Selecting and Ranking Business Processes with Preferences: An Approach Based on Fuzzy Sets

Katia Abbaci[1], Fernando Lemos[2], Allel Hadjali[1], Daniela Grigori[2],
Ludovic Liétard[3], Daniel Rocacher[1], and Mokrane Bouzeghoub[2]

[1] IRISA/ENSSAT, Rue de Kérampont BP 80518 Lannion, France
{katia.abbaci,allel.hadjali,daniel.rocacher}@enssat.fr
[2] PRiSM Lab, 45 Av. des États Unis 78000 Versailles, France
{fernando.lemos,daniela.grigori,mokrane.bouzeghoub}@prism.uvsq.fr
[3] IRISA/IUT, Rue Edouard Branly BP 30219 Lannion, France
ludovic.lietard@univ-rennes1.fr

Abstract. Current approaches for service discovery are based on semantic knowledge, such as ontologies and service behavior (described as process model). However, these approaches still remain with a high selectivity rate, resulting in a large number of services offering similar functionalities and behavior. One way to improve the selectivity rate and to provide the best suited services is to cope with user preferences defined on quality attributes. In this paper, we propose and evaluate a novel approach for service retrieval that takes into account the service process model and relies both on preference satisfiability and structural similarity. User query and target process models are represented as annotated graphs, where user preferences on QoS attributes are modelled by means of fuzzy sets. A flexible evaluation strategy based on fuzzy linguistic quantifiers (such as *almost all*) is introduced. Then, two families of ranking methods are discussed. Finally, an extensive set of experiments based on real data sets is conducted, on one hand, to demonstrate the efficiency and the scalability of our approach, and on the other hand, to analyze the effectiveness and the accuracy of the proposed ranking methods compared to expert evaluation.

Keywords: web service retrieval, quality of services, preferences, fuzzy set theory, linguistic quantifier.

1 Introduction

Searching for a specific service within service repositories become a critical issue for the success of service oriented and model-driven architectures and for service computing in general. This issue has recently received considerable attention and many approaches have been proposed. Most of them are based on the matchmaking of process input/outputs [1], service behavior (described as process model) [2,3,4] or ontological knowledge [4]. However, these approaches have high selectivity rate, resulting in a large number of services offering similar functionalities and behavior [4].

R. Meersman, T. Dillon, and P. Herrero (Eds.): OTM 2011, Part I, LNCS 7044, pp. 38–55, 2011.

One way to discriminate between similar services is to consider non-functional requirements such as quality preferences (response time, availability, etc.). Indeed, for a given query in a given context, there is no need to provide all possible services but only those satisfying user preferences and contextual constraints. A recent trend towards quality-aware approaches has been initiated [5,6,7], but it is limited to atomic services. Our goal is to go further these approaches into a unique integrated approach dealing with functional and non-functional requirements in service retrieval. Targeting this goal poses the following challenges: (i) At the description level, provide a model allowing to specify non-functional requirements at different granularity levels of the service functional description; (ii) At the discovery level, define an evaluation method that efficiently computes the satisfiability of a target service w.r.t. the functional and non-functional requirements of a user query.

More specific challenges related to non-functional characteristics should also be taken into account: (i) Services are deployed over dynamic and heterogeneous environments such that their non-functional properties are often given or derived with different accuracies; (ii) Users are not always able to precisely specify their non-functional constraints; (iii) Users have different points of view over what is a satisfactory service according to the same set of non-functional constraints; (iv) The service retrieval should avoid empty or overloaded answers due to the imprecision of the user's query.

Preferences are a natural way to facilitate the definition of non-functional constraints in user query. They are flexible enough, on one hand, to avoid empty returns caused by very strict user constrains and, on the other hand, to provide an adequate set of relevant results even when user specifies too general constraints. In addition, fuzzy logic has been used as a key technique to take into account human point of view in preference modelling and evaluations [8].

In [9], it is proposed a QoS-aware process discovery method whereas the user query is a graph annotated with QoS factors. Starting from [9], this paper investigates a novel approach for service selection and ranking taking into account both behavior specification and QoS preferences. User query and target process models are represented as graphs, where queries are annotated with preferences on QoS properties and targets are annotated with QoS attributes. Preferences are represented by means of fuzzy sets as they are more suitable to the interpretation of linguistic terms (such as *high* or *fast*) that constitute a convenient way for users to express their preferences. To avoid empty answers for a query, an appropriate flexible evaluation strategy based on fuzzy linguistic quantifiers (such as *almost all*) is introduced.

In the remainder of this paper, Section 2 provides some basic background and discusses related works. Section 3 describes process model specification with preferences. Section 4 addresses fuzzy preference modelling and evaluation. Section 5 presents our interpretation of process models similarity based on linguistic quantifiers. Section 6 discusses service ranking methods. Section 7 proposes an illustrative example and Section 8 presents a set of experiments conducted to evaluate our approach. Finally, Section 9 concludes the paper.

2 Background and Related Work

Here, we recall some notions on preference modelling (e.g., Pareto and fuzzy set based models) and we review preference-based service discovery approaches.

2.1 Preference Modelling

The semantics of preferences assumed in this work is the one provided by the databases area: preferences are used to help in reducing the amount of information returned in response to user queries and to avoid the happening of empty answers. Generally, two families of approaches can be distinguished to model preferences. The first one relies on commensurability assumption which leads to a total pre-order [10,11,8]. We highlight the *SQLf* proposal [11], which is based on the extension of the relational algebra to fuzzy set theory. The second family assumes that commensurability does not hold, in this case no compensation is allowed between criteria and only a partial order is obtained [12,13,14].

One popular approach of this last family is *Preference SQL* [13]. It provides foundations for a *Pareto*-based preference model for database systems. A preference is formulated as a strict partial order on a set of attribute values. It introduces a number of preference operators to express and compose preferences. Let us note that all tuples returned by a Preference SQL query satisfy the *Pareto* principle. A compensatory strategy between different atomic conditions is not possible due to the fact that Preference SQL makes use of different functions for evaluating the distance with which a tuple disagrees with an atomic condition. Moreover, the most preferred tuples are returned to the user without being capable to distinguish how better is one tuple compared to another.

Fuzzy sets were introduced in [15] for dealing with the representation of classes or sets whose boundaries are not well defined. Then, there is a gradual transition between the full membership and the full mismatch (an order relation on membership levels can be established). Typical examples of such fuzzy classes are those described using adjectives of the natural language, such as *cheap*, *fast*, etc. Formally, a fuzzy set F on the universe X is described by a membership function $\mu_F : X \rightarrow [0,1]$, where $\mu_F\left(x\right)$ represents the **membership degree** of x in F. By definition, if $\mu_F\left(x\right) = 0$ then the element x **does not belong at all** to the fuzzy set F, if $\mu_F\left(x\right) = 1$ then x **fully belongs** to F. When $0 < \mu_F(x) < 1$, one speaks of **partial membership**. The set $\{x \in F | \mu_F(x) > 0\}$ represents the **support** of F and the set $\{x \in F | \mu_F(x) = 1\}$ represents its **core**.

In addition, the closer $\mu_F\left(x\right)$ to the value 1, the more belonging to F. Therefore, given $x, y \in F$, one says that x is preferred to y iff $\mu_F(x) > \mu_F(y)$. If $\mu_F(x) = \mu_F(y)$, then x and y are equally preferred. In practice, the membership function associated to F is often represented by a *trapezoid* $(\alpha,\ \beta,\ \varphi,\ \psi)^1$, where $[\alpha, \psi]$ is its support and $[\beta, \varphi]$ is its core. Among other forms (Gaussian, sigmoidal, bell, etc), this one is very easy to be defined and to manipulate.

[1] In our case, the quadruplet $(\alpha, \beta, \varphi, \psi)$ is user-defined to ensure the subjectivity property.

A fuzzy set-based approach to preference queries proposed in [8] is founded on the use of fuzzy set membership functions that describe the preference profiles of the user on each attribute domain involved in the query. This is especially convenient and suitable when dealing with numerical domains, where a continuum of values is to be interfaced for each domain with satisfiability degrees in the unit interval scale. Then satisfiability degrees associated with elementary conditions are combined using fuzzy set connectives, which may go beyond conjunctive and disjunctive aggregations (by possibly involving fuzzy quantifiers, if the satisfiability of most of the elementary conditions in a query is required).

2.2 Preference-Based Service Discovery

Crisp Logic-based Approaches. Most of the first approaches for service discovery using preferences were based on crisp logic solution and considered the services as black boxes [16,6,17]. With regard to the specification model, some of them do not deal with preferences; instead, they compute for each service a score based on set of the non-functional properties of the service [16]. The other approaches does not propose or use preference constructors to help user better define his preferences or interpret the results [6,17]. The models presented are not abstract enough to provide a widely use of the approach in different contexts; some of them imposes a restricted set of properties over which user can work.

Fuzzy Logic-based Approaches. In last decades, several service discovery approaches based on fuzzy set theory have been proposed [18,19]. In [19] the authors treat the web service selection for composition as a fuzzy constraint satisfiability problem. They assign to each QoS criterion five fuzzy sets describing its constraint levels. In [20,21], QoS based service selection is modelled as a fuzzy multiple criteria decision making problem. In [22], a service selection mechanism is presented allowing the service broker to select a set of services from a query specifying imprecise constraints defined by fuzzy sets. The query evaluation is based on the aggregation of the obtained degrees over constraints. Şora et al. [5] propose an approach to automatically generate fuzzy rules from user preferences and rank the candidate services using a fuzzy inference process. The global score of each web service is given in a scale of satisfiability levels instead of an aggregation of the satisfiability degrees of the preferences.

The aforementioned fuzzy approaches take into account only the satisfiability of preferences whereas they ignore the structural similarity of web services. Most of them do not verify the *subjectivity property*, which considers the user point of view when defining the membership functions. Moreover, these works deal only with services as black boxes. In this paper, user can also define preferences over the activities of the service behavior specification and both structural similarity and user preference satisfiability are considered.

3 Preferences in Process Model Specification

Many languages are currently available to describe service process models, e.g., WS-BPEL and OWL-S. They represent a process model as a set of atomic

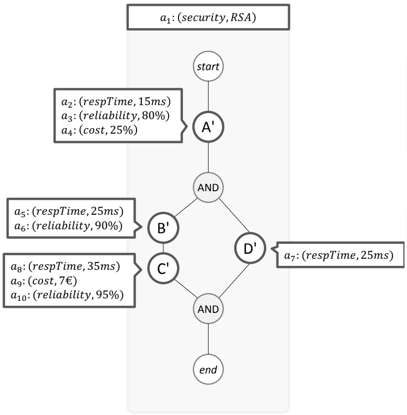

a_1: $(security, RSA)$
start
a_2: $(respTime, 15ms)$
a_3: $(reliability, 80\%)$
a_4: $(cost, 25\%)$
A'
AND
a_5: $(respTime, 25ms)$
a_6: $(reliability, 90\%)$
B'
D'
a_7: $(respTime, 25ms)$
a_8: $(respTime, 35ms)$
a_9: $(cost, 7€)$
a_{10}: $(reliability, 95\%)$
C'
AND
end

Fig. 1. Target Graph t_1

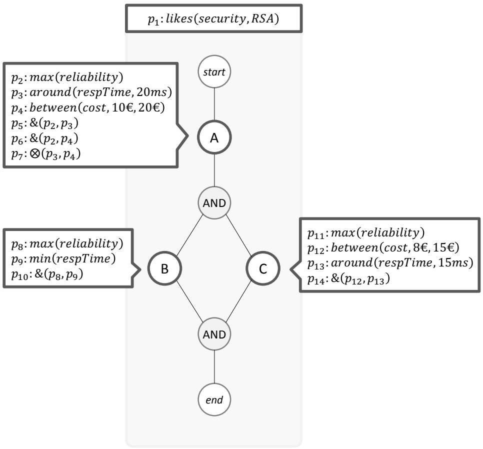

p_1: $likes(security, RSA)$
start
p_2: $max(reliability)$
p_3: $around(respTime, 20ms)$
p_4: $between(cost, 10€, 20€)$
p_5: $\&(p_2, p_3)$
p_6: $\&(p_2, p_4)$
p_7: $\otimes(p_3, p_4)$
A
AND
p_8: $max(reliability)$
p_9: $min(respTime)$
p_{10}: $\&(p_8, p_9)$
B
C
p_{11}: $max(reliability)$
p_{12}: $between(cost, 8€, 15€)$
p_{13}: $around(respTime, 15ms)$
p_{14}: $\&(p_{12}, p_{13})$
AND
end

Fig. 2. Query Graph q_1

activities combined using control flow structures. As a consequence, these languages can be abstracted as a direct graph $G = (V, E)$, where the vertices represent activities (e.g., *hotel reservation*, *payment*) or control flow nodes (e.g., *and*, *or*), while the edges represent the flow of execution between activities.

In this work, services are specified as graphs annotated with QoS properties and user queries are specified as graphs annotated with preferences. Figure 1 presents a global annotation indicating the security of the process model and activity annotations indicating other QoS attributes of some activities. Figure 2 shows a sample user query annotated with a global preference indicating user prefers services providing RSA encryption and some activity preferences involving reliability, response time and cost. It is worth mentioning that our model can be implemented by extension mechanisms in OWL-S.

We precise that, in this work, target models are considered already annotated with QoS attributes while the user is the one to define the preference annotations of his query. Techniques to obtain the QoS information of a process model can be found in [23]. Next, we present the formal definitions of our model:

Definition 1. An **annotation** is a pair (m, r), where m is a QoS attribute and r is a value for m[2]. It can be specified over a process model graph (**global annotation**) or over an atomic activity (**activity annotation**).

Definition 2. A **preference** is an expression that represents a desire of the user over the QoS attributes of a process model or activity. It can be specified over a process model graph (**global preference**) or over an atomic activity (**activity preference**). It can be of one the following forms[3]:

[2] We abstract from the different units in which a value can be described.

[3] Based on a subset of preferences defined in [13].

- *around* $(m, r_{desired}, \mu_{around})$: it favors the value $r_{desired}$ for attribute m; otherwise, it favors those close to $r_{desired}$. The membership function μ_{around} evaluates the degree to which a value r satisfies $r_{desired}$;
- *between* $(m, r_{low}, r_{up}, \mu_{between})$: it favors the values inside the interval $[r_{low}, r_{up}]$; otherwise, it favors the values close to the limits. The function $\mu_{between}$ evaluates the degree to which a value r satisfies the interval $[r_{low}, r_{up}]$;
- *max* (m, μ_{max}): itfavors the highest value; otherwise, the closest value to the maximum is favored. For example, the maximum of availability is equal by default to 100%. The function μ_{max} evaluates the degree to which a value r satisfies the highest value of m;
- *min* (m, μ_{min}): it favors the lowest value; otherwise, the closest value to the minimum is favored, as example: the minimum of response time or cost is equal by default to 0. μ_{min} evaluates to which degree a value r satisfies the lowest value of m;
- *likes* $(m, r_{desired})$: it favors the value $r_{desired}$; otherwise, any other value is accepted;
- *dislikes* $(m, r_{undesired})$: it favors the values that are not equal to $r_{undesired}$; otherwise, $r_{undesired}$ is accepted;
- Pareto $\otimes$ (p_i, p_j): it states that the two soft preference expressions p_i and p_j are equally important;
- Prioritized & (p_i, p_j): it states that the soft preference expression p_i is more important than the soft preference expression p_j.

The work in [13] distinguishes two types of preferences: *atomic* (*around*, *between*, *max*, *min*, *likes* and *dislikes*) and *complex* ($\otimes$ and &). It also distinguishes two types of atomic preferences: *numerical* (*around*, *between*, *max* and *min*) and *non-numerical* (*likes* and *dislikes*). The values in non-numerical preferences are taken from a global ontology of a type "is-a" O, given by the user.

4 A Fuzzy Model to Evaluate Preferences

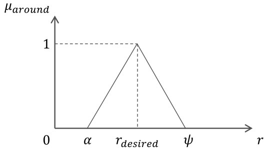
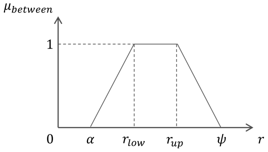
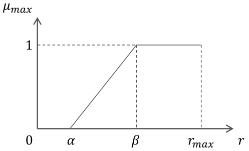
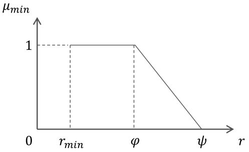
Here, we introduce a fuzzy semantics of the atomic preferences discussed in the Section 3, and show how they can be evaluated. In particular, we propose a metric, called *satisfiability degree* (δ), that measures how well a set of annotations of a target process model satisfies a set of preferences present in the query. The computation of this degree is done both for atomic and complex preferences.

4.1 Atomic Preferences

For numerical atomic preferences, the satisfiability degree is obtained thanks to user-specific membership functions. Table 1 summarizes the fuzzy modelling of numerical preferences of interest. Given a preference p and an annotation $a : (m, r)$, one is interested in computing the degree to which the annotation a satisfies the fuzzy characterization underlying p.

For example, consider the constructor *between*: a fuzzy preference p : *between* (m, r_{low}, r_{up}) is characterized by the membership function $(\alpha, \beta, \varphi, \psi)$, where

Table 1. Fuzzy modelling of numerical preferences

NUMERICAL PREFERENCE	FUZZY INTERPRETATION
$around(m, r_{desired}, \mu_{around})$	$\mu_{around}(r) = \begin{cases} 0, & r \leq \alpha, r \geq \psi \\ \frac{r-\alpha}{r_{desired}-\alpha}, & \alpha < r < r_{desired} \\ 1, & r = r_{desired} \\ \frac{\psi-r}{\psi-r_{desired}}, & r_{desired} < r < \psi \end{cases}$ 
$between(m, r_{low}, r_{up}, \mu_{between})$	$\mu_{between}(r) = \begin{cases} 0, & r \leq \alpha, r \geq \psi \\ \frac{r-\alpha}{r_{low}-\alpha}, & \alpha < r < r_{low} \\ 1, & r_{low} \leq r \leq r_{up} \\ \frac{\psi-r}{\psi-r_{up}}, & r_{up} < r < \psi \end{cases}$ 
$max(m, \mu_{max})$	$\mu_{max}(r) = \begin{cases} 0, & r \leq \alpha \\ \frac{r-\alpha}{\beta-\alpha}, & \alpha < r < \beta \\ 1, & \beta \leq r \leq r_{max} \end{cases}$ 
$min(m, \mu_{min})$	$\mu_{min}(r) = \begin{cases} 1, & r_{min} \leq r \leq \varphi \\ \frac{\psi-r}{\psi-\varphi}, & \alpha < r < \psi \\ 0, & r \geq \psi \end{cases}$ 

$\beta = r_{low}$; $\varphi = r_{up}$; α and ψ are two values from the universe X. Let $a : (m, r)$ be an annotation of a target graph, the satisfiability degree of preference p according to a is given by: (i) p is completely satisfied iff $r \in [r_{low}, r_{up}]$: $\mu_{between}(p, a) = 1$, i.e. $\delta(p, a) = 1$; (ii) the more r is lower (resp. higher) than r_{low} (resp. r_{up}), the less p is satisfied: $0 < \mu_{between}(p, a) = \delta(p, a) < 1$; (iii) for $r \in]-\infty, \alpha] \cup [\psi, +\infty[$, p is not satisfied: $\mu_{between}(p, a) = \delta(p, a) = 0$.

For non-numerical atomic preferences, the satisfiability degree is based on the semantic similarity between concepts. We applied the widely known semantic similarity proposed in [24], which states that given an ontology O and two concepts c_1 and c_2, the semantic similarity wp between c_1 and c_2 is given by $wp(O, c_1, c_2) = {}^{2N_3}/_{N_1+N_2+2N_3}$, where c_3 is the least common super-concept of c_1 and c_2, N_1 is the length of the path from c_1 to c_3, N_2 is the length of the path from c_2 to c_3, and N_3 is the length of the path from c_3 to the root of the ontology. Given a non-numerical atomic preference p and an annotation a, the satisfiability degree $\delta(p, a)$ is given by:

- If $p = likes(m, r_{desired})$, then $\delta(p, a) = \begin{cases} 1, & r_{desired} = r \\ wp(O, r_{desired}, r), & otherwise \end{cases}$
- If $p = dislikes(m, r_{undesired})$, then $\delta(p, a) = 1 - \delta(likes(m, r_{undesired}), a)$

One can use other semantic similarity measures between business processes [25,26]. This issue is not discussed here and it is beyond the scope of this study.

4.2 Complex Preferences

To compute the satisfiability degree of complex preferences, we first construct a *preference tree* t_p that represents the complex preference structure of a set of preferences S_p. In that preference tree, the nodes represent atomic preferences and the edges represent a *more important than* relation (*prioritized preference*, denoted by &) from parent to child. Preferences belonging to the same level and having the same parent express *Pareto preference*, denoted by $\otimes$. Each level i of the tree is associated with a weight $\omega_i = {}^1\!/\!_i$ except the $level0$.

For example, consider the preference tree of q_1 in Figure 3. Preference p_{11} is an atomic preference that is not component of any complex preference. p_5 : $\& \, (p_2, p_3)$ is a complex preference composed of preferences p_2 and p_3; it means that p_2 is more important than p_3. $p_7 : \otimes \, (p_3, p_4)$ is a complex preference composed of preferences p_3 and p_4; it means that p_3 and p_4 are equally important.

Considering that each atomic preference p_i has a satisfiability degree δ_i, a new satisfiability degree δ'_i is computed taking into account the weight ω_i underlying p_i in the spirit of [8]. δ'_i is defined using the formula (1) (we assume that $\max_{i=1,n} w_i = 1$).

$$\delta'_i = \max\left(\delta_i, 1 - \omega_i\right) \tag{1}$$

This new interpretation of p_i considers as acceptable any value outside of its support with the degree $1-\omega_i$. It means that the larger ω_i (i.e., p_i is important), the smaller the degree of acceptability of a value outside the support of p_i. At the end, we have calculated the satisfiability degree of user atomic preferences considering their constructors and the complex preferences composing them.

5 Process Model Similarity: A Linguistic Quantifier-Based Method

We describe here a method to compute preference satisfiability between process model graphs. We also discuss a method to assess the structural similarity

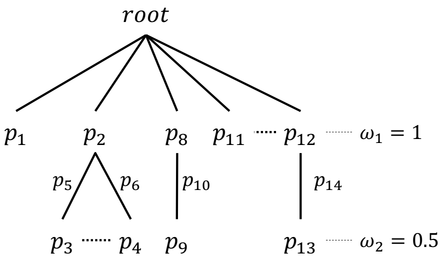


Fig. 3. Sample preference tree

between two process model graphs. Both degrees will be used to rank potential targets (see Section 6). We precise that this work is not interested in discovering a mapping between two process models; we suppose a mapping already exists such that we can compare matched activities annotations against user preferences. In this issue, please consider the work in [4] for an algorithm that returns a mapping between two process models.

To evaluate the structural similarity of two graphs q and t, we propose to use a graph matching algorithm like in [4]. This algorithm returns a mapping M and a set E of edit operations necessary to transform q into t. A mapping between q and t is a set of pairs (v, w), such that v is an activity of q and w is an activity of t. The edit operations considered are simple graph operations: node/edge deletion and addition. Figure 4 illustrates a mapping between query graph q_1 and target graph t_1. Let $SS\,(v, w)$ denotes the structural similarity between activities v and w; we use the metric proposed in [4]. Let $\delta\,(q_1.S_p, t_1.S_a)$ be the satisfiability degree between global preferences and annotations and let $\delta\,(v, w)$ be the satisfiability degree between activities v and w (see Section 4).

Next, we rely on the linguistic quantifier *"almost all"* for the similarity evaluation process. This quantifier is a relaxation of the universal quantifier "*all*" and constitutes an appropriate tool to avoid empty answers since it retrieves elements that would not be selected when using the quantifier "*all*".

5.1 Preference Satisfiability between Process Models

A natural user interpretation of the similarity between query and target PMs according to preferences is given by the truth degree of the following proposition:

γ_1: Almost all preferences of q are satisfied by t

The above statement is a fuzzy quantified proposition of the form "$Q\ X\ are\ P$", where (i) Q is a relative quantifier (e.g., *almost all*, *around half*, etc.) [27] which is defined by a function μ_Q such as $\mu_Q\,(\varpi)$ is the degree of truth of "$Q\ X\ are\ P$" when a proportion ϖ of elements of X fully satisfy A and the other elements being not satisfied; (ii) X is a set of elements; (iii) P is a fuzzy predicate. In [28], a decomposition method to compute the truth degree δ_γ of $\gamma : Q\,X\,are\,P$ is proposed. The method is a two-step procedure:

- Let $\Omega = \{\mu_1, \ldots, \mu_n\}$ be a set of degrees of the elements of X w.r.t. P, ordered in decreasing way; i.e. $\mu_1 \geq \ldots \geq \mu_n$;
- The truth degree δ_γ is given by the equation (2), where $\mu_Q\,({}^{i}/_{n})$ is a membership degree of the element ${}^{i}/_{n}$ to Q.

$$\delta_\gamma = \max_{1 \leq i \leq n} \min\left(\mu_i, \mu_Q\left({}^{i}/_{n}\right)\right) \tag{2}$$

In our case, $\Omega = \left\{\mu_1 : \delta_1^{'}, \ldots, \mu_n : \delta_n^{'}\right\}$ is the set of satisfiability degrees of all (global and activity) atomic preferences of query q, where $\delta_i^{'}$ is the satisfiability degree of an atomic preference p_i computed by formula (1). The semantics of

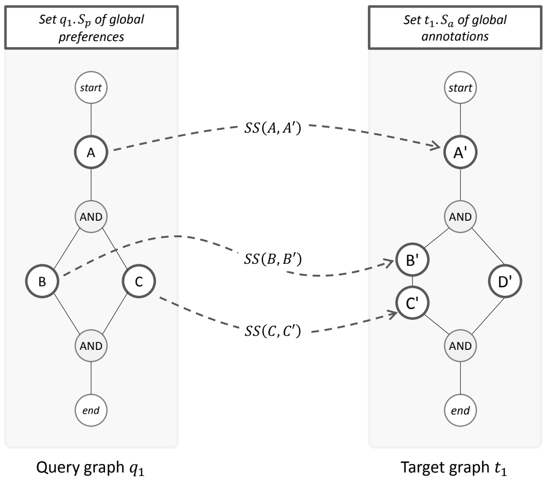


Fig. 4. Sample mapping M between query graph q_1 and target graph t_1

the linguistic quantifier *almost all* is given in Table 2. In this case, (i) the user is totally satisfied if at least 80% of preferences are satisfied and (ii) the user is not satisfied at all if at most 50% of preferences are satisfied.

5.2 Structural Similarity between Process Models

Similarly, we can apply the technique of fuzzy quantifiers to obtain a structural similarity degree between two process models. The structural similarity between a query and target process models can be given by the truth degree of the following propositions "γ_1, γ_2 and γ_3" (defined in Table 2):

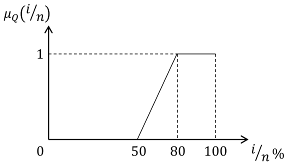
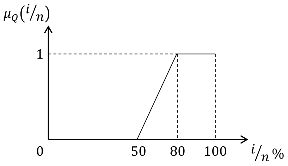
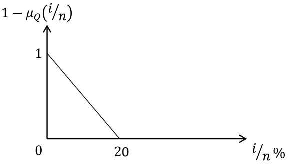
γ_2: Almost all the activities of q are mapped with activities of t, and
γ_3: Almost no edit operation is necessary to transform q into t

The truth degree of proposition γ_2 is obtained from the formula (2), where $\Omega = \{\mu_1 : SS_1, \ldots, \mu_n : SS_n\}$ is the set of semantic similarity degrees of all mapped activities of q, and SS_i is the semantic similarity degree of a query activity v mapped with a target activity w. In the case of the proposition γ_3, the expression "almost no edit operation is necessary to transform q into t" is equivalent to the expression "*almost all* edit operations are *not* necessary to transform q into t". Therefore, its truth degree is computed as follows:

$$\delta_\gamma = \max_{1 \leq i \leq n} \min \left(1 - \mu_i, 1 - \mu_Q \left({}^{i}\!/\!_{n}\right)\right) \tag{3}$$

In this case, $\Omega = \{\mu_1 : C_1, ..., \mu_n : C_n\}$ is the set of transformation costs of mapped target activities with the corresponding activities of q, and C_i is the transformation cost of a target activity w into a query activity v.

Table 2. Decomposition-based interpretations of propositions γ_1, γ_2, γ_3

PROPOSITION	SET X	MEMBERSHIP FUNCTION $\mu_Q(^i/_n)$
γ_1	$X = \{\mu_1 : \delta(v_1, w_1), \ldots, \mu_n : \delta(v_n, w_n)\}$, where n is the number of mapping elements	
γ_2	$X = \{\mu_1 : SS(v_1, w_1), \ldots, \mu_n : SS(v_n, w_n)\}$, where n is the number of mapping elements	
γ_3	$X = \{\mu_1 : C(v_1), \ldots, \mu_n : C(v_n)\}$, where n is the number of query activities	

So, the structural similarity between q and t is evaluated as follows:

$$SS = \min\left(\delta_{\gamma_2}, \delta_{\gamma_3}\right) \tag{4}$$

In our approach, we consider particularly the formulae (2) and (3), where $\mu_Q\left(^i/_n\right) = {}^i/_n$. Thus, the meaning of delivered degrees has a simple and clear semantics for the user [29]. The evaluation of γ_1, γ_2 and γ_3 means that:
"At least $\delta^*_{\gamma_1}$ % of preferences of q are satisfied by t to at least a degree of δ_{γ_1}, at least $\delta^*_{\gamma_2}$ % of the activities of q are mapped with t to at least a degree of δ_{γ_2}, and at least $\delta^*_{\gamma_3}$ % of q does not need edit operation to transform q into t to at least a degree of δ_{γ_3}" (where $\delta^*_{\gamma_i} = 100 \times \delta_{\gamma_i}$).

6 Process Model Ranking

Previous section has presented an fuzzy set-based approach to compute the similarity between one query and one target graphs. In this section, given a set of target graphs that are relevant to the query, we discuss some methods to rank-order these graphs according to their structural and preference similarities. Let $\delta\left(q, t, M\right)$ be the satisfiability degree between query graph q and target graph t according to a mapping M. Similarly, let $SS\left(q, t, M, E\right)$ be the structural similarity between q and t according to a mapping M and a set E of edit operations. We classify ranking methods into two categories:

Ranking Methods based on Aggregation. In this first category, ranking methods aggregate both structural and preference similarities into a unique degree used to rank-order the target graphs. Two kind of aggregations are considered:

Weighted Average-Based Aggregation. The weighted average of $SS\left(q,t,M,E\right)$ and $\delta\left(q,t,M\right)$ is given by:

$$rank\left(q,t\right) = \omega_{SS} \times SS\left(q,t,M,E\right) + \left(1-\omega_{SS}\right) \times \delta\left(q,t,M\right) \tag{5}$$

where $0 < \omega_{SS} < 1$ is a weight assigned to the structural similarity criterion.

Min-Combination Based Aggregation. The min-combination method [30] selects the smallest value of the two similarity degrees $SS\left(q,t,M,E\right)$ *and* $\delta\left(q,t,M\right)$:

$$rank\left(q,t\right) = \min\left(SS\left(q,t,M,E\right), \delta\left(q,t,M\right)\right) \tag{6}$$

Ranking Method without Aggregation. The two distinct similarity degrees are used to rank-order target graphs. The answers are ranked by using the *lexicographic order*. A priority is given to the structural similarity while the preference similarity is only used to break ties.

7 Illustrative Example

We give here an example of service discovery for query q_1 of Figure 2. We consider a set $\{t_1,\ldots,t_8\}$ of eight potential answers to q_1 retrieved by a matchmaking algorithm as discussed in Section 5. First, we compute the preference satisfiability between q_1 and the potential target graphs (see Section 5.1). Next, we compute the structural similarity between q_1 and the potential targets (Section 5.2). Then, we apply the ranking methods described in Section 6. To illustrate, we evaluate the preference satisfiability and structural similarity between q_1 and target t_1 of Figure 1. We consider the mapping between them as depicted in Figure 4.

Preferences Satisfiability. First, the satisfiability degree δ_i' of each preference p_i of q_1 is calculated as shown in Table 3. For instance, the satisfiability degree $\delta_2 = \delta\left(p_2, a_2\right)$ between preference p_2 and annotation a_2 is obtained by function $\mu_{max}\left[reliability\right]$. According to equation (1) and the generated preference tree, the new interpretation of the satisfiability degrees is presented in column δ_i'. Second, we apply the truth degree described in Section 5.1 to obtain the global satisfiability degree between q_1 and t_1, as follows: $\delta_{\gamma_1}\left(q_1,t_1\right) =$ max (min $(1, \mu_Q\left(1/9\right))$, ..., min $(0.5, \mu_Q\left(9/9\right)))$ $= 0.67$. This means that at least 67% of preferences of q_1 are satisfied by t_1 to at least a degree 0.67.

Structural Similarity. Assume now that the structural similarities between activities are given by $SS\left(A,A'\right) = 0.72$, $SS\left(B,B'\right) = 0.85$ and $SS\left(C,C'\right) = 0.66$, and the costs of transformation of target activities are $C\left(start\right) = C\left(end\right) = C\left(A'\right) = 0$, $C\left(AND-split\right) = 0.1$, $C\left(B'\right) = C\left(C'\right) = 0.2$, $C\left(D'\right) = 0.4$, $C\left(AND-join\right) = 0.1$. In a similar way, the structural similarity degree between q_1 and t_1 is obtained as $\delta_{\gamma_2}\left(q_1,t_1\right) = 0.66$ and $\delta_{\gamma_3}\left(q_1,t_1\right) = 0.75$. Now,

Table 3. Satisfiability degrees of each pair of matched activities

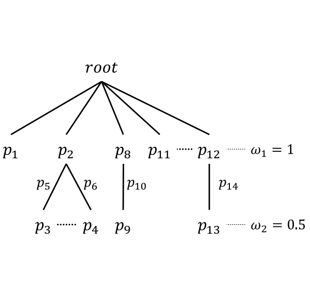
SATISFIABILITY DEGREE CALCULATION				
ATOMIC PREFERENCES			COMPLEX PREFERENCES	
PREF.	MEMBERSHIP FUNCTION	δ_i	PREFERENCE TREE	δ'_i
p_1	-	$\delta(p_1, a_1) = 1$	 root p_1 p_2 p_8 p_{11} p_{12} $\omega_1 = 1$ p_5 p_6 p_{10} p_{14} p_3 p_4 p_9 p_{13} $\omega_2 = 0.5$	$\delta'_1 = 1$
p_2	$\mu_{max}[reliability] = (50,80,100,100)$	$\delta(p_2, a_3) = 1$		$\delta'_2 = 1$
p_3	$\mu_{around}[respTime] = (10,20,20,30)$	$\delta(p_3, a_2) = 0.5$		$\delta'_3 = 0.5$
p_4	$\mu_{between}[cost] = (0,10,20,30)$	$\delta(p_4, a_4) = 0.5$		$\delta'_4 = 0.5$
p_8	$\mu_{around}[respTime] = (10,20,20,30)$	$\delta(p_8, a_6) = 1$		$\delta'_8 = 1$
p_9	$\mu_{min}[respTime] = (0,0,20,60)$	$\delta(p_9, a_5) = 0.9$		$\delta'_9 = 0.9$
p_{11}	$\mu_{max}[reliability] = (50,80,100,100)$	$\delta(p_{11}, a_{10}) = 1$		$\delta'_{11} = 1$
p_{12}	$\mu_{between}[cost] = (4,8,15,19)$	$\delta(p_{12}, a_9) = 0.75$		$\delta'_{12} =0.75$
p_{13}	$\mu_{around}[respTime] = (5,20,20,30)$	$\delta(p_{13}, a_8) = 0$		$\delta'_{13} = 0.5$

Table 4. Structural similarity and preference satisfiability degrees of a set of target graphs.

TARGET GRAPH	STRUCTURAL SIMILARITY SS	SATISFIABILITY DEGREE δ
t_1	0.66	0.67
t_2	0.29	0.72
t_3	0.85	0.40
t_4	0.78	0.35
t_5	0.78	0.21
t_6	0.68	0.72
t_7	0.66	0.72
t_8	0.66	0.35

Table 5. Ranking of target graphs according to weighted average, min-combination and lexicographic order methods.

WEIGHTED AVERAGE		MIN-COMBINATION		LEXICOGRAPHIC ORDER
t_3	$wa = 0.74$	t_6	$mc = 0.68$	t_3
t_6	$wa = 0.69$	t_7	$mc = 0.66$	t_4
t_7	$wa = 0.68$	t_1	$mc = 0.66$	t_5
t_4	$wa = 0.67$	t_3	$mc = 0.40$	t_6
t_1	$wa = 0.66$	t_4	$mc = 0.35$	t_7
t_5	$wa = 0.64$	t_8	$mc = 0.35$	t_8
t_8	$wa = 0.58$	t_2	$mc = 0.29$	t_1
t_2	$wa = 0.40$	t_5	$mc = 0.21$	t_2

$SS\,(q, t, M, E)$=min $(\delta_{\gamma_2}, \delta_{\gamma_3}) = 0.66$, which means that at least 66% of query activities are mapped to at least a degree 0.66 and at most 66% of target activities have transformation cost to at most 0.66.

Ranking. Consider the preference satisfiability and structural similarity degrees of each potential target presented in Table 4. Table 5 summarizes the results of the different ranking methods discussed in Section 6 (where $\omega_{SS} = 0.75$).

The Lexicographic order ensures that the first in the ordered list is that having the best structural similarity and, in case of ties, that having the best preference satisfiability. For example t_3 is better than all the other target graphs because its structural similarity is the greatest value. However, a drawback of this method is that the rank can be too drastic, as for the case of $t_5 : (0.78, 0.21)$ and $t_6\,(0.68, 0.72)$. In a such case, the idea of a weighted average is more suitable since it allows for a compensation. Now, with the weighted average t_6 is better than t_5 but generally it does not provide a clear semantics of the induced order.

Finally, the min-combination method relies on the worst satisfiability for each service and does not highlight the structural similarity versus the preference satisfiability. The weighted min-combination can overcome the above limitation.

8 Complexity Analysis and Experimental Results

In what follows, we first study the complexity of our approach and then present the set of experiments conducted to (i) measure the time the preference evaluation task takes in the process model matchmaking and to (ii) evaluate the effectiveness of the results.

8.1 Complexity Analysis

The complexity of our solution can be analyzed in three steps. In the case of the evaluation of atomic preferences, it implies the time to find the relevant annotation and the time to evaluate the atomic preference itself. Considering the time to find the relevant annotation in a set of m annotations per activity, the time to evaluate all the n atomic preferences of a user query is $O\left(n \cdot m\right)$, if we consider that to evaluate an atomic preference is either trivial in the case of numerical preferences or *polynomial* in the case of non-numerical preferences[4] [24]. The complexity remains *polynomial* even if we consider that each query activity defines as much atomic preferences as the number of considered non-functional properties.

In the case of the evaluation of complex preferences, the worst case is when all atomic preferences of each query activity are aggregated by complex preferences. Therefore, we have the time to evaluate each atomic preference and the time to construct and to evaluate the preference tree. The time to construct the tree is *linear*, since we only analyze the complex preferences, which are never more than half of the total of preferences. The time to evaluate the preference tree is also *linear* w.r.t. the quantity of preferences. Finally, the evaluation of the linguistic quantifiers is also *polynomial*, since it consists of an ordering of degrees plus the choosing for an element satisfying a condition. As a conclusion, we can see that the complexity of our solution is *polynomial*.

8.2 Experiments Setup

To run our experiments, we implemented a prototype that works over the system proposed by [4]. We adapted their business process model to consider non-functional annotations and their query model to consider preference annotations. We also reused their test set of process models and queries.

The main goals of our experiments are to: (i) Measure the overhead time w.r.t. the matchmaking time. It is important to note that matchmaking algorithms are NP-complete; (ii) Measure the effectiveness of our results by means of Normalized Discounted Cumulative Gain (NDCG) score; (iii) Compare the effectiveness of our results with the crisp logic-based approach presented in [9].

[4] The least common ancestor and the distances between concepts in an ontology can be calculated previously, off query time.

Test set setup. In our experiments, we considered two real-data sets containing target graphs: the first one is composed of 24 graphs of flight reservation domain having an average size of 18 activities, while the second has 32 graphs of hotel reservation domain having an average size of 12 activities, which means that the graphs have a quite considerable size. The graphs in each group have similar structure, which induces the matchmaking results to be close and not empty. We annotated the activities of each target with 10 annotations, one for each of the 10 considered QoS attributes. The attributed values were generated randomly.

Three different query process models were proposed: *FlightReservationQuery1* (FR-1), *FlightReservationQuery2* (FR-2) and *HotelReservationQuery1* (HR-1). The activities of these queries were annotated with *textual preferences* pertinent to the domain of each activities. These textual preferences were described using natural language and their semantics considered the concept of atomic and complex preferences.

We generated adapted versions of these queries according to the model proposed in our approach (Fuzzy logic-based approach) and in [9] (Crisp logic-based approach), since our objective is also to compare both approaches.

Definition of the ideal ranking. A group of experts was invited to manually analyze the satisfiability of each target graph w.r.t. to the textual queries considering the behavior specification and QoS preferences. After the analysis, the experts gave one single note to each target in a 1-7 Likert scale (1 for strongly different, 7 for strongly similar). At the end, an expert ranking was defined for each query.

Experiment execution. Five rankings were obtained after query evaluation:

1. (*Crisp AVG*) Results from crisp approach ordered by the weighted average of structural similarity and preference satisfiability;
2. (*Crisp LEX*) Results from crisp approach ordered by the lexicographic order of structural similarity and preference satisfiability;
3. (*Fuzzy AVG*) Results from our approach ordered by the weighted average of structural similarity and preference satisfiability;
4. (*Fuzzy LEX*) Results from our approach ordered by the lexicographic order of structural similarity and preference satisfiability;
5. (*Fuzzy MIN*) Results from our approach ordered by the min-combination of structural similarity and preference satisfiability;

From the results of each ranking, the top-k targets were selected and the NDCG scores were computed. The overhead time was calculated over the whole set of results. All the evaluations were conducted on a machine with an Intel i5 2.8GHz processor, 4GB of memory, running Windows 7 OS and Java VM version 1.6.

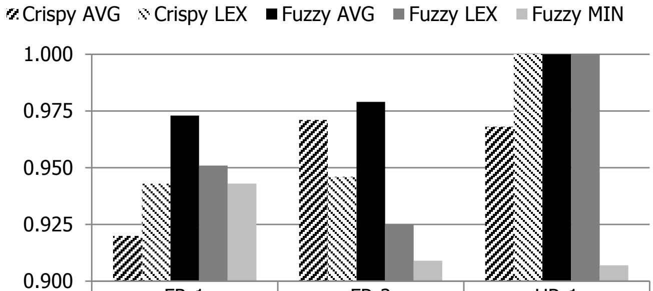
8.3 Experimental Results

As can be seen from the results presented in Table 6, the extra time taken to evaluate the hard preferences is insignificant w.r.t. the matchmaking time. It barely represents 1% of the matchmaking time.

Table 6. Matchmaking and preference evaluation times

Query/Time (ms)	AMT	APET
FR-1	82.8	0.9
FR-2	180.8	0.8
HR-1	50.9	0.8

Legend:
- *AMT:* Average Matchmaking Time
- *APET:* Average Preference Evaluation Time



	FR-1	FR-2	HR-1
Crispy AVG	0.920	0.971	0.968
Crispy LEX	0.943	0.946	1.000
Fuzzy AVG	0.973	0.979	1.000
Fuzzy LEX	0.951	0.925	1.000
Fuzzy MIN	0.943	0.909	0.907

Fig. 5. Effectiveness using NDCG measure

Figure 5 presents the NDCG scores according to the different approaches and their proposed ranking methods. In this case, the closer the score is to 1, the closer the ranking proposed by the corresponding approach is to the ranking defined by the experts. For query FR-1, all scores of fuzzy approaches overcame the crisp ones. For query FR-2, fuzzy AVG score was better than crisp results. For query HR-1, some crisp and fuzzy approaches provided the expert ranking.

The results clearly show that both crisp and fuzzy approaches provided a good effectiveness, although the scores of fuzzy AVG method always overcome crisp scores. Fuzzy LEX score was very unstable w.r.t. to the expert ranking since the experts tried to find a compromise between structure and quality, whereas in lexicographic order, the priority is given to the structural similarity while the preference similarity is only used to break ties. The restrictiveness of Fuzzy MIN proved to be very ineffective, although the semantics of its results is very strong.

9 Conclusion

In this paper, we have proposed an approach for web service selection and ranking. In our approach, the evaluation process takes into account two aspects: (i) structural similarity, and (ii) preference satisfiability. User preferences are modelled with fuzzy predicates. Both preference satisfiability and structural similarity are interpreted thanks to linguistic quantifiers. This makes the matchmaking process more flexible and realistic. Some ranking methods have been proposed as well. We also introduced a complexity analysis of our solution and we showed that the preference evaluation does not raise the complexity of process model matchmaking. Finally, we presented the set of experiments conducted over an implementation of our approach to measure the effectiveness of the results. These experiments showed that our approach gathered with the weighted average proposes a better ranking than the considered crisp solution.

As future work, we plan to apply fuzzy set-based techniques to evaluate hard constraints over QoS attributes (such as $cost \geq 20$) in process model matchmaking. We also plan to investigate other fuzzy aggregation and ranking methods that minimize the restrictiveness of those presented in this work.

Acknowledgment. This work has received support from the French National Agency for Research (ANR) on the reference ANR- 08-CORD-009.

References

1. Klusch, M., Fries, B., Sycara, K.: Automated semantic web service discovery with owls-mx. In: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems, ser. AAMAS 2006, pp. 915–922 (2006)
2. Dijkman, R., Dumas, M., García-Bañuelos, L.: Graph matching algorithms for business process model similarity search. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) BPM 2009. LNCS, vol. 5701, pp. 48–63. Springer, Heidelberg (2009)
3. van Dongen, B., Dijkman, R., Mendling, J.: Measuring similarity between business process models. In: Bellahsène, Z., Léonard, M. (eds.) CAiSE 2008. LNCS, vol. 5074, pp. 450–464. Springer, Heidelberg (2008)
4. Grigori, D., Corrales, J.C., Bouzeghoub, M., Gater, A.: Ranking bpel processes for service discovery. IEEE Transactions on Services Computing 3, 178–192 (2010)
5. Şora, I., Lazăr, G., Lung, S.: Mapping a fuzzy logic approach for qos-aware service selection on current web service standards. In: ICCC-CONTI, pp. 553–558 (2010)
6. Zhang, Y., Huang, H., Yang, D., Zhang, H., Chao, H.-C., Huang, Y.-M.: Bring qos to p2p-based semantic service discovery for the universal network. Personal Ubiquitous Computing 13(7), 471–477 (2009)
7. Kritikos, K., Plexousakis, D.: Semantic qos metric matching. In: Proc. of ECOWS, pp. 265–274 (2006)
8. Dubois, D., Prade, H.: Using fuzzy sets in flexible querying: Why and how? In: Proc. of FQAS, pp. 89–103 (1996)
9. Lemos, F., Gater, A., Grigori, D., Bouzeghoub, M.: Adding preferences to semantic process model matchmaking. In: Proc. of GAOC (2011)
10. Hristidis, V., Koudas, N., Papakonstantinou, Y.: Prefer: A system for the efficient execution of multi-parametric ranked queries. In: SIGMOD Conference, pp. 259–270 (2001)
11. Bosc, P., Pivert, O.: Sqlf: a relational database language for fuzzy querying. IEEE Trans. on Fuzzy Systems 3(1), 1–17 (1995)

12. Chomicki, J.: Preference formulas in relational queries. ACM Transactions on Database Systems 28(4), 427–466 (2003)
13. Kießling, W.: Foundations of preferences in database systems. In: VLDB. VLDB Endowment, pp. 311–322 (2002)
14. Börzsönyi, S., Kossmann, D., Stocker, K.: The skyline operator. In: ICDE, pp. 421–430 (2001)
15. Zadeh, L.A.: Fuzzy sets. Information and Control 8(3), 338–353 (1965)
16. D'Mello, D.A., Kaur, I., Ram, N., Ananthanarayana, V.S.: Semantic web service selection based on business offering. In: Proc. of EMS, pp. 476–481 (2008)
17. Agarwal, S., Lamparter, S., Studer, R.: Making Web services tradable: A policy-based approach for specifying preferences on Web service properties. Web Semantics: Science, Services and Agents on The World Wide Web 7(1), 11–20 (2009)
18. Sathya, M., Swarnamugi, M., Dhavachelvan, P., Sureshkumar, G.: Evaluation of qos based web- service selection techniques for service composition. IJSE 1, 73–90 (2010)
19. Lin, M., Xie, J., Guo, H., Wang, H.: Solving qos-driven web service dynamic composition as fuzzy constraint satisfaction. In: Proc. of EEE, pp. 9–14 (2005)
20. Chen, M.-F., Gwo-Hshiung, T., Ding, C.: Fuzzy mcdm approach to select service provider. In: Proc. of ICFS (2003)
21. Xiong, P., Fanin, Y.: Qos-aware web service selection by a synthetic weight. In: Proc. of FSKD (3), pp. 632–637 (2007)
22. Hafeez, O., Chung, S., Cock, M.D., Davalos, S.: Towards an intelligent service broker with imprecise constraints: Fuzzy logic based service selection by using sawsdl. TCSS 702 Design Project in Computing and Software Systems, University of Washington (2008)
23. Dumas, M., García-Bañuelos, L., Polyvyanyy, A., Yang, Y., Zhang, L.: Aggregate quality of service computation for composite services. In: Maglio, P.P., Weske, M., Yang, J., Fantinato, M. (eds.) ICSOC 2010. LNCS, vol. 6470, pp. 213–227. Springer, Heidelberg (2010)
24. Wu, Z., Palmer, M.S.: Verb semantics and lexical selection. In: Proc. of ACL, pp. 133–138 (1994)
25. Ehrig, M., Koschmider, A., Oberweis, A.: Measuring similarity between semantic business process models. In: Proc. of the Fourth APCCM, vol. 67, pp. 71–80 (2007)
26. Koschmider, A., Oberweis, A.: How to detect semantic business process model variants? In: Proc. of the 2007 ACM SAC, pp. 1263–1264 (2007)
27. Glöckner, I.: Fuzzy Quantifiers in Natural Language: Semantics and Computational Models. Der Andere Verlag, Osnabrück (2004)
28. Yager, R.R.: General multiple-objective decision functions and linguistically quantified statements. International Journal of Man-Machine Studies 21, 389–400 (1984)
29. Liétard, L.: A new definition for linguistic summaries of data. In: IEEE World Congress on Computational Intelligence, Fuzzy. IEEE, Hong-Kong (2008)
30. Dubois, D., Prade, H.: Handling bipolar queries in fuzzy information processing. In: Galindo, J. (ed.) Handbook of Research on Fuzzy Information Processing in Databases. IGI Global, pp. 97–114 (2008)