Università di Pisa
Dipartimento di Informatica

Technical Report: TR-09-21

Adaptive Service Selection - A Fuzzy-valued Matchmaking Approach

Davide Bacciu
Dipartimento di Informatica, Università di Pisa,
Largo B. Pontecorvo 3, 56127 Pisa, Italy
bacciu@di.unipi.it

Maria Grazia Buscemi
IMT Lucca Institute for Advanced Studies,
P.zza San Ponziano 6, 55100 Lucca, Italy,
m.buscemi@imtlucca.it

Lusine Mkrtchyan
IMT Lucca Institute for Advanced Studies,
P.zza San Ponziano 6, 55100 Lucca, Italy,
l.mkrtchyan@imtlucca.it

October 31, 2009

Address: Largo B. Pontecorvo 3, 56127 Pisa, Italy. Tel: +39 050 2212700 Fax: +39 050 2212726

Adaptive Service Selection - A Fuzzy-valued Matchmaking Approach

Davide Bacciu
Dipartimento di Informatica, Università di Pisa,
Largo B. Pontecorvo 3, 56127 Pisa, Italy
bacciu@di.unipi.it

Maria Grazia Buscemi
IMT Lucca Institute for Advanced Studies,
P.zza San Ponziano 6, 55100 Lucca, Italy,
m.buscemi@imtlucca.it

Lusine Mkrtchyan
IMT Lucca Institute for Advanced Studies,
P.zza San Ponziano 6, 55100 Lucca, Italy,
l.mkrtchyan@imtlucca.it

October 31, 2009

Abstract

Service composition concerns both integration of heterogeneous distributed applications and dynamic selection of services. QoS-aware selection enables a service requester with certain QoS requirements to classify services according to their QoS guarantees. In this paper we present a method that allows for a fuzzy-valued description of QoS parameters. Fuzzy sets are suited to specify both the QoS preferences raised by a service requester such as 'response time must be as lower as possible and cannot be more that 1000ms' and approximate estimates a provider can make on the QoS capabilities of its services like 'availability is roughly between 95% and 99%'. We propose a matchmaking procedure based on a fuzzy-valued similarity measure that, given the specifications of QoS parameters of the requester and the providers, selects the most appropriate service among several functionally-equivalent ones, using a fuzzy ordering relation. We also devise a method for dynamical update of service offers by means of runtime monitoring of the actual QoS performance.

This Technical Report is an extended version of the article in press D. Bacciu, M. Buscemi and L. Mkrtchyan, "Adaptive Fuzzy-valued Service Selection", *Proceedings of the 25th ACM Symposium On Applied Computing (SAC'10)*, 2010, ACM

1 Introduction

In the recent years, the widespread development of web-based applications has promoted an incredible growth in the number of Web Services offered throughout a highly dynamical environment, thus making adaptive service discovery and composition a critical issue. Dynamic Web services composition essentially deals with the discovery of service providers that satisfy consumer-related functional and nonfunctional requirements, including cost and *Quality of Service* (QoS) constraints, such as performance and availability. By dynamical environments we refer to those contexts where services appear and disappear unpredictably, being characterized by a continuous evolution of the runtime conditions, such as changes to resource availability and/or to network connectivity. Hence, an effective exploitation of such dynamic Web Service environments requires tools and models supporting dynamic and adaptive service collaborations.

Meeting QoS requirements is one of the most fundamental aspects for ensuring a successful service composition. The typical scenario comprises a service requester, characterized by certain QoS requirements, which seeks the service provider that, amongst a pool of candidate solutions, can best satisfy its QoS constraints. The matchmaking of the service requests and offers is thus composed of two parts

1. All providers fulfilling the requirements from the service requester are selected (i.e. the *candidate services*).

2. Candidate services are ranked to choose the best solution, i.e. that offer that is closer to and/or best satisfies the client QoS requirements.

We present a novel approach to adaptive QoS specification and service selection that exploits ideas and concepts from *Fuzzy Set Theory* (FST). FST has been developed starting from the observation that most real-life problems can hardly be modeled by conventional mathematical tools, as they are characterized by an inherent ambiguity and uncertainty. The basic idea of the fuzzy approach is to allow an element s to belong to a set φ with a degree of membership ranging in the continuous real interval $[0, 1]$ rather than in $\{0, 1\}$. Within a service selection scenario, such *fuzzyness* in the description of the concepts allows to intuitively model the vagueness of the service level specification and matching.

In this paper we exploit FST to extend the service selection method described in [15], thus obtaining a more expressive service specification as well as a flexible service selection process, that can effectively deal with the inherent vagueness and imprecision of QoS description both on the client and on the provider side. More in detail, by smoothing the interval of legal QoS values to be fuzzy sets, we can obtain a larger set of candidate services, i.e. of services meeting both functional and QoS requirements, hence selecting a service even when other non-fuzzy methods find no solution. Conversely, once we restrict to candidate services, our technique yields results that are in accordance with the original approach.

The proposed method allows for a fuzzy-valued description of QoS parameters. Fuzzy sets are suited to specify both the QoS preferences raised by a service requester such as 'response time must be as lower as possible and cannot be more that 1000ms' and approximate estimates a provider can make on the QoS capabilities of its services like 'availability is roughly between 95% and 99%'. We propose a matchmaking procedure based on a fuzzy-valued similarity measure that, given the specifications of QoS parameters of the requester and the providers, selects the most appropriate service among several functionally-equivalent ones, using a fuzzy ordering relation. We also devise a method for dynamical update of service offers by means of runtime monitoring of the actual QoS performance.

Our model has a prototype implementation, which is available as MATLAB code. Consequently, we claim that our approach can be readily integrated within the existing Dino platform with only minor modifications to the service specification and matching modules. Nevertherless, we remark that tailoring our method to Dino has been a matter of choice. In fact, we argue that our technique can be smoothly adapted to provide fuzzy-valued extensions of other non-fuzzy QoS matchmaking methods.

Related Work General approaches to QoS-based service selection are presented in [12] and [20]. Both models compare the QoS offered by a provider with that of other competing providers during service selection, but they do not take into account the QoS requirements by the service requester itself. In [12] the QoS value offered by a provider is compared with the average QoS value offered by all providers. In [20], each QoS offer is compared with the maximum and minimum QoS values exposed by the alternative providers. Some works model QoS-aware composition as an optimization problem: [4], for instance, uses genetic algorithms to find a near-optimal service composition, while in [1], the authors formulate the problem as a global optimization task by jointly optimizing the composition across each possible execution path.

Research in QoS-aware service composition using fuzzy logic has produced several interesting models in the recent years. In [14], for instance, a fuzzy analytic hierarchy is used to identify suitable service offers by accepting/discarding solutions based on crisp priority weights resulting from the comparison of fuzzy-valued QoS descriptions. In [11], QoS-aware service composition is discussed as a fuzzy constraint satisfaction problem, while in [16] the problem is explored as a distributed fuzzy constraint satisfaction task in which service providers from different nested levels in a composition can collaborate to solve the joint service aggregation problem. In [9], the authors propose a multi-agent consensus approach based on fuzzy logic and semantic web technologies, that assists providers and consumers in reaching consensus on their service advertisement and request by taking into consideration their expectations and preferences. A QoS-aware approach to service contract negotiation is presented in [2]: this model uses fuzzy reasoning to provide tailored service configurations with an adaptive pricing policy that allows trading the offer-request similarity with the

price of the final solution.

The typical limitation of the above approaches is that they restrain the use of fuzzy methodology to the requirements specification. In particular, the fuzzy representation is typically abandoned when aggregating the solution over multiple QoS dimensions and when ranking the alternative proposals. In this paper, on the other hand, we provide a complete fuzzy-valued matchmaking framework that exploits the power of fuzzy modeling throughout the whole QoS-aware service agreement process, including ranking the acceptable solutions by means of an ordering relation over fuzzy quantities.

The rest of this paper is organized as follows: Section 2 gives a brief description of the Dino approach. The extended fuzzy-Dino model is described in Section 3, while Section 4 describes the adaptation mechanism that allows to dynamically update fuzzy service descriptions as well as priority and trustworthiness weights. To show the effectiveness of the approach, in Section 3 and Section 4 we provide numerical examples based on an automotive case study of the EU Project Sensoria [10]. Finally, we give concluding remarks and future work plans in Section 5.

2 QoS-Aware Service Composition in Dino

This section gives a brief overview of the service selection algorithm developed within the Dino approach. The interested reader can refer to [15] for a more detailed description of the method.

Dino provides a runtime infrastructure consisting of a number of brokers. These brokers are responsible for, among other things, service discovery and selection. Services can be discovered using a service registry or a search engine, and by matching functional and QoS properties of each required service with those of the offered services. Service requirements are specified in an ad-hoc defined XML specification language. The requirement specification document (ReqDoc) of a service requester declares the service requirements and, in particular, an interval of admitted values for every QoS dimension. Similarly, the capability specification document (CapDoc) of a service provider contains a range of values that expresses the capability offered for every QoS dimension.

For every request (using the ReqDoc documents), the broker first discovers the candidate services (possibly supplied by different providers) by performing functional and QoS matchmaking (using the CapDoc documents). If there is more than one service that meets both functional and QoS requirements of the requester, the broker assigns an appropriateness value to each of these services. Such a value takes into account the relative benefit offered by a specific service with respect to the requester-specified QoS criteria, as well as the trustworthiness of the provider of that service, which is computed on the basis of the provider's past record in meeting its QoS obligations. Every time a Dino broker discovers a violation of the service agreement, it decreases the trustworthiness of the provider and starts adapting the service binding.

In Dino the appropriateness value AV_j of the j-th service is made up of two

components: a benefit value (ABV_j) and a trust rate (TR_j). ABV_j is obtained by the weighted sum of the BV_{ij} values, where each BV_{ij} represents the relative benefit offered by the j-th service with respect to the i-th QoS dimension:

$$BV_{ij} = \begin{cases} 1 - \frac{maxVal_{Q_i} - mpVal_{Q_i}}{lpVal_{Q_i} - mpVal_{Q_i}} & \text{if } mpVal_{Q_i} < lpVal_{Q_i} \\ \\ 1 - \frac{mpVal_{Q_i} - minVal_{Q_i}}{mpVal_{Q_i} - lpVal_{Q_i}} & \text{if } mpVal_{Q_i} > lpVal_{Q_i} \end{cases}$$

In the above formula, Q_i is the i-th QoS dimension, $minVal_{Q_i}$ and $maxVal_{Q_i}$ are the minimum and maximum values of Q_i as offered by the service provider, while $mpVal_{Q_i}$ and $mpVal_{Q_i}$ are the most preferred and the least preferred values of Q_i, respectively, as required by the service requester. The upper part of the above formula is used when the most preferred value is smaller than the least preferred one, like with a QoS parameter such as response time. In this case, note that $minVal_{Q_i}$ is not used. Indeed, the service selection procedure only has to take into account the worse QoS values the provider can offer, i.e. $maxVal_{Q_i}$. For instance, when considering response time, the only relevant value offered by a given service is the maximum, while minimum is disregarded. A similar argument applies to the lower part of the formula above, noting that now the most preferred value is greater than the least preferred one, like in the case of (service) availability.

Next, the value of BV_{ij} is multiplied by the confidence level (if specified) of the service provider in the offered values. That is, for every provider $\overline{BV}_{ij} = C_{ij} \cdot BV_i$, where the value C_{ij} specifies the required minimum confidence level of the j-th service provider in delivering the capabilities of the i-th QoS dimension. The value of the confidence attribute is between 0 and 1, and it is an indicative of the probability that the stated QoS values will be delivered by the provider. Once the values of BV_{ij} for i and j are computed, the aggregate benefit value ABV_j of the j-th solution is obtained as the weighted average of the BV_{ij} values:

$$ABV_j = \frac{1}{n} \sum_{i=0}^{n} P_i \cdot BV_{ij}$$

where P_i is the relative priority of Q_i as expressed by the service requester, and which has a value between 0 and 1.

The appropriateness value AV_j of the j-th service is now computed as the weighted average of ABV_j and the trust rating TR_j. By default, ABV_j and TR_j are assigned equal weights, i.e. $AV_j = \frac{1}{2}(BV_j + TR_j)$. A service requester can assign different weights to BV_j and TR_j if required, by providing the weights to be used to the Dino broker. Finally, the service with highest AV_j is selected by the broker.

3 Fuzzy-valued Service Provider Selection

This section proposes a generalization of the Dino's matchmaking process by extending it to allow the specification and processing of imprecise QoS proper-

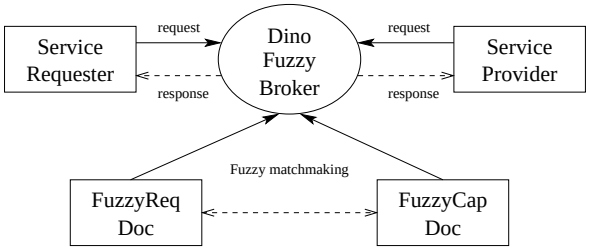
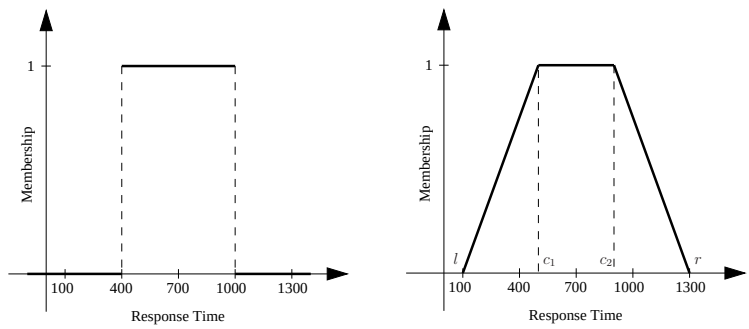


Figure 1: Fuzzy Dino Architecture

ties by means of fuzzy numbers. Such representation is intended to provide Dino with an increased expressivity, that enables a straightforward human-understandable interpretation of the QoS specification while introducing a more robust matchmaking process that is able to smoothly cope with the underlying vagueness of QoS requirements. To this end, we introduce the use of fuzzy sets in the XML specification language provided by Dino, allowing the definition of requirement and capability documents containing fuzzy terms. Following the architectural sketch in Figure 1, a fuzzy broker first receives the fuzzy requirement specification from the service requester and the fuzzy capability specification documents from the service providers. Then, it performs a fuzzy matchmaking process, selecting the best matching offer to be presented to the requester based on a novel ordering relation defined over fuzzy-valued appropriateness values. In the remainder of the section, we first introduce the basic idea of fuzzy-valued service description (Section 3.1), before moving to the details of the fuzzy matchmaking process (Section 3.2).

3.1 Fuzzy-Valued Service Description in Dino

Dino's approach to QoS specification can be well explained by a graphical interpretation of its constraint description. Consider, for instance, a customer requesting a specific service with a *Response Time* between $400ms$ and $1000ms$. The specification of such QoS request results in the scenario described in Fig.2.a: the membership value on the y-axis indicates the acceptability of the response time values, where 1 specifies acceptable values while 0 identifies unacceptable response times. Clearly, there is an abrupt change between acceptable and unacceptable values. In real life, the transition from satisfactory to unsatisfactory solutions is seldom so sharp, with a smooth interpolation between what it is considered fully adequate to what is perceived as totally unacceptable. Graphically, this can be well represented by a smooth membership function which degrades softly from its supremum to its infimum values. Formally, such vagueness and imprecision can be modeled by fuzzy sets and fuzzy sets theory. Figure 2.b, for instance, formalizes the linguistic QoS constraint "*Response Time is roughly between* $400ms$ *and* $1000ms$" by means of a *trapezoidal fuzzy number* whose membership function indicates that values in $[500, 900]$ are fully satisfactory, while



(a) Interval-valued QoS (b) Fuzzy-valued QoS

Figure 2: QoS specification using flat intervals (a) and trapezoidal fuzzy sets (b).

acceptability reduces as the response time falls below $500ms$ (or increases above $900ms$), being totally unsatisfactory under $100ms$ (respectively, over $1300ms$). A similar argument applies if the above statement '*Response Time* between $400ms$ and $1000ms$' refers to the QoS capability of a given service. In this last case the membership function may for instance express the degree of accuracy of the provider measurement.

Before proceeding with the formalization of fuzzy Dino approach, we briefly review the basic concepts of fuzzy set theory that are used throughout the paper, starting with the definition of fuzzy sets and fuzzy numbers.

Definition 3.1 *A* fuzzy set $\widetilde{A} \in \mathcal{F}_U$ *over a universe of discourse* U *is characterized by a* membership function $\mu_{\widetilde{A}}(x)$ *associating each element* $x \in U$ *to a value in the interval* $[0, 1]$*, that is the* membership value *of* x *in* $\widetilde{A}$*.*

Definition 3.2 *A fuzzy set* $\widetilde{A}$ *on* U *is* convex *if and only if for all* $x_1, x_2 \in U$ *it holds*

$$\mu_{\widetilde{A}}(\lambda x_1 + (1 - \lambda)x_2) > \min(\mu_{\widetilde{A}}(x_1); \mu_{\widetilde{A}}(x_2))$$

where $\lambda \in [0, 1]$*.*

Definition 3.3 *A fuzzy set* $\widetilde{A}$ *on* U *is* normal *if* $\exists x_i \in U$ *s.t.* $\mu_{\widetilde{A}}(x_i) = 1$*.*

Definition 3.4 *A* fuzzy number *is a fuzzy subset on* U *that is both convex and normal. A* trapezoidal fuzzy number $\widetilde{n}$ *is a quadruplet* (n_1, n_2, n_3, n_4) *such that its membership function* $\mu_{\widetilde{n}}(x)$ *is defined as*

$$\mu_{\widetilde{n}}(x) = \begin{cases} 0, & \text{if } x < n_1 \\ \frac{x-n_1}{n_2-n_1}, & \text{if } n_1 \leq x \leq n_2 \\ 1, & \text{if } n_2 \leq x \leq n_3 \\ \frac{n_4-x}{n_4-n_3}, & \text{if } n_3 \leq x \leq n_4 \\ 0, & \text{if } x > n_4 \end{cases}.$$

Table 1: Priority/trustworthiness fuzzy values and corresponding linguistic terms [19]

Linguistic Variables	Triangular Fuzzy Values
Very low (VL)	$(0, 0, 0.1)$
Low (L)	$(0, 0.1, 0.3)$
Medium Low (ML)	$(0.1, 0.3, 0.5)$
Medium (M)	$(0.3, 0.5, 0.7)$
Medium High (MH)	$(0.5, 0.7, 0.9)$
High (H)	$(0.7, 0.9, 1.0)$
Very High (VH)	$(0.9, 1.0, 1.0)$

If $n_2 = n_3 = c$, $\widetilde{n} = (l, c, r)$ *is a* triangular fuzzy number.

Notice that standard concepts and operators from crisp set theory can be seamlessly generalized to fuzzy sets, yielding to the definition of fuzzy emptiness, equality, intersection and union [22]. Further, traditional crisp functions can be extended to work on fuzzy sets, via the *extension principle* [6]: for instance crisp arithmetic operators can be generalized to fuzzy numbers, yielding to the definition of fuzzy algebras. By exploiting such generalization, we build a fuzzy matchmaking procedure that extends the crisp service provider selection process implemented by Dino. In particular, following the definitions given above, we describe Dino's QoS specifications as trapezoidal fuzzy numbers (see Fig. 2.b), while priority and trustworthiness are expressed as fuzzy triangular weights. Following a standard approach from fuzzy decision making literature [19], we model priority P_i by a *linguistic variable* taking values from a set of linguistic terms, such as *Very low* or *High*, each associated to a specific triangular fuzzy set as described in Table 1. The use of linguistic variables allows the user to gather an immediate interpretation of the priority constraint that it is being placed on the QoS attribute by choosing a P_i value. The same approach, possibly with different specification of the linguistic terms, is taken for modeling the fuzzy-valued trustworthiness of the j-th provider, i.e. the linguistic variable TR_j.

With respect to the XML specification of the fuzzy QoS constraints, we need to extend Dino's specification language to admit the description of service properties as trapezoidal fuzzy sets. In particular, the original interval value couples $(lpVal, mpVal)$ used in the specification of ReqDoc and the worse value the provider can offer as expressed in CapDoc (either $minVal$ or $maxVal$) are replaced by the quadruplets $(leftSp, leftCore, rightCore, rightSp)$, that denote the trapezoidal fuzzy numbers with support in $[leftSp, rightSp]$ and core (i.e. elements with membership equal to 1) in $[leftCore, rightCore]$. Similarly, the original crisp specification of priority and trustworthiness TR are replaced by triangular fuzzy numbers with support in $[0, 1]$, defined by the triplet $(leftSp, centr, rightSp)$. Each of such triangular fuzzy numbers can be associated to a linguistic term (see Table 1), possibly defined within a constraints' ontology, to facilitate human interpretation of priority and trustworthiness values.

Table 2: QoS Requirements and Priorities for Map Services

	Request	Priority
Response time	$(0, 0, 0, 2000)$	$(0.9, 1, 1)$
Availability	$(.94, .99, 1, 1)$	$(0, 0.1, 0.3)$

Table 3: QoS Capabilities and Trustworthiness values of Map Services

	Resp. Time Caps.	Avail. Cap.	Trustworth.
M_1	$(1200, 1300, 1500, 1600)$	$(.92, .93, .95, .96)$	$(0.3, 0.5, 0.7)$
M_2	$(1000, 1100, 1300, 1400)$	$(.93, .94, .96, .97)$	$(0.7, 0.9, 1)$
M_3	$(300, 400, 600, 700)$	$(.90, .91, .93, .94)$	$(0.3, 0.5, 0.7)$

In [15] the selection algorithm shown in Section 2 has been applied over an example that is taken from an automotive case study of the EU Project Sensoria. In this scenario, a service entity called Driving-Assistant provides a route planning service for giving route-guiding instructions to a driver. In order to provide the route planning service, Driving-Assistant in turn requires a Map service. Dino allows to select the most appropriate Map service according to two QoS requirements placed by Driving-Assistant, namely response time and availability. Throughout this section we will consider a fuzzy-valued variant of that example to show how our procedure extends the original Dino approach by offering a more flexible tool for QoS-aware service selection. The results are generated by running a prototype implementation of our method.

Table 2 shows the fuzzy sets representing requirements of Driving-Assistant on response time and availability of the Map service, along with their respective priority. For instance, the interpretation of availability is that the requester: (i) does not accept a service with availability less than 94%, (ii) is increasingly more satisfied with values between 94% and 99%, and (iii) is fully satisfied with any value over 99%. Moreover, the priority values correspond to assigning to response time a very high priority while giving availability a low priority. Assume now the broker has discovered three functionally equivalent Map services. In Table 3, we report the fuzzy sets ($\mathit{leftSp}, \mathit{leftCore}, \mathit{rightCore}, \mathit{rightSp}$) specifying approximated yet informative estimates of the worse offered value for every QoS dimension (see Section 4 for more details on how such performance evaluation can be obtained) and the trustworthiness TR of the service providers.

3.2 Matchmaking with Fuzzy-Valued Similarity Metrics

In the previous section, it has been shown how Dino's service description can be extended to allow specifying vague and imprecise QoS requirements by means of trapezoidal fuzzy numbers. Similarly, there are several ways by which the matchmaking process can be extended to cope with such fuzzy service descriptions. The solution we propose in the following is motivated by the idea of maintaining the expressivity of the fuzzy representation as much as possible, by ranking the provider solutions directly from their aggregated fuzzy descrip-

tion. To this aim, we need to define proper operators for implementing fuzzy matchmaking, that are

- A function $BV_{ij} = \overline{\text{Sim}}(R_i, O_{ij})$ providing a fuzzy-valued assessment BV_{ij} of the *similarity* of the fuzzy QoS request R_i with respect to the fuzzy QoS value O_{ij} offered by the j-th provider for the i-th service attribute.
- An *aggregation function* $BV_j = \overline{\text{Sum}}(P_1, \ldots, P_n, BV_{1j}, \ldots, BV_{nj})$ that, for a given provider j, computes a fuzzy-valued score measuring the suitability of the aggregated solution as a function of the fuzzy valued similarity of the request-offer on each QoS attribute, i.e. BV_{ij}, weighted by the respective priority value P_i.
- A suitable *ordering relation* [17] on fuzzy quantities $\leq_{\mathcal{F}}$ that can be used to rank the fuzzy appropriateness values $AV_1, \ldots, AV_j, \ldots, AV_p$, i.e. fuzzy numbers measuring the appropriateness degree of each providers offer, and to select the provider publishing the best-matching service.

In particular, the definition of $\overline{\text{Sim}}$ and of $\leq_{\mathcal{F}}$ is not immediate, as it requires a sound choice of $\leq_{\mathcal{F}}$ among the many alternative options in the literature [17], as well as the definition of a proper $\overline{\text{Sim}}$. Due to this tight coupling between the definition of the fuzzy-valued similarity function and the choice of the ordering relation, we will first consider this two issues altogether, before moving further to describe fuzzy valued aggregation.

To the best of our knowledge, the only approach to fuzzy-valued similarity of fuzzy quantities has been introduced by Dubois and Prade [5, 7]. Their approach, however, provides fuzzy-valued similarities that cannot be correctly ranked by most of the existing ordering relations of fuzzy quantities $\leq_{\mathcal{F}}$ (see [3] for a detailed discussion). To solve this issue, in [3] it has been introduced an *ordered fuzzy-valued similarity measure* $\overline{\text{Sim}}_{OFM}$ that allows to preserve a correct ranking of the fuzzy similarity values. Given two fuzzy quantities k_1 and k_2, the similarity $\overline{\text{Sim}}_{OFM}(k_1, k_2)$ is a triangular fuzzy quantity with support in $[0, 1]$. The *core* of such triangular fuzzy set (i.e. the element $x \in [0, 1]$ with membership 1) is computed by a standard set-theoretic similarity measure between two fuzzy quantities $k_1, k_2 \in \mathcal{F}$, that is the well know Jaccard index [5]

$$\text{Sim}_J(k_1, k_2) = \frac{|k_1 \cap_{\mathcal{F}} k_2|}{|k_1 \cup_{\mathcal{F}} k_2|}, \tag{1}$$

where $|\cdot|$ is set cardinality, while $\cap_{\mathcal{F}}$ and $\cup_{\mathcal{F}}$ are fuzzy intersection and union, respectively. The triangle extremes l, r of the $\overline{\text{Sim}}_{OFM}(k_1, k_2)$ fuzzy number (i.e. the left l and right r spread of the triangle) are computed by the proximity-based normalized Minkowski 1-metric [5]

$$l = \frac{1}{2} \cdot (\text{Sim}_M(k_1, k_2)) = \frac{1}{2} \left(1 - \frac{\int_X |\mu_{k_1}(x) - \mu_{k_2}(x)| \, dx}{\int_X dx} \right).$$

Roughly speaking, the proximity-based index allows assessing the uncertainty of the similarity measured by the set-theoretic index, resulting in the infimum

Table 4: Benefit values for Response Time and Availability

	Response Time	Y_2 R. T.	Availability	Y_2 A.
Map_1	$(0, 0.09, 0.54)$	0.18	$0.10, 0.13, 0.15$	0.05
Map_2	$(0.02, 0.05, 0.08)$	0.19	$0.03, 0.23, 0.43$	0.12
Map_3	$(0, 0.12, 0.50)$	0.26	$0.2, 0.05, 0.08$	0.05

$\tilde{0}_{OFM}$ (i.e. maximum discrepancy) and the supremum $\tilde{1}_{OFM}$ (i.e. equality) being the singletons in 0 and 1, respectively. More formally, we have $\mu_{\tilde{1}_{OFM}}(1) = 1$ and $\mu_{\tilde{1}_{OFM}}(x) = 0 \ \forall x \neq 1$, and similarly for $\mu_{\tilde{0}_{OFM}}(x)$. Given this choice of infimum and supremum, we can assess fuzzy ordering by means of a standard fuzzy ordering measure, i.e. Yager's second index [21]

$$Y_2(k) : \mathcal{F} \to \mathbb{R} = \int_0^1 \frac{\sup(k_\alpha) + \inf(k_\alpha)}{2} d\alpha. \tag{2}$$

By using such Yager index Y_2 to define a ranking $\leq_{\mathcal{F}}$ of the fuzzy-valued similarities, it holds that $\tilde{0}_{OFM} \leq_{\mathcal{F}} \tilde{k} \leq_{\mathcal{F}} \tilde{1}_{OFM}$ for all $\tilde{k} \in \mathcal{F}_U$.

Given these sound ranking properties, we can compute Dino's benefit value $BV_{ij} = \overline{\text{Sim}}_{OFM}(R_i, O_{ij})$ of the j-th service as the fuzzy-valued similarity between QoS request R_i and offer O_{ij}. The Yager index Y_2 can then be used to define the ordering $\leq_{\mathcal{F}}$ over fuzzy-valued aggregated solutions, allowing to select the provider offering the best-matching service. Following Dino's matchmaking specification, we need to smooth the benefit value BV_{ij} by including the contribution of the confidence level C_{ij} that the provider of the j-th service has on the offered values for the i-th QoS dimension. Assuming the confidence level to be a crisp value, we can compute the smoothed benefit by scalar multiplication

$$\overline{BV}_{ij} = C_{ij} \cdot BV_{ij}$$

resulting in $\overline{BV}_{ij}$ to be a triangular fuzzy number.

Going back to the Map service example, Table 4 gives the benefit values BV_{ij} and their respective Yager indexes Y_2. We assume confidence to take the default value 1, i.e. $\overline{BV}_{ij} = BV_{ij}$. Indeed, confidence is kept a crisp parameter in our model and, hence, it is of no particular interest for our method. A simple summation of the BV values for every QoS-dimension suggests that map service M_2 is likely to be the most favourite candidate.

To provide a cumulative measure-of-fit ABV_j for the j-th solution, we need to define a suitable aggregation function that sums the fuzzy-valued smoothed benefits $\overline{BV}_{ij}$ for all the required service dimensions. Following the definition in Section 2, we can generalize the crisp weighted sum to fuzzy quantities as

$$ABV_j = (\frac{1}{n}) \bigoplus_{i=1}^{n} P_i \otimes \overline{BV}_{ij} \tag{3}$$

where P_i is the fuzzy priority of $\overline{BV}_{ij}$ defined in Section 3.1, while $\bigoplus$ and $\otimes$ are the fuzzy equivalents of crisp summation and multiplication, respectively.

Table 5: Aggregated benefit values and Yager indexes

	ABV	Y_2
Map_1	$(0, 0.05, 0.28)$	0.09
Map_2	$(0, 0.07, 0.28)$	0.10
Map_3	$(0.01, 0.12, 0.23)$	**0.12**

A standard approach for the implementation of extended arithmetic operators on fuzzy numbers is based on Zadeh's *extension principle* which, however, results in computationally complex procedures that are equivalent to nonlinear programming problems. On the other hand, we recall that both the P_i and $\overline{BV}_i$ terms in (3) are triangular fuzzy numbers, for which there exist computationally efficient operators based on a shape preserving approximation of the extension principle. Hence, we can define Dino's fuzzy matchmaking process in terms of such approximated operators: given two triangular fuzzy numbers $T_1 = (l_1, c_1, r_1)$ and $T_2 = (l_2, c_2, r_2)$, we define fuzzy multiplication as

$$T_1 \otimes T_2 = (l_1, c_1, r_1) \otimes (l_2, c_2, r_2) = (l_1 \cdot l_2, c_1 \cdot c_2, r_1 \cdot r_2) \tag{4}$$

and fuzzy addition as

$$T_1 \oplus T_2 = (l_1, c_1, r_1) \oplus (l_2, c_2, r_2) = (l_1 + l_2, c_1 + c_2, r_1 + r_2) \tag{5}$$

which can be easily generalized to the summation operator $\bigoplus_{i=1}^{n}$. Notice that the addition operator in (5) is intrinsically shape preserving and provides the same results as the addition implemented through the extension principle. Conversely, the multiplication based on the extension principle does not result in a triangular fuzzy number, given that the triangle edges are polynomials of degree ≥ 2 instead of being linear functions. The shape preserving operator in (4) is essentially providing a linear approximation to such complex polynomial. The goodness of such approximation clearly depends on the degree of the polynomial. In the proposed model, we only need to approximate 2nd degree polynomials, given that the expression in (3) requires multiplication of only two fuzzy triangular number. In [8], it is shown how shape preserving multiplication is an acceptable approximation of the actual products for polynomials up to the 3rd degree.

Considering again our running example, Table 5 reports the ABV_j values and their corresponding Yager indexes. At this intermediate stage, the most favorite solution could be M_3 rather than M_2 as expected above. This fact comes with no surprise since response time has a much higher priority than availability. In fact, the ABV_j values would confirm M_2 being the favorite solution if response time and availability had the same priority.

To complete the fuzzyfication of the crisp matchmaking process described in Section 2, we need to integrate the fuzzy aggregated solution ABV_j with the contribution given by the *trustworthiness* estimate for j-th provider. By defining trustworthiness to be a triangular fuzzy set $TR_j = (trl_j, trc_j, trr_j)$, we

Table 6: Crisp Requirements and Priorities for Map Services

	Request	Priority
Response time	$[0, 2000]$	0.97
Availability	$[.94, 1]$	0.15

Table 7: Crisp Capabilities and Trustworthiness values of Map Services

	Resp. Time MaxVal	Avail. MinVal	Trustworth.
M_1	1400	.94	0.5
M_2	1200	.95	0.9
M_3	500	.92	0.5

can compute the final *appropriateness value* of the j-th provider as the triangular fuzzy set

$$AV_j = 0.5 \cdot (ABV_j \oplus TR_j)$$

Once that the appropriateness values AV_j have been computed for all providers, the ranking index in (2) can be used to determine an ordering of the solutions $AV_{min} \leq_{\mathcal{F}} \cdots \leq_{\mathcal{F}} AV_j \leq_{\mathcal{F}} \cdots \leq_{\mathcal{F}} AV_{max}$ where AV_{max} corresponds to the best matching service configuration.

Consider again the Map service example. Recalling that the provider trust rates are as in Table 3, the appropriateness values for the three map services are $Y_2(AV_1) = 0.30$, $Y_2(AV_2) = 0.49$, and $Y_2(AV_3) = 0.31$. Hence, M_2 is the solution selected by the provider.

Note that the algorithm described in this Section is a 'sound' generalization of the procedure presented in Section 2. Indeed, if we consider a dummy fuzzyfication of the data supplied to the original selection algorithm and we restrict to candidate services (i.e. services that satisfy both functional and QoS requirements), both procedures select the same service. Specifically, this fact holds because the similarity function $\overline{\mathrm{Sim}}_{OFM}(R_i, O_i)$ gives results that are in agreement with the formula for computing benefit values BV_i given in Section 2, when applied to crisp values that are expressed as fuzzy sets. For instance, consider a crisp counterpart of the above-mentioned Map service example whose input data are given in Tables 6 and 7. It can be seen that M_3 is not a candidate service as it does not match the availability requirements and that the original service selection method would also choose M_2. On the other side, remark that our fuzzy-valued generalization increases the flexibility of the Dino's approach. Indeed, by smoothing the intervals of admitted values, our technique can yield a broader class of candidate services and, hence, possibly select a service even if a solution cannot be found in a crisp setting.

4 Adaptive Service Specification

The runtime monitoring of the service being delivered is a fundamental step for ensuring client satisfaction, i.e. through the detection of violations of ser-

vice agreement, as well as for allowing the provider to best allocate and use its resources. This process entails measuring the actual QoS performance of the service dimensions, providing a suitable representation for the measurements that allows comparing the actual QoS values with the agreed service configuration. Again, fuzzy numbers seem a natural approach to provide a flexible and rich model of such performance values. Consider, for instance, to be able to perform a runtime monitoring of the service (whether this is done at the level of the client, provider or broker is of little importance at this point). Monitoring a given QoS dimension i will, in general, result in a sequence of performance measurements V_i^t across time t. A compact representation of the information described by such time series can be built by the descriptive statistics called *five-number summary*, consisting of

1. Min_i - the smallest observation in the series;
2. Q_{i1} - the lower/first quartile, i.e. the sample that is larger than exactly 25% of the observations;
3. M_i - the median observation, i.e. the second quartile;
4. Q_{i3} - the upper/third quartile, i.e. the sample that is larger than exactly 75% of the observations;
5. Max_i - the largest observation in the series.

From this five-number summary, we can build a fuzzy representation of the actual performance V_i^t by defining the corresponding trapezoidal fuzzy number as $RTM_i = (Min_i, Q_{i1}, Q_{i3}, Max_i)$. In other words, the measured QoS of the i-th attribute is defined as a fuzzy set whose support is determined by the maximum and minimum observed values, while the lower and upper quartile determine the core of the fuzzy number (see Fig. 3 for a graphical interpretation of a Normally-distributed population).

Such a representation can be used at the provider side to dynamically update the service offer, adapting it to the varying load conditions of the service host. Previously, it has been noted how a provider defines its service offer by specifying the QoS attributes by means of trapezoidal fuzzy numbers O_i. On the other hand, the runtime monitoring of the performance of the i-th dimension results in a fuzzy trapezoidal number RTM_i. Hence, the provider can check, either at runtime or a-posteriori, the degree of matching between the agreed service configuration and its actual instantiation, for instance by comparing O_i and RTM_i using the Jaccard similarity in (1). Several policies can be implemented at the provider side to dynamically update the offer: for instance, the provider might decide to keep the offered QoS for the i-th dimension unchanged until the similarity $Sim_J(RTM_i, O_i)$ falls under a given threshold θ, in which case it might set the new offered QoS $O_i^{new} = RTM_i$. More articulated policies can be envisaged, such as setting the new offered QoS to a fuzzy description of the mean performance across several service instantiations or selectively blocking

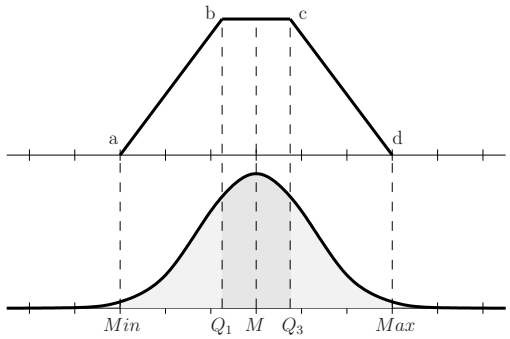


Figure 3: Trapezoidal fuzzy set built on quartiles of a Normally distributed population: minimum *min* and maximum *max* elements define the fuzzy number support (a and d points); lower (Q_1) and upper quartile (Q_3) identify the core (b and c points, respectively).

certain service configurations if the similarity check suggests that the provider is congested.

Considering again the Map scenario, assume that the provider of M_1 is supplying the services described in Table 3; suppose that the provider of M_1 monitors its *Response Time* and *Availability* across a given time interval T, collecting $N = 1000$ measurements. We have simulated such measurements by sampling response times from a Normal distribution $\mathcal{N}(m = 1500, \sigma^2 = 100)$ and by drawing availability from a uniform distribution $\mathcal{U}(.92, .96)$. By using the five number summary, the provider can build the fuzzy representation of the actual QoS measurements during T. Figure 4.a and 4.b compare the offered fuzzy QoS, described in Table 3, with the actual response times and availability, respectively. These plots suggest that the provider can trust its current offer for the availability dimension while it should adapt its response time offer, since it appears to be unable to deliver the offered QoS (e.g. the corresponding fuzzy sets have Jaccard similarity equal to 0.43).

A detailed and formal description of the policies for dynamic offer update is beyond the scope of this paper, since the choice of such rules is essentially a decision related to the internal policies of the provider. Here, we focus primarily on providing the tools and the descriptive power for implementing such policies within an organic framework. The approach described above can be effectively exploited within Dino's architecture to enforce the adaptivity of the broker, e.g. by identifying runtime violations of the contract terms or by defining an adaptive process for the dynamic update of providers trustworthiness. Assume, for instance, that the broker knows the fuzzy QoS measurements $RTM_1, \ldots, RTM_i, \ldots, RTM_n$ corresponding to a completed service between a client C_l and a provider P_j. Further, the broker knows the fuzzy QoS specification of such service agreement, e.g. $O_1, \ldots, O_i, \ldots, O_n$. Given these two sets of fuzzy numbers, it can compute the i-th client satisfaction with respect to

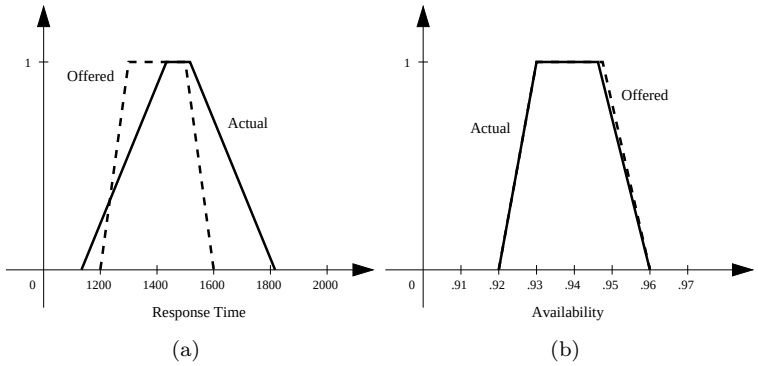


Figure 4: Trapezoidal fuzzy sets describing the offered (dashed) and the actual (solid) *Response Time* (a) and *Availability* (b) for M_1.

provider P_j as

$$SV_{ij} = (\frac{1}{n}) \bigoplus_{i=1}^{n} \overline{Sim}_{OFM}(RTM_i; O_i),$$

where $\overline{Sim}_{OFM}$ is the fuzzy-valued similarity and $\bigoplus$ is the fuzzy sum operator discussed previously. Indeed, a careful and clever application of this formula should ensure that the singleton $\tilde{1}_{OFM}$ is straightforwardly returned in place of $\overline{Sim}_{OFM}(RTM_i; O_i)$ whenever the provider is delivering a service where the actual QoS measurement RTM_i is clearly better that the offered QoS O_i, e.g. when the actual response time is strictly lower than the offered one. The term SV_{ij} resulting from the equation above is a triangular fuzzy number that can, for instance, be compared with the standard trustworthiness values in Table 1 to select the most similar trustworthiness class, e.g. the linguistic terms *High*, *Medium*, *Medium Low*, etc. For each delivered service, a provider receives an evaluation in terms of the closest trustworthiness class, accumulating a variable e_c counting the number of times it has been assigned to each class c (similarly to what happens with the customer feedback on E-Bay). Finally, each provider can be assigned a fuzzy trustworthiness weight corresponding either to the fuzzy set of the prevalent class from Table 1 (e.g. TR_j = MediumHigh) or based on a weighted average of the evaluations received so far, i.e.

$$TR_j = \bigoplus_{c=1, e_c>0} \frac{e_c}{E} \cdot \mathrm{TC_c}$$

where TC_c is the triangular fuzzy set corresponding the c-th trustworthiness class and $E = \sum_c e_c$. For instance, Fig. 5 shows the SV values computed for provider M_1 using the simulated response time and availability, as well as the cumulative trustworthiness value. Since actual availability matches closely the offered availability, then the corresponding SV value shows a *Very High* trustworthiness (see the linguistic terms definition in Table 1); on the other

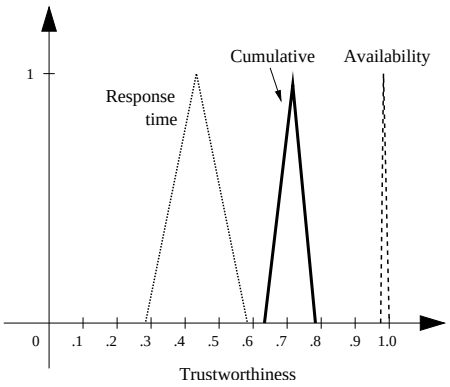


Figure 5: Fuzzy valued client satisfaction with respect to provider M_1: the dotted triangle refer to *Response Time* (with Jaccard index 0.43), the dashed triangle to *Availability* (with Jaccard index 0.986) and the solid one to the cumulative satisfaction.

hand, the discrepancy between the offered and the actual response time yields the corresponding SV value to be associated to a *Medium* trustworthiness. As a result, provider M_1 is assigned a cumulative performance evaluation that is closer to the *Medium High* linguistic term.

5 Conclusion

In this paper we have introduced a QoS-aware service selection procedure that exploits fuzzy set theory both for specifying QoS requirements and capabilities and for ranking the acceptable solutions by means of an ordering relation over fuzzy quantities. Technically, we have extended an existing approach named Dino to express preferences and vague or imprecise QoS capabilities/requirements as fuzzy sets, thus achieving a more flexible service selection mechanism. In order to make our method effective in practice, we plan to integrate the prototype implementation of our fuzzy-valued method into the Dino framework. On the other side, our method is quite independent of the choice of crisp QoS-aware algorithm, hence we expect that our procedure can be adapted to generalize other similar techniques.

In future works, it would be interesting to study this issue deeply, by providing a general and flexible fuzzy matchmaking framework and by showing how it can be tailored to the specificities of the main crisp matchmaking approaches in literature. Another point worth noting is that the Yaeger index we use for ranking the proposals can be computed only for normal fuzzy sets. We intend to investigate whether a suitable alternative notion of ranking index can be defined to account for 'subnormal' fuzzy sets. Finally, we have to note that, in our effort to design a fully fuzzyfied matchmaking procedure, we have given little consideration to a parameter that in the Dino framework is called confidence.

Such a parameter is a probabilistic measure of the confidence regarding the QoS measurements at the providers side. The model described in the paper considers confidence as a crisp parameter. On the other hand, the modern fuzzy logic offers a powerful means for incorporating such uncertainty within the fuzzy description of the offered QoS, that is *Type II fuzzy sets* [13, 18]. These are fuzzy sets that map each element of the crisp domain to a fuzzy number bounded in the range $[0, 1]$, rather than to a crisp value in $[0, 1]$. This representation allows to straightforwardly incorporate the confidence information within such map. In our future work, we plan to extend further our model to deal with such type II fuzzy sets and fuzzy confidence intervals.

Acknowledgment

The authors would like to thank Alessio Botta for providing the source code for computing the fuzzy-valued similarity measure.

References

[1] D. Ardagna and B. Pernici. Global and local QoS guarantee in web service selection. In *Proc. of Business Process Managment Workshop*, pages 32–46, 2005.

[2] D. Bacciu, A. Botta, and H. Melgratti. A fuzzy approach for negotiating quality of services. In *Proc. of TGC 2006*, volume 4661 of *Lecture Notes in Computer Science*, pages 200–217. Springer, 2007.

[3] D. Bacciu, A. Botta, and D. Stefanescu. A framework for semantic querying of distributed data-graphs via information granules. In *Proc. of ISC 2007*. Acta Press, 2007.

[4] G. Canfora, M. D. Penta, R. Esposito, and M. Villani. An approach for qos-aware service composition based on genetic algorithms. In *Proc. of Genetic and Evolutionary Computation Conference*, 2005.

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

[6] D. Dubois and H. Prade. *Fuzzy Sets and Systems - Theory and Applications*. Academic Press, NY, 1980.

[7] D. Dubois and H. Prade. A unifying view of comparison indices in a fuzzy set-theoretic framework. In *Recent developments in fuzzy set and possibility theory*, pages 3–13. Pergamon, 1982.

[8] R. E. Giachetti and R. E. Young. Analysis of the error in the standard approximation used for multiplication of triangular and trapezoidal fuzzy numbers and the development of a new approximation. *Fuzzy Sets and Systems*, 91(1):1 – 13, 1997.

[9] C. L. Huang, C. C. Lo, Y. Li, K. M. Chao, J. Chung, and Y. Huang. Service discovery through multi-agent consensus. In *Proc. of IEEE Int. Workshop on Service-Oriented Syst. Eng.*, pages 37–44, 2005.

[10] N. Koch and D. Berndl. Sensoria deliverable 8.2a: Requirements modelling and analysis of selected scenarios - automotive case study. Technical report, FAST, 2007.

[11] M. Lin, J. Xie, H. Guo, and H. Wang. Solving QoS-driven web service dynamic composition as fuzzy constraint satisfaction. In *Proc. of IEEE Int. Conf. on e-Technology, e-Commerce and e-service*, pages 9–16, 2005.

[12] Y. Liu, A. Ngu, and L. Zeng. QoS computation and policing in dynamic web service selection. In *Proc. of the 13th International World Wide Web Conference*, 2004.

[13] J. Mendel and R. John. Type-2 fuzzy sets made simple. *IEEE Trans. on Fuzzy Syst.*, 10(2):117–127, Apr 2002.

[14] L. Mikhailov and P. Tsvetinov. Evaluation of services using a fuzzy analytic hierarchy process. In *Applied Soft Computing 5*, pages 23–33, 2004.

[15] A. Mukhija, A. Dingwall-Smith, and D. S. Rosenblum. QoS-aware service composition in dino. In *Proc. of ECOWS 2007*, pages 3–12. IEEE Comp. Society, 2007.

[16] X. T. Nguyen, R. Kowalczyk, and M. T. Phan. Modelling and solving QoS composition problem using fuzzy discsp. In *Proc. of IEEE ICWS 2006*, pages 55–62, 2006.

[17] X. Wang and E. E. Kerre. Reasonable properties for the ordering of fuzzy quantities. *Fuzzy Sets Systems*, 118(3):375–405, 2001.

[18] D. Wu and J. M. Mendel. Uncertainty measures for interval type-2 fuzzy sets. *INFORMATION SCIENCES*, 177(23):5378–5393, 2007.

[19] Z.-S. Xu and J. Che. An interactive method for fuzzy multiple attribute group decision making. *Information Sciences*, 177(1):248 – 263, 2007.

[20] X.Wang, T. Vitvar, M. Kerrigan, and I. Toma. A QoS-aware selection model for semantic web services. In *Proc. of ICSOC 2006*, 2006.

[21] R. Yager. On choosing between fuzzy subsets. *Kibernetes*, 9:151–154, 1980.

[22] L. Zadeh. Fuzzy Sets. *Information and Control*, 3(8):338–353, 1965.