OWLS-MX: Hybrid OWL-S Service Matchmaking*

Matthias Klusch, Benedikt Fries, Mahboob Khalid
German Research Center for Artificial Intelligence
Deduction and Multiagent Systems Lab
Stuhlsatzenhausweg 3, Saarbruecken, Germany

Katia Sycara
Carnegie Mellon University
Robotics Institute
4000 Forbes Ave, Pittsburgh PA, USA

Abstract

We present and evaluate an approach to hybrid semantic Web service matching, called OWLS-MX, that utilizes both logic based reasoning and content based information retrieval techniques for services specified in OWL-S. Results of our comparative measurements of performance and scalability of OWLS-MX variants and selected token-based IR similarity metrics provide experimental evidence that building semantic Web service matchmakers purely on description logic reasoners artificially limits their potential. Experimental results show that logic based only approaches to semantic OWL-S service matching can be outperformed by both content-based and hybrid approaches to semantic service matching.

1 Introduction

Realizing the future semantic Web requires innovative solutions to the challenge of agent-coordinated semantic service discovery, composition planning, negotiation, and provision to the user. In particular, key to the success of answering the question of whether semantic Web services are relevant to a given query is how well intelligent service agents may perform semantic matching in a way that goes far beyond of what standard service discovery protocols such as UPnP, Jini, or Salutation-Lite can deliver.

Central to the majority of contemporary approaches to semantic Web service matching is that formal semantics of both service advertisements, i.e. the description that the service provider publishes in a registry or matchmaker, and requests, i.e. the description that a user looking for a desired service sends to the matchmaker, written in the same description language such as, for example, OWL-S should be defined in an underlying decidable description logic based ontology language such as OWL-DL, or OWL-Lite (Horrocks, Patel-Schneider, & van Harmelen 2004). This way, standard means of description logic reasoning can be used to automatically determine services that semantically match with a given service request based on the kind of terminological concept subsumption relations computed in the underlying ontology. Prominent examples of such logic-based approaches to semantic service matchmaking are the OWLS-UDDI matchmaker (Sycara *et al.* 2003), RACER (Li & Horrock 2003), and MAMA (Colucci *et al.* 2004).

However, like Sheth and his colleagues (Sheth, Ramakrishnan, & Thomas 2005), we believe that purely logic-based reasoning on respectively annotated content and services may not be not enough. It would artificially limit service matching to one type of representation only where expressiveness and value reasoning has been compromised at the expense of computational properties such as decidability [1]. For example, relevant semantic Web services which logical concept descriptions only differ from the request in one pair of unmatched conjunctive constraints such as for sibling concepts in a given ontology would not be found by pure logic based approaches to service retrieval. One approach to cope with this problem is to tolerate logical failures by complementary approximate matching based on syntactic similarity computations. We acknowledge that the adaptation to the latter eventually is on the user end.

Current approaches to semantic Web service matching do not exploit semantics that are implicit, for example, in patterns or relative frequencies of terms in service descriptions as computed by techniques from data mining, linguistics, or content-based information retrieval (IR). In this paper, we provide experimental evidence in favor of our hypothesis that hybrid approaches to semantic matching that exploit both formal and implicit semantics may improve the retrieval performance of semantic service matching over purely logic-based ones, and indicate tradeoffs.

The remainder of the paper is structured as follows. We briefly introduce and report on preliminary results of our experiments on IR based OWL-S service retrieval, and then present our hybrid semantic matching approach OWLS-MX in section 2 and 3, respectively. In section 4, we provide the results of our experiments of measuring the performance and scalability of OWLS-MX matchmaking variants. We briefly comment on related work in section 5, and conclude in section 6.

*This work has been supported by the German Ministry of Education and Research (BMBF 01-IW-D02-SCALLOPS), the European Commission (FP6 IST-511632-CASCOM), and the DARPA DAML program under contract F30601-00-2-0592.

[1] This limitation in principle has already been investigated since the mid 1980's in the area of federated database systems, in particular, logic-based schema integration and discovery of inter-database dependencies.

2 Content based OWL-S service retrieval

In content based service retrieval, the meaning of concept expressions of service descriptions is not a function of the way the parts are syntactically combined by description logical language operators and model-theoretically interpreted. Rather, it is implicit in the relative frequencies of indexed terms of these expressions, and exploited by string edit or token based IR similarity metrics with associated term weighting schemes.

2.1 OWL-S services

Though, in this paper, we assume the interested reader to be moderately familiar with the service description language OWL-S, we will briefly summarize its essential components needed to understand the concepts of service matching presented in subsequent sections. For more details, we refer the reader to, for example, (OWL-S).

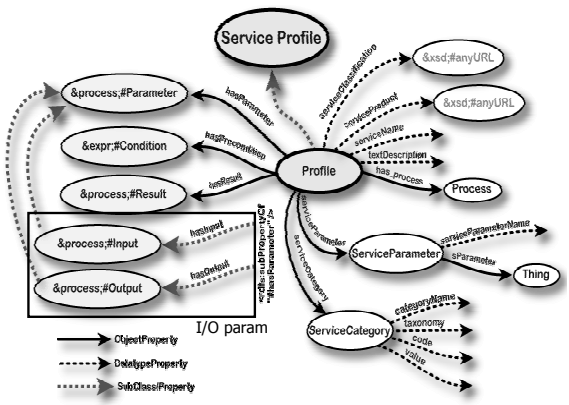

Figure 1: Parametric structure of OWL-S service profiles

OWL-S is an OWL-based Web service ontology, which supplies a core set of markup language constructs for describing the properties and capabilities of Web services in unambiguous, computer-intepretable form. The overall ontology consists of three main components: the service profile for advertising and discovering services; the process model, which gives a detailed description of a service's operation; and the grounding, which provides details on how to interoperate with a service, via messages. Specifically, it specifies the signature, that is the inputs required by the service and the outputs generated; furthermore, since a service may require external conditions to be satisfied, and it has the effect of changing such conditions, the profile describes the preconditions required by the service and the expected effects that result from the execution of the service.

The OWLS-MX matchmaker performs OWL-S service I/O matching that only exploits parameter values of hasInput and hasOutput (cf. figure 1). To the best of our knowledge, this holds for the majority of OWL-S matchmakers today, while exceptions include service process based approaches like (Bernstein & Klein 2004), but none of them actually does perform a real integrated service IOPE matching. Related work on semantic web rule languages such as SWRL and RuleML and associated reasoning means are ongoing.

2.2 Service index and similarity metrics

For our experimental evaluation, we created a service retrieval test collection and indexed the OWL-S service descriptions in an inverted category index that allows to perform structured text IR similar to the approach taken by (Grabs & Schek 2003) for XML documents. Each OWL-S service profile or process element of the document parse tree such as serviceName, textDescription, and hasInput, is considered as text category with content. Each word or concept in the content is canonically unfolded in the underlying ontology; the resulting concept expression contains primitive components from the shared minimal basic vocabular only. Stemming of these primitive components yields a set of index terms. Each index term is stored together with a document posting list that contains related categories and pointers to documents in which the index term occurs, along with its numeric weight according to the selected term weighting scheme.

Structured content based service retrieval restricts the matching of content to corresponding categories with aggregated content such that the implicit semantics of any part of OWL-S service description can be represented by a weighted category index term vector. This allows to compute the numeric degree of similarity between any combination of parameters of given service and query documents by means of traditional IR similarity metrics within the framework of the vector space model.

Recently, Cohen and his colleagues (Cohen, Ravikumar, & Fienberg 2003) provided a comprehensive experimental analysis of most prominent string similarity metrics for text IR in terms of recall and precision. Based on their experimental results, we selected the top performing, symmetric, and token-based string similarity measures for our experiments on OWL-S service retrieval. These metrics are defined as follows.

- The *cosine* similarity metric

$$Sim_{Cos}(S,R) = \frac{\vec{R}\cdot\vec{S}}{||\vec{R}||_2^2 \cdot ||\vec{S}||_2^2} \quad (1)$$

  with standard TFIDF term weighting scheme, and the unfolded concept expressions of request R and service S are represented as n-dimensional weighted index term vectors $\vec{R}$ and $\vec{S}$ respectively. $\vec{R}\cdot\vec{S} = \sum_{i=1}^{n} w_{i,R} \times w_{i,S}$, $||X||_2 = \sqrt{\sum_i^n w_{i,X}^2}$, and $w_{i,X}$ denotes the weight of the i-th index term in vector X.

- The *extended Jacquard* similarity metric $Sim_{EJ}(S,R) =$

$$\frac{\vec{R}\cdot\vec{S}}{||\vec{R}||_2^2 + ||\vec{S}||_2^2 - \vec{R}\cdot\vec{S}} \quad (2)$$

  with standard TFIDF term weighting scheme.

- The intensional *loss of information* based similarity metric $Sim_{LOI}(S,R) =$

$$1 - \frac{LOI_{IN}(R,S) + LOI_{OUT}(R,S)}{2} \quad (3)$$

$$LOI_x(R,S) = \frac{|PC_{R,x} \cup PC_{S,x}| - |PC_{R,x} \cap PC_{S,x}|}{|PC_{R,x}| + |PC_{S,x}|} \quad (4)$$

with $x \in \{IN, OUT\}$, $PC_{R,x}$ and $PC_{S,x}$ set of primitive components in unfolded logical input/output concept expression of request R and service S

- The *Jensen-Shannon information divergence* based similarity measure $Sim_{JS}(S,R) = log2 - JS(S,R) =$

$$\frac{1}{2log2}\sum_{i=1}^{n} h(p_{i,R}) + h(p_{i,S}) - h(p_{i,R} + p_{i,S}) \quad (5)$$

with probability term frequency weigthing scheme, *e.g.*, $p_{i,R}$ denotes the probability of i-th index term occurrence in request R, and $h(x) = -xlogx$,

The extended Jacquard metric is a standard for measuring the degree of overlap as the ratio of the number of shared terms (primitive components) of unfolded concepts of both query and service, and the number of terms possessed by either of them. The Jensen-Shannon measure is based on the information-theoretic, non-symmetrical Kullback-Leibler divergence and measures the pairwise dissimilarity of conditional probability term distributions within a given service document. Loss of (intensional) information in case some concept A is terminologically substituted by concept B, can be measured as the inverse ratio of the number of matching primitive components with those which remain unmatched in terminologically disjoint unfolded concept constraints. The symmetric LOI-based similarity value for given pair of service and request is then computed analogously for all I/O concept definitions involved.

For testing purposes, we created an OWL-S service retrieval test collection, called OWLS-TC V1. It consists of more than 400 OWL-S services covering six domains (education, medical care, food, travel, communication, economy). The majority of these services were retrieved from public IBM UDDI registries, semi-automatically transformed from WSDL to OWL-S, extended with appropriate input and output concepts, and then stored in the inverted text category index. OWLS-TC V1 provides a set of 9 test queries each of which is associated with a set of 7 to 10 services that two of the co-authors subjectively defined as relevant according to the standard TREC definition of binary relevance (TREC).

Please note, that no standard test collection for OWL-S service retrieval does exist yet. Therefore, like with any other reported results on retrieval performance of alternative OWL-S service matchmakers developed by different research groups world wide, we have to consider our test collection and experimental results as preliminary. Our collection OWLS-TC V1 is available as open source at http://projects.semwebcentral.org/projects/owls-tc/

2.3 Experimental results

We applied each of the similarity measures to different parts (categories) of the OWL-S service descriptions in OWLS-TC. These alternative parts are comprised of

1. the parameters hasInput, hasOutput (as commonly used for profile or signature matching), or
2. the parameters hasInput, hasOutput, serviceName, textDescription, or
3. all profile parameters, or
4. all process model parameters, or
5. all parameters of both profile and process model, or
6. the plain full text of the description

We then measured the precision and recall for each of the similarity metrics. In terms of measuring the retrieval performance, we adopted the evaluation strategy of micro-averaging the individual precision-recall curves (van Rijsbergen 1979). Let Q be the set of test queries (service requests) in OWLS-TC, A the sum of relevant documents of all requests in Q, A_R the answer set of relevant services (service advertisements) for request $R \in Q$. For each request R, we consider $\lambda = 20$ steps up to its maximum recall value, and measure the number $B_{\lambda R}|$ of relevant documents retrieved (recall) at each of these steps. Similarly, we measure related precision with the number B_λ of retrieved documents at each step λ. The micro-averaging of recall and precision (at step λ) over all requests, as we used it for performance evaluation is then defined as

$$Rec_\lambda = \sum_{R \in Q} \frac{|A_R \cap B_{\lambda R}|}{|A|}, Prec_\lambda = \sum_{R \in Q} \frac{|A_R \cap B_{\lambda R}|}{|B_\lambda|} \quad (6)$$

Preliminary results of our experiments on IR based OWL-S service retrieval are summarized in figure 2.

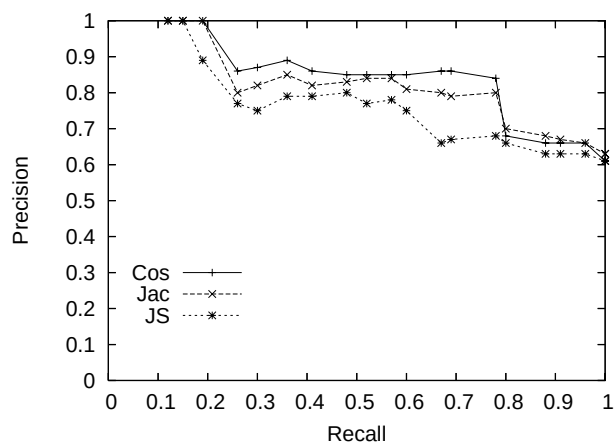

Figure 2: Retrieval performance using service profile I/O concepts that are terminologically unfolded in the respective ontologies

The use of terminologically unfolded concepts instead of names only significantly improves service retrieval performance. Both extended service profile matching with unfolded concepts, and plain full text retrieval with Jensen-Shannon similarity metric perform best. The Jensen-Shannon similarity metric is superior due to its characteristics of exploiting the implicit semantics reflected by the distance between the probabilistic term distributions of considered pairs of service and request documents rather than looking at the whole collection as it is the case for the TFIDF/cosine, or the extended Jacquard metric. Please note that current purely logic-based OWL-S algorithms use only

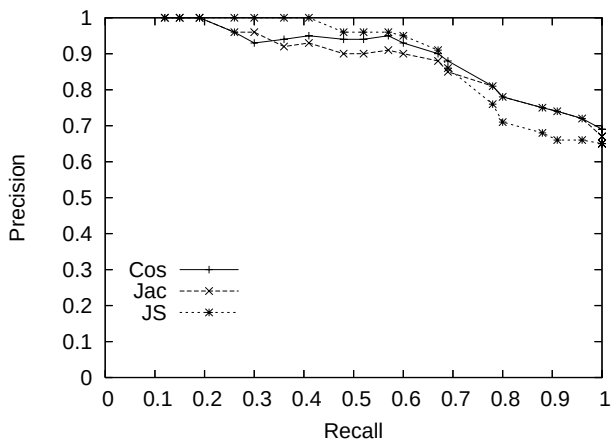

Figure 3: Retrieval performance using full plain text of OWL-S service descriptions w/ unfolded concepts

the service I/O parameters for matching whereas we tested the application of IR metrics for combinations of other service description parameters as well. We compare the performance of IR vs logic based vs hybrid service I/O matching in section 4.

In the light of these results, one question is whether purely logic-based service retrieval using the same test collection and way of performance measurement would be superior to IR based service retrieval. This has motivated our research and development of a hybrid approach to semantic OWL-S service retrieval, called OWLS-MX.

3 Hybrid semantic service matching

Hybrid semantic service matching by OWLS-MX exploits both logic-based reasoning and content-based information retrieval techniques for OWL-S service profile I/O matching. In the following, we define the hybrid semantic filters, the generic OWLS-MX algorithm and its variants, and shortly illustrate its use by example.

3.1 Matching filters of OWLS-MX

OWLS-MX computes the degree of semantic matching of given pair of service advertisement and request by successively applying five different filters EXACT, PLUG IN, SUBSUMES, SUBSUMED-BY and NEAREST-NEIGHBOR. The first four are purely logic-based filters whereas the last two are hybrid ones due to required additional computation of IR similarity values.

Let T be the terminology of the OWLS-MX matchmaker ontology specified in OWL-Lite (SHIF(D)) or OWL-DL (SHOIN(D)); CT_T the concept subsumption hierarchy of T; $LSC(C)$ the set of least specific concepts (direct children) C' of C, i.e. C' is immediate sub-concept of C in CT_T; $LGC(C)$ the set of least generic concepts (direct parents) C' of C, i.e., C' is immediate super-concept of C in CT_T; $Sim_{IR}(A, B) \in [0, 1]$ the numeric degree of syntactic similarity between strings A and B according to chosen IR metric $IR \in \{Cos, EJ, JS, LOI\}$ with used term weighting scheme and document collection (cf. section 2.1), and $\alpha \in [0, 1]$ given syntactic similarity threshold.

Exact match. Service S EXACTLY matches request R $\Leftrightarrow$ $\forall$ IN_S $\exists$ IN_R: $\text{IN}_S \doteq \text{IN}_R \wedge \forall$ OUT_R $\exists$ OUT_S: $\text{OUT}_R \doteq \text{OUT}_S$. The service I/O signature perfectly matches with the request with respect to logic-based equivalence of their formal semantics.

Plug-in match. Service S PLUGS INTO request R $\Leftrightarrow$ $\forall$ IN_S $\exists$ IN_R: $\text{IN}_S \dot{\geq} \text{IN}_R \wedge \forall$ OUT_R $\exists$ OUT_S: $\text{OUT}_S \in$ LSC(OUT_R). Service S requires less input than it has been specified in the request R. This guarantees that S will be executable with the provided input iff the involved OWL input concepts can be equivalently mapped to WSDL input messages and corresponding service signature data types. In addition, service S is expected to return more specific output data whose semantics is exactly the same or very close to what has been requested by the user. This kind of match is borrowed from the software engineering domain, where software components are considered to plug-in match with each other as defined above but not restricting the output concepts to be direct children of those of the query.

Subsumes match. Request R SUBSUMES service S $\Leftrightarrow$ $\forall$ IN_S $\exists$ IN_R: $\text{IN}_S \dot{\geq} \text{IN}_R \wedge \forall$ OUT_R $\exists$ OUT_S: $\text{OUT}_R \dot{\geq} \text{OUT}_S$. This filter is weaker than the plug-in filter with respect to the extent the returned output is more specific than requested by the user, since it relaxes the constraint of immediate output concept subsumption. As a consequence, the returned set of relevant services is extended in principle.

Subsumed-by match. Request R is SUBSUMED BY service S $\Leftrightarrow \forall$ IN_S $\exists$ IN_R: $\text{IN}_S \dot{\geq} \text{IN}_R \wedge \forall$ 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) $\geq \alpha$. This filter selects services whose output data is more general than requested, hence, in this sense, subsumes the request. We focus on direct parent output concepts to avoid selecting services returning data which we think may be too general. Of course, it depends on the individual perspective taken by the user, the application domain, and the granularity of the underlying ontology at hand, whether a relaxation of this constraint is appropriate, or not.

Logic-based fail. Service S fails to match with request R according to the above logic-based semantic filter criteria.

Nearest-neighbor match. Service S is NEAREST NEIGHBOR of request R $\Leftrightarrow \forall$ IN_S $\exists$ IN_R: $\text{IN}_S \dot{\geq} \text{IN}_R \wedge \forall$ OUT_R $\exists$ OUT_S: $\text{OUT}_R \dot{\geq} \text{OUT}_S \vee$ SIM_{IR}(S, R) $\geq \alpha$.

Fail. Service S does not match with request R according to any of the above filters.

The OWLS-MX matching filters are sorted according to the size of results they would return, in other words according to how relaxed the semantic matching. In this respect, we assume that service output data that are more general than requested relaxes a semantic match with a given query. As a consequence, we obtain the following total order of matching filters

EXACT < PLUG-IN < SUBSUMES < SUBSUMED-BY < LOGIC-BASED FAIL < NEAREST-NEIGHBOR < FAIL.

3.2 Generic OWLS-MX matching algorithm

The OWLS-MX matchmaker takes any OWL-S service as a query, and returns an ordered set of relevant services that match the query each of which annotated with its individual degree of matching, and syntactic similarity value [2]. OWLS-MX first classifies the service query I/O concepts into its local service I/O concept ontology. As usual, we assume that the type of computed terminological subsumption relation determines the degree of semantic relation between pairs of input and concepts.

Attached to each concept in the concept hierarchy are auxiliary information on whether it is used as an input or output concept by any registered service. These lists of service identifiers are used by the matchmaker to compute the set of relevant services that I/O match the given query according to its five filters as defined in section 3.1. In particular, it not only pairwisely determines the degree of logical match but syntactic similarity between the conjunctive I/O concept expression built by unfolding each of the considered query and service input (output) concept in the local matchmaker ontology. This way, logical subsumption failures produced by the integrated description logic reasoner of OWLS-MX are tolerated, if the syntactic similarity value computed by means of a specific IR similarity metric is sufficient, i.e. it exceeds a given similarity threshold.

The pseudo-code of the generic OWLS-MX matching process is given below (algorithms 1 - 3). Let $\text{INPUTS}_S = \{\ \text{IN}_{S,i} | 0 \leq i \leq s\}$, $\text{INPUTS}_R = \{\ \text{IN}_{R,j} | 0 \leq j \leq n\}$, $\text{OUTPUTS}_S = \{\ \text{OUT}_{S,k} | 0 \leq k \leq r\}$, $\text{OUTPUTS}_R = \{\ \text{OUT}_{R,t} | 0 \leq t \leq m\}$, set of input and output concepts used in the profile I/O parameters HASINPUT and HASOUTPUT of registered service S in the set $Advertisements$, and the service request R, respectively. Attached to each concept in the matchmaker ontology are auxiliary data that informs about which registered service is using this concept as an input and/or output concept.

In the following sections, we present five variants of this generic OWLS-MX matchmaking scheme, and briefly illustrate its potential use by example.

3.3 OWLS-MX variants

We implemented different variants of the generic OWLS-MX algorithm, called OWLS-M1 to OWLS-M4, each of which uses the same logic-based semantic filters but different IR similarity metric $\text{SIM}_{IR}(R, S)$ for content-based service I/O matching. In addition, the variant OWLS-MO performs purely logic-based semantic service I/O matching.

OWLS-M0. The logic-based semantic filters EXACT, PLUG-IN, and SUBSUMES are applied as defined in section 3.1, whereas the hybrid filter SUBSUMED-BY is utilized without checking the syntactic similarity constraint (cf. section 3.1).

OWLS-M1 to OWLS-M4. The hybrid semantic matchmaker variants OWLS-M1, OWLS-M3, and OWLS-M4 compute the syntactic similarity value SIM_{IR} (OUT_S, OUT_R) by use of the loss-of-information measure, extended Jacquard similarity coefficient, the cosine similarity value, and the Jensen-Shannon information divergence based similarity value, respectively (cf. section 2).

[2] The user is allowed to specify the desired degree, and syntactic similarity threshold.

Algorithm 1 *Match*: Find advertised services S that best hybridly match with a given request R; returns set of $(S, degreeOfMatch, SIM_{IR}(R, S))$ with maximum degreeOfMatch unequal FAIL, and syntactic similarity value exceeding a given threshold α.

```
function MATCH(Request R, α)
    local result, degreeOfMatch, hybridFilters = {
        SUBSUMED-BY, NEAREST NEIGHBOUR}
    for all (S, dom) ∈ CANDIDATES_inputset(INPUTS_R) ∧ (S, dom') ∈
        CANDIDATES_outputset(OUTPUTS_R) do
        degreeOfMatch ← MIN(dom, dom')
        if degreeOfMatch ≥ minDegree ∧ (
            degreeOfMatch ∉ hybridFilters ∨
            SIM_IR(R,S) ≥ α) then
            result := result ∪ { (S, degreeOfMatch,
                SIM_IR(R,S) ) }
        end if
    end for
    return result
end function
```

3.4 Example

Suppose the concept taxonomy of the local matchmaker ontology, the service request R for physicians of hospital h that provide treatment to patient p, and relevant service advertisements S_1 and S_2 as shown in figure 3.

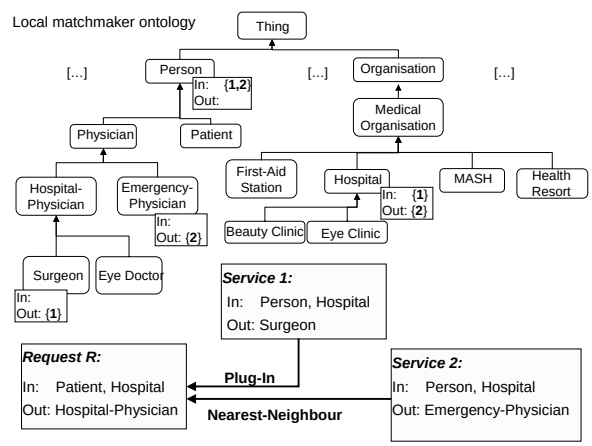

Figure 4: Example of service matching with OWLS-MX

Service S_1 is considered relevant to R, since it returns for given person p and hospital h, the individual surgeon of h that operated on p. Likewisely, service S_2 is relevant to R, since it returns those emergency physicians who provided emergency treatment to p before her transport to hospital

Algorithm 2 Find services which *input* matches with that of the request; returns set of (S, dom) with minimum degree of match dom unequal FAIL.

function CANDIDATES$_{inputset}$(INPUTS$_R$)
  local H, dom, r
  ▷ *If a service input matches with multiple request inputs the best degree is returned*
  $H := \{ (S, \text{IN}_{S,i}, dom) \in \bigcup_{j=1..n} \text{CANDIDATES}_{input}(\text{IN}_{R_j}) \mid dom = argmax_l\{ (S, \text{IN}_{S,i}, dom_l) \mid 1 \leq l \leq n, 1 \leq i \leq s\} \}$
  ▷ *If all inputs of service S are matched by those of the request, S can be executed, and the minimum degree of its potential match is returned*
  for all $S \in Advertisement$ **do**
    if $\{ (S, \text{IN}_{S_1}, dom_1), \cdots, (S, \text{IN}_{S_s}, dom_s) \} \subseteq H$ **then**
      $r := r \cup \{ (S, \text{MIN}(dom_1, \cdots, dom_s)) \}$
    end if
  end for
  ▷ *Services with no input can always be executed and are preliminary* EXACT*-match candidates:* SERVNOIN*()* = $\{$ *(S,* EXACT*)* $\mid S \in Advertisements \wedge \text{INPUTS}_S = \emptyset \}$
  ▷ *Remaining, unmatched services are at least* NEAREST NEIGHBOUR*-match candidates:* REMSERV*()* = $\{$ *(S,* NEAREST NEIGHBOUR*)* $\mid S \in Advertisements \wedge \langle S, degreeOfMatch' \rangle \notin H \}$
  return $r := r \cup \text{SERVNOIN}() \cup \text{REMSERV}()$
end function

function CANDIDATES$_{input}$(IN$_{R,j}$)
  ▷ *Classify request input concept into ontology, and use the auxiliary concept data to collect services that at least plug-in match with respect to its input.*
  local r
  $r := r \cup \{ (S, \text{IN}_S, \text{EXACT}) \mid S \in Advertisements, \text{IN}_S \in inputs_S, \text{IN}_S \doteq \text{IN}_{R,j}, \}$
  $r := r \cup \{ (S, \text{IN}_S, \text{PLUG-IN}) \mid S \in Advertisements, \text{IN}_S \in inputs_S, \text{IN}_S \dot{\geq} \text{IN}_{R,j}, \}$
  return r
end function

Algorithm 3 Find services which *output* matches with that of the request; returns set of (S, dom) with minimum degree of match unequal FAIL.

function CANDIDATES$_{outputset}$(OUTPUTS$_R$)
  local r, dom
  if OUTPUTS$_R$ = $\emptyset$ **then**
    return $\{ (S, \text{EXACT}) \mid S \in Advertisements \}$
  end if
  for all $S \in Advertisements$ **do**
    if $(S, dom_t) \in \text{CANDIDATES}_{output}(\text{OUT}_{R,t}) \wedge dom_t \geq$ SUBSUMES **for** $t = 1..m$ **then**
      $r := r \cup \{ (S, \text{MIN}\{dom_1, \cdots, dom_m\})\}$
    else if $(S, dom_t) \in \text{CANDIDATES}_{output}(\text{OUT}_{R,t}) \wedge dom_t \in \{$ EXACT, SUBSUMES $\}$ **for** $t = 1..m$ **then**
      $r := r \cup \{ (S,$ SUBSUMED-BY $\}$
    end if
  end for
  ▷ *Any remaining, unmatched service is a potential* NEAREST NEIGHBOUR*-match:* REMSERV*()* = $\{$ *(S,* NEAREST NEIGHBOUR*)* $\mid S \in Advertisements \wedge S \notin r \}$
  return $r := r \cup \text{REMSERV}()$
end function

function CANDIDATES$_{output}$(OUT$_{R,t}$)
  ▷ *Classify request output concept into ontology, and use the auxiliary concept data to collect services with output concepts that match with* OUT$_{R,t}$.
  local r
  $r := r \cup \{ (S, \text{EXACT}) \mid \text{OUT}_S \doteq \text{OUT}_{R,t} \}$
  $r := r \cup \{ (S, \text{PLUG-IN}) \mid \text{OUT}_S \in \text{LSC}(\text{OUT}_{R,t}) \wedge S \notin r \}$
  $r := r \cup \{ (S, \text{SUBSUMES}) \mid \text{OUT}_S \dot{\leq} \text{OUT}_{R,t} \wedge S \notin r \}$
  $r := r \cup \{ (S, \text{SUBSUMED-BY}) \mid \text{OUT}_S \in \text{LGC}(\text{OUT}_{R,t}) \}$
  return r
end function

h. Hence, both S_1 and S_2 should be returned as matching results to the user.

However, logic based OWLS-M0 determines S_1 as plug-in matching with R but fails to return S_2, since the formal semantics of the output concept siblings "emergency physician" and "hospital physician" in the ontology are terminologically disjoint. In this simple example, the set of terminological constraints of unfolded concepts c correspond to the set of primitive components (c^p) of which the individual concepts are canonically defined in the local matchmaker ontology T. In our example, the unfolded concept expressions are as follows.

- unfolded(Patient, T) = (and Patient^p Person^p)
- unfolded(Hospital, T) = (and Hospital^p (and MedicalOrganisation^p Organisation^p))
- unfolded(HospitalPhysician, T) = (and HospitalPhysician^p (and Physician^p Person^p))
- unfolded(Surgeon, T) = (and Surgeon^p (and HospitalPhysician^p (and Physician^p Person^p)))
- unfolded(EmergencyPhysician, T) = (and EmergencyPhysician^p (and Physician^p Person^p))

As a result, for example, OWLS-M1 returns S_1 as plug-in matching service with similarity value of Sim_{LOI} $(R, S_1) = 0.87$. In contrast to OWLS-MO, it also returns S_2, since this service is nearest-neighbor matching with R: Their implicit semantics exploited by IR similarity metric LOI (cf. (5), (6)) with Sim_{LOI} $(R, S_2) = \frac{(1-\frac{5-4}{5+4})+(1-\frac{4-2}{3+3})}{2} = 0.78 \geq \alpha = 0.7$ is sufficiently similar. Our preliminary experimental results show that this kind of matching relaxation may be useful in practice.

4 Experimental results

We implemented the OWLS-MX matchmaker variants in Java, using OWL-S 1.0, and the tableaux OWL-DL reasoner Pellet developed at University of Maryland (cf. www. mindswap. org). In our initial comparative experiments, we measured the service I/O retrieval performance of each OWLS-MX variant using unfolded I/O concept expressions and the OWL-S service retrieval test collection OWLS-TC v1. The micro-averaged R-P curves of the top and worse performing content-based IR similarity metric together with those for the OWLS-MX variants as well as the average query response time plots are displayed in figure 4 and 5, respectively. Analysis of these preliminary experimental results provide, in particular, evidence in favor of following conclusions.

- The best IR similarity metric (Cosine/TFIDF) applied to unfolded service profile I/O concepts performs close to the pure logic based OWLS-M0 (see figure 4). But OWLS-M0 is only superior to IR based matching at the very expense of its recall. However, for discovery of semantic web services, precision may be more important to the user than recall, since the set of relevant services is subject to continuous change in practice.
- Pure logic based semantic matching by OWLS-M0 can be outperformed by hybrid semantic matching, in terms of both recall and precision. That is the case, for example, by use of the best performing hybrid matchmaker OWLS-M4 (cf. figure 4). The main reason for this is, that the additional IR based similarity check of the nearest-neighbor filter allows OWLS-MX to find relevant services that OWLS-M0 would fail to retrieve such as those differing from the query in concepts that are siblings in the matchmaker ontology.

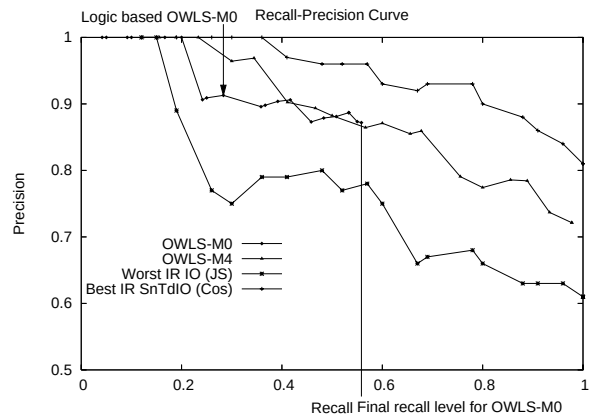

Figure 5: Recall-precision performance of logic based OWLS-M0 vs best hybrid OWLS-M4 vs IR based service I/O matching

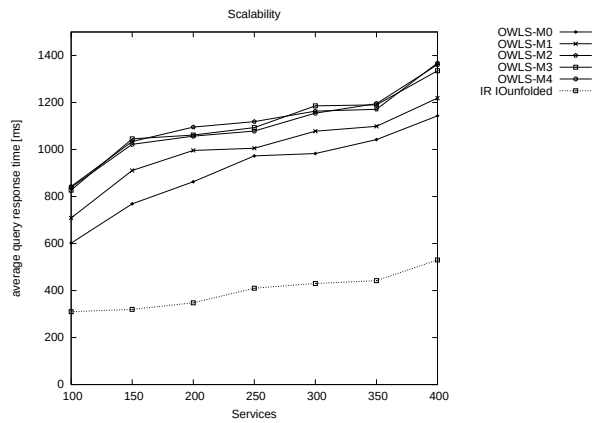

Figure 6: Average query response time of OWLS-MX

- Hybrid semantic matching by OWLS-MX can be outperformed by each of the selected syntactic IR similarity metrics to the extent additional parameters with natural language text content are used. That is the case, for example, by use of the cosine similarity metric for the service profile parameter combination of hasInput, hasOutput, serviceName, and textDescription (cf. figure 4).
- Both pure logic based and hybrid OWLS-MX matchmakers are significantly outrun by IR based service retrieval in terms of average query response time (cf. figure 5). This is due to the additional computational efforts required by OWLS-MX to determine concept subsumption relationships in NEXPTIME description logic OWL-DL.

5 Related work

Quite a few semantic Web service matchmakers have been developed in the past couple of years such as the OWLS-UDDI matchmaker (Sycara *et al.* 2003), RACER (Li & Horrock 2003), SDS (Mandell & McIllraith 2003), MAMA (Colucci *et al.* 2004), HotBlu (Constantinescu & Faltings 2003), and (Klein & Koenig-Ries 2004). Like OWLS-MX, the majority of them does perform profile based service signature (I/O) matching. Alternate approaches propose service process-model matching (Bernstein & Klein 2004), recursive tree matching (Bansal & Vidal 2003), P2P discovery (Banaei-Kashani, Chen, & Shahabi. 2004), automated selection of WSMO services (Keller *et al.* 2005) and METEOR-S for WSDL-S services (Verma *et al.* 2004). Except LARKS (Sycara *et al.* 2002), none of them is hybrid, in the sense that it exploits both explicit and implicit semantics by complementary means of logic based and approximate matching. The OWLS-MX matchmaker bases on LARKS (Sycara *et al.* 2002). However, LARKS differs from OWLS-MX in that it uses a different capability description language and description logic, neither performs subsumes and subsumed-by nor nearest-neighbour matching, and has not been experimentally evaluated yet. The purely logic-based variant OWLS-M0 is similar to the OWLS-UDDI matchmaker (Sycara *et al.* 2003) but differs from it as follows. Firstly, the latter makes use of a different notion of plug-in matching, and does not perform additional subsumed-by matching. Secondly, OWLS-M0 classifies arbitrary query concepts into its dynamically evolving ontology with commonly shared minimal basic vocabulary of primitive components instead of limiting query I/O concepts to terminologically equivalent service I/O concepts in a shared static ontology as the OWLS-UDDI matchmaker does. To the best of our knowledge, OWLS-MX has been the first hybrid matchmaker for OWL-S services.

6 Conclusions

Our approach to hybrid semantic Web service matching, called OWLS-MX, utilizes both logic based reasoning and IR techniques for semantic Web services in OWL-S. Preliminary results of our comparative experiments provide some experimental evidence in favor of the proposition that building semantic Web service matchmakers *purely* on description logic reasoners may be insufficient, and should give a clear impetus for further studies, research and development of more powerful approaches to service matching in the semantic Web across disciplines. OWLS-MX is available as open source software at http://projects.semwebcentral.org/projects/owls-mx/

References

Banaei-Kashani, F.; Chen, C.-C.; and Shahabi., C. 2004. Wspds: Web services peer-to-peer discovery service. In *Proceedings of International Symposium on Web Services and Applications (ISWS)*.

Bansal, S., and Vidal, J. 2003. Matchmaking of web services based on the daml-s service model. In *Proceedings of Second International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), Melbourne, Australia*.

Bernstein, A., and Klein, M. 2004. Towards high-precision service retrieval. In *IEEE Internet Computing, 8(1):30-36*.

Cohen, W.; Ravikumar, P.; and Fienberg, S. 2003. A comparison of string distance metrics for name-matching tasks. In *Proc. IJCAI-03 Workshop on Information Integration on the Web (IIWeb-03)*. DBLP at http://dblp.uni-trier.de.

Colucci, S.; Noia, T. D.; Sciascio, E. D.; Donini, F.; and Mongiello, M. 2004. Concept abduction and contraction for semantic-based discovery of matches and negotiation spaces in an e-marketplace. In *Proc. 6th Int Conference on Electronic Commerce (ICEC 2004)*. ACM Press.

Constantinescu, I., and Faltings, B. 2003. Efficient matchmaking and directory services. In *Proceedings of IEEE/WIC International Conference on Web Intelligence*.

Grabs, T., and Schek, H.-J. 2003. Flexible information retrieval on xml documents. In *Intelligent Search on XML Data, Applications, Languages, Models, Implementations, and Benchmarks*. Springer.

Horrocks, I.; Patel-Schneider, P.; and van Harmelen, F. 2004. From shiq and rdf to owl: The making of a web ontology language. *Web Semantics, 1(1), Elsevier*.

Keller, U.; Lara, R.; Polleres, A.; and Fensel, D. 2005. Automatic location of services. In *Proceedings of European Semantic Web Conference (ESWC), Springer, LNAI 3532*.

Klein, M., and Koenig-Ries, B. 2004. Coupled signature and specification matching for automatic service binding. In *Proceedings of European Conference on Web Services, Springer, LNAI, 183-197*.

Li, L., and Horrock, I. 2003. A software framework for matchmaking based on semantic web technology. In *Proc. 12th Int World Wide Web Conference Workshop on E-Services and the Semantic Web (ESSW 2003)*.

Mandell, D., and McIllraith, S. 2003. A bottom-up approach to automating web service discovery, customization, and semantic translation. In *Proc. 12th Int Conference on the World Wide Web (WWW 2003)*. ACM Press.

OWL-S. Semantic markup for web services; w3c member submission 22 november 2004. http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/.

Sheth, A.; Ramakrishnan, C.; and Thomas, C. 2005. Semantics for the semantic web: The implicit, the formal, and the powerful. *Semantic Web and Information Systems, 1(1), Idea Group*.

Sycara, K.; Klusch, M.; Widoff, S.; and Lu, J. 2002. Larks: Dynamic matchmaking among heterogeneous software agents in cyberspace. *Autonomous Agents and Multi-Agent Systems, 5(2), Kluwer*.

Sycara, K.; Paolucci, M.; Anolekar, A.; and Srinivasan, N. 2003. Automated discovery, interaction and composition of semantic web services. *Web Semantics, 1(1), Elsevier*.

TREC. Text retrieval conference. http://trec.nist.gov/data/.

van Rijsbergen, C. 1979. *Information Retrieval*.

Verma, K.; Sivashanmugam, K.; Sheth, A.; Patil, A.; Oundhakar, S.; and Miller., J. 2004. Meteor-s wsdi: A scalable infrastructure of registries for semantic publication and discovery of web services. *Information Technology and Management*.