An Improved Fuzzy Guided Genetic Algorithm for the Selection of Web Services

K. Mohan[# 1], C. Kalaiarasan[*2]
[#]Department of AM & CS, PSG College of Technology, Coimbatore, Tamil Nadu, India,
[1]mrmohan78@gmail.com
[*] Tamilnadu College of Engineering, Coimbatore, Tamil Nadu, India,
[2]ckalai2001@yahoo.com

Abstract— Web services are integrated web-based applications that function over a protocol backbone. Web service selection is one of the most important steps in many of the web service applications such as UDDI registries and web service repositories. Each web service has its own functionality which matches the service request. QoS plays a major role in selecting web services in terms of qualitative measure. The problem addressed in this paper is to select the exact web service in a dynamic environment which matches the consumer request with optimum solution quality. At first, a three-parent genetic algorithm has been introduced in this paper and compared with the existing two parent GA. Next An Improved FGA is proposed that considers the user's preference given for the quality attributes and looks for the services in dynamic composite services. As in FGA, the fuzzy controller adjusts the crossover and mutation rates accordingly and the Gaussian membership function is used in fuzzy engine to obtain finer granularity in fuzzy sets. As a result an exact service that matches the consumer's quality requirements is obtained. Results were compared with the existing FGA and presented.

Keywords- Composite Services, Fuzzy controller, Gaussian Membership Function, QoS.

I. INTRODUCTION

Soft computing provides a model for cognitive behaviour of human mind. Soft computing aims to exploit imprecision tolerance, uncertainty, robustness and low solution. The constituents of soft computing are fuzzy logic, Probabilistic reasoning and Neuro computing [1]. These are intelligent technologies that have drawn an increasing attention in web services. Web services are becoming technology that is used to share data in different domains. Web Service acts as an interface that describes a set of operations that are accessed through network using standardized XML language. This description using XML notion is termed as service description. It contains all the necessary details to communicate with the services such as messages, protocols and destination. Since web services are platform and language independent these are loosely coupled and component oriented. Programs that provide simple services may interact with each other for delivering sophisticated value added services called as composite services. The processes of developing the composite services are called service composition [2].

To develop such composite services, it is necessary to discover and select those web services based on service requests. Web Service Description Language (WSDL) standard is used to describe the functional aspects of web service in the form of a service description that is expressed in Universal Description, Discovery and Integration (UDDI) registries [3]. When service request is given it is compared with the available Web service description to find the service that provides exact functionality. As the web service increases, there can be more than one web service which satisfies the user requirements. They differ in Performance, Response time and security. Similar to the service description, the quality of services (QoS) is also provided by the providers. The QoS is of two types that is technical and managerial. The general technical QoS attributes includes reliability, throughput, response time, reputation, cost and availability. The managerial QoS attributes has ownership, contract, payment, service provider [4]. Therefore, there must be a standardized mechanism based on the consistent view of non-functional requirements [5].

The objective of the web service selection is to maximize the QoS values. Selected web service should have high reputation, reliability and availability whereas the cost and response time should be less. Since the services come and goes online dynamically there must be a mechanism to handle the dynamic nature of web services. There are some related works which enable the users to select web services dynamically. But these works failed to express the quality and the optimum result generation. In this paper, we propose a stochastic model for the web service selection using Improved FGA technique. The usage of the fuzzy algorithm with genetic algorithm helps in finding the better match for the service request. The fuzzy logic controller is used to adjust the crossover and mutation rates dynamically for every ten generations thereby increasing the solution quality [5]. Here, fuzzy Gaussian membership function is used instead of triangular membership function to increase the reliability and robustness of the system in fuzzy logic design [6] and the reproduction phase in genetic algorithm follows a three-parent crossover. A study of related works is done and the results are compared. The remainder of this paper is structured as follows Section-II describe the problem formulation, Section-III and Section-IV reviews the related work done. Section-V and Section-VI presents the proposed improved fuzzy genetic algorithm. In Section VII the experiments and results of this study were analysed and compared, Section-VIII presents the conclusions drawn from the proposed work.

II. PROBLEM FORMULATION

A. Web Service Selection Problem

Consider a travel plan domain given in Fig. 1. A person needs to spend his vacation and looks for a travel Agency that offers best services at low cost. The services which the Agent offers are:

- Airline ticket Reservation
- Renting a car or bus from Airport to hotel
- Hotel Booking
- Engaging a guide

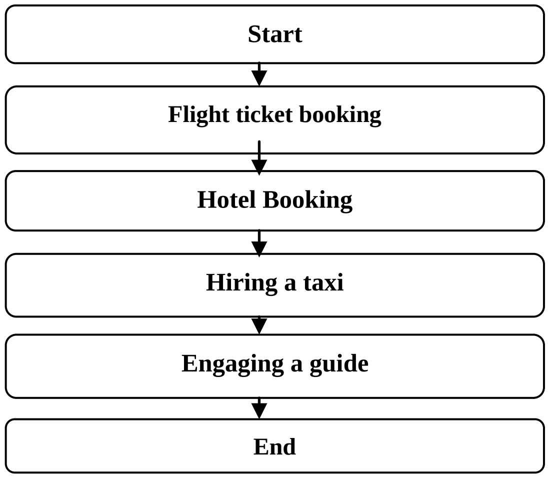

Fig. 1. Example for Travel Agent Web Service

The service providers (Taxi, guide, Airlines, hotel) provide web services and offerings to perform reservations. To guarantee payments made by the customers credit card companies also offer services.

Since the nature of the web service is loosely coupled the agent may not have any prior agreement with the service providers thereby allowing independent access to more web services and offer more options to customers and to receive sophisticated services from credit card companies making the customers happy and the service providers can offer services broadly and therefore generating business for themselves.

The problem addressed here is simple and the agent must be able to get the exact services what customers are looking from the service provider. The customer looks for quality in the service offered by the agents. The agent must choose the service with quality at lowest price. Hence, quality plays a major role. The web service architecture must be remodelled to meet the consumer needs.

Given,

- A composite web service consisting of number of web services and it is denoted as $\overline{W}$s= {$\overline{W}s_1$, $\overline{W}s_2$, $\overline{W}s_3$…$\overline{W}s_n$} where n is the number of services available in web service registry.
- QoS values for the Reliability, Availability, Reputation, cost, Response time are $Q_{ij}^{Rel}, Q_{ij}^{Ava}, Q_{ij}^{Rep}, Q_{ij}^{Cost}, Q_{ij}^{Respt}$
- Weights assigned for the QoS parameters are $W_{Respt}, W_{Rel}, W_{Rep}, W_{Cost}, W_{Ava}$

  In general the QoS criteria must be $Q_{ij}^{Rel}, Q_{ij}^{Ava}, Q_{ij}^{Rep} > Q_{ij}^{Cost}, Q_{ij}^{Respt}$

III. RELATED WORK

Most of the user feel that the result generated in their web search are infeasible and unrelated. This has drawn the researcher's attention to concentrate on the web service selection problem. Some of the related work in this area includes Dynamic QoS management, Web service selection using different algorithms.

Imprecise QoS constraints allow users to flexibly specify the QoS demands. Fuzzy logic is proposed for the web service discovery and selection problem. Methods for specifying fuzzy QoS constraints and for ranking web service based on their fuzzy representation were discussed [3]. Fuzzy controller is used to control the crossover and mutation rates. As fitness increases the mutation and crossover rates also increases [4].

Global QoS optimizing and multi-objective ant colony optimization proposed for the dynamic web service composition. A model of web service selection with QoS global optimization is taken and converted in to a multi-objective optimization with user constraints [5]. The evolution of fuzzy classifier for data mining is the current research topic. An application of genetic programming to the evolution of fuzzy classifiers can detect faulty products and shall discover intrusions in computer network [6]

A heuristic approach based on Hill-Climbing which makes use of linear programming that works on a reduced space can reduce the time complexity and find near optimal solutions [7]. A fuzzy QoS architecture is proposed to include user preferences and quality of services. An approach for storing and retrieving syntactic and semantic information about web services and Semantic Annotated WSDL to create semantic annotated web service description is proposed [8].

A collaborating filtering approach for predicting the QoS values and web recommendations based on the previous users experience of the users. [9]. A method to determine the extent to which a service satisfies the QoS requirements is done by evaluating satisfaction scores for each service and the overall satisfaction score is used to determine the exact match instead of finding a best service which over qualifies the QoS requirements of the user [10].

A penalty based genetic algorithm is proposed to handle the web service composition problem. A local optimizer is used to improve the individuals in the population. If the infeasible individuals are excluded from the population the genetic algorithm may not produce optimal results. Hence, a penalty is given for the infeasible individuals and included in the population [11]. A QoS based dynamic service composition with Ant colony optimization for web services is proposed. A multi-objective optimal-path selection for QoS-based dynamic web service composition problem is addressed with the Ant colony optimization (ACO) algorithm [12].

Composition of web service through Domain Specific language for the specification of functional requirements and expected QoS in the form of constraints is proposed. Two different techniques for optimization were discussed constraint programming and integer programming approach [13].

In this paper we are following the idea of using Genetic Algorithm for the optimization problem and fuzzy logic to match the QoS parameters of requestor and provider. The fuzzy logic controller is used to control the crossover and mutation rates for every 50 generations and the result is given to fuzzy engine, so that an optimal solution with best the quality is obtained.

IV.MATERIALS AND METHODS

B. QoS, Weight Metrics and Fitness Function Evaluation

The QoS may be of two types static and dynamic. A static property of a web service is defined at the time when it is described and the dynamic property requires measuring and updating the value periodically. The QoS values can be positive or negative. The QoS values are generated dynamically between (0,1) [11]. For example, customer may want to buy a service with low cost and low response time which leads to negative values whereas the performance and integrity might result in positive values [14]. The QoS values considered are Reliability, Availability, Response time, Cost and Reputation.

1) Reliability: Reliability was measured from the ratio of all of the times that the users request for the service successfully divided by all of the times that the users request for the service in specific time which can be calculated from the following equation:

$$\text{Rel} = \frac{T_s}{T_A} \tag{1}$$

Where, Rel is the reliability. T_s denotes all the times that the users request for the service successfully in a specific time. T_A means all of the times that the users request for the service in specific time.

For instance, the users requested for services four times in the specific time, but three of them are successfully, while the remaining one is failed. So, the reliability is ¾ or 0.75.

2) *Availability:* It is the ratio of period of time the service is accessed successfully and the period of time the requestor spends in requesting the service.

$$\text{Ava}=\frac{N_S}{N_A} \tag{2}$$

Where Ava is the Availability, N_S refers to the time service is accessed and N_A refers to the time requestor spends in requesting the service

3) Response time: Response time refers to the time since the users sends the request for a particular service and the time taken by the server to response to the request.

$$\text{Respt}= \frac{1-R_T}{\sum R_{T_i}} \tag{3}$$

4) Cost: Cost refers to the ratio of service charge laid by the providers and rate of service charge of all service providers; this can be calculated by the following equation

$$\text{Cost} = \frac{1-S_C}{S_{ci}} \quad (4)$$

S_C is the service charge; S_{C_i} is the sum of rate of service charge of all service providers. [15]

5) Reputation: Reputation refers to the trustworthiness. It mainly depends on the user's experience towards a service. It is measured by the following formula.

$$\text{Rep} = \frac{\sum_{i=1}^{n} R_i}{N_S} \quad (5)$$

Where R_i is the ranking of the service based on user's experience, N_S is the number of times the service has been graded [16].

C. Fitness Evaluation

A QoS matrix is built to record the quality information of each service (Q_{ij}, where 1<i<n, 1<j<m) in which columns represent to the quality attribute and rows corresponds to services. The quality attributes may be positive or negative. Higher the value higher the quality. The overall fitness function is given by,

$$F_{ij} = \frac{W_{Respt} Q_{ij}^{Respt} + W_{Cost} Q_{ij}^{Cost}}{W_{Ava} Q_{ij}^{Ava} + W_{Rel} Q_{ij}^{Rel} + W_{Rep} Q_{ij}^{Rep}} \quad (6)$$

The above formula is used to compute the overall fitness score for each web service.

D. Weight Metrics

The users can express their preference towards service by specifying the weight given to each quality attribute. For each consumer request R_i there are l_i candidate services that are available to which the request can be assigned. The problem addressed in the QoS based service selection is to select an exact match for the request from the available candidate services so that the overall QoS of the composite service can be maximized.

i.e. $\text{Max}\ (\sum_{i=1}^{n} \sum_{j=1}^{l_i} W_{ij}\, F_{ij})$

$$\sum_{j=1}^{l_i} W_{ij} = 1 \quad (7)$$

Where F_{ij} is the fitness score of the j^{th} candidate service for the R^{th} request in the service composition.

V. THREE PARENT GENETIC ALGORITHM

E. Genetic Algorithm

Genetic algorithms (GA) were invented by John Holland in 1960. The GA works by creating a random population of candidate solutions for an optimization problem to be solved for a better solution. This is an iterative process in which population at each iteration is termed as generation. The fitness is evaluated at each generation for every individual and the function used is known as objective function or fitness function. The more fitted solutions are identified and the parent which produces it were altered (using generic operators: Crossover and Mutation) to produce a new offspring. The same algorithm is applicable for the newer generation. This algorithm continues until maximum generations are reached or the satisfactory fitness level is obtained.

F. Two –Parent Genetic Algorithm

A two parent genetic algorithm takes two individuals from the population, consider it to be the parents and produces off spring accordingly. The parents are selected based on the fitness score of the individuals. After the selection, crossover is made. Algorithm for two-parent genetic algorithm is given in Fig 2.

1) Selection: Tournament Selection is a most popular methodology for off spring selection because of its easy implementation and efficiency. The n individuals are randomly selected from the huge population and the selected individuals are made to compete with one another. The individual with highest fitness wins and is placed in the next generation. The number of individuals in each tournament is referred to as tournament size (k) [20].

1. // **Set GA parameters**
   n->population size, G->Total Number of Generations
   μ-> Mutation Rate, α->Crossover Rate
2. // **Initialize the population**
   P_k: population created randomly with specified population size
3. // **Evaluate the P_k**
   Calculate fitness (i) for each I ∈ P_k
   do
   {
   Selection:
   Tournament selection with specified k value is applied for selection.
   Crossover:
   Select pairs from sorted list, perform crossover such that α x n =No. of pairs to be coupled for mating
   Place the newer off springs in P_{k+1}
   Mutation:
   Select μ x n individuals of P_{k+1}; invert the bits of selected individual.
4. // **Evaluate P_{k+1}**
   Compute fitness (i) of each individual, i ∈ P_{k+1}
5. // **Increase the generation**
   k=k+1;
   }
   Return fittest individual at each generation, Continue steps 3 to 5 until the specified G is reached.

Fig. 2. Algorithm for Two-Parent Genetic Algorithm

2) *Two-Parent Uniform Crossover:* In two parent uniform crossover, offspring is produced by copying the bits randomly from the first parent or from the second parent. The off spring thus produced acquires the genes of both parent and thus becomes a better solution than parents. For example (see TABLE I)

TABLE I Example of Two-parent uniform crossover

First Parent	**11001011**
Second parent	**11011101**
Off Spring 1	**11011111**
Off Spring 2	**11001001**

G. Three-Parent Genetic Algorithm

Three-parent genetic algorithm in turn takes three individuals from the population and produces three off spring. At first, fitness is calculated for each individual in the population. The individuals are ranked based on the fitness score. The individual with highest score is chosen as the first parent, second and third parent are taken by measuring the distance from the first parent. The three parents are altered and mutated by the process of crossover and mutation to produce newer off spring. The algorithm for three parent genetic algorithm is shown in the Fig. 3

1. // **Set GA parameters**

   n->population size, G->Number of Generations

   μ-> Mutation Rate, α->Crossover Rate

2. // **Initialize the population**

   P_k: population created randomly with specified population size

3. // **Evaluate the P_k**

   Calculate fitness (i) for each i ∈ P_k
   do
   {

   Selection:
   Tournament selection with specified k value is applied for selection.
   Crossover:
   first_parent= Individual with highest fitness score
   second_parent and third_parent= Individual with minimum distance from first parent.
   α x n =No. of triplets to be coupled for mating. Place the newer off springs in P_{k+1}
   Mutation:
   Select μ x n individuals of P_{k+1}; invert the bits of selected individual.

4. // **Evaluate P_{k+1}**

   Compute fitness (i) of each individual, I ∈ P_{k+1}

5. // **Increase the generation**

   k=k+1;

   }

   Return fittest individual at each generation.

   Continue steps 3 to 5 until the specified G is reached.

Fig. 3. Algorithm for three parent genetic algorithm

H. Mutation

By mutation the individuals in the population are randomly altered. They are applied to the variables of the individuals with a low probability.

1**1**001001->1**0**001001

VI. AN IMPROVED FUZZY GUIDED GENETIC ALGORITHM

An Improved fuzzy guided genetic algorithm is similar to the FGA except that this has the ability to meet the dynamic consumer requests and services. The construction for the fuzzy engine is shown in the Fig. 4. A fuzzy engine with two controllers, crossover and mutation controller is built. The fuzzy engine receives two inputsΔf and d where Δf is the average fitness difference and d is the bit difference between the all pairs of individuals in the population

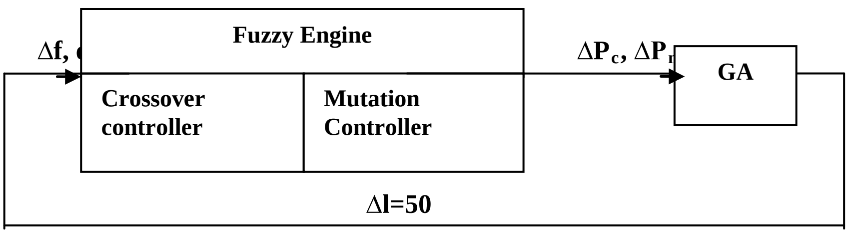

Fig. 4. Fuzzy Engine

The bit difference is calculated as:

$$d = \frac{1}{\frac{N(N-1)}{2}} \sum_{I=1}^{N-1} \sum_{J=I+1}^{N} \sum_{K=1}^{C} \frac{\Delta(pop_{ik}, pop_{jk})}{C}$$

Where N is the population size and C is the Genome Length. Here pop_{ik} and pop_{jk} are the k^{th} bit value of i^{th} individual and k^{th} bit value of j^{th} individual respectively. If pop_{ik}, pop_{jk} have same value then $\Delta(pop_{ik}, pop_{jk})$=0 else $\Delta(pop_{ik}, pop_{jk})$ returns 1. The fuzzy engine in turn produces two outputs crossover rate (ΔP_c) and mutation rate (ΔP_c) and sends to GA. GA uses the updated crossover and mutation rates continues its execution for 50 generations. This process continues until the exact solution is reached. Fuzzy engine has three parts Fuzzification, Fuzzy rules, Defuzzification

I. Fuzzification

The fuzzification module has two inputs and two outputs namely Δf and d, ΔP_c and ΔP_m. The universe of discourse is determined through experiments. Fig.5 shows the Δf. Fig. 6 shows the d, Fig. 7 shows the ΔP_c and Fig. 8 shows the ΔP_m

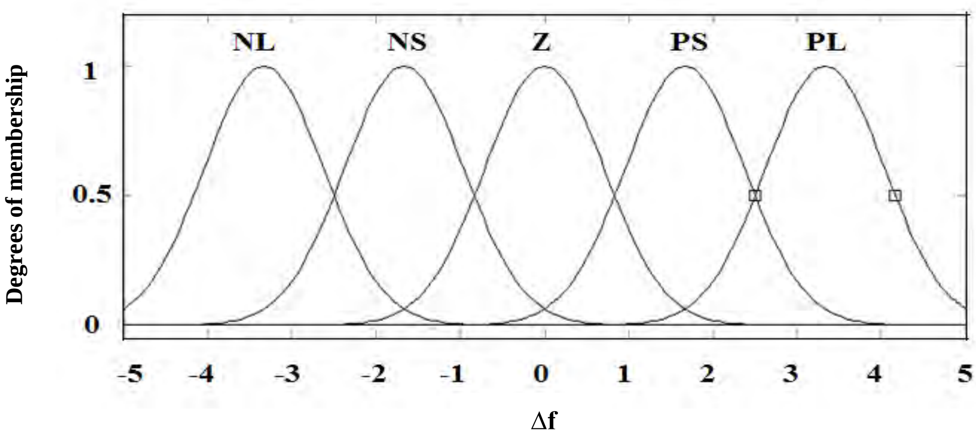

Fig. 5. Gaussian Fuzzy membership function of Δf

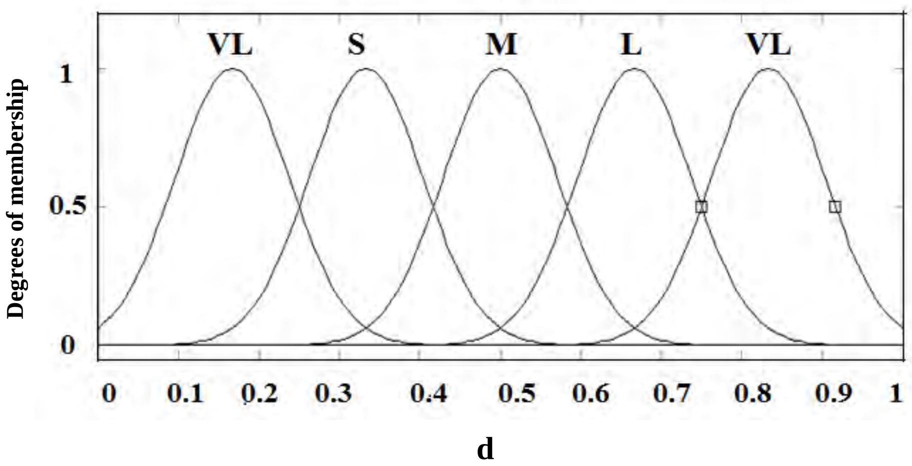

Fig. 6. Gaussian Fuzzy membership function of d

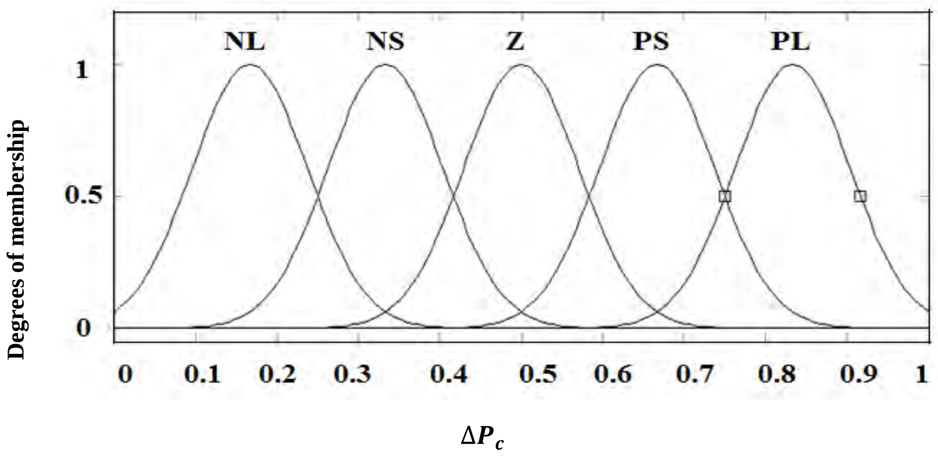

Fig. 7. Gaussian Fuzzy membership function of ΔP_c

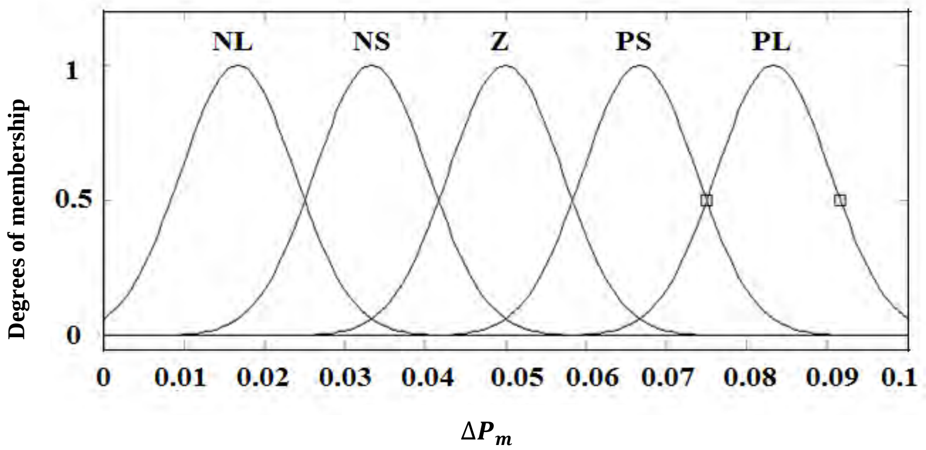

Fig. 8. Gaussian Fuzzy membership function of ΔP_m

J. *Fuzzy rules*

Fuzzy rules are conditional statements in which both the antecedents and consequences are fuzzy rather than crisp. For the most common interpretation, consider the fuzzy rule

"If x is A then, y is B"

AxB is a collection of fuzzy rules and x and y are linguistic variables (See Table II and Table III).

TABLE II Linguistic terms of Δf, ,ΔP_c, ΔP_m and d

Δf	Meaning	d	Meaning
NL	Negative Large	VL	Very Large
NS	Negative Small	L	Large
Z	Zero	M	Medium
PS	Positive Small	S	Small
PL	Positive Large	VS	Very Small

TABLE III Decision table for $\Delta P_c$and ΔP_m

Δf d	NL	NS	Z	PS	PL
VS	PL	PL	PS	PS	Z
S	PL	PS	PS	Z	NS
M	PS	PS	Z	NS	NS
L	PS	Z	NS	NS	NL
VL	Z	NS	NS	NL	NL

K. Defuzzification

Defuzzification is the process of producing the quantifiable result in fuzzy logic with corresponding membership functions and fuzzy sets. These are used in control systems. These will have a rule that is to be transformed in to results according to the inputs. This produces crisper outputs.

1. // **Set GA parameters**

   n->population size, K->Number of Generations, μ-> Mutation Rate, α->Crossover Rate

   Initial parameters were set according to table III

2. // **Generate Fitness Attributes**

   Quality Attributes:

   Quality attributes are generated randomly, 0<Q<1

   $Q_{ij}^{Rel}, Q_{ij}^{Ava}, Q_{ij}^{Rep}, Q_{ij}^{Cost}, Q_{ij}^{Respt}$

   Weight Attributes:

   Weights given for each service is generated such that

   $\sum_{j=1}^{l_i} W_{ij} = 1$

3. // **Set IMPROVED FGA Parameters**

   Set range for two inputs ∆f and d

   ∆f-> Average fitness difference, d-> bit difference

   Set range for two outputs ΔP_c and ΔP_m

   ΔP_c->Crossover rate, ΔP_m->Mutation rate

   Generate rules for fuzzy engine as given in table II

4. **Perform two parent genetic algorithm as given in figure.2**
5. **Calculate ∆f and d**
6. **Input the values of ∆f and d in to fuzzy Engine**
7. **Fuzzy engine produces the two outputs ΔP_c and ΔP_mUpdate the crossover and mutation rate**

Repeat the steps 4 to 8 until the maximum number of generations are reached

Fig. 9. Algorithm for Improved Fuzzy guided Genetic Algorithm

VII. EXPERIMENTAL RESULTS

Firstly, Genetic algorithm is implemented with two parents and three parents and the results were compared as shown in the Fig. 10. The performance of three parent GA increases as the number of generations increases. Whereas the two parent GA decreases its performance as generation increases and both starts to converge after 400generations.

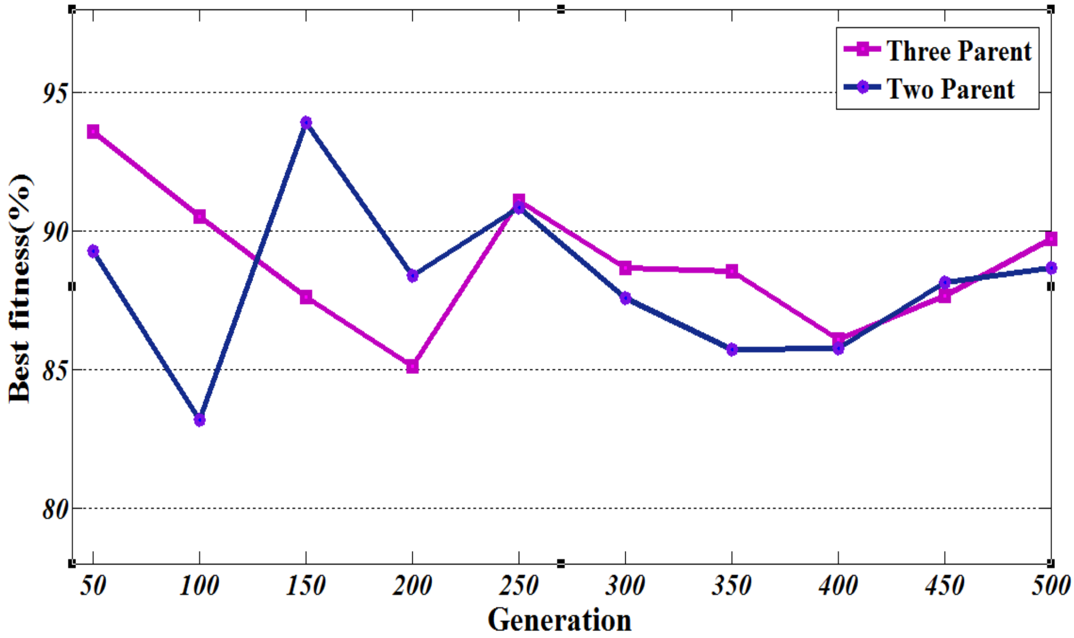

Fig. 10. Comparison of two parent and three parent using uniform crossover

Next, FGA and Improved FGA discussed in the previous sections were implemented and the comparison is made between them. The experiments were conducted on a core of Intel® Pentium® 4 CPU 2.80 GHZ with 1GB RAM memory using java JDK Runtime Environment on Windows 7. All measurements were taken for 100 consumer pairs. The parameters for the optimization problem are tabulated in Table IV.

TABLE IV. Parameters for GA, FGA and Improved FGA

Parameter	**Value/Type**
Population size	100
Maximum Generation	500
Selection method	Tournament with k=10
Initial crossover rate	0.6
Initial mutation rate	0.05

A. Universe of Discourse for Fuzzy Engine

The universe of discourse is set as in [4].The original range for fuzzy inputs is Δf is set to [-0.005 0.005] and the range of d is [0, 1] since the bit difference of all pair of individuals in the population cannot be varied. The two outputs ΔP_c and ΔP_m are set to [0.1 1] and [0.01 0.1]. The effect of fitness values can be found by narrowing the universe of discourse. After that the same is applied on the ΔP_c and ΔP_m To narrow the range of Δf, ΔP_c, ΔP_m multiply the numbers with the values ranging from 0.7 to 1.4. Fig. 11 shows the variation of crossover rate.

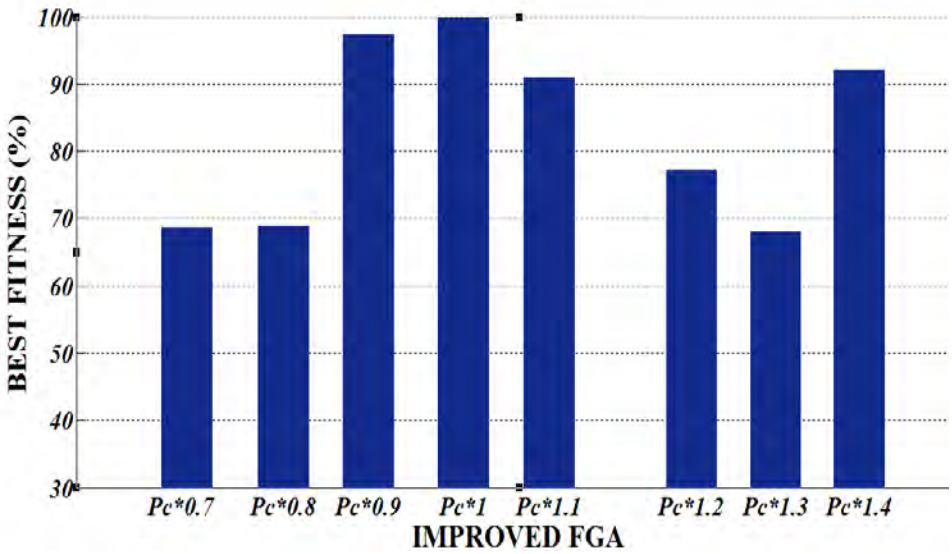

Fig. 11. Variation of ΔP_c

The measurements are done using the single point crossover with generations =500.The best fitness is obtained for Improved FGA at Δf*1.0 Fig. 12 shows the variation of ΔP_m. The best fitness values are obtained for ΔP_m*1.0

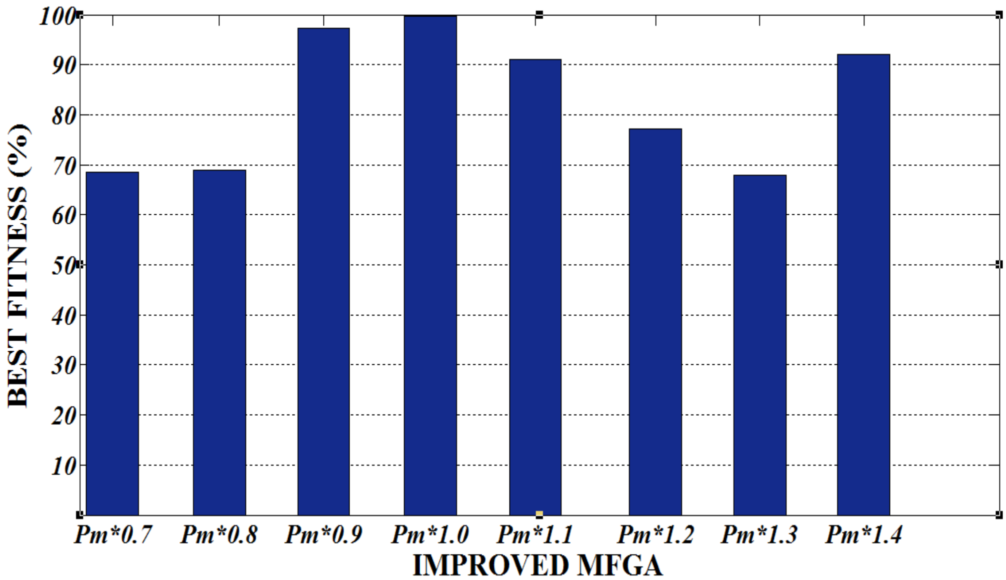

Fig. 12. Variation of ΔP_m

B. Overall Performance

Fig. 13 and Fig. 14 shows the best fitness and execution time achieved for the single point and two point crossover for 500 generations with 100 consumers in GA. Best fitness is obtained in two-point crossover, but time taken is more. Here, we have focused on time complexity and hence the single point crossover is chosen for Improved FGA. Fig. 15 shows the comparison of the best fitness values obtained using GA, FGA and Improved FGA. It is shown that Improved FGA achieves higher fitness than GA and FGA. The fitness can also be increased by applying the various crossover and mutation techniques.

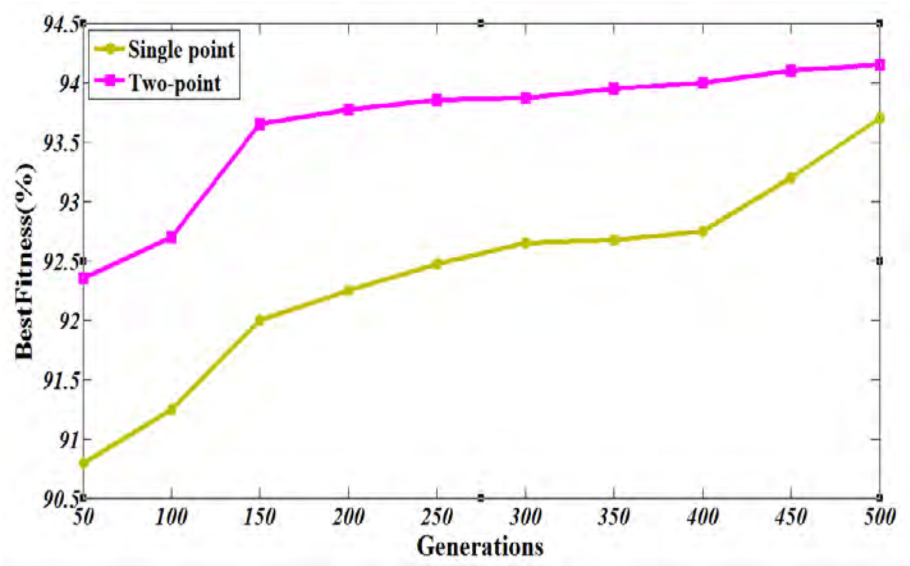

Fig. 13. Comparison of Best Fitness between single point and two point crossover

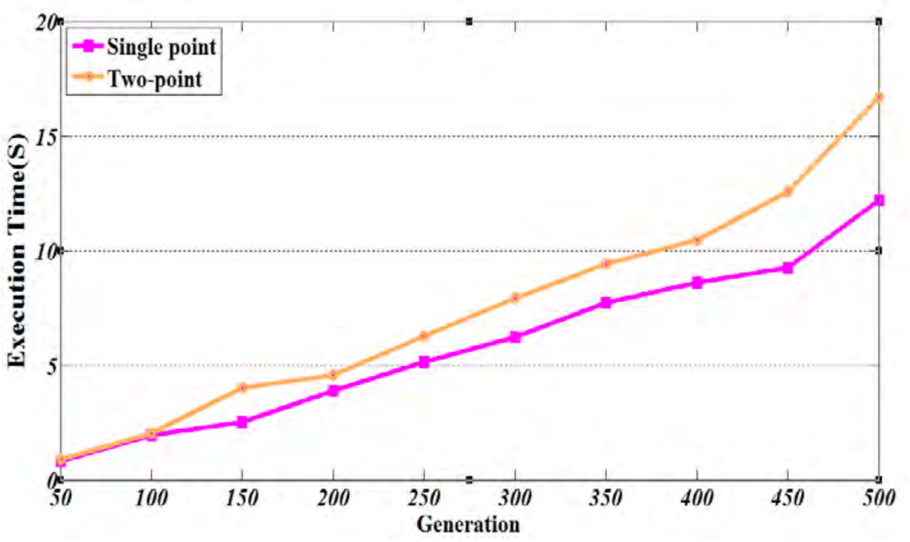

Fig. 14. Comparison of Execution Time between 2-Point and single point crossover

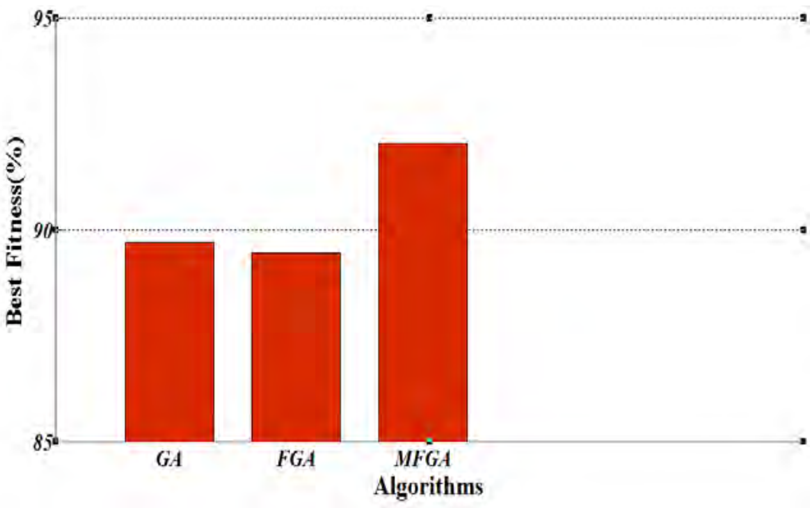

Fig. 15. Comparison between GA, FGA and Improved FGA

C. Variation of Crossover and Mutation rates

GA has constant crossover and mutation rates whereas Improved FGA uses two controllers namely crossover and mutation controllers which control the crossover and mutation rates. Fig. 16 shows that the crossover rate increases as the generation increases and starts to converge after 300 generations.

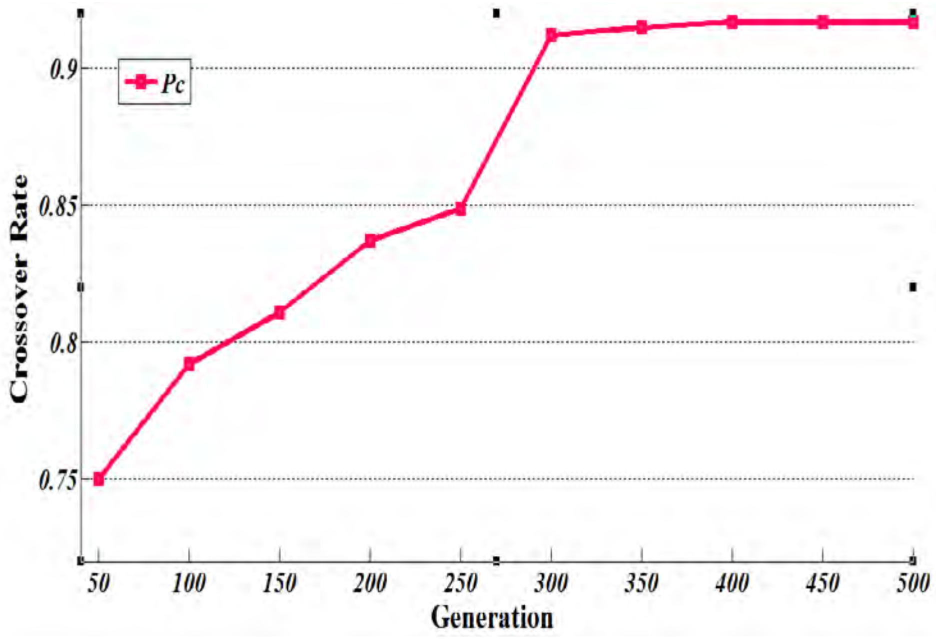

Fig. 16. Variation of crossover rate using single-point crossover

Similarly the mutation rate increases as the generation increases and starts to converge after 300 generations as in Fig. 17.

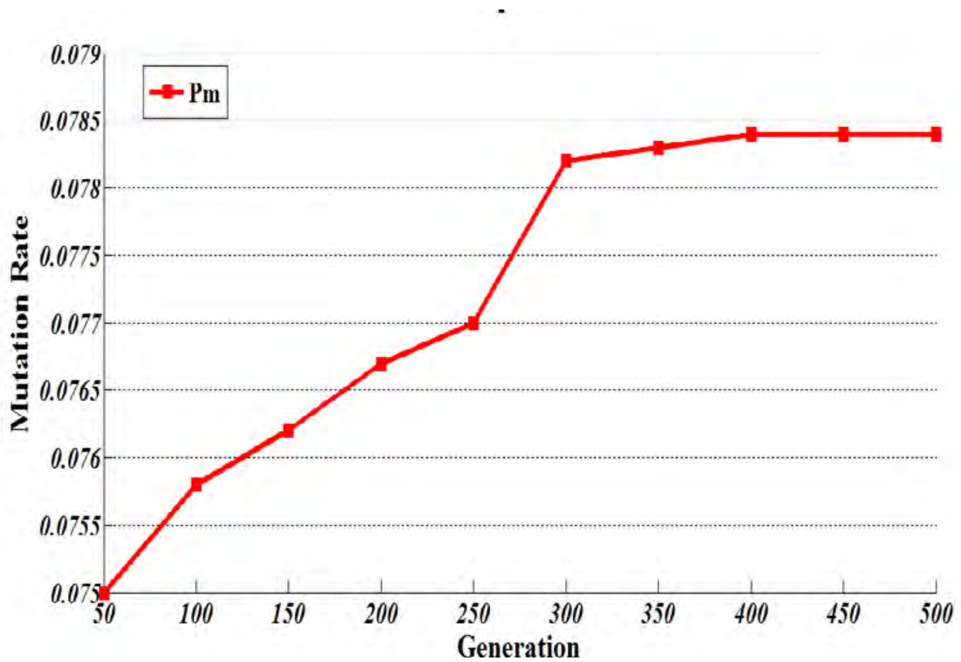

Fig. 17. Variation of mutation rate using single-point crossover

VIII. CONCLUSION

In this paper, an improved fuzzy guided genetic algorithm is proposed for the optimization of web service selection problem. Traditional Genetic Algorithms uses fixed crossover and mutation rates. FGA uses the fuzzy engine to control the crossover and mutation rates in a static environment using triangular membership functions. Here, we have proposed Improved FGA technique which can address the selection problem in a dynamic environment and the usage of Gaussian membership function to obtain finer granularity in the results.

First the GA was implemented using three parent and two parent techniques using uniform crossover. It is shown that the best fitness values are obtained in a lesser generations as the number of parent's increases. Secondly, the Improved FGA was implemented using single point crossover with the universe of discourse values. The overall performances of the three algorithms (GA, FGA, and Improved FGA) were compared in terms of the best fitness value. It is inferred from the experiments that Improved FGA scores higher than the FGA and GA. Since, the fuzzy component adjusts the crossover and mutation rates accordingly it takes more execution time. The advantage of Improved FGA is that the dynamic nature of the selection problem is addressed and better results are obtained. This is applicable for the optimization problems where services come online and go offline.

Future work will include the implementation of different crossover methods and applying the different algorithms to reduce the time complexity. Multiparent reproduction techniques can also be used to obtain better results at the lesser number of generations.

REFERENCES

[1] Zadeh, L.A, “Fuzzy Logic, Neural Networks, and Soft Computing”, Communication of the ACM, vol. 34, no. 3, pp. 77-84, Mar. 1994.
[2] Heather Kreger, “Web service conceptual architecture”, IBM Software Group, May 2001.
[3] V. X. Tran, H. Tsuji, “QoS based ranking for web services: Fuzzy approaches”, Proceedings of the 2008 4th International Conference on Next Generation Web Services Practices, IEEE Computer Society, pp. 77–82, 2008.
[4] M. Chen, S.A.Ludwig, ”Fuzzy-guided Genetic Algorithm applied to the Web Service Selection Problem”, Proceedings of the 2012 IEEE International Conference on Fuzzy Systems, pp.1-8, Jun. 2012.
[5] Qiqing F, Xiaoming P, Qinghua L, Yahui H, “A global QoS optimizing web services selection algorithm based on MOACA for dynamic web service composition”, vol. 1,pp. 37–42, IFITA 2009.
[6] Pavel Kr¨omer, Jan Platoˇs, V´aclav Sn´aˇsel, and Ajith Abraham, “Fuzzy classification by evolutionary algorithms”, in IEEE International Conference on Systems, Man, and Cybernetics, IEEE System, Man, and Cybernetics Society, pp. 313 – 318, 2011.
[7] A. Klein, Adrian, F. Ishikawa, S. Honiden, "Efficient Heuristic Approach with Improved Time Complexity for QoS-aware Service Composition," in IEEE ICWS 2011: The 9th International Conference on Web Services, 2011.
[8] Sora, I., G. Lazˇar, S. Lung, “Mapping a fuzzy logic approach for qos-aware service selection on current web service standards” in IEEE International Joint Conference on Computational Cybernetics and Technical Informatics (ICCC-CONTI), pp. 553 – 558, 2010.
[9] Z. Zheng, H. Ma , M.R. Lyu, I. King "QoS-Aware Web Service Recommendation by Collaborative Filtering", IEEE Transactions on Services. Computing, vol. 4, no. 2, pp. 140 -152, 2011.
[10] Stephen S. Yau and Yin Yin, “QoS-based Service ranking and Selection for Service-based Systems”, IEEE International Conference on Services Computing (SCC), 2011.
[11] Maolin Tang, Lifeng Ai, “A hybrid genetic algorithm for the optimal constrained web service selection problem in web service composition, Proceedings of IEEE Congress on Evolutionary Computation(CEC), pp. 1-8, 2010.
[12] W. Zhang, C.K. Chang, T. Feng, Hsin-yi. Jiang, “QoS-Based Dynamic Web Service Composition with Ant Colony Optimization”, Proceedings of IEEE 34th Annual Conference on. Computer Software and Applications, pp. 493-502, 2010.
[13] F. Rosenberg, P. Celikovic, A. Michlmayr, P. Leitner and S. Dustdar, "An End-to-End Approach for QoS-Aware Service Composition", Proceedings of IEEE International Conference on Enterprise Distributed Object Computing, pp. 151-160, 2009.
[14] Molood Makhlughian, Seyyed Mohsen Hashemi, Yousef Rastegari, Emad Pejman, “Web Service Selection based on Ranking of QoS using Associative Classification", Published International Journal of Web Service Computing (IJWSC), Vol.3, No.1, Jun. 2012.
[15] Ekkachai Naenudorn, Suphakit Niwattanakul,”Web Services Selection Algorithm Based on the Quality of Service” Lecture Notes on Software Engineering, Vol. 2, No. 1, Feb. 2014.
[16] N. Kokash, ”A Service Selection Model to Improve Composition Reliability”, Proceedings of the International Workshop on Artificial Intelligence for Service Composition, 2006.
[17] Ibrahim A. Hameed, “Using Gaussian membership functions for improving the reliability and robustness of students”, Expert Systems and Application,Elsevier, vol. 38, iss. 16, pp. 7135-7142, 2011.
[18] F. Herrera, M. Lozano, J.L. Verdegay, “Tuning fuzzy logic controllers by genetic algorithms”, International Journal of Approximate Reasoning vol. 12, iss 3–4, May 1995.
[19] P. Thrift, "Fuzzy Logic Synthesis with Genetic Algorithms", Proceedings of Fourth International Conference on Genetic Algorithms, pp. 509 -513, 1991.
[20] Noraini Mohd Razali, John Geraghty, “Genetic Algorithm Performance with Different Selection Strategies in Solving TSP flowcchart”, Proceedings of the World Congress on Engineering, vol. II, 2011.