An Effective Fault-Tolerance Technique in Web Services: An Approach based on Hybrid Optimization Algorithm of PSO and Cuckoo Search

Fen He
School of Information Engineering, Guangzhou Nanyang Polytechnic College, China
hefen@gznylg.edu.cn

Kimia Rezaei Kalantari
Department of Computer Engineering, Sari Branch, Islamic Azad University, Iran
k.rkalantari@iausari.ac.ir

Ali Ebrahimnejad
Department of Mathematics, Qaemshahr Branch, Islamic Azad University, Iran
a.ebrahimnejad@qaemiau.ac.ir
*corresponding author

Homayun Motameni
Department of Computer Engineering, Sari Branch, Islamic Azad University, Iran
motameni@iausari.ac.ir

Abstract: *Software rejuvenation is an effective technique to counteract software aging in continuously-running application such as web service based systems. In client-server applications, where the server is intended to run perpetually, rejuvenation of the server process periodically during the server idle times increases the availability of that service. In these systems, web services are allocated based on the user's requirements and server's facilities. Since the selection of a service among candidates while maintaining the optimal quality of service is an Non-Deterministic Polynomial (NP)-hard problem, Meta-heuristics seems to be suitable. In this paper, we proposed dynamic software rejuvenation as a proactive fault-tolerance technique based on a combination of Cuckoo Search (CS) and Particle Swarm Optimization (PSO) algorithms called Computer Program Deviation Request (CPDR). Simulation results on Web Site Dream (WS-DREAM) dataset revealed that our strategy can decrease the failure rate of web services on average 38.6 percent in comparison with Genetic Algorithm (GA), Decision-Tree (DT) and Whale Optimization Algorithm (WOA) strategies.*

Keywords: *Software aging, software rejuvenation, web service, cuckoo search, particle swarm optimization.*

Received January 2, 2020; accepted March 23, 2021
https://doi.org/10.34028/iajit/19/2/10

1. Introduction

Web service refers to a client server application for the purpose of communication. Moreover, it provides a flexible solution for software integration under web environments [26]. The web services have been broadly adopted in e-business, e-government, finance, multimedia services and automotive systems. Multiple web services can provide similar functionalities with different non-functional properties, known as Quality Of Service (QOS) attributes. The most critical metrics to measure the QOS of long-running, real-time applications include reliability, availability and deadline guarantees. It is generally an essential but complex task to produce a service composition offering an optimal QOS value which satisfies customer requirements [14]. Selection of the proper web service providers among candidates while maintaining the optimal quality of service is an NP-hard problem, as it belongs to task scheduling problems [5]. Koutras and Platis [13] formulated multi-objective optimization problems for deriving the rejuvenation policies that optimize the system's overall performance. These studies were limited in order to model in a more realistic way software systems' behavior. Dynamic software rejuvenation was proposed in [7] to determine the optimal times. Their method was based on the ant colony optimization heuristic method with gravitational emulation local search. The low speed of the suggested algorithm makes it unfit in real time tasks. By adjusting on improvement of operation steps and parameter selection, a hybrid optimization algorithm based on genetic algorithm and ant colony optimization was presented in [25] to intensify the scheduling process of web services.

The next sections of this paper are organized as follows. Section 2 covers the previous relevant works. Section 3 deals with fault tolerance in web services. Section 4 provides and overview of the newly proposed dynamic software rejuvenation based on the hybrid of Cuckoo Search (CS) and Particle Swarm Optimization (PSO). The experimental results have been reported in section 5. Finally, the conclusions and future work have been discussed in section 6.

2. Literature Review

Considered one of the key requirements for web service based systems, reliability needs to provide

services with high availability, high fault tolerance, and dynamic deployment capabilities.

Kalantari *et al*. [8] presented a fuzzy system for selecting web services aimed at software rejuvenation. The method carried out the training phase based on the features of the service providers as well as the receivers' requirements while including a threshold for the rejuvenation of each web services. Kumaresan and Ganeshkumar [12] presented well established statistical time series (S) Auto Regressive Integrated Moving Average (ARIMA) approach for developing a forecasting model that can provide significantly improved reliability prediction by using real time publicly available software failure sets. The proposed method has been restricted itself in to linear event correlation function and similarity mapping techniques. However, their method has an effect on the achievement of requests. The other metaheuristic method based on whale optimization algorithm was proposed in [10]. The criterion of movement radius in their method was employed to flexible web service provider chooses and the threshold for the rejuvenation was examined.

Bai *et al*. [1] showed a semi-Markov process based approach and derived formulas for calculating AS availability and task execution pattern under the condition that all the aging, failure, Virtual machine Manager (VMM) fixing and live VM migration time follow general distributions. In [23], a dynamic method by using checkpoints in case of quality of service degradation was presented. The limitation of the proposed method was based on no assigning a weight to all inputs and no checking checkpoints until reduction of performance.

To achieving the optimum allocating in cloud services, Kalantari *et al*. [9] proposed a novel method employed a fuzzy neural network called Neural Fuzzy Dynamic Software Rejuvenation (NFDWSR). Their neuro fuzzy dynamic software rejuvenation could identify threshold of multiple fuzzy sets. In [22], an experimental study of software aging and rejuvenation on the docked daemon, following the Software Again and Rejuvenation Experiments (SWARE) approach was presented.

In [15], the system state was saved on a reliable storage through software rejuvenation coupled with occasional check pointing, in a way that the mission task could be resumed from the last saving checkpoint. This study was limited because the assumptions of the full state independent rejuvenation led to a degraded system.

Generally speaking, the majority of previous methods proposed to optimize software rejuvenation in different software systems come with certain shortcomings. The current study attempted to overcome most of these shortcomings, a few important of which are summarized below:

1. The methods proposed for optimization of software rejuvenation in software systems are mostly one-dimensional, i.e., only one single aspect of software rejuvenation is covered [16, 18].
2. The majority of newly proposed methods for software rejuvenation involve static management in systems [2, 3, 16, 18].
3. Many software rejuvenation methods stop the software system during rejuvenation; i.e., user requests are not responded throughout the process of rejuvenation. This implementation mechanism simplifies the procedure, but reduces the system's availability [20].
4. One of the problems in optimization of software rejuvenation is how to choose the most suitable server for web application allocation that can reduce the number of failures.

This study attempted to overcome the mentioned shortcomings, through utilizing different aspects of the software rejuvenation process. We develop a dynamic management model, capable of responding new web service requests even in the middle of the rejuvenation process. It not only improves the performance of the system, but also increase the availability of the system for new web service requests. In order to efficiently solve the allocation of web service providers to the web service requests, a hybrid metaheuristic algorithm based on cuckoo search and particle swarm optimization is presented.

3. Fault Tolerance in Web Services

As a criterion for web services, fault tolerance has grown into a significant research topic revolving around the selection procedure of a web service. The conventional fault tolerance web service models come with certain limitations. For instance, various essential sources of fault tolerance do not integrate, nor do they focus on different attributes of service quality namely performance and accessibility to fully meet user requirements. A static fault-tolerant scheduling algorithm has been presented in [6]. It was based on a list of scheduling heuristics which satisfy the application time constraints even in the presence of faults by exploring the spare capacity of available processors in the architecture. Mogk *et al*. [19] proposed a calculus that enables formal reasoning about applications' dataflow within and across individual devices. As a result of their study, programmers were relieved of handling intricate details of distributing change propagation and coping with distribution failures in the presence of interactivity. In [21], a fault tolerance-based load balancing approach by considering the dynamic nature of resources was proposed. A fault tolerance dynamic load balancing model was applied for task execution based on resource load and fault index value. In continue, for fault tolerance, checkpoints were set at various

determined intervals to resume tasks at the next possible instance that avoids unnecessary placement of checkpoints. The overhead due to checkpoint interval based of fault index restricted their approach.

4. Proposed Method

Meta-heuristic algorithms such as PSO [4], Whale Optimization Algorithm (WOA), Ant Colony Optimization (ACO) and CS [24] are important tools for solving complex real world optimization problems.

4.1. Brief Review of CS

CS is used for various optimization problems as it idealized breeding behavior. In CS, each egg in a nest represents a solution, and a cuckoo egg represents a new solution. The aim is to replace the new solutions (cuckoos) with not-so-good solutions in the nests. In the simplest form, each nest has one egg and can be extended to multiple eggs representing a set of solutions. Initially, each cuckoo lays an egg and is placed inside a randomly selected nest. In the second step, the best nests having high quality eggs moves to the next generation. In the last step, the number of available host nests is fixed, and the laid egg by a cuckoo is discovered by the host bird with the probability $r \in [0,1]$ $r \in [0,1]$ [11]. This algorithm is very simple and easy to implement as it involves only a single parameter r a in CS.

4.2. Brief Review of PSO

Some of the applications have used PSO to solve NP-Hard problems like Scheduling and the task allocation problems [17]. The PSO algorithm is a meta-heuristic based on the movement of a population (swarm) of individuals (particles) randomly distributed in the search space, each one with its own position and velocity. It is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. In PSO the swarm of particle is randomly generated initially and each particle position represents a possible solution. Each particle is positioned in the search space and has a fitness value and velocity to determine the speed and direction of its moves. Particle move around in the search space based on the particle's update position and velocity to get an optimized solution.

4.3. Combined CS and PSO

Some problems like dynamic rejuvenation in web-based systems have a large-scale space. So, finding the optimal solution with high efficiency seems to be impossible by previous works. In this paper, Dynamic rejuvenation has been proposed based on PSO and Cuckoo Search in web services that called Computer Program Deviation Request (CPDR). The PSO algorithm becomes easily trapped in local optimum in solving a large complex problem like software rejuvenation. The main reasons to choose this algorithm are its performance-complexity tradeoff and fairness features regarding other heuristic optimization methods. In addition, the PSO approach is much easier to implement data, requires less computational bookkeeping for resource allocation problems and it was previously applied in resource and power allocation, preventive maintenance effectively. Using CS algorithm could be efficient in finding an optimal combination of web service allocation in cloud computing system which is dynamically changing. But the CS has some shortcoming too. It has slow convergence speed. In CS algorithm, high randomness of the Levy flight makes the search process quickly jump from one region to another one. By this property, the algorithm initiates a blind search process, convergence speed become slowly and the efficiency reduced close to the optimal solution. To overcome these shortcomings, the proposed method used a hybrid of PSO and CS. Pseudo code of the proposed CPDR algorithm can be seen in Algorithm (1).

Algorithm 1. Pseudo code of service allocation for users, using CPDR algorithm

1: Load requested web services dataset
2: Initialize each search solution to contain m randomly web service providers
3: while t < maximum number of Iteration do {
4: for each search solution i do {
5: for each data vector X_P do {
6: Calculate the Euclidean distance of X_P to all web service types.
7: Assign X_P to the web service type l such that $|X_{Pij}-Z_l| = \min|X_{Pij}-Z_{ws}|$; ws = 1.2. …. k
8: Calculate the fitness using

$$fitness = \sum_{j=1}^{k}\sum_{i=1}^{n} W_{ij}\left| X_{Pij} - Z_l\right|$$

9: end for
10: end for
11: Update population using hybrid CS and PSO
12: t = t + 1
13: if (CPDR algorithm convergence)
14: break;
15: end if
16: end while
17: return X^*
18: Assigning web services to the user requests according to X^*
19: Rejuvenation process according to Eqs. (2) and (3)

The framework of the proposed procedure has been illustrated in Figure 1. At the beginning, first organization of the system is showed. Then, all the web service requests according to their priority. Now, in order to solve the mentioned lack, the presented method acts by measuring web services' reliability and time threshold metrics along software executing by using the combination of CS and PSO in Figure 2. In continue the parameters of web services updated and the web services which has less QoS parameters go to rejuvenation state.

The major difficulty in software rejuvenating is how to select suitable service provider by considering reduction failures. The main reason for using Meta-heuristics algorithms is that the selection of a service among candidates while maintaining the optimal quality of service is an NP-hard problem. For this purpose, an efficient algorithm based on hybrid of CS and PSO were used to find the best allocation of the web services for users in order to reduce failed web services as documented in Figure 2.

There are several web services which can allocated to user based on their quality of service parameters. The proposed method in Figure 2 finds the best allocation with minimum failure rate than random allocation.

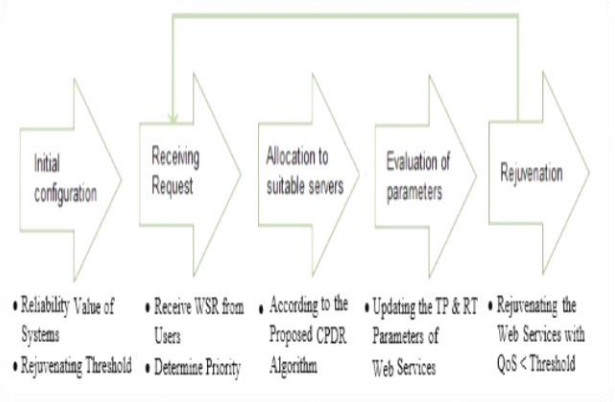

Figure 1. The basic block diagram in our work.

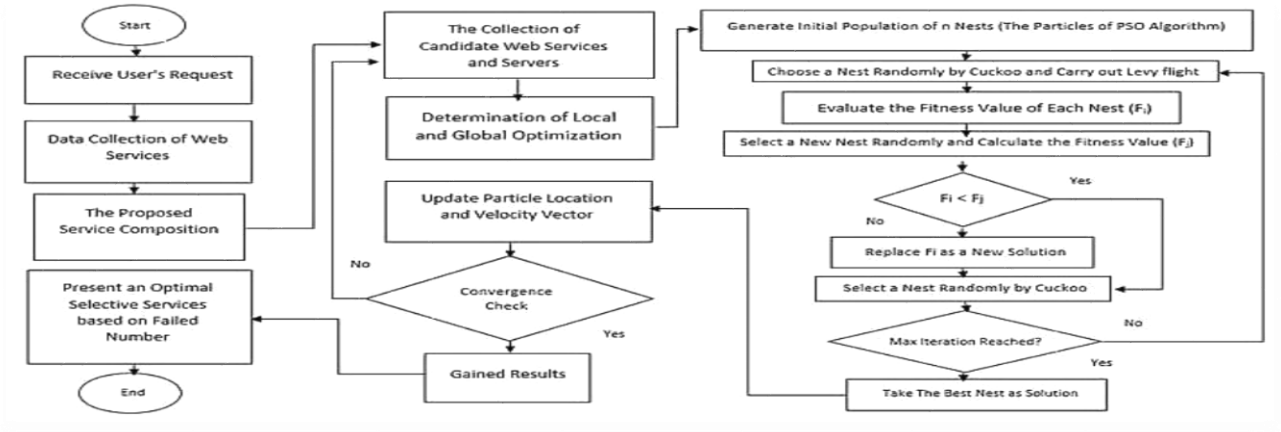

Figure 2. Overall steps of the proposed method.

After create matrix of initial cuckoo's habitat some randomly produced number of eggs for each cuckoo will be considered. The number of assigned eggs to cuckoo showed the solution for service allocation in the paper. In general, each egg represented a service and the sum of eggs in each habitat showed a solution in system. The rapid convergence of CS algorithm can be substantially improved by replacing abandoned nests instead of using the random replacements from original method. Reducing the failed service number based on their attribute was considered as a competency factor in our method. Over time, the particle accelerates to the particle with high competency value. The failed web services number, less than five percent of the whole web services, was considered as the convergence in the proposed method.

In our proposed method, the objective is to minimize the number of failure web services assigned to all users. Since the less respond time and the more throughput results less probability of a web service to be failed, the fitness function of the proposed method and all method that was used for comparison can be represented as Equation (1). The objective is to minimize the average respond time and simultaneously maximize the average throughput of the assigned web services to all users.

$$fitnessfunction = \frac{\sum_{i=1}^{n}(\frac{1}{m}\sum_{j=1}^{m} rt_{ij})}{\sum_{i=1}^{n}(\frac{1}{m}\sum_{j=1}^{m} tp_{ij})} \quad (1)$$

Where n indicates the number of users and m demonstrates the maximum service request number. Also, rt_{ij} represents the respond time of service j by user i and rp_{ij} represents the throughput of service j by user i. It is worth mentioning that with the increasing of the availability, the throughput will strengthen in system. Owing to the assumption that suitable result has minimum response time and maximum throughput; the fitness function must be minimized in the suggested method. To continue, all solutions were sorted in ascending order based on their fitness function value. It means after sorting, the first one has the most appropriate performance at the current stage.

After this suitable process for allocation, in the next steps all services without appropriate condition go to rejuvenation process. The value of time out threshold is application-dependent and could be set by user in the dataset based on the need of their applications. This threshold determines the condition of rejuvenation for each web service, considering its respond time and throughput. More specifically, each web service i should be rejuvenated if $\frac{tp_i}{rt_i} < threshold$. The respond time and throughput of aged services updated according to the following rate:

$$\mathrm{rt_i^{new}} = \mathrm{rt_i^{old}} * \left(1 - \mathrm{e}^{-\left(\frac{\mathrm{tp_i^{old}}}{\mathrm{rt_i^{old}}}\mathrm{t}\right)}\right) \quad (2)$$

$$tp_i^{new} = tp_i^{old} * (1 + e^{-\left(\frac{tp_i^{old}}{rt_i^{old}} t\right)}) \quad (3)$$

Where rt_i^{old} and tp_i^{old} are the current respond time and throughput of the aged service i, respectively. Moreover, rt_i^{new} and tp_i^{new} are the new respond time and throughput of the aged service i (after rejuvenation process), respectively. Then, the process and allocations will perform again from the beginning.

5. Simulations

In this paper, we conducted simulations on Windows 7 and a Core i5 system. It is worth mentioning that increasing reliability in web service providers is so important in software rejuvenation. This can be done with reduction of failed web service request numbers. Hence, several methods based on WS-DREAM dataset are considered in this study. Figure 3 shows the comparison of the convergence speed of the proposed CPDR method with two existing metaheuristics: GA and WOA. In this approach, each implementation of the algorithm was iterated 500 times. Moreover, for each one of these iterations 100 search agents (population size) were considered. The NFE (number of function evaluation) was 50,000 for all methods. As seen in Figure 3, both WOA and CPDR have better convergence than GA at the early iterations. At first, the convergence speed of the WOA is better than the CPDR. By increasing the number of iterations in WOA, the fitness function value is first decreased and then remains fixed after iteration 150 because trapping in local optima. However, the proposed CPDR can overcome local optima points and continue reducing the fitness function value until iteration 500. After approximately 150 iterations, the proposed method has the less fitness function value and the best convergence speed among other methods.

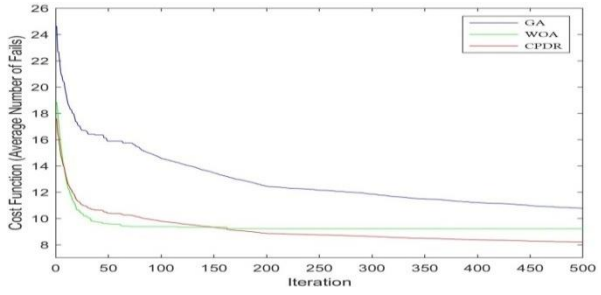

Figure 3. Comparison of the convergence speed graph of the proposed method.

As Figure 3 shows, at first, the convergence speed graph of the WOA is better than proposed method. After approximately 150 iterations the proposed method has the less cost function and the best convergence speed among other methods. The proposed method is compared with the existing algorithms in term of run-time. Run-time is the amount of time taken for a computer program to perform a task. In Table 1, the optimization time and overall run-time (on average for all dataset) of the proposed method is compared with WOA and GA.

Table 1. Comparison of average run-time (in sec) of the proposed method with GA.

	GA	WOA	CPDR
Optimization Time (Assignment)	9.5	8.3	9.2
Overall Time (Assignment + Rejuvenation)	12.9	11.7	12.6

Comparison of the average number of failed services for different techniques (DT, GA, WOA, and CPDR) for 10 data samples can be summarized in Table 2. According to the average results, the reduction rate of the proposed CPDR algorithm in the number of failed services is 49.8%, 33.7%, and 13.9%, as compared with the DT, GA, and WOA, respectively.

The experimental results in Table 2 are proved to be statistically significant, using ANOVA test, which is used to measure the significance of differences between different techniques over multiple datasets.

The experimental results in Table 2 are proved to be statistically significant, using ANOVA test, which is used to measure the significance of differences between different techniques over multiple datasets.

Table 2. Comparison of average failed web services for different techniques on different datasets.

Data	DT	GA	WOA	CPDR
1	12.2	9.6	7.2	6.5
2	11.8	9.3	7.4	6.1
3	12.6	9.4	6.9	6.2
4	11.5	8.9	6.5	5.5
5	12.3	9.2	7	6
6	11.1	8.5	6.6	5.7
7	11.6	9.2	7.3	6.5
8	12.8	9.3	7.2	5.7
9	12.2	8.8	6.8	6
10	12.6	9.2	7.5	6.4
Average	12.07	9.14	7.04	6.06

To apply the Analysis of Variance (ANOVA) test, 10 samples of the average failed web services achieved by each algorithm (DT, GA, WOA, and CPDR) are taken with $\alpha = 0.05$. The null hypothesis is considered as

$$H_{null}: \mu_{CPDR} = \mu_{DT} = \mu_{GA} = \mu_{WOA} \quad (4)$$

While, the alternative hypothesis is defined as

$$H_{alternative}: \mu_{CPDR} \neq \mu_{DT} \neq \mu_{GA} \neq \mu_{WOA} \quad (5)$$

According to the obtained results of the ANOVA test in Table 3, p-value much less than 0.05 have been obtained. As a result, the null hypothesis is rejected, which means that the results are significantly different for the compared algorithms.

Table 3. Results of ANOVA test for average failed web services.

Source of Variation	Sum of Square	Df	Mean Square	F-statistic	p-value
Between Groups	212.157	3	70.7189	439.48	2.115e-28
Within Groups	5.793	36	0.16.9		
Total	217.95	39			

6. Conclusions

The rejuvenation policies are the powerful tool for the improvement of the performance of web service based systems. In this paper, the effective use of hybrid optimization algorithm of CS and PSO for dynamic software rejuvenation based on characteristics of service providers and needs of the users has been investigated. In addition to the allocation of Web services, the threshold for rejuvenation was checked dynamically and the systems which required to rejuvenation were detected.

As a result of implementation of the proposed technique, there was a considerable improvement in the performance in compared to well-known algorithms such as WOA, GA and Decision Tree. The results obtained from the proposed method has on the average 14 percent, 38 percent and 64 Percent less failed web services in comparison with WOA, GA and Decision Tree methods respectively. This indicates that the proposed method based on hybrid optimization algorithm of PSO and Cuckoo Search can be used for environments that require high fault-tolerance. For future research, to improve the accuracy and runtime of the presented method, parallel algorithms with fuzzy meta-heuristics can be used.

Acknowledgment

This research is funded by the IT Innovation and Application Project (No.: NYCQPT-2018-01) and the New Generation IT Industry Education and Collaborative Innovation Project（NY-2019CQZX-01) of Guangzhou Nanyang Polytechnic College of China.

References

[1] Bai J., Chang X., Machilda F., Trivedi K., and Han Z., "Analyzing Software Rejuvenation Techniques in Virtualized System: Service Provider and User Views," *IEEE Access*, vol. 8, pp. 6448-6459, 2020.

[2] Cotroneo D., Iannillo A., Natella R., Pietrantuono R., and Russo S., "The Software Aging and Rejuvenation Repository", *in Proceedings of International Symposium on Software Reliability Engineering Workshops*, Gaithersburg, pp. 108-113, 2015.

[3] Cui H., Li Y., Liu X., Ansari N., and Liu Y., "Cloud Service Reliability Modeling and Optimal Task Scheduling," *IET Comunication*, vol. 11, no. 2, pp. 161-167, 2017.

[4] Eberhart R. and Kennedy J., "Particle Swarm Optimization," *in Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942-1948, 1995.

[5] Fanian F., Bardsiri V., and Shokouhifar M., "A New Task Scheduling Algorithm Using Firefly and Simulated Annealing Algorithms in Cloud Computing," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 2. pp. 195-202, 2018.

[6] Kada B. and Kalla H., *Research Anthology on Architectures, Frameworks, and Integration Strategies for Distributed and Cloud Computing*, IGI Global, 2019.

[7] Kalantari K., Ebrahimnejad A., and Motameni H., "Efficient Improved Ant Colony Optimization Algorithm for Dynamic Software Rejuvenation in Web Services," *IET Software*, vol. 14, no. 4, pp. 369-376, 2020.

[8] Kalantari K., Ebrahimnejad A., and Motameni H., "Presenting A New Fuzzy System for Web Service Selection Aimed at Dynamic Software Rejuvenation," *Complex and Intelligent System*, vol. 6, no. 11, 2020.

[9] Kalantari K., Ebrahimnejad A., and Motameni H., "A Fuzzy Neural Network for Web Service Selection Aimed at Dynamic Software Rejuvenation," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 28, no. 5, pp. 2718-2734, 2020.

[10] Kalantari K., Ebrahimnejad A., and Motameni H., "Dynamic Software Rejuvenation in Web Services: A Whale Optimization Algorithm-Based Approach," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 28, no. 2, pp. 890-903, 2020.

[11] Komaki M., Teymourian E., Kayvanfar V., and Booyavi Z., "Improved Discrete Cuckoo Optimization Algorithm for The Three-Stage Assembly Flowshop Scheduling Problem," *Computer and Industrial Engineering*, vol. 105, pp. 158-173, 2017.

[12] Kumaresan K. and Ganeshkumar P., "Software Reliability Prediction Model with Realistic Assumption Using Time Series(S) ARIMA Model," *Ambient Intelligence and Humanized Computing*, vol. 11, vo. 3, pp. 1-8, 2020.

[13] Koutras V. and Platis A., "On the Performance Of Software Rejuvenation Models with Multiple Degradation Levels," *Software Quality Journal*, vol. 28, no. 1, pp. 1-37, 2019.

[14] Levitin G., Xing L., and Huang H., "Optimization of Partial Software Rejuvenation Policy," *Reliability Engineering and System Safety*, vol. 188, pp. 289-296, 2019.

[15] Levitin G., Xing L., and Luo L., "Joint Optimal Checkpointing and Rejuvenation Policy for Real-Time Computing Tasks," *Reliability Engineering and System Safety*, vol. 182, pp. 63-72, 2019.

[16] Machida F. and Miyoshi N., "An Optimal Stopping Problem for Software Rejuvenation in a Job Processing System," *in Proceedings of Software Reliability Engineering Workshops*, Gaithersburg, pp. 139-143, 2015.

[17] Marimuthu P., Arumugam R., and Ali J., "Hybrid Metaheuristic Algorithm for Real Time Task Assignment Problem in Heterogeneous Multiprocessors," *The International Arab Journal of Information Technology*, vol. 15, no. 3, pp. 445-453, 2018.

[18] Meng H., Liu J., and Hei X., "Modeling and Optimizing Periodically Inspected Software Rejuvenation Policy Based on Geometric Sequences," *Reliability Engineering and System Safety*, vol. 133, pp. 184-191, 2015.

[19] Morgk R., Drechsler J., and Salvaneschi G., "A Fulat-Tolerant Programming Model for Distributed Interactive Applications," *in Proceedings of the ACM on Programming Languages*, pp. 1-29, 2019.

[20] Okamura H. and Dohi T., "Optimization of Opportunity-Based Software Rejuvenation Policy," *in Proceedings of 23rd International Symposium on Software Reliability Engineering Workshops*, Dallas, pp. 283-286, 2012.

[21] Shukla A., Kumar S., and Singh H., "Fault Tolerance Based Load Balancing Approach for Web Resources," *Journal of the Chinese Institute of Engineers*, vol. 42, no. 7, pp. 583-592, 2019.

[22] Torquato M. and Viera M., "An Experimental Study of Software Aging and Rejuvenation in Dockerd," *in Proceedings of 15th European Dependable Computing Conference*, Naples, pp. 1-6, 2019.

[23] Vargas-Santiago M., Morales-Rosales L., Monroy R., Pomares-Hernandez S., and Drira K., "Autonomic Web Services Based on Different Adaptive Quasi-Asynchronous Checkpointing Techniques," *Computing and Artificial Intelligence*, vol. 10, no.7, pp. 2495, 2020.

[24] Yang X. and Deb S., "Cuckoo Search Via Levy Flights," *in Proceedings of World Congress on Nature and Biologically Inspired Computing*, Coimbatore, pp. 210-214, 2009.

[25] Yang Y., Yang B., Wang S., Liu F., Wang Y., and Shu X., "A Dynamic Ant-Colony Genetic Algorithm for Cloud Service Composition Optimization," *International Journal of Advanced Manufacturing Technology*, vol. 102, no. 1, pp. 355-368, 2019.

[26] Zhang Y., Wang K., He Q., Chen F., Deng Sh., Zheng Z., and Yang Y., "Covering-based Web Service Quality Prediction Via Neighborhood-Aware Matrix Factorization," *IEEE, Transaction on Services Computing*, vol. 14, no. 5, pp. 1333-1344, 2019.

Fen He received the B.E. degree in Computer Science from Hunan Normal University, Hunan, China, in 2003, and the M.S. degree in Software Engineering from Huazhong University of Science and Technology, Wuhan, China in 2012. He is currently an Assistant Professor with Guangzhou Nanyang Institute of Technology of China. She is currently Lecturer with Guangzhou Nanyang Polytechnic College of China. Her research interests include Artificial Intelligence and Big Data.

Kimia Rezaei Kalantari serves as an Assistant Professor at the Computer Department, Sari Branch, Islamic Azad University. She received her B.S. degree in Software Computer Engineering in Iran University of Science and Technology and M.S. degree in Computer Engineering-Software from Qazvin Islamic Azad University 2005 and 2010 Respectively. She Received Ph.D degree in Software Computer Engineering from Babol Islamic Azad University in 2019. Her current research interests include Metaheuristic Algorithms, Data mining and software quality assurance.

Ali Ebrahimnejad as the corresponding of this paper serves as a Full Professor at the Mathematics Department, Qaemshahr Branch, Islamic Azad University, Iran. He is editor in chief of Fuzzy Optimization and Modeling Journal. He is on the editorial board of the International Journal of Intelligent Computing and Cybernetics, Annals of Fuzzy Mathematics and Informatics, International Journal of Information and Decision Sciences, Iranian Journal of Optimization, and International Journal of Enterprise information Systems. His research interests include operations research, network flow, data envelopment analysis and fuzzy optimization.

Homayun Motameni received B.S. degree in Computer Engineering-Software Engineering in Shahid Beheshti of Tehran University and M.S. degree in Computer Engineering-Machine Intelligence from Islamic Azad University-Science and Reasearch Branch in 1995 and 1998. Respectively. He Received Ph.D degree in Computer Engineering-Software Engineering from Islamic Azad University-Science and Research Branch in 2007. His current research interests include Evolution Algorithms, Petri Net, and machine learning.