Towards Fine-grained Service Matchmaking by Using Concept Similarity

Alberto Fernandez[1], Axel Polleres[2], and Sascha Ossowski[1]

[1] Universidad Rey Juan Carlos, Móstoles, Spain
{alberto.fernandez,sascha.ossowski}@urjc.es
[2] Digital Enterprise Research Insitute (DERI), National University of Ireland, Galway
{axel.polleres}@deri.org

Abstract. Several description frameworks to semantically describe and match services on the one hand and service requests on the other have been presented in the literature. Many of the current proposals for defining notions of match between service advertisements and requests are based on subsumption checking in more or less expressive Description Logics, thus providing boolean match functions, rather than a fine-grained, numerical degree of match. By contrast, concept similarity measures investigated in the DL literature explicitely include such a quantitative notion. In this paper we try to take a step forward in this area by means of an analysis of existing approaches from both semantic web service matching and concept similarity, and provide preliminary ideas on how to combine these two building blocks in a unified service selection framework.

1 Introduction

In the quest to provide the underpinnings for Service-Oriented Architectures, proper methods to enable the automatic location and selection of suitable services in order to solve a given task or user request are an essential ingredient. To this end, several description frameworks to semantically annotate provided services on the one hand and express service requests on the other, both based on the same shared, formal ontologies, have been presented in the literature. Complementarily, numerous proposals for defining notions of match between such semantic descriptions of service advertisements and requests have been developed over the last few years.

Many of these are based on subsumption checking in more or less expressive Description Logics, thus providing boolean match functions, but not a fine-grained, numerical degree of match. On the contrary, concept similarity measures investigated in the DL literature provide precisely this missing piece but their application to the concrete domain of service matching is very limited. This is not as surprising as it may seem because, as pointed out in this work, the combination of service matching notions and concept similarity in a unified framework is not as straightforward as could be expected.

The objective of this paper is to take a step forward in this area by a systematical analysis of existing approaches from both semantic web service matching and concept similarity in order to combine these two building blocks into a unified service selection framework.

The rest of the paper is organized as follows. In section 2 we provide a review of notions of concept similarity in ontologies. Then, we survey current approaches found in literature to semantic service descriptions by means of ontologies and notions of match between formally defined requests and service advertisements, and discuss how concept similarity can be used to refine them. We provide also preliminary ideas where concept similarity could be beneficial to refine the notions of service matchmaking, aiming at a framework for a numerical notion of service match aimed at refineing the notions defined in literature. We conclude with an outlook and future work.

2 Preliminaries

In order to enable semantic matchmaking, it is necessary that possible communication partners, say service providers and requesters, agree on a certain *specification of a conceptualization*[13] of the domain, i.e. a shared, formal *ontology*. In the context of the Semantic Web and Semantic Web Services, this term which originally sets from Philosophy, is usually conceived by Computer Scientists as a logical theory defining and axiomatizing the concepts and properties used to describe the domain. Common to almost all ontology languages (like DAML+OIL[3] ,OWL [6], KIF [12], WSML [5], Common Logic [7]) is that in principle they are based on first-order languages, usually representing concepts as unary predicates and properties (i.e., relations between concepts) as binary predicates. In such a language a subclass-hierarchy (or taxonomy) of concepts can be expressed simply by a set of implications, where e.g.

$$\forall x OnlineBankingService(x) \rightarrow FinancialService(x) \tag{1}$$

expresses that concept $OnlineBankingService$ is a subclass of *FinancialService* and simple facts like $FinancialService(myDepotService)$ denote membership of certain instances in classes. With these basic ingredients, it is already possible to describe simple taxonomies of concepts.

2.1 Description Logics

In the context of conceptual and ontological reasoning especially the Description Logics (DL) fragments [1] of first order logics have gained momentum, due to their desirable features such as decidability of core reasoning tasks such as concept subsumption and concept membership. Among these, especially SHIQ, SHIF, and SHOIN deserve attention, being the logical foundations of DAML+OIL, OWL Light, and OWL DL, respectively. As opposed to simple subclass hierarchies expressible with formula like (1), DLs allow more sophisticated definitions of concept hierarchies by relating concepts by roles (binary relations) and defining subclass relations via these roles. Roles may also be viewed as object attributes or predicate-value pairs assigned to objects, respectively, and are usually modelled via binary predicates, where e.g.

$$\forall x.CreditCardAccountService(x) \rightarrow (\exists y.input(x,y) \rightarrow CredidCardNumber(y)) \tag{2}$$

expresses that the class of $CreditCardAccountService$ is a subclass of the "services which have a CreditCardNumber as input", or, in other words that all $CreditCard$-$AccountService$s have a $CreditCardNumber$ as $input$. In order to express such complex subclass relations, DLs provide an easier to read syntax to define complex class descriptions as follows (we take here the syntactic constracts of SHOIN, the base language of OWL DL, as a basic example), where C,D are class descriptions and R is a role name:

DL Syntax	First-order Syntax
C	$C(x)$
$\exists R.C$	$\exists y.R(x,y) \wedge C(y)$
$\forall R.C$	$\forall y.R(x,y) \rightarrow C(y)$
$C \sqcap D$	$C(x) \wedge D(x)$
$C \sqcup D$	$C(x) \vee D(x)$
$\geq nR$	$\exists y_1, y_2, \ldots y_n. \bigwedge_{1 \leq i \leq n} R(x, y_n) \wedge \bigwedge_{i \neq \rfloor} y_i \neq y_j$
$\leq nR$	$\neg \exists y_1, y_2, \ldots y_{n+1}. \bigwedge_{1 \leq i \leq n} R(x, y_{n+1}) \wedge \bigwedge_{i \neq \rfloor} y_i \neq y_j$

The above subclass statment (2) would then be written *CreditCardAccountService* $\sqsubseteq \exists input.CreditCardNumber$ in DL notation. Now, if you had for instance additional information that $CreditCardNumber \sqsubseteq PaymentCredential$, and that everything which has a payment credential as input, is in the class $CommercialService$, i.e. $\exists input.PaymentCredential \sqsubseteq CommercialService$ we could additionally infer that $CreditCardAccountService$ is a subclass of $CommercialService$, i.e. $Credit$-$CardAccountService \sqsubseteq CommercialService$ Commercial Services are not necessarily only ones dealing with credit card account management, another subclass of $CommercialService$ is for instance *CarRentalService*. Figure 1 shows a simple concept hierarchy for the concepts mentioned here showing explicit (arrows) and some inferred (dashed arrows) subclass relationships.

By complex concept definitions and inferred concepts, terms like *least common subsumer (lcs)*[3], depth or *distance* in the concept lattice, which are quite intuitive for simple taxonomies, become a bit blurry. In fact, we can observe that literature which talks about measures of distance in description logics such as [2] usually only consider very simple description logics. We will get back to this point later on and leave the reader at the moment with the question to intuitively try to assess whether $CarRentalService$ is more "similar" to $CommercialService$ than $CreditCardAccountService$?

2.2 Concept Similarity

Getting back to our goal to find measures for "matches" we find several similarity measures having been proposed in the Literature. Some authors define the *similarity* while others use *distance*. Both measures are inversely proportional and usually are taken as the inverse of each other.

Simple (atomic) concepts In this section we describe some of the approaches proposed in the literature to measure similarity between two simple concepts.

[3] also known as the *most specific ancestor (msa)*

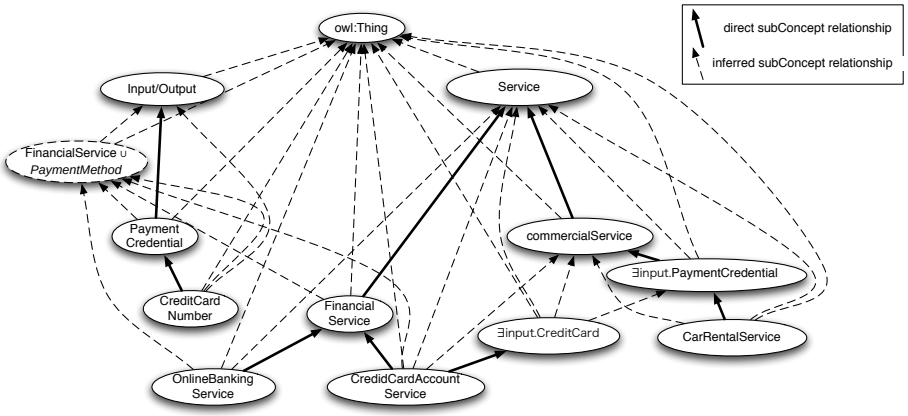

Fig. 1. A simple concept hierarchy for services

One of the most well known *distance* measures between concepts is the **length of the shortest path** between them in the taxonomy, proposed by Rada et al. [24]. As both concepts might not be along the same branch of the taxonomy tree, it can be calculated as the sum of the path length from each concept to their *lcs*.

$$dist(c_1, c_2) = depth(c_1) + depth(c_2) - 2 \times depth(lcs(c_1, c_2)) \tag{3}$$

where $depth(c)$ is the number of edges from c to the root concept.

Leacock & Chodorow [17] define the similarity between two terms as the *relatedness*, which they define as the inverse of the semantic distance.

$$relatedness(t_1, t_2) = -\log \frac{dist(t_1, t_2)}{2D} \tag{4}$$

where $dist(t_1, t_2)$ is the same as (3), and D is the maximum depth of the structure.

In their role-based[4] service matchmaking approach Fernández et al. [11] consider similarity (*degree of match)* as asymmetric. They consider some degree of similarity between concepts if there is a subsumption relation between them in the taxonomy. They define the following function, which is also based on the path length between them.

$$sim(c_1, c_2) = \begin{cases} 1 & if\ c_1 = c_2 \\ \frac{1}{2} + \frac{1}{2 \cdot e^{\|c_1, c_2\|}} & \text{if } c_2 \text{ subsumes } c_1 \\ \frac{1}{2} \cdot e^{\|c_1, c_2\|} & \text{if } c_1 \text{ subsumes } c_2 \\ 0 & otherwise \end{cases} \tag{5}$$

[4] Here we refer to roles of an actor or agent, as opposed to the roles related to ontologies that we mentioned above.

Here, $\| \cdot, \cdot \|$ is the distance $(depth(c_2) - depth(c_1))$ in the taxonomy tree.

The aforementioned measures (3) and (5) are independent of the absolute location of concepts in the taxonomy tree. Other proposals further refine these approaches by taking into account the **depth of the concepts in the taxonomy**. This makes sense under the assumption that concepts at upper layers have more general semantics and less similarity between them, while concepts at lower layers have more concrete semantics and thus stronger similarity.

In this line, Wu & Palmer [28] use the terminology *score* to define the similarity between two terms:

$$score(t_1, t_2) = \frac{2N_3}{N_1 + N_2 + 2N_3} \tag{6}$$

Where N_1 and N_2 are the length sof the shortest path from t_1 and t_2 (respectively) to the *lcs*, and N_3 is the length of the shortest path from the *lcs* to the root.

Li et al. [19] define the similarity between two concepts as:

$$sim(c_1, c_2) = \begin{cases} e^{-\alpha l} \cdot \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} & si\ c_1 \neq c_2 \\ 1 & otherwise \end{cases} \tag{7}$$

where $\alpha \geq 0$ and $\beta \geq 0$ are parameters scaling the contribution of the shortest path length (l) between the two concepts and the depth of the *lcs* (h) in the concept hierarchy, respectively.

Other authors do not base concept similarity on the distance between the concepts in the taxonomy.

Tversky [27] proposes an approach in which a concept C is characterized by a set of features, $ftrs(C)$. He introduces two kind of measures:

1. *contrast model*

$$contrast(C, D) = \theta f(ftrs(C) \cap ftrs(D)) - \alpha f(ftrs(C) \backslash ftrs(D)) - \beta f(ftrs(D) \backslash ftrs(C)) \tag{8}$$

   where $\backslash$ is set difference, θ, α and β are non-negative constants, and $f(\cdot)$ is usually the count of features in the set. That is, the number of common minus the number of non-common features.

2. *ratio model*

$$sim(C, D) = \frac{f(ftrs(C) \cap ftrs(D))}{f(ftrs(C) \cap ftrs(D)) + \alpha f(ftrs(C) \backslash ftrs(D)) + \beta f(ftrs(D) \backslash ftrs(C))} \tag{9}$$

   When asymmetry of similarity is not desired, $\alpha = \beta = 0.5$ can be chosen, and under the assumption that f is distributive over disjoint sets ($f(V \cup W) = f(V) + f(W)$), similarity is commonly taken as:

$$dist(C, D) = \frac{2 \times f(ftrs(C) \cap ftrs(D))}{f(ftrs(C)) + f(ftrs(D))} \tag{10}$$

Resnik [25] proposes an information-content based model in which there are information about the probability of an individual being described by a specific concept C ($pr(C)$). He uses the $lcs(c_1, c_2)$ as the representative of the similarity of c_1 and c_2, and proposes the *information content* as similarity measure:

$$sim(c_1, c_2) = IC(lcs(c_1, c_2)) = -\log pr(lcs(c_1, c_2)) \tag{11}$$

This approach has the advantage of not being transparent to changes in the hierarchy.

Jiang & Conrath [14] refine Resnick's measure:

$$sim(c_1, c_2) = IC(c_1) + IC(c_2) - 2 \times IC(lcs(c_1, c_2)) \tag{12}$$

Lin [20] proposes:

$$sim(c_1, c_2) = \frac{2 \times IC(lcs(c_1, c_2))}{IC(c_1) + IC(c_2)} \tag{13}$$

Borgida et al. [2] apply some of the previous approaches to a very simple DL (A), involving only conjunctions. Di Noia and colleagues [22] focus on DL and propose a ranking function for what they call *potential match* (some requests in demand D are not specified in supply S). The ranking function *rankPotential(S,D)* counts:

- the number of concepts names in D not in S,
- the number of number restrictions of D not implied by those of S,
- add recursively *rankPotential* for each universal role quantification in D,

assuming 0 to be the best ranking.

Fanizzi & d'Amato [9] define a similarity measure between concepts in ALN DL. They decompose the normal form of the concept descriptions and measure the similarity of the subconcepts:

- Primitive concepts: ratio of the number of common individuals with respect to the number of individuals belonging to either conjunct.
- Value restrictions: computed recursively, the average value is taken.
- Numeric restrictions: ratio of overlap between the two intervals and the larger interval (whose extremes are minimum and maximum), the average value is taken.

In the OWLS-MX [15] semantic Web service matching approach, logic-based reasoning is complemented by IR (Information Retrieval) based similarity computation. In particular, they allow four different token-based string metrics: the *cosine*, the *loss of information*, the *extended Jacquard* and the *Jensen-Shannon information divergence* similarity metrics. This metrics are applied to unfolded concepts, e.g. the unfolded expression *(and C (and B (and A)))* corresponds to the concept C ($C \sqsubseteq B \sqsubseteq A$).

Table 1 summarizes the different approaches to concept similarity described in this section.

The first characteristic determines whether a taxonomy tree based (*structural*) model, a *feature* based model, or a *DL* based model is applied (including the DL language used).

	Approach	focus	symmetry	distance	depth	measure	range
Rada et al.	structural	concept	yes	yes	no	distance	0..2H
Leacock-Chodorow	structural	concept	yes	yes	no	similarity	$0..\infty$
Fernndez et al.	structural	concept	no	yes	no	similarity	0..1
Wu-Palmer	structural	concept	yes	yes	yes	similarity	0..1
Li et al.	structural	concept	yes	yes	yes	similarity	0..1
OWLS-MX	structural	concept	yes	yes*	yes*	similarity	0..1
Tversky-contrast	feature	concept	yes	no	no	similarity	0..N
Tversky-ratio	feature	concept	yes/no	no	no	similarity	0..1
Resnick	structural	instance	yes	no	no	similarity	$0..\infty$
Jiang-Conrad	structural	instance	yes	no	no	distance	$0..\infty$
Lin	structural	instance	yes	no	no	similarity	0..1
Borgida et al.-feature	DL A	concept	yes	no	no	similarity	0..1(N)
Borgida at al.-struct.	DL A	concept	yes	yes	no	distance	0..2H
Borgida et al.-IC	DL A	instance	yes	no	no	distance/sim	$0..\infty(1)$
Di Noia et al.	DL	concept	no	no	no	distance	≥ 0
Fanizzi-d'Amato	DL ALN	concept	yes	no	no	similarity	0..1

Table 1. Summary of concept similarity approaches

Most approaches described here make use of *concept* definitions but others base their similarity measures on the number of instances of concepts, which are less affected by changes in the taxonomy.

Although *symmetry* has been defined by several authors as a desirable property of similarity functions, not all approaches readily comply with it. Consider, for instance, a semantic service matching scenario where it is important whether an input concept in the query subsumes or is subsumed by an input concept of an advertised service (this determines if the service can, at least, be invoked). Note that the Tversky-ratio approach allows both, symmetric and asymmetric options, depending on the values of some parameters in its similarity function.

Distance between concepts in the taxonomy is the main parameter used by structural approaches (including the Borgida et al. DL based on Rada's function). This measure makes sense under the assumption of equally distributed instances over concepts; otherwise pairs of concepts in a fine grained part of the taxonomy would be ranked with lower similarity than concepts at the same distance in another part. In the case of DL, distance is difficult to be used unless some kind of "canonical representation" is adopted. However, as we illustrated by the example of Figure 1, such a canonical representation is hard to find for expressive DLs. Common DL reasoners allow to "pre-classify" the TBox of an ontology, thus computing all subclass relations of named concepts. One could take the spanning tree of such a pre-classification, removing all transitive edges as a starting point for distance measures, but would miss the difference then for concepts defined by restrictions. Another possible way to circumvent this and maybe arriving at a more precise canonical representation would be to recursively introduce new atomic concept names for all atomic restrictions $\exists R.C$, $\forall R.C$, such that R is a role and C is an atomic concept occurring in the TBox and use these as well in the pre-classification.

DLs allowing disjunction ($\sqcup$) are hard to grasp by such approaches, we will mention more on complex compound concepts below.

As for assumptions, we mentioned above that often one assumes that in a taxonomy tree, higher nodes represent more general concepts (less similar semantically), while lower levels contain more specific concepts. For this reason, some approaches also take into account the depth of the concepts in the taxonomy. However, we note that this assumption only makes sense if we additionally assume equal distribution among instances among subclasses in the ontology in the general case.

Note that, the way OWLS-MX applies IR techniques by unfolding concept names, indirectly uses the distance and depth of concepts, but could also be viewed as kind of feature-based similarity measure mentioned above.

We also detail whether they define a *similarity* function or a *distance* function. Although both measures can be easily obtained from each other (e.g. $sim = \frac{1}{dist}$) we prefer keeping their original definition, as they vary on the *range* of the returned value (last column). A unified range of values is convenient in order to make it easier to combine/aggregate similarity values in case of complex expressions involving several concepts.

Complex (compound) concepts Rada et al. also extend the definition of *distance* to handle compound concepts represented by a set of concepts. Concepts in those sets can be interpreted as conjunctions or disjunctions. In the case of a disjunction of concepts the distance is defined as:

$$dist(C_1 \sqcup \ldots \sqcup C_k, C) = min_i dist(C_i, C) \tag{14}$$

where C_i and C represent concepts (elementary or compound). When C itself is a disjunctive concept, the same function ($dist(C_i, C)$) is in turn applied.

The distance between conjunctive concepts is defined as:

$$dist(V_1, V_2) = \begin{cases} 0 & if\ V_1 = V_2 \\ \frac{1}{|V_1||V_2|} \sum\limits_{u \in V_1} \sum\limits_{v \in V_2} dist(u, v) & otherwise \end{cases} \tag{15}$$

where V_1 and V_2 are sets representing compound concepts consisting of a conjunction ($\sqcap$) of its elements, $|\cdot|$ is set cardinality, and $dist(u, v)$ is the shortest path length between nodes u and v.

Ehrig et al. [8] analyze three layers on which similarity between concepts can be measured: data, ontology and context layer. We are interested on the ontology level. They use the function proposed by Li et al. in case of similarity between concepts. They also propose the following formula (cosine) to calculate the similarity between two sets of concepts:

$$sim(E, F) = \frac{\sum\limits_{e \in E} \boldsymbol{e} \cdot \sum\limits_{f \in F} \boldsymbol{f}}{\left| \sum\limits_{e \in E} \boldsymbol{e} \right| \cdot \left| \sum\limits_{f \in F} \boldsymbol{f} \right|} \tag{16}$$

with $E = \{e_1, e_2, \ldots\}$, $\boldsymbol{e} = (sim(e, e_1), sim(e, e_2), \ldots, sim(e, f_1), sim(e, f_2), \ldots)$, and the analogously for F and $\boldsymbol{f}$, respectively.

Sierra & Debenham [26] define the semantic similarity between two logical formulas as the maxmin similarity between the sets of concepts ($O(\cdot)$) that appear in the formulas:

$$sim(\varphi, \psi) = \max_{c_i \in O(\varphi)} \min_{c_j \in O(\psi)} \{sim(c_i, c_j)\} \tag{17}$$

In total, it seems that combinations of these approaches could be beneficial. Especially feature-set approaches seem to be worthwhile to be combined with approaches handling compound, complex DL expressions in order to get to more precise overall measures.

3 Matching Semantic Web Services

When having a closer look at current proposals to effectively annotating Web Services with Semantic descriptions, we can identify the following "hooks" for adding such annotations referring to ontologies as discussed so far. We focus here on components of semantic Web Service descriptions for which concepts in a taxonomy or complex ontology can be used for annotating them.

Service Taxonomies The entirety of the functionality offered by a Web Service can be described by a taxonomy, grouping service instances hierarchically, such as for instance the $CarRentalService$ or $CreditCardAccountService$ mentioned in Figure 1.

Operations When searching for a certain functionality, one often searches for a particular operation to execute, rather than the entirety of service functionality. Thus, most description frameworks support assigning offered operations to a taxonomy of operations in a service. Such an Operation could for instance be $RequestCreditCardBalance$, $BookRentalCar$, or all operations having a payment credential as input – which could be modeled by something like $WSDLOperation \sqcap \exists input.PaymentCredential$ – etc., all of which again may be grouped in a taxonomy/ontological hierarchy.

Inputs/Outputs Input values or output values of web services or certain service operations might be bound to a certain concept in an ontology. The problem of relating a concrete input or output message format (as for instance described in a WSDL file) is often referred to as lifting/lowering [16] problem and solved slightly different in the various Semantic Web service description approaches.

Preconditions/Postconditions Frameworks like OWL-S and WSMO offer functionality to annotate services and/or operations with pre- and postconditions, i.e. logical formulae expressing conditions over the state of the world. Since these conditions can usually not be expressed in a taxonomy or ontological hierarchy, rather more complex formalisms than Description Logics or OWL are proposed to describe these, like WSML logical expression in WSMO or SWRL rule bodies, expressing conjunctive queries on the "state

space" in OWL-S. Our focus in the current paper is on applying concept similarity to service matching and we are not aware of service matching approaches which practically exploit pre-/postcondition matching at the moment. In summary, it seems to be not entirely clear, how pre-/postcondition matching can be done at all in open service environments, which might also be a reason why they have not been considered e.g. in SAWSDL.

Summarizing, we will try to take into account those parts of the service description which allow for a conceptualication in a formal ontology, namely inputs/outputs, overall service functionality, and operations. Moreover, we deem useful to assume the following attributes/roles:

- **hasInput:** domain: $Service \sqcup Operation$, range: $Input$
- **hasOutput:** domain: $Service \sqcup Operation$, range: $Output$
- **hasOperation:** domain: $Service$, range: $Operation$

Services or Operations might have additional attributes assigned, e.g. describing non-functional properties which likewise might be useful for precise matchmaking, but which we consider out of scope for the current paper being focused on matchmaking by concept similarity.

SWS frameworks In the following table, let us briefly analyse if and how the aforementioned components are supported by three of the most common Semantic Web Service Description Frameworks[5], namely, OWL-S [21], WSMO [4] and SAWSDL [10], which has just reached the status of a proposed recommendation within W3C. We note

	Service	Operation	Input/Output	Pre/Postcondition
OWL-S	yes	OWL-S service models	OWL-S service models	OWL-S service models
WSMO	yes	WSMO capabilities	WSMO choreography model	WSMO capabilities
SAWSDL	modelReference in wsdl:interface	modelReference in wsdl:operation	modelReference in xsd:element	no

Table 2. Where SWS approaches allow annotations by concepts from given ontologies

that while SAWSDL is in general to be viewed a simpler framework than the other two, it offers useful features in comparison to its predecessors, e.g. having sophisticated support to annotate inputs and output messages; SAWSDL allows to add annotations on the level of single XML Schema elements directly within XML Schema, describing parts of the allowed input/output messages, whereas OWL-S and WSDL concepts can be assigned only per input and tying to a particular XML Schema describing the concrete message format has to be defined in the so-called grounding, typically via an XSL transformation. We mention this, because at the time being, the main development effort and activity, as well as chance of making it through to becoming a standard is on SAWSDL.

[5] (in order of "appearance")

3.1 Notions of match

In the following, we will try to analyze how existing approaches for service matching cater for ranking and make suggestions where concept similarity measured could be plugged to refine the proposed notions of match.

Paolucci [23] Many of the current approaches to semantic web services matching, particularly those based on OWL-S, started from the work of Paolucci et al. [23]. This approach proposes a matching algorithm that takes into account inputs and outputs of advertised and requested services. An output matches iff for each output of the request there is a matching output in the service description. The authors differentiate four (ranked) degrees of match (OUT_S and OUT_R correspond to outputs of the advertised and requested services, respectively)[6]:

- *exact*: if $\text{OUT}_R \doteq \text{OUT}_S$; or OUT_R is a direct subclass of OUT_S under the assumption that by advertising OUT_S the provider commits to provide outputs consistent with every immediate subtype of OUT_S.
- *plug-in*: if $\text{OUT}_S \succeq \text{OUT}_R$, that is, OUT_S could be plugged in place of OUT_R.
- *subsumes*: if $\text{OUT}_R \succeq \text{OUT}_S$.
- *fail*: no subsumption relation between OUT_S and OUT_R exists.

If there are several outputs with different degree of match, the minimum degree is used. The same algorithm is used to compute the matching between inputs, but with the order of request and advertisement reversed. Finally, the set of service advertisements is sorted by comparing output matches first, if equally scored, considering the input matches.

Applying concept similarity In Paolucci's approach services are sorted according to their degree of match, being $exact > plug-in > subsumes > fail$. However, services falling into the same category (e.g. plugin) have the same priority. A concept similarity approach can be used to refine the ranking of services inside each degree of match category. In particular, only *plug-in* and *subsumes* should be refined. As they base their classification on the subsumption relation between concepts in a taxonomy tree, one of the structural (path length based) similarity approaches might be adequate. In case of several inputs (or outputs), they consider the minimum among their degrees of match. In the same line, the minimum value can also be used to compare their similarity measures.

OWLS-MX The OWLS-MX matchmaker [15] performs hybrid semantic matching that complements logic based reasoning with syntactic IR based similarity metrics. The first three degrees of match are logic based only and, although using the same naming as Paolucci, they are defined differently (e.g. in OWLS-MX inputs of the advertisement always must at least subsume the ones in the request, so the service can be invoked).[7]

[6] $\doteq$ and $\succeq$ terminological concept equivalence and subsumption, respectively.

[7] $LSC(C)$ (set of least specific concepts (direct children) of C), $LGC(C)$ (set of least generic concepts (direct parents) of C), $Sim_{IR}(A, B) \in [0, 1]$ the numeric degree of syntactic similarity between strings A and B according to chosen IR metric IR

- *exact*: iff $\forall$ IN_S $\exists$ IN_R: $\text{IN}_S \doteq \text{IN}_R \wedge \forall$ OUT_R $\exists$ OUT_S: $\text{OUT}_R \doteq \text{OUT}_S$.
- *plug-in*: $\forall$ IN_S $\exists$ IN_R: $\text{IN}_S \succeq \text{IN}_R \wedge \forall$ OUT_R $\exists$ OUT_S: $\text{OUT}_S \in$ LSC(OUT_R). S is expected to return more specific output data whose logically defined semantics is exactly the same or very close to the requested by the user.
- *subsumes* $\forall$ IN_S $\exists$ IN_R: $\text{IN}_S \succeq \text{IN}_R \wedge \forall$ OUT_R $\exists$ OUT_S: $\text{OUT}_R \succeq \text{OUT}_S$. This relaxes the constraint of immediate output concept subsumption.
- *subsumed-by* $\forall$ IN_S $\exists$ IN_R: $\text{IN}_S \succeq \text{IN}_R \wedge \forall$ OUT_R $\exists$ OUT_S: ($\text{OUT}_S \doteq \text{OUT}_R \vee \text{OUT}_S$ $\in$ LGC(OUT_R)) $\wedge$ SIM_{IR}(S, R) $\geq \alpha$. Output data is more general than requested. It is focused on direct parent output concepts to avoid selecting services returning data too general. It is combined with the syntactic similarity.
- *logic-based fail*: matching fails according to the above logic-based semantic criteria.
- *nearest-neighbor* $\forall$ IN_S $\exists$ IN_R: $\text{IN}_S \succeq \text{IN}_R \wedge \forall$ OUT_R $\exists$ OUT_S: $\text{OUT}_R \succeq \text{OUT}_S \vee$ SIM_{IR}(S, R) $\geq \alpha$.
- *fail*: service advertisement and request do not match according to the above criteria.

Applying concept similarity As occurred in the previous approach, concept similarity could be applied when the subsumption relation is checked ($\succeq$). Now the aggregation of similarity values is a little more complicated since, besides the set of inputs and outputs, it has also to be combined with the IR similarity value (in the case of *subsumed-by* and *nearest-neighbor*). This combination is not straightforward, maybe a parametrized function which allows scaling the contribution of each measure might be appropriate.

Li Horrocks[18] A DL concept is used to describe the inputs and another for the outputs of a service advertisement or request. They extend the degrees of match proposed by Paolucci et al. by adding an *intersection* match. Formally,

- *exact*: if $A \equiv R$.
- *plug-in*: if $R \sqsubseteq A$.
- *subsume*: if $A \sqsubseteq R$.
- *intersection*: $\neg(A \sqcap R \sqsubseteq \bot)$
- *disjoint*: $A \sqcap R \sqsubseteq \bot$.

Applying concept similarity Since this approach is focused on DL, in this case distance is difficult to be used, unless some canonical representation is found. Although some approaches to concept similarity for DL were reviewed in section 2.2, they resulted to be applied on very simple DL, thus more investigation in this line is needed.

3.2 Towards a combined notion of similarity-based Service matchmaking

In this section we provide preliminary ideas on how concept similarity might be combined with notions of match. Our aim is to provide a unified matching function which returns a numeric value that can be used for ranking services. We consider such a function with range [0..1] although, of course, any other range would be acceptable as well.

In section 3 we identified three components of semantic service descriptions: *inputs*, *outputs* and *operations*. For each component, a function should return its degree

of match, I_M, O_M, and Op_M, respectively. Following current approaches, relations between components are usually classified according to several categories or *notions of match*(e.g. *exact*, *plug-in*, *intersection*,...). As analyzed in section 3.1 this classification is coarse grained and might be refined with concept similarity approaches. The ranking function must compare the *notion of match* first, and then the (numerical) similarity value. A way to facilitate the use of such a function is by dividing the range [0..1] in non overlapping intervals and scaling the similarity value to the interval corresponding to its category. Any division is acceptable under the condition that it keeps the order as is done for their categories. We define the following functions (being c_1 and c_2 concepts):

- $nom(c_1, c_2)$: returns the *notion of match* category $\in \{cat_1, cat_2, ..., cat_N\}$, where $cat_1 > cat_2 > cat_N$. Note that usually $cat_1 = exact$ and $cat_N = fail$.
- $inf(cat)$: returns the lower limit of the interval corresponding to category cat.
- $sup(cat)$: returns the upper limit of the interval corresponding to category cat.
- $sim(c_1, c_2)$: returns the concept similarity, which is a value $\in [0..1]$.
- $nosm(c_1, c_2)$: returns the *notion of similarity match* between c_1 and c_2, which is a value resulting of scaling the $sim(c_1, c_2)$ into the interval corresponding to the category $nom(c_1, c_2)$. This can be defined as

$$nosm(c_1, c_2) = inf(nom(c_1, c_2)) + sim(c_1, c_2) \cdot (sup(nom(c_1, c_2)) - inf(nom(c_1, c_2)))$$

I_M, O_M and Op_M should be defined based on $nosm$ applied to its individual elements (e.g. each of its inputs for I_M). Such a functions, for instance I_M might use aggregation functions, like the ones described for similarity of compound concepts in section 2.2 or others.

Finally, the three values need to be combined, for instance by taking a weighted sum:

$$match(S, R) = \alpha \cdot I_M + \beta \cdot O_M + \theta \cdot Op_M \tag{18}$$

where α, β and $\theta \in [0..1]$, and $\alpha + \beta + \theta = 1$.

4 Conclusions

In this paper we have provided a survey of current approaches to semantic service descriptions by means of ontologies and notions of matching between requests and service advertisements. These proposals rank service advertisements following a (coarse-grained) notion of match classification. We have also reviewed concept similarity frameworks in ontologies and discussed how these could be incorporated into the existing service description and matchmaking methodologies, so as to provide a fine-grained ranking of services. Finally, we have provided preliminary ideas aiming at a numerical notion of service match which combines notions of match with concept similarity.

This paper has reported on our work in progress. Some of the identified open issues include:

- What service description framework should we focus on? Should we select an existing one such as OWL-S, WSMO, etc, or a new one to which these approaches could be mapped?

- Which concept similarity measure better fits our framework? Is there a single "best" measure? What are the conditions that it must fulfill?
- How should values corresponding to different elements be combined?
- do different applications require the same framework or should it be configured for each of them?

Some of these questions are being tackled presently, while the in-depth coverage of others is subject to our future work.

Acknowledgements

This research has been partially supported by the European Commission under grants FP6-IST-511632 (CASCOM) and FP-6-IST-034718 (inContext), by the Spanish Ministry of Education and Science through projects URJC-CM-2006-CET-0300, TIN2006-14630-C03-02 and "Agreement Technologies" (CONSOLIDER CSD2007-0022, INGENIO 2010), as well as by the Science Foundation Ireland under the Lion project (SFI/02/CE1/I131).

References

1. Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge, January 2003.
2. Alexander Borgida, Thomas Walsh, and Haym Hirsh. Towards measuring similarity in description logics. In Ian Horrocks, Ulrike Sattler, and Frank Wolter, editors, *Description Logics*, volume 147 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2005.
3. Dan Connolly, Frank van Harmelen, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, and Lynn Andrea Stein. DAML+OIL (march 2001) reference description. W3c note, 2001.
4. J. de Bruijn, C. Bussler, J. Domingue, D. Fensel, M. Hepp, U. Keller, M. Kifer, B. König-Ries, J. Kopecky, R. Lara, H. Lausen, E. Oren, A. Polleres, D. Roman, J. Scicluna, and M. Stollberg. *Web Service Modeling Ontology (WSMO)*. W3C Member Submission, 3 June 2005. Available from `http://www.w3.org/Submission/WSMO/`.
5. J. de Bruijn, D. Fensel, U. Keller, M. Kifer H. Lausen, R. Krummenacher, A. Polleres, and L. Predoiu. *Web Service Modeling Language (WSML)*. W3C Member Submission, 3 June 2005. Available from `http://www.w3.org/Submission/WSML/`.
6. M. Dean and G. Schreiber, editors. *OWL Web Ontology Language Reference*.
7. Harry Delugach, editor. *ISO Common Logic*. 2006. Available at `http://philebus.tamu.edu/cl/`.
8. Marc Ehrig, Peter Haase, Mark Hefke, and Nenad Stojanovic. Similarity for ontologies - a comprehensive framework. In *ECIS*, 2005.
9. N. Fanizzi and C. d'Amato. A similarity measure for the aln description logic. In *Proceedings of CILC 2006 - Italian Conference on Computational Logic, Bari, Italy, June 26-27, 2006*, 2006.
10. Joel Farrell and Holger Lausen, editors. *Semantic Annotations for WSDL and XML Schema*. W3C Proposed Recommendation, 05 July 2007. Available from `http://www.w3.org/TR/sawsdl/`.

11. Alberto Fernández, Matteo Vasirani, César Cáceres, and Sascha Ossowski. A role-based support mechanism for service description and discovery. In Jingshan Huang, Ryszard Kowalczyk, Zakaria Maamar, David Martin, Ingo Müller, Suzette Stoutenburg, and Katia P. Sycara, editors, *SOCASE*, volume 4504 of *Lecture Notes in Computer Science*, pages 132–146. Springer, 2007.
12. Michael R. Genesereth and Richard E. Fikes. Knowledge interchange format, version 3.0 reference manual. Technical Report Logic-92-1, Computer Science Department, Stanford University, June 1992.
13. Tom R. Gruber. A translation approach to portable ontologies. *Knowledge Acquisition*, 2(5):199–220, 1993. Available from `http://www-ksl.stanford.edu/kst/what-is-an-ontology.html`.
14. Jay J. Jiang and David W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. *CoRR*, cmp-lg/9709008, 1997.
15. Matthias Klusch, Benedikt Fries, and Katia Sycara. Automated semantic web service discovery with owls-mx. In *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 915–922, New York, NY, USA, 2006. ACM Press.
16. Jacek Kopecky, Dumitru Roman, Matthew Moran, and Dieter Fensel. Semantic web services grounding. In *Proceedings of the Advanced Int'l Conference on Telecommunications and Int'l Conference on Internet and Web Applications and Services*, 2006.
17. Claudia Leacock and Martin Chodorow. Combining local context and WordNet similarity for word sense identification. In Christiane Fellbaum, editor, *WordNet: An Electronic Lexical Database*, pages 265–283. MIT Press, 1998.
18. Lei Li and Ian Horrocks. A software framework for matchmaking based on semantic web technology. *Int. J. of Electronic Commerce*, 8(4):39–60, 2004.
19. Yuhua Li, Zuhair Bandar, and David McLean. An approach for measuring semantic similarity between words using multiple information sources. *IEEE Trans. Knowl. Data Eng.*, 15(4):871–882, 2003.
20. Dekang Lin. An information-theoretic definition of similarity. In *Proc. 15th International Conf. on Machine Learning*, pages 296–304. Morgan Kaufmann, San Francisco, CA, 1998.
21. D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDemott, S. McIlraith, S. Narayanan, M. Paolucci, B. Parsia, T. Payne, E. Sirin, N. Srinivasan, and K. Sycara. *OWL-S: Semantic Markup for Web Services*. W3C Member Submission, 22 November 2004. Available from `http://www.w3.org/Submission/OWL-S/`.
22. Tommaso Di Noia, Eugenio Di Sciascio, Francesco M. Donini, and Marina Mongiello. Semantic matchmaking in a p-2-p electronic marketplace. In *SAC*, pages 582–586. ACM, 2003.
23. M. Paolucci, T. Kawamura, T. Payne, and K. Sycara. Semantic Matching of Web Service Capabilities. In *ISWC*, pages 333–347. Springer Verlag, 2002.
24. R. Rada, H. Mili, E. Bicknell, and M. Blettner. Development and application of a metric on semantic nets. *IEEE Transactions on Systems, Man and Cybernetics,*, 19(1):17–30, 1989.
25. Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI*, pages 448–453, 1995.
26. Carles Sierra and John K. Debenham. Trust and honour in information-based agency. In Hideyuki Nakashima, Michael P. Wellman, Gerhard Weiss, and Peter Stone, editors, *AAMAS*, pages 1225–1232. ACM, 2006.
27. A. Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, 1977.
28. Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138, Morristown, NJ, USA, 1994. Association for Computational Linguistics.