Selection of Web Services with Imprecise QoS Constraints

Martine De Cock
Dept. of Applied Mathematics
and Computer Science
Ghent University
Krijgslaan 281 (S9), 9000 Gent, Belgium
Martine.DeCock@UGent.be

Sam Chung, Omar Hafeez
Institute of Technology
University of Washington Tacoma
1900 Commerce Street, Tacoma,
Washington 98402-3100, USA
{chungsa,mohafeez}@u.washington.edu

Abstract

When several functionally equivalent web services are available to perform the same task, their Quality of Service (QoS) characteristics such as performance and reliability become important in the selection process. Consumers that specify their QoS requirements too strictly however, risk not finding any web services meeting their demands. Therefore, in this paper we allow QoS constraints to be described imprecisely as fuzzy sets. We compare the effectiveness of an intelligent web service selection algorithm that takes these imprecise QoS constraints into account with a baseline algorithm acting on precise QoS values.

1. Introduction

Web services are loosely coupled software components that can be invoked across the web, providing a mechanism for integrating various applications [13]. For example, a system for booking airline tickets may rely on different web services for ticket browsing, price comparison, currency rate conversion, credit card validation, and payment. In a service oriented architecture, a provider publishes a description of the service that it offers to a broker. A consumer then looks for a suitable service by querying the broker. If no suitable service is available, a further search might lead to a composed service, i.e. a chain of web services that can be invoked successively to obtain the desired result. Web service selection and composition are currently among the most addressed issues in service oriented computing. The expectation that the number of available web services will grow significantly in the years to come, motivates a growing interest in automated mechanisms to perform these tasks.

This research topic has recently been studied from different perspectives. Web services can be described by terms that are syntactically different but still carry a similar meaning. Many efforts therefore focus on semantic matchmaking and composition. This research effort is supported by the development of languages such as WSDL-S and OWL-S that allow to annotate web services with semantic concepts from ontologies, and encouraged through contests such as the Web Services Challenge[1] and the SWS Challenge[2].

In this paper we assume that semantic matchmaking has taken place to identify functionally equivalent web services. When several of these services are available to perform the same task, their quality aspects become important in the selection process. The QoS requirements for web services may include performance, reliability, scalability, capacity, robustness, exception handling, accuracy, integrity, accessibility, availability, interoperability, security, and network-related QoS requirements [6]. The consumer might specify constraints, e.g. that the execution time should be below 20 ms, as well as preferences, e.g. that a higher reliability is more important than a lower execution time. Taking into account these constraints and preferences, the service broker then has the task of selecting the services that maximize the overall QoS. This optimization problem becomes especially difficult in dynamic web service composition. A considerable amount of effort has already been spent on intelligent algorithms for web service composition, including AI planning, genetic algorithms, and linear integer programming (see e.g. [1, 4, 14]).

The focus in this paper is on the selection among functionally equivalent web services based on specifications of non-functional characteristics which relate to the QoS. From the side of the web service consumer, these are commonly expressed as hard constraints involving precise numbers, e.g. the execution time should be at most 20 ms and the availability, i.e. the percentage of time that a service is operating, should be at least 90%. If no such service with the required functionality is available, one can imagine that a service doing the job in 21 ms with an availability of 96%

[1] http://ws-challenge.org/
[2] http://sws-challenge.org/

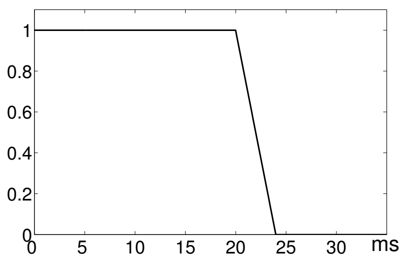

Figure 1. Constraint "at most around 20 ms"

is also still acceptable — in fact, it might even be preferable — but a conventional selection mechanism based on precise constraints will overlook it. As a result, the burden to relax the QoS constraints and query the broker again, is on the consumer.

To refine the automated web service selection process, in this paper we allow QoS constraints to be expressed by means of fuzzy sets [15]. In general, a fuzzy set A in a universe X is characterized by an $X \rightarrow [0, 1]$ mapping, called the membership function of A, that associates with every x in X the degree $A(x)$ to which x belongs to the fuzzy set A. Figure 1 depicts for example a membership function for the imprecise QoS constraint "at most around 20 ms" in the universe of execution times. All execution times smaller than or equal to 20 ms are considered to satisfy this soft constraint to degree 1, all execution times greater than or equal to 24 ms satisfy the soft constraint to degree 0 (i.e. they do not satisfy it), and in between there is a gradual transition. As we discuss further on, using such fuzzy sets gives the service consumer more flexibility in expressing his requests. The aim of this paper is to investigate whether this expressive power can lead to a greater overall satisfaction in an automated QoS-aware service selection process.

This paper is structured as follows. In Section 2, we discuss related work on the use of fuzzy logic for web service selection. Several researchers have already proposed the use of fuzzy sets for expressing soft QoS constraints. However, to the best of our knowledge, the effect of this design choice on the number of selected web services and the resulting overall satisfaction has not yet been studied, which is a main contribution of the current paper. In Section 3 we contrast a baseline algorithm for QoS-aware selection of web services with an intelligent selection algorithm using fuzzy logic. We discuss the advantages that fuzzy logic offers over the baseline. In Section 4 we report on experimental results comparing the average number of selected web services and the average overall satisfaction with the selections made by the baseline algorithm and the intelligent algorithm. Conclusions are presented in Section 5.

2. Related work

The use of fuzzy logic for QoS based web service selection has been studied by several authors. In [4] the use of fuzzy sets to represent imprecise QoS is proposed, both in soft constraints on the side of the consumer, as in specifications on the side of the broker. The latter is justified by the fact that service providers can not always express the QoS descriptions precisely at publishing time, as these might change dynamically. The focus in [4] is on matching these two sides to obtain a fitness function for a genetic algorithm based web service composition approach. To the best of our knowledge, no evaluation of such a QoS based matching mechanism for web services has been carried out yet, which is a main contribution of the current paper.

[10] applies a fuzzy multi–attribute decision making algorithm to rank web services based on the values of their QoS attributes. In contrast to our work, no flexible constraints from users are taken into account. The fuzzy constraint satisfaction approaches to web service composition in [7] and [8] do allow soft user constraints. The experiments however focus on the time needed to find an acceptable execution plan for a composed web service.

[11] proposes the use of a fuzzy rule base to infer the matching degree between the QoS of a web service and a user's demand; in a fuzzy web service search interface the user can specify a vague query like "high reliability and slow response time". This is different from our work since we assume that the user in any case wants the best QoS, i.e. the highest reliability and the fastest response time, within some additional constraints, e.g. the response time can be at most about 5 ms.

Unlike ours, some work also explicitly concerns ranking of web services based on QoS ratings by users who have previously used the service. In [12] users express their ratings as fuzzy numbers and a group decision method is applied to get an overall ranking. In [3] a fuzzy collaborative filtering approach is used to rank web services based on QoS ratings by users with similar expectations.

Finally, the work in [2, 5] concerns the discovery of web services based on their semantics. The service consumer specifies the desired kind of service by means of a vague query, e.g. a hotel booking service that offers *cheap* rooms; the service provider describes what he is offering in a similar way. Since different people tend to understand vague terms such as *cheap* differently, the attention in [5] is mainly focussed on establishing general consensus on the fuzzy sets that should be used to represent terms such as *cheap*.

3. QoS-aware web service selection

Baseline algorithm As already mentioned in the introduction, many aspects determine the QoS of a web service.

Without affecting the generality of our results, throughout this section we will illustrate the approach with two QoS parameters, namely execution time and availability. The execution time is a measure of the performance of a web service; it is the time taken by the service to process its sequence of activities [6]. In the most commonly used approach, the service broker has in its knowledge base a precise value for the execution time of each web service. This can be the execution time advertised by the service provider, or the average of the execution times observed by the service broker over the previous invocations. Availability is the probability that the system is up [6]; it is commonly expressed as the percentage of time that the service is operating.

The consumer can impose constraints on the allowed QoS values by specifying upper bounds for QoS parameters that need to be minimized (such as execution time), and lower bounds for QoS aspects that need to be maximized (such as availability). He can for example specify that the service should take no longer than 20 ms and that it should have an availability of at least 90%. In general, the broker will select a web service with execution time t and availability a iff

$$(t \leq \beta_1) \wedge (a \geq \beta_2) \tag{1}$$

where β_1 and β_2 define the consumer's QoS constraints; in the example above they equal 20 ms and 90% respectively. In general, a consumer's demand is expressed as a conjunction of constraints c_i $(i = 1, \ldots, m)$ on the different QoS parameters q_i. On a side note, we mention that the consumer does not have to specify a constraint for each QoS parameter. A "no constraint" can be easily implemented by choosing the highest or the lowest possible QoS value, depending on whether the QoS parameter is to be minimized or maximized. For example, the constraint $a \geq 0$ for availability corresponds in reality to no constraint at all.

Each constraint c_i induces a set C_i of QoS values that satisfy the constraint. The characteristic functions for the sets C_1 and C_2, corresponding to the conjuncts in Formula (1), i.e. $t \leq \beta_1$ (constraint c_1) ad $a \geq \beta_2$ (constraint c_2) are given by

$$C_1(x;\beta_1) = \begin{cases} 1 & \text{if } x \leq \beta_1 \\ 0 & \text{else} \end{cases} \tag{2}$$

$$C_2(x;\beta_2) = \begin{cases} 0 & \text{if } x < \beta_2 \\ 1 & \text{else} \end{cases} \tag{3}$$

As an example, characteristic function $C_1(\cdot;20)$ is depicted in Figure 2. Note that for ease of reference, we sometimes include the threshold value β_1 for the constraint in the notation, i.e. writing $C_1(\cdot;\beta_1)$ instead of C_1. In the same way as each constraint c_i corresponds to a set C_i, the conjunction of all constraints corresponds to the cartesian product of the C_i's, which we denote by C. An alternative for Formula (1)

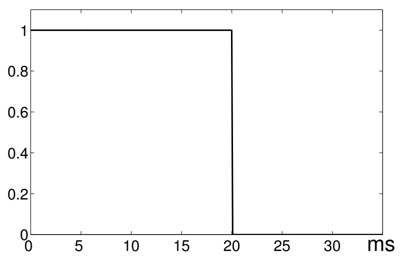

Figure 2. Constraint "at most 20 ms"

is therefore that the broker selects a web service with QoS (t, a) iff

$$(t, a) \in C_1(\cdot;\beta_1) \times C_2(\cdot;\beta_2) \tag{4}$$

In the remainder of this paper, we use A_i to denote the set of available web services whose value for QoS parameter q_i belongs to C_i. In other words, A_i is the set of web services that satisfy constraint c_i. Furthermore, we use A to denote the set of available web services whose QoS values are all in the cartesian product C, hence

$$A = \bigcap_{i=1}^{m} A_i$$

As A is the set of web services that the broker selects, we refer to it as the result set. Before moving on to the intelligent selection algorithm, we first recall some preliminaries from fuzzy set theory and fuzzy logic.

Fuzzy logic preliminaries As a generalization of binary logic, fuzzy logic considers the interval $[0, 1]$ of truth values on which it redefines the classical logical operators. Of particular interest for this paper is the fuzzy logical conjunction, which will serve to generalize Formula (1). In fuzzy logic, conjunction is represented by a triangular norm (t-norm for short), i.e. an increasing, commutative and associative $[0,1]^2 \to [0,1]$ mapping $\mathcal{T}$ satisfying the boundary condition $\mathcal{T}(1,x) = x$, for all x in $[0,1]$. One can verify that this definition coincides with the classical conjunction when restricting the allowed truth values to 0 and 1. Among the most used t-norms are the minimum $\mathcal{T}_{\mathrm{M}}$, the product $\mathcal{T}_{\mathrm{P}}$, and the Łukasiewicz t-norm $\mathcal{T}_{\mathrm{L}}$, respectively defined as $\mathcal{T}_{\mathrm{M}}(x,y) = \min(x,y)$, $\mathcal{T}_{\mathrm{P}}(x,y) = x \cdot y$, and $\mathcal{T}_{\mathrm{L}}(x,y) = \max(x + y - 1, 0)$ for all x and y in $[0,1]$. In fuzzy set theory, t-norms are used to define the cartesian product and the intersection of fuzzy sets. Let A_1 and A_2 be fuzzy sets in X, then their $\mathcal{T}$–intersection is a fuzzy set in X, defined as $(A_1 \cap_{\mathcal{T}} A_2)(x) = \mathcal{T}(A_1(x), A_2(x))$ for all x in X. Furthermore, the $\mathcal{T}$-cartesian product of a fuzzy set C_1 in X and a fuzzy set C_2 in Y is a fuzzy set in $X \times Y$, denoted by $C_1 \times_{\mathcal{T}} C_2$ and defined as $(C_1 \times_{\mathcal{T}} C_2)(x,y) =$

$\mathcal{T}(C_1(x), C_2(y))$ for all (x, y) in $X \times Y$. The intelligent selection algorithm presented next considers fuzzy sets of web services, i.e. services can belong to a certain degree to the result set. To count the number of web services in the result set, we consider both the cardinality of the support of the result set, as well as the sum of the individual membership degrees. Recall that the support of a fuzzy set A in X is the set of elements that have a membership degree greater than 0, i.e. $supp(A) = \{x | x \in X \wedge A(x) > 0\}$. Finally, the cardinality of a fuzzy set A is defined as

$$|A| = \sum_{x \in X} A(x)$$

More information on fuzzy logic can be found in a wide range of available textbooks (see e.g. [9]).

Intelligent selection algorithm In the example mentioned in the beginning of this section, the consumer asked for a service that does not take longer than 20 ms and is at least 90% of the time available. If no such service is available, one can imagine that a service doing the task in 21 ms with an availability of 96% is also acceptable. Soft constraints such as that the desired service should take no longer than *around* 20 ms and that it should have a reliability of at least *around* 90%, can be expressed using fuzzy sets C_1 and C_2 defined as

$$C_1(x; \alpha_1, \gamma_1) = \begin{cases} 1 & \text{if } x \leq \alpha_1 \\ \frac{x-\gamma_1}{\alpha_1-\gamma_1} & \text{if } \alpha_1 < x < \gamma_1 \\ 0 & \text{else} \end{cases} \quad (5)$$

$$C_2(x; \alpha_2, \gamma_2) = \begin{cases} 0 & \text{if } x < \gamma_2 \\ \frac{x-\gamma_2}{\alpha_2-\gamma_2} & \text{if } \gamma_2 \leq x < \alpha_2 \\ 1 & \text{else} \end{cases} \quad (6)$$

These membership functions are generalizations of the characteristic functions used in the baseline approach. When α_1 equals γ_1, formula (5) coincides with formula (2). In general however, formula (5) has more expressive power. All execution times up unto α_1 are considered to be fully acceptable, i.e. acceptable to degree 1. Execution times greater than γ_1 are not acceptable, i.e. acceptable to degree 0. In between, there is a gradual transition from being acceptable to not being acceptable, which allows to express that some execution times are still acceptable to a certain degree. The natural language constraint "at most around 20 ms" can for example be expressed as $C_1(\cdot; 20, 24)$ as depicted in Figure 1, while the requirement for the availability to be "at least around 90%" can be expressed with the membership function $C_2(\cdot; 90, 85)$. The degree to which a web service with QoS (t, a) belongs to the result set is then computed as its membership degree in the $\mathcal{T}$-cartesian product of C_1 and C_2, i.e.

$$(C_1 \times_{\mathcal{T}} C_2)(t, a) = \mathcal{T}(C_1(t; \alpha_1, \gamma_1), C_2(a; \alpha_2, \gamma_2)) \quad (7)$$

with $\mathcal{T}$ a t-norm (more details on the choice of a suitable t-norm are given below). We refer to the degree computed by Formula (7) as the overall satisfaction value of the consumer with the service; note that it can vary between 0 and 1. While formula (4) separates the set of web services in those that are acceptable to the consumer and those that are not, formula (7) defines a fuzzy set of web services that are acceptable to the consumer to a certain degree, ranging from 0 (not acceptable) to 1 (fully acceptable). Similarly, by themselves formula (5) and (6) induce the fuzzy sets of web services A_1 and A_2 that are acceptable according to execution time and according to availability respectively. The $\mathcal{T}$-intersection of A_1 and A_2 is the fuzzy result set A.

A first observation is that Formula 7 allows to rank the web services in increasing order of their violation of the constraints. Secondly, the intelligent selection algorithm is less sensitive to changes in the thresholds, as the following example illustrates.

Example 1 (Robustness) Two consumers are looking for a web service with the same functionality. The first consumer wants the execution time to be no more than 20 ms and the availability at least 90%. The second consumer also wants an availability of at least 90% and an execution time of at most 21 ms. Their hard constraints can be modelled by the characteristic functions $C_1^{(1)}(\cdot; 20)$ and $C_2^{(1)}(\cdot; 90)$ for consumer 1, and $C_1^{(2)}(\cdot; 21)$ and $C_2^{(2)}(\cdot; 90)$ for consumer 2. Corresponding fuzzy membership functions are for example $C_1^{(1)}(\cdot; 20, 24)$ and $C_2^{(1)}(\cdot; 90, 85)$ for consumer 1, and $C_1^{(2)}(\cdot; 21, 25)$ and $C_2^{(2)}(\cdot; 90, 85)$ for consumer 2. Assume that there is only one web service with the desired functionality, and that this web service has an execution time of 21 ms and an availability of 96%. The baseline algorithm will be able to pick this web service for consumer 2 (whose constraints are satisfied to degree 1) but has nothing to offer to consumer 1 (whose constraints are satisfied to degree 0). A minor difference in the constraints they expressed, i.e. 21 ms versus 20 ms, had a big impact on the outcome, illustrating that the baseline algorithm is very sensitive to a good choice of the thresholds. The intelligent selection algorithm on the other hand will find that the web service is satisfactory to consumer 2 to degree 1, and to consumer 1 to degree 0.75. The difference in the outcome is still noticeable but not as drastic as with the baseline algorithm. □

A possible approach for the intelligent broker is to show to the consumer all web services that have a strictly positive satisfaction degree, i.e. all those that belong to the support of the fuzzy result set. A valid question is how this approach really differs from simply relaxing the hard constraints, e.g. going from "at most 20 ms" (with support $[0, 24)$) as depicted in Figure 2 to "less than 24 ms" as depicted in Figure 3. The answer is that in the presence of

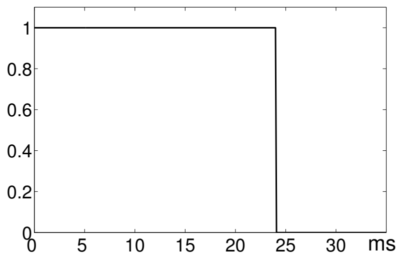

Figure 3. Constraint "less than 24 ms"

	t	a	$t \leq 20$ $a \geq 90$	$t < 24$ $a > 85$	$t \in C_1(\cdot; 20, 24)$ $a \in C_2(\cdot; 90, 85)$
s_1	18	92	1	1	1.00
s_2	21	87	0	1	0.15
s_3	23	87	0	1	0.00

Table 1. Selection of web services

only one constraint, the baseline algorithm and the intelligent selection algorithm select the same web services to show to the consumer. For two or more constraints, the difference between the two selection approaches is closely related to the choice of the t-norm. Using the t-norms $\mathcal{T}_M$ and $\mathcal{T}_P$ to model fuzzy logical conjunction, the algorithm with soft constraints selects the same web services as a selection algorithm with relaxed hard constraints. As a bonus however, the intelligent algorithm also outputs degrees of satisfaction that reflect the extent to which the constraints are violated and is, as we have illustrated with the previous example, less sensitive to the particular choices of thresholds. Another advantage of fuzzy logic becomes apparent when using $\mathcal{T}_L$, as this t-norm allows to model that a web service can be selected if violation of one QoS-constraint is sufficiently compensated by satisfaction of another, as the following example illustrates.

Example 2 (Number of selected web services) Consider a population of three functionally equivalent web services with execution and availability as depicted in Table 1. Using the initial hard constraints $t \leq 20 \wedge a \geq 90$, only web service s_1 is selected. Relaxing the hard constraints to $t < 24 \wedge a > 85$ results in the selection of all three web services, as they all satisfy these relaxed constraints. The last column considers the soft constraints "at most around 20" and "at least around 90" expressed as before by the fuzzy sets $C_1(\cdot; 20, 24)$ and $C_2(\cdot; 90, 85)$ respectively. Their cartesian product is computed by means of the t-norm $\mathcal{T}_L$, i.e. the values in the last column are obtained as $\mathcal{T}_L(1, 1) = 1$, $\mathcal{T}_L(0.75, 0.40) = 0.15$, and $\mathcal{T}_L(0.25, 0.40) = 0$. The intelligent algorithm will therefore select s_1 and s_2 because they have a strictly positive satisfaction degree. As such, it behaves differently from the baseline algorithm with the initial hard constraints that selected only s_1, but also differently from the baseline approach with the relaxed hard constraints that selected all three services. □

No relaxation of the constraint So far we have only considered relaxation of the constraints, e.g. going from the hard constraint "at most 20 ms" (depicted in Figure 2) to the soft constraint "at most around 20 ms" (depicted in Figure 1). Such a relaxation means that the consumer lowers his demands. An interesting question is whether fuzzy logic can also improve the search process when the transition from hard to soft constraints does not correspond to a relaxation. This can be compared to looking for an apartment to rent: your budget may allow you to spend up to 900 USD a month (that is your hard constraint). When you are asked how much you intend to spend, you might say something like "up to around 850 USD" (that is your soft constraint) and when you find the apartment of your dreams for 920 USD you will probably also make it work. Note that the amount you specify with the soft constraint (850 USD) is lower than the amount you specify with the hard constraint (900 USD). Hence expressing a soft constraint does not necessarily imply that the consumer is more easily satisfied; it just gives him more flexibility to express his constraints.

Note that the area under the curve of a membership function is a measure of the ease with which a consumer is satisfied. For example, the larger the value for β_1 in formula (2), the weaker the constraint imposed by the consumer is, and the larger the area under the curve of $C_1(\cdot; \beta_1)$ will be. To transition from a hard to a soft constraint without relaxation, we should therefore ensure that the area under the curve of $C_1(\cdot; \beta_1)$ matches the area under the curve for $C_1(\cdot; \alpha_1, \gamma_1)$, and likewise for $C_2(\cdot; \beta_2)$ and $C_2(\cdot; \alpha_2, \gamma_2)$. This can be achieved by choosing

$$\alpha_1 = \beta_1 - \delta_1 \text{ and } \gamma_1 = \beta_1 + \delta_1 \tag{8}$$

$$\alpha_2 = \beta_2 + \delta_2 \text{ and } \gamma_2 = \beta_2 - \delta_2 \tag{9}$$

for positive values δ_1 and δ_2. These δ-values characterize the size of the fuzzy margins within which web services are still considered acceptable by the consumer to a certain degree. Figure 4 depicts the soft constraint "at most around 18 ms" which originates from, but is no relaxation of, the hard constraint "at most 20 ms" as depicted in Figure 2. In the following section we show that this transition for hard to soft constraints, even without being a relaxation, can still improve the search process, especially when more QoS parameters are taken into account.

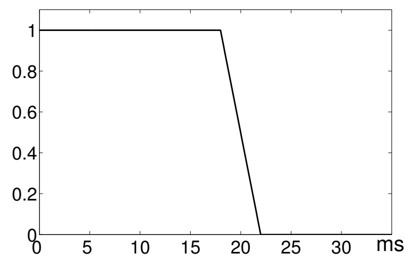

Figure 4. Constraint "at most around 18 ms"

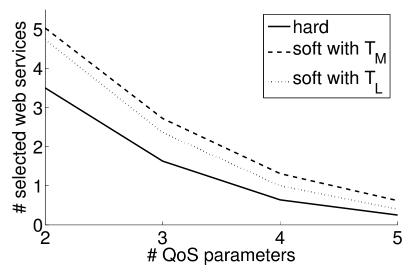

Figure 5. Average number of selected web services (for p = 20)

m	hard	soft with $\mathcal{T}_{\mathrm{M}}$	soft with $\mathcal{T}_{\mathrm{L}}$
2	3.50	3.55	3.45
3	1.63	1.68	1.57
4	0.64	0.69	0.60
5	0.25	0.29	0.23

Table 2. Average overall satisfaction on a scale from 0 to 20 (for p = 20)

4. Experimental results

In our experiment we assume that the service broker has a population of 20 functionally equivalent web services in his knowledge base. Each web service is characterized by values for up to 5 different QoS attributes. For our experimental purposes, in each iteration we randomly generate the QoS values for each of the 20 web services. For a fixed population, we then run 100 random consumer requests. The presented results are averaged out over 100 such iterations.

To generate the results presented in Figure 5 and Table 2, each request is characterized by a randomly generated value for β_1 used in the characteristic function of formula (2). The values for α_1 and γ_1 used in the fuzzy membership functions of formula (5) are computed by means of formula (8) which means that the transition from hard to soft constraints does not correspond to a relaxation. The δ-value is dynamically computed as $\delta_1 = (p \cdot \beta_1)/100$ for a percentage p. The underlying idea is to link the size of the fuzzy margin to the absolute value expressed in the hard constraint. E.g. if a user is looking for an execution time of at most around 65 ms, then 70 ms might still be acceptable to an extent. However, when he is looking for 5 ms, then 10 ms is probably not going to be acceptable to any extent. Because of space restrictions we only present results for $p = 20$. Note however that higher p-values typically allow for a larger difference between the baseline and the fuzzy approach, while for $p = 0$ both approaches coincide.

Figure 5 displays the average cardinalities of the supports of the (fuzzy) result sets obtained with the baseline, as well as those with the intelligent algorithm for the t-norms $\mathcal{T}_{\mathrm{M}}$ and $\mathcal{T}_{\mathrm{L}}$. As more QoS parameters are taken into account, less web services are selected by all approaches, but the intelligent algorithms always select more than the baseline. The fuzzy approach with $\mathcal{T}_{\mathrm{M}}$ selects those web services that satisfy all constraints to a strictly positive degree, even though this might be very small. The approach with $\mathcal{T}_{\mathrm{L}}$ is more selective as a significant violation of one constraint will only be tolerated when compensated by sufficiently high satisfaction of the other constraints. This explains why the curve for $\mathcal{T}_{\mathrm{L}}$ lies under the curve for $\mathcal{T}_{\mathrm{M}}$.

While all web services selected by the baseline satisfy the hard constraints, some of the services selected by the intelligent algorithm satisfy the constraints only to a certain degree (and hence also violate them to a certain extent). Table 2 displays the average sum of the satisfaction degrees for the web services in the (fuzzy) result set. The first column indicates the number m of QoS parameters taken into account in the user request. Since with the baseline algorithm all the selected services have a satisfaction degree equal to 1, the second column of Table 2 corresponds to the baseline values plotted in Figure 5. The last two columns differ from Figure 5, which indicates that the fuzzy result sets contain services that satisfy the constraints only to a degree. We stress once more that the transition from hard to soft constraints does not correspond to a relaxation in this experiment, as the area under the curves of the characteristic and the membership functions remains the same. This means for example that a QoS value of β_1 satisfies the soft constraint used by the intelligent algorithm only to degree 0.5, while satisfying the hard constraint to degree 1. Still, as the results in Table 2 show, the intelligent algorithm with $\mathcal{T}_{\mathrm{M}}$ consistently outperforms the baseline slightly.

5. Conclusion

Allowing service consumers to express their QoS requirements as soft constraints, and subsequently using fuzzy logic in the web service selection process, offers several extras over the traditional approach with hard constraints. First, we have studied the effect on the overall satisfaction caused by a transition from hard to soft constraints, however without relaxation of the constraints (i.e. the use of fuzzy logic does not mean that the consumer is a more easily satisfied person). In this case, we found that the use of fuzzy logic offers small improvements in the average overall satisfaction level. These improvements become more apparent when taking into account more QoS–constraints, because then it becomes harder and harder for the conventional approach to find any web services at all. Secondly, we have compared the number of web services with strictly positive satisfaction degrees found by the conventional and the fuzzy logic selection process. Using the t-norms $\mathcal{T}_{\mathrm{M}}$ and $\mathcal{T}_{\mathrm{P}}$ to model fuzzy logical conjunction, the algorithm with soft constraints produces the same results as a selection algorithm with relaxed hard constraints. However, the advantage of fuzzy logic becomes apparent when using $\mathcal{T}_{\mathrm{L}}$, as this t-norm allows to model that a web service can be selected if violation of one QoS-constraint is sufficiently compensated by satisfaction of another. A third important observation is that the use of soft constraints allows to rank selected web services on the screen in a decreasing order of satisfaction, thereby assisting the consumer in making the final choice. Finally, the intelligent selection algorithm is less sensitive to changes in the thresholds for the QoS constraints.

Acknowledgement The research reported on in this paper was carried out during a visit of Martine De Cock at the University of Washington Tacoma, supported by the Research Foundation – Flanders. Martine De Cock would like to thank the members of the Institute of Technology for their hospitality and inspiring cooperation, leading to the current paper.

References

[1] R. Akkiraju, B. Srivastava, A.-A. Ivan, R. Goodwin, T. Syeda-Mahmood, SEMAPLAN: Combining Planning with Semantic Matching to Achieve Web Service Composition. Proceedings of the 2006 International Conference on Web Services (ICWS2006), 2006

[2] K.-M. Chao, M. Younas, C.-C. Lo, T.-H. Tan, Fuzzy Matchmaking for Web Services. Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA'05), 2005

[3] V. Deora, J. Shao, W.A. Gray, N.J. Fiddian, Modelling Quality of Service in Service Oriented Computing. Proceedings of the Second IEEE International Symposium on Service–Oriented System Engineering (SOSE'06), 2006

[4] M. Di Penta, L. Troiano, Using Fuzzy Logic to Relax Constraints in GA-Based Service Composition. Late-breaking paper presented at the Genetic and Computation Conference (GECCO 2005), 2005

[5] C.-L. Huang, C.-C. Lo, K.-M. Chao, M. Younas, Reaching Consensus: A Moderated Fuzzy Web Services Discovery Method. Information and Software Technology 48(6), p. 410–423, 2006

[6] K. Lee, J. Jeon, W. Lee, S.-H. Jeong, S.-W. Park, QoS for Web Services: Requirements and Possible Approaches. W3C Note, available at http://www.w3c.or.kr/kr-office/TR/2003/ws-qos/, 2003

[7] M. Lin, J. Xie, H. Guo, H. Wang, Solving QoS–driven Web Service Dynamic Composition as Fuzzy Constraint Satisfaction. Proceedings of the 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE '05), 2005

[8] X.T. Nguyen, R. Kowalczyk, M.T. Phan, Modelling and Solving QoS Composition Problem using Fuzzy DisCSP. Proceedings of the 2006 International Conference on Web Services (ICWS2006), 2006

[9] V. Novák, I. Perfilieva, J. Močkoř, Mathematical Principles of Fuzzy Logic. Kluwer Academic Publishers, 1999

[10] H. Tong, S. Zhang, A Fuzzy Multi–Attribute Decision Making Algorithm for Web Services Selection based on QoS. Proceedings of the 2006 IEEE Asia–Pacific Conference on Services Computing (APSCC'06), 2006.

[11] H.-C. Wang, C.-S. Lee, T.-H. Ho, Combining Subjective and Objective QoS Factors for Personalized Web Service Selection. Expert Systems with Applications 32, p. 571–584, 2007

[12] P. Wang, K.-M. Chao, C.-C. Lo, C.-L. Huang, Y. Li, A Fuzzy Model for Selection of QoS Aware Web Services. Proceedings of the 2006 IEEE International Conference on e–Business Engineering (ICEBE'06), 2006

[13] S. Weerawarana, F. Curbera, F. Leymann, T. Storey, D.F. Ferguson, Web Services Platform Architecture. Prentice Hall, 2005

[14] L. Zeng, B. Benatallah, A. Ngu, M. Dumas, J. Kalagnanam, H. Chang, QoS-aware Middleware for Web Service Composition. IEEE Transactions on Software Engineering, 30(5), p. 311–328, 2004

[15] L.A. Zadeh, Fuzzy sets. Information and Control 8, p. 338–353, 1965