Evolutionary algorithm MCSE 662 -MCE 079 05536

Short Notes on the article titled-

Service Composition in IoT using Genetic algorithm and Particle swarm optimization

Open Comput. Sci. 2020; 10:56–64; Published by De Gruyter Open Access March 13, 2020

Research Article by -Neeti Kashyap, A. Charan Kumari, and Rita Chhikara.

https://doi.org/10.1515/comp-2020-0011

Abstract: Web service compositions are commendable in structuring innovative applications for different Internet-based business solutions. The existing services can be reused by other applications via the web. Due to the availability of services that can serve similar functionality, suitable Service Composition (SC) is required. There is a set of candidates for each service in SC from which a suitable candidate service is picked based on certain criteria. Quality of service (QoS) is one of the criteria to select the appropriate service. A standout amongst the most important functionality presented by services in the Internet of Things (IoT) based system is the dynamic composability.

In this paper, two of the metaheuristic algorithms namely Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) are utilized to tackle QoS-based service composition issues. QoS has turned into a critical issue in the management of web services because of the immense number of services that furnish similar functionality yet with various characteristics. Quality of service in service composition comprises of different non-functional factors, for example, service cost, execution time, availability, throughput, and reliability. Choosing appropriate SC for IoT-based applications in order to optimize the QoS parameters with the fulfillment of users' necessities has turned into a critical issue that is addressed in this paper. Experimental results demonstrate that GA can enhance the proficiency of solutions for SC problems in IoT. It can also help in identifying the optimal solution and also shows preferable outcomes over PSO.

Problem (The Service Composition Optimization Problem)**:**

The IoT is distributed, scalable, highly dynamic, and changeable with new types of things being added on, and individual things coming and going from the IoT. Service composition is one of the most popular problems in IoT. Furthermore, a hybrid version of multi-population GA is used to make the algorithm simple. The algorithm has simplicity due to the changing of a population into subpopulations. This algorithm considers some of the QoS parameters to find the optimal solution for the service composition problem.

To obtain results via simulation, the PSO algorithm is used to solve the SC problem in IoT. This is further assessed and contrasted with GA. Experimental results demonstrate that GA can enhance the proficiency of solutions for SC problems in IoT. It can also help in identifying the optimal solution and also shows preferable outcomes over PSO. Multiple services on the cloud from different providers can be combined as per the requirements of the user resulting in SC problems. The proposed algorithm has less running time. The stated problem is complex and comes under the category of NP-hard problems. So, the approach based on heuristics is used to solve the problem.

Service Composition Optimization Problem equations- The combination of services is referred to as service composition. The solution to this problem can be obtained by utilizing or consolidating accessible services. It encourages proficient application advancement, service reuse, and the culmination of services with limited use. The objective of SC is to locate an arrangement of services (one for every task) that adds to the composite service and addresses the user's request. There are no specific means of defining the SC that must fulfill the requirements of the user. There are various QoS attributes that are used to characterize the web services. Three main types of QoS parameters such as execution time(t), service cost(c), and reliability(r) are considered.

• Execution time (t): It is the time utilized by an IoT service to respond.

• Service cost (c): It is the cost associated with an IoT service.

• Reliability (r): It is defined as the degree of failure of an IoT service.

In light of this, the SC problem is depicted as below:

For a given set of 'n' tasks, the task is defined as the service which is required to fulfill the user requirement,

G = {g1, g2, g3 . . . , gn}

Each task has a 'k' no. of candidate services that provide similar functionality,

gi={s1i , s2i,...,ski), ∀i∈[1,2,...n]

Where 'i' defines the class of a group with similar services in accordance to their task gi.

The composite service C represents a group of tasks

C = {sa1, sb2 , . . . sk3, ∀a, b, c ∈ [1 . . . k]

The objective functions for execution time, service cost and reliability are defined as follows:

$$f_1 = \sum_{i=1}^{n} t_i \qquad (1)$$

$$f_2 = \sum_{i=1}^{n} c_i \qquad (2)$$

$$f_3 = \prod_{i=1}^{n} r_i \qquad (3)$$

Where ti is the time of the ith instance of a task. ci is the cost of an ith instance of a task. ri is the reliability of the ith instance of a task. By giving equal weight to each QoS property described in equation no. (1), (2) and (3) as three objective functions respectively can be merged into a single objective function as follows in equation no. (4).

$$f = \text{minimize}(w_1 f_1 + w_2 f_2 - w_3 f_3) \qquad (4)$$

such that $\sum_{i=1}^{3} w_i = 1$

Background:

The IoT is distributed, scalable, highly dynamic, and changeable with new types of things being added on, and individual things coming and going from the IoT. Service composition is one of the most popular problems in IoT. Prior researchers have addressed the service composition problem using Multicriteria based on QoS parameters. Furthermore, a hybrid version of multi-population GA is used to make the algorithm simple. The algorithm has simplicity due to the changing of a population into subpopulations. This algorithm considers some of the QoS parameters to find the optimal solution for the service composition problem. Some researchers have defined a method to solve the service composition problem with reduced energy consumption. The key challenge while designing the energy-efficient algorithm is to handle the interoperability of the heterogeneous devices which are integrated together via cloud-based applications. The proposed algorithm is evaluated in multiple cloud environments and has shown efficient performance

The term IoT connotes the interconnected devices which allude to an accumulation of different physical gadgets around the globe that are presently associated with the web to gather and share information. The blend of information is retrieved through different sources. IoT is an accumulation of systems and gadgets that can speak with one another. It is perceived as a standout amongst the most imperative territories of future innovation and has got wide consideration from various applications. The peculiarity of the IoT isn't in any new risky development. It is seen as the most important area among the most basic domains of future development and is expanding hugely from a broad assortment of endeavors. Many devices like cameras, television, and refrigerators could be connected to the world via the Internet. The devices become part of the IoT by providing some information that can further be utilized by some applications.

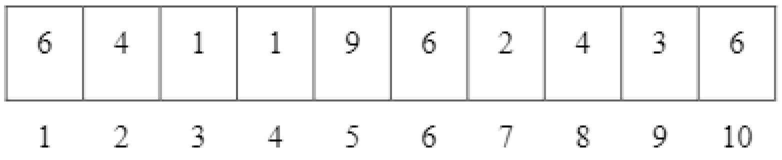
These devices can be uniquely identified with the help of RFID tags. For example, an IoT-based application that is part of the Smart Home can keep track of the devices which are switched on and their power consumption at a particular time. Few IoT devices offer the same service with completely different quality parameters like high reliability, low response time, and low cost. These days cloud foundations give a section point to revelation, determination, combination, and devouring for such appropriate IoT services. Thus, another sort of middleware administration ought to be conceived by a cloud to choose and create the desired services dependent on the end client's requirements. In the current literature on optimization in engineering applications, the nature-inspired algorithms have shown promising performance and thus these are popularly and widely used to solve various problems. The contributions of the paper are briefly described as follows:

1. GA and PSO are applied to solve the service composition problem in IoT. These meta-heuristic search algorithms are used to optimize the service composition multi-objective problem into a single objective problem by applying equal weights to all three QoS parameters.
2. The implemented algorithms are compared using the fitness value and time. Fitness value plays a key role in comparing their performance.
3. The effectiveness of implemented algorithms has been illustrated using a service composition application scenario in IoT environments with the real-world application data set.

The purpose of this research was to compare the two most popular metaheuristic search algorithms GA and PSO for single-objective optimization to solve the multi-objective problem of service composition in IoT using QoS parameters such as execution time, service cost, and reliability.

Methodology:

The solution to this problem is to include the possible set of tasks with minimum service cost, minimum execution time, and maximum reliability of the service. Therefore, each chromosome represents a possible composition of the service. Each solution is coded as a series of integers as an array of integer type. Indices of an array represent the task number and the data indicate the candidate number for the task. The coding process is shown schematically below:



6	4	1	1	9	6	2	4	3	6
1	2	3	4	5	6	7	8	9	10

Figure 1: Solution Encoding

The above instance shows a string of solutions with 10 tasks and 10 candidates for executing each task. In this service composition, the sixth candidate for the first task is shortlisted, the fourth candidate for the second task is shortlisted, and the first candidate for the third task is shortlisted. To compose the set of most appropriate services, two of the metaheuristic algorithms namely GA and PSO has been used. The fitness value for each row of the solution set is computed and the minimum value corresponding to the composing services is selected.

The primary reason for optimization is to utilize the assets in the best way. This implies maximizing or limiting characteristics dependent on a function that in a given circumstance is additionally called an objective function from the scope of accessible qualities. In view of this, we can pick the best solution. The objective might be to save expenses, augment benefits, limit endeavors, etc.

Algorithm 1: Genetic Algorithm:-
Step 1: Initialize the population randomly (P)
Step 2: Compute fitness value corresponding to P using a
suitable fitness function
Step 3: Repeat until the best fitness value is found
Step 4: Select parents P1 and P2 from P
Step 5: Crossover is performed on parents resulting in the population (P + 1)
Step 6: Mutation operator is applied to population (P + 1)
Step 7: Compute the fitness of the population (P + 1) and then compare it with the best fitness value, update the best value if the fitness value obtained is better than the best value.

Particle Swarm Optimization (PSO) is a stochastic optimization method applied to a population and is influenced by the social behavior of birds and fish schools. It consists of aggregate particles moving around the search space affected by their best location in the past and the best location of the past of all nearby or close neighbors. Each repetition of particle speed is updated using:

$$V(t+1) = wV(t) + c_1 \times rand() \times (X_{pbest} - X(t)) \quad (5)$$
$$+ c_2 \times rand() \times (X_{gbest} - X(t))$$

The new location of the particle is calculated as:

$$X(t+1) = X(t) + V(t+1) \quad (6)$$

Where X(t+ 1)denotes a new position of the particle. V(t)defines the velocity of the particle. rand is the random number generated between 0 and 1. ‘W’ inertia weight has been added to control the velocity, ci and c2 are the learning factors to computeXpbestparticle’s personal best position andXgbestglobal best position. The algorithm (below) describes the PSO

Algorithm 2: Particle Swarm Optimization

```
Step 1: Randomly initialize position Xi(0) and velocity of particles Vi(0)
Step 2: Repeat for all number particles
Step 3: Compute the fitness value using equation no. (4)
Step 4: ComputeXpbestparticle’s personal best positionandXgbestglobal best position
Step 5: Update velocity using equation no. (5)
Step 6: Calculate the position using equation no. (6)
Step 7: until the best value is found
```

Table 1: Parameter settings

Population size	100
No. of generations	100
GA	
Parent selection Method	roulette wheel
Type of Crossover	one-point
Value of Crossover probability	0.8
Type of Mutation	Random
Mutation probability	0.1
PSO	
W(inertia weight)	1
c1, c2(acceleration coefficients)	2
Maximum no. of iterations	20

Experimental settings:

The three objective functions defined in equations (1), (2), and (3) have equal weight. For both algorithms, the population size is 100 and the algorithm is executed with an evaluation function of up to 10,000 to guarantee the equivalent basis for its performance. In GA, random mutations have been applied to maintain individual diversity by changing the value of randomly selected genes with a probability of 0.1. The parent selection mechanism is a roulette wheel along with the one-point type of crossover. The crossover probability is taken to be 0.8 as we get better results on this value. In PSO, the value of inertia weight is taken as 1 acceleration coefficients value of 2.

Data settings- As a part of the experiment, the number of tasks ranged from 10 to 30 in steps of 10. For each task ‘n’, ‘m’ is the no. of candidates having different QoS parameters. The value of n ranges from 10 to 30. The QoS parameters have different ranges. The range of service costs is from 1 to 100, execution time varies from 0.1 to 0.3 and reliability is between 0.7 and 0.9.

$1 \leq$ execution cost ≤ 100 …. (7)

$0.1 \leq$ execution time ≤ 0.3 … (8)

$0.7 \leq$ reliability ≤ 0.9 …..(9)

Experimental Results:

The implemented algorithms are executed using real-world data sets in order to determine the efficacy of the SC solution in IoT. The empirical analysis of the real data set has been implemented using two QoS parameters namely response time and reliability for 10 tasks and 40 candidates. Both algorithms are run 30 times and the average value with the standard deviation of fitness values is obtained. The lesser value of the execution time and more value of reliability are the indicators of better performance.

Table 3: Comparison of fitness values obtained by both algorithms

	Parameter: Response Time	
Algorithm	**Mean**	**Std. Dev**
GA	1.24E−02	3.49E−03
PSO	6.15E−03	3.74E−01
	Parameter: Reliability	
Algorithm	**Mean**	**Std. Dev**
GA	1.77	3.39E−03
PSO	0.004284	4.96E−01

Table 3 results show the superiority of GA is more prominent as the candidates are 40 which is large for a composition of 10 services. As it can be seen the average value obtained for execution time is less and the value for reliability is more thus experimental results validate that the results obtained using GA are better than the PSO. The results illustrate that the usage of metaheuristic search algorithms to obtain the solution for SC problem in IoT are feasible and effective in providing the solution. The performance of the algorithms depends on the fitness values. The fitness value is an important component to compare the performance of both algorithms. Since the calculations are stochastic so, to represent their inherent arbitrariness, the algorithms are run 30 times and the average value with the standard deviation of fitness values is obtained. The fitness values are analyzed using a t-test with 95% confidence.

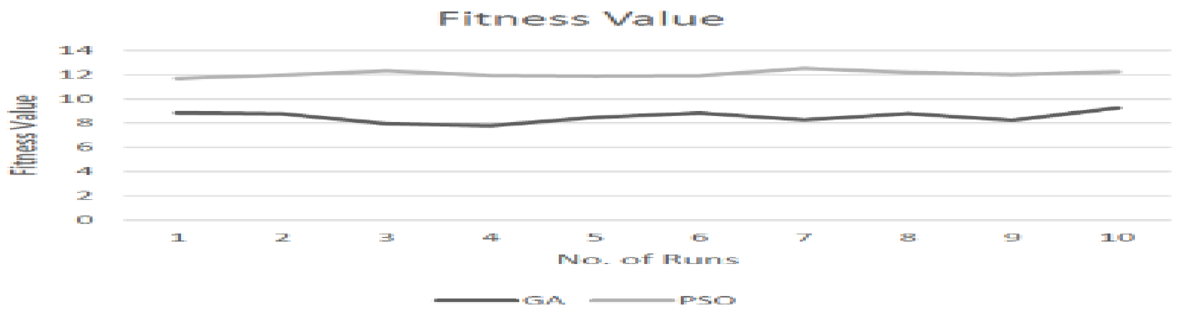


Figure 2: Comparison of Fitness values for 30 tasks and 20 instances of each task

Figure 2 shows the comparison of fitness values for 30 tasks and 20 instances of 10 runs. It can also be seen that GA converges to a better minimum value than PSO.

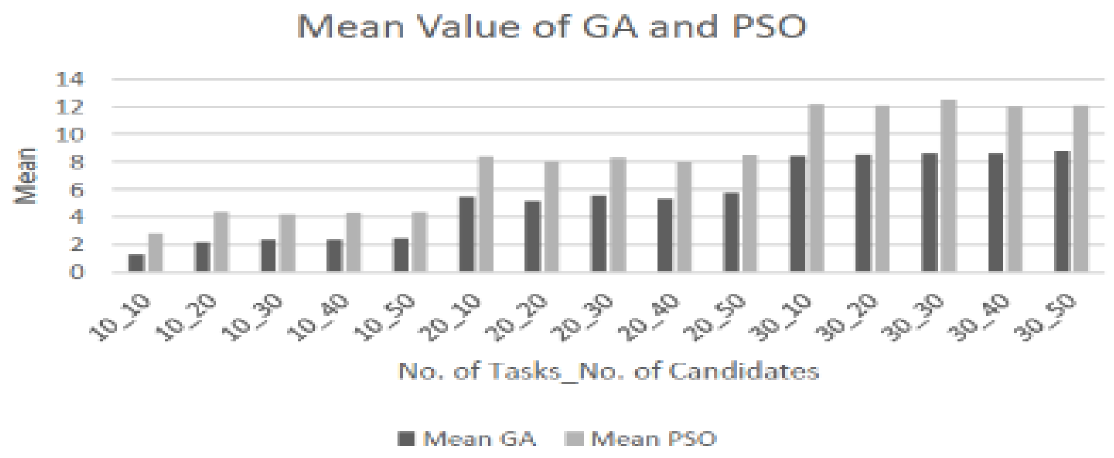


Figure 3: Comparison of Mean value of Fitness function for varying no. of tasks and their instances

As shown in Figure 3, the fitness value obtained using GA is always lesser than that obtained using PSO for any no. of tasks ranging from 10 to 30 with any no. of candidates ranging from 10 to 50 with a step size of 10 for service composition problems in IoT. Additionally, the mean and standard deviation of the fitness obtained by the algorithms in each test instance along with the p-value (of t-test) is shown in Table 2. The p-value obtained for each case is less than 0.05, which indicates that there is a statistically significant difference between the fitness values obtained using both algorithms. Figure 3 shows a comparison of the Mean value of the Fitness function for varying no. of tasks and their instances.

Table 2: Comparison of the fitness values obtained by the algorithms

No. Tasks	No. of Candidates	GA		PSO		
		Mean GA	Std_GA	Mean PSO	Std_PSO	P-value
10	10	1.3074	0.1732	2.7942	0.157	8.74E-14
10	20	2.1869	0.3085	4.378	0.2065	3.16E-13
10	30	2.3831	0.2709	4.2213	0.1611	3.89E-13
10	40	2.3906	0.3897	4.2965	0.1807	3.91E-11
10	50	2.4936	0.1851	4.3747	0.107	3.04E-16
20	10	5.4635	0.311	8.3558	0.2251	4.60E-15
20	20	5.1382	0.4612	8.0456	0.2719	1.31E-12
20	30	5.5686	0.3737	8.2768	0.1727	4.88E-14
20	40	5.3205	0.3002	8.0194	0.147	1.37E-15
20	50	5.7806	0.3359	8.4911	0.1099	3.38E-15
30	10	8.3923	0.4337	12.1446	0.1973	2.13E-15
30	20	8.5143	0.4529	12.0583	0.2434	2.17E-14
30	30	8.6302	0.7546	12.5006	0.2278	7.22E-12
30	40	8.6247	0.6114	12.0243	0.2338	2.80E-12
30	50	8.7545	0.3954	12.0655	0.2634	1.80E-14

From Table 2, it is clear that the GA fitness value is four times better than the PSO's. As a result, we found that GA can find a better solution for SC problems than PSO in terms of reliability, cost and execution time.

Conclusions:

The role of optimization algorithms using empirical evaluation of GA and PSO is shown to solve the problem of service composition in IoT. The problem was to minimize service cost, execution time, and reliability and select candidates to execute each task in the service configuration. The experiment is executed for a step size of 10 with the number of tasks ranging from 10 to 30, and the number of candidates for each assignment is also in the range of 10 to 50 with the same step size. The results of the experiments obtained with the algorithms generate 15 test data instances. Also, statistically significant results are obtained with a 95% confidence level. The results obtained after the empirical evaluation of the implemented algorithms on the real-world application data set revealed that the solutions obtained using GA are more eloquent than that of PSO. As part of future research, we will try to hybridize the genetic approach with other metaheuristic search methods. It shall be also applied to the datasets from various other applications based on IoT.