# Multi-Objective QoS-based Reusable Service Selection enhanced with Artificial Bee Colony (MQRSABC)

A. Florence Deepa[#1], J.G.R. Sathiaseelan[*2]

*[#1]Ph.D. Research Scholar, [*2]Associate Professor and Head,*
*Bishop Heber College (Autonomous), Tiruchirappalli, Tamil Nadu, India.*

***Abstract* — *Multi-objective optimization has been a difficult problem and a research focus in the field of science and engineering. An enhanced Artificial Bee Colony (ABC) optimization algorithm, called the QoS-Based Reusable Service Selection with Multi-Objective Constraints using Artificial Bee Colony (MQRSABC) Algorithm, is proposed to measure the quality of the search time and execution time better. In this proposal, the fast non-dictated population selection strategy are applied to measure the quality of the solution and select the better ones. An innovative solution generation strategy is designed to exploit the neighbourhood of the existing solutions. Likewise, a novel fitness calculation method is presented to calculate the selecting probability for onlookers. The proposed algorithm is validated on benchmark functions in terms of search time, execution time and reusability constraint with the existing approaches PSO and ACO. The experimental results show that the proposed approach can find solutions with competitive convergence and diversity within a shorter period of time, when compared with the traditional multi-objective algorithms. Subsequently, it can be considered as a feasible alternative to solve the multi-objective optimization problems.*

Keywords — *Artificial Bee Colony, Particle Swarm Optimization, Ant Colony Optimization, Multi-Objective Constraints, Quality of Service, Reusable Service Selection.*

## I. Introduction

In recent years, the Web services is an enormous research area which has been devoted to study the related aspects of quality of services (QoS) based selection and composition. One of the vital researched topics in the Web services is the reusability of the Web service composition which is motivated for two main reasons. First, the web clients can requires multiple services to be utilized and may be complex. Second, dynamically changing set of available services that are used to add/remove. Here, the manual process is not effective. To handle the concept, the reusability of services is essential.

When web engineers compose existing services into renewed one, they have to make decisions about service selection from service repositories. These decisions are often based on QoS parameters, where engineers aim for an optimal choice in service selection. The existing web services are composed to get a new value-added function that has become one of the most features of web services that are used by web services composition [1].

## II. Related Work

The Composite Web Services technology plays a vital role in services computing that allows business processes by composing elementary services under a shared workflow thereby providing the business process as a single-service. The technical challenges for integrating the services in terms of compatibility and adaption are carried out in the earlier research papers. Service composition is the process to find the existing and appropriate service components to compose a new service which has an aggregation function. The difference is that the problem is nonlinear and can be solved with intelligent optimization algorithms, such as Genetic Algorithms (GA) [2, 3], Particle Swarm Optimization (PSO) algorithm [4, 5], Ant Colony Optimization (ACO) algorithm [6], and so on. Intelligent optimization methods can solve the complex service composition problem within a short period of time with high quality, thus it has got the widespread concern in recent years.

## III. Standard Artificial Bee Colony (ABC) Algorithm Model

The standard Artificial Bee Colony (ABC) algorithm is a population-based optimization algorithm. According to the working principal of standard ABC, has classified into five main phases namely, initialization, employed-bees, onlooker-bees, scout-bee, and termination phases which consist of a total of twelve stages or processes.

In this standard ABC, three phases namely, employee bees, onlooker bees and scout bees phases are the performance deciding phases, whereas others are supporting phases. The searching process of the algorithm starts with the employee bees' phase, then the onlooker bees discover the neighbourhood of the food sources that are allocated to them by the employee bees and apply the fitness proportion selection pattern so that to select the food sources that are satisfied.

In the initialization phases, the food sources represent the possible solution among the population of a problem that are randomly initialized. Based on the user predetermined values of the population size, these food sources are then assigned to the employed-bees. Then, the nectar amounts that represent the fitness value of each food source are calculated using equation (1) [18, 29, 30]:

$$fit_i = \begin{cases} \frac{1}{1+f_i} & f_i \geq 0 \\ 1 + abs(f_i) & f_i < 0 \end{cases} \quad (1)$$

where $f_i$ is objective function value of $i^{th}$ food source.

In the employed bees phase, the bees explore the neighbourhood of the food sources assigned to them and update the food sources using the mutation equation (2) given by

$$z_{ij} = y_{ij} + \phi_{ij}(y_{ij} - y_{kj}) \quad (2)$$

where $z_{ij}$ is the candidate solution of food sources, $y_{ij}$ is the $j^{th}$ dimension of the $i^{th}$ food sources, and $y_{kj}$ is the $k^{th}$ food sources that are randomly chosen from a neighbourhood of $i^{th}$ food sources for $k \in [1,2,\ldots, S_N]$ and $S_N$ is the number of food sources. Subscripts $k$ and $i$ are mutually exclusive food sources. For the equation, $k$ and $j$ are chosen randomly and $j \in [1,2,\ldots, D]$ where $D$ represents the dimension of the search space and $\phi_{ij}$ is the control parameter that represents random number from $[-1,1]$, inclusively. The explorations by employed-bees generate new food sources (i.e., candidate solutions of food sources). A selection between the candidate solution and the old food sources is based on which of them exhibits the best fitness value by using greedy-selection scheme. The chosen food sources are potentially fitter food sources and are shared with onlooker-bees in onlooker-bees phase.

During the onlooker bee's phase, the bees apply the fitness proportion selection scheme to choose the selected-fitter food sources from all the food sources that are shared by employed bees that are not updated actually. The exploitation of the food sources by onlooker-bees has actually made the algorithm converge fast. The fitness-proportion selection scheme is dependent on the probability value, $P_i$ given by

$$P_i = \frac{fit_i}{\sum_{j=1}^{S_N} fit_j} \quad (3)$$

where $P_i$ is the probability of $i^{th}$ food source, $fit_i$ is the fitness value of $i^{th}$ food source, and $S_N$ represents the number of available food sources. Onlooker-bees then explored the neighbourhood of the selected-fitter food sources and update the food sources using the equation (2). The new candidate solution is then compared with the old food source using the greedy-selection scheme. Next, the best food source so far for that generation is recalled before entering the scout-bee phase.

In scout-bee phase, a food source which has become exhausted and does not show improvement over a *limit* is abandoned [23]. *Limit* is a control parameter used to signify exhausted food source [19]. Employed-bee whose food source has reached *limit* will become scout-bee. The scout-bee will take consequent flights and search the search space randomly to find new food source using

$$Y_i^j = Y_{min}^j + rand(0,1)(Y_{max}^j - Y_{min}^j) \quad (4)$$

where $Y_{min}^j$ and $Y_{max}^j$ are the lower and upper limit of the search space, respectively. Rand (0, 1) is a function which randomly generates numbers within [0, 1]. It is necessary action for the scout-bee to replace the abandoned food source with new food source so that the balance the number of populations can be identified again.

The termination criterion of the algorithm is based on the maximum number of generations or maximum cycle number (MCN) [18]. This number is stipulated by user prior to the simulation of ABC algorithm.

## IV. Multi-objective QoS-based Reusable Service Selection (MQRSABC) Model

The Multi-objective QoS-based Reusable Service Selection with Artificial Bee Colony (MQRSABC) algorithm is enhanced version of QoS-based Reusable Service Selection (RUQSS) Algorithm to encounter the neighbourhood positions gently through the association of three kinds of bees, Employed bees, Onlookers bees and Scout Bees. It gains the optimal or near-optimal solution in the feasible space by correlating with RUQSS Algorithm. The ReUsable QoS-based Service Selection (RUQSS) Algorithm is anticipated to predict the reusability of a web service by quantifying *availability* and *reliability* of atomic services using the Weibull analysis and Linear Regression and service reusability is estimated to decrease the complexity during the new development.

The original Artificial Bee Colony (ABC) algorithm catches the optimal solution by steering development of the candidate solution group which calculates the intelligent seeking activities of honey bees denoted by m-dimensional real vector. It represents a feasible solution for each food source that to be solved, and the nectar quality of the food source that denotes the fitness of that feasible solution which indicates the quality of the solution.

In order to improve quality of the Service Composition Optimal Selection (SCOS), a new MQRSABC algorithm which is enhanced version of the RUQSS algorithm with standard ABC is proposed. The figure 1 anticipated the working principle of MQRSABC Algorithm. According to the SCOS problem, each candidate solution represents a feasible service composition solution, and each dimension of the candidate solution must be an integer satisfying the boundary conditions. Consequently, the food source navigating and the approach of identifying candidate solutions need to be enhanced.

The objective of SCOS is to select and integrate the appropriate candidate services from each concrete services group, containing *n* abstract services $AS_{AP}$, where $AS_{AP} = \{as_1, \cdots, as_n\}$for an abstract process AP, and a set of composite patterns $CP = \{cp_1, \cdots, cp_m\}$ associated with each $as_i$ that satisfies the constraints on the basis of the previous work. Subsequently, each feasible solution is a service composition optimal selection solution

In order to improve quality of the Service Composition Optimal Selection (SCOS), a new MQRSABC algorithm which is enhanced version of the RUQSS algorithm with standard ABC is proposed. The figure 2 anticipated the working principle of MQRSABC Algorithm. According to the SCOS problem, each candidate solution represents a feasible service composition solution, and each dimension of the candidate solution must be an integer satisfying the boundary conditions. Consequently, the food source navigating and the approach of identifying candidate solutions need to be enhanced.

The objective of SCOS is to select and integrate the appropriate candidate services from each concrete services group, containing *n* abstract services $AS_{AP}$, where $AS_{AP} = \{as_1, \cdots, as_n\}$for an abstract process AP, and a set of composite patterns $CP = \{cp_1, \cdots, cp_m\}$ associated with each $as_i$ that satisfies the constraints on the basis of the previous work. Subsequently, each feasible solution is a service composition optimal selection solution.

## V. MQRSABC ALGORITHM MODEL

The main steps of the MQRSABC algorithm can be summarized as follows:

***Step 1. Initialization***

The number of food source is *SN*. In the initialization phase of the MQRSABC algorithm, *SN* feasible solutions (food sources) $\{x_1, x_2, ..., x_{SN}\}$ are generated and their quality value *qv(forP)*can be calculated using the RUQSS algorithm[afd]. Then, the fitness value of each food source are calculated using objective function (6).

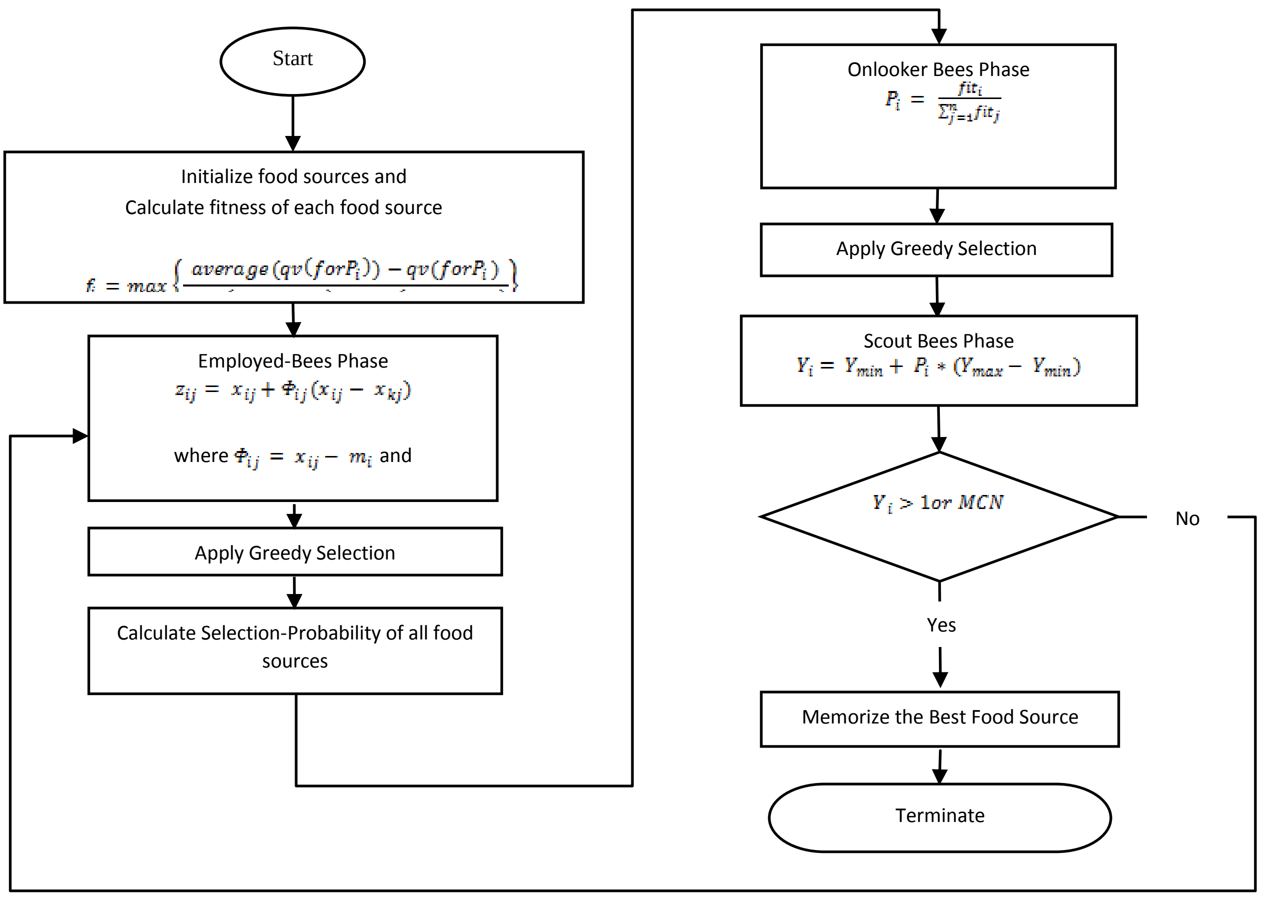
$$f_i = \left\{ \frac{average(qv(forP_i)) - qv(forP_i)}{max(qv(forP_i)) - min(qv(forP_i))} \right\} \quad where\ (1 \leq i \leq sn) \qquad (5)$$

Subject to

$$qv(forP_i) = \begin{Bmatrix} \max_{1 \leq i \leq sn} qv(forCP_i) & \forall + ve\ criteria \\ \min_{1 \leq i \leq sn} qv(forCP_i) & \forall - ve\ criteria \end{Bmatrix} \qquad (6)$$

Among the *SN* feasible solutions, the maximum value of the fitness $f_i$ will be memorized. After initialization, all the feasible solutions (food sources) will be exploited. Let the maximum cycle number be *MCN* and each cycle contains the behaviours of employed bees, onlookers and scouts.

**Fig 1: Flowchart of ABC-RUQSS Algorithm**



Start
Initialize food sources and
Calculate fitness of each food source
$f_i = max\left\{\frac{average(qv(forP_i)) - qv(forP_i)}{\quad}\right\}$
Employed-Bees Phase
$z_{ij} = x_{ij} + \Phi_{ij}(x_{ij} - x_{kj})$
where $\Phi_{ij} = x_{ij} - m_i$ and
Apply Greedy Selection
Calculate Selection-Probability of all food sources
Onlooker Bees Phase
$P_i = \frac{fit_i}{\sum_{j=1}^{n} fit_j}$
Apply Greedy Selection
Scout Bees Phase
$Y_i = Y_{min} + P_i * (Y_{max} - Y_{min})$
$Y_i > 1 or\ MCN$
No
Yes
Memorize the Best Food Source
Terminate

***Step 2. Employed bee phase***

Assign an employed bee to each food source, thus the number of the employed bees is *SN* too. At the beginning of the cycle, the employed bee j exploits the neighbourhood of the food source $x_{ij}$, j ∈{1, 2,..., *SN*}.

$$z_{ij} = x_{ij} + \Phi_{ij}(x_{ij} - x_{kj}) \qquad (7)$$

In the equation (7), $i^{th}$ is a dimension selected from the *m*-dimension feasible solutions, $i \in \{1, 2,..., m\}$, where $\Phi_{ij} = x_{ij} - m_i$ and $m_i = \frac{(x_{max} + x_{min})}{2}$. $z_{ij}$, represents the $i^{th}$ cycle with $j^{th}$ employee bee of the current optimal solution.

***Step 3. Onlooker phase***

Each onlooker selects a food source in the population depending on the selecting probability associated with that food source, and searches its neighbour to produce a new solution. Then, calculate the objective function value and apply the greedy selection strategy to update the position.

$$P_i = \frac{fit_i}{\sum_{j=1}^{n} fit_j} \qquad (8)$$

The selecting probability $P_i$ is calculated by (8), where

$$fit_i = \begin{cases} \frac{1}{1+f_i} & f_i > 0 \\ 1 + f_i & otherwise \end{cases} \qquad (9)$$

***Step 4. Scout phase***

In this phase, the scout is send to the search area for discovering a new food source. If there is no any updation after a maximum cycle number *MCN*, then the particular food source will be discarded. The associated employed bee will become a scout, and generate a new food source through (7). If there is any updation before the maximum cycle number *MCN*, then the particular food source will be updated through (10).

$$Y_i = Y_{min} + P_i * (Y_{max} - Y_{min}) \quad (10)$$

After all the food sources are explored, $z_{ij}$ and $Y_i$ will be memorized, and the next iteration will begin. The entire process is repeated until the condition satisfied. In the final phase, the reusability criteria has been defined by using the value of $Y_i$. If the $Y_i > 1$ then, it is proposed that the particular solution will be reuse.

## VI. Experimental Analysis

In order to evaluate the proposed approach MQRSABC algorithm, it is compared it with other two approaches such as PSO [20], ACO [21] in terms of search time, execution time and the reusability criteria.

### A. Evaluation Setup

The proposed approach is implemented on Windows 10 platform using Microsoft Visual Studio 12 .NET development environment with Microsoft Visual C# 4.5 as a programming language and MATLAB R2008a. Simple XML structures are used to create the service ontologies, and executed them on an Intel I Core™ i3 CPU, M 350 @ 2.27 GHz, 3 GB RAM Laptop with 100 MB/s Ethernet card.

### B. Experimental Results

In order to evaluate the proposed approach MQRSABC algorithm, it is compared it with other two approaches such as PSO [20], ACO [21] in terms of Search Time. In the figure 2, the search time of the proposed algorithm with respect to the number of services. Here, the number of services is 10. In the figure 3, the number of services is increased to 25. And finally, in the figure 4, the number of services is increased as 100.

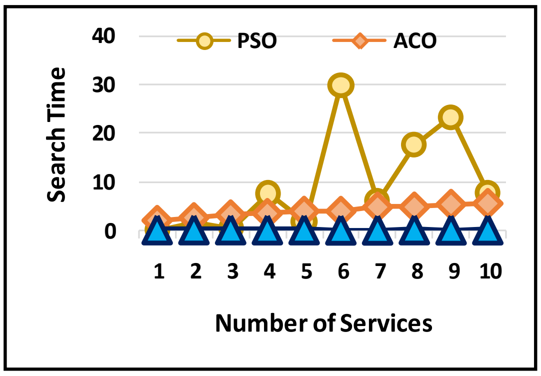


**Fig 2: Search time when Number of Services is 10**

The figure 2 depicts the searching time is compared with other existing approaches such as PSO, ACO, when the number of services is 10. However, the proposed approach MQRSABC has the less searching time. The average of PSO is high (9.5522) and the average of ACO is medium (3.83) and the average of MQRSABC is absolutely low (0.083).

While the searching time of the proposed approach is compared with PSO and ACO, which is represents in the figure 3, when the number of services is 25. However, the proposed approach MQRSABC has the less searching time. Again, the average of PSO is high (23.6732) and the average of ACO is medium (5.2336) and the average of MQRSABC is also low (0.2468).

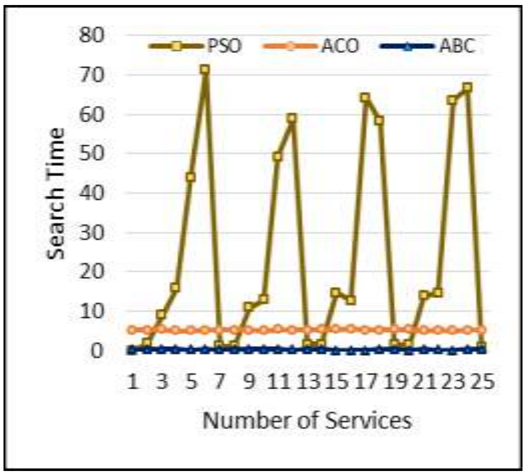


**Fig 3: Search time when Number of Services is 25**

When the proposed approach MQRSABC, while the number of services is 100, again it has the less (0.0598) searching time, when it is compared with PSO (23.3618) and ACO (5.2531). The figure 4 illustrates it.

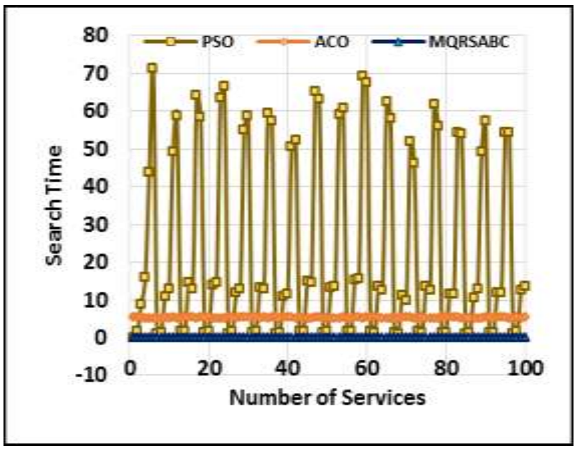


**Fig 4: Search time when Number of Services is 100**

When the number of services is 10, the execution time of the proposed approach is 0.010565 which is less than 1 second. When the number of services is 100, the execution time is about 0.018467 which is again less than 1 second. The time cost is very short and depicted in the table 1. Hence, according to its average execution time (0.014356), the proposed approach is much suitable to service selection than other two approaches for users with real-time requirements.

**TABLE I: COMPARISON OF WITH EXISTING APPROACHES**

| N | PSO | ACO | MQRSABC |
|---|---|---|---|
| 10 | 0.004457 | 0.009004 | 0.010565 |
| 25 | 0.005708 | 0.023324 | 0.014036 |
| 100 | 0.019815 | 0.017623 | 0.018467 |
| Average | 0.009993 | 0.01665 | 0.014356 |

The figure 5 depicts that the reusability of the services of the proposed approach MQRSABC is high, when compared with existing approaches.

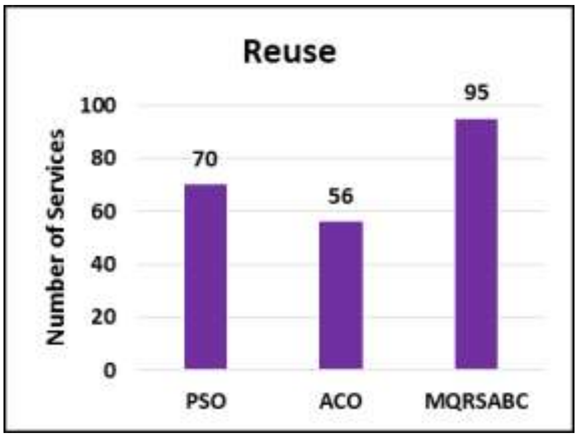


**Fig 5: Comparison of Reuse**

The figure 6 depicts that the execution time of the proposed approach MQRSABC is obviously short, regardless of the number of services.

According to the average execution time of the three approaches, the proposed approach is able to find best services with shorter search time rather than other existing approaches such as PSO and ACO.

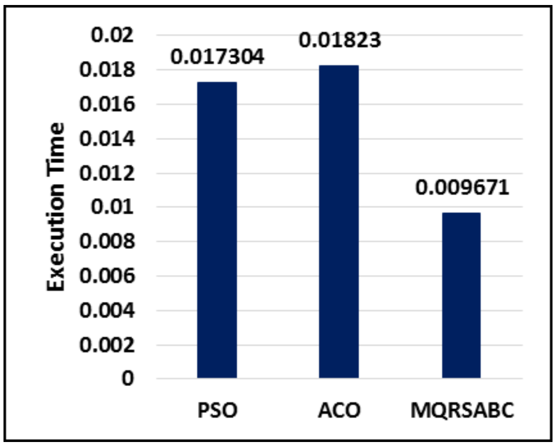


**Fig 6: Comparison of Execution Time**

The MQRSABC algorithm illustrates that from the experimental results of Table I, Figure 5 and 6, is much effective than other two existing approaches for Service Composition Optimal Selection.

## V. CONCLUSIONS

This paper proposed a novel QoS-based evolutionary algorithm with multi-objective constraints namely, the Multi-objective QoS-based Reusable Service Selection with Artificial Bee Colony (MQRSABC) algorithm which is the enhanced version of QoS-based Reusable Service Selection (RUQSS) Algorithm. The fast non-dictated population selection strategy is defined to measure the quality of the solution, to exploit the neighbourhood of the existing solutions and select the better ones. Likewise, a novel fitness calculation method is presented to calculate the selecting probability for onlookers, because the original single objective fitness calculation is not applicable for the multi-objective problem. The proposed algorithm is validated on benchmark functions in terms of search time, execution time and reusability constraint with the existing approaches PSO and ACO. The experimental results show that MQRSABC algorithm has the ability to provide competitive performance in terms of search time and execution time. In the future work, the practical applications of this algorithm will be researched.

## REFERENCES

[1] Nakamura Kazuto, et al. Value-based dynamic composition and evaluation of Web Services. IPSJ SIG Technical Reports, 2006, 9-16.

[2] Yilmaz, A. Erdinc, and Pinar Karagoz. "Improved genetic algorithm based approach for QoS aware web service composition." Web Services (ICWS), 2014 IEEE International Conference on. IEEE, 2014.

[3] Ding, Zhijun, Youqing Sun, Junjun Liu, Meiqin Pan, and Jiafen Liu. "A genetic algorithm based approach to transactional and QoS-aware service selection." *Enterprise Information Systems* 11, no. 3 (2017): 339-358.

[4] Zhang, An, et al. "Service composition based on discrete particle swarm optimization in military organization cloud cooperation." Journal of Systems Engineering and Electronics 27.3 (2016): 590-601.

[5] da Silva, Alexandre Sawczuk, et al. "Particle swarm optimisation with sequence-like indirect representation for web service composition." European Conference on Evolutionary Computation in Combinatorial Optimization. Springer, Cham, 2016.

[6] Al-Ani, Aymen Dawood, and Jochen Seitz. "QoS-aware routing in multi-rate ad hoc networks based on ant colony optimization." *Network Protocols and Algorithms* 7, no. 4 (2016): 1-25.

[7] Karaboga D., Akay B. A comparative study of artificial Bee colony algorithm. *Applied Mathematics and Computation*. 2009;214(1):108–132. doi: 10.1016/ j.amc.2009.03.090.

[8] Abro A. G. *Performance enhancement of artificial bee colony optimization algorithm [Ph.D. thesis]* 2013.

[9] Karaboga D., Basturk B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization*. 2007;39(3):459–471. doi: 10.1007/s10898-007-9149-x.

[10] Zhu G., Kwong S. Gbest-guided artificial bee colony algorithm for numerical function optimization. *Applied Mathematics and Computation*. 2010;217(7):3166–3173. doi: 10.1016/j.amc.2010.08.049.

[11] Gao W., Liu S., Huang L. A global best artificial bee colony algorithm for global optimization. *Journal of Computational and Applied Mathematics*. 2012;236(11):2741–2753. doi: 10.1016 /j.cam.2012.01.013.