I. *A METHOD TO SIMPLIFY PATTERNS WITH PROBABILISTIC STRUCTURE IN WEB SERVICE COMPOSITION*

Narjes Zahiri
PhD student, Department of Software Engineering,
University of Kashan, Nargess.zahiri@gmail.com

Seyed Morteza Babamir*(correspondence author)
Professor, Department of Software Engineering,
University of Kashan, babamir@kashanu.ac.ir

***Abstract*— One of the challenging issues in the field of web services is web service composition. A hybrid model of web services is the interaction between a number of web services for a specific purpose. For each web service, there are candidate services with the same function but different quality parameters. So far, many methods have been presented for the optimal web service composition, but rarely a method has been presented for web services composition with complex probabilistic structures. In this article, we first identified two new patterns of nested and parallel loops. Then, in order to help the optimal candidate selection, a new method was proposed to simplify the web services composition. In this method, the web-service composition graph is divided into two ways: 1) web-service composition graph without a conditional pattern and 2) web-service composition graph with the conditional pattern. In graphs with conditional patterns, the probability of each path is converted to the number of repetitions of that path, and then the NSGAII algorithm is used to determine the best path in the graph and find more realistic solutions, which on average leads to improvement of quality parameters of availability by 17 percent, response time by 16 percent and cost by 0.07 percent compared to the probabilistic simplification method.**

Keywords*— Web service composition graph simplification, optimal selection of web services, complex probabilistic structures, evolutionary algorithms, NSGAII algorithm

II. INTRODUCTION

In In recent years, web service has developed rapidly as a computing model [1,2]. Each web service with a specific function needs to communicate with other web services to respond to the complex needs of users [3,4]. The number of web services that have the same function but the different quality has increased, so optimal selection and composition of web services is one of the most important and complex topics in this field [5].

In a graph of web services, for each service (which is called an abstract service), there are many candidate services with the same function but different quality parameters. The most important problem is to select a candidate service for each abstract service in graphs with probabilistic patterns. To solve this problem, the graph was simplified to convert: (1) a node or (2) a tree which is consist of separate paths [6]. Graph simplification methods are divided into three categories: 1) Node-based method that we just have a node after removing sequence, parallel, conditional and loop patterns. 2) path-based method that extracts all paths between two nodes of the beginning and end of the graph. 3) Hybrid method which is a combination of the two mentioned methods. In this method, to solve the problems of two node-based and path-based methods, they can be combined [6]. In the hybrid method, sequence, parallel, and loop patterns are removed by the node-based method, but conditional patterns remain for 1) to preserve the probability of each graph path and 2) to prevent the reduction of the accuracy of the algorithm (multiplying the probability of a transition in the quality parameters of the nodes are not performed). This article uses the third method [3,6]. Since graph simplification depends on the quality of the candidate services in addition to the patterns that make up the graph, so in the following, we will discuss the quality parameters of web services and the problem of their optimal selection.

The quality of web services includes non-functional features such as execution cost, execution time, and availability of the web service. The quality features of the web service are classified into two categories. Some of these features are negative and should be minimized, such as response time and cost. The second category is features that are positive and should be maximized, such as availability [7, 8].

In the approaches presented in the articles [9, 10, 11], although they provide a solution for the optimal selection of candidate web services, but: (1) only the sequence pattern in the web services composition graph is considered, and 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 it has problem number 2. In the approaches presented in [13, 14, 15], they consider all kinds of patterns in web service composition graph and also support the probabilistic conditional pattern and repeated loops, but they do not support probabilistic parallel and loop patterns and also have problem number 2. Because our focus is on supporting various types of web service composition graph structural patterns and its simplification methods, therefore, the most complete approach is presented in the article [6], which we consider as a 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. But: (1) In this approach, the pattern of nested and parallel loops is not intended. (2) The problem is a constraint satisfaction problem and not an optimization problem. (3) A non-evolutionary algorithm was used to select the optimal candidate services, so it is not scalable in terms of time. (4) After simplifying and finding the probability of each path, has analyzed each path separately, which is also an obstacle to give a definitive answer to the customer. In this article, we have discussed these 4 cases. Therefore, the innovations of this article are:

A) Supporting all types of patterns and probabilistic transitions between services, as well as the introduction of two new patterns of parallel and nested loops.

B) Using an evolutionary algorithm for optimal candidate selection to maintain scalability.

C) Providing a hybrid method based on repeating the path to respond more accurately to the customer.

III. RELATED WORKS

This section is divided into several sub-sections based on the methods of simplifying the patterns in the web services composition graph.

A. Simplifying the web services composition graph with the node-based method

Yuji Yao et al. (2009) used the NSGAII to find optimal solutions in graphs with complex patterns and compared it with the GA in terms of the convergence speed and generation of optimal solutions [13]. Li et al. (2010) used the NSGAII to find optimal solutions in the web service composition by considering 10 objective functions and they evaluated their algorithm by reporting the number of dominant and non-dominant solutions [10]. Parvin Sharif-Ara et al. (2014) have presented a method that can find the optimal solution at a high speed. In this method, first, the problem becomes a single-objective problem and to increase speed, accuracy, and reliability, a combination of GA and fuzzy method is used. Then, it has become a multi-objective problem, which is solved by the NSGAII [14]. Li Liu et al. (2015), have compared NSGAII, MOPSO, a hybrid of NSGAII and hierarchical analysis, a hybrid of MOPSO and hierarchical analysis with NSGAII and MOPSO which finally used hierarchical analysis in their solutions [15]. Sura Gohain et al. (2016) have used the hybrid algorithm of ACO and PSO by converting the five objective functions of reliability, availability, throughput, cost, and response time into one objective [9]. Ying Huo et al. (2017) have presented the elitist multi-objective bee colony algorithm, which consists of the bee colony and NSGAII. This algorithm is better than NSGAII and PSO and bee colony in terms of spread, GD, and execution time [11]. Chibani Sadouki et al. (2019), have used the discrete multi-objective elephant algorithm. They are also inspired by the Pareto approach and SPEAII sorting method. By comparing this algorithm with the PSO and SPEAII algorithms, it was concluded that it is significantly better in terms of coverage ratio, spread, and hypervolume [16]. Fadal Dahan et al. (2021), Introduced a hybrid algorithm, which combines ACO and GA. The GA is used to tune the ACO's parameters automatically and the ACO adapts its performance based on the parameters tuning. The main contribution of their work is to help the ACO algorithm to avoid stagnation problem. The Experimental results have been shown that the proposed method called GACSCSC requires more CPU time compared to other methods, but it is better in terms of cost, response time, reliability and availability [17]. Fadal Dahan et al. (2021), proposed a new method named Enhanced Flying Ant Colony Optimization (EFACO) in three main ways. First, avoiding the execution time problem, by restricting the flying process to only occur when there are improvements in solution quality. Second, applying a neighboring selection method to avoid scanning all the neighboring nodes. Both of these decrease the solutions quality. At last, they introduced the third modification for overcoming the drawback of the first, and second developments, which transformed the algorithm into a multi-pheromone algorithm. The results demonstrate that EFACO outperforms the other methods in terms of solutions Quality and execution time [18].

B. Simplifying the web services composition graph with the path-based method

Danilo Ardagna et al. (2005) presented a mixed integer linear programming algorithm, which was 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].

C. Simplifying the web services composition graph with the hybrid method

Huiyuan 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, and to simplify the web-service composition graph, they have used a hybrid method, which also calculates the probability for each path. Integer programming algorithm has been used for the optimal selection of candidates [6].

IV. The Proposed Method

The proposed method in this article is briefly divided into two main parts: In the first part, the problem of web-service composition is discussed, in addition to identifying two new patterns, a method for simplifying the composition graph is presented, and the result of which is it will be a more accurate and definitive answer to customers. In this method, after simplifying the graph by replacing a different candidate, a population is generated (composition algorithm). In the second part, the NSGAII algorithm is used to find optimal solutions in this problem (selection algorithm). The proposed method has been implemented using MATLAB 2016 in a system with Corei 7 processor and 32 GB RAM and 10 operating system.

A. *Identification of two new patterns*

In this subsection, two new patterns have been identified.

- Parallel loop pattern: Two nodes v2 and v3 are called parallel loops if these two nodes: (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. Case b means that there are two input edges ej_2 and ej_3 as $ej_2=(v_2,v_1,p_{21})$ and $ej_3=(v_3,v_1,p_{31})$ (Figure 1).

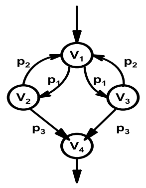

Figure 1: parallel loop

- Nested loop pattern: The sequence of nodes v_1, v_2, v_m, v_{m+1}, …, v_n form nested loop if: (a) each set of nodes v_1, v_2, …, v_m and the set of nodes v_m, v_{m+1}, …, v_n form a loop [6] and (b) these two loops have at least one shared node. Case B means that a node from the first loop (v_m) is the input node of the second loop. (Figure 2).

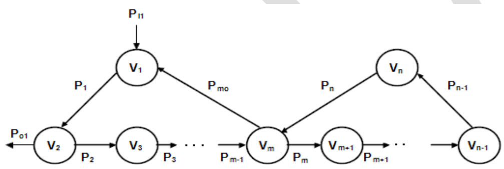

Figure 2: nested loop

B. *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 checked in such a way that the sequence pattern is checked first, then the loop, and finally the parallel pattern, and this work is repeated until none of these three patterns are present in the graph. During the simplifying of each pattern, the candidate quality values and the available possibilities are also updated (according to the tables 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 members of the population[3].

C. *Optimal selection of web services*

After simplifying the graph, in order to the optimal selection of web services the selection algorithm is called. To use this algorithm, 1) the initial population (set of solutions), and 2) the fitness function used to select the best candidate for each web service must be defined. Therefore, in the following, we first describe the initial population and the fitness function defined for the problem, and then the selection algorithm is defined.

- **The 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). Spop.Index is a vector with dimension z, where z is equal to the number of abstract services. Each dimension of this vector represents the index number of the available candidate for that abstract service. First, for each dimension, a candidate service is randomly selected from the service store. After selecting the candidate, by calling the combination algorithm, the graph is simplified, 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) that A, R, and C are respectively the values of availability, response time, and cost resulting from graph simplification.

- **Fitness function**

After determining the members of the population, the selection algorithm needs a fitness function to determine the best members. The fitness function for the NSGAII algorithm is defined using Spop.Quality. For both members, i and j of the Spop set, member i overcomes member j if $Spop_j$.Quality < $Spop_i$.Quality. The crowding distance for member i is given in equations (1) and (2). The value range of the parameter k in this problem is from 1 to 3.

$$d_{i,k} = \frac{\mathrm{Spop_{i+1}.Quality}_k - \mathrm{Spop_{i-1}.Quality}_k}{\mathrm{Spop_{Npop}.Quality}_k - \mathrm{Spop_1.Quality}_k} \quad [1]$$

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

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

- **Selection algorithm:**

First step: the composition algorithm is called to find the initial members of the population.

Second step: If the structure of the graph after simplification is a node, the NSGAII algorithm is called for it, and if it is a tree, this algorithm is called separately for each path as many times as it is repeated. In this algorithm, mutation and crossover are used to create a new population, and the defined fitness function is used to find the best members.

Third step: Calculate the average of quality parameters for each remaining node or a set of paths obtained from the graph.

V. TESTS AND EVALUATION OF RESULTS

In order to evaluate the proposed method, the web-service composition graph (Figure 3) is considered, which consists of 22 web services. Also, the QWS data set [19] has been used as the candidate for each abstract web service and we consider three parameters of availability, response time and cost, as well as two quality indicators Spread and IGD to compare the proposed method with other methods. First, we have a brief definition of these indicators, and then we describe the steps of simplifying the graph and comparing the results of the proposed method with other methods.

- **Spread:**

The spread shows the average distance of the solutions obtained in the pareto front from each other [3]. The smaller the average distance of the solutions from each other, the better the data is distributed.

- **IGD:**

The IGD measures the average distance between the solutions in the optimal pareto front and the pareto front obtained from the algorithm [3]. The smaller this distance is, the pareto front solutions obtained from the proposed algorithm are closer to the optimal solutions.

The steps of simplifying the web services composition graph have been shown in Figure 4. In general, two methods have been used to simplify graphs with conditional patterns. 1) Probabilistic method: in this method, after extracting the paths, each path is first simplified according to the sequence pattern, and then by multiplying the probabilities of each path by the qualitative values of the remaining nodes in that path, we simplify the graph to Let's reach a node. Then the selection algorithm is executed for the entire graph. 2) Proposed method: In this 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 article, the probabilities of each path have been converted to the number of execution times in each path, and the selection algorithm for each path has been implemented separately. With this method, the number of executions of each path plays a fundamental role in determining the optimal solution by the algorithm without the need to multiply the probabilities by qualitative values, also with this method, the best path from among a set of paths has been selected. In Figure 4-5, the average parameters for the first path after 30 repetitions, the second path after 4 repetitions, ... and finally the seventh path after 8 repetitions are reported. The results show that the first path is the best path to choose. The proposed method was compared with the Hybrid-based with probability presented in [6] and the results was shown in Table 1. Also, the comparison results of the proposed method with the article [15] are shown in Table 2, compared with 8 methods. In Table 1, the selection algorithm is the same and only the composition algorithm is different, but in Table 2, the composition and selection algorithm are both different. The results show that the proposed method has performed better than all 8 other methods in terms of reliability and worse than eighth method in terms of response time. It is also better in the spread than the first and second methods and in the IGD than the second method.

Table1: comparison of the proposed method and Hybrid-based with probability method

Methods		availability	Response time	Cost
Composition	Selection			
Hybrid-based with probability	NSGAII	0.52	0.38	0.017
Hybrid-based with repetition	NSGAII	0.69	0.22	0.010

Table2: comparison of the proposed method and 8 other methods

Methods			Objectives		Indicators	
Composition		Selection	Reliability	Response time	IGD	Spread
1th-mothod	Node-based	NSGAII	---	---	0.015	0.563
2th-mothod	Node-based	MOPSO	---	---	0.027	0.610
3th-mothod	Node-based	GA with AHP	14.1	1408	---	---
4th-mothod	Node-based	PSO with AHP	14.4	1648	---	---
5th-mothod	Node-based	NSGAII with AHP	28.7	1224	---	---
6th-mothod	Node-based	NSGAII-AHP	24.6	1131	---	---
7th-mothod	Node-based	MOPSO with AHP	19.7	1190	---	---
8th-mothod	Node-based	MOPSO-AHP	19.3	1036	---	---
Proposed method		NSGAII	58	1103	0.464	0.016

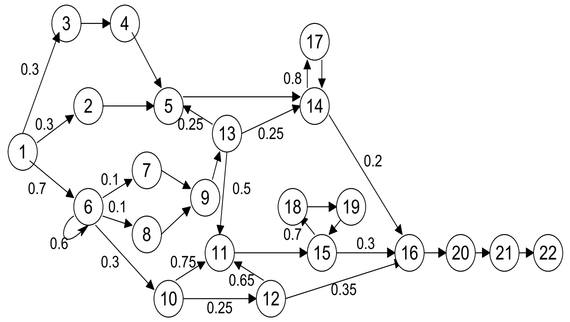

Figure 3: web service composition

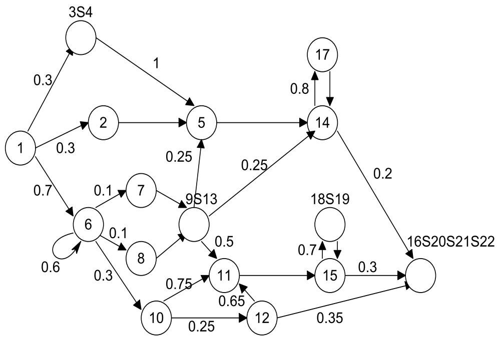

Figure 4-1: removing the sequence pattern

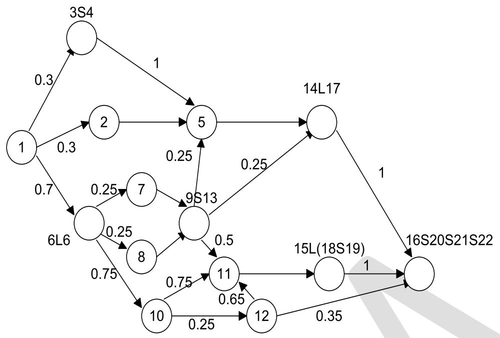

Figure 4-2: removing the loop pattern

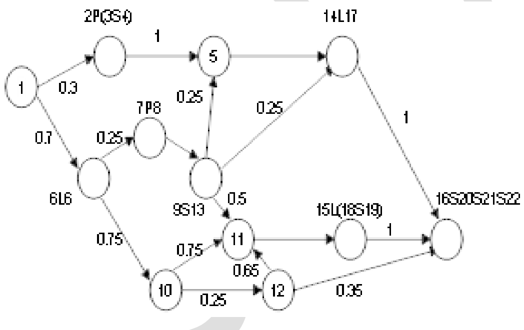

Figure 4-3: removing the parallel pattern

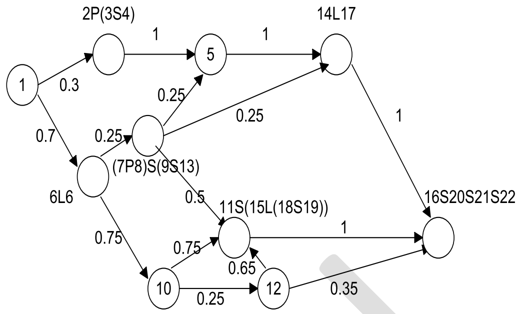

Figure 4-4: removing the sequence pattern

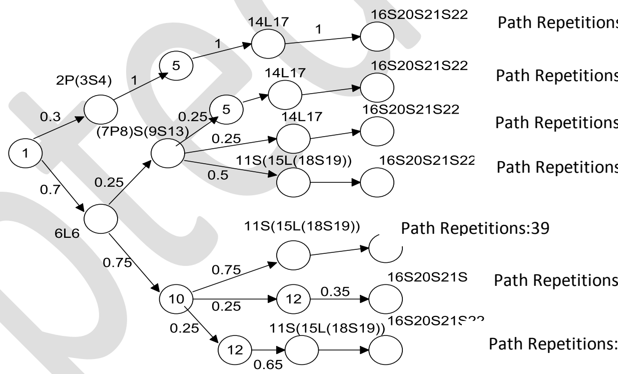

Figure 4-5: path extraction with proposed algorithm

VI. Conclusion

With the rapid growth of user complex requests, the need to compose web services in order to reuse and also create powerful flexible services is felt more than ever. Therefore, in addition to the need to know all kinds of complex patterns in the web services composition, there is a need for methods to simplify these compositions and then choose the best candidate. In this article, we introduce two new patterns in the web-service composition (nested and parallel loops) and then propose a method to simplify the composition to increase the accuracy of the final quality of the composition. We used the evolutionary algorithm which is called NSGAII for Optimal candidate selection.

References

[۱] رسولزاردگان عباس، بصیری محدثه "اندازه گیری کمی کیفیت در مهندسی نرم افزار سرویس گرا: روش ها، کاربردها و چالش ها" . نشریه علمی محاسبات نرم. ۱۳۹۳; ۳ (۱) :۲-۱۹ .

[۲] دهقانی مژگان، عمادی سیما، "ارائه یک مدلی جدید برای بلوغ حاکمیت بر معماری سرویس گرا". نشریه علمی محاسبات نرم. ۱۳۹۴; ۴ (۲) :۵۴-۶۷.

[۳] ظهیری نرجس، بابامیر سید مرتضی "روشی برای ساده‌سازی الگوها در ترکیب وب‌سرویس‌ها و انتخاب بهینه ترکیب با ساختار احتمالی" .نشریه علمی محاسبات نرم. ۱۳۹۹; ۹ (۲) :۴۴-۷۱.

[4] *Chen, F., et al., A flexible QoS-aware Web service composition method by multi-objective optimization in cloud manufacturing. Computers & Industrial Engineering, 2016. 99: p. 423-431.*

[5] *Strunk, A. QoS-Aware Service Composition: A Survey. in 2010 Eighth IEEE European Conference on Web Services. 2010..*

[6] *Zheng, H., et al., QoS Analysis for Web Service Compositions with Complex Structures. IEEE Transactions on Services Computing, 2013.* ***6****(3): p. 373-386*

[7] *Brahmi, Z. and M.M. Gammoudi. QoS-aware Automatic Web Service Composition based on Cooperative Agents. in 2013 Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises. 2013.*

[8] *Canfora, G., et al., A lightweight approach for QOS aware service composition. Proceedings of the 2nd International Conference on Service Oriented Computing, 2004.*

[9] *Gohain, S. and A. Paul. Web service composition using PSO — ACO. in 2016 International Conference on Recent Trends in Information Technology (ICRTIT). 2016.*

[10] *Li, L., et al. Genetic Algorithm-Based Multi-objective Optimisation for QoS-Aware Web Services Composition. in Knowledge Science, Engineering and Management. 2010. Berlin, Heidelberg: Springer Berlin Heidelberg.*

[11] *Huo, Y., et al., Multi-objective service composition model based on cost-effective optimization. Applied Intelligence, 2018. 48(3): p. 651-669.*

[12] *Ardagna, D., & Pernici, B. (2005, September). Global and local QoS guarantee in web service selection. In International Conference on Business Process Management (pp. 32-46). Springer, Berlin, Heidelberg.*

[13] *Yao, Y. and H. Chen, QoS-aware service composition using NSGA-II1, in Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human. 2009, Association for Computing Machinery: Seoul, Korea. p. 358–363.*

[14] *Sharifara, P., 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.*

[15] *Liu, L. and M. Zhang, Multi-objective Optimization Model with AHP Decision-making for Cloud Service Composition. KSII Transactions on Internet and Information Systems, 2015.* ***9****: p. 3293-3311.*

[16] *Chibani Sadouki, S. and A. Tari, Multi-objective and discrete Elephants Herding Optimization algorithm for QoS aware web service composition. RAIRO-Oper. Res., 2019.* ***53****(2): p. 445-459.*

[17] *Dahan, F., Binsaeedan, W., Altaf, M., Al-Asaly, M. S., & Hassan, M. M. (2021). An efficient hybrid metaheuristic algorithm for QoS-Aware cloud service composition problem. IEEE Access, 9, 95208-95217.*

[18] *Dahan, F., El Hindi, K., Ghoneim, A., & Alsalman, H. (2021). An enhanced ant colony optimization based algorithm to solve QoS-aware web service composition. IEEE Access, 9, 34098-34111.*

[19] *https://qwsdata.github.io/*