Biomedical Research 2017; Special Issue: S96-S103 *ISSN 0970-938X*
www.biomedres.info

Optimized service oriented architecture for multi domain physician/patient interface system.

R Rajadevi[1*], RC Suganthe[2]

[1]Department of Information Technology, Kongu Engineering College, Erode, India

[2]Department of Computer Science & Engineering, Kongu Engineering College, Erode, India

Abstract

A physician extensively relies on computers for collection of data, Insurance, analysis of medication and also his schedule. The same applies to the service receiver that is the patient. This process involves interacting with different web domains. Web service refers to a software system which was formulated for supporting inter-operable machine-to-machine interactions across networks. At times, one web service provided alone does not fulfill user requirements. In most cases, many web services are required to be composed for achieving the user's aim. A number of web services offer the same functionalities however, they vary in non-functional Quality of Service (QoS) requirements. Given 'm' abstract service and 'n' concrete services for every abstract service, there are nm possibilities of composed solutions. As a result, the search space has large number of composed web services. Therefore it is necessary to discover the best composed web service with QoS constraints. The existing work addresses the web service composition task by considering various QoS constraints like Availability, Reliability, Response Time as well as Execution Costs. The Particle Swarm Optimization (PSO) protocol is applied for finding optimum solution by mapping the particles with the composed workflow. The proposed work combines PSO with Genetic Algorithm (GA) to improve the results of PSO since standard PSO is capable of rapidly causing particles to stagnate as well as premature convergence on sub-optimum solutions. The results show that the suggested hybridized model performs better than the standard PSO.

Keywords: Physician, Patient, Partical swarm optimization.

Accepted on January 14, 2017

Introduction

Web Services are software applications which may be described, published, located as well as invoked across networks. It is based on open standards such as, Hyper Text Transfer Protocol (HTTP) and eXtensible Markup Language (XML). Protocols formed out of the latter are Simple Object Access Protocol (SOAP) as well as Web Service Description Language (WSDL). Web Services are independent of hardware, programming languages or operating systems. This implies that applications coded in various programming languages or running on various platforms may exchange data across intranet or through the Internet via web services, which are powered by XML as well as 3 other core technologies: WSDL, SOAP, as well as Universal Description Discovery Integration (UDDI) [1]. WSDL refers to an XML-based form for description of web services. Client who is willing to access web services may read as well as interpret their WSDL files for learning about location of services as well as their available operations. WSDL file contains the location of the web service. SOAP refers to an XML-based protocol from W3C for sharing data across HTTP. It offers a basic standard technique for transmitting XML messages between applications. UDDI is a particularity to create XML-based registries which list information regarding information regarding businesses as well as their various web services. UDDI offers businesses a common method to list their services as well as finding services provided by others [2].

Table 1. Example for Web Service Composition.

Simple user request	Complex user request
Find the address of Apollo Hospital.	Book an appointment in Apollo Hospital, inform insurance company of impending operation and book hotel accommodation for attender.
A single service like 'hospital_finder' will satisfy this request.	Need three services like 'hosp_appointment', 'insurance_claims' and 'hotel_service' to satisfy this request.

Most of the time one service will not be able to offer the required functionality. Hence, combination of various services is required for producing the required functionality. This is known as composition which refers to the task of joining as well as linking already present web services for creating a workflow that achieves the user goal. Example for Web Service Composition is given in Table 1. A composite service can be described as a process that involves the execution of several standalone services to satisfy the user request. An

Abstract Service (AS) is a service that contains only the service interface definition. It manages the list of Concrete or Candidate Service (CS). The CS is a service, which provides the required capabilities of an AS.

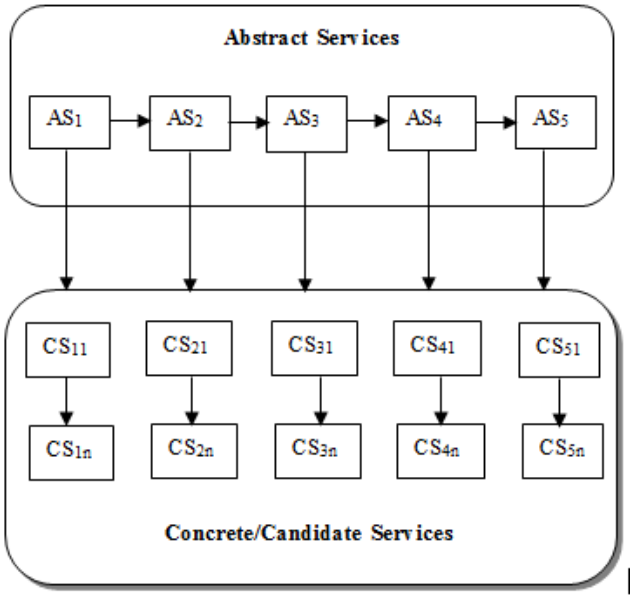

Figure 1. *Mapping of abstract services with concrete services.*

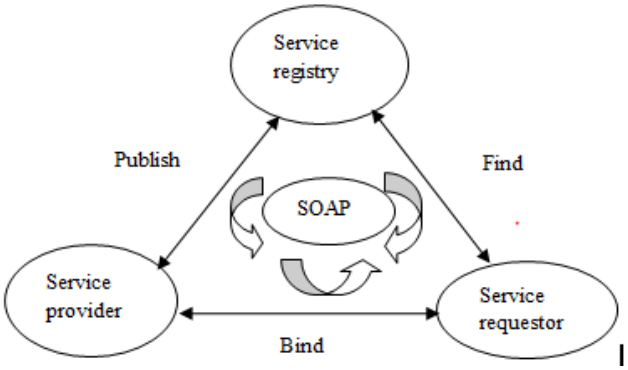

Figure 2. *Service Oriented Architecture (SOA).*

The service functionality or service activity that applies to a specific domain is referred as a task and is the fundamental component of composite web service. It contains four attributes-name, function, input and output. All tasks finish their related component functions with one of their candidate services [3]. The allowable order in which the operation of the service may be performed to fulfill the user request is referred as a workflow. A sample workflow is shown in Figure 1. The sample workflow is presented in Figure 1. The CSxy of each ASn provides the same functional, but various non-functional characteristics i.e. QoS features [4]. A Service-Oriented Architecture (SOA) is the base architecture that supports transmissions between Web Services [5]. The architecture of SOA is shown in Figure 2. There may exist 'n' possible solutions for a problem, but determining the best solution for that problem under certain circumstances becomes NP-Hard. For such problems, in order to find an efficient solution that best fits the problem statement, the optimization techniques are used. An optimization protocol refers to a process that is implemented in an iterative manner through comparison of several solutions until an optimal or adequate solution is reached.

Related Works

Xia et al. [6] suggested a novel multiobjective optimization based PSO for solving global optimization issue for based services choosing in web service composition technologies. The protocol takes web service choosing as a multiobjective restrained optimization issue with restrictions. Liao et al. [7] presented a service composition model for generic Service Overlay Network (SON) taking into consideration both several QoS restrictions as well as load balance factor. Furthermore, service selection protocol on the basis of niching method as well as PSO was suggested for service composition issue. Jiang-Hong and Zhao [8] proposed a query optimization system on the basis of aggregation of multiattribute QoS variables of various web services. Fanjiang et al. [9] introduced the SOA and orchestration technology turns into a trend of software development. The advantage of SOA was that it saves costs as well as time because the services utilized are already present and they are reused and integrated. Ding et al. [10] addressed the problem of choosing as well as composing web services through a genetic algorithm and provides a transaction as well as QoS-aware selection method. Firstly, it presents transactional characteristics of one web service as well as Composite Web Service (CWS) as well as the transactional rules utilized for composing them.

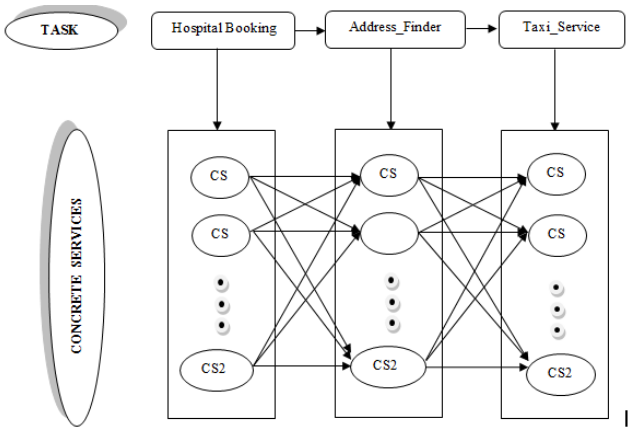

Figure 3. *Service composition process.*

Problem Definition

The composition of web services on the basis of functional parameters is not enough to satisfy the user goal. Therefore a composition based on non-functional parameters is also needed. In particular QoS attributes needs to be considered for binding abstract service with concrete service. In addition, the workflow generated must be dynamic. Given 'm' abstract service and 'n' concrete services for each abstract service, there are nm possibilities of composed solutions. The scenario for the service composition process as shown in Figure 3

considered three tasks namely H_Booking, Address_Finder and Taxi_Service. Each task is accomplished by using one of the candidate services. Hence, there are 3 abstract service and 20 concrete services for every abstract service. Therefore, 203 possibilities of composite services are generated to complete the user request.

To select the optimal service among this search space, an optimization technique is necessary which improves reliability, availability, response time and reduces execution cost. Metaheuristic techniques have been extensively used in literature to solve the NP problem. Among them Genetic Algorithm (GA), Particle Swarm Optimization (PSO) have been extensively used. GA suffers from slow convergence and PSO prematurely converges. For overcoming the restriction of particle swarm optimization, hybridized protocols with genetic algorithm are suggested. The idea underlying this is that such a hybrid method is anticipated to have benefits of particle swarm optimization with those of genetic algorithm. In this work three different hybrid approaches are proposed.

Methodology

The composition based on functional constraints alone is not enough to satisfy the service requestor and hence the existing work employs a QoS-driven Web Service composition PSO approach. The proposed work employs hybrid approach of PSO with GA. The resulting search space of web service composition consists of huge number of composed services and to select the optimal solution, PSO optimization technique is applied with GA to improve the results. Three different hybridization approaches of PSO-GA are,

- **Type 1:** In this approach, the total number of iterations are equally shared by PSO and GA.
- **Type 2:** In this approach, the population size is divided among PSO and GA on the basis of fitness values and the iterations are run successively by both the algorithms.
- **Type 3:** In this approach, the stagnated pbest particles alter their positions by mutation operator of genetic algorithm.

Hybridization of PSO with GA

Determine the fitness function: The objective function for QoS-driven Web Service composition using PSO with GA is a maximization function and is computed as given in the equation (1).

$$F = \sum_{k=1}^{4} \omega_k Q_k = \omega_1 \sum_{j=1}^{n} C_j + \omega_2 \prod_{j=1}^{n} A_j + \omega_3 \sum_{j=1}^{n} T_j + \omega_4 \prod_{j=1}^{n} R_j \rightarrow (1)$$

Where,

ωk-related weight of the i-th QoS (Qi) and $\sum_{k=1}^{4} \omega_i = 1$

Q_k-QoS attributes values (C, A, T, R) of a web services composition correspondingly.

n-task number; C-Execution cost; A-Availabilty; T-Response time; R-Reliability

Based on the user input, the weight ω_k can be varied but its summation must be equal to

Computing QoS attribute values: The positive features, that is, Reliability and Availability are taken the product as shown in the equation (2) and (3). The negative features, that is, Response time and execution cost are taken the summation as shown in equation (4) and (5). For sequential flow, the non-functional QoS attributes are computed as follows [11]:

$$R = \prod_{i=1}^{n} R_i \rightarrow (2)$$

$$A = \prod_{i=1}^{n} A_i \rightarrow (3)$$

$$T = \sum_{i=1}^{n} T_i \rightarrow (4)$$

$$C = \sum_{i=1}^{n} C_i \rightarrow (5)$$

Where, R-Reliability; A-Availability; T-Response time; C-Execution cost; i-iteration; n-number of services

Normalization of attribute value: Normalization of ratings implies adjustment of values assessed on various scales to a notionally common scale. QoS features may be sorted as either positive or negative. For a positive feature, for example, availability and reliability, a greater value denotes better quality, which is normalized as given in equation (6).

$$Q_i^{'} = \begin{cases} \dfrac{Q_i - Q_i^{\min}}{Q_i^{\max} - Q_i^{\min}}, & Q_i^{\max} - Q_i^{\min} \neq 0 \\ 1, & Q_i^{\max} - Q_i^{\min} = 0 \end{cases} \rightarrow (6)$$

A negative feature, for example, cost as well as response time, displays the opposite effect that is normalized as given in equation (7).

$$Q_i^{'} = \begin{cases} \dfrac{Q_i^{\max} - Q_i}{Q_i^{\max} - Q_i^{\min}}, & Q_i^{\max} - Q_i^{\min} \neq 0 \\ 1, & Q_i^{\max} - Q_i^{\min} = 0 \end{cases} \rightarrow (7)$$

Where, Q_i Attribute values before normalization; $Q_i^{'}$ Attribute values after normalization; $Q_i^{\min}$ Maximum attribute value of all quantified services; $Q_i^{\max}$ Minimum attribute value of all quantified services.

Mapping of Web Service Composition with PSO: One particle denotes a potential workflow, that is, each particle's position in search space ought to relate to a potential workflow. Workflow, i.e., the position, is executed as a vector. Dimensions in vector relate to workflows, and values relate to single services. Velocities are executed as lists of alterations which may be

employed to a particle as well as will move the particle to a novel position (novel workflow).

$X_i=(x_{i1}, x_{i2}, \dots, x_{in})$ denote the combinations of a composite service, wherein xi is the position of particles in PSO. $V_i=(v_1, v_2, \dots, v_m)$ is the velocity utilized to update the direction of particles, where $v_i=0$ or 1.

Velocity and Position Update equation for PSO: The Velocity update equation for web service composition optimization problem is given in below equation (8):

$$V_{i+1} = p_1 V_i \oplus p_2\left(P_i \Theta X_i\right) \oplus p_3\left(P_g \Theta X_i\right) \rightarrow (8)$$

Where, Pi – Local best; Pg – Global best; p1 – 0.2; p2 – 0.3; p3 – 0.5

The Position update equation for web service composition optimization problem is given in below equation (9).

$$X_{i+1} = X_i \circledast V_{i+1} \rightarrow (9)$$

The addition, multiplication and subtraction operations are redefined to suite the web service composition problem as follows:

Subtraction (Θ): Xi Θ Xj means the position variance of particles Xi and Xj. The result is 1 if the corresponding values of the vector Xi and Xj are the same. Otherwise the result is 0.

For example,

X_i = (1, 2, 3, 4, 4, 5)

X_j = (1, 3, 2, 4, 5, 5)

Then,

$X_i \Theta X_j$ = (1, 2, 3, 4, 4, 5) Θ (1, 3, 2, 4, 5, 5) = (1, 0, 0, 1, 0, 1).

Addition (⊕): refers to an operator which represents the search direction, while the outcome is a novel velocity. p1V1 ⊕ p2V2 ⊕ … ⊕ pnVn is the equation for velocity updation of particles. It states that the outcome of the equation keeps V1 with probability p1, … , and keep Vn with the probability Pn in respective dimension, wherein p1+p2+…+pn = 1.

For instance,

p1 = 0.1

p2 = 0.9

V1 = (1, 0, 0, 1, 1, 1)

V2 = (1, 0, 1, 0, 1, 0)

Then,

p1V1 ⊕ p2V2 = 0.1(1, 0, 0, 1, 1, 1) ⊕ 0.9(1, 0, 1, 0, 1, 0) = (1, 0, *, *, 1, *)

Where * represents uncertain to be 0 or 1. In this instance, the first is 0 with probability of 0.1 and 1 with probability of 0.9.

Multiplication (⊛): represents updation procedure of particle's position and the outcome is a novel position. Operating rules: if the value of Vi is 1, the value of Xi remains the same otherwise, the value of Xi will change.

For example,

P1 = (1, 2, 4, 3, 2, 5)

V1 = (1, 0, 1, 0, 0, 1)

Then,

P1 V1 = (1, *, 4, *, *, 5)

Where * denotes the value can either added or subtracted based on the random threshold value set in the algorithm.

Mapping of web service composition with GA: The chromosomes represent the possible workflows. Each gene represents candidate solution within each task. Therefore set of genes constitute a composite service that satisfies the user goal. The total number of chromosomes represents the population size.

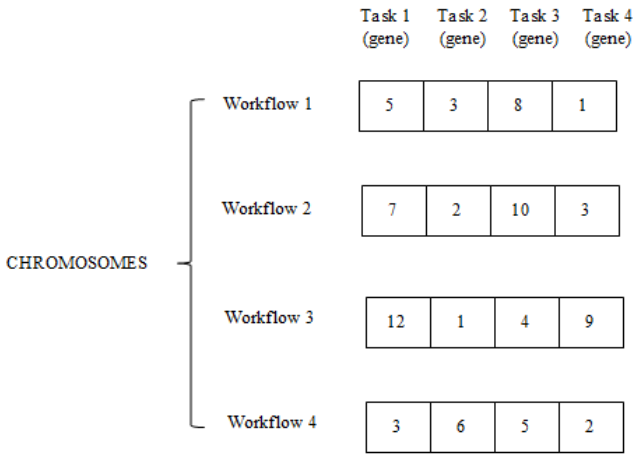

Figure 4. *GA operator–Encoding*

GA operators–encoding, selection, crossover and mutation: In GAs, chromosomes or genomes denote the potential choices present in the population. Potential solutions are coded with genomes. Here in web service composition, composite services are encoded as chromosomes. Every gene in the chromosome codes atomic services for every task. Every gene denotes selected atomic service from every service set. For instance, in Figure 4, atomic service S5 is selected to execute task1 in workflow1. In this instance, the workflow comprises 4 tasks, so the chromosome comprises 4 genes. The proposed work employs real value encoding [12]. The fittest chromosomes have the higher probability to be selected for the next generation. So the selection process involves selection of chromosomes with higher fitness values. The next step is the crossover between two parent chromosomes decided by a crossover probability followed by mutation.

Hybrid PSO-GA type I

In this approach, the total number of iterations is equally shared by PSO and GA. Here first half iterations are run by PSO and second half iterations are run by PSO. For example,

total iterations of 150 can be shared as 75 iterations for PSO and 75 iterations for GA. Since GA outperforms PSO, the fitness values resulted by PSO is maximized during the iterations of GA. Thus this approach combines PSO with GA using the iterations count [13].

Algorithm for Hybrid PSO-GA Type I

Initialization of population

Initialization of QoS parameters

Calculate initial_fitness_value using the equation (1)

DO

FOR iterations <=iterations/2

Find initial_lbest and initial_gbest

Update workflow's velocity and position using equation (8) and (9)

END

FOR iterations>iterations/2

Perform Selection, Crossover and Mutation

END

Output best composed workflow

UNTIL maximum iterations or minimum error criteria is attained

Hybrid PSO-GA type II

In this approach, the population size is divided among PSO and GA based on the fitness values and the iterations are run successively by both the algorithms. Here, the population is ranked based on the fitness values. The population that have best fitness values are given input to PSO and the population that have the worst fitness values are given as input to GA since GA computes better result than PSO. For example, population size of 50 workflows are divided into two category based on the fitness value evaluation. The population which have top 25 best fitness values are iterated using PSO and the population which have least 25 worst fitness values are iterated using GA. Thus this approach combines PSO with GA using population count based on fitness function evaluated [14].

Algorithm for Hybrid PSO-GA Type II

Initialization of population

Initialization of QoS parameters after normalization using equation (6), (7)

Calculate initial_fitness_value using the equation (1)

DO

Rank the population based on fitness value

FOR best fitness values

Find initial_lbest and initial_gbest

Update workflow's velocity and position using equation (8) and (9)

END

FOR worst fitness values

Perform Selection, Crossover and Mutation

END

Output best composed workflow

UNTIL maximum iterations or minimum error criteria is attained

Hybrid PSO-GA type III

In this approach, the stagnated pbest particles alter their positions through mutation operator of genetic algorithm. Here the particles of PSO which does not change their positions for designated iterations are applied the mutation operation of GA. Thus stagnated pbest particles alter their positions and results in higher fitness values [15].

Algorithm for Hybrid PSO-GA Type III

Initialization of population

Initialization of QoS parameters after normalization using equation (6), (7)

Calculate initial_fitness_value using the equation (1)

DO

Find lbest

FOR stagnated lbest particles

Perform Mutation

END

Find gbest

Update workflow's velocity and position using equation (8) and (9)

END

Output best composed workflow

UNTIL maximum iterations or minimum error criteria is attained

The mapping of web service composition with PSO is shown in Table 2.

Table 2. *Mapping PSO and GA with web service composition.*

PSO	GA	QoS-Driven web service composition
Particle	Chromosome/Genome	Workflow
Position	-	Implemented as a vector. (Dimensions-workflow, Values-single service)

Velocity	-	Implemented as a vector with [0, 1] values.
-	Gene	Candidate Service within a task
No_of_particles	No_of_chromosomes	Population Size

Evaluation of fitness function: Fitness function is calculated by taking summation of all four QoS attributes along with ω_i (weight defined for each QoS). For example, ω_1=0.2 for 'C' i.e., Execution cost, ω_2= 0.2 for 'A' i.e., Availability, ω_3=0.2 for 'T' i.e., Response Time and ω_4= 0.4 for 'R' i.e., Reliability. The summation of all four QoS attributes results in 1.

QoS computation: Positive QoS attribute values-Availability and Reliability are computed using the equation (2) and (3). Negative QoS attribute values-Response time and Execution cost are computed using the equation (4) and (5).

Normalization of QoS attributes: For preventing wrong evaluations because of several measurements, measures of QoS features ought to be normalized to some scale, that is, lower execution price, shorter response time, higher availability and reliability. Using equations (6) and (7), the QoS attribute values are normalized.

Hybrid approaches: PSO-GA Type I - In this approach, the total number of iterations are equally shared by PSO and GA as shown in the Section 4.2. PSO-GA Type II - In this approach, the population size is divided among PSO and GA based on the fitness values and the iterations are run successively by both the algorithms as shown in the Section 4.3. PSO-GA Type III - In this approach, the stagnated pbest particles alter their positions through mutation operator of genetic algorithm as shown in the Section 4.4.

Results and Discussion

Experiments are formulated for measuring fitness values with the number of iterations. The QWS Dataset is used as the QoS data set of potential services. The generated workflows consist of varying number of tasks with fixed number of potential service in each task.

The parameters are set to:

Population size = 40

Number of iterations = 150

ω_1= 0.2

ω_2= 0.2

ω_3= 0.2

ω_4= 0.4

p1 = 0.2

p2 = 0.3

p3 = 0.5

Mutation rate = 0.2

Selection rate = 0.5

The fitness values of PSO, GA, Hybrid PSO-GA - Type I, Type II, Type III are measured by varying the number of tasks and candidate services. The tasks are varied as 10, 20, 30, 40 web service instances and every task contain 20 potential services with different QoS constraints. The search space of the above four composite web services contains 20^{10}, 20^{20}, 20^{30} and 20^{40} number of workflows respectively. The best fitness value for varying number of tasks with corresponding candidate services is given in Table 3.

Table 3. *Fitness values for different task number.*

Number of Candidate services	Number of Tasks	Algorithm	Best Fitness Values	Computational Time (Seconds)
20	10	PSO	3.1077	1.4905
		GA	3.7959	40.7868
		Type I	3.2375	13.5842
		Type II	3.0629	1.4269
		Type III	3.9326	92.0029
	20	PSO	5.2102	1.8255
		GA	6.3025	121.8498
		Type I	5.7795	48.2124
		Type II	5.5230	1.7214
		Type III	6.5076	343.7339
	30	PSO	7.6774	2.1578
		GA	7.9268	225.5123
		Type I	7.8012	107.6697
		Type II	7.7332	2.2328
		Type III	7.9974	679.7698
	40	PSO	9.7717	2.8192
		GA	11.5076	433.1417
		Type I	10.2872	189.8606
		Type II	10.1512	2.6416
		Type III	11.7410	1459.6024

Number of iterations *vs.* fitness value in case of PSO, GA, Hybrid PSO-GA - Type I, Type II, Type III with 10 tasks and 20 candidate services is shown in Figure 5. From this graph, the best fitness value 3.7959 is got for GA than 3.9326 is obtained for the Hybrid PSO-GA Type III. Number of iterations *vs.* fitness value in case of PSO, GA, Hybrid PSO-GA - Type I, Type II, Type III with 20 tasks and 20 candidate services is shown in Figures 6 and 7. From this graph, the best fitness value 6.3025 is got for GA than 6.5076 is obtained for the Hybrid PSO-GA Type III. Number of iterations *vs.* fitness value in case of PSO, GA, Hybrid PSO-GA - Type I, Type II, Type III with 40 tasks and 20 candidate services is shown in

Figure 8. From this graph, the best fitness value 11.5076 is got for GA than 11.7410 is obtained for the Hybrid PSO-GA Type III.

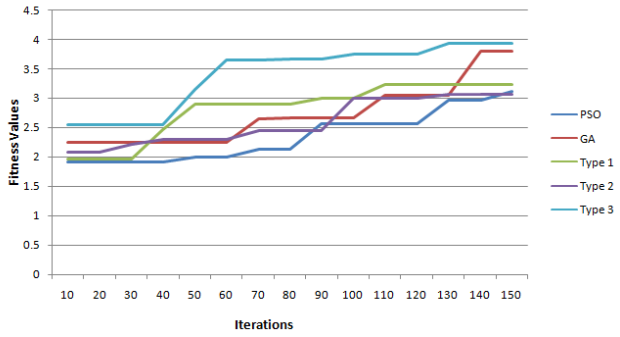

Figure 5. *Number of iterations vs. fitness value with 10 tasks and 20 candidate services.*

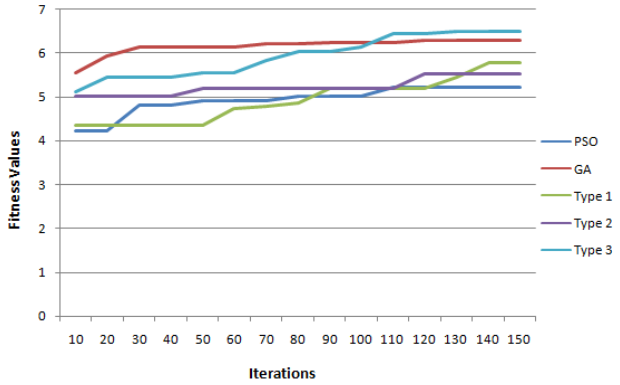

Figure 6. *Number of iterations vs. fitness value with 20 tasks and 20 candidate services.*

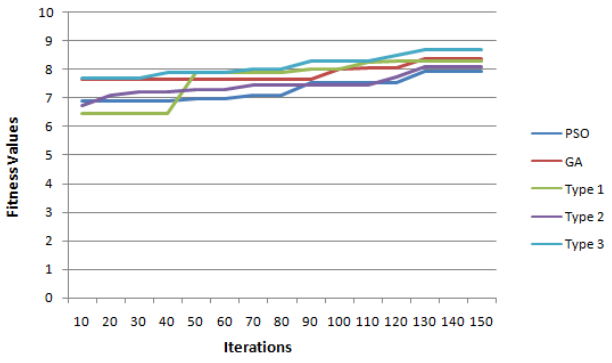

Figure 7. *Number of iterations vs. fitness value with 30 tasks and 20 candidate services.*

As expected, all the three hybrid PSO-GA displays an excellent increase in the fitness value in contrast to standard particle swarm optimization when increasing the quantity of tasks in the workflow that satisfies the user goal. The result analysis shows that, Hybrid PSO-GA Type III outperforms all the other algorithms.

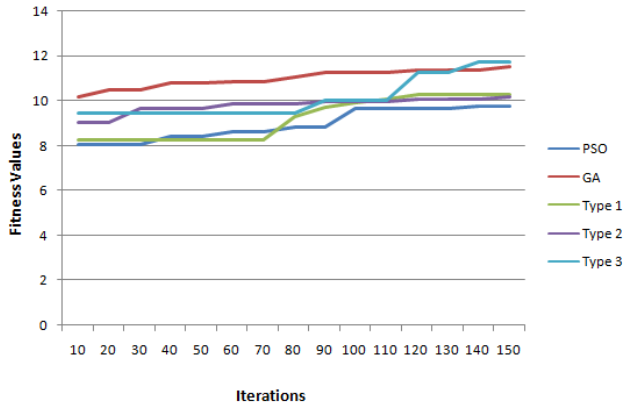

Figure 8. *Number of iterations vs. fitness value with 40 tasks and 20 candidate services.*

Conclusion

Web services selection with QoS restrictions is an ongoing research topic. It is necessary to choose which services are to be utilized in composite web service as per service requestor's QoS requirements. This project addressed the service composition task using Hybrid PSO i.e., PSO with GA based on QoS attributes since functional attributes alone is not enough for fulfilling user demands. Also the dynamic workflow for service composition is considered. The suggested hybridized particle swarm optimization models discover improved solutions without being forced into local maxima and attain more rapid convergence rates. This is due to the fact that when particle swarm optimization particles stagnate, genetic algorithm diversifies particle positions. In PSO-GA, particle motion utilizes arbitrariness in its search. Therefore, it is a type of optimization protocol which can search complex as well as uncertain areas. This ensures that PSO-GA is more flexible as well as resilient. Unlike generic PSO, PSO-GA is more dependable in providing improved quality solutions with adequate computation time, because hybrid scheme obviates premature convergence of searches to local optimum and offers improved exploration of searches.

References

1. Allameh-Amiri M, Derhami V, Ghasemzadeh M. QoS-Based web service composition based on genetic algorithm. J AI Data Mining 2013; 1: 63-73.
2. Pejman E, Rastegari Y, Esfahani PM, Salajegheh A. Web service composition methods: A survey. In Proceedings of the International MultiConference of Engineers and Computer Scientists (Vol. 1), 2012.
3. Sheng QZ, Qiao X, Vasilakos AV, Szabo C, Bourne S, Xu X. Web services composition: A decade's overview. Informa Sci 2014; 280: 218-238.
4. Alrifai M, Risse T, Nejdl W. A hybrid approach for efficient Web service composition with end-to-end QoS constraints. ACM Transactions on the Web (TWEB) 2012; 6: 7.

5. Bozkurt M, Harman M, Hassoun Y. Testing and verification in service-oriented architecture: a survey. Software Testing Verification Reliability 2013; 23: 261-313.
6. Xia H, Chen Y, Li Z, Gao H, Chen Y. Web service selection algorithm based on particle swarm optimization. In Dependable, Autonomic and Secure Computing, 2009.
7. Liao J, Liu Y, Wang J, Zhu X. Service composition based on niching particle swarm optimization in service overlay networks. KSII Transact Internet Informa Syst (TIIS) 2012; 6: 1106-1127.
8. Jiang-Hong J, Zhao W. An optimization model for dynamic QoS-aware web services selection and composition. Chinese J Computers 2009; 5: 019.
9. Fanjiang YY, Syu Y, Wu CH, Kuo JY, Ma SP. Genetic algorithm for QoS-aware dynamic web services composition. Int Confer Machine Learn Cybernet 2010; 6: 3246-3251.
10. Ding Z, Liu J, Sun Y, Jiang C, Zhou M. A transaction and QoS-aware service selection approach based on genetic algorithm. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2015; 45: 1035-1046.
11. Wang W, Sun Q, Zhao X, Yang F. An improved particle swarm optimization algorithm for QoS-aware web service selection in service oriented communication. Int J Comput Intel Syst 2010; 3: 18-30.
12. Zhao X, Song B, Huang P, Wen Z, Weng J, Fan Y. An improved discrete immune optimization algorithm based on PSO for QoS-driven web service composition. Applied Soft Computing 2012; 12: 2208-2216.
13. Wu SY, Zhang P, Li F, Gu F, Pan Y. A hybrid discrete particle swarm optimization-genetic algorithm for multi-task scheduling problem in service oriented manufacturing systems. J Central South University 2016; 23: 421-429.
14. Ludwig SA. Applying particle swarm optimization to quality-of-service-driven web service composition. In 2012 IEEE 26th International Conference on Advanced Information Networking and Applications, IEEE.
15. Rahmani R, Othman MF, Yusof R, Khalid M. Solving economic dispatch problem using particle swarm optimization by an evolutionary technique for initializing particles. J Theor Appl Informa Technol 2012; 46: 526-536.

***Correspondence to**

R Rajadevi

Department of Information Technology

Kongu Engineering College

India