, 23, 2026
Available online: March, 25, 2025

Keywords:
Keyword Cloud computing,
Keyword Web service composition,
Keyword QoS

Corresponding Author:
Full Name
dinashibani@uomustansiriyah.edu.iq

DOI:
https://doi.org/ 10.61710/kjcs.v4i1.153

Abstract

The increasing volume of data in the cloud computing environment has led to a situation where a single, abstract cloud web service has lost its ability to meet the complex requirements of various customers. Therefore, a service composition is becoming a necessity in the cloud computing environment. In this paper, Chicken Swarm Optimization (CSO) based QoS Web Service Composition in Cloud Computing Environment CSOQSC_CCE algorithm has been proposed. CSO is a meta-heuristic algorithm that inspired from the hierarchy and foraging habits of the chicken swarm. The proposed CSOQSC_CCE algorithm is validated through using the Quality of Web Service (QWS) which is a real-world service dataset. The investigation results show that the proposed CSOQSC_CCE algorithm overcome the existing approaches in terms of response time and delay.

1. Introduction

Composite service can be defined as a group of abstract cloud services that are designed and built to handle complex user requests in an active service-oriented computing environment. The process of creating a composite web service includes combining various abstract service sets from various sources to build a unified service that meets customer requirements [1]. In other words, the composite service is created from a pool of services where individual abstract services are selected based on their QoS attributes. The QoS refers to a wide range of attributes such as response time, cost, latency, and reliability [2, 3].

The growth of the data volumes has added new challenges to the process of the large service compositions with optimal QoS values. These challenging issues raised the need for intelligent approaches to be able to process the huge search spaces of combined services while improving maintainability and adaptability to diverse data and ensuring compliance with service level agreements. To address this challenge, several studies have been introduced in the literature to solve QoS-aware abstract web service composition including heuristic [4], machine learning [5], and meta-heuristic algorithms [6]. Each approach presents distinctive advantages and limitations. Heuristic approaches facing the problem of finding optimal service composition within polynomial time which makes them a poor choice for cloud web service composition because it is classified as non-polynomial problems. On the other hand, the machine learning techniques have their dependency on extensive training data that can be excessive particularly in complex situations. While, meta-heuristic algorithms offer promising results in large-scale service composition [7].

This research article has introduced a method that uses the Chicken Swarm Optimization (CSO) to balance multiple QoS factors and satisfy service composition constrains. Experimental results demonstrate the superiority of the introduced method over previous ones in terms of response time, and delay.

The rest of the paper is arranged as follows: Section 2 previous studies. Section 3 presents the basic steps of CSO; Section 4 introduce the model description; Section 5 presents the outcome of the proposed method; Section 6 introduces conclusion and the future work.

2. Previous Studies

In this section a summary of previous studies related to the proposed algorithm is provided:

For efficient cloud service composition in terms of energy consumption minimization in multi-cloud environments a heuristic approach is proposed in [8]. The proposed approach found that the service composition in one cloud has a great impact on reducing the energy consumption via studying the effect of the split point positions strategy on reducing the energy consumption. The investigation results demonstrate that the proposed approach offered a fair performance regarding energy consumption and execution time. To satisfy the requirements of QoS over the cloud an integration of honey badger and chameleon optimization algorithm based cloud service composition is introduced in [9]. The empirical investigation showed that the proposed algorithm outperforms the baseline cloud service composition approaches used for investigation by improving the availability, reliability, and response time. An efficient cloud service composition by balancing multiple QoS factors using the integration of fuzzy logic and the Firefly Optimization Algorithm (FOA) has been introduced in [10]. The examination results showed the superiority of the proposed method over the existing approaches. To achieve the optimal service composition in the healthcare field a multi-layered approach based on repeated training, Kalman filtering, and Deep Reinforcement Learning (Deep RL) has been introduced in [11]. The outcomes of the proposed method show that the proposed method has achieved fair results in terms of reliability, availability, and response time, energy consumption as compared to the existing methods. A modified version of the Moth-Flame Optimization (MFO) algorithm is used in [12] to construct cloud computing service compositions by employing a new diversity optimization mechanism called Stagnation Finding and Replacement (SFR). Empirical evaluations show that the proposed algorithm outperforms current methods for constructing multi-cloud computing services. A multi-layered cloud service composition framework has been proposed in [13]. The proposed framework which is entitled IPAA combines Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO) with a multi-agent system to achieve better convergence speed. The empirical results demonstrate that the proposed IPAA framework provides a suitable, robust method for cloud service composition. In addition, the proposed framework has a better results compared with PSO and ACO in terms of scalability, utility score, and execution time. In [14] a multi-objective cloud manufacturing service composition has been presented to improve resource utilization rate, service quality, cost and time, based on the integration of fuzzy sets and an improved version of the NSGA-III algorithm. The research results confirm the effectiveness of the presented methods compared to existing approaches. For efficient cloud service composition, a chemistry-based algorithm which is inspired from the electron shell theory and the periodic table's is introduced in [5]. The results of the proposed approach compared with the well-known optimization algorithms including: Tabu search, simulated annealing and Genetic algorithms. These results prove the proposed approach efficiency in improving the computational overhead of the muli-cloud service composition.

3. Chicken Swarm Optimization (CSO)

The hierarchy and foraging habits of the chicken swarm are modeled using CSO. CSO's three types of chickens are roosters, hens, and chicks [15]. The following assumptions are made by CSO when solving the optimization problem:

1. The chicken swarm is divided into several groups. There is a dominant rooster, followed it some hens and chicks in each group.

2. According to the individual's fitness values, the fittest individuals are chosen as roosters, the least fit are chosen as chicks, and the remaining individuals are the hens.
3. Each hen randomly chooses a rooster to join his flock as her mate, and each chick similarly chooses one hen to serve as its mother.
4. After every G time iteration, the rooster-hen-chick hierarchical order, which represents the matting relationship between roosters and hens, in addition to the mother-child relationship between hens and chicks, will be updated.
5. In each subgroup, the roosters are the ones who take the lead in foraging for food, followed by the hens and then the chicks. To put it in another way, three separate search equations are used by three different types of chickens as follows [16]:

Rooster position update:

$$xR_{i,j}^{t+1} = xR_{i,j}^{t} \times (1 + \mathrm{Random}(0, \sigma^2)) \qquad (1)$$

$$\sigma^2 = \begin{cases} 1 \text{ if } f_i < f_k \\ \exp(\frac{f_i - f_k}{f_i + \varepsilon}) \end{cases} \qquad (2)$$

Where xRi,j is the chosen rooster to update its position, Random (0, σ2) is a Gaussian distribution with standard deviation σ2 and mean 0. ε is used to avoid the error of zero-division as the minimum value in the computer. The index of a randomly selected rooster from the group of roosters is k. fi is the fitness value of the associated rooster xRi,j. These Hen position updates can be mathematically formulated as in Eqs. (3), (4) and (5).

$$xH_{i,j}^{t+1} = xH_{i,j}^{t} + C_1 + R1 \times \left(xR_{r1,j}^{t} - xH_{i,j}^{t}\right) + C_2 + R2 \times \left(xR_{r2,j}^{t} - xH_{i,j}^{t}\right) \quad (3)$$

$$C_1 = \exp(\frac{f_i - f_{r1}}{\mathrm{abs}(f_i) + \varepsilon}) \qquad (4)$$

$$C_2 = \exp(f_{r1} - f_i) \qquad (5)$$

Where, C1 and C2 represent the learning factors, R1 and R2 are random values following uniform distribution in the scope of [0, 1], xRr1 is the spouse rooster index of ith hen, xR_{r2} is the index number of a hen or a rooster which is randomly selected, and $xR_{r1} \neq xR_{r2}$.
These chick position updates can be mathematically formulated as in equation (6) [17,18]:

$$xC_{i,j}^{t+1} = xC_{i,j}^{t} + LF * (xMother_{i,j}^{t} - xC_{i,j}^{t}) \qquad (6)$$

Where $xMother_{i,j}^{t}$the index of hen mother of chicks and LF is a randomly chosen value in the range of [0, 2].

4. Model Description

In this section, the design and the implementation of the proposed CSOQSC_CCE algorithm are explained in detail:

4.1 The Problem Description

Assume a complex user request T that can be divided into n subtasks. T={T1,T2,..Tn}. Each subtask Ti can be serviced by one abstract service Ai, and the whole combination of the abstract services Ai forms the composite service C.
The set C={C1, C2,..,Cn} is a subset of a global service pool GS={GS1,GS2,..GSm} .Where m is the total number of available services. It is worth mentioning that all GSs have the same functionality with different QoS.
The utility function of the composite service is computed as defined in Eq.7.

$$U(T) = \sum_{i=1}^{n} W_0 \times Q_{la}(i) + W_1 \times Q_{re}\ (i) \qquad (7)$$

Where U(T) is the QoS value of the composite service T. n is the total number of abstract services which construct T. w0 and w1 are the weights assigned to each QoS. Q_{la} (i) is the latency aggregated function, while Q_{re} (i) is the response time aggregated function. Table 1 depicts the aggregated function for the QoS used in this research article [19].

Table (1): Aggregate function

QoS Attribute	Aggregate Function
Latency or delay	$Q_{la}(S) = \sum_{i=1}^{n} Q_{la}^{i}$
Response time	$Q_{re}(S) = \sum_{i=1}^{n} Q_{re}^{i}$

4.2 The Proposed CSOQSC_CCE Algorithm Implementation

The proposed CSOQSC_SCE algorithm consists mainly of three sequential steps. The representation of the cloud service composite is the first step, followed by the generation of the initial composite service set, while selecting the optimal composition is the final step. Fig. 1 show the general structure of the proposed CSOQSC_CCE.

4.2.1 Composite Service Representation

The composite cloud service is represented as a 1-dimensional vector:

$$coms=[s1,s2,...,sn] \qquad (8)$$

Where n is the number of the of subtasks, s is the selected service for subtask i.

4.2.2 Construction of the Initial Service Composition Set

The second step of the proposed CSOQSC_SCE algorithm is the generation of the initial composite service set. The generation is performed randomly as defined in Eq. 9.

$$coms_{i,j}=floor\ (LB_i+rand\times(UB_i-LB_i+1)) \qquad (9)$$

Where i=0,..,n , n is the number of the subtasks , j=0,..,Isize, Isize the number of the initial composite services set, rand is a random number between 0 and 1 , LB is the lower limit here it is 0 , while UB is the upper limit which describe the complete number of the offered services.

4.2.3 Optimal Composite Service Identification

After constructing the initial composite set, the proposed algorithm processed to the last step which is select the optimal composition regarding delay and response time using CSO algorithm. The main step of the proposed is illustrated in algorithm 1. Fig. 1 show the general structure of the proposed CSOQSC_CCE.

Algorithm 1: CSOQSC_CCE

Input:

Number of submitted requests n, number of services s, the size of the initial service composition (Isize), maximum number of iterations (MIter), proportion of roosters, hens, and chickens, W1 and W2 the weights assigned to each QoS.

Output:

The best service composite with the optimal QoS in terms of response time and delay.

Start

1. Generate the initial cloud service combinations matrix *icgroup* of size Isize×n, randomly, as defined in Eq.9. Where each cloud service combination is subset of s.
2. For each initial cloud service combination *csa* in *icgroup* perform the following:

2.1 Calculate the normalized version of the response time and the delay. Both the response time and the delay are negative QoS meaning the less the better according to this the normalization is performed as defined in Eq. 10.

$$Q_j = \frac{Q_{jmax} - Q_{i,j}}{Q_{jmax} - Q_{jmin}} \qquad (10)$$

2.2 Calculate the aggregation function of response time and delay according to Table 1.

2.3 Calculate the utility function *U* value as defined in Eq 7.

3. End for

```
4.   Sort the utility function value U in ascending order.
5.   Save the optimal QoS value in bS.
5.1  While MIter is not reached
5.2  Determine the roosters, hens, and chickens according to the predefined portions and the values of the utility
     function.
5.3  For each initial cloud service combinations csa in icgroup perform the following:
     Determine csa type according to the predefined portions and the utility function.
     If csa is a rooster update its position according to Eq.1
     Else if csa is a hen update its position according to Eq.3.
     Else if csa is a chick update its position according to Eq. 6.
5.4  End for
5.5  Digitize the new constructed floated cloud service composite set by using Eq.11.
                    $S_i=((\lfloor|x_i|\times10^4\rfloor \bmod s)+1+D_i) \bmod s+1$        (11)
     Where m D is number of the previous repetition.
5.6  Compute the utility function for the newly generated cloud service composition after normalize them.
5.7  Select the best utility function and update bS.
6.   End while
7.   Return the optimal cloud service composition bS.
End of algorithm.
```

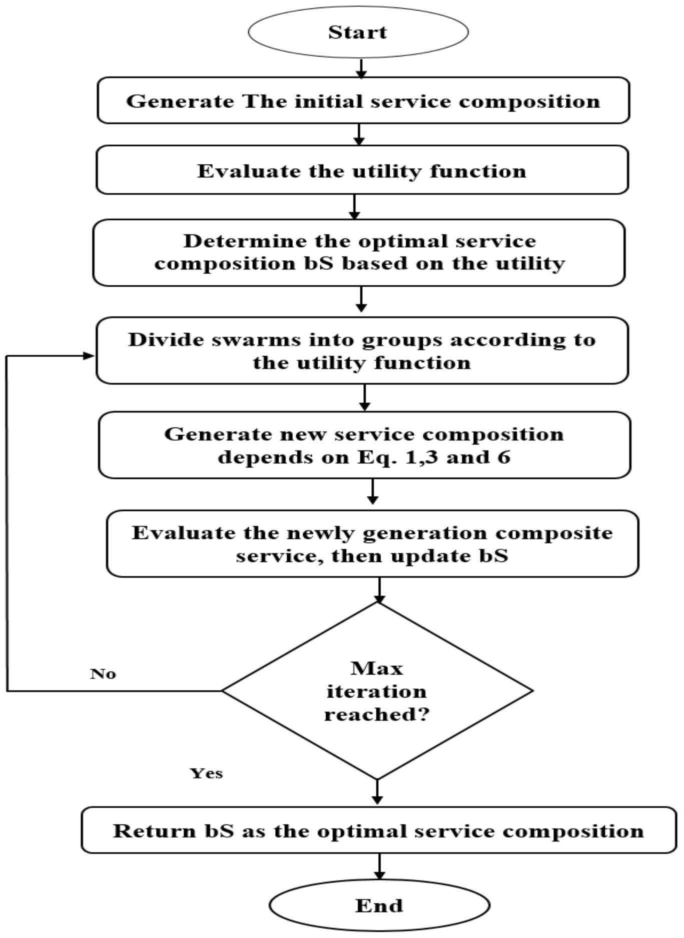
Start
Generate The initial service composition
Evaluate the utility function
Determine the optimal service composition bS based on the utility
Divide swarms into groups according to the utility function
Generate new service composition depends on Eq. 1,3 and 6
Evaluate the newly generation composite service, then update bS
Max iteration reached?
No
Yes
Return bS as the optimal service composition
End

Figure (1): The general steps of the proposed CSOQSC_CCE algorithm

5. Results and Discussions

To verify the effectiveness of the proposed CSOQSC_CCE algorithm several experiments have been conducted in which JAVA language is utilized for the proposed CSOQSC_CCE implementation. All experiments are accomplished on a Lenovo laptop with a Core i7 2.4 GHz CPU, Windows 11 pro and 16 GB RAM. The utilized QWS dataset [20] includes a set of 2507 web services and measures their QoS. Two QoS have been used in this paper which are the response time and the delay both measured in milliseconds. The dimensions of response time are 30-5000, while the delay times are 0.1 to 4500. The portion of roosters, hens and chicks are 20%, 20% and 60%, respectively. The number of tasks and services ranged from 100 to 1000, with a step of 100. The weights assigned for each of the response and the delay time is 0.5. A comparison has been achieved between the proposed CSOQSC_CCE and the re-implemented FOA [9] and MFO-SFR [11] to confirm the efficiency of the proposed method. Performance has been evaluated based on two metrics: the response and the delay time.

Two parameters related to the CSO algorithm need to be tuned in order to determine their optimal values. These parameters include the initial service composite set size and the number of iterations. The effect of these parameters on the average fitness value for 15 run is illustrated in Fig. 2 and Fig. 3, respectively. Fig. 2 show that the minimum average fitness value is obtained when the size of the initial service composite set size is 200.

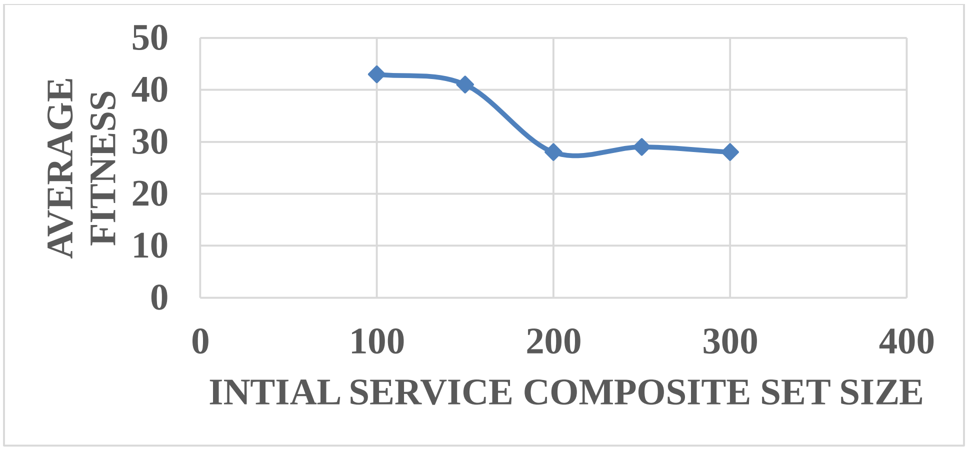

Figure (2): the relationship of the initial service composite set size and the average fitness.

While Fig. 3 show that the stability of the proposed CSOQSC_CCE is achieved after reaching the optimal region 300 by having the minimum average fitness, further increases do not provide additional improvement. Therefore, a stability value around 300 is recommended as the optimal setting for achieving the best convergence.

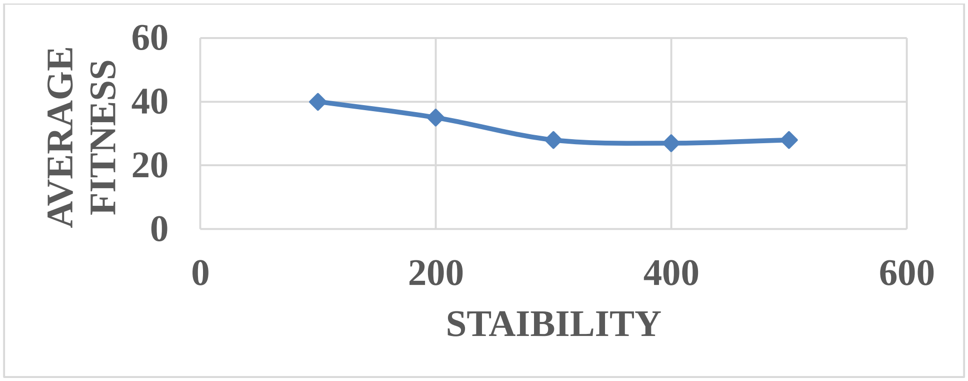

Figure (3): the stability of the proposed

Fig. 4 depicts the outcome of the proposed CSOQSC_CCE algorithm and the compared algorithms in terms of response time. The results in Fig. 4 clearly demonstrate the superiority of the proposed CSOQSC_CCE algorithm in minimizing response time in all the tested cases.

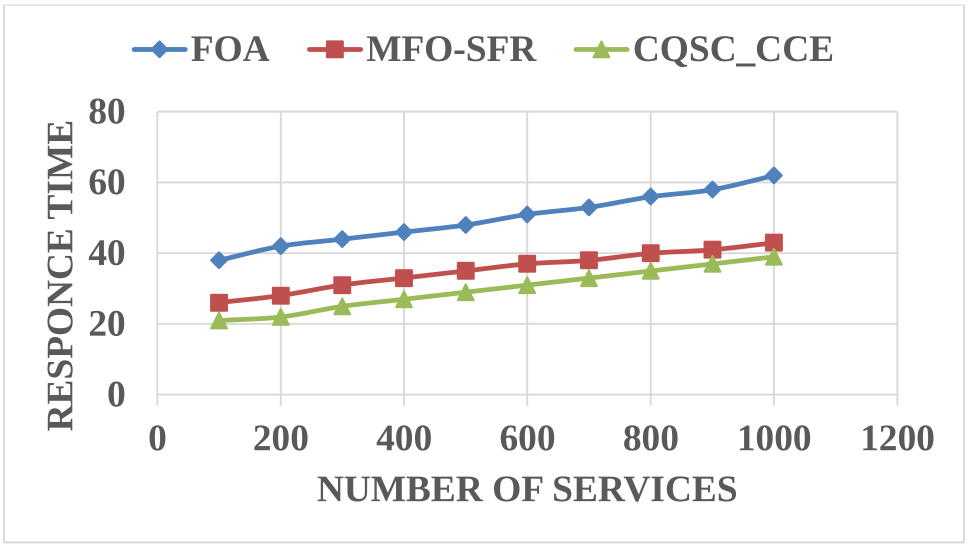

Figure (4): response time comparison

Fig. 5 shows the outcome of the proposed CSOQSC_CCE algorithm and the compared algorithms in terms of delay time. Obviously the proposed algorithm outperforms the compared algorithms in reducing delay time.
The results of the proposed CSOQSC_CCE algorithm promise unlimited effectiveness when studying both response time and delay. On the one hand, reducing response time can contribute to improving the performance efficiency of any service. While minimizing delays demonstrates how the time-sensitive aspect of cloud computing service delivery can be improved, ensuring user satisfaction with the service. Therefore, the gained results support the study's stated objectives regarding QoS standards in the environment of cloud computing and further emphasize the algorithm's practical importance. Compared to existing methods, the proposed CSOQSC_CCE algorithm has achieved better results in terms of scalability and efficiency in addressing the challenges of configuring complex services in cloud computing environments.

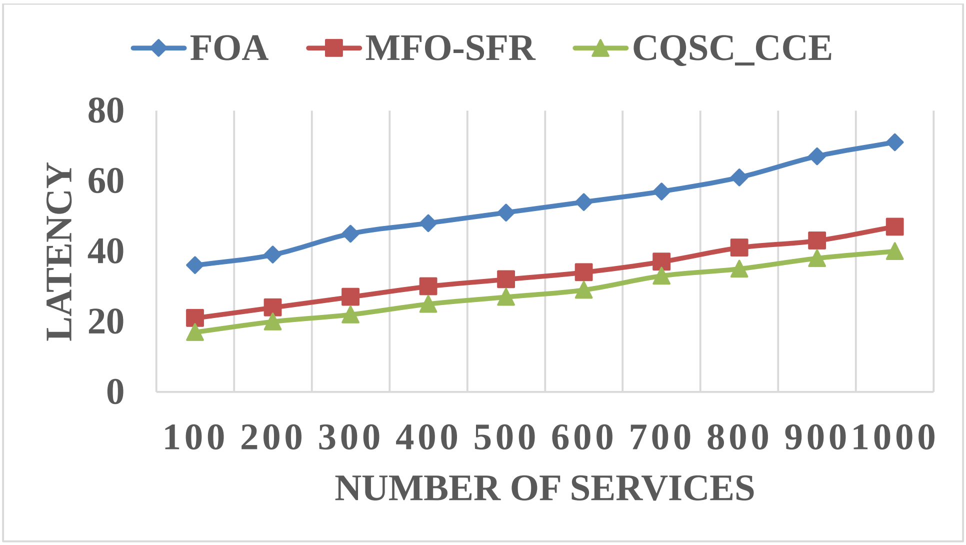

Figure (5): Latency comparison

6. Conclusions

With the recent growing of data volumes, a single service became useless to large and complex requests. Thus, a big service composition is becoming an essential matter in the cloud computing environment. A chicken swarm optimization algorithm is proposed in this paper to solve the composition problem. The experimental results prove that the proposed algorithm outperforms the existing approaches by having the optimal service composition in terms of response time, and delay. While also being able to fulfill complex service requests with minimal overhead. This is mainly returned to the CSO searching mechanism which can dynamically maintain an effective searching process to have an optimal service combination, resulting in a more diverse service combination. In future work, the proposed method can be implemented in a multi-cloud environment and using multi-agent approaches.

Acknowledgement: “The author would like to thank Mustansiriyah university for supporting this work”

Conflict of Interest: “The author declares that there no conflicts of interest “

References

[1] C. Li, J. Li, and H. Chen, “A meta-heuristic-based approach for QoS-aware service composition,” *IEEE Access*, vol. 8, pp. 69579–69592, 2020.

[2] M. Rahimi, N. J. Jafari Navimipour, M. Hosseinzadeh, M. H. Moattar, and A. Darwesh, “Toward the efficient service selection approaches in cloud computing,” *Kybernetes*, vol. 51, no. 4, pp. 1388–1412, 2022.

[3] S. S. Sefati and S. Halunga, “A hybrid service selection and composition for cloud computing using the adaptive penalty function in genetic and artificial bee colony algorithm,” *Sensors*, vol. 22, no. 13, p. 4873, 2022.

[4] J. Jayaudhaya, R. Jayaraj, and K. K. Ramash, “A new integrated approach for cloud service composition and sharing using a hybrid algorithm,” *Mathematical Problems in Engineering*, vol. 2024, Art. no. 2024, 2024.

[5] M. Aldakheel and H. Kurdi, “A chemistry-based optimization algorithm for quality of service-aware multi-cloud service compositions,” *Mathematics*, vol. 13, no. 8, p. 1351, 2025.

[6] L. Bei, L. Wenlin, S. Xin, and X. Xibin, “An improved ACO-based service composition algorithm in multi-cloud networks,” *Journal of Cloud Computing*, vol. 13, Art. no. 17, 2024.

[7] M. Maurya, G. Vidhya, F. Akram, M. Saravanan, K. V. Ramana, and S. Rajakumari, “Ant colony optimization for service composition in multi-cloud environments,” in *Proc. 15th Int. Conf. Computing, Communication and Networking Technologies (ICCCNT)*, IEEE, 2024, pp. 1–6.

[8] G. Manimala and A. Chinnasamy, “Hybrid chameleon and honey badger optimization algorithm for QoS-based cloud service composition problem,” *Computer Systems Science & Engineering*, vol. 47, no. 1, 2023.

[9] W. Wang and Z. Liu, “Cloud service composition using firefly optimization algorithm and fuzzy logic,” *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 3, 2023.

[10] C. Zhong, M. Darbandi, M. Nassr, A. Latifian, M. Hosseinzadeh, and N. J. Navimipour, “A new cloud-based method for composition of healthcare services using deep reinforcement learning and Kalman filtering,” *Computers in Biology and Medicine*, vol. 172, p. 108152, 2024.

[11] Y. Yang and M. Song, “Modified moth-flame optimization algorithm for service composition in cloud computing environments,” *International Journal of Advanced Computer Science & Applications*, vol. 16, no. 1, 2025.

[12] D. Tian, “A novel quality of service-aware service composition method for cloud computing using enhanced prairie dog metaheuristic optimization algorithm,” *Journal of Engineering and Applied Science*, vol. 72, no. 1, p. 142, Dec. 2025.

[13] H. Yang, F. Xue, H. Zhu, and L. Li, “Web service composition in the cloud environment based on modified beetle antennae particle swarm optimization,” in *Proc. 2nd Int. Conf. Machine Learning, Big Data and Business Intelligence (MLBDBI)*, IEEE, 2020, pp. 362–366.

[14] X. Liu, R. Yang, X. Li, and X. V. Wang, “A cloud manufacturing service composition optimization method for fuzzy demands based on improved NSGA-III algorithm,” *Robotics and Computer-Integrated Manufacturing*, vol. 97, p. 103106, 2026.

[15] B. Chen, L. Cao, C. Chen, Y. Chen, and Y. Yue, “A comprehensive survey on the chicken swarm optimization algorithm and its applications: State-of-the-art and research challenges,” *Artificial Intelligence Review*, vol. 57, no. 7, p. 170, 2024.

[16] M. Saraswathi and E. Logashanmugam, “Chicken swarm optimization modelling for cognitive radio networks using deep belief network-enabled spectrum sensing technique,” *PLoS ONE*, vol. 19, no. 8, p. e0305987, 2024.

[17] D. C. Diana, R. Hema, and R. Ramya, “Chicken particle swarm optimization algorithm for MMSE equalization in MIMO systems,” *Franklin Open*, p. 100496, 2026.

[18] G. Anitha, N. Supriya, F. Alenezi, E. L. Lydia, and G. P. Joshi, “Chicken swarm optimization with deep learning-based packaged rooftop units fault diagnosis model,” *Computer Systems Science & Engineering*, vol. 47, no. 1, 2023.

[19] Z. Guo and X. Yu, “Cloud service quality assessment based on entropy weight method,” in *Proc. Int. Conf. Cryptography, Network Security, and Communication Technology (CNSCT 2023)*, SPIE, vol. 12641, pp. 253–259, May 2023.

[20] R. R. Kumar, A. Tomar, M. Shameem, and M. N. Alam, “OptCloud: An optimal cloud service selection framework using QoS correlation lens,” *Computational Intelligence and Neuroscience*, vol. 2022, Art. no. 2019485, 2022.