Fuzzy-guided Genetic Algorithm applied to the Web Service Selection Problem

Min Chen
North Dakota State University
Fargo, ND, USA
min.chen@my.ndsu.edu

Simone A. Ludwig
North Dakota State University
Fargo, ND, USA
simone.ludwig@ndsu.edu

***Abstract*—The benefits of Quality of Service (QoS) aware service selection is undisputed. The selection process based on QoS allows the user to specify their requirements not only based on functional attributes but also on non-functional attributes. The automation of this selection process can be done via optimization. Several different exact but also approximate algorithms have been proposed in the past. Genetic algorithm is one such method that can find approximate solutions during the service selection task. In this paper, we propose an improved version of the standard genetic algorithm approach by making use of fuzzy logic during the stochastic genetic search process. The fuzzy component dynamically adjusts the crossover and mutation rates of the evolution for every ten consecutive generations. Results show that the fuzzy-guided Genetic algorithm approach improves the solution quality.**

I. Introduction

Services computing is a discipline combining science and technology. In particular, it is a bridge for the gap between Business Services and IT Services. The core technology suite consists of Web services and service-oriented architecture (SOA), cloud computing, business consulting methodology and utilities, business process modeling, transformation and integration [1]. Services computing covers the whole life-cycle of services innovation research. It includes services modeling, creation, annotation, deployment, discovery, composition, delivery, monitoring, optimization, as well as management. The major goal of services computing is to provide the infrastructure necessary to perform the business processes more efficiently and effectively [1].

Services are components that support the composition of distributed applications, and are offered by service providers or organizations. Since services are offered by different businesses they provide a distributed computing infrastructure for both intra- and cross-enterprise application integration and collaboration. Descriptions of services are used to advertise the capabilities, interface, behavior, and quality of the service. Service descriptions are necessary for discovery, selection, binding, and composition of services. The service capability description presents the conceptual requirements of the service. The service interface description publishes the service's input, output, and message types. The service behavior description describes the "expected" behavior of a service during its execution. Finally, the Quality of Service (QoS) description publishes important non-functional service quality attributes, such as cost, reliability, and availability to name a few. Service clients, and service workflow aggregators utilize service descriptions to achieve their business tasks [2].

WSDL (Web service description language)[1] provides a standard model for specifying service functionality by the separation of the abstract representations of service input and output messages from the concrete descriptions of each end point's bindings. However, currently there is no standard description method that includes all aspects of services' non-functional characteristics. Therefore, this lack of a standardized mechanism based on a consistent view of non-functional service characteristics is a current research topic. Different approaches addressing this deficiency have been proposed.

QoS attributes generally considered are cost, availability, response time, and reliability. Cost is defined as the fee to be paid to the service provider by the service requester for executing a particular service. The cost is always associated with the value of the service functionality, i.e. the more complex the function it provides, the higher the service cost. Availability of a web service is the probability that the service operation is accessible. This is defined by the proportion of the service's uptime and downtime. Response time is the expected delay between the time instant when a request is sent and the time when the result is obtained. Reliability is a measure of the service invocation trustworthiness. It is defined as the ratio between the numbers of service invocations that comply with the negotiated QoS over the total number of service invocations.

This paper proposes a fuzzy-guided Genetic algorithm (FGA) approach that makes use of fuzzy logic while guiding the stochastic genetic search. The fuzzy component dynamically adjusts the crossover and mutation rates of the evolution for every ten consecutive generations, which leads to improved solution quality applied to the web service selection problem. Therefore, better matches between service requesters and service providers can be found.

The remainder of the paper is structured as follows. In the next section related work is outlined. Section III describes the proposed fuzzy-guided genetic algorithm approach. In Section IV the experiments and results are given, followed by the conclusions presented in Section V.

[1]http://www.w3.org/TR/wsdl

II. Related Work

Related work in the area of web service selection include approaches using trust and reputation mechanisms, computational intelligence methods, QoS-based selection techniques and fuzzy approaches.

An evidence-based scheme for Web service selection is proposed to estimate the degree of consumer trust in a particular service by considering consumers' direct experience and indirect recommendation of the service in [3]. The proposed model enables deception detection and excludes the fraudulent evidence of malicious evaluators from the selection process.

In [4], a reputation measure approach of Web service is introduced to enhance the reputation measure accuracy. Three phases, feedback checking, feedback adjustment and malicious feedback detection, have been employed to improve the service selection process in order to obtain reliable services.

A new approach for designing and developing a QoS ontology and its QoS-based ranking algorithm for Web services are introduced in [5]. The QoS ontology is used to support QoS descriptions and facilitates service participants to express their QoS offers and demands. Analytic Hierarchy Process (AHP) is adopted in the QoS-based ranking algorithm to rank the Web services with similar functionality in order to select the best services for a request.

A Web service selection framework by introducing a QoS-based cost function concept is proposed for composite distributed Web services in [6]. The proposed approach composes a few Web services chosen from a given service community by their cost function to find the optimal composite service.

A novel ANN-based (Artificial Neural Network) service selection algorithm (ANNSS) is proposed and applied to a ubiquitous Web service environment in [7]. The users can acquire effective guidance, and choose the most appropriate service evaluated by an ANN-based evaluation standard. A three-term method that improves the traditional back propagation algorithm is used to satisfy the requirements of time issue in real-time systems.

Linear-programming, an exact optimization method, has been applied to the workflow service selection problem in [8], [9]. Another implementation has used complex workflow patterns to address the service selection problem [10]. Linear programming is used to solve the optimization problem using an aggregation function for different QoS attributes.

However, one problem the exact optimization methods have is the time-complexity. Therefore, approximate techniques can be used to improve the execution time of the selection process, however, only sub-optimal assignments of the services based on the service requests are likely to be found. One such approximate method is genetic algorithm (GA).

A stochastic demand multi-product supplier selection model with service level and budget constraints using a GA is described in [11]. The relationship between the expected profit and the number of trials, and between the expected profit and the combination of mutation and crossover rates, are studied to identify better parameter values to run the GA.

A new cooperative evolution algorithm, which consists of stochastic particle swarm optimization (SPSO) and simulated annealing (SA), is introduced to solve the service selection problem in [12]. The approach resolves the service selection with multi-objective and QoS global optimization. The proposed cooperative evolution algorithm owns better global convergence ability with faster convergence speed, in addition, the multi-objective SPSO is feasible and efficient.

In [13], a system for supporting the user in the discovery of semantic Web services is used to model an ad-hoc service request by selecting conceptual terms rather than using strict syntax formats. The selection exploits the fuzzy formal concept analysis to request the system to return a list of semantic Web services that match the user query.

In [14], a service selection method based on the technique for Order Preference by Similarity to an ideal Solution (TOPSIS) with fuzzy opinions is used to evaluate the weights of various criteria and the rating of each alternative Web service. The approach uses triangular fuzzy membership functions to represent the weights of criteria and the ratings of Web services.

In [15], a fuzzy-based UDDI (Universal Description, Discovery and Integration) with QoS support is proposed to consider the non-functional quality of QoS information for personalized Web service selection. This approach considers the objective factors described by service providers, and subjective information with trustability evaluations from users by adapting genetic algorithms (GA) to learn the user preferences, and to apply fuzzy logic to make decisions. The users can determine the most suitable Web service with a fuzzy query interface to provide subjective and objective factors.

A decision model under consumer's vague perception of intuitionistic fuzzy set for QoS-aware Web services selection is proposed in [16]. The selection method is modeled as a fuzzy multi-criteria decision-making problem by considering the non-functional QoS properties that heavily rely on the perceptions of service providers and consumers.

The classical approach to the assignment problem is given by the Munkres algorithm [17], which was originally proposed in 1955, also known as the Hungarian algorithm because the algorithm was largely based on the earlier works of two Hungarian mathematicians named Denes Konig and Jano Egervary. Later, it was reviewed by [18] in 1957 and he observed the algorithm is (strongly) polynomial. Since then, the algorithm was given the name Kuhn-Munkres algorithm or Munkres assignment algorithm. The algorithm has a time complexity of $O(n^3)$.

In this paper, we are following the idea of using an approximate technique, namely GA, to be applied to the optimization process. Furthermore, the matches of service requesters with service providers are based on QoS parameters as described in the following section. We also introduce the concept of an adaptive crossover and mutation rate using fuzzy logic. We will show that fuzzy logic is suitable for controlling the crossover and mutation rates throughout the GA run, and that it improves the solution quality.

III. Fuzzy-guided Genetic Algorithm

A. Quality of Service Metric and Fitness Function

There are many measures available for different QoS criteria, however, we consider the following five generic quality criteria for services (QoS parameters): reliability, availability, reputation, execution duration, and execution price.

The reliability of a service q_1 is the fraction of requests correctly responded to within a maximum expected time frame. The availability of a service q_2 is the fraction of time that the service is accessible. The reputation of a service q_3 is a measure of its trustworthiness and depends on the end user's experience of using a service. The execution duration q_4 denotes the expected delay in seconds from the moment a request is made until the moment when the results are returned. The execution price q_5 represents the amount of money a user has to pay for executing a service.

Our goal is to maximize the overall QoS of the service requests simultaneously. The overall fitness function for the optimization of N simultaneous service requests is the following:

$$f_{obj} = max\sum_{i=1}^{N}(\sum_{j=1}^{3}\frac{q_{ij}-q_j^{min}}{q_j^{max}-q_j^{min}}+\sum_{i=4}^{5}\frac{q_j^{max}-q_{ij}}{q_j^{max}-q_j^{min}}) \quad (1)$$

whereby we define q_{ij} to be the value for service i and the j^{th} QoS parameter, and we define q_j^{max} to be the maximum score any of the considered services achieves for the j^{th} QoS parameter as defined above, and similarly, q_j^{min} to be the minimum score any of the considered services achieves for the j^{th} QoS parameter. Note that the individual QoS parameters are treated differently depending on whether its value is to be minimized or maximized. In general, the user wants to maximize reliability, availability and reputation, whereas cost and execution time should be minimized.

B. Genetic Algorithm (GA)

GA [19] is a global optimization strategy that models natural evolution. A GA works on a population of chromosomes that is created as a set of possible solutions to the optimization problem. Within a generation of a population, the chromosomes are randomly altered by processes of selection, crossover and mutation in order to create new chromosomes that have better evaluation scores. The next generation population of chromosomes is randomly selected from the current generation with selection probability based on the evaluation score of each chromosome.

For the web service selection problem at hand, an individual is modeled to correspond to one match. The match is implemented as a vector, which is also referred to as a chromosome. Dimensions in the vector correspond to providers, and values correspond to consumers. Therefore, if the vector has value 3 at its 5th position (dimension), consumer 3 is matched with provider 5. Every number representing a consumer can only be at one position in the vector, otherwise, the vector represents a non-valid match.

At the beginning, the first population is randomly initialized. After that, the fitness of the individuals is evaluated using the fitness function (Equation 1). After the fitness is evaluated, individuals have to be selected for paring. The selection method used is tournament selection. Always two individuals are paired, resulting in an offspring of two new individuals. In the pairing phase, a random crossover mask is used, i.e. the positions (dimensions) for which crossover occurs are selected randomly. If crossover occurs at certain positions (dimensions), individuals that are mated exchange their values at that position and the resulting individuals are used as offspring. The crossover has to make sure that the offspring present a valid match. Therefore, if two values are exchanged, other positions in the two match vectors are usually effected as well. The offspring faces mutation with a certain low probability. After mutation, the fitness of the offspring is calculated. Then, either all individuals from the last generation compete against the whole offspring, or the offspring only compete with its corresponding parents. In this implementation, all individuals from the old generation compete with all individuals in the new generation. After the new generation is selected, the GA will start over, and continue with parent selection and crossover.

C. Fuzzy-guided Genetic Algorithm (FGA)

Fuzzy logic, which began with the fuzzy set theory by Lotfi Zadeh [20] in 1965, has been applied to many fields and is a useful tool for decision-making problems involving uncertainty and vagueness. We are making use of fuzzy logic in order to guide the crossover and mutation rates of our GA implementation.

In order to adjust the crossover rate and mutation rate in the GA based on the feature of changing environment, we use a fuzzy engine including two controllers: crossover controller and mutation controller. The framework of the proposed fuzzy-guided genetic algorithm (FGA) is shown in Figure 1. The two inputs of the fuzzy engine are Δf and d, and two outputs are ΔP_c and ΔP_m. Let Δf be the average fitness difference between the current generation and the previous generation. d captures the average of the bit difference of all pairs of individuals of the population, and can be calculated as:

$$d = \frac{1}{\frac{N(N-1)}{2}}\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\sum_{k=1}^{C}\frac{\delta(b_{ik},b_{jk})}{C} \quad (2)$$

where N is the population size, C is the length of the individual, b_{ik} and b_{jk} are the kth bit value of the ith individual and the kth bit value of jth individual respectively. If b_{ik} and b_{jk} have the same value, $\delta(b_{ik},b_{jk}) = 0$, otherwise $\delta(b_{ik},b_{jk}) = 1$.

The fuzzy engine determines the two outputs ΔP_c and ΔP_m calculated by the crossover controller and mutation controller respectively. ΔP_c is the updated value for the crossover rate and ΔP_m is the updated value for the mutation rate. Once ΔP_c and ΔP_m are calculated, the GA will update the crossover and mutation rates accordingly (similar to [21]). Afterwards, GA will generate another two inputs and send them to the fuzzy engine after 10 consecutive generations.

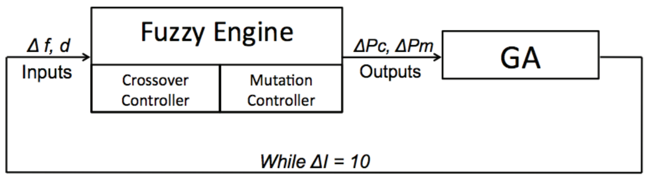

Fig. 1. Framework of fuzzy-guided genetic algorithm.

The fuzzy engine consists of three parts: fuzzification module, fuzzy rule base module, and defuzzification module.

(1) *Fuzzification module*: it converts the two inputs (Δf, d) and two outputs (ΔP_c, ΔP_m) into triangular membership functions. The meanings of the linguistic terms for the two inputs and two outputs are tabulated in Table I. The membership functions of the two inputs are shown in Figures 2 and 3, ΔP_c and ΔP_m are depicted in Figures 4 and 5, respectively. Please note that the universe of discourse for Δf, ΔP_c, ΔP_m was determined by experiments as outlined in the following section.

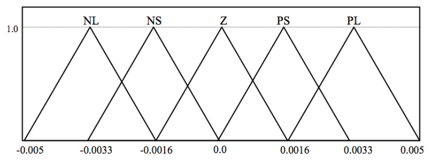

Fig. 2. Fuzzy membership functions of Δf.

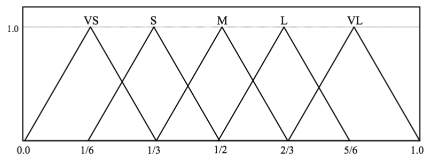

Fig. 3. Fuzzy membership functions of d.

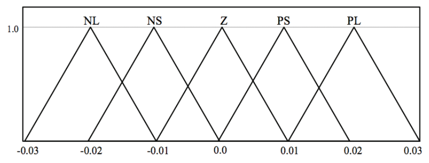

Fig. 4. Fuzzy membership functions of ΔP_c.

(2) *Fuzzy rule base*: a set of rules capture the relationships between the different values of the inputs and outputs. A rule is made up of two antecedent and one consequent linguistics statement connected by the fuzzy connections (AND, THEN). One example is the following rule (see Table II in bold):

IF Δf is PL, AND d is VL, THEN ΔP_c is NL

(3) *Defuzzification module*: the outputs of the fuzzy engine are crisp values that serve as the most representative value of

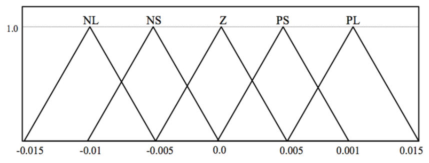

Fig. 5. Fuzzy membership functions of ΔP_m.

TABLE I
LINGUISTIC TERMS OF Δf,ΔP_c, ΔP_m AND d

Δf,ΔP_c and ΔP_m	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

the output. Several techniques have been proposed in previous works [22]. In the case of our FGA with continuous output variable, one suitable method is the moment defuzzification technique, which computes the first moment of area of a fuzzy set, thus the crisp output x' is calculated as:

$$x' = \frac{\sum_{i=1}^{m} x_i^{'} \cdot A_i}{\sum_{i=1}^{m} A_i} \tag{3}$$

where $x_i^{'}$ is the local center of gravity (and therefore our defuzzified mutation or crossover rate respectively), A_i is the local area, and m is the total number of outputs.

D. Encoding, Crossover and Mutation Methods

The encoding method in this implementation is permutation encoding. Thus, no duplicate integers are allowed in a chromosome. The performance of the GA/FGA depends on the crossover and mutation, and uniform, one-point and two-point crossover have been implemented. In uniform crossover, integers are randomly copied from the first or from the second parent. In one-point crossover, one crossover point is selected. The integer string from the beginning of the chromosome to the crossover point is copied from one parent, then the second parent is scanned and if the number is not yet in the offspring it is added. In two-point crossover, two crossover points are selected. The chromosome string from the beginning of chromosome to the first crossover point is copied from one parent, then the string from the first crossover point to the second crossover point is copied from the second parent. The remainder is copied from the first parent. Afterwards, the duplicated integers are replaced and the integers that are not yet in the offspring are randomly allocated. As for mutation, the swap mutation is implemented since no duplicated integers are allowed.

IV. Experiments and Results

A. Experimental Setup

The two approaches GA and FGA as introduced in the previous section were implemented. The Munkres algorithm was also implemented in order to use it for comparisons.

TABLE II
DECISION TABLE FOR ΔP_c AND ΔP_m

d \ Δf	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

The experiments were conducted on a core of an Intel(R) Core(TM)2 Duo CPU T6600 (2.20GHz) on an Asus laptop with 4096MB RAM memory using a Java version 1.6.0 JDK Runtime Environment on a Windows 7. All measurement points are average results taken from 25 runs with 300 consumer-provider pairs (i.e., 300 consumers and 300 providers). In addition, all fitness values shown are normalized with respect to the Munkres algorithm. The required parameters of the selection problem using GA and FGA are tabulated in Table III.

TABLE III
PARAMETERS OF GA AND FGA

Parameter	Value/Type
Population size	100
Maximum generations	1000
Selection method	Tournament with K = 10
Initial crossover rate	0.6
Initial mutation rate	0.05
Consumer-provider pairs	300

B. *Universe of Discourse for Fuzzy Sets of the Fuzzy Engine*

In order to set the universe of discourse of the fuzzy membership functions (Δf, ΔP_c, and ΔP_m) preliminary experiments were run to find the best fitness values that can be achieved. This was done by varying the ranges of the fuzzy input membership functions. For example, the original range of the fuzzy input membership function Δf (see Figure 2) is set to $[-0.005, 0.005]$ by choosing the upper-bound and lower-bound of the best fitness value between the previous and the next generation. The other fuzzy input d (see Figure 3) is fixed to $[0, 1]$, since it is the average of the bit difference of all pairs of chromosomes in the population and cannot be varied. The two outputs ΔP_c (see Figure 4) and ΔP_m (see Figure 5) are $[-0.03, 0.03]$ and $[-0.015, 0.015]$ respectively. In order to find the best range for Δf, ΔP_c and ΔP_m, first, we vary the range of Δf by multiplying the number with values ranging from 0.7 to 1.3 (narrowing or enlarging the range of the universe of discourse) in order to see the effect on the fitness values of the FGA. Afterwards, we perform the same on ΔP_c and ΔP_m respectively, and multiply the number with values ranging from 0.7 to 1.3. As we mentioned before, the best fitness value is the average result taken from 25 runs. The measurements are done using uniform crossover with generations = 500.

Figure 6 shows the variation of Δf. It can be seen that the best fitness values are obtained for FGA of $\Delta f * 1.0$.

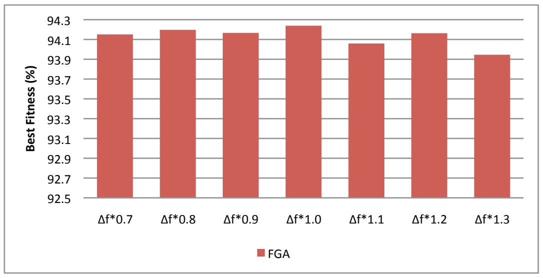

Fig. 6. Variation of Δf.

The variation of ΔP_c in Figure 7 shows that the best fitness values are obtained for $\Delta P_c * 1.0$, therefore we have chosen $\Delta P_c * 1.0$.

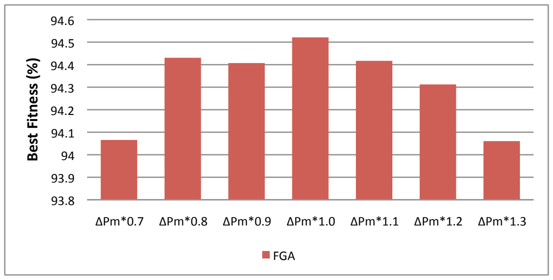

Fig. 7. Variation of ΔP_c.

In Figure 8, the variation of ΔP_m are shown. We can see that the best fitness values are obtained for $\Delta P_m * 1.0$ and therefore has been chosen.

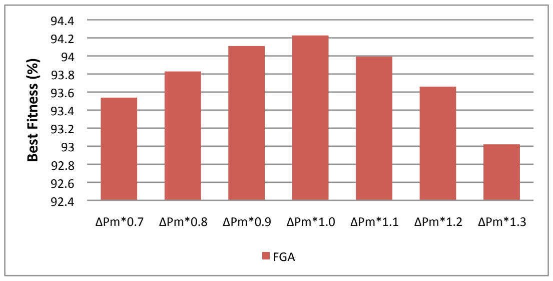

Fig. 8. Variation of ΔP_m.

C. *Overall Performance*

Figure 9 shows the execution time in seconds of the GA, FGA and Munkres algorithms using the uniform, one-point and two-point crossover methods with generations = 100. As mentioned before, we used 300 consumer-provider pairs. As expected, the Munkres algorithm (having a cubic time-complexity), which is an optimal algorithm, has the largest execution time, followed by the FGA algorithm, then the GA algorithm. In addition, Table IV shows the best fitness values obtained using the three algorithms. As can be seen from the fitness values, similar values for both methods (GA and FGA) are obtained for all crossover methods. For 100 generations the GA method scores slightly better than the FGA method, however, the optimal algorithm (Munkres) scores 100%.

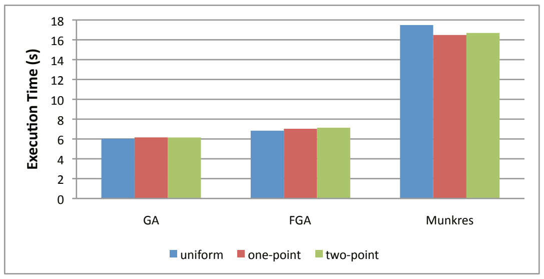

Fig. 9. Time Comparison of GA, FGA and Munkres for number of generations = 100.

TABLE IV
BEST FITNESS VALUES WITH GENERATIONS = 100.

Crossover Method \ Algorithm	GA (%)	FGA (%)	Munkres (%)
Uniform	87.006	86.948	100
One-point	87.531	86.921	100
Two-point	87.170	87.064	100

D. Comparisons of GA and FGA with Different Crossover Methods

Figure 10 plots the best fitness values against the number of generations using uniform crossover. Both GA and FGA steadily increase as the number of generations increases. At the beginning, no significant differences between GA and FGA can be observed. However, as the number of generations increases, the best fitness values obtained by FGA increase faster than the values obtained by GA. A linear increase in time for increasing numbers of iterations for the GA and FGA are shown in Figure 11. Due to the additional time needed for the fuzzy component, FGA shows the larger increase.

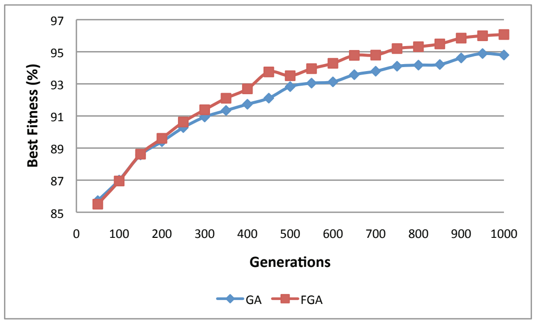

Fig. 10. Best fitness values obtained using uniform crossover.

Similarly, Figures 12 and 13 show the best fitness values and execution times obtained by GA and FGA using one-point crossover.

For two-point crossover the same trend of the best fitness values and the execution times are observed as shown in Figures 14 and 15, respectively. All the three crossover

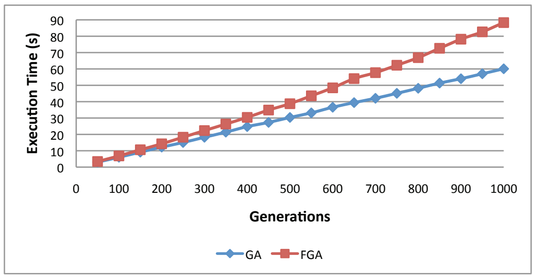

Fig. 11. Execution times obtained using uniform crossover.

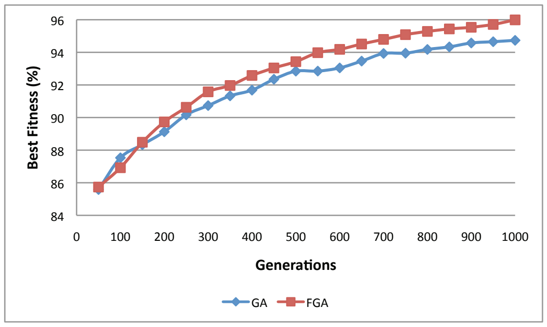

Fig. 12. Best fitness values obtained using one-point crossover.

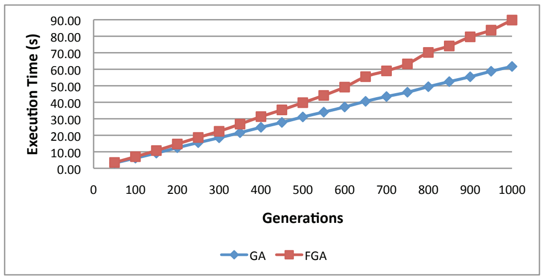

Fig. 13. Execution times obtained using one-point crossover.

methods, uniform, one-point and two-point, show that FGA is consistently better than GA, in particular, as the number of generations increases.

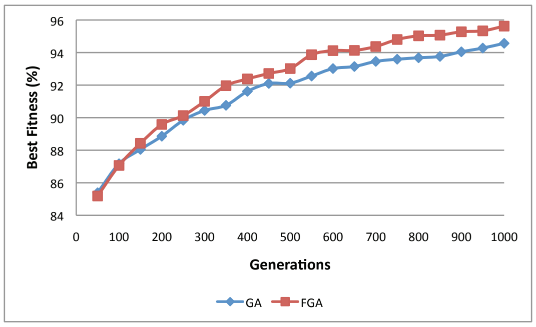

Fig. 14. Best fitness values obtained using two-point crossover.

E. Variation of Crossover Rate and Mutation Rate

It is known that standard GA has a constant crossover rate and mutation rate. Our FGA uses two controllers to update

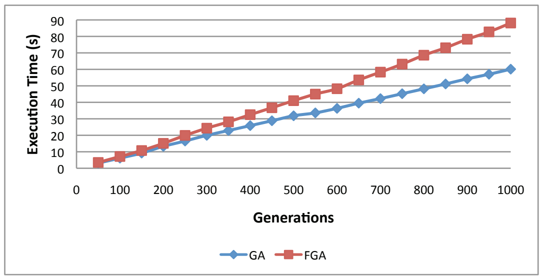

Fig. 15. Execution time obtained using two-point crossover.

the crossover rate and mutation rate. In Figure 16, it is shown that the crossover rate steadily increases as the number of generations (using uniform crossover) increases. After 400 generations, the crossover rate starts to converge to 1.

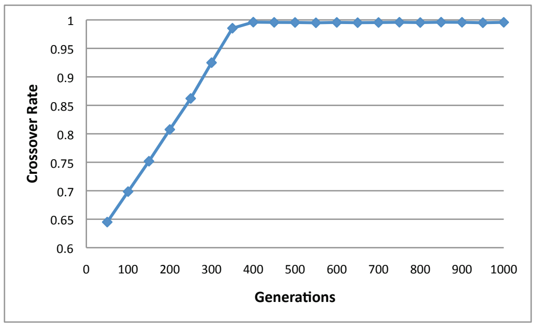

Fig. 16. Variation of crossover rate using uniform crossover.

Figure 17 plots the variation of mutation rate using uniform crossover. At the beginning, until about 100 generations, the mutation rate increases. After 100 generations the mutation rate fluctuates around a value of 0.077.

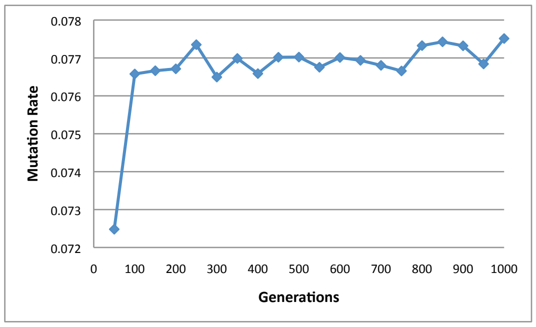

Fig. 17. Variation of mutation rate using uniform crossover.

V. Conclusion

In this paper, a fuzzy-guided genetic algorithm (FGA) has been applied to the Web service selection problem based on five QoS attributes. Given that standard GA uses fixed crossover and mutation rates, that are usually determined by preliminary runs, we have proposed FGA that uses fuzzy logic to adapt the crossover and mutation rates automatically.

FGA was implemented and different crossover methods have been investigated (uniform, one-point and two-point crossover). Several experiments were carried out. First, the universe of discourse for determining the fuzzy sets were looked at. Secondly, the overall performance of all algorithms were measured, both in terms of fitness value and execution time. Afterwards, different crossover methods were investigated and compared.

The experiments revealed that FGA score higher fitness values than standard GA. The best fitness values of GA using uniform, one-point and two-point crossover after 1000 generations are 94.57%, 94.74%, and 94.80%, while the best fitness values of FGA are 95.62%, 95.99%, and 96.07% respectively. However, this fitness improvement comes at the cost of execution time. Due to the fuzzy component for adjusting the crossover and mutation rates the execution time of FGA is necessarily longer. The advantage however FGA has is that the crossover and mutation rates do not need to be predefined before the optimization is run, and therefore it enables automated rate adjustments that can be applied to any kind of optimization problem.

Future work will include a finer granularity of the fuzzy sets, the use of gaussian fuzzy membership functions instead of the triangle membership functions, and the implementation of different mutation methods. In addition, the dynamic nature of the selection problem will also be addressed since services come online and go offline, and therefore this dynamicity needs to be taken into consideration.

Acknowledgment

This research is partly based on work supported by North Dakota EPSCoR and National Science Foundation Grant EPS-0814442.

References

[1] IEEE Computer Society, Overview, Available from: http://tab.computer.org/tcsc/scope.htm. Last retrieved March 2012.
[2] M.P. Papazoglou, D. Georgakopoulos, "Service-Oriented Computing", Communications of the ACM, 46(10):2565, 2003.
[3] P. Wang, K. Chao, C. Lo, R. Farmer, "An Evidence-Based Scheme For Web Service Selection", Information Technology and Management 12.2:161-172, 2011.
[4] S. Wang, Q. Sun, H. Zou, and F. Yang, "Reputation measure approach of web service for service selection", IET Software 5, no. 5: 466-473, 2011.
[5] V. Tran, H. Tsuji, R. Masuda, "A new QoS ontology and its QoS-based ranking algorithm for Web services", Simulation Modelling Practice and Theory, 17(8):1378-1398, 2009.
[6] A. E. Saddik, and D. Liu, "QoS based Selection Algorithms for Composite Distributed Web Services", Journal Of Interconnection Networks 10.4:421-434, 2009.
[7] H. Cai, X. Hu, Q. Lu, Q. Cao, "A Novel Intelligent Service Selection Algorithm And Application For Ubiquitous Web Services Environment", Expert Systems With Applications 36.2:2200-2212, 2009.
[8] T. Yu, Y. Zhang, and K.J. Lin, "Efficient Algorithms for Web Services Selection with end-to-end QoS Constraints", ACM Transactions on the Web, 1(1), 2007.
[9] D. Ardagna and B. Pernici, "Adaptive Service Composition in Flexible Processes", IEEE Transactions on Software Engineering, 33(6):369384, 2007.

[10] D. Schuller, J. Eckert, A. Miede, S. Schulte, R. Steinmetz, “QoS-Aware Service Composition for Complex Workflows”, Proceedings of the 2010 Fifth International Conference on Internet and Web Applications and Services, 2010.

[11] P. Yang, H. Wee, S. Pai, Y. Tseng, “Solving A Stochastic Demand Multi-Product Supplier Selection Model With Service Level And Budget Constraints Using Genetic Algorithm”, Expert Systems With Applications 38.12:14773-14777, 2011.

[12] X. Fan, X. Fang, and C. Jiang, “Research On Web Service Selection Based On Cooperative Evolution”, Expert Systems With Applications 38.8:9736-9743, 2011.

[13] G. Fenza, and S. Senatore, “Friendly Web Services Selection Exploiting Fuzzy Formal Concept Analysis”, Soft Computing - A Fusion Of Foundations, Methodologies and Applications 14.8:811-819, 2010.

[14] C. Lo, D. Chen, C. Tsai, K. Chao, “Service Selection Based on Fuzzy TOPSIS Method,” Advanced Information Networking and Applications Workshops, Proceedings of the 2010 IEEE 24th International Conference on Advanced Information Networking and Applications, pp. 367-372, 2010.

[15] H. Wang, C. Lee, and T. Ho, “Combining Subjective and Objective QoS Factors for Personalized Web Service Selection”, Expert Systems With Applications 32.2:571-584, 2007.

[16] P. Wang, “Qos-Aware Web Services Selection With Intuitionistic Fuzzy Set Under Consumers Vague Perception”, Expert Systems With Applications 36.3:4460-4466, 2009.

[17] H.W. Kuhn, “The Hungarian method for the assignment problem”, Naval Research Logistics, 52(1), 1955.

[18] J. Munkres, “Algorithms for the Assignment and Transportation Problems”, Journal of the Society for Industrial and Applied Mathematics, 5(1):32-39, 1957.

[19] J.H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, 1975.

[20] L.A. Zadeh, “Fuzzy set”, Inf. Contr., vol. 8, no. 3, pp. 338-353, 1965.

[21] H. Lau, T.M. Chan, W.T. Tsui, “Item-Location Assignment Using Fuzzy Logic Guided Genetic Algorithms”, IEEE Transactions on Evolutionary Computation, vol. 12, no. 6, pp. 765-780, Dec. 2008.

[22] D.P. Filev, R.R. Yager, “Fuzzy models induced by alternative defuzzification methods”, Proceedings of the Fifth IEEE International Conference on Fuzzy Systems, vol. 1, no. 1, pp. 457-462, 1996.