Quest Journals
Journal of Software Engineering and Simulation
Volume 2 ~ Issue 1 (2014) pp: 04-12
ISSN(Online) :2321-3795 ISSN (Print):2321-3809
www.questjournals.org

Research Paper

Dynamic configuration of optimal web services composition based on the quality

Samira Zirak[1], Naser Nematbakhsh[2], Kamran zaminfar[3]

Received 20 December, 2013; **A**ccepted 30 January, 2014 © The author(s) 2013. Published with open access at **www.questjournals.org**

ABSTRACT:- *Nowadays, some special attributes of web services such as being independent from platform, reusability, having a loosely coupled architecture and the ability to compose together in contrast to other applications have made them as components enjoying the capability of supporting various changes. Therefore, most organizations tend to use these application components in order to provide their customers with better services. Furthermore, the use of a single service cannot meet the needs of most customers. Consequently, the issue of composing these services aimed at increasing their efficiency is one of the important issues. As the number of these services is increasing day to day, the process of selecting and composition appropriate web services in terms of their quality and users' need from among numerous single services having same functionality but different quality attributes such as cost, response time, reliability and availability has become one of the main challenges associated with composing these web services.*
To solve this problem, much works have been done so far in various ways. In recent years, most of these studies have used a variety of heuristic algorithms or a combination of them. The reason for this is the acceptable running time of these algorithms to find a solution close to the optimal solution. Accordingly, the proposed methodology for this research is based on particle swarm heuristic algorithm inspired by patterns of birds. The main reasons to choose this algorithm include its flexibility, less parameters, easy to implement and low cost. In contrast, the disadvantage of this algorithm is premature convergence that has been tried to be solved by adding two functions of Inertia Coefficient Adjustment and Particle Modification to main algorithm. Following the N iterations of the algorithm, the inertia coefficient adjustment function is called. Modifying the amount of inertia coefficient, the function will be able to control how the search is performed and to decrease the running time. As the running process continues, the second function used to modify the particles aimed at improving them is called following the M iterations of the algorithm. It tries to prevent algorithm being trapped in a local optimum. Applying this function, some web services on top particles which do not satisfy the desired quality features of users will be replaced by a number of services on the dataset. This is done on condition that the services on the dataset are better than services on the particle. If the algorithm gets trapped in a local optimum, this function makes the algorithm to perform the search in a new sample space. The efficiency of this method is evaluated in a simulated environment and is compared with both conventional birds and genetic algorithms. The results indicate that the proposed method is more effective than conventional algorithms in terms of running time and success rate as well as addressing the problem of being trapped in a local optimum.

Keywords:- *particle swarm optimization, quality of service , services composition, web service*

I. INTRODUCTION

In the recent years, the ability to respond quickly to users' requests has become one of the main challenges most organizations facing with in competitive global marketplace. Due to frequent changes in the needs and requests of customers and environmental dynamics, organizations are forced to adapt their systems with existing conditions. Consequently, the use of Service Oriented Architecture as one of the leading architectural practices is recommended for these organizations. This is an ideal approach to integrate technology in an environment where there are a variety of software and hardware platforms [1]. Web services could be used to implement Service Oriented Architecture. The basic structure of the service used in this architecture is adequate to implement simple interactions with customers, but the diversity, complexity and variability of users' demands requires the simultaneous use of multiple services and a need to compose the services. The main

*Corresponding Author: Samira Zirak
Department of *Computer Engineering, Azad University, Najafabad Branch, Isfahan, Iran.*

problem here is to obtain the best composition in accordance with user requirements. As web services are in a dynamically changing computing environment, it is better to model the problem as an optimization problem and to apply heuristic algorithms for solving them [2]. In this research, section 2 will review related works in this field. The process of composing web services as well as the patterns to evaluate their quality then will be studied in section 3. The next section, i.e. section 4, presents particle swarm optimization (PSO) algorithm and the proposed method of this article that is an improved version of the algorithm and finally the proposed method will be evaluated in section 5.

II. RELATED WORKS

So far, many different methods have been proposed for solving the problem of web service composition based on quality attributes. According to [3], these methods can be classified into two general categories of Exhaustive and Approximate methods. The first category that is also known as non-heuristic methods calculates all of the candidate paths to choose the best plan, so it gives us a more precise solution. Two works done by Yu et al [4 ,5] can be cited as examples of these methods. These researches have used two methods for solving the problem of web service composition. In the first method, the problem was regarded as a knapsack problem in which each item (candidate service) has a weight (quality attribute) and a value and the knapsack itself has a certain capacity. The capacity of knapsack is a global constraint. The algorithm to select a service from among all candidate services tries to maximize the total services value by considering the capacity of knapsack. Knapsack method has also been suggested in the other works [6]. The second method like some other studies [7] has used graph theory in which the problem of finding the optimal composition has been transformed to a problem of finding the shortest path in a graph. Another method to solve this problem is linear programming [8] that is applicable for those problems having a linear objective function. Non-scalability and linearity of all parameters are among the main disadvantages of this method.

Researchers have shown that the first group of methods is suitable for small and simple environments or when there are a small number of candidate web services. However, as the number of web services increases and the environment becomes larger and more complex, the second group of methods including the algorithms with higher speed and less complexity will be more suitable because of some factors such as their non-scalability and exponential rise of running-time with the linear increase of services [3]. In the second group of methods, namely heuristic methods, unlike the first group, an ideal plan closest to the best and most accurate solution is selected. Paper [9] presented a Genetic Algorithm (GA) for solving the problem of web services composition. Another research [10] proposed a combination of genetic algorithm and Tabu search algorithm in order to avoid being trapped in local optimum. Other algorithms combined with genetic algorithm to improve it include Simulated Annealing [11] and Hill-Climbing [12] algorithms. Paper [13] introduced a combination of genetic and Ant colony optimization algorithms aimed to overcome the shortcomings of Ant colony algorithm and to achieve a better efficiency and faster convergence. In [14], the author proposed a method based on combining Ant colony algorithm and graph theory. An improved version of Imperialist Competitive Algorithm (ICA) was offered in the paper [15] to solve the problem of web services composition based on quality attributes. In this technique, Revolution operator has been integrated to basic algorithm in order to avoid being trapped in minimum local optimum.

III. WEB SERVICES COMPOSITION PROCESS

Most service-oriented systems are formed from combining a set of services. These services are composed together according to certain patterns. As shown in Figure 1, these patterns can be created and then replace them with a composite service. This can be continued until only one service is left. The service left is composite service.

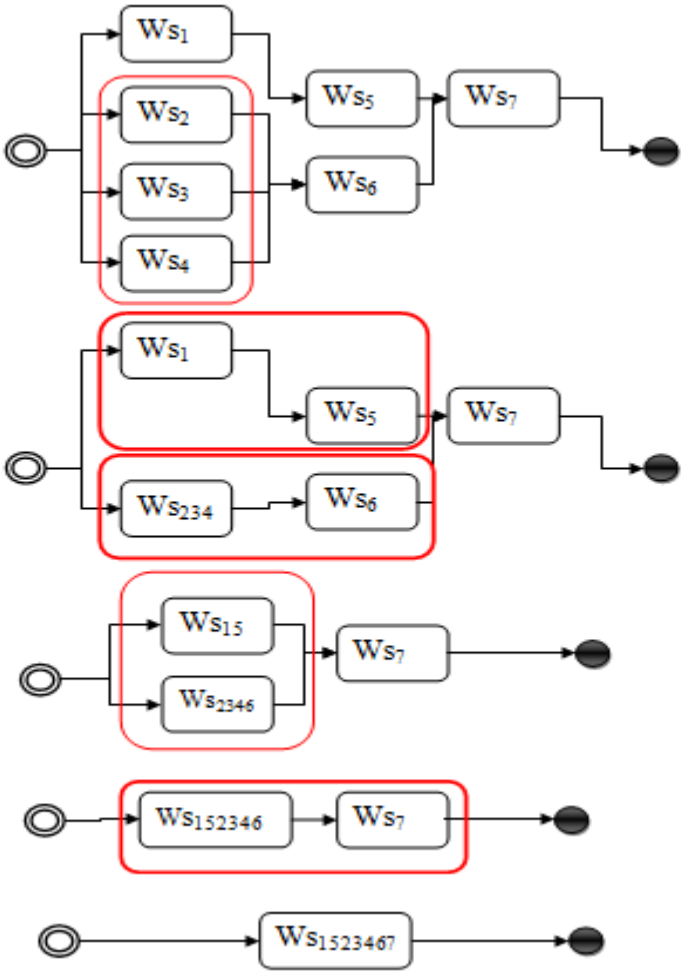

Fig 1: Formation of a composite service

To calculate the quality attributes of a composite service, its basic pattern (sequential, parallel, conditional or loop) firstly must be detected and then its quality attributes are derived in accordance with Table (1)[16].

Table 1: Formulas for calculating quality attributes of composite web services

Attributes and patterns	loop	conditional	parallel	sequential
Response time	n*Rt(x1)	max{ Rt(x1) ... Rt(xn-1)}+ Rt(xn)	Rt(x1)+ max{ Rt(x2) ... Rt(xn)}	Rt(x1) + Rt(x2)
Cost	n*P(x1)	P(x1) +……+ P(xn)	P(x1) +……+ P(xn)	P(x1) + P(x2)
Accessibility	A(x1) ^ n	A(x1) *……* A(xn)	A(x1) * …… *A(xn)	A(x1) * A(x2)
Reliability	R(x1) ^ n	R(x1) *….* R(xn)	R(x1) *….* R(xn)	R(x1) * R(x2)

IV. PROPOSED METHOD

The method presented in this paper is based on Particle Swarm Optimization (PSO) algorithm. The algorithm has been integrated with inertia coefficient adjustment and particle modification functions, leading to reduced run time, increased success rate and improved rate of the convergence of the algorithm.

4.1 Particle Swarm Optimization (PSO) algorithm

The operational procedure of PSO algorithm is similar to Genetic algorithm. In both these heuristic algorithms, the initial population is generated and initialized randomly at first. After generating the population, each of the particles are assessed and their evaluation function are calculated as long as a particle, by coincidence, to be able to meet the user's requests, leading algorithm to stops. Then, the initial population and constants are initialized and after that, algorithm begins. Actually, PSO algorithm maintains a set of particles in

which each particle represents a solution. Xi is a set of coordinates that displays the current position of the particle. At each step of the iterative algorithm, xi is computed as a solution to the problem. If this position is better than previous solutions, it would be saved in xi,best. Also, the best position found by all particles will be shown as xg,best. In each iteration of the algorithm, the velocity and position of the particles are updated according to equations (1) and (2). The algorithm aims to improve xi,best and to achieve a final solution.

$$vi[t+1]=wvi[t] + c1r1\ (xi,best[t]-xi[t]) + c2r2(xg,best[t]-xi[t]) \quad (1)$$

$$Xi[t+1]=xi[t] + vi[\ t+1] \quad (2)$$

w is inertial coefficient; r_1 and r_2 are the random numbers in the interval [0,1] with uniform distribution and c_1 and c_2 are the learning coefficients. The variables r_1 and r_2 cause some type of scattering in the solutions and therefore a more complete search is done on the space. The variable c_1 is the learning coefficient associated with personal experiences of each particle, while variable c_1 is the learning coefficient related to total experiences of all particles. From equation (1), it can be concluded that each particle during the movement takes into consideration its previous direction, the best position where it has been and the best position experienced by all particles[17].

This algorithm is superior to Genetic algorithm in terms of selecting few parameters, easy implementation and low cost. Other features of PSO algorithm include being indifferent towards comparing while designing the variables, doing parallel tasks for concurrent projects and its highly efficient global search [18].

4.2 Inertia coefficient adjustment function

This function, which is not a part of PSO algorithm but has been added to improve its efficiency, is called after a certain number of iterations of the algorithm. The function is shown in **Figure 2.**

```
Function Ajust _ w(){
if Max(particles) better than Max(particles_Prev){
        Decrease(w);
 }
 else
{           Increase(w);
              }
         }
```

Fig 2: Inertia coefficient adjustment function

In general, the high value of inertia coefficient results in a more global search and the low value of this coefficient lead to a more local search. Therefore, whenever the value of evaluation function in the PSO algorithm decreases, it means that the algorithm has become closer to the optimum solution. In this case, it is better that the value of inertia coefficient to be decreased such that search can be performed more local. On the contrary, whenever the value of evaluation function increases, it means that the algorithm is far from away optimal solution. Here, it is better that the value of inertia coefficient to be increased such that search can be performed more global. As a result after a certain number of iterations of the algorithm, inertia coefficient adjustment function is called and performed. Of course, it should be noted that if the value of inertia coefficient becomes constantly more or less, it will have a negative impact on searching. Thus, any increase or decrease in the value inertia coefficient is done with a certain probability. The probability is determined by considering the different values in the simulations.

4.3 Particle modification function

Particles modification function is regarded as a function that is executed after running a certain number of algorithms. The function is designed to help algorithm find a suitable structure if that algorithm was not able to find a suitable structure at any given time (i.e. algorithm is most likely trapped in local optimum). Function acts such that it selects some particles having higher evaluation function value than other particles and then modifies them. The reason for this is that algorithm is trapped in local optimum. If after a certain period of time the evaluation function value no longer improves, it means that algorithm is trapped in local optimum. Local optimum means a structure having good quality attributes but does not meet desirable attributes of the customer. On the other hand, neighborhood structure does not make much improvement in the value of particle's

evaluation function. Neighborhood structure is regarded as a structure that differs with original structure only in single or multiple web services. In such a case, the algorithm constantly moves around neighborhood structure.

To modify particles in order to get out of the local optimum, the following tasks can be performed:

- Random particles modification: This method is used in algorithms such as genetic algorithm (GA). The GA's mutation function is designed just for that purpose of getting algorithm out of the local optimum. In this function, which runs itself probabilistically, some chromosomes genes modify randomly to get algorithm out of the local optimum. Due to the randomness, this technique can help the algorithm to get out of the local optimum but the possibility of improving the genes is very low. In other words, it is possible for algorithm to be moved from local optimum to a point where is worse in quality and as a result, the algorithm have to again start a new search in the sample space, leading to increased running time.
- Replacement of the best locals: This method has been used in [9] too. In this study, which is based on genetic algorithm, some chromosomes are replaced with those chromosomes whose web services have been selected from among best locals. This approach has been effective in improving the running time, but the problem is that best locals would not necessarily lead to good structure.
- Dynamic method: In this method, which is used in the current paper and has been one of the innovations of our research, the best structures for composite web service should be selected at first. Then for each structure, the following sequence tasks are executed according to algorithm presented in Figure 3.
  - In the first step, the best particles are selected as the candidates for modification because they have the greatest chance to achieve a proper solution.
  - In the next step, for each of these particles, the requirements that are not met yet should be identified. For example, suppose that reliability is not met.
  - Then, we would identify those web services having this requirement and their attribute value also is less than that of other web services. As an example, suppose that the structure is in sequential form in which 70 services are done sequentially. Web services with the numbers 25, 38, 42, 63, 57, 51 and 69 have the lowest reliabilities among all 70 services. Therefore, by replacing some of these services with those services having the highest reliabilities, we can increase the value of particle's attribute.
  - During the replacement of web services, other attributes also will be modified absolutely. In every replacement of web services, if the value of other attributes did reach to the un-met threshold, then replacing that service should be avoided.

Mentioned operations are performed for 20% of top particles. Thus, the reliability of these particles increases, and at worst case, remains constant. Then, we'll re-run the PSO algorithm on all particles.

*Particle **B** = select_Best_Particles();*
*property **R**= select unsatisfied property for **B**;*
*List **S**=list of web services in **B** that have lowest **R** among other web services;*
*replace **S** with web services with high **R** in web service dataset;*

Fig 3: Particle modification function

V. EVALUATION AND CONCLUSION

This section evaluates the proposed method and indicates the effect of inertia coefficient adjustment and particles modification functions on improving the efficiency of PSO algorithm. This method is also compared with genetic algorithm.

5.1 The effect of inertia coefficient adjustment function

Firstly, the effect of inertia coefficient adjustment function on PSO algorithm has been studied. Figure 4 shows a compression between the running times of these two cases.

As is evident from the figure, inertia coefficient adjustment function has a positive impact on the running time of the algorithm. Of course, the running time of the algorithm is still high. The effect is due to the appropriate search type. This means that at the beginning of the algorithm, when the particles are completely dispersed in the sample space, the high value of inertia coefficient results in an overall search and when the algorithm approaches to the solution, search becomes more local by reducing the value of inertia coefficient. Naturally, local search can be appropriate on condition that the algorithm is not trapped in the local optimum. In

the case of the algorithm being trapped in the local optimum, the second function modifying the particles can help it to get out of the local optimum.

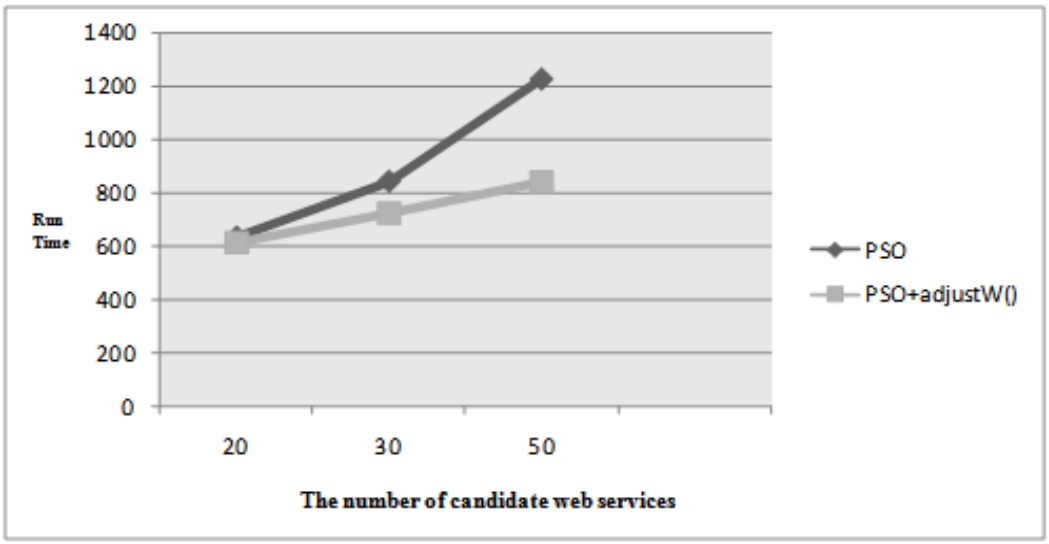

Fig 4: A comparison between the running times of normal PSO algorithm and PSO algorithm Integrated with inertia coefficient adjustment function

Then, the running success rates for these two cases were compared. As shown in Table 2, there is not much difference between these two cases in terms of running success rate. The reason for this is that inertia coefficient adjustment function cannot help algorithm to get out of the local optimum. In other words, when the algorithm is trapped in local optimum, it means that inertia coefficient also has been decreased and the search has become more local. Therefore, the algorithm is not able to get out of the local optimum.

Table 2: The effect of inertia coefficient adjustment function on the running Success rate of the PSO algorithm

Particle Swarm Optimization (PSO) algorithm	PSO algorithm integrated with inertia coefficient adjustment function
68.34%	**69.45%**

5.2 The effect of particles modification function

As is evident from the figure 5, this function has a great effect on improving the running time of the PSO algorithm. The reason could be the replacement of services forming the particles with those services having a better quality conditions than that of existing services.

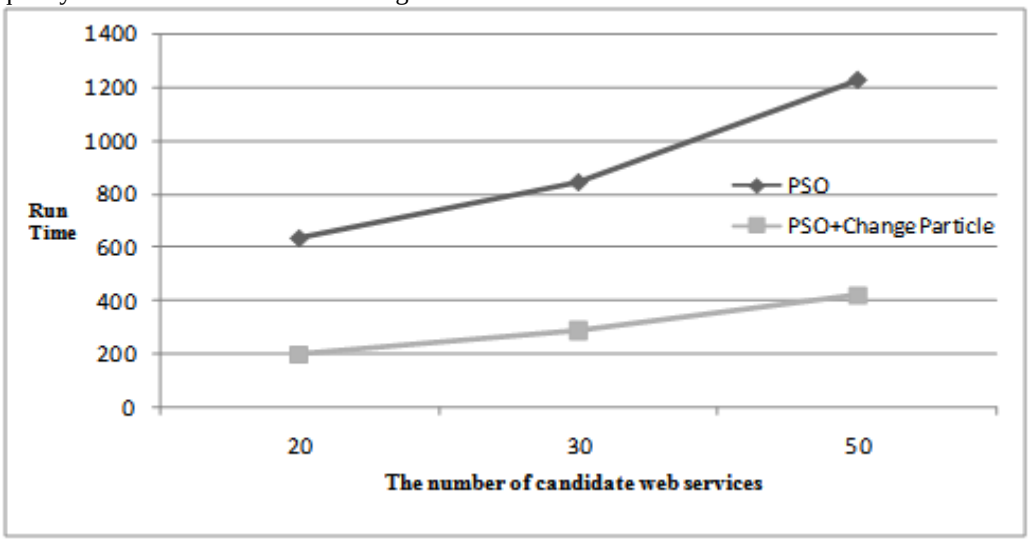

Fig 5: A comparison between the running times of normal PSO algorithm and PSO algorithm integrated with particles modification function

The number of candidate web services

As this function is integrated with the algorithm, the running success rate of the algorithm increases significantly (according to Table 3(. This could be due to the performance of particles modification function. Since this function modifies the particles, modified particles no longer exist in the previous location of the sample space and their location has been changed. It means that after executing this function, the modified particles are far away from the local optima and since the particles have been modified in such a way that their evaluation function would be improved, it could be concluded that the particles are out the local optimum.

Table 3: The effect of particles modification function on the running success rate of the PSO algorithm

Particle Swarm Optimization (PSO) algorithm	PSO algorithm integrated with particles modification function
68.34%	93%

5.3 The effect of inertia coefficient adjustment and particles modification functions

In the last step, both proposed functions are integrated with PSO algorithm and then its running time, success rate and rate of convergence are compared with those of the cases indicated in the previous sections. It is clear that the efficiency of the PSO algorithm increases relative to the previous cases when it is integrated with both functions. Figure 6 shows a comparison between the running times of the algorithms.

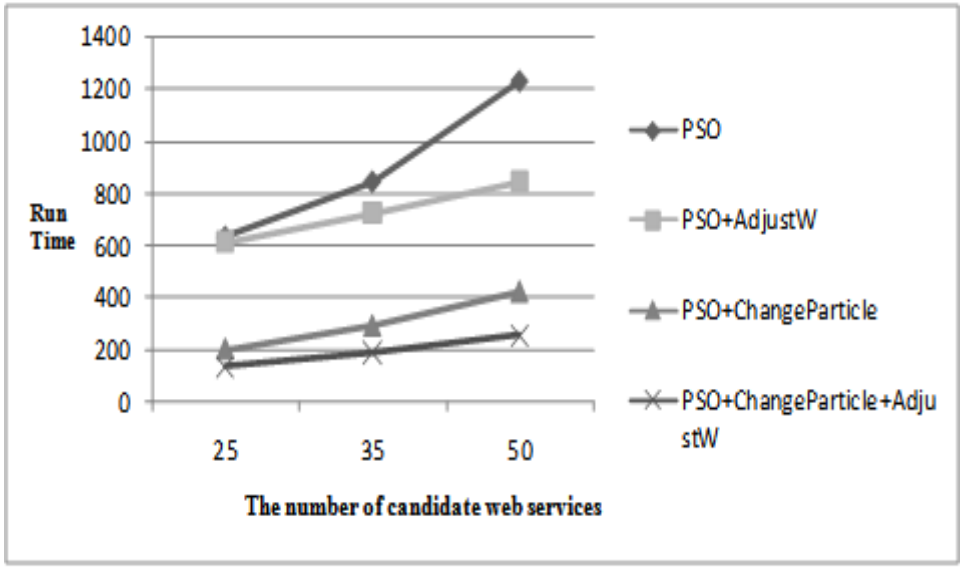

Fig 6: comparison between the running times of the algorithms.

Also, the running success rate of the algorithm in this case could be seen in Table 4.

Table 4: The effect of inertia coefficient adjustment and particles modification functions on the running success rate of the PSO algorithm

Particle Swarm Optimization (PSO) algorithm	PSO algorithm integrated with inertia coefficient adjustment and particles modification functions
68.34%	93.49%

In this paper, the efficiency of the proposed algorithm in terms of the rate of convergence also has been studied. Rate of convergence means the number of iterations from a moment when algorithm begins until the moment of finding a desired composed structure. Figure 7 presents the rate of convergence of proposed algorithm and its comparison with PSO algorithm.

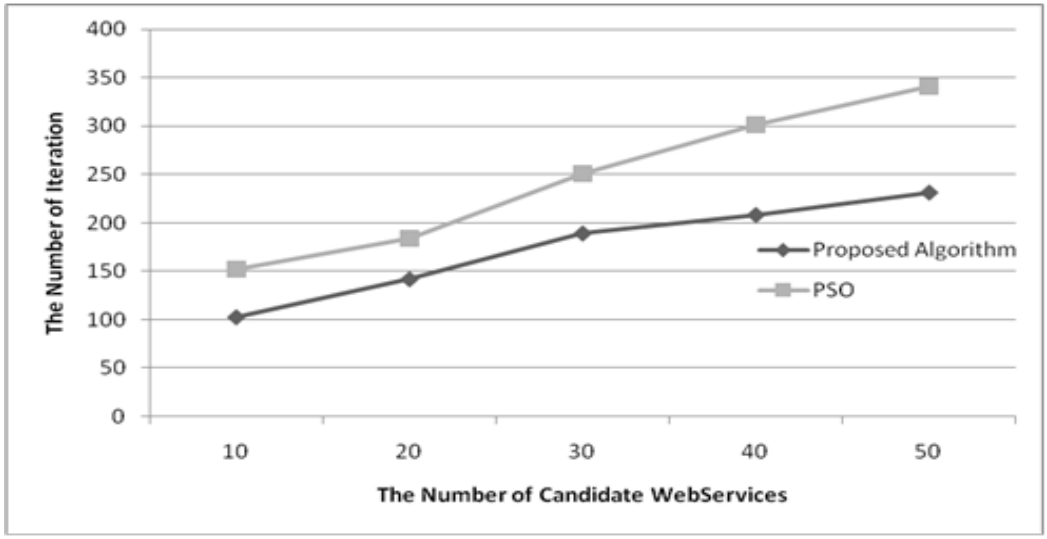

Fig 7: A comparison between the rate of convergence of proposed algorithm and PSO algorithm

5.4 A comparison between the proposed algorithm and Genetic algorithm

Genetic algorithm is one of the most popular and innovative algorithms for solving optimization problems. For this purpose, the efficiency of the proposed method is compared with the genetic algorithm. The results of a comparison between the proposed method and genetic algorithm are shown in Figure 8. As shown in the figure, the difference between the efficiencies of the Genetic method and proposed algorithm, when the number of web services and processes are small, is very low. However, as the number of web services and processes increases, the efficiency of the Genetic algorithm relative to the proposed algorithm decreases significantly. This is due to the cost-intensive operators of the Genetic algorithm. Genetic algorithm has the selection, crossover and mutation operators with crossover operator impose a high cost.

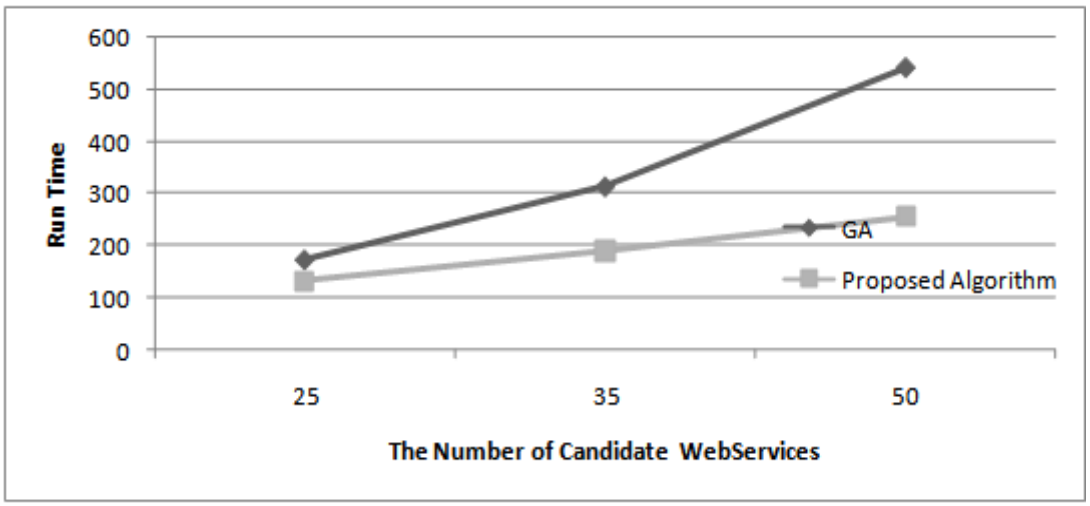

Fig 8: A comparison between the running time of PSO algorithm and Genetic algorithm

REFERENCES

[1] T.Erl, Service – Oriented Architecture : Concepts , Technology and Design , Prentice Hall , 2007.

[2] C.B.Pop, V.R.Chifu, I. Salomie, M. Dinsoreanu ,M. Fodor and I. Condor, A Bee-inspired Approach for Selecting the Optimal Service Composition Solution, 10th International Conference on Development and Applications Systems , Suceava, Romania, May 27-29, 2010.

[3] L. Wang , J. Shen and J. Yong , A Survey on Bio-inspired Algorithms for Web service composition , Proceedings of the IEEE 16th International Conference on Computer Supported Cooperative Work in Design, 2012, 569-574.

[4] T. Yu, K.J. Lin, Service Selection Algorithms for Web Services with End-to-end QoS Constraints , e-Commerce Technology CEC , Proceedings IEEE International, August 2004 ,129 – 136.

[5] T. Yu, K.J. Lin, Service Selection Algorithms for Web Services with End-to-end QoS Constraints, Journal of Information Systems and E-Business Management, 3(2) , Springer , 2005.

[6] T. Yu, Y. Zhang, and K. J. Lin, Efficient Algorithms for Web Services Selection with End-to-End QoS Constraints, ACM Transactions on the Web,1(1), May 2007.

[7] Y.Chao, Y. Meng-ting, Web Service Composition using Graph Model, IEEE,2010.
[8] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, Q. Z. Sheng, Quality Driven Web Services Composition , Budapest, Hungary, May 2003, 20-24.
[9] M.A. Amiri, H. Serajzadeh, QoS aware Web Service Composition based on Genetic Algorithm , 5th Internation Symposium on Telecommunications, 2010 , 502-507.
[10] S.Bahadori, S. Kafi, K.Zamanifar, M.R. Khayyambashi, Optical Web Service Composition Using Hybird GA-Tabu Search, Journal of Theoretical and Applied Information Technology, 2009.
[11] G.Zhipeng, CH. Jian, Q. Xuesong, M. Luoming, QoE/QoS driven simulated annealing-based genetic algorithm for web services selection , The Journal of China Universities of Posts and Telecommunications, 16*(* 08) , 2009,102-107 .
[12] L. Ai, M. Tang, QoS-based web service composition accommodating inter-service dependencies using minimal-conflict hill-climbing repair genetic algorithm, IEEE Fourth International Conference on e-Science, Dec 2008, 119.126 .
[13] Z. Yang, Ch. Shang, Q. Liu and Ch. Zhao , A Dynamic Web Services Composition Algorithm Based on the Combination of Ant Colony Algorithm and Genetic Algorithm, Journal of Computational Information Systems, 6(8) , 2010 , 2617-2622.
[14] C.B.Pop et al , Ant-inspired Technique for Auto - matic Web Service Composition and Selection, 12th International symposium on Symbolic and Numeric Algorithms for Scientic Computing, 2010 , 449-455.
[15] Sh.Lotfi and Kh. Mowlani, A QOS-Aware Imperialist Competitive Algorithm for Web Service Selection, Springer-Verlag Berlin Heidelberg, 2011, 135–148.
[16] A. F.M. Huang, C.W. Lan, S. J.H. Yang , An optimal QoS-based Web service selection scheme, journal elsevier, 2009, 3309–3322.
[17] H. Xia, Z. Li, Particle swarm algorithm for the quality of service-oriented web services selection, Second International Symposium on Knowledge Acquisition and Modeling , 2009, 303-306.
[18] M. Chen, Z. Wang, An Approach for Web Services Composition Based on QoS and Discrete Particle Swarm Optimization, ACIS Int. Conf. on Software Engineering Artificial Intelligence, Networking and Parallel/Distributed Computing ,2008,37-41 .