Soft Computing Journal, vol. 12, no. 1, pp. 27–33, 2023
https://doi.org/10.22052/scj.2023.246636.1084

University of Kashan

Contents lists available at SCJ

Soft Computing Journal

Journal homepage: https://scj.kashanu.ac.ir/

Optimizing web service composition through hybrid graph simplification and NSGAII algorithm ✧

Narjes Zahiri [1], PhD student, Seyed Morteza Babamir [1,*], Professor
[1] *Department of Software Engineering, University of Kashan, Kashan, Iran.*

ARTICLE INFO.

Article history:
Received July 21, 2022
Accepted December 13, 2022

Keywords:
Web service composition
Graph simplification
Service selection
Complex probabilistic structures
NSGAII algorithm

ABSTRACT

In the rapidly evolving landscape of web services, efficient interaction and optimal selection among services with different quality parameters are essential. This paper addresses the complex challenge of selecting candidate services for abstract services within probabilistic graph structures. We propose a new hybrid method that combines node-based and path-based graph simplification techniques, allowing for the identification of new patterns, such as parallel and nested loops. We use NSGAII to improve scalability and accuracy in service selection. The proposed approach simplifies the composition graph while optimizing the selection process by considering important quality parameters such as execution cost, response time, and availability. Through systematic simplification and a robust fitness function, we ensure a definitive and accurate response to user queries. The results show significant improvements in the proposed approach compared to existing methods, making it a comprehensive solution for effectively composing web services in dynamic environments.

1 Introduction

Web services, as a computing model, have developed very fast in recent years [1, 2]. Since each web service is designed for one specific function, multiple web services need to collaborate with each other in order to provide effective responses to the complex needs of users [3, 4]. Moreover, many web services provide similar functionalities but differ in their quality. Hence optimal selection and composition of these services are the most important and complex challenges in this area of research [5].

In a web service graph, each service, also called an abstract service, has several candidate services that can perform the same function but with different quality parameters. The most important problem is selecting a candidate service for each abstract service in the graphs with probabilistic patterns. To solve this problem, the graph can be simplified through two primary conversions: (1) a node or (2) a tree with different distinct paths [6]. Graph simplification methods are divided into three categories: 1) node-based methods, which yield a single node after removing sequential, parallel, conditional and loop patterns, 2) path-based methods, which extract all paths between the initial and terminal nodes of the graph, and 3) Hybrid methods, which are the combination of node- and path-based methods [6]. In the hybrid method, sequential, parallel, and loop patterns are removed by the node-based method, while conditional patterns remain to preserve the probability of each graph path and to maintain the accuracy of the algorithm (as the multiplication of transition probabilities in the quality parameters of the nodes is not performed) [3, 6]. This article uses the hybrid method. Since graph sim-

✧ Article type: original research (short paper)

* Corresponding author.
Email addresses: nargess.zahiri@gmail.com (N. Zahiri), babamir@kashanu.ac.ir (S. M. Babamir)

How to cite this article: N. Zahiri, S. M. Babamir, "Optimizing web service composition through hybrid graph simplification and NSGAII algorithm", *Soft Comput. J.*, vol. 12, no. 1, pp. 27–33, 2023, doi: 10.22052/scj.2023.246636.1084.

plification depends on the quality of the candidate services, in addition to the structural patterns of the graph, in the following, we examine the quality parameters of web services and the challenges associated with their optimal selection.

The quality of web services includes non-functional features such as execution cost, execution time, and service availability. These quality features are classified into two categories: negative features, which should be minimized (e.g., response time and cost), and positive features, which should be maximized (e.g., availability) [7, 8].

The approaches presented in references [9–11] provide solutions for the optimal selection of candidate web services; however, they exhibit two notable limitations: (1) only the sequence pattern in the web services composition graph is considered, as a result, they do not support probabilistic structures, and (2) they use the node-based method. Similarly, in the approach presented in [12], in addition to the sequence pattern, the parallel pattern is considered, but this approach has the second limitation. The approaches presented in [13–15] consider all kinds of patterns in web service composition graphs and also support the probabilistic conditional pattern and repeated loops. However, they do not support probabilistic parallel and loop patterns and also face the second limitation. Our primary focus is on supporting various types of web service composition graph structural patterns and their simplification methods. On the other hand, the most complete approach is presented in reference [6], which we consider as the basic work. In this approach, complex patterns including loops with more than two nodes and various conditional patterns are covered, and the probability of each path is also calculated. Nevertheless, it has several shortcomings: (1) does not consider nested and parallel loop patterns; (2) it models the problem as a constraint satisfaction problem rather than an optimization problem; (3) it employs a non-evolutionary algorithm for the selection of optimal candidate services, resulting in limited scalability in terms of time; and (4) After simplifying and finding the probability of each path, it analyzes each path separately, which is an obstacle to give a definitive answer to the customer. In this article, we address these 4 cases. Therefore, the innovations of this article are as follows:

- Supporting all types of patterns and probabilistic transitions between services, as well as the introduction of two new patterns of parallel and nested loops.
- Using an evolutionary algorithm for optimal candidate selection to maintain scalability.
- Providing a hybrid method based on repeating the path to improve the accuracy of responses to customers.

2 Related work

This section is organized into several subsections that focus on methods for simplifying patterns in the web services composition graph.

2.1 Node-based methods

Yao et al. (2009) used the NSGAII to find optimal solutions in graphs with complex patterns and compared its performance with the genetic algorithm (GA) in terms of the convergence speed and the generation of optimal solutions [13]. Li et al. (2010) utilized the NSGAII to determine optimal solutions in the web service composition by considering 10 objective functions. They evaluated their algorithm by reporting the number of dominant and non-dominant solutions [10]. Sharif-Ara et al. (2014) proposed a method that can find the optimal solution at a high speed. This approach initially simplifies the problem to a single-objective format, subsequently employing a combination of GA and fuzzy methods to enhance speed, accuracy, and reliability, before transforming it into a multi-objective problem addressed by the NSGAII [14]. Liu et al. (2015) compared NSGAII, MOPSO, a hybrid of NSGAII and hierarchical analysis, and a hybrid of MOPSO and hierarchical analysis. According to this comparison, they used hierarchical analysis in their solutions [15]. Gohain et al. (2016) applied a hybrid algorithm of ACO and PSO by converting the five objective functions—reliability, availability, throughput, cost, and response time—into a single objective [9]. Ying Huo et al. (2017) proposed the elitist multi-objective bee colony algorithm, integrating the bee colony and NSGAII, which demonstrated superior performance compared to NSGAII, PSO, and the bee colony algorithm in terms of spread, GD, and execution time [11]. Sadouki et al. (2019) used the discrete multi-objective elephant algorithm. They are inspired by the Pareto approach and SPEAII sorting method. Their results indicated that this algorithm significantly outperformed the PSO and SPEAII algorithms in terms of coverage ratio, spread, and hypervolume [16]. Dahan et al. (2021) Introduced a hybrid algorithm that combines ACO and GA, where GA is used to automatically tune the parameters of ACO and ACO adapts its performance based on the parameters tuning. The main contribution of their work is to help the ACO algorithm to avoid the stagnation problem. The results show that their algorithm requires more CPU time than other methods; however, it is better in terms of cost, response time, reliability, and availability [17]. Additionally, Dahan et al. (2021) proposed a new method named Enhanced Flying Ant Colony Optimization (EFACO), which incorporates three main innovations. First, it avoids the execution time problem by restricting the flying process to only occur when there are improvements in solution qual-

ity. Second, a neighboring selection method is applied to avoid scanning all the neighboring nodes, which may decrease the quality of the solutions. Lastly, they introduced a third modification that transforms the algorithm into a multi-pheromone algorithm, effectively addressing the drawbacks of the first two modifications. The results demonstrate that EFACO outperforms the other methods in terms of solution quality and execution time [18].

2.2 Path-based methods

Ardagna et al. (2005) proposed a mixed integer linear programming algorithm evaluated by considering four objective functions and converting them to a single objective function. This problem has been solved by converting the web service composition graph into different execution paths. In this method, there are selected paths that provide the most optimal solution [12].

2.3 Hybrid methods

Zheng et al. (2012) proposed a new pattern to develop their previous work in 2009. This pattern is called a multiple-entry multiple-exit unstructured loop pattern. To simplify the web-service composition graph, they used a hybrid method, which calculates the probability of each path. An integer programming algorithm has been used for the optimal selection of candidates [6].

3 Proposed method

The proposed method is briefly divided into two main components. In the first component, the problem of web service composition is addressed. In addition to identifying two new patterns, we propose a method for simplifying the composition graph, resulting in a more accurate and definitive answer to customers. After simplifying the graph by replacing a different candidate, the proposed method generates a population (composition algorithm). In the second component, the NSGAII algorithm is used to find optimal solutions for this problem (selection algorithm).

3.1 Identification of two new patterns

In this section, two new patterns are identified.

Parallel loop pattern: Two nodes v_2 and v_3 are called parallel loops if they: (a) are parallel to each other (according to the definition of parallel pattern [6]), and (b) both nodes have an input edge to the shared input node. Condition (b) indicates that there are two input edges, e_{j2} and e_{j3}, defined as $e_{j2} = (v_2, v_1, p_{21})$ and $e_{j3} = (v_3, v_1, p_{31})$ (see Fig. 1).

Nested loop pattern: The sequence of nodes $v_1, v_2, \cdots, v_m, v_{m+1}, \cdots, v_n$ form a nested loop if:

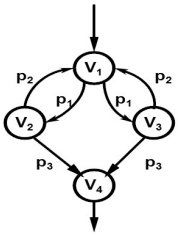

Fig. 1. Parallel loop

(a) two sets $\{v_1, v_2, \cdots, v_m\}$ and $\{v_m, v_{m+1}, \cdots, v_n\}$ form a loop [6], and (b) these two loops have at least one shared node. Condition (b) states that a node from the first loop (v_m) is the input node for the second loop (see Fig. 2).

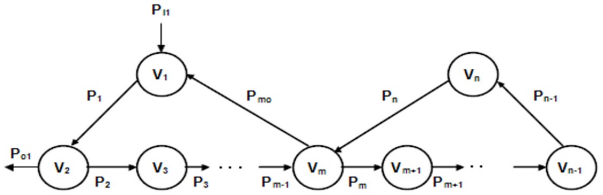

Fig. 2. Nested loop

3.2 Composition algorithm

In this section, we explain the details of the composition algorithm to simplify the composition graph.

First step: For each web service in the composition graph, a random candidate is selected from the QWS dataset.

Second step: The patterns in the graph are examined in such a way that the sequence pattern is checked first, then the loop, and finally the parallel patterns are checked. This process is repeated until none of these three patterns remain present in the graph. During the simplifying of each pattern, the candidate quality values and the available possibilities are also updated (as detailed in [6]).

Third step: If the result of simplifying the patterns is a node, it is considered a member of the population; otherwise the tree is created from the graph, and each path, after simplifying the sequence pattern (the existing pattern in each path), is considered as a member of the population separately.

Fourth step: The first to third steps are repeated according to the number of initial population members [3].

3.3 Optimal selection of web services

After simplifying the graph, the selection algorithm is invoked to find the optimal selection of web services. To implement this algorithm, it is necessary to define: 1) the initial population (set of solutions), and 2)

the fitness function used to select the best candidate for each web service. Therefore, we first describe the initial population and the fitness function defined for the problem, followed by the introduction of the selection algorithm.

Initial population: Each member of the initial population, which is the result of simplifying the composition graph by replacing the normalized candidate, is defined as a pair of $Spop = (Index, Quality)$. Here, $Spop.Index$ is a vector with dimension z, where z is the number of abstract services. Each dimension of this vector represents the index number of the available candidate for that abstract service. Initially, for each dimension, a candidate service is randomly selected from the service store. After selecting the candidate, the composition algorithm is invoked to simplify the graph, meanwhile, the quality values of the graph are also updated and $Spop.Quality$ is set. This variable is defined as $Spop.Quality = (A, R, C)$, where A, R, and C are the values of availability, response time, and cost resulting from graph simplification, respectively.

Fitness function: After determining the population members, the selection algorithm needs a fitness function to identify the best members. The fitness function for the NSGAII algorithm is defined based on $Spop.Quality$. For any two members, i and j, in the $Spop$ set, member i overcomes member j if $Spopj.Quality < Spopi.Quality$. The crowding distance for member i is defined in Eqs. (1) and (2). The value of parameter k in this problem ranges from 1 to 3.

$$d_{i,k} = \frac{Spop_{i+1}.Quality_k - Spop_{i-1}.Quality_k}{Spop_{Npop}.Quality_k - Spop_1.Quality_k} \quad (1)$$

$$CD_i = \sum_{k=1}^{3} d_{i,k} \quad (2)$$

In the selection algorithm, after finding the fitness of each member, the population members (solutions), which include binary pairs $(Index, Quality)$, are sorted and additional members are removed.

3.4 Selection algorithm

In the initial step, we invoke the composition algorithm to find the initial members of the population. The second step checks the structure of the graph post-simplification; if it is a node, the NSGAII algorithm is applied. If this structure is a tree, the algorithm is invoked separately for each path, as many times as it is repeated. This algorithm uses mutation and crossover to create a new population, and the defined fitness function to find the best members. In the third step, we calculate the average quality parameters for each remaining node or a set of paths obtained from the graph.

4 Results

To evaluate the proposed method, the web-service composition graph (Fig. 3) is considered, which consists of 22 web services. Additionally, the QWS dataset [1] is used as a candidate for each abstract web service. We consider three parameters, availability, response time, and cost, as well as two quality indicators, Spread and IGD, to compare the proposed method with other methods. To this end, we provide brief definitions of these indicators, describe the graph simplification steps, and analyse the results of comparing the proposed method with other methods. The proposed method is implemented using MATLAB 2016 on a system with a Core i7 processor, 32 GB of RAM, and Windows 10 operating system.

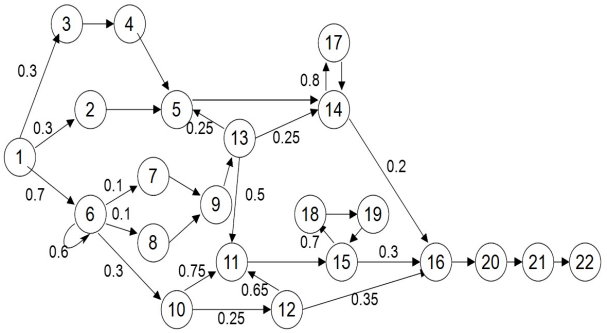

Fig. 3. Web service composition

The spread metric shows the average distance between the solutions obtained in the Pareto front [3]. A smaller average distance among the solutions signifies a better distribution of data. The IGD metric indicates the average distance between the solutions on the optimal Pareto front and those obtained from the algorithm [3]. A smaller distance demonstrates that the solutions obtained from the proposed algorithm are closer to the optimal solutions.

The steps for simplifying the web services composition graph are illustrated in Fig. 4 to Fig. 8. In general, two methods are used to simplify graphs with conditional patterns: the probabilistic method and the proposed method. In the probabilistic method, after extracting the paths, each path is initially simplified according to the sequence pattern. Next, the probabilities of each path are multiplied by the qualitative values of the remaining nodes in that path to simplify the graph to a single node. Then the selection algorithm is executed for the entire graph. In the proposed method, instead of multiplying the probabilities by the qualitative values for each extracted path, the algorithm is executed for each path as many times as it is repeated. Therefore, in the method presented in this paper, the probabilities of each path are converted into the number of executions for each

[1] https://qwsdata.github.io/

path, and the selection algorithm is implemented separately for each path. In this approach, the number of executions of each path plays a crucial role in determining the optimal solution without the need to multiply probabilities by qualitative values. Moreover, this method facilitates the selection of the best path from a set of paths. Fig. 4 and Fig. 5 report the average parameters for the first path after 30 repetitions, the second path after 4 repetitions, and so forth. The results show that the first path is the best path to choose.

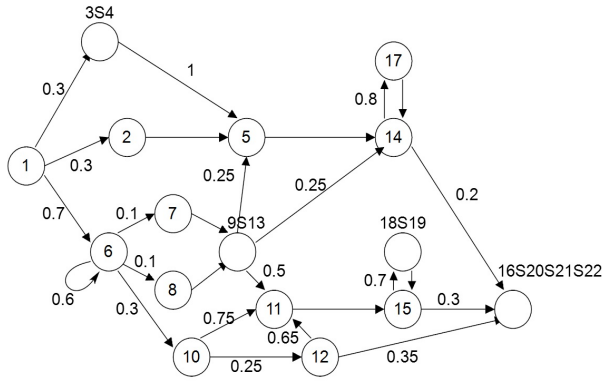

Fig. 4. Removing the sequence patterns

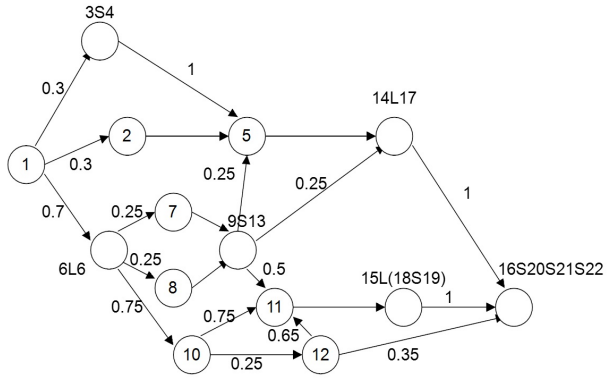

Fig. 5. Removing the loop patterns

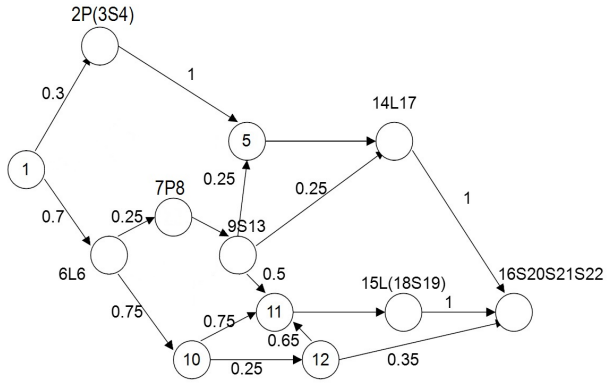

Fig. 6. Removing the parallel patterns

The proposed method is compared with the hybrid-based approach presented in [6], and the results are

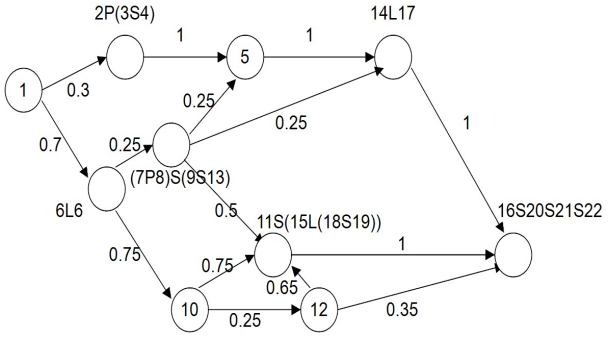

Fig. 7. Removing the sequence patterns

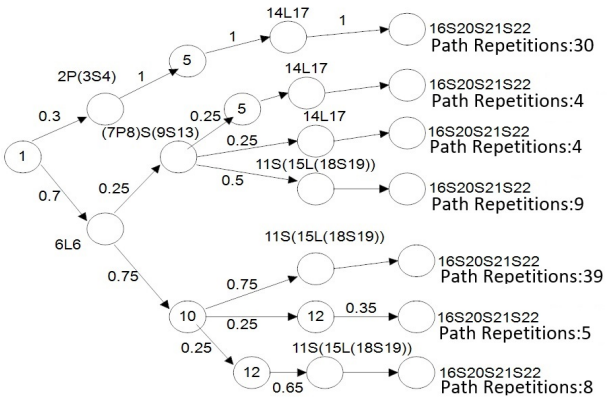

Fig. 8. Path extraction with the proposed algorithm

detailed in Table 1. Moreover, Table 2 compares the results of the proposed method with the findings of reference [15]. In Table 1, the selection algorithm is the same, and the composition algorithm is different, while in Table 2, both the composition and selection algorithms are different. The results demonstrate that the proposed method outperforms all the compared methods in terms of reliability while having worse performance in response time. Furthermore, its performance is better in the spread compared to the first and second methods, as well as in IGD compared to the second method.

Table 1. The comparison of the proposed method and the hybrid-based method, where the selection algorithm is NSGAII.

Composition Method	Availability	Response Time	Cost
Hybrid-based with probability	0.52	0.38	0.017
Hybrid-based with repetition	0.69	0.22	0.010

5 Conclusion

With the rapid growth of user complex requests, the need to compose web services in order to reuse and create powerful flexible services is intensified. Therefore, there is an increasing requirement not only for understanding various complex patterns in web service composition but also for methods that can simplify these compositions and identify the best candi-

Table 2. The comparison results, where the composition algorithm is node-based.

Selection Method	Objectives		Indicators	
	Reliability	Response Time	IGD	Spread
NSGAII	-	-	0.015	0.563
MOPSO	-	-	0.027	0.610
GA with AHP	14.1	1408	-	-
PSO with AHP	14.4	1648	-	-
NSGAII with AHP	28.7	1224	-	-
NSGAII-AHP	24.6	1131	-	-
MOPSO with AHP	19.7	1190	-	-
MOPSO-AHP	19.3	1036	-	-
Proposed method	58	1103	0.464	0.016

date. In this paper, we introduce two new patterns in the web service composition (i.e., nested and parallel loops) and then propose a method aimed at simplifying the composition to improve the overall quality and accuracy of the final outcomes. We used the NSGAII evolutionary algorithm for the optimal candidate selection.

Conflict of interest

The authors declare that they have no conflict of interest.

References

[1] A. Rasoolzadegan and m. Basiri, "The quantitative measurement of quality in service-oriented software engineering: Methods, applications, and issues," *Soft Comput. J.*, vol. 3, no. 1, pp. 2–19, 2014, dor: 20.1001.1.23223707.1393.3.1.54.5 [In Persian].

[2] M. Dehghani and S. Emadi, "Developing a new model for governance maturity of service oriented architecture," *Soft Comput. J.*, vol. 4, no. 2, pp. 54–67, 2016, dor: 20.1001.1.23223707.1394.4.2.57.7 [In Persian].

[3] N. Zahiri and S. M. Babamir, "A method to simplify patterns in web services compositions and select optimal probabilistic composition," *Soft Comput. J.*, vol. 9, no. 2, pp. 44–71, 2021, doi: 10.22052/scj.2021.243188.1003 [In Persian].

[4] F. Chen, R. Dou, M. Li, and H. Wu, "A flexible qos-aware web service composition method by multi-objective optimization in cloud manufacturing," *Comput. Ind. Eng.*, vol. 99, pp. 423–431, 2016, doi: 10.1016/J.CIE.2015.12.018.

[5] A. Strunk, "Qos-aware service composition: A survey," in *8th IEEE European Conference on Web Services (ECOWS 2010), 1-3 December 2010, Ayia Napa, Cyprus*, A. Brogi, C. Pautasso, and G. A. Papadopoulos, Eds. IEEE Computer Society, 2010, pp. 67–74, doi: 10.1109/ECOWS.2010.16.

[6] H. Zheng, W. Zhao, J. Yang, and A. Bouguettaya, "Qos analysis for web service compositions with complex structures," *IEEE Trans. Serv. Comput.*, vol. 6, no. 3, pp. 373–386, 2013, doi: 10.1109/TSC.2012.7.

[7] Z. Brahmi, "Qos-aware automatic web service composition based on cooperative agents," in *2013 Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, Hammamet, Tunisia, June 17-20, 2013*, S. Reddy and M. Jmaiel, Eds. IEEE Computer Society, 2013, pp. 27–32, doi: 10.1109/WETICE.2013.1.

[8] G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani, "A lightweight approach for qos aware service composition," in *Proceedings of 2nd international conference on service oriented computing (ICSOC'04)*, 2004, pp. 1–10.

[9] S. Gohain and A. Paul, "Web service composition using pso — aco," in *2016 International Conference on Recent Trends in Information Technology (ICRTIT)*, 2016, pp. 1–5, doi: 10.1109/ICRTIT.2016.7569553.

[10] L. Li, P. Yang, L. Ou, Z. Zhang, and P. Cheng, "Genetic algorithm-based multi-objective optimisation for qos-aware web services composition," in *Knowledge Science, Engineering and Management, 4th International Conference, KSEM 2010, Belfast, Northern Ireland, UK, September 1-3, 2010. Proceedings*, ser. Lecture Notes in Computer Science, Y. Bi and M. Williams, Eds., vol. 6291. Springer, 2010, pp. 549–554, doi: 10.1007/978-3-642-15280-1_50.

[11] Y. Huo, P. Qiu, J. Zhai, D. Fan, and H. Peng, "Multi-objective service composition model based on cost-effective optimization," *Appl. Intell.*, vol. 48, no. 3, pp. 651–669, 2018, doi: 10.1007/S10489-017-0996-Y.

[12] D. Ardagna and B. Pernici, "Global and local qos guarantee in web service selection," in *Business Process Management Workshops, BPM 2005 International Workshops, BPI, BPD, ENEI, BPRM, WSCOBPM, BPS, Nancy, France, September 5, 2005, Revised Selected Papers*, C. Bussler and A. Haller, Eds., vol. 3812, 2005, pp. 32–46, doi: 10.1007/11678564_4.

[13] Y. Yao and H. Chen, "Qos-aware service composition using nsga-ii$_1$," in *Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human (ICIS 2009), Seoul, Korea, 24-26 November 2009*, ser. ACM International Conference Proceeding Series, S. Sohn, L. Chen, S. Hwang, K. Cho, S. Kawata, K. Um, F. I. S. Ko, K. Kwack, J. H. Lee, G. Kou, K. Nakamura, A. C. M. Fong, and P. C. M. Ma, Eds., vol. 403. ACM, 2009, pp. 358–363, doi: 10.1145/1655925.1655991.

[14] P. Sharifara, A. Yari, and M. M. R. Kashani, "An evolutionary algorithmic based web service

composition with quality of service," in *7'th International Symposium on Telecommunications (IST'2014)*, 2014, pp. 61–65, doi: 10.1109/ISTEL.2014.7000670.

[15] L. Liu and M. Zhang, "Multi-objective optimization model with AHP decision-making for cloud service composition," *KSII Trans. Internet Inf. Syst.*, vol. 9, no. 9, pp. 3293–3311, 2015, doi: 10.3837/TIIS.2015.09.002.

[16] S. C. Sadouki and A. Tari, "Multi-objective and discrete elephants herding optimization algorithm for qos aware web service composition," *RAIRO Oper. Res.*, vol. 53, no. 2, pp. 445–459, 2019, doi: 10.1051/RO/2017049.

[17] F. Dahan, W. Binsaeedan, M. Altaf, M. S. Al-Asaly, and M. M. Hassan, "An efficient hybrid metaheuristic algorithm for qos-aware cloud service composition problem," *IEEE Access*, vol. 9, pp. 95 208–95 217, 2021, doi: 10.1109/ACCESS.2021.3092288.

[18] F. Dahan, K. M. E. Hindi, A. Ghoneim, and H. Alsalman, "An enhanced ant colony optimization based algorithm to solve qos-aware web service composition," *IEEE Access*, vol. 9, pp. 34 098–34 111, 2021, doi: 10.1109/ACCESS.2021.3061738.