Towards Fuzzy QoS Driven Service Selection with User Requirements

Jiajun Xu, Lin Guo, Ruxia Zhang, Yin Zhang, Hualang Hu, Fei Wang, and Zhiyuan Pei
Key Laboratory of Cultivated Land Use, Ministry of Agriculture, P. R. China;
Chinese Academy of Agricultural Engineering, Beijing, P. R. China
xujiajun@agri.gov.cn

***Abstract*—Many QoS-aware service selection approaches assume that the QoS attributes are crisp values and the actual user requirements are not taken into consideration, when the service-oriented applications are constructed. As a result, users searching result may not be correct and good, because there are uncertainties in the data and the optimal solutions but not satisfying some requirements may not be acceptable to some users. In this paper, we propose to use Fuzzy Set Theory (FST) and fuzzy genetic algorithm (FGA) for QoS-based service selection. FST is applied to specify the triangular fuzzy-valued description of the QoS properties. A FGA is proposed to solve the QoS-aware service composition problem, which considers the actual QoS requirements from users in the selection process. Empirical comparisons with two algorithms on different scales of composite service indicate that FGA is highly competitive regards to searching capability.**

***Index Terms*—service selection, QoS, fuzzy set theory, genetic algorithm**

I. Introduction

SOA (Service-Oriented Architecture) has provided the unprecedented opportunity for enterprises to build distributed applications by orchestrating loosely coupled service [1]. Meeting QoS requirements is quintessential in ensuring a successful service selection. QoS criteria are measured on many different aspects, such as reliability, availability, latency, etc and have become a key role in selecting services from a large scale of functionally equivalent ones [2]. QoS-aware service selection known as a NP-hard problem remains one of the notable research challenges [3]. For service selection, ranking not only depends on the topic relevancy of the service on user's functional requirements, but also on its QoS attributes.

Various selection models are favorable in solving these problems. Many of these models usually consider service selection as an optimization problem and their goal is to find a service which can optimize multiple QoS criteria. However, in this process, the actual user requirements are not taken into consideration. Therefore all users get a same result no matter how different their QoS requirements might be. Also a service with a best overall ranking score but not satisfying some key requirements may not be acceptable to some users. And another problem with these models is that they normally treat QoS attributes as crisp values. QoS attributes are estimated from a large amount of historical data so that the confidence level of data lies in the interval [0, 1] and most conventional methods where the level of confidence of data is taken as 1 are inadequate to handle this problem.

In this paper we propose to use FST and GA for QoS-based service selection which incorporates the concepts of fuzzy technique and the evolutional process of standard genetic-based algorithm, and meanwhile, we take the actual QoS requirement into consideration in the ranking process. Experimental comparisons with standard GA (SGA) and standard nonlinear programming (NLP) demonstrate its superiority in exploring promising solutions. There are two major contributions of the paper. First, fuzziness in the data is usually not the focus of the research on service selection. Second, many selection models ignore the actual QoS requirements from users, whereas our system includes them in the selection algorithm. The remainder of this paper is structured as follows. Section 2 introduces the related works. Section 3 describes the FGA in detail. Section 4 demonstrates the proposed approach through comparisons of the SGA and NLP algorithms. Section 5 concludes the paper.

II. Related Work

QoS-based service selection is usually considered as an optimization problem, and thus various optimization models have been used to solve this problem. Fundamentally, the main objective of this problem is to produce a set of composite services of a certain quality. Heuristic algorithms form a category to find the best composite service solutions from a global perspective. Yu et al. [4] propose heuristic algorithms to find a near-to-optimal solution better than exact solutions. These methods lack scalability because the computation complexity of the search algorithms is exponential. A heuristic solution often employed to solve this selection problem is GA [5], because the genetic-based algorithm is normally more scalable [6]. The problem of these approaches is that they ignore the user requirements. The other problem within these models is that they usually treat QoS attributes as crisp values, without considering uncertainty in the data. However, some attributes such as reliability and latency are uncertain in the runtime. To enhance the ability to describe more valuable information, numerical values including the maximum and minimum can be described with an interval.

Research in Fuzzy QoS-driven service selection with the consideration of user requirements has produced promising

Special Fund of Major Information Platform Construction and Maintenance of the Ministry of Agriculture of China(2130104)

978-1-5386-1978-0/17/$31.00 ©2017 IEEE

approaches. More and more scholars are suggesting that the QoS properties should be expressed as uncertainty and vagueness to enhance the practical ability of representing more valuable data information. Mikhailov and Tsvetinov [7] have proposed a fuzzy analytic hierarchy process which uses fuzzy pairwise comparison judgments rather than exact numerical values of the comparison ratios and transforms the initial fuzzy prioritization problem into a non-linear program, to identify suitable service offers. Bacciu and Botta et al. [8] have proposed a general framework for handling SLA negotiation in which agreements rely on the fuzzy approach where required and offered quality levels are described by fuzzy sets. Bacciu and Buscemi et al. [9] provides a quality model based on trapezoidal fuzzy set and a matchmaking procedure based on a fuzzy-valued similarity measure of the fuzzy QoS parameters of the requester and the relative benefit value offered by a provider, to select the most suitable service. Their typical limitation is that they apply a set of QoS parameters to select the most appropriate services without considering the structure of the application. However, most of this work has concentrated on how to fit each association to any abstract service individually and thus such approaches ignore possible associations of other abstract services and fail to reach an acceptable QoS for the composite services.

Previous works aim to overcome the single requester-provider problem. However, to reach an acceptable QoS for a composite service, the service composition structure must be taken into account, as it determines the topology of the decentralized execution of individual service, which affects the overall quality of the executed composite services. The computation of QoS values based on a QoS matrix is an appropriate solution. Services are ranked by computing each row of the normalized fuzzy QoS matrix [10]. Anyway, it was only a local optimization algorithm, but not a global one. The local optimization technique could not take global QoS constraints into consideration. When service size is very large, the overhead of global planning becomes enormous. Our method addresses these issues by considering both user requirements and the global optimal values.

III. Fuzzy QoS-aware selection algorithm

A. Problem Statement

The structure of an abstract service can be represented in terms of various control structures (e.g. sequence, flow, loop and choice). These structures present how various basic services are combined as a composite service. For instance, Fig. 1 reveals the structure of a sales order (SO) service. The loop structure is labeled with β indicating the number of iterations and the choice structure is labeled with α showing the probability of choosing one path.

A quality matrix $Q = ([q_{ij}]_{m*n})_k$ is defined where k is the QoS attribute for service j that is assigned to the abstract service i, m is size of the largest service list, and n is the scale. For the user, we specify QoS requirements and priority Pr_k meaning the approximated opinions of individuals. The priorities are restricted in [0, 1] and need to be normalized

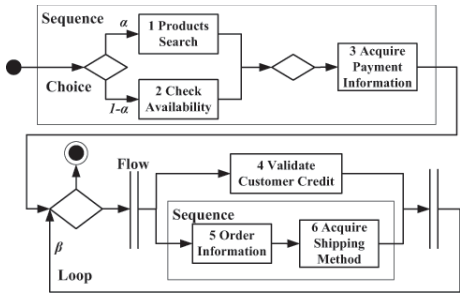

Fig. 1. The structure of sales order composite service.

by summing them together, then dividing each by the sum. Considering the SO process, The requirements on quality attributes (reliability, availability, latency, throughput, successability) are (78, 91, 72, 11.7, 84) and their respective priorities are (0.5, 0.1, 0.6, 0.2, 0.7). The problem now is to find the best combination which optimizes quality solution subject to the user requirements. Hence, it becomes a multi-objective optimization problem and we utilize FGA to solve it.

B. Fuzzy QoS Attributes of Services

Each candidate service is characterized by a set of QoS attributes, including: (1) reliability(R), (2) availability(A), (3) latency(L), (4) throughput(T), and (5) successability(S). We assume that all the concrete services have been discovered by the service requests and the values of their attributes for all these services have been given.

The QoS attributes should be extracted and fuzzied. QoS properties of all services are collected from different sources such as past logs, historical databases, archived records, etc. Due to the fact that the collected quality data, on which QoS analysis depends, is either out of date or gathered under different operation system and network conditions. This leads to the uncertainty in the current data. Hence, to handle the fuzziness in the data, the gained crisp data need be converted into Triangular fuzzy numbers (TFNs), denoted as [l, m, r], where the parameter m gives the modal value, and respectively, l, r, denote the smallest and the largest possible values [11]. Within a service selection scenario, crisp QoS values of the collected data need to be converted into TNFs with certain numerical spread φ. Hence, one can assume $\varphi = 15\%$, a crisp QoS value middQ can be expressed as a TFN $[middQ * 85\%, middQ, middQ * 115\%]$. For each QoS attribute k, when the quality matrix $([q_{ij}]_{m*n})_k$ (where $Q_{ij} = [leftQ_{ij}, middQ_{ij}, rightQ_{ij}]$) has been constructed, different matrices of various QoS features are incommensurable with each other, because the units and the value ranges of the collected QoS attributes are different. Thus, it is necessary to normalize the quality matrix $([q_{ij}]_{m*n})_k$ and this matrix can be normalized by dividing $([q_{ij}]_{m*n})_k$ by the maximum value of $([q_{ij}]_{m*n})_k$.

C. Aggregation Functions

A set of aggregation functions to calculate the aggregated QoS values for a service composition are presented. The basic

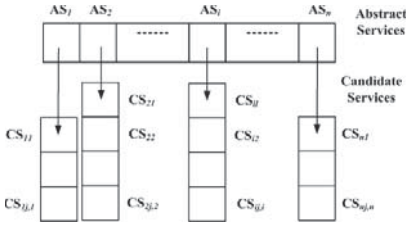

Fig. 2. Coding scheme.

ideas for computing the aggregated QoS are extended from crisp method with TFNs. Once the normalized quality matrix $([q_{ij}]_{m*n})_k$ are constructed, the aggregated QoS values can be computed using the developed principles of fuzzy arithmetic operations on TFNs [12], as listed in table I.

The ideas are basically reduction methods by recursively applying a reduction rule to shrink the composition structure. On every recursive step when applying a reduction method, R, A, L, T and S of the services concerned will be calculated and the structure immediately changes. It continues until an exact single service exists. At last, the results are the equivalent QoS values of the composition.

D. Defuzzification

In real situations, it is necessary to use crisp values in the calculations and thus to defuzzify the data. Fuzzification is the conversion of a precise quantity to a fuzzy quantity, and defuzzification is vice versa, fuzzy to precise quantity conversion. Defuzzification techniques includes Yager index, MC, COA, etc. The crisp value of the fuzzy number can be assessed by means of a standard measure and we adopt the index proposed by Yager to obtain this value [11]:

$$Y_3(\alpha) = \int_0^{\beta_{max}} M(\alpha_\beta) d\beta \tag{1}$$

Where $\beta_max = superu(x)$, α_β is the $\beta - cut$ of the fuzzy set α, and $M(\alpha_\beta)$ is the mean value of α_β. For any TFN $\alpha = [l, m, r], Y_3(\alpha) = (l+2*m+r)/4$. Then, the defuzzifed vaule for the TFN of each QoS property x (reliability, availability, latency, throughput, and successability) can be calculated.

E. Service Selection Using FGA

GA has gained significant attention for solving the QoS-based service composition problems. As an extension of GA, the FGA is elaborated on for implementing this approach.

Fig. 2 illustrates the procedure of encoding the control structures as a genome. FGA for the service selection problem is now presented as algorithm 1. Given a property space, the original fuzzy matrix for each QoS attribute is normalized. Then, all genomes of the population are generated through function $P(ind)$ with $ind = 0$ and the fitness values of all individuals in population $P(0)$ are calculated. A rank-based selection mechanism, a reproduction function and a mutation process are executed. According to the fitness function, the solution of each generation is returned. By comparing the current optimal solution with the optimal solution of the older generations, the global best solution is updated. Finally, it repeats and increases ind by 1 until the termination criterion is satisfied and outputs the optimal solutions.

Algorithm 1 Fuzzy Genetic Algorithm (FGA)

Input: The property space of the concrete services Q; The QoS requirements;
Output: Global Optimal Solution GS

1: Coding; Normalize Q;
2: Initialize the population $P(0)$; set $ind = 0$;
3: Evaluate $P(0)$ by fitness function;
4: **while** (not terminating condition) **do**
5: Select fitter individuals for reproduction;
6: Crossover the parents to form new offspring;
7: Mutate new offspring at each locus;
8: Evaluate fitness of each offspring with fitness function;
9: Compare to $pbest$ (population best): Compare each population with its $pbest$. If it is better than $pbest$, then let $pbest$ be the current population;
10: Compare to $gbest$ (global best): Compare all $pbests$ with $gbest$. If the highest in $pbests$ is larger than $gbest$, then update $gbest$;
11: $ind = ind + 1$, generate a new population $P(ind)$;
12: **end while**
13: **return** $gbest$ and stop the algorithm;

In the following section, we present the fitness function of FGA, as illustrated in Algorithm 2. It is used to find optimal solutions and eliminate inferior solutions by comparing their QoS values. It requires maximizing some QoS attributes (e.g., reliability), while minimizing the other qualities (e.g., latency). The fitness for each individual is calculated from the inner control structures of composition in the viewpoint of five QoS attributes. According to the control type, it recursively calculates the aggregated QoS values of its inner control structures and different formulas are applied to calculate its aggregated QoS value, with regards to table I. Afterwards, equation 1 is employed to compute the crisp value for each QoS type, to further obtain the fitness values.

The function $DV_{ik} = Dis(Q_D^k, O_j^k)$ represents the distance value of the requested QoS demands Q_D^k in connection with the QoS value O_j^k provided by the j-th selected service composition for the k-th attribute. We derive the distance measure, which gives the distance of two quantities. Suppose two variables A and B, the distance $DV = Dis(A, B) = 1 - \frac{|A-B|}{|A+B|}$ is computed by the normalized Manhattan distance [13].

Algorithm 2 computes the fitness for each individual which is gained from an inner control structure of the composition in consideration of five QoS properties. $D = ([d_{ij}]_{m*n})_k$, web service i in each service composition j, and QoS property k are given. This algorithm recursively calculates the aggregated QoS of a given service composition from its inner control structures to the outer control structures. Then, according to the current control structure type and type of aggregation, one mathematical formula is applied to compute its aggregated

TABLE I
FUZZY AGGREGATION FUNCTIONS FOR COMPUTING THE QOS OF A SERVICE COMPOSITION

QoS At.	Sequence	Choice	Flow	Loop
Rel.(R)	$\prod_{i=1}^{n}\left(d_R^l, d_R^m, d_R^r\right)_i$	$\sum_{i=1}^{n}\alpha\left(d_R^l, d_R^m, d_R^r\right)_i$	$\prod_{i=1}^{n}\left(d_R^l, d_R^m, d_R^r\right)_i$	$\left(d_R^l, d_R^m, d_R^r\right)^{\beta}$
Ava.(A)	$\prod_{i=1}^{n}\left(d_A^l, d_A^m, d_A^r\right)_i$	$\sum_{i=1}^{n}\alpha\left(d_A^l, d_A^m, d_A^r\right)_i$	$\prod_{i=1}^{n}\left(d_A^l, d_A^m, d_A^r\right)_i$	$\left(d_A^l, d_A^m, d_A^r\right)^{\beta}$
Lat.(L)	$\sum_{i=1}^{n}\left(d_L^l, d_L^m, d_L^r\right)_i$	$\sum_{i=1}^{n}\alpha\left(d_L^l, d_L^m, d_L^r\right)_i$	$\mathbf{Max}_{i=1}^{n}\left(d_L^l, d_L^m, d_L^r\right)_i$	$\beta\left(d_L^l, d_L^m, d_L^r\right)$
Res.(T)	$\sum_{i=1}^{n}\left(d_T^l, d_T^m, d_T^r\right)_i$	$\sum_{i=1}^{n}\alpha\left(d_T^l, d_T^m, d_T^r\right)_i$	$\mathbf{Max}_{i=1}^{n}\left(d_T^l, d_T^m, d_T^r\right)_i$	$\beta\left(d_T^l, d_T^m, d_T^r\right)$
Suc.(S)	$\prod_{i=1}^{n}\left(d_S^l, d_S^m, d_S^r\right)_i$	$\sum_{i=1}^{n}\alpha\left(d_S^l, d_S^m, d_S^r\right)_i$	$\prod_{i=1}^{n}\left(d_S^l, d_S^m, d_S^r\right)_i$	$\left(d_S^l, d_S^m, d_S^r\right)^{\beta}$

Algorithm 2 Fitness

Input: The property space of each concrete service $(D(j)_k$, for property k , and each service i for each service composition j); The aggregate functions $(D_1; D_2; ...; D_n)_k \rightarrow D_k$; The QoS requirements

Output: Ranked optimal solution $pbest$ of this population

1: **for** each service composition j **do**
2: **for** each candidate service i **do**
3: TypeOfFunction = CheckType (j, k);
4: **if** TypeOfFunction is Sequence **then**
5: Calculate $D(k, i)$ using $SequenceD(i, j)$;
6: **else** {TypeOfFunction is Switch}
7: Calculate $D(k, i)$ using $SwitchD(i, j)$;
8: **else** {TypeOfFunction is Flow}
9: Calculate $D(k, i)$ using $FlowD(i, j)$;
10: **else** {TypeOfFunction is Loop}
11: Calculate $D(k, i)$ using $LoopD(i, j)$;
12: **end if**
13: **end for**
14: Find the defuzzified value for property k;
15: Calculate DV_j^k of all i in j, for property k;
16: Calculate the fitness values F_j; Generate ranked list $S(j)$ by sorting their fitnesses;
17: Calculate the best $pbest$ among $S(j)$;
18: **end for**
19: **return** $pbest$;

QoS value in the light of Table I. Based on the aggregated QoS values for each service composition solution, the defuzzified matrix D for each QoS attribute is obtained. Finally, this defuzzified QoS matrix is used to calculate the fitnesses of all compositions and the best fitness $pbest$ returns as a output.

A fitness F_j for a given individual j, represents relative value offered by the service provider with respect to the QoS requirements. An F_j for the j-th solution can be computed as a function of DV_i^k in (2) of the requester-provider on each QoS attribute, weighted by the respective priority value Pr_k. The Fitness Function is denoted by:

$$F_j = \sum_{h=1}^{k}(Pr_j)_h * (DV_j)_h \tag{2}$$

Once the F_j have been calculated for each individual j in population, it can be used to rank the best-matching solution. The lower value of the fitness means a better solution.

IV. EXPERIMENTAL RESULT

We consider the SO service example shown in Fig. 1 and compare it to two other algorithms SGA and NLP which solve the same global optimization problem. To examine the solution quality of FGA compare to SGA and NLP, the fitness value is used based on two main standards: the composition with different number of abstract and candidate services accomplishing each abstract service. All the experiments are conducted in a computer with a 2.4 GHz Intel Core 2 Duo CPU and a 4 GB RAM. The input data used by the algorithms are derived from two data sources: a dataset containing 2507 real services and about 9 QoS values [14] and a randomly generated experimental QoS dataset whose values are within the same range of the former dataset. To configure the parameters in FGA, we use generation 100, crossover 0.85, and mutation 0.05, as it rendered the optimal solution in our experiments. All the algorithms are examined with the same data and their minimum fitness values are reported.

In our first experiment, we want to evaluate how the algorithms perform with different sizes of abstract services. Ten test cases are generated in the SO service through configuring parameter β. The scale ranges from 6 to 33 with an increment of 3. The sizes of the candidate services and the initial population are set as 75 and $[\#abstractservices/3] \times 25$. In the choice structure, the probability of choosing one path is 1/2. Fig. 3(a) presents that FGA outperforms SGA and NLP with a variable number of abstract services. Fig. 3(b) demonstrates that with the growth scale, the time of FGA and SGA methods increase slowly and approximately. This is because the complexity of two algorithms largely depends upon the population size, which is the same for both algorithms. In comparison to NLP, the execution time of FGA becomes significantly slower while the scale grows.

Our second experiment is conducted to assess how three algorithms act in regards to different number of candidate services per abstract service, with $\alpha = 1/2$ and $\beta = 10$. The number of services is in the range from 10 to 2000. We use such a large number of candidate services so that the QoS values of services are different from each other. The initial population size is $[\#webservices/25] \times 3$. Fig. 4(a) shows that it demonstrates a series of fractional changes in the fitness result, as the number of services per abstract service are increasing so that there are more choices available. The fitness curve figures that the superiority of FGA to SGA and NLP is much more apparent with available service number

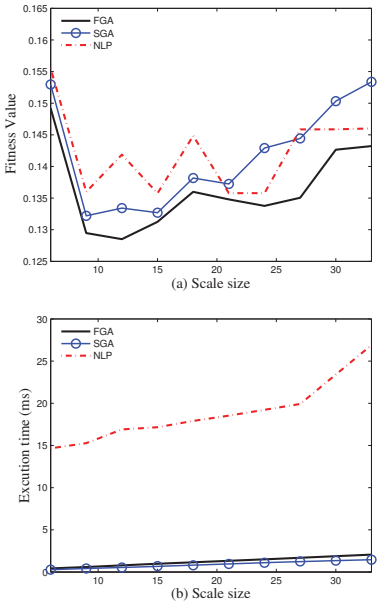

Fig. 3. Fitness and time illustration with number of abstract services.

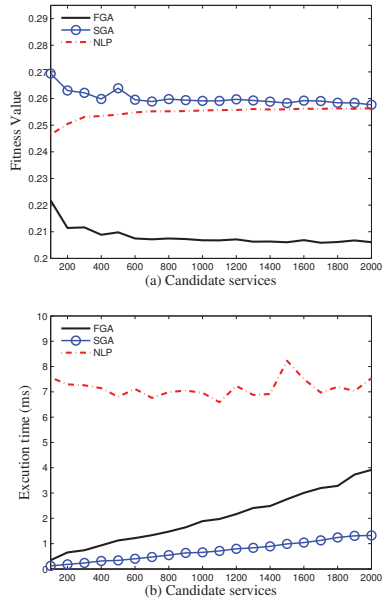

Fig. 4. Fitness and time illustration with number of candidate services.

rising. Then a conclusion could be drawn as that the space exploitation ability of FGA is much better than SGA and NLP. Fig. 4(b) presents that FGA is slightly slower than SGA, because SGA only handles the crisp data, while FGA deals with the fuzzy data which contains the left and right spreads of the modal data. If both fitness and time are taken into consideration, it demonstrates that FGA can work out the problem with lower fitness and better efficiency.

From the results, we can conclude that FGA outperforms SGA and NLP. The lowest fitness scores of FGA are inferior to the scores of the other two algorithms, which demonstrate the powerful global exploration-ability of FGA in finding the optimal solutions. The running time of FGA and SGA methods increase in a slow and similar speed, but the time of NLP is larger. This is because the time of FGA and SGA are mostly dependent on the population size. For NLP, the running time is largely effected by the population size and the range of the QoS values. The results indicate that the proposed FGA remarkably surpasses SGA and NLP with respect to the number of abstract and candidate services.

V. Conclusions

Many of the current selection algorithms are rather limited due to the uncertainty in the quality data and user QoS requirements. In this paper, the FGA is proposed based on FST and GA to contribute to QoS-aware service selection problem. Our algorithm takes the fuzziness of the data into consideration and ranks the services based on how close they are to the optimal values as well as how close they are to the actual user request. The result is more tailored towards real cases and users with specific requirements. We provide comparisons with SGA and NLP illustrating its superior effectiveness in regards to the powerful searching-ability.

There are a few directions we would like to work on in the future. Firstly, we would like to conduct the comparative experiments with more QoS attributes and more algorithms, to evaluate their impacts on the performance. Secondly, we would like to incorporate the fuzzy technique with the crisp method and evaluate our approach in complex applications.

References

[1] D. Benslimane, S. Dustdar, and A. Sheth, "Services mashups the new generation of web applications," *IEEE Internet Computing*, vol. 12, no. 5, pp. 13–15, 2010.

[2] F. Mardukhi, N. Nematbakhsh, K. Zamanifar, and A. Barati, "Qos decomposition for service composition using genetic algorithm," *Applied Soft Computing*, vol. 13, no. 7, pp. 3409–3421, 2013.

[3] A. Klein, I. Fuyuki, and S. Honiden, "Sanga a self-adaptive network-aware approach to service composition," *TSC*, pp. 1–14, 2013.

[4] 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*, vol. 6, pp. 1–26, 2007.

[5] P. Wang, K.-M. Chao, and C.-C. Lo, "On optimal decision for qos-aware composite service selection," *ESA*, vol. 37, pp. 440–449, 2010.

[6] Z. Ye, X. Zhou, and et al, "Genetic algorithm based qos-aware service compositions in cloud computing," in *ICDSAA*, 2011, pp. 321–334.

[7] L. Mikhailov and P. Tsvetinov, "Evaluation of services using a fuzzy analytic hierarchy process," *Applied Soft Computing*, pp. 23–33, 2004.

[8] D. Bacciu, A. Botta, and D. C. Stefanescu, "A framework for semantic querying of distributed data-graphs via information granules," in *The 10th IASTED ICISC*, 2007, pp. 161–166.

[9] D. Bacciu, M. G. Buscemi, and L. Mkrtchyan, "Adaptive fuzzy-valued service selection," *ACM SAC*, pp. 22–26, 2010.

[10] W. Zhou and J. W. et al., "A qos preference-based algorithm for service composition in service-oriented network," *Optik*, pp. 4439–4444, 2013.

[11] R. R. Yager, "A procedure for ordering fuzzy subsets of the unit interval," *Inform Science*, vol. 24, pp. 143–161, 1981.

[12] H. Garg, "Reliability analysis of repairable systems using petri nets and vague lambda-tau methodology," *ISA Transactions*, pp. 6–18, 2013.

[13] V. Cross and T. Sudkamp, *Similarity and compatibility in fuzzy set theory*. Physica-Verlag, 2002.

[14] E. Al-Masri and Q. Mahmoud, "The qws dataset," Website, 2014, http://www.uoguelph.ca/qmahmoud/qwsindex.html.