ISWC 2008

The 7th International Semantic Web Conference

Ruben Lara
Tommaso Di Noia
Ioan Toma

Service Matchmaking and Resource Retrieval in the Semantic Web (SMR^2 2008)

October 27, 2008

The 7th International Semantic Web Conference
October 26 – 30, 2008
Congress Center, Karlsruhe, Germany

ISWC 2008

Platinum Sponsors

Ontoprise

Gold Sponsors

BBN
eyeworkers
Microsoft
NeOn
SAP Research
Vulcan

Silver Sponsors

ACTIVE
ADUNA
Saltlux
SUPER
X-Media
Yahoo

Organizing Committee

General Chair
Tim Finin (University of Maryland, Baltimore County)

Local Chair
Rudi Studer (Universität Karlsruhe (TH), FZI Forschungszentrum Informatik)

Local Organizing Committee
Anne Eberhardt (Universität Karlsruhe)
Holger Lewen (Universität Karlsruhe)
York Sure (SAP Research Karlsruhe)

Program Chairs
Amit Sheth (Wright State University)
Steffen Staab (Universität Koblenz Landau)

Semantic Web in Use Chairs
Mike Dean (BBN)
Massimo Paolucci (DoCoMo Euro-labs)

Semantic Web Challenge Chairs
Jim Hendler (RPI, USA)
Peter Mika (Yahoo, ES)

Workshop chairs
Melliyal Annamalai (Oracle, USA)
Daniel Olmedilla (Leibniz Universität Hannover, DE)

Tutorial Chairs
Lalana Kagal (MIT)
David Martin (SRI)

Poster and Demos Chairs
Chris Bizer (Freie Universität Berlin)
Anupam Joshi (UMBC)

Doctoral Consortium Chairs
Diana Maynard (Sheffield)

Sponsor Chairs
John Domingue (The Open University)
Benjamin Grosof (Vulcan Inc.)

Metadata Chairs
Richard Cyganiak (DERI/Freie Universität Berlin)
Knud Möller (DERI)

Publicity Chair
Li Ding (RPI)

Proceedings Chair
Krishnaprasad Thirunarayan (Wright State University)

Fellowship Chair
Joel Sachs (UMBC)

Preface

Welcome to the International Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web, SMR^2, Karlsruhe, Germany, October 27, 2008. The workshop, at its second edition, is devoted to the discussion of theoretical, technical and methodological solutions to the problem of finding the best service or the best resource when searching in a web of meanings like the Semantic Web is.

This year we accepted 8 papers covering many aspects of matchmaking and retrieval in the Semantic Web. As in the last edition, many of them focus on (semantic) web service discovery and selection.

In **Semantic Web Service Selection with SAWSDL-MX**, Matthias Klusch and Patrick Kapahne extend and adapt their hybrid -MX matchmaking framework to SAWDL semantic web service descriptions. SAWSDL is also the main character of the paper **Uncovering WSDL Specifications' Data Semantics** by George A. Vouros et al. Here the focus is on automatic annotation of WSDL specifications mapping input/output WSDL specifications to ontology classes.

Whenever a service matchmaker returns a list of service satisfying a specific goal the main question is: how satisfactory is the result with respect to the provided goal? An answer to this basic question is provided in **Evaluating Semantic Web Service Matchmaking Effectiveness Based on Graded Relevance** (Ulrich Küster and Birgitta König-Ries) where the authors propose a graded relevance scale to evaluate SWS matchmakers. In all interoperability scenarios, the ultimate goal of matchmakers is the eventual orchestration of discovered services. In **Model-Driven Semantic Service Matchmaking for Collaborative Business Processes**, Matthias Klusch et al. propose to apply the principles of model driven-design to Semantic Web service technology to assist a business orchestrator finding suitable services at design time, and composing work-flows for agent-based execution.

Usually, a negotiation phase follows the matchmaking/discovery one. In **Combining Boolean Games with the Power of Ontologies for Automated Multi-Attribute Negotiation in the Semantic Web**, Thomas Lukasiewicz and Azzurra Ragone propose a new formal framework to combine Semantic Web technologies with a game theoretic approach for multi-attribute negotiation.

Talking about Semantic Web we do not have to forget that related technologies can also be applied in scenarios different from the Web. In **Match'n'Date: Semantic Matchmaking for Mobile Dating in P2P Environments**, Michele Ruta et al. describe an application of Semantic Web technologies to a mobile environment.

Finally, in **Look Ma, No Hands: Supporting the semantic discovery of services without ontologies** (George A. Vouros et al.) and **Closing the Service Discovery Gap by Collaborative Tagging and Clustering Techniques** (Alberto Fernandez et al.) the authors show how to use and combine techniques and tools of the current web to solve problems in the Semantic Web.

Our thanks go to all authors for their valuable submissions and to the invited speaker Holger Lausen for his talk: **Enabling Discovery of Web Services on the Internet**. We are also very grateful to the members of the Program Committee and the external reviewers for their time and efforts.

Tommaso Di Noia, Ruben Lara and Ioan Toma

SMR^2 PC chairs

Workshop Organization

Program co-chairs

Tommaso Di Noia (Technical University of Bari, Italy)
Ruben Lara (Telefónica I&D, Spain)
Ioan Toma (STI Innsbruck, Austria)

Steering Committee

Abraham Bernstein (U. Zurich, Switzerland)
Tommaso Di Noia (TU Bari, Italy)
Takahiro Kawamura (Toshiba, Japan)
Matthias Klusch (DFKI, Germany)
Ulrich Kster (U. Jena, Germany)
Ruben Lara (Telefonica R&D, Spain)
Alain Leger (France Telecom, France)
David Martin (SRI International, USA)
Terry Payne (U. Southampton, UK)
Axel Polleres (DERI, National University of Ireland, Galway)
Massimo Paolucci (NTT DoCoMo Europe, Germany)
Ioan Toma (STI Innsbruck, Austria)

Program Committee

Sudhir Agarwal (University of Karlsruhe, Germany)
Rama Akkiraju (IBM, USA)
Sinuh Arroyo (U. Alcala de Henares, Spain)
Djamal Benslimane (Universit Claude Bernard Lyon, France)
Gheorghe Cosmin Silaghi (Babes-Bolyai University Cluj-Napoca, Romania)
Eugenio Di Sciascio (Technical University of Bari, Italy)
Stephan Grimm (FZI Karlsruhe, Germany)
Sung-Kook Han (Won Kwang University, Korea)
Frank Kaufer (University of Potsdam, Germany)
Uwe Keller (STI Innsbruck, Austria)
Holger Lausen (seekda, Austria)
Freddy Lecue (Orange-France Telecom, France)
Ioan Alfred Letia (Technical University of Cluj-Napoca, Romania)
Christophe Rey (ISIMA, University of Clermont Ferrand, France)
Dumitru Roman (STI Innsbruck, Austria)
Farouk Toumani (ISIMA, University of Clermont Ferrand, France)

External Reviewers

Claudio Baldassarre (The Open University, UK)
Claudia d'Amato (University of Bari, Italy)
Alessio Gugliotta (The Open University, UK)
Jacek Kopecky (University of Innsbruck, Austria)
Tomasz Kaczmarek (Poznan University of Economics, Poland)

Table of Contents

Invited Talk: Enabling Discovery of Web Services on the Internet 1
Holger Lausen

Semantic Web Service Selection with SAWSDL-MX 3
Matthias Klusch and Patrick Kapahne

Uncovering WSDL Specifications' Data Semantics 19
George A. Vouros, Alexandros Valarakos, Konstantinos Kotis

Evaluating Semantic Web Service Matchmaking Effectiveness Based on Graded Relevance 35
Ulrich Küster and Birgitta König-Ries

Model-Driven Semantic Service Matchmaking for Collaborative Business Processes 51
Matthias Klusch, Stefan Nesbigall, and Ingo Zinnikus

Combining Boolean Games with the Power of Ontologies for Automated Multi-Attribute Negotiation in the Semantic Web 67
Thomas Lukasiewicz and Azzurra Ragone

Match'n'Date: Semantic Matchmaking for Mobile Dating in P2P Environments 83
Michele Ruta, Tommaso Di Noia, Eugenio Di Sciascio, and Floriano Scioscia

Look Ma, No Hands: Supporting the semantic discovery of services without ontologies 99
George A. Vouros, Fragkiskos Dimitrokallis, Konstantinos Kotis

Closing the Service Discovery Gap by Collaborative Tagging and Clustering Techniques 115
Alberto Fernandez, Conor Hayes, Nikos Loutas,Vassilios Peristeras, Axel Polleres, Konstantinos Tarabanis

Invited Talk: Enabling Discovery of Web Services on the Internet

Holger Lausen

seekda OG
Museumstrae 21/302a – 6020 Innsbruck
Austria

The Web is moving from a collection of static documents to a set of Web Services. Todays major search engines provide fast and easy access to existing Web pages, however only little attention has been paid to provide a similar easy and scalable access to find existing publicly available Web Services. We present an approach that considers existing practical realities and has been used to build the seekda.com Web Service search engine. Using this approach seekda has indexed the largest pool of Web Service known so far. The talk will give details on how existing Web Service related data can be obtained from the Web, how it can be analyzed to obtain semantic annotations, how availability monitoring can be used to assure accuracy and finally ideas on how user feedback can be used to improve the quality of the available information.

Semantic Web Service Selection with SAWSDL-MX

Matthias Klusch and Patrick Kapahnke

German Research Center for Artificial Intelligence
Stuhlsatzenhausweg 3, Saarbrücken, Germany
klusch@dfki.de, patrick.kapahnke@dfki.de

Abstract. In this paper, we present an approach to hybrid semantic Web service selection of semantic services in SAWSDL based on logic-based matching as well as text retrieval strategies. We discuss the principles of semantic Web service description in SAWSDL and selected problems for service matching implied by its specification. Based on the result of this discussion, we present different variants of hybrid semantic selection of SAWSDL services implemented by our matchmaker called SAWSDL-MX together with preliminary results of its performance in terms of recall/precision and average query response time. For experimental evaluation we created a first version of a SAWSDL service retrieval test collection called SAWSDL-TC.

1 Introduction

As a W3C recommendation dated August 28, 2007, the SAWSDL[1] specification proposes mechanisms to enrich Web services described in WSDL[2] (Web Service Description Language) with semantic annotations. However, there is no SAWSDL semantic service matchmaker publicly available to the community yet. To fill this gap, we initially adopt the ideas of semantic Web service matching of our hybrid matchmakers OWLS-MX and WSMO-MX (see [8, 6]), for service description languages OWL-S[3] and WSML respectively, to this environment. A detailed discussion of the SAWSDL specification, particularly addressing the problems arising for semantic Web service selection, is also given.

In this paper, we present the first version of our hybrid SAWSDL Web service matchmaker called SAWSDL-MX. It exploits both crisp logic-based matching (subsumption reasoning) and IR-based (text retrieval) matching. Our preliminary experimental analysis shows, that in line with OWLS-MX and WSMO-MX, hybrid matching can outperform both variants applied stand-alone in terms of recall and precision.

The remainder of this paper is structured as follows. After a brief introduction to SAWSDL and discussion of implied challenges of semantic service selection in

[1] http://www.w3.org/TR/sawsdl/
[2] http://www.w3.org/TR/wsdl/ and http://www.w3.org/TR/wsdl20/
[3] http://www.daml.org/services/owl-s/1.1/

section 2, the hybrid matching approach of SAWSDL-MX is described in detail in section 3. Section 4 presents the architecture and implementation details of SAWSDL-MX. Preliminary results of our experimental evaluation of SAWSDL-MX over a initial test collection SAWSDL-TC1 in terms of recall, precision and average query response time are shown in 5. We comment on related work in section 6 and conclude in section 7.

2 SAWSDL Services

In the following, a brief introduction of the semantically enabled service description language SAWSDL is given. Language specific problems for semantic service discovery arising from the W3C recommendation and methods of resolution and assumptions for avoiding them respectively are also discussed.

SAWSDL is designed as extension of WSDL enabling service providers to enrich their service descriptions with additional semantic information. For this purpose, the notion of *model reference* and *schema mapping* have been introduced in terms of XML attributes that can be added to already existing WSDL elements as depicted in figure 1. More precisely, the following extensions are used for annotation:

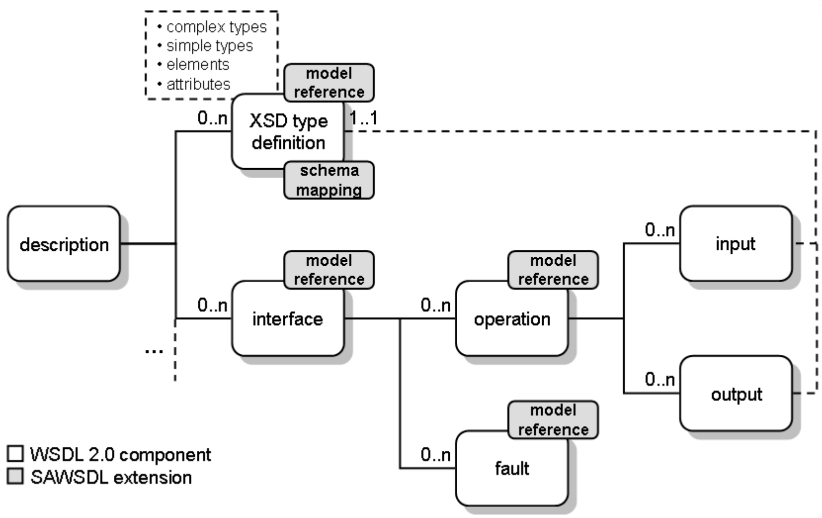


Fig. 1. SAWSDL extensions of WSDL interface components

– *modelReference*: A *modelReference* points to one ore more concepts with equally intended meaning expressed in an arbitrary semantic representation language. They are allowed to be defined for every WSDL and XML Schema element, though the SAWSDL specification defines their occurrence only

in WSDL interfaces, operations, faults as well as XML Schema elements, complex types, simple types and attributes. The purpose of a model reference is mainly to support automated service discovery.
– *liftingSchemaMapping*: Schema mappings are intended to support automated service execution by providing rules specifying the correspondences between semantic annotation concepts defined in a given ontology (the "upper" level) to the XML Schema representation of data actually required to invoke the Web service using SOAP (the "lower" level), and vice versa. A *liftingSchemaMapping* describes the transformation from the "lower" level in XML Schema up to the ontology language used for semantic annotation.
– *loweringSchemaMapping*: The reference tag *loweringSchemaMapping* describes the transformation from the "upper" level of a given ontology to the "lower" level in XML Schema.

Since the specification of SAWSDL does not restrict the developer of a semantic service in SAWSDL to a particular ontology language, any service selection has to cope with the implied semantic interoperability problem of both heterogeneous ontologies and heterogeneous ontology languages. Therefore, as an initial starting point, we restricted our inital SAWSDL service matchmaker to "understand" only the standard OWL[4]. More concrete, we assume for SAWSDL-MX 1.0 that model references in SAWSDL service offers and requests are pointing to ontological concepts exlcusively defined in OWL-DL. That allows to apply standard subsumption reasoning used for OWL-S matchmaking such as in [14, 4, 8]. Besides, there is no retrieval test collection for SAWSDL publicly available yet, but for OWL-S, namely OWLS-TC, which we converted semi-automatically into SAWSDL services such that we could use the resulting SAWSDL-TC for initially evaluating our matchmaker.

Another problem with the SAWSDL specification with respect to service matching is that so-called *top-level annotation* and *bottom-level annotation* are defined as to be considered independent from each other. The term *top-level annotation* describes the case, where a complex type or element definition of a message parameter is described by a model reference as a whole. A *bottom-level annotation* pursues the idea of semantically annotating the parts that are contained inside the definition of a complex type or element. However, it remains unclear how to evaluate matching *between* top-level and low-level annotated parameters, or which one to prefer if both levels are available. To circumvent this problem, we decided to rely on top-level annotations of upper parameter type definitions, and ignore bottom-level annotation in the first version of our matchmaker. In addition to that, element and type definition specifying a message component can be annotated at the same time. The specification does not imply a solution for this case either, so we decided to rely on the annotation directly attatched to the referenced XML Schema object if available.

Further, multiple references to multiple ontologies defined in different languages and formats such as logic theories, plain text documents or structured

[4] http://www.w3.org/2004/OWL/

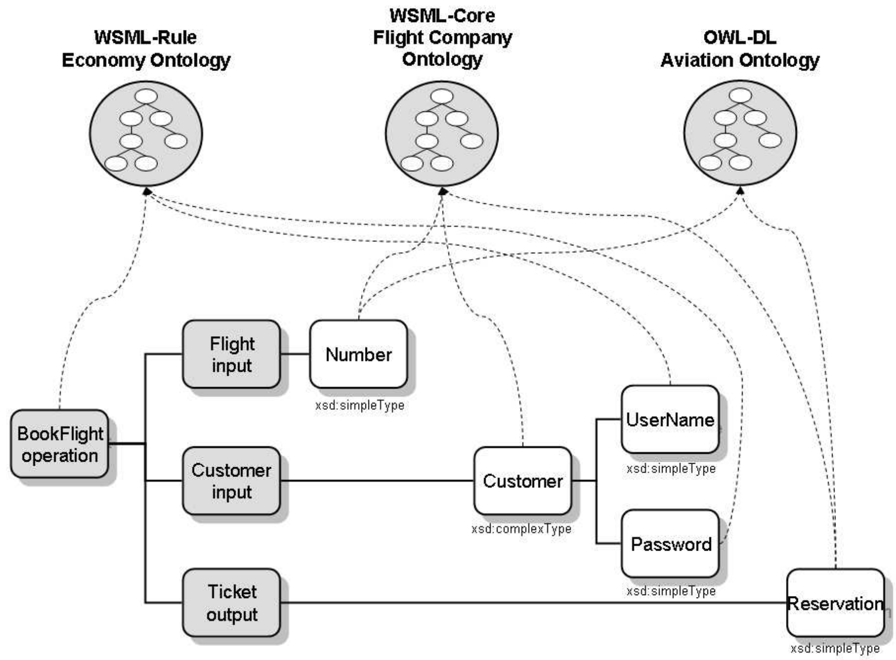


Fig. 2. SAWSDL service example

thesauri can be used to describe the semantic of even the *same* element. Therefore, a matchmaker, in principle, cannot know whether these different types of semantic descriptions of the element are intended to be treated as complementary or equivalent. In the first case, how to aggregate the complementing descriptions, in the latter case, which one to select best for further processing? This opens up a wide range of pragmatic approaches to deal with this for service matching. SAWSDL-MX 1.0 checks only the first model reference of an element. However, different variants dealing with multiple model references connected to a single object are topic of further development, since they are to be treated as sets without order. One possible approach would be to check every combination of request and service offer reference part and perform some kind of aggregation afterwards.

To illustrate this problem by example, consider figure 2: A flight company offers a WSDL Web service with different operations concerning flight booking (*BookFlight* operation), account administration (omitted in the picture), and so on. The *BookFlight* operation is defined to take information of the desired flight (*Flight* input) and customer information in form of a tuple containing a user name and appropriate password (*Customer* input) as input parameters and delivers information about the ticket reservation (*Ticket* output).

To support automated Web service selection, this service is semantically annotated in compliance with the SAWSDL specification as shown in the figure.

In particular, the service developer of the flight company uses WSML-Core, WSML-Rule and OWL-DL concept descriptions for service element annotation. As a consequence, a matchmaker agent cannot perform single language-specific reasoning and matching mechanisms but has to apply an appropriate combination of them instead. This problem can be straight-forwardly solved by use of language mappings available for WSML-Core and OWL-DL[5] but remains hard to solve for comparing concepts in WSML-Rule and OWL-DL.

Further, in the example, the XML Schema description attached to the service input element *Customer* contains annotations for the compound complex type as well as the simple types (referenced by the elements contained in the complex type, element nodes are omitted in the picture). How to handle this situation? Selecting only one annotation level may neglect additional information while looking at all references as a conjunction of ontological concepts can lead to either logical inconsistencies, or is not possible due to incomparable description languages. This problem is exaggerated in the example by providing multiple references (multiple levels of annotations) for the same element. SAWSDL-MX 1.0 only checks the top-level annotation of the most generic element of the XML Schema description of a service parameter.

3 Service Matching with SAWSDL-MX

In the following, we describe one approach to SAWSDL-service selection which we implemented in an initial version of a matchmaker called SAWSDL-MX based on the assumptions stated above. SAWSDL-MX performs service selection in terms of logic-based, syntactic (text similarity-based) and hybrid matching of I/O parameters defined for potentially multiple operations of a Web service interface (signature matching)[6]. As service requests, standard SAWSDL Web service definition documents are used. This approach is particularly inspired by the hybrid semantic service matchmakers OWLS-MX [8] and WSMO-MX [6] for OWL-S and WSML.

3.1 Service Interface Matching

The matching process of SAWSDL-MX on the service interface level is performed as follows. For every pair of service offer O and service request R, every combination of their operations is evaluated by either logic-based matching, text

[5] http://www.wsmo.org/TR/d16/d16.1/v0.21/

[6] For SAWSDL-MX 1.0, we assume only one interface but multiple operations per service. Extending the proposed service matching algorithm to services with even multiple interfaces only requires additionally combined valuation of the respective interface matching results. The restriction to signature matching for SAWSDL-MX 1.0 is due to the fact that, in SAWSDL, preconditions and effects can be added as input and output model references only, which makes it hard for any matchmaker to identify them as such in general, and before actually analyzing the name and content of referenced models in particular.

retrieval-based matching, or both. The matching of operations is described in more detail later.

In order to compute an optimal injective mapping of operations for service offer and request, SAWSDL-MX applies bipartite graph matching, where nodes in the graph represent the operations and the weighted edges are built from possible one-to-one assignments with their weights derived from the computed degree of operation match. If there exists such a mapping, then it is guaranteed that there exists an operation provided by the service offer for every operation a requester defined in her query. That is, there exists no request operation that cannot be provided by the service offer, disregarding the quality of match at this point.

As an example, consider the service request and service offer given in figure 3. Every request operation RO_i (with $i \in \{1, 2\}$) is compared to every advertisement operation O_j (with $j \in \{1, 2, 3\}$) with respect to logic-based filters defined in the next section. In this example, RO_1 exactly matches with O_1, but fails for O_2 and O_3. O_3 is a weaker plug-in match for RO_2 (the subsumed-by match of RO_2 with O_2 is even weaker than a plug-in match). The best (max) assignment of matching operations is $\{\langle RO_1, O_1\rangle, \langle RO_2, O_3\rangle\}$.

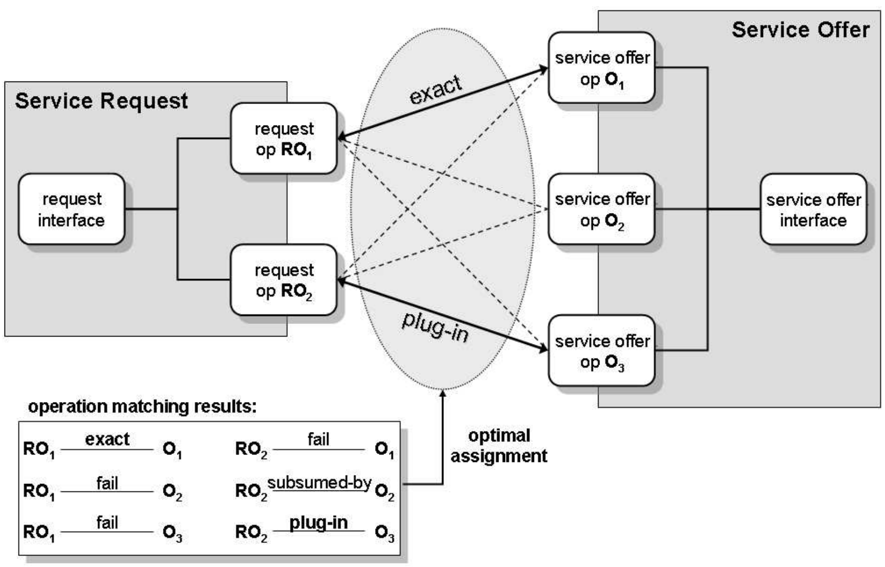


Fig. 3. Interface level matching of SAWSDL-MX

One conservative (min-max) option of determining the matching degree between service offer and request based on their pairwise operation matchings is to assume the worst result of the best operation matchings, to guarantee a fixed *lower* bound of similarity for *every* requested operation. This is what

SAWSDL-MX 1.0 is doing, so in this example shown in figure 3, the service offer is considered a *plug-in* match for the request. Other possibilities are to merge the operation matching results based on, for example, their average syntactic similarity values, and to provide more detailed feedback to the user on the operation matchings involved.

Please note that SAWSDL-MX aims at finding service matches solely based on single service offer documents. The problem of semantic Web service composition is somehow related, but additional state-based planning strategies have to be applied to solve this problem, which is out of the scope of this work. To accomplish on that, a Web service composition planner like e.g. OWLS-XPlan or SHOP2 could be considered (see [18, 19] for details).

3.2 Logic-based Operation Matching

As mentioned above, we assume for SAWSDL-MX 1.0 that model references in SAWSDL service offers and requests are pointing to ontological concepts exlcusively defined in OWL-DL or WSML-DL. That allows to apply standard subsumption reasoning for description logics (see [20]). Therefore, the logic-based operation matching part of SAWSDL-MX computes the degree of logic-based match for a given pair of service offer operation O_O and service request O_R by successively applying four filters of increasing degree of relaxation: *Exact*, *Plug-in*, *Subsumes* and *Subsumed-by*, which are, in essence, adopted from those of OWLS-MX 2.0 but modified in terms of an additional bipartite concept matching to ensure an injective mapping between offer and request concepts, if required. The reason of this modification is that previous experiments with OWLS-MX showed that many logic-based only failures could have been avoided by this additional constraint.

Exact match: Service operation O_O *exactly* matches service operation $O_R \Leftrightarrow$ ($\exists$ injective assignment $M_{in} : \forall m \in M_{in} : m_1 \in in(O_O) \wedge m_2 \in in(O_R) \wedge m_1 \equiv m_2) \wedge$ ($\exists$ injective assignment $M_{out} : \forall m \in M_{out} : m_1 \in out(O_R) \wedge m_2 \in out(O_O) \wedge m_1 \equiv m_2$). There exist a one-to-one mapping of perfectly matching inputs as well as perfectly matching outputs. Assuming that an operation fullfills a requesters need if every input can be satisfied and every requested output is provided, the assignments only require to be injective (but not bijective), thus additional available information not required for service invocation and additional provided outputs not explicitly requested are tolerated.

Plug-in match: Service operation O_O *plugs into* service operation $O_R \Leftrightarrow$ ($\exists$ injective assignment $M_{in} : \forall m \in M_{in} : m_1 \in in(O_O) \wedge m_2 \in in(O_R) \wedge m_1 \sqsupseteq m_2) \wedge$ ($\exists$ injective assignment $M_{out} : \forall m \in M_{out} : m_1 \in out(O_R) \wedge m_2 \in out(O_O) \wedge m_2 \in lsc(m_1)$). The filter relaxes the constraints of the *exact* matching filter by additionally allowing input concepts of the service offer to be arbitrarily more general than those of the service request, and advertisement output concepts to be direct child concepts of the queried ones.

Subsumes match: Service operation O_O *subsumes* service operation $O_R \Leftrightarrow$ ($\exists$ injective assignment $M_{in} : \forall m \in M_{in} : m_1 \in in(O_O) \wedge m_2 \in in(O_R) \wedge m_1 \sqsupseteq m_2) \wedge$ ($\exists$ injective assignment $M_{out} : \forall m \in M_{out} : m_1 \in out(O_R) \wedge m_2 \in$

$out(O_O) \wedge m_1 \sqsupseteq m_2$). This filter further relaxes constraints by allowing service offer outputs to be arbitrarily more specific than the request outputs (as opposed to the *plug-in* filter, where they have to be direct children). Thus, a *plug-in* can be seen as special case of a *subsumes* match resulting in a more fine-grained view at the overall service ranking.

Subsumed-by match: Service operation O_O is *subsumed by* service operation $O_R \Leftrightarrow$ ($\exists$ injective assignment $M_{in} : \forall m \in M_{in} : m_1 \in in(O_O) \wedge m_2 \in in(O_R) \wedge m_1 \sqsupseteq m_2$) $\wedge$ ($\exists$ injective assignment $M_{out} : \forall m \in M_{out} : m_1 \in out(O_R) \wedge m_2 \in out(O_O) \wedge m_2 \in lgc(m_1)$). The idea of the *subsumed-by* matching filter is to determine the service offers that the requester is able to provide with all required inputs and at the same time deliver outputs that are at least closely related to the requested outputs in terms of the inferred concept classification.

At this filtering step, services that offer equivalent or more specific outputs already have been discovered. The *subsumed-by* filter additionally returns service offers that provide more general output concepts, namely direct parents. These may be of value for a user to know, though it depends on the granularity of the matchmaker ontology. For example, it would not make sense to return a service operation providing information on vehicles, if the user explicitly requested information on a very special brand of a car which concept is inappropriately modelled as a direct child of the concept vehicles in the ontology.

The overall algorithm for logic-based matching of operations considers the filters in the following order based on the degree of relaxation: *exact* > *plug-in* > *subsumes* > *subsumed-by* > *fail*. The notion of *fail* applies to cases where none of the filtering tests succeeded.

3.3 Syntactic Operation Matching

In addition, SAWSDL-MX can perform syntactic-based matching based on selected token-based text similarity measures. That is, a syntactic similarity value is computed for every pair of service offer and request operation which is used to rank operations with same logic-based matching degree. The implemented similarity measures for SAWSDL-MX 1.0 are the same as for OWLS-MX, that are the *Loss-of-Information*, the *Extended Jaccard*, the *Cosine* and the *Jensen-Shannon* similarity measures. The architecture of SAWSDL-MX allows the integration of other text similarity measures such as those provided by SimPack[7] which is also used in the iMatcher matchmaker [7].

The weighted keyword vectors of inputs and outputs for every operation are generated by first unfolding the referenced concepts in the ontologies (as defined for standard tableaux reasoning algorithms). The resulting set of primitive concepts of all input concepts of a service operation is then processed to a weighted keyword vector based on TFIDF weighting scheme, the same is done with its output concepts. The text similarity of a service offer operation and a request operation is the average of the similarity values of their input and output vectors according to the selected text similarity measure.

[7] http://www.ifi.uzh.ch/ddis/research/semweb/simpack/

3.4 Hybrid Operation Matching

Inspired by OWLS-MX [8], SAWSDL-MX combines logic-based and syntactic-based matching to perform *hybrid* semantic service matching. There are different options of combination: A *compensative* variant using syntactic similarity measures in cases where none of the logic-based filters applies helps to improve the service ranking with respect to logic-based false negatives by re-considering them again in the light of their computed syntactic similarity. An *integrative* variant deals with problems concerning logic-based false positives by not taking the syntactic similarity of concepts into account only when a logical matching fails, but as a conjunctive constraint in each logical matching filter. Our experiments showed that OWLS-MX 2.0 using the integrative variant performs better than the original one with the complementary use of syntactic similarity. However, SAWSDL-MX 1.0 inherited the compensative variant from OWLS-MX 1.0, that is, only the logic-based subsumed-by filter is modified to a hybrid filter by integrative checking of syntactic simliarity of concepts, and the syntactic nearest-neighbour filter is compensative in the sense that it is only performed in case all other filters fail.

Subsumed-by match: Service operation O_O is *subsumed by* service operation $O_R \Leftrightarrow (\exists$ injective assignment $M_{in} : \forall m \in M_{in} : m_1 \in in(O_O) \wedge m_2 \in in(O_R) \wedge m_1 \sqsupseteq m_2) \wedge (\exists$ injective assignment $M_{out} : \forall m \in M_{out} : m_1 \in out(O_R) \wedge m_2 \in out(O_O) \wedge m_2 \in lgc(m_1)) \wedge sim_{IR}(O_R, O_O) \geq \alpha$. A *subsumed-by* match computed by hybrid matching additionally requires the IR-based similarity computed using one of the measures from $IR = \{LOI, ExtJacc, Cos, JS\}$ to be above a given threshold α. This helps to avoid logic-based false positives to be introduced by the pure logic-based variant of this filter.

Nearest-neighbour match: This filter compensates logic-based false negatives as described above. Its condition is $sim_{IR}(O_R, O_O) \geq \alpha$ and thus considers all services not already catched in previous filter steps whose IR-based similarity is above the threshold.

4 SAWSDL-MX Implementation

SAWSDL-MX 1.0 has been fully implemented in Java using the sawsdl4j[8] API (handling SAWSDL for WSDL 1.1) and the OWL API[9] for access to SAWSDL and OWL files, the DIG 1.1[10] as standard interface to handle SHOIQ knowledge base queries, and the Pellet[11] reasoner as inference engine for logic-based matchmaking.

Figure 4 gives an broad overview of the overall system architecture. Basically, SAWSDL-MX consists of the following components: *SAWSDL Matching Engine*, *Service Registry*, *Ontology Handlers*, *Local Matchmaker Ontology* and *Similarity*

[8] http://knoesis.wright.edu/opensource/sawsdl4j/
[9] http://owlapi.sourceforge.net/
[10] http://dig.sourceforge.net/
[11] http://pellet.owldl.com/

Measures. These are described in more detail in the following. From the perspective of service providers, SAWSDL-MX allows the registration of SAWSDL Web service offers at the service registry. For requesters, SAWSDL-MX provides an interface for submitting queries by means of a SAWSDL document specifying details about the desired service interface. After the service discovery process, the SAWSDL-MX matching engine returns a ranked list of service offers that match the query.

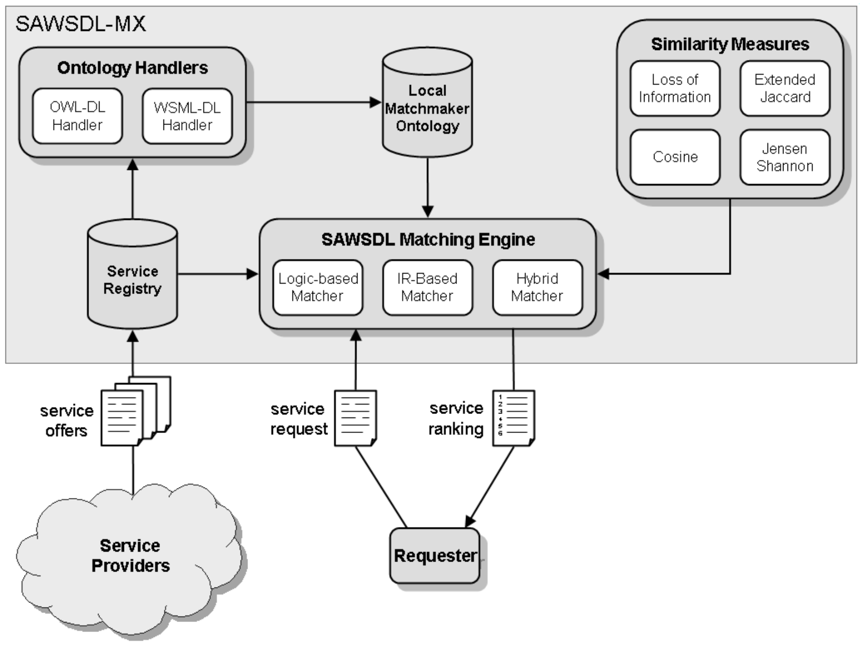


Fig. 4. SAWSDL-MX architecture

SAWSDL Matching Engine: The *SAWSDL Matching Engine* is the core component of SAWSDL-MX. It provides several matching variants of SAWSDL-MX 1.0 as described in previous sections: The *Logic-based Matcher* computes service ranking by means of crisp-logic subsumption reasoning and the logic-based matching filters described in section 3.2. The *IR-based Matcher* produces the ranked results using syntactic similarity measures as described in section 3.3. Finally, the *Hybrid Matcher* performs the combined approach of logic-based reasoning and syntactic similarity comparison as described in 3.4. The matching engine component is designed to provide easy integration of additional matching variants by means of Java interface implementation.

Service Registry: This component is the storage for service offers provided by service providers. It is accessed by the matching engine to produce the ranked results for a query.

Ontology Handlers: After the service registration process, the semantic annotations of a SAWSDL service (by means of model references) are processed using *Ontology Handlers*. Therefore, an appropriate handler able to parse and reason about the referenced ontology is selected and the concepts are stored locally to facilitate logic-based reasoning as well as concept unfolding for IR-based matching at query time. As for the matching engine component, the *Ontology Handlers* package is designed to allow the proper integration of additional knowledge representation formalisms by means of Java interfaces.

Local Matchmaker Ontology: This component is in fact part of the *ontology handlers* in the actual implementation but depicted as seperate component for reasons of clarity. The Local Matchmaker Ontology is a storage for all relevant concepts referenced by registered service offers as proposed in [8]. However, since SAWSDL allows the use of various knowledge representation formalisms, parts of the component relevant for certain ontology handlers are directly covered inside the handlers. In case of our current implementation of SAWSDL-MX, it consists of the Pellet reasoner, which is accessed by handlers able to process description logic based ontology languages via DIG 1.1. Currently, only the *OWL-DL Handler* is actually implemented, but expanding the system to WSML-DL is straight-forward, since they rely on subsets of the SROIQ description language, which is addressed by Pellet[12].

Similarity Measures: This package currently contains the four similarity measures *loss-of-information*, *extended Jaccard*, *cosine* and *Jensen-Shannon*. However, adding more variants for IR-based matching can be easily accomplished again via interfaces. An proprietary document indexing structure based on hash tables is also provided. The integration of additional syntactic similarity measures (e.g. from *SimPack*) and better indexing strategies is intended for following versions of SAWSDL-MX.

5 Evaluation of Performance

The experimental evaluation of the retrieval performace of the first version SAWSDL-MX focuses on measuring its recall and precision based on a first SAWSDL test collection semi-automatically derived from OWLS-TC 2.2[13] using the OWLS2WSDL[14] tool, as there is currently no standard test collection for SAWSDL matchmaking available. OWLS2WSDL transforms OWL-S service descriptions (and concept definitions relevant for parameter description) to WSDL through syntactic transformation. The collection consists of 894 Web services covering different application domains: education, medical care, food, travel, communication, economy and weaponry. For this set of service offers, 26 queries have been selected and relevance sets have been created for each of them. These where subjectively defined as relevant according to the standard TREC definition of binary relevance [16]. As the creation of this test collection has been done

[12] With exception of n-ary datatypes
[13] http://projects.semwebcentral.org/projects/owls-tc/
[14] http://projects.semwebcentral.org/projects/owls2wsdl/

by transforming OWL-S services contained in OWLS-TC 2.2, which provides services containing only one atomic process per description, every SAWSDL advertisement only contains a single interface with a single operation (but possibly multiple I/O's). Therefore and because all automatically derived model references exclusively point to OWL ontologies, this test collection can only be seen as a first attempt towards a commonly agreed testing environment for SAWSDL service discovery and our evaluation has to be considered as preliminary. The performance measures used for evaluation are defined as follows:

$$Recall = \frac{|A \cap B|}{|A|}, Precision = \frac{|A \cap B|}{|B|},$$

where A is the set of all relevant documents for a request and B the set of all retrieved documents for a request. The so-called *F1-measure* equally weights recall and precision and is defined as:

$$F1 = \frac{(2 \cdot Precision \cdot Recall)}{(Recall + Precision)}.$$

We adopt the prominent macro-averaging of precision. That is, we compute the mean of precision values for answer sets returned by the matchmaker for all queries in the test collection at standard recall levels $Recall_i$ $(0 \leq i < \lambda)$. Ceiling interpolation is used to estimate precision values that are not observed in the answer sets for some queries at these levels; that is, if for some query there is no precision value at some recall level (due to the ranking of services in the returned answer set by the matchmaker) the maximum precision of the following recall levels is assumed for this value. The number of recall levels from 0 to 1 (in equidistant steps $\frac{n}{\lambda}, n = 1 \ldots \lambda$) we used for our experiments is $\lambda = 20$. Thus, the macro-averaged precision is defined as follows:

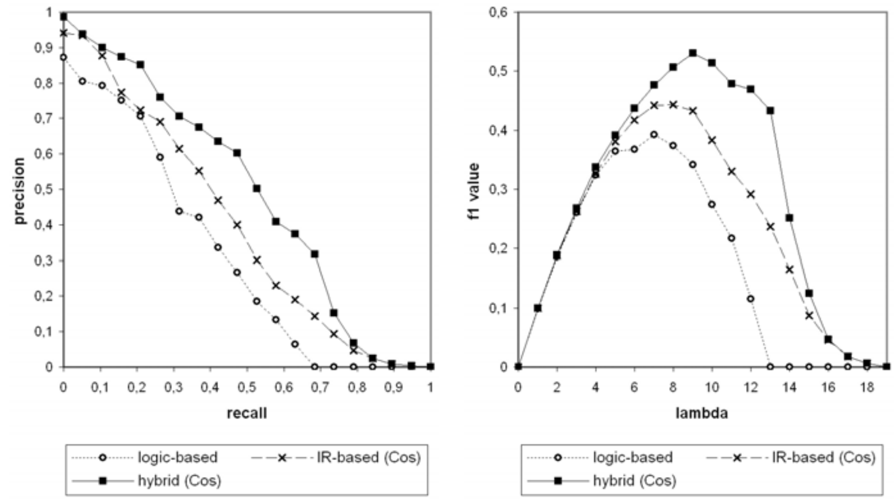
$$Precision_i = \frac{1}{|Q|} \times \sum_{q \in Q} \max\{P_o | R_o \geq Recall_i \wedge (R_o, P_o) \in O_q\},$$

where O_q denotes the set of observed pairs of recall/precision values for query q when scanning the ranked services in the answer set for q stepwise for true positives in the relevance sets of the test collection. For evaluation, the answer sets are the sets of all services registered at the matchmaker which are ranked with respect to their (totally ordered) matching degree.

The performance tests have been conducted on a machine with Windows 2000, Java 6, 1,7 GHz CPU and 2 GB RAM using SME^2 [15] as evaluation environment.

As can be seen in figure 5(a), the *hybrid* variant utilizing *cosine* measure performs best in both finding correct results among the top of the ranking as well as returning positives at high precision towards full recall. It is followed by pure IR-based service discovery (also using *cosine* measure), which is surprisingly at first glance, since it is assumed by the semantic Web community that

[15] http://projects.semwebcentral.org/projects/sme2/



(a) recall/precision (b) F1

Fig. 5. Performance of SAWSDL-MX

semantically enabled ressource retrieval should be able to outperform standard information retrieval purely relying on syntactic information in general. However, as Wang et al. show in [17] exemplarily for OWL, the currently established Web ontology landscape provides mainly poor specification of concepts in terms of the used expressivity of description languages. In fact, many ontologies currently available are just simple taxonomies that do not rely on advanced features provided by for example OWL-DL, thus IR-based matching techniques are often good enough to compare service parameters. The crisp logic-based variant of SAWSDL-MX performs worst with respect to precision. This is mainly due to the problem with ontologies just described and due to the coarse-grained concept descriptions available. Equal consideration of recall and precision using the F1 measure yields the results given in figure 5(b), which recapitulates the observations. Regarding query response times, IR-based matching performs best, namely 1,7 seconds on average per query, while crisp logic-based matching takes 4,7 seconds on average and their combination in hybrid matching is the slowest (6,4 seconds). These evaluation results are in line with the performance of OWLS-MX and WSMO-MX and thus fortify the proposition that hybrid matching outperforms pure logic-based as well as IR-based matching in terms of recall and precision.

6 Related Work

To the best of our knowledge, there exist only very few implemented semantic service discovery systems for SAWSDL. [10] presents a solution to SAWSDL Web service discovery using UDDI registries called FUSION. In FUSION, any service description is classified at the time of its publishing and then mapped to UDDI to allow for fast lookups. In case of unknown semantic service requests reasoning has to be done at query time. In contrast to SAWSDL-MX, each service offer has only to satisfy one matching condition based on subsumption relationships inferred by a reasoner, thus the ranking is not affected by different degrees of logic-based match, neither does FUSION perform a syntactic or hybrid semantic match. Like SAWSDL-MX 1.0, FUSION is strictly bound to OWL-DL, since for each service, a semantic representation in terms of an individual of a predefined OWL concept is constructed. Lumina [11] developed in the METEOR-S project[16] follows a similar approach based on a mapping of WSDL-S (and later on SAWSDL respectively) to UDDI but performs syntactic service matching only. For a survey of semantic service matchmakers in general, we refer the interested reader to [9].

7 Conclusion

SAWSDL-MX performs hybrid semantic Web service matching for SAWSDL operations based on both logic-based reasoning and IR-based syntactic similarity measurement, and combines the results to provide a matching result for service interfaces with multiple operations. The requester formulates queries in terms of SAWSDL service interface descriptions and is presented a service ranking containing service offers from the local registry. The version SAWSDL-MX 1.0 presented in this paper has been implemented and evaluated in terms of recall and precision using a preliminary SAWSDL test collection called SAWSDL-TC1 which we derived from the existing collection OWLS-TC 2.2. As the experimental results show, hybrid matching of SAWSDL services can outperform both logic-based and IR-based matching in terms of precision at the cost of increased average query response time.

We are currently working on several aspects of SAWSDL service discovery and extensions of SAWSDL-MX. As SAWSDL is not restricted to semantically represent service components using a fixed knowledge representation formalism, the integration of additional ontology language support is intended. While description logics have already been discussed for the first version SAWSDL-MX 1.0, the support for languages originating from logic programming such as WSML-Flight and WSML-Rule is subject to our future work.

Besides, inspired by the monolithic logic-based semantic service matchmaker MaMaS [1, 2], we are currently working on an adaptive variant called SAWSDL-MXA which exploits means of ontology patching such as concept contraction and abduction combined with machine learning based on implicit feedback [5].

The semantic interoperability problem induced by the inevitable occurrence of heterogeneous ontologies used for semantic service annotation can be addressed by appropriate ontology alignment techniques [13]. In SAWSDL-MX, one option is to perform an additional matching of concept primitives (that

[16] http://lsdis.cs.uga.edu/projects/meteor-s/

are left undefined in the matchmnaker ontology) in unfolded concepts to be compared using a *shared minimum vocabulary* of requesters and providers like WordNet[17], or by consistent introduction of additional equivalence axioms to the local knowledge base of SAWSDL-MX [12].

SAWSDL-MX 1.0 and SAWSDL-TC1 are both publicly available at *semweb-central.org*.

References

1. Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F. M., Mongiello, M.: Concept Abduction and Contraction in Description Logics. Proceedings of the 16th International Workshop on Description Logics (DL'03), Volume 81 - Sept. 2003
2. Colucci, S., Coppi, S., Di Noia, T., Di Sciascio, E., Donini, F. M., Pinto, A., Ragone, A.: Semantic-Based Resource Retrieval using Non-Standard Inference Services in Description Logics. Proceedings of Thirteenth Italian Symposium on ADVANCED DATABASE SYSTEMS Sistemi Evoluti per Basi di Dati (SEBD-2005), pp. 232-239, 2005
3. Euzenat, J., Valtchev, P.: Similarity-based ontology alignment in OWL-Lite. Proceedings of the European Conference on Artificial Intelligence ECAI, 333-337, 2004
4. Jaeger, M. C., Rojec-Goldmann, G., Liebetruth, C., Mühl G., Geihs K.: Ranked Matching for Service Descriptions Using OWL-S. KiVS 2005: 91-102, 2005
5. Joachims, T., Radlinski, F.: Search Engines that Learn from Implicit Feedback. Computer Volume 40, Issue 8, Aug. 2007 Page(s):34 - 40, 2007
6. Kaufer, F., Klusch, M.: WSMO-MX: A Logic Programming Based Hybrid Service Matchmaker. Proceedings of the 4th IEEE European Conference on Web Services (ECOWS 2006), IEEE CS Press, Zurich, Switzerland, 2006
7. Kiefer, C., Bernstein, A.: The Creation and Evaluation of iSPARQL Strategies for Matchmaking. Proceedings of the 5th European Semantic Web Conference (ESWC), Lecture Notes in Computer Science, Vol. 5021, pages 463–477, Springer-Verlag Berlin Heidelberg, 2008
8. Klusch, M., Fries, B., Sycara, K.: Automated Semantic Web Service Discovery with OWLS-MX. Proceedings of 5th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), Hakodate, Japan, ACM Press, 2006
9. Klusch, M.: Semantic Web Service Coordination. In: M. Schumacher, H. Helin, H. Schuldt (Eds.) CASCOM - Intelligent Service Coordination in the Semantic Web. Chapter 4. Birkhuser Verlag, Springer, 2008
10. Kourtesis, D., Paraskakis I.: Combining SAWSDL, OWL-DL and UDDI for Semantically Enhanced Web Service Discovery. Proceedings of the 5th European Semantic Web Conference (ESWC 2008), Lecture Notes in Computer Science (LNCS), vol. 5021, Springer-Verlag Berlin Heidelberg, pp. 614628, 2008
11. Li, K., Verma, K., Mulye, R., Rabbani, R., Miller, J. A., Sheth, A. P.: Designing Semantic Web Processes: The WSDL-S Approach. Chapter submitted to Semantic Web Processes and Their Applications. J. Cardoso, A. Sheth, Editors. Springer
12. Meilecke, C., Stuckenschmidt, H.: Applying Logical Constraints to Ontology Matching. Proceedings of KI 2007: Advances in Artificial Intelligence: 30th Annual German Conference on AI, 2007
13. Shvaiko, P., Euzenat, J.: A Survey of Schema-based Matching Approaches Journal on Data Semantics, 2005.
14. Sycara, K., Paolucci, M., Ankolekar, A., Srinivasan, N.: Automated discovery, interaction and composition of Semantic Web services. Journal of Web Semantics, vol 1, Elsevier, 2003
15. Toch, E., Gal, A., Reinhartz-Berger, I., Dori D.: A Semantic Approach to Approximate Service Retrieval. ACM Transactions on Internet Technology, 8(1), 2008
16. TREC. Text Retrieval Conference. http://trec.nist.gov/data/.
17. Wang, T. D., Parsia, B., Hendler, J.: A survey of the web ontology landscape. Proceedings of International Semantic Web Conference (ISWC), 2006
18. Klusch, M., Gerber, A., Schmidt, M.: Semantic Web Service Composition Planning with OWLS-Xplan. 1st Intl. AAAI Fall Symposium on Agents and the Semantic Web, Arlington VA, USA, 2005
19. Sirin, E., Parsia, B., Wu, D., Hendler, J., Nau, D.: HTN planning for web service composition using SHOP2. Proceedings of the 2nd International Semantic Web Conference (ISWC), pages 20-23, Sanibel Island, Florida, USA, 2003
20. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider P.F.: The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press 2003, ISBN 0-521-78176-0

[17] http://wordnet.princeton.edu/

Uncovering WSDL Specifications' Data Semantics

George A. Vouros, Alexandros Valarakos, Konstantinos Kotis

AI-Lab, University of the Aegean, Karlovassi, Samos, Greece
{georgev, alexv, kotis}@aegean.gr

Abstract. The work[1] reported in this article aims towards computing web services' data semantics. The paper provides extensive experimental results towards the automatic semantic annotation of WSDL specifications. Specifically, this paper reports on combinations of state-of-the-art methods for automatically mapping the part elements of the WSDL input and output messages to ontology classes: These combinations result to a specific method for Uncovering web Services Data Semantics (USDS). We study the performance of USDS even in challenging cases where lexical items are rather scarce and misleading. Experimental results are being thoroughly discussed, showing the potential and limitations of USDS.

Keywords: semantic annotation, mapping methods, WSDL specification, data semantics

1 Introduction

The automatic discovery of Web services is a highly important service manipulation task. WSDL is mainly focusing on operational and syntactic details regarding the implementation and execution of Web services. The lack of explicit semantics in WSDL specifications makes them insufficient to satisfy the requirements for flexible and effective Web service 'manipulation' tasks, as they force the relevant mechanisms to be based mostly on keyword matches. While we may relate different types of semantics to the web services, we can distinguish two widely recognized types of semantics: (a) Data semantics (introducing the semantic signature of services: semantics of input/output messages of service operations), (b) Functional Semantics (function of operations and of the service itself). We may further add to this list: Protocol semantics, execution semantics, non-functional semantics (security, QoS), and others. This paper focuses on data semantics, which is generally accepted to be one of the most critical aspects regarding services' semantic description. Although other important aspects exist with respect to web services' signature (e.g. goals, pre/post conditions [10], services' classification [2]), mainly due to their strong dependence on data semantics, they have not received the attention that data semantics do.

[1] This work has been supported by the European Commission project Grid4All under grant number FP6, project IST-Grid4All-2006-034567

The main objective of this paper is to study the automatic semantic annotation of WSDL specifications, given ontologies related to the services' domains. Towards this objective, we devise a method for mapping input/output messages' part names to ontology classes and study its performance to variations of the OWL-TC corpus 2 [6]. This method, aiming to uncovering the data semantics of web services (USDS), combines state-of-the-art string similarity methods and vector-based methods, aiming to mitigate difficulties and limitations of other approaches, even in challenging cases. The aim is for services to be automatically translated to semantically enriched specifications, and thus be registered to semantic registries automatically.

As pointed in [11], the process of semantic annotation of services requires input from multiple sources, including the source code of the service, the API documentation and description, as well as "external" information sources such as users' license agreements and sources of background knowledge [2]. The main assumption behind this article is that these information sources, in conjunction to service specification elements, can provide valuable information for the automatic annotation of web services. It is within the goals of this work to reduce human intervention to the subsidiary tasks that humans can do better: introduce descriptive documentation of WSDL specifications and validate results.

This paper is structured as follows: Section 2 states the problem that this article deals with. Section 3 presents related work and motivates the proposed method. Section 4 presents in detail the mapping techniques used and the overall proposed method. Section 5 presents the experimental setup and results, and section 6 concludes the paper, sketching future work.

2 Problem Statement

The problem that this paper deals with is as follows: "*Given (a) a WSDL specification and (b) an ontology O= (S,A), S being the set of terms lexicalizing ontology elements and A the ontology axioms, provide mappings of WSDL messages' part names to ontology classes with respect to the intended semantics of the service"*.

We deal only with services' signatures specified in WSDL (i.e. with data semantics). More specifically we consider the following specifications:

- The signature of a service *s* specifies a set of input/output messages. These are denoted <*service_id, message_type, message_id*>, where *message_type* can be *input* or *output*.
- Each message <*service_id, message_type, message_id*> has one or more parameters: <*service_id, message_id, name, data_type*>, where *name* is the name of the parameter, and *data_type* specifies the (atomic or complex) data type of the parameter.
- Each parameter is associated with textual annotations: <*service_id, message_id, name, annotation_type, text*>, where the type of annotation is *description* or *comment*, and *text* is the actual annotation text. Each parameter may be associated with more than one annotation.

Computed mappings between WSDL messages' part names and ontology elements are of the form <*service_id, message_id,name,element,rel*>where *name* is the name of the parameter, *element* is a term in the ontology signature S, and *rel* is the assessed relation between *element* and *name*: This may be *equivalence*, *subsumes* or *subsumed*. For the purposes of this paper we restrict our attention to the equivalence case.

We have to notice that the above problem statement follows the WSDL 1.1 specification (also addressing the great number of existent services): However, this can be easily restated for WSDL 2.0 specifications considering that no message part names exist.

3 Related Work and Motivation

3.1 Related Work

The work reported in this article, aims to provide services with data semantics by exploiting of domain ontologies, services' WSDL specifications and textual annotations. Close to the aims of this work are efforts that exploit textual descriptions of services for the annotation, classification, and for the assessment of similarities between web-services.

The METEOR-S Annotation Framework [1] is one of the most prominent approaches, aiming at semi-automatically marking up web service descriptions with ontologies, by means of a schema matching algorithm: Ontologies and XML schemata used by WSDL specifications are converted to SchemaGraphs that are compared by computing linguistic similarities between elements and structural similarities between the schemata. For the purposes of linguistic similarity, the proposed algorithm consults WordNet to find synonyms. In this paper we emphasize on the exploitation of textual information for uncovering the semantics of WSDL parts. In contrast to our approach which aims, among others, to compute latent features that explicate the semantics of WSDL parts, METEOR-S exploits "shallow" features concerning XML schema and ontology elements' names: This for instance affects methods' efficacy in cases where polysemous terms appear, or in cases where specifications are at different granularity levels, as far as the conceptualization of the domain is concerned.

The work reported in [2] aims at assessing WSDL specifications' similarity by exploiting the structure of data types and operations of services, as well as the semantics of natural language descriptions and identifiers. Although the aim of this work is to support query-by-example discovery of services, the emphasis on semantic matching given textual descriptions of services, and the exploitation of identifiers' semantics, brings this work close to our work. This approach uses a vector model, exploiting the textual descriptions of services, and consults WordNet to calculate the semantic distances between identifiers of WSDL elements. While semantic matching is restricted to the exploitation of identifiers, identifiers' senses are not disambiguated, making the calculation of semantic distances rather problematic. Disambiguation is a vital task in our research since we aim to match WSDL elements to specific ontology concepts, explicating their semantics.

Aiming to show how content-based approaches can contribute to semantic matching of OWL-S service specifications, OWLS-MX [3], aims to exploit the implicit semantics of any part of OWL-S service description by representing it as a weighted category-index term vector. Index terms are stemmed lexical items from a shared minimal vocabulary. This vocabulary results from the canonical unfolding in an underlying ontology of the words or concepts that exist in text categories that correspond to OWL-S elements (hasInput, ServiceName, TextDescription etc). Although the aims of this work are quite different from our aims, it provides firm evidence towards our conjecture: That the use of textual descriptions of service parts in conjunction to their specifications can help to uncovering their implicit semantics. However, as shown in [4], there are pitfalls to the logic-based and syntactic matchmaking methods (due to granularity of ontology specifications, surjective mapping of concepts, and incomplete coverage of service semantics) that require explicating the semantics of services' input/output parameters, before these are being compared by logic-based matchmakers.

Further evidence is provided by experiments with the ASSAM's annotation wizard [9]. ASSAM casts the problem of classifying operations and datatypes in a Web Service as a text classification problem. The tool learns from Web Services with existing semantic annotations. Given this training data, a machine learning algorithm can generalize and predict semantic labels for previously unseen Web Services. The approach described in the current paper does not require the pre-existence of semantic annotations to decide the mapping of WSDL elements to ontology concepts.

3.2 Motivation and overview of the method

Our work is being motivated by the view that, for web-service matchmakers to perform accurately, they need the precise, intended meaning of web services' signature in a fine-grained way: This means that parts of web service messages must be mapped to specific terms that are being axiomatized in a formal ontology.

To do so, we need to mitigate pitfalls related to phenomena concerning synonym terms, homonym terms (i.e. polysemy), typographic variations of terms, differences between the granularity of services and ontological specifications.

In conjunction to these pitfalls, we need to take also into account the "nature" of WSDL specifications, which are being produced from program code, with few and, in most of the cases, misleading comments, descriptions, and "tricky" names of the parameters being involved, with improper or faulty use of domain terminology.

To mitigate these pitfalls and avoid difficulties that are inherent to the specifications of web services, we employ the combination of different methods towards a system for uncovering the data semantics of web services (USDS): The core configuration of this system comprises two state of the art methods: COCLU [8], and LSA-based-mapping [13]. Specifically, COCLU is expected to tolerate typographic variations of terms, assessing similarities between terms whose appearance is quite similar, even if one of them is an abbreviation or a concatenation of the other term parts: These are variations that edit-distance measures are hard to capture.

The LSA-based-mapping aims to mitigate problems concerning synonym and homonym terms, as its aim is to disambiguate the meaning of web service parameters, mapping them to WordNet senses that best capture their intended meaning, according to their own lexicalization, their associated types, as well as according to descriptions and comments given. The same method maps ontology classes to WordNet senses that are assessed to capture the human-intended meaning of the formal specifications: Having done these mappings, and in case the intended meanings (WordNet senses) of an ontology class and of a service parameter are related, then these can be mapped. More specifically, in case the corresponding WordNet senses coincide, or they are related via a synonym relation, then the method assesses an exact match. In case their corresponding terms are being related via a hyponym relation, then given that there is a short distance between them (in terms of the number of hyponym/hyperonym relations between them) then the method may assume that there is a subsumption relation between the parameter term and the ontology class. Doing so, the method facilitates tackling problems related to having specifications (WSDL and ontology) at different granularity levels, and relating elements with a “semantic distance”. However, in this paper we only consider cases of exact matches.

To evaluate USDS, we provide extensive experimental results with different configurations of string-matching based and vector-model based methods in different sets of WSDL specifications.

4 Semantic Annotation of WSDL specifications

4.1 Annotating WSDL

As pointed out, we consider that the overall semantic annotation of WSDL specifications comprises three distinct stages: The annotation stage, the mappings stage, and the validation stage. However, for the purposes of this paper we require human intervention only in the annotation stage.

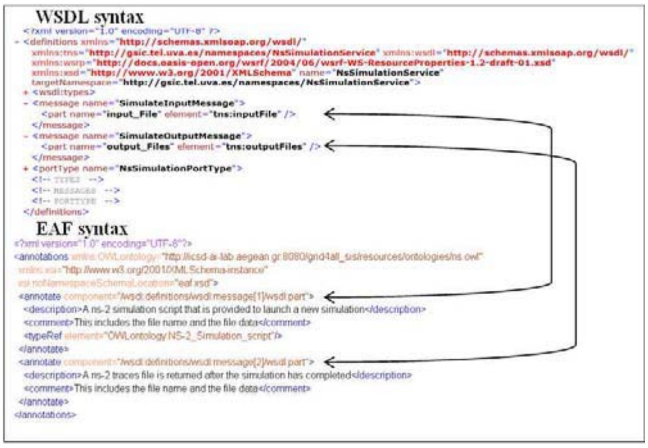


Fig. 1 Annotating WSDL specifications

During the annotation stage, humans provide textual descriptions for elements of the WSDL specification. The annotation stage takes as input the WSDL specification and produces an external xml annotation file (EAF) based on a specific annotation schema that we have specified for this purpose. This is an annotation-template file that provides “slots” for the description of WSDL elements: For the service itself, for each interface, operation and input/output messages’ elements it provides elements for “comments”, “description”, as well as support for mapping mechanisms between WSDL elements and ontologies. As Fig.1 shows, the EAF is aligned with the WSDL specification via XPATH expressions. The annotation schema can be extended for supporting other types of textual information that are necessary for the annotation of WSDL elements.

Although we plan to incorporate SAWSDL [5] into our framework, we do not commit to the use of SAWSDL at this stage, emphasizing mostly on the use of textual descriptions/comments for WSDL elements.

4.2 The USDS Semantic Annotation Method

The USDS semantic annotation system takes as input a WSDL document together with the corresponding xml annotation file, as well as a domain ontology. As already specified in section 2, the output of this system is a set of assessments concerning exact mappings between messages’ part names and ontology classes. USDS, in addition to the part names of the input/output messages, exploits services’ types’ specifications, as well as textual descriptions and comments in the annotation file.

As already pointed out, the core of USDS comprises the combination of two state of the art methods: COCLU, and LSA-based-mapping. COCLU is a compression-based clustering algorithm (COmpression-based CLUstering). The algorithm is based on the assumption that different lexicalizations of a term (typographic variants) use a common set of `core' characters. Therefore, typographic variants that `mostly' use this set are potential alternative lexicalizations of the same concept, while those ones that are `far' from this set are potentially related to the lexicalization of a different concept. Further details for this model and partition-based clustering algorithm are provided in [8].COCLU has been used for the comparison of (a) the description of each WSDL input/output message part, with the labels of ontology classes, (b) the elements in the atomic/complex types specifications of messages’ part elements with the name and labels of ontology classes, and (c) the comment of each WSDL input/output massage part with the comments of ontology classes.

The LSA-based-mapping [13] aims to disambiguate the meaning of WSDL messages’ part names, mapping them to WordNet senses that best capture their intended meaning, according to their lexicalization, their associated types, as well as according to descriptions and comments given. As already said, the same method maps ontology classes to WordNet senses that capture the human-intended meaning of the formal specifications: Having done these mappings, and in case the intended meanings (WordNet senses) of a class and of a service parameter coincide, then these are mapped. The Latent Semantic Indexing (LSI) method assumes that there is an underlying latent semantic space that it estimates by means of statistical techniques using an association matrix ($n \times m$) of terms-documents: Documents in our case correspond to WordNet senses. Terms are selected from the vicinity of WordNet senses (i.e. from the senses themselves and from their hyponym/hypernyms). Latent Semantic Analysis (LSA) computes the arrangement of a k-dimensional semantic space to reflect the major associative patterns in the data. This is done by deriving a set of k uncorrelated indexing factors, which may be considered as “latent concepts”. Then, each term and document is represented by its vector of factor values, indicating its strength of association with each of these “latent concepts”. By virtue of dimension reduction from the N terms space to the k factors space, where k<N, terms that did not actually appear in a document may still end up close to the document, if this is consistent with the major patterns of association in the data.

When one searches an LSI-indexed database of documents, it provides a query (i.e. a pseudo-document), which is a list of terms. The similarity between two documents is computed by means of the dot product between the corresponding representation vectors. Doing so, LSI returns a set of graded documents, according to their similarity to the query.

In our case the semantic space is constructed by terms in the vicinity of the senses S1, S2,…Sm of the WordNet entry matching an ontology class name C, or a WSDL message part name C. The set of terms in the semantic space include:

- The term C’ that corresponds to C. C’ is a lexical entry in WordNet that is a linguistic variation of C.
- Terms that appear in C’ WordNet senses S1, S2,…Sm.
- Terms that constitute hyperonyms / hyponyms of each C’ sense.
- Terms that appear in hyper(hyp)onyms of C’ senses.

As far as ontology classes is being concerned, each query is being constructed by extracting terms from the label, and the comment of an ontology class, as well as from the names of its properties’. Concerning the WSDL specifications, for each input/output message part element we extract terms from its name attribute, from the names of the elements defined in the complex types of the type attribute, as well from its related annotation elements (description and comments). Specifically, terms may result from the tokenization of phrases and may be either simple terms, or compound terms.

To combine the above mentioned methods we have used the AUTOMS-F framework [7]: Specifically, in our implementation the combination of methods produces the union of the mappings produced by each of them. This improves the recall of the final method, but it may result to less precise results. Although more sophisticated types of methods’ combinations have been tested, these have given less encouraging results: Future work concerns the thorough investigation of these techniques.

Two additional string-matching based methods have been also employed. These methods compare the names of ontology classes and the part names of the service messages. Comparisons take place for every potential mapping pair. The better matches (those with the largest similarity values) are selected for every WSDL part. The Exact String Matching (ExactString) identifies a match if the compared names are exactly the same. This allows us to show the “difficulty” of our cases, given that such a simple method fails to exhibit effective performance.The Levenshtein matching method incorporates a distance measure that specifies the minimum number of operations needed to transform one string into another. The valid operations are: insertion, deletion, or substitution of a single character. Our implementation considers a match between two strings if and only if at most two operations are required for their transformation.

In addition to the above, we have also used a Vector Space Model – based (VSM) method, which computes the matching of documents pairs. Each document is represented by a vector of n weighted index terms. Index terms correspond to the simple terms that are extracted from all documents. Here we construct (pseudo-) documents that correspond to ontology classes and WSDL messages' part elements: As it is done in the LSA-based method, these pseudo-documents include terms from the label, and the comment of an ontology class, as well as from the names of its properties'. Concerning the WSDL specifications, for each input/output message part element we extract terms from its name attribute, from the names of the elements defined in the complex types of the type attribute, as well from its related annotation elements (description and comments). The VSM-based method builds the vector of a pseudo-document by assigning to the weight of a term the frequency of its appearance in the document. The similarity between two vectors (each corresponding to a WSDL message part name and to an ontology class) is computed by means of the cosine similarity measure. This computation ranges in [0, 1]: A threshold (currently set to 0.35) is set for deciding when a match occurs.

5 Experiments and Discussion

5.1 Experimental Setting

To evaluate our approach, we have used the OWL-S Service Retrieval Test Collection (OWLS-TC) version 2 [6]. From the OWLS-TC collection, we have translated to WSDL a subset of 87 services, due to problems we faced with OWL-S-to-WSDL translation, and due to duplicate WSDL part elements. In total, we have been experimenting with 5 different domains and 6 different domain ontologies, which result to several different sets of experiments, with an initial given varied degree of difficulty. Additionally, we have produced additional experiments by creating variations of the corpus in order to test the robustness of the USDS different configurations.

Table 1. Information about the experimental domains, ontologies and services.

Domain	Domain Ontology (.owl)			# Services (WSD)	Total # of part elements.	Distinct part elements to be annotated
	Ontology	#concepts	#properties			
(1): Travel	Travel	34	6	9	22 (4)	10
	Portal	171	104	9	27 (4)	17
(2): Education	Books	60	11	8	19 (4)	19
	Portal	171	104	31	63 (3)	63
(3): Economy	Books	60	11	11	31 (4)	19
	Concept	17	3	11	31 (4)	12
(4): Weapon	SUMO	613	208	3	7 (3)	7
(5): Medical	Hospital Physician	64	45	5	39 (10)	39

Table 1 summarizes information concerning the initial sets of experiments and characteristics of the services and ontologies used. The first and the second column present the domain and the ontology used. The third and fourth columns provide information concerning the number of concepts and properties in each ontology. The fifth column provides the number of services (WSDL specifications) that exist in each set, while the sixth one presents the total number of part elements that exist in a set, and the maximum number of part elements that exist in a WSDL specification. The last column specifies the total number of the distinct part elements that should be annotated in each set of experiments.

Concerning WSDL specifications for domains 1 to 4, message part names are mainly composed by a single-word term capitalized and an underscore character as a prefix (e.g. _COUNTRY). This variation slightly differentiates part names from ontology class names. For the domain 5, the messages part names are composed by multi-word terms, either separated with an underscore or with no separator, or using a combination of these (e.g. GetPatientMedicalRecords_AuthorizedMedicalRecords). Such terms are not included in the related domain ontologies, however, their substrings match to ontology class names (e.g. MedicalRecords). We handle individual, distinct terms of multi-word terms separately, only in cases these are separated by an underscore separator, which is one of the most generic case considered.

The characteristics of the domain ontologies are important for the experiments: These include the size of the ontologies, the annotation of classes/properties, i.e. labels and comment annotations, as well as the richness of specifications for each class/property, i.e. the number of subclasses or related properties, depth of hierarchy for each class, etc. Apart from SUMO, which is an upper, widely-accepted ontology, ontologies accompanying the OWL-TC corpus have been developed independently from OWL-TC. Our analysis for these ontologies shows that there are ontologies with some annotations for their classes (e.g. the Travel ontology provides only comments for the defined classes, and the Portal ontology provides comments or labels for some of the classes), and ontologies with no annotations at all (e.g. the HospitalPhysician ontology). As far as richness of specifications is concerned, there are ontologies whose elements are mildly interrelated, such as SUMO (max parents: 3, mean parents: 2, max siblings: 15, mean siblings: 7, all properties have a domain and range specified), but there are others not so rich, such as the HospitalPhysician (max parents: 1, mean parents: 1, max siblings: 8, mean siblings: 4) or the Books ontology (none property have a domain and range specified).

WSDL part names have been manually annotated by human annotators that have adequate knowledge of the related domains. They have been advised to carefully choose the annotations in order to indicate as close as possible the intended meaning of the input/output message parts annotated. Where possible, annotators have been advised to get feedback from xsd-schema complex types included in the WSDL specifications. More specifically, we can identify 3 different annotation cases that have been applied in the corpus: (a) Annotations that are formed by "*free text including terms from xsd-schema types <u>and</u> from the WSDL message part name element*". (b) Annotations that are formed by "*free text including terms <u>only</u> from the WSDL message part name element*". (c) Annotations that are formed by "*a single term*". Annotators choose a single term without considering xsd-schema types or WSDL message part name element information. For instance, for the "*wsdl:part name="Capital_City"*", the annotator creates the description annotation "*<description> Capital </description>*", which is the intended meaning (or synonym) of the entity "Capital City".

The result of this process is a set of annotations with terms belonging in one of the following three categories: a) terms from xsd-schema types, b) terms from WSDL message part names, or c) terms chosen by human annotators.

The use of "free text" in combination with xsd-schema types' terms and/or WSDL message part names' terms, means that the annotator is allowed to form a natural language sentence: E.g. "*<description> The service requests accommodation using country information </description>*". Although the use of free text may distract the matching methods, the freedom that the approach gives to the annotator is important and realistic. In this example of annotation, the annotator has combined terms (e.g. "country") from messages' part names (case b). As another example, in the comment "*<comment> The country name is a string. A country is described with its capital, its currency, and its government </comment>*" the annotator has included terms (terms "capital", "currency", "government") from the xsd-schema type that corresponds to the specific message part name that is annotated (case a). Table 2 summarizes information concerning annotations per domain. It must be noticed that all annotations contain 5 to 7 "significant" terms ie. non stop-words that may drive the computation of the intended mappings. In addition to the above, motivated by our experience with vector-model-based ontology alignment methods (perform better with ontologies that have rich information, i.e. with ontologies that contain labels and comments for every ontology class), we artificially enriched the annotation information of ontology classes with textual information. All experiments ("*enriched ontologies*" cases) have been run with these "enriched" ontologies.

Table 2. Information concerning annotations per domain

Domain Id	Annotation case	Terms that match a class name
1	Descriptions: case (b), Comments: case (a)	Approx. 50%
2	Descriptions: case (b), Comments: case (a)	NA
3	Descriptions: case (b), Comments: case (a)	Approx. 70%
4	Descriptions: case (b), Comments: case (a)	Approx. 40%
5	Descriptions: case (c), Comments: case (a)	Approx. 70%

Furthermore, we have modified the WSDL specifications by replacing the name of their input/output message part elements with a unique random string which ranges in length between 9 and 11 characters. These new specifications double the number of experiments that have been contacted. Such a setting aims to unveil the importance of the natural language descriptions of the input/output message parts for our approach.

5.2 Results and Discussion

Due to space limitations, we present the precision and recall for all experiments contacted with the annotated WSDL specifications and the enriched ontologies (Fig. 2): These provide the best results for all experiments' configurations. Figure 2 shows that (as it was expected) the higher recall is achieved by the combination of all methods. The recall of an individual method for a specific domain/ontology pair may not be high due to the characteristics of the specific domain/ontology pair (e.g. due to the compound terms in name values of the WSDL part elements). For instance, the recall for the Domain5/hospitalPhysician.owl pair is low even for combined methods (e.g. LSA+COCLU). So even if the composition of the “core methods” of USDS (LSA+COCLU) achieves recall equal to 0.8 for all the domains (in average), it achieves 0.1 for this specific domain/ontology pair.

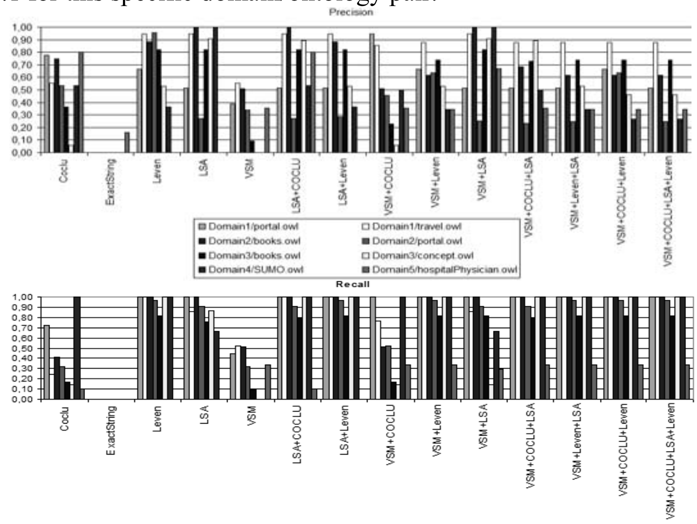


Fig. 2. Precision and recall for different USDS configurations using annotated WSDL specifications and enriched versions of ontologies.

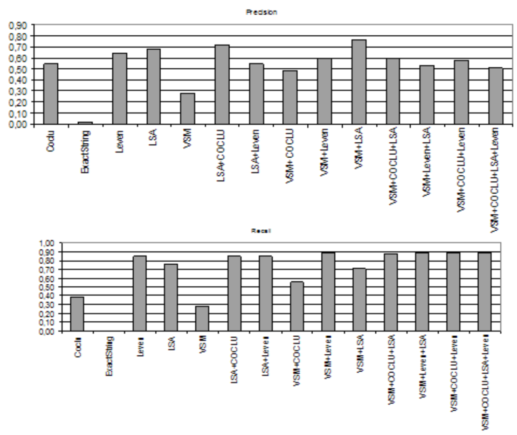


Fig. 3. Average precision and recall for each configuration

The higher precision is achieved by the composition of LSA and VSM. Specifically, the VSM+LSA configuration gives the higher average precision for all the domains. This is shown in Fig. 3. However this configuration achieves low precision in cases that most of the other methods do so as well. For instance, this is the case for the Domain2/portal.owl pair shown in Fig. 2, where most of the other methods achieve precision equal to 0.3. It seems that some methods are much better for specific ontologies. This can shape the conjecture that the performance of the method chosen for the automatic annotation of services is closely related to the choice of the domain ontology used. This may also help us to reach a criterion for the selection of the most "proper" ontology to annotate a service. However this is part of our future work.

The exact matching method fails to deliver valid results in almost any set of experiment, as it was expected. Amongst the other two string matching methods, Levenshtein (Leven) outperforms COCLU in all sets of experiments (Fig. 3). This is mainly due to the existence of compound terms with many different characters.

It must be noticed that the precision achieved by string matching techniques is larger than this achieved by the individual VSM method. This is due to the fact that VSM is being misled by the artificial annotations created for the OWL classes. This process adds one or two terms to a class's virtual document, which may lead the VSM method astray. This supports the argument that ontology classes should be commented with rich and accurate natural language annotations for this method to perform adequately. However this adds extra effort to the ontology developers. As far as the LSA-based method is concerned, it must be noticed that WSDL annotations that include descriptions irrelevant to WordNet senses distract in most of the cases this method from finding the correct mappings due to the inclusion of non-relevant terms.

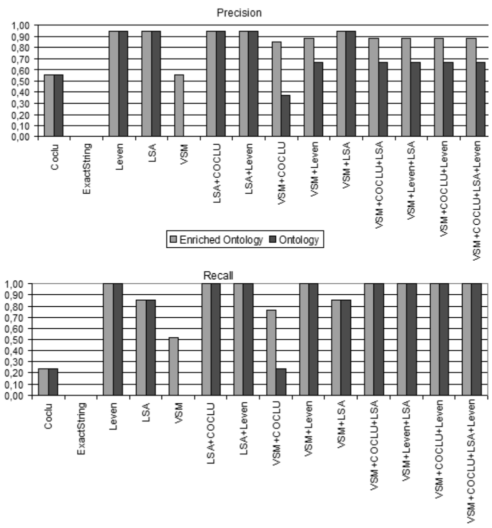


Fig. 4. Precision and Recall for all configurations in experiment Domain1/travel.owl with an artificially enriched version of the ontology

As far as the enrichment of ontologies with informative labels and comments is concerned, as it is depicted in Fig. 4 for the case concerning the Domain 1/travel.owl - a representative case -, the enrichment affects the precision and the recall of the USDS configurations that use virtual documents in the computation of their similarity function. Such an experiment indicates that state-of-the-art mapping methods that are used for the automatic semantic annotation of WSDL documents are more effective when domain ontologies are rich with descriptive information.

Concerning the cases where random strings replace part element names, the lexical mapping methods cannot perform effectively for obvious reasons: strings such as "1qazxsw2" cannot be directly mapped to an OWL class. Even the LSA method that relies on WordNet cannot perform well. This is due to the fact that part names do not have WordNet entries. In contrast to this, the VSM-based mapping method that does not rely on an external lexicon, can overcome this problem resulting in a mildly good performance. This is shown in the experiment configuration Domain3/Books.owl depicted in Fig. 5. This shows that the adequate and rich annotation of WSDL specifications is vital to the effective performance of the most adequate mapping methods. This is further evidenced by the results provided by VSM in the domain 5, where the “complexity” of the terms involved provides major obstacles to the other methods.

As a final conclusion of the above results, we can state that the USDS configuration that results from the composition of different complimentary methods seems to be the most promising one.

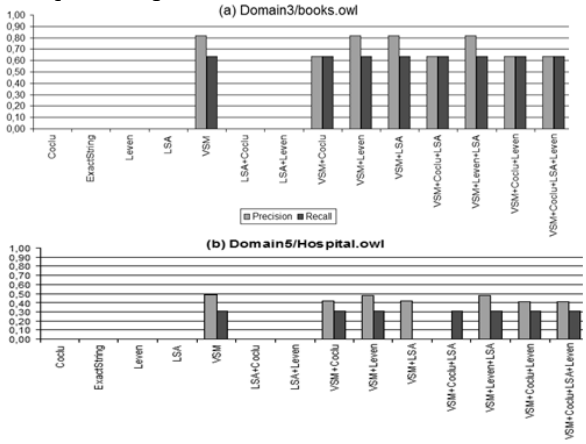


Fig. 5. Precision and Recall for experiments (a) Domain3/books.owl and (b) Domain5/HospitalPhysician.owl with random strings (e.g. “1qazxsw2”) as part names

5.3 Top-*k* evaluation

Although we have so far presented a quite extensive set of experiments showing the potential of the approach to support the effective automatic annotation of WSDL specifications, we have in addition measured the effectiveness of the approach to providing assistance to human annotators [17]. Therefore we have measured the accuracy of the approach. Results are presented and discussed in the following paragraphs.

The top-*k* evaluation method has been used for the evaluation of Web information retrieval approaches (e.g. [14, 15]) and also in the evaluation of semantic services' matchmaking systems (e.g. [16]). The top-*k* (or topK) evaluation method measures the percentage of correct results in the top *k* results in a list of ranked results (accuracy of returned ranked results). In our case, WSDL part names are matched against ontology classes, and the most relevant pairs are returned as suggested mappings, together with their ranking. The higher ranked pairs of each pair are being assessed to provide the “correct” mappings. In case of conflict, the method that provided the first mapping wins. Of course this is not a proper policy. Rankings of different methods need to be normalized appropriately so as to rich a global list of ranked pairs. However, even in this case, a sophisticated policy for reconciling conflicts is necessary: This is something to be further investigated. Therefore, top-*k (accuracy)* evaluation method identifies if the "correct" *pair (i.e. the one provided by the gold-mappings)* is among the *k* ranked suggested mappings. *Accuracy* is a widely accepted metric in the Information Retrieval community and has been successfully used for the evaluation of many systems. An example of related work concerning the support of human annotators is reported in [17]. The work is more focused on the simplification of the manual annotation task of services rather than on the validation of top-*k* ranked semantic annotations. Furthermore, the work facilitates the detecting of mistakes in existing annotations, discovering however a relatively small number of annotations.

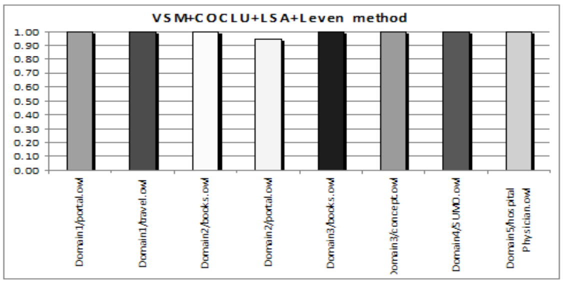


Fig. 6. Top-3 precision for VSM+COCLU+LSA+Leven configuration

The VSM+COCLU+LSA+Leven USDS configuration has produced the best precision/recall percentages in top-*1* experiments. Although recall was really high, the precision of the method can be further improved to support full-automation of the annotation process. Fig. 6 shows the top-*3* precision of this method. As it is shown, the top-*3* precision is 100% for 7 of the domain/ontology pairs and 95% for the Domain2/portal.owl pair. The method failed to retrieve the correct ontology class for a specific part name, although some of its constituent methods successfully retrieved it. This is due to the restriction in the number of retrieved ontology classes in combination with the absence of a global normalisation method amongst the match values produced by each method.

6 Concluding Remarks

Automatic service registration, to facilitate web services' semantic matchmaking, requires efficient and effective semantic annotation of services. In this paper we conjecture that this is possible when WSDL specifications are adequately annotated with textual annotations, and in cases where the suitable mapping method(s) are selected for mapping WSDL elements to ontology classes. Since the efficiency of a mapping method is influenced by the characteristics of the ontology(ies) and the WSDL annotations used in task, their careful selection and composition lead to better performance. In an application context, where users are being advised for the annotation of WSDL parts, the method resulting from the combination of lexical and vector-based methods (Coclu, VSM, LSA, Leven) proved to be extremely valuable since it manages to return with high accuracy the "proper" class for annotating each of the WSDL message parts among the 3 highly ranked matching pairs.

In this paper we have presented extended experimentation cases with different configurations of the USDS method for uncovering web services' data semantics. We have reached a configuration where different complimentary methods achieve very good performance is a set of experiments of varying difficulty. Future goals that have been indicated throughout the sections of this article provide the many different facets of future work that this work entails.

References

1. Oundhakar, P. S., Sheth, A., Verma, K.: METEOR-S Web Service Annotation Framework, WWW 2004, 553-562 (2004)
2 Stroulia, E., Wang, Y.: Structural and Semantic Matching for Assessing Web-Service similarity, IJCIS 5:14, 1-30 (2004).
3. Klusch, M., Fries, B., Khalid, M., Sycara, K.: OWLS-MX: Hybrid OWL-S Service Matchmaking, AAAI (2005).
4. Klusch, M., Fries, B. Hybrid OWL-S service retrieval with OWLS-MX: Benefits and Pitfals, SMR2, 2007.
5. SAWSDL, http://www.w3.org/2002/ws/sawsdl/ (2002)
6. OWLS-TC, http://projects.semwebcentral.org/projects/owls-tc/
7. Valarakos, A., Spiliopoulos, V., Kotis, K., Vouros, G.: AUTOMS-F: A Java Framework for Synthesizing Ontology Mapping Methods, I-KNOW 2007 Special Track on Knowledge Organization and Semantic Technologies 2007, September 5, (2007)
8. Valarakos, A., Paliouras, G., Karkaletsis, V., Vouros, G.: A Name Matching Algorithm for Supporting Ontology Enrichment, In Proceedings of the 3rd Hellenic Conference on Artificial Intelligence (SETN04), LNAI, Vol. 3025, pp. 381-589, Springer-Verlag, Samos, Greece (2004)
9. Heß, A., Johnston, E., Kushmerick, N.: ASSAM: A Tool for Semi-Automatically Annotating Semantic Web Services, ISWC 2004, LNCS 3298, 320–334, (2004).
10. Keller, U., Lara, R., Lausen, H., Polleres, A., Fensel, D.: Automatic Location of Services, European Semantic Web Conference (ESWC'05) (2005)
11. Ringelstein, C., Franz, T., Staab, S.: 'The Process of Semantic Annotation of Web Services', In: J. Cardoso (ed.) (2007) Semantic Web Services - Theory, Tools, and Applications, Idea Publishing Group, USA, (2007)
12. OWL-S API, http://www.mindswap.org/2004/owl-s/api/ (2004)
13. Kotis, K., Vouros, G., Stergiou, K.: Towards Automatic Merging of Domain Ontologies: The HCONE-merge approach. Elsevier's Journal of Web Semantics (JWS), vol. 4:1, pp. 60-79 (2006)
14. Liu, J., Dong, X., Halevy, A.: Answering Structured Queries on Unstructured Data. WebDB '06 Chicago, Illinois USA (2006)
15. Dong, X., Halevy, A.Y., Madhavan, J., Nemes, E., and Zhang, J. "Similarity Search for Web Services". *Proceedings of the Thirtieth International Conference on Very Large Data Bases (VLDB)* (pp. 372-383), Toronto, (2004)
16. Basters, U. and Klusch, M.: RS2D: Fast Adaptive Search for Semantic Web Services in Unstructured P2P Networks. 5th International Conference of Semantic Web (ISWC 2006), Athens, GA, USA (2006)
17. Belhajjame, K., Embury, S. M., Paton, N. W., Stevens, R., and Goble, C. A.: Automatic Annotation of Web Services based on Workflow Definitions, 5th International Conference of Semantic Web (ISWC 2006), Athens, GA, USA (2006)

Evaluating Semantic Web Service Matchmaking Effectiveness Based on Graded Relevance

Ulrich Küster and Birgitta König-Ries

Institute of Computer Science, Friedrich-Schiller-University Jena
D-07743 Jena, Germany
ukuester|koenig@informatik.uni-jena.de

Abstract. Semantic web services (SWS) promise to take service oriented computing to a new level by allowing to semi-automate time-consuming programming tasks. At the core of SWS are solutions to the problem of SWS matchmaking, i.e., the problem of comparing semantic goal descriptions with semantic offer descriptions to determine services able to fulfill a given request. Approaches to this problem have so far been evaluated based on binary relevance despite the fact that virtually all SWS matchmakers support more fine-grained levels of match. In this paper, a solution to this discrepancy is presented. A graded relevance scale for SWS matchmaking is proposed as are measures to evaluate SWS matchmakers based on such graded relevance scales. The feasibility of the approach is shown by means of a preliminary evaluation of two hybrid OWL-S matchmakers based on the proposed measures.

1 Introduction

In recent years, semantic web services (SWS) research has emerged as an application of the ideas of the semantic web to the service oriented computing paradigm [1]. The grand vision of SWS is to have a huge online library of component services available, which can be discovered and composed dynamically based upon their formal semantic annotations. One of the core problems in the area concerns SWS matchmaking, i.e. the problem of comparing a set of semantic service advertisements with a semantic request description to determine those services that are able to fulfill the given request. A variety of competing approaches to this problem has been proposed [2]. However, the relative strengths and shortcomings of the different approaches are still largely unknown. For the future development of the area it is thus of crucial importance to establish sound and reliable evaluation methodologies. The recent formation of international SWS evaluation campaigns[1] is a promising step in this direction.

One of the core problems of SWS matchmaking is that it is unrealistic to expect advertisements and requests to be either a perfect match or a complete

[1] Semantic Web Service Challenge: http://sws-challenge.org
S3 Contest on Semantic Service Selection:
http://www-ags.dfki.uni-sb.de/∼klusch/s3/

fail. Thus, virtually all SWS matchmakers support multiple degrees of match, i.e. they classify the set of advertisements into a hierarchy of different match levels or even assign a continuous degree of match to each offer. Nevertheless, existing approaches for the evaluation of the retrieval effectiveness of matchmaking approaches have so far been based exclusively on binary relevance, i.e. for evaluation purposes an advertisement is considered to be either a match or not, but no further distinction is made. This is a remarkable discrepancy that may distort evaluation results and compromise their reliability. This paper presents an approach to overcome this problem.

The rest of the paper is structured as follows. In the following Section, we provide information about related previous work. In Section 3, we discuss the notion of relevance in the domain of SWS matchmaking and propose a graded relevance scale customized to this domain. In Section 4, we introduce a number of evaluation measures capable to deal with graded relevance. In Section 5, we report on a preliminary experiment on applying the graded relevance scale and the evaluation measures to evaluate two OWL-S matchmakers. We discuss our results with a particular focus on the influence of switching measures and definitions of relevance. Finally, in Section 6, we draw conclusions and outline aspects of future work.

2 Related Work

Experimental evaluation of SWS retrieval has received very little attention so far. The few approaches that were thoroughly evaluated so far exclusively relied on binary relevance and standard measures based on precision and recall. This was also the case with the first edition of the S3 Contest on Semantic Service Selection[2].

The first approach, and the only that we are aware of, to apply graded relevance in SWS retrieval evaluation is the work by Tsetsos et al. [3]. They propose to use a relevance scale based on fuzzy linguistic variables and the application of a fuzzy generalization of recall and precision that evaluates the degree of correspondance between the rating (not ranking) of a service by an expert and a system under evaluation. In this aspect this measure is very similar to the ADM (average distance measure) measure proposed by [4]. Unlike measures that evaluate the ranking created by a retrieval system these measures evaluate the absolute score assigned to a retrieved item by the system. This can lead to counterintuitive results since such measures are obviously biased against systems that rank services correctly but generally assign relatively higher or lower scores [5]. The measures that we use in this work avoid this issue.

Di Noia et al. obtained reference rankings for service matchmaking evaluations by directly asking human assessors to rank the available services [6]. This approach avoids the imprecision related to binary relevance judgments and generally yields more stable results than inducing a reference ranking via relevance judgments. However, it also requires much more effort from the human

[2] http://www-ags.dfki.uni-sb.de/~klusch/s3/

assessors and is thus difficult to scale to large datasets. Di Noia et al. evaluate the matchmaking performance using rank correlation measures from statistics. These measures estimate the difference between two rankings but, for instance, do not differentiate whether the rankings differ in the top ranks or the bottom ranks. Yet, for most retrieval settings, the correctness of the top ranks is much more important than that of the bottom ranks. The measures proposed in this work allow to take such considerations into account.

There is a large body of related work from the area of Information Retrieval that concerns the development of measures based on graded relevance as well as investigations of their properties [5, 7–12]. We rely heavily on these achievements and our work can be viewed as an application and adaptation of this work to the SWS retrieval evaluation domain. We are not aware of any previous work on relevance schemes specifically designed for the SWS retrieval domain and discussions on how to provide reliable and consistent relevance judgments within this domain.

3 Relevance for SWS Retrieval Evaluation

The criteria most often used for experimental retrieval evaluation has been the effectiveness of a retrieval system, i.e. how good a system is in retrieving those and only those items that a user is interested in. Effectiveness evaluations are thus based on the notion of *relevance* of an item to a query [13]. Most evaluation campaigns, in particular TREC[3], have primarily been based on binary relevance, i.e. a document (in the terminology of TREC) was considered to be either relevant or irrelevant to a topic, but no further distinction was made.

The few attempts for quantitative SWS retrieval effectiveness have so far adopted this binary approach [14–17]. However, it has been argued that binary relevance is too coarse-grained to evaluate SWS matchmaking approaches [3, 18]. This view is supported by the fact that nearly all SWS matchmaking algorithms are designed to support multiple degrees of match (DOMs). In a classic paper, Paolucci et al., for instance, proposed the use of *exact*, *plug in*, *subsumes*, and *fail* [19]. This scale or variations thereof have been adopted by many approaches.

It is thus desireable to employ a graded relevance scale instead of a binary one in SWS retrieval evaluations. However, the design of such a scale is far from trivial.

To be practically useful it must have clear definitions that enable domain experts to provide reference relevance judgments as unambiguously as possible. In this aspect a scale like *very relevant*, *relevant*, *somewhat relevant*, *slightly relevant*, and *irrelevant* as used by [3] is very difficult to judge objectively. On the other hand, human assessors should judge the relevance of a service offer with respect to a service request on the level of the original services and not their semantic formalizations. After all, the appropriateness and quality of these formalizations is also part of what is being evaluated. It is therefore not appropriate to directly use the DOMs by Paolucci et al. as a relevance scale for

[3] http://trec.nist.gov/

general SWS retrieval evaluation either. The definition of these DOMs is only meaningful in the context of DL subsumption reasoning, i.e. in the context of a particular formalization approach. It can not be meaningfully applied outside of this context.

To define a relevance scale that is equally applicable to different approaches but still sufficiently well defined to allow objective judgments, some assumptions and central terms need to be clarified. To this end, we recapitulate the basic definitions from a conceptual architecture for semantic web services presented by Preist [20]. According to this architecture, a service is defined as a *provision of value* in some given domain, e.g. the booking of a particular flight ticket. Web services are technical means to access or provide such services. Service providers typically do not offer one particular service, but a *set* of coherent and logically related services, e.g. booking of flight tickets in general and not a specific flight ticket. Service descriptions will thus describe the set of services a provider is able to offer respectively a requester is interested in. Due to dynamics involved, privacy issues, and limited precision and detailedness, service descriptions will not always precisely capture the set of services that a provider is able to deliver or that a requester is interested in. Instead, they may be incorrect (not all described services can be provided or are of interest) as well as incomplete (not all services that can be provided or are of interest are covered by the description).

Keller et al. extended this model by remarking, that descriptions based on this model are not semantically unambiguous without knowing the intention of a modeler, which can be that either all or only some of the elements contained in the described service set are requested respectively can be delivered [21]. Based upon this consideration they formally define different set theoretic match relationships between service offer and request descriptions. Because of its flexibility combined with clear definitions and its grounding to a well-defined conceptual model we propose a relevance scale that builds upon the match relationships introduced by Keller et al., extended by the notions of RelationMatch and ExcessMatch that we will explain below:

Match: The offer satisfies the request completely.

PossMatch: The offer might satisfy the request completely, but due to the incompleteness of the descriptions this can not be guaranteed based upon the available information.

ParMatch: The offer satisfies the request, but only partially (it offers some of the services which are requested but not all).

PossParMatch: The offer might satisfy the request partially, but due to the incompleteness of the descriptions this cannot be guaranteed based upon the available information.

RelationMatch: The offer does not provide services as requested, but related functionality. Thus, it could be useful in coordination with other offers.

ExcessMatch: The offer is able to provide the requested services but would result in additional undesirable effects that are not requested by the client.

NoMatch: None of the above, the offer is completely irrelevant to the request.

As a first remark, please note that these relevance degrees are not totally ordered. It will depend on the particular use case at hand, whether e.g. a definite partial match is preferable or not to a possible full match. Match, PossMatch, ParMatch, PossParMatch, and NoMatch have been introduced by Keller et al. We omit a detailed discussion due to space limitations and refer the interested reader to [21]. Instead, we will focus on RelationMatch and ExcessMatch and motivate why these extensions are necessary.

A ParMatch characterizes a situation where the client requests multiple services and a provider is capable of delivering only some of those. A similar situation arises, when, for instance, a web service is able to deliver the desired effects, but the client is unable to provide the required inputs. Consider for instance a web service able to provide flight bookings between airports identified by the international airport code and a client that requests a flight between two particular cities. The web service can not be used directly to fulfill the client's request but intuitively it would still constitute a partial match. Such situations may arise in the context of all of the four typical elements of services: inputs, outputs, preconditions and effects. To distinguish such advertisements from completely irrelevant ones, but also from the clear defined ParMatch, we added the notion of RelationMatch.

We continue with a discussion of ExcessMatch. Typically, a full match between a service advertisement and request is defined as meeting the following conditions [2]: All inputs required by the offer are available, the preconditions of the advertisement are satisfied by the state of the world prior to the service execution, and the offer provides all outputs and effects required by the client. The first two conditions concern the applicability of a service in a given situation, the last concerns its usefulness with respect to the client's request. Most approaches disregard a problem that arises, if a web service delivers more effects than are requested by the client. A client wanting to purchase a cell phone (only requested effect) would likely reject an advertisement that sells a cell phone (Effect 1) bundled with a contract with a specific telecommunication company (Effect 2). Nevertheless, most SWS matchmaking approaches would consider this a perfect match since all requested effects are delivered by the provider at hand. Similarly, a client looking for apartments in Berlin may or may not accept a web service providing a listing of apartment offers if that listing can not be restricted to offers located in Berlin. To accommodate such situations, we added the notion of ExcessMatch.

Finally, we would like to point out that, strictly spoken, the differentiations between Match and PossMatch (level of guarantee in the presence of imprecise descriptions), ParMatch and Match (level of horizontal completeness), RelationMatch and Match (issue of partial incompatibilities), and ExcessMatch and Match (issue of unwanted additional effects) are actually four unrelated dimensions that would result in 16 (2^4) levels of relevance even if each dimension is considered to be binary. To keep relevance levels manageable by the domain experts providing reference judgments, we restrict the scale to the seven relevance levels listed above for the time being. These seem to be the most important, but a further investigation of the optimal number of relevance levels is necessary and will be done in future work.

4 Evaluation Measures Based on Graded Relevance

To leverage the extra information contained in graded relevance judgments and graded degrees of match in a retrieval effectiveness evaluation, the retrieval measures for binary relevance need to be generalized to graded relevance. In this section, we present such generalized measures. To make the paper self-contained, we start by briefly recalling some basic definitions for the binary case.

Throughout this paper, we use the following definitions. Let R be the set of relevant items for a query. Let L be the set of items returned in response to that query. Then *Recall* is defined as the proportion of relevant items returned ($Recall = \frac{L \bigcap R}{R}$) and *Precision* as the proportion of returned items that are relevant ($Precision = \frac{L \bigcap R}{L}$).

Recall and Precision are set-based measures. However, there is an obvious trade-off between them. By returning more items, a system can usually increase its Recall at the expense of its Precision. Thus, in the following we assume that systems return a ranked output ordered by estimated confidence in relevance. Let $r, 1 <= r <= |L|$ denote a specific rank in this output. Let $isrel(r) = 1$, if the item at rank r is relevant and 0 otherwise. Let $count(r)$ be the number of relevant items among the top r retrieved items, i.e. $count(r) = \sum_{i=1}^{r} isrel(i)$.

This allows to measure Precision as a function of Recall by scanning L from the top to the bottom and measure the Precision at standard Recall levels. These measures average well for different queries and the corresponding R/P charts are the most widely used measure to compare the retrieval performance of systems. It is also possible to measure Precision and Recall at predefined ranks ($Precision_r$ and $Recall_r$, r is often referred to as document cutoff level). However, these measures do not average well for queries where $|R|$ varies greatly.

If a system's performance needs to be captured in a single measure, the probably most often used one is *Average Precision* over relevant items which is defined as: $AveP = \frac{1}{|R|} \sum_{r=1}^{|L|} isrel(r) \frac{count(r)}{r}$.

Since about 2000, there is an increased interest in measures based on graded or continous relevance. Various proposals have been made to generalize the measures introduced above from binary to graded relevance (see [12] for a discussion). Most of these are based on or can be expressed in terms of *Cumulated Gain* proposed by Järvelin and Kekäläinen [8]. Intuitively, Cumulated Gain at rank r measures the gain that a user receives by scanning the top r items in a ranked output list. More formally let $g(r) >= 0$ denote the gain value (or the relevance level) of the item at rank r and from now on $isrel(r) = 1$, if $g(r) > 0$ and 0 otherwise. Then Cumulated Gain at rank r is defined as $cg(r) = \sum_{i=0}^{r} g(r)$. Moreover consider an ideal ranking, i.e. $\forall(r > 1, r <= |R|) : isrel(r) = 1$ and $\forall(r > 1) : g(r) <= g(r-1)$. Let $icg(r)$ (*Ideal Cumulated Gain* at rank r) denote the Cumulated Gain for this ideal ranking.

Since $cg(r)$ can take arbitrarily large values for queries with many relevant items it has to be normalized to average or compare results across queries. *Normalized Cumulated Gain*[4] at rank r is defined as the retrieval performance relative to the optimal retrieval behavior, i.e. $ncg(r) = \frac{cg(r)}{icg(r)}$.

It allows a straightforward extension of AveP which has sometimes been referred to as Average Weighted Precision [5]: $AWP = \frac{1}{|R|}\sum_{r=1}^{|L|} isrel(r)\frac{cg(r)}{icg(r)}$.

Unfortunately, $ncg(r)$ has a significant flaw that AWP inherits: since $icg(r)$ has a fixed upper bound ($icg(r) <= icg(|R|)$), $ncg(r)$ and AWP cannot penalize late retrieval of relevant documents properly since $ncg(r)$ cannot distinguish at which rank relevant documents are retrieved for ranks greater than R [11]. This can be illustrated by comparing $ncg(r)$ and $Precision_r$ for the last rank in a full output ($R \subseteq L$). In this case $ncg(|L|) = 1$ but $Precision(|L|) = \frac{|R|}{|L|}$, which is usually much smaller than one. Several measures have been proposed that resolve this flaw of AWP.

Järvelin and Kekäläinen [8] suggested to use a discount factor to penalize late retrieval and thus reward systems that retrieve highly relevant items early. They defined *Discounted Cumulated Gain* at rank r as $dcg(r) = \sum_{i=0}^{r} \frac{g(r)}{disc(r)}$ with $disc(r) >= 1$ being an appropriate discount function. Järvelin and Kekäläinen suggest to use the log function and use its base b to customize the discount which leads to $disc(r) = \log_b r$ for $r > b$ and $disc(r) = 1$ otherwise (the distinction is necessary to maintain $disc(r) >= 1$ to avoid boosting the first ranks).

We use an according definition of *Ideal Discounted Cumulated Gain (idcg(r))* to define an adapted Version of AWP that we call *Average Weighted Discounted Precision*:

$$AWDP = \frac{1}{|R|}\sum_{r=1}^{|L|} isrel(r)\frac{dcg(r)}{idcg(r)}.$$

Similarly, Kishida [12] proposed a generalization of AveP that also avoids the flaw of AWP:

$$genAveP = \frac{\sum_{r=1}^{|L|} isrel(r)\frac{cg(r)}{r}}{\sum_{r=1}^{|R|} \frac{icg(r)}{r}}$$

Furthermore, Sakai [5] proposed an integration of AWP and AveP called Q-measure which inherits properties of both measures and possesses a parameter β to control whether Q-measure behaves more like AWP or more like AveP:

$$Q\text{-}measure = \frac{1}{|R|}\sum_{r=1}^{|L|} isrel(r)\frac{\beta cg(r) + count(r)}{\beta icg(r) + r}$$

All measures allow to finetune the extent to which highly relevant items are preferred over less relevant items (by setting appropriate gain values) but differ in the degree of control that is possible with respect to the extent to which

[4] A similar measure has been proposed by Pollack in 1968 under the name *sliding ratio*.

late retrieval is penalized. Q-Measure controls the penalty by its β Parameter, $AWDP$ by the choice of an appropriate discounting function, and genAveP lacks such control. Sakai [9] discusses this issue in detail but unfortunately disregards choices of disounting functions for $ndcg(r)$ besides the logarithm.

5 Experimental Retrieval Evaluation

We now report on the evaluation of our appraoch by means of a preliminary experiment on using the relevance scale introduced in Section 3 and the measures introduced in the previous section to evaluate the retrieval effectiveness of two matchmakers. We start by describing the test data we used and in particular our experiences on obtaining graded relevance judgments. We continue by describing the parameters that we chose for the experiment and complete our report with a discussion of our results.

5.1 Test Data

Unfortunately, there is still a lack of *standard* test collections in the area of SWS [18]. To test the proposed evaluation approach, we chose the Education subset of the OWLS-TC 2.2 test collection[5]. This subset contains 276 OWL-S service descriptions and six request descriptions together with binary relevance judgments. We chose this subset mainly for two reasons. First, this subset[6] had been used previously in an experiment with graded relevance judgments which allows to compare our results with the results from that previous experiment [3]. Second, for OWLS-TC, ranked outputs from two different matchmakers, OWLS-M3 [14] and iMatcher [16], are available through the organizers of the S3 Matchmaker Contest[7]. However, it turned out that iMatcher was unable to process one of the six queries which was thus excluded from the test data. Further information including all test data and results are available online[8].

To collect and manage graded relevance judgments for this subset, we used the OPOSSum portal[9] which already lists all the OWLS-TC services. Therefore, throughout this paper we identify queries by their id from that portal (5654, 5659, 5664, 5668, and 5675). We extended OPOSSum with a user interface that allows to conveniently enter graded relevance judgments for large numbers of services. We developed some guidelines for relevance judges[10] and three persons (one expert in the area of SWS as well as two volunteers that had only a basic understanding of SWS) judged the complete subset.

Unfortunately, it turned out that the judgments of the three judges did not correspond with each other very well. We believe that this is largely caused by

[5] http://projects.semwebcentral.org/projects/owls-tc/
[6] More precisely a similar subset from a smaller previous release of this test collection.
[7] http://www-ags.dfki.uni-sb.de/~klusch/s3/
[8] http://fusion.cs.uni-jena.de/OPOSSum/ISWC08-SMRR/
[9] http://fusion.cs.uni-jena.de/OPOSSum
[10] http://fusion.cs.uni-jena.de/OPOSSum/index.php?action=relevanceguideline

	Match	Poss	Par	PossPar	Relation	Excess	None
Relevant	130	12	33	5	6	-	20
Irrelevant	8	3	7	1	37	-	1408
Average	0.94	0.8	0.83	0.83	0.14	-	0.01

Table 1. Correspondance with binary OWLS-TC 2.2 judgments

	Match	Poss	Par	PossPar	Relation	Excess	None
Very r.	24	1	4	0	0	-	0
Relevant	19	1	2	0	0	-	2
Slightly r.	11	7	1	0	1	-	1
Somewhat r.	10	2	3	2	2	-	4
Irrelevant	3	0	0	1	0	-	15
Average	2.75	1.64	2.9	1.33	1.5	-	0.68

Table 2. Correspondance with fuzzy judgments by Tsetsos et al.

insufficient textual documentation of the services in the employed test collection. This lack of detail required relevance judges to make a lot of assumptions regarding the semantics of the services. Consequently, single judges were able to judge consistently but judgments varied between the judges depending on the different assumptions that were made (for instance whether a lecturer or a research assistant are considered researchers or not). For the rest of this paper and the reported preliminary experiment we used the judgments of the SWS expert exclusively.

We compared these judgments with the binary OWLS-TC judgments. Table 1 shows that correspondance. For each graded relevance level it shows how many of the services judged into this level were evaluated relevant versus irrelevant by the OWLS-TC authors. The average row shows the arithmetic mean that is computed by assigning a value of one/zero to the binary relevant/irrelevant services. Please note that none of the services in the Education subset of OWLS-TC was judged an ExcessMatch by our judges. Nevertheless we believe that this relevance level has its own right of existence for other collections.

Since OWLS-TC employs a very liberal definition of relevance, we were surprised to see eight services judged irrelevant by OWLS-TC but judged a perfect Match by our judges. A closer look revealed that seven of those eight mismatches seem to indicate errors in the OWLS-TC reference judgments. The remaining mismatch is caused by different context knowledge assumptions. Such assumptions also explain most of the other mismatches, like the twenty services judged irrelevant by us but relevant by OWLS-TC. Most of these, for instance, are related to a query for scholarships. Services providing information about loans were judged relevant by OWLS-TC but irrelevant by our expert.

Finally, we compared our judgments with the fuzzy relevance judgments made by Tsetsos and colleagues [3] for the OWLS-TC 2.1 Education subset, which contains the same requests as the 2.2 subset but only 135 instead of 276 services. Tsetsos et al. used a fuzzy scale with the values irrelevant, slightly relevant, somewhat relevant, relevant, and very relevant. For each graded relevance level Table 2 shows how many of the services judged into this level by our judges were judged into each of their fuzzy levels by Tsetsos et al. The average values are computed by assigning values of zero through four to the relevance levels used by Tsetsos et al. The small numbers in the Irrelevant row are caused by the fact that we used only explicit judgments, but Tsetsos et al. provided most "irrelevant" judgments only implicitly. Thus, with a full set of explicit judgments, numbers in the Irrelevant row would have been much higher and the Averages in particular in the last column much lower. We were surprised that the services judged as a perfect Match by our judges were relatively evenly distributed among the four top relevance levels of Tsetsos et al. (see first column). Since we could not obtain information about the rationale behind those judgments or the precise definitions of the relevance levels we lack an explanation for this phenomena but we expect it to be caused by the same issues that caused our judges to judge differently relatively often, too.

5.2 Evaluation Parameters

The measures described in Section 4 allow to evaluate SWS retrieval systems based on the graded relevance scheme introduced in Section 3 but leave open the question about the proper parameter combinations to use in an evaluation. As Järvelin and Kekäläinen remark, "the mathematics work for whatever parameter combinations and cannot advise us on which to choose. Such advise must come from the evaluation context in the form of realistic evaluation scenarios" [8]. In order to perform an investigation in particular of the effects of switching from binary to graded relevance, we chose two gain value settings that actually correspond to binary relevance and two settings which leverage the potential of graded relevance. The corresponding gain values are displayed in Table 3. *Strict Binary* and *Relaxed Binary* correspond to strict versus relaxed definitions of binary relevance. *Graded 1* corresponds to a setting with a focus on high precision which is appropriate in a use case of automated dynamic binding whereas *Graded 2* reflects a more balanced preference between precision and recall which seems more appropriate in use cases where a human programmer is searching for a service. Additionally (not shown in Table 3) we used the binary relevance judgments that come together with OWLS-TC 2.2 for comparison.

For each of the five queries and each of the five gain value settings, we computed the following measures for both matchmakers: AWDP using the discount functions r (AWDP-R), $\sqrt{r}$ (AWDPSQRT), $\log_2 r$ (AWDPLog2) and $\log_{10} r$ (AWDPLog10) as well as without discount function (AWP), Q-Measure with $\beta = 5$, $\beta = 1$, $\beta = 0.5$, and $\beta = 0$, and genAveP. Using a quickly growing discount function in conjunction with AWDP (e.g. AWDP-R) rewards systems that retrieve highly relevant items early, i.e. it puts the emphasis of the evaluation on the top ranks. Using no discount function (AWP) leads to a more balanced consideration of all ranks at the prize of loosing the ability to penalize a very late retrieval of items. Slowly growing discount functions (e.g. AWDPLog2) constitute a compromise between these extremes. In the case of Q-Measure a larger

	Strict Binary	Relaxed Binary	Graded 1	Graded 2
Match	1	1	6	4
PossMatch	0	1	2	2
ParMatch	0	1	1	2
PossParMatch	0	1	0.5	1
RelationMatch	0	1	0	2
ExcessMatch	0	1	0	1
NoMatch	0	0	0	0

Table 3. Experimental gain value settings

β makes Q-Measure more similar to AWP, i.e. rewards retrieving highly relevant items prior to marginally relevant items but makes it vulnerable to not penalizing very late retrieval of relevant items. Small choices for β make Q-Measure more similar to the traditional binary AveP which does not differentiate between highly and marginally relevant items but correctly penalizes late retrieval of relevant items. A choice of $\beta = 0$ completely reduces Q-Measure to binary AveP. Similarly, genAveP is reduced to AveP in settings with binary relevance.

5.3 Results

As expected, results vary significantly over queries. For Query 5675, for instance, M3 is rated higher by 40 out of the 50 possible combinations of measures and gain value settings. In contrast, for Query 5675 iMatcher is rated higher by all measures. Given this large variation, the relatively small size of our data set and in particular the fact that we had data only from two matchmakers, the results that we report in the remainder of this section need to be taken with a grain of salt. Nevertheless, they indicate some interesting preliminary findings.

Our results confirm the expectation, that the choice of measure matters, not only in terms of absolute numbers but also in terms of which matchmaker is rated higher. This is illustrated by Figure 1 that shows the values of the various measures for Request 5654 and Strict Binary and Graded 1 gain value settings. For this request, AWDP with large discounting favors iMatcher while AWDP with little or no discounting as well as Q-measure favor M3.[11]

While results frequently changed with different measures, we found that, except for $\beta = 0$, the choice of β has little influence on the absolute and relative performance of the matchmakers (see Figure 1). In fact, with our data, different parameterizations for Q-measure almost never made a difference in terms of which matcher is rated higher. Furthermore, genAveP always ranked the two matchmakers the same way Q-measure did.

For the binary cases, this behavior of Q-measure can be well explained. In this case $cg(r) = count(r)$ and $icg(r) = r$ if $r <= |R|$. Thus, Q-measures fraction can

[11] Please note that this finding (Q-measure favors M3) are specific to this request. We found frequent changes of the favored matchmaker when changing the measure but no general trend that a particular measure favors a particular matchmaker.

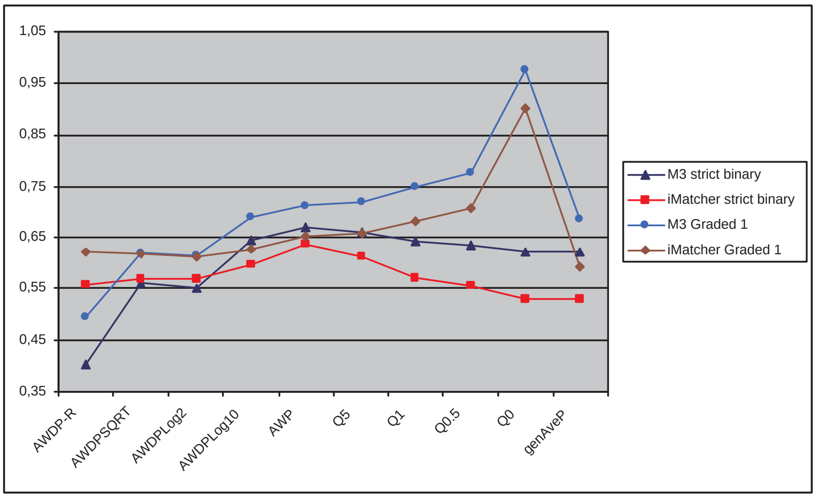


Fig. 1. Results for Request 5654 with Strict Binary and Graded 1 gain values.

be reduced by $\beta+1$ if $r <= |R|$. In other words, in binary cases, β influences the value of Q-measure only for relevant items retrieved after rank $|R|$. Relatively few relevant items are retrieved at such ranks in our experiment, thus the influence of β to the value of Q-measure is limited with our data.

Compared to the influence of Q-measure's β, the choice of discount function of AWDP caused more changes in the ratings. It didn't cause changes in the ratings for Queries 5664, 5668, and 5675 but for the remaining two queries the different versions of AWDP disagreed in eight of ten cases (two queries times five gain value settings), including those displayed in Figure 1.

The notable peak of both matchmaker's performance for the Graded 1 gain value setting measured with Q0 that is visible in Figure 1 highlights how the use of graded relevance influences measure results. Using $\beta = 0$ reduces Q-measure to AveP and thus Graded 1 to a binary scale which largely resembles the original OWLS-TC judgments. For both matchmakers, this results in a significantly increased absolute performance, albeit not in a change of their performance relative to each other.

Generally, changes in the settings of the gain values caused more significant changes in how the matchmakers were rated than changes in the parameterizations of AWDP and Q-measure. However, the Q-measure variations and genAveP were again less sensitive towards changes in the evaluation parameters than the AWDP-family. Their ratings did not change regardless of the gain values used except for Query 5664 where they all preferred M3 with the Strict Binary and iMatcher with the other settings[12]. In contrast, with the one exception of Query

[12] Except for Q-measure with $\beta = 5$ and Graded 2. This case favored M3, too.

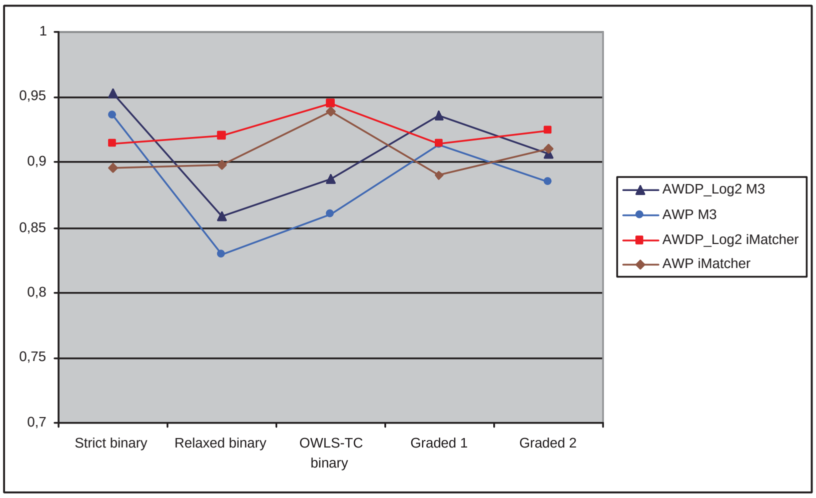


Fig. 2. AWDP measures for Request 5664 for different gain values.

5668 where iMatcher outperforms M3 regardless of the measure, changes in the gain value settings caused changes in the ratings of the AWDP measure family in nearly half of the cases. As an example, Figure 2 shows the values of AWDPLog2 and AWP for Request 5664: M3 is favored by both measures for Strict Binary and Graded 1 while iMeasure is favored for Relaxed Binary, OWLS-TC Binary, and Graded 2. Generally, Relaxed Binary and OWLS-TC Binary tend to benefit iMatcher while the other settings tend to benefit M3. The most likely interpretation is that M3 performs a stricter selection and thus outperforms iMatcher in ranking more relevant services higher. Another influence factor may be that iMatcher applies machine learning techniques and has been trained with the binary relevance judgments of OWLS-TC. Switching to other definitions of relevance (e.g. strict binary relevance) seems to have a negative impact on iMatcher's performance relative to that of M3.

6 Conclusions and Future Work

In this paper, we investigated how to evaluate the retrieval effectiveness of SWS matchmakers based on graded instead of binary relevance. We discussed the notion of relevance in this particular context and proposed a well-founded scale of relevance levels customized to the SWS matchmaking domain. We presented a number of evaluation measures for graded relevance and described an experiment of using those measures to perform a retrieval evaluation of two SWS matchmakers.

We need to note once more, that our results have to be considered preliminary because of the nature of the test data used. First, there was a significant variation in how the different judges judged our test data. We believe this to be caused by the insufficient documentation of the services in our data and expect this issue to improve if more realistic and better documented services are used than are currently available in the form of OWLS-TC. Second, we have compared only two matchmakers based on a relatively small data set. In terms of investigating the effects of different measures and different relevance scales it would be particularly desireable to have access to a larger number of directly comparable matchmakers. This will be the case if either more readily implemented matchmakers for a particular formalism (for instance SAWSDL) become available or test collections across formalisms will be developed.

Despite of these two restrictions, our results allow to draw a number of interesting conclusions. First, retrieval evaluation based on graded relevance is feasible both in terms of the effort to obtain graded instead of binary relevance judgments and in terms of the availability of measures suitable for graded relevance. Second, the choice of gain values (i.e. relevance levels) and the choice of measure has a significant influence on the evaluation results. Our results indicate that the choice of gain values has a greater impact than the choice of measure. Third, AWDP seems to be more sensitive towards changes in the parameterizations (regarding both, the penalty for late retrieval and changes of the gain values) than Q-measure and thus should probably be the first choice for future evaluations.

In our future work we plan to verify these findings with better data. As a first step, we would like to investigate whether relevance judgments really become more consistent across judges when more realistic and well documented services are used. A second step will then be to compare a larger number of matchmakers based on that more realistic data.

Acknowledgments

We would like to thank Patrick Kapahnke and Matthias Klusch for providing us with test data from the S3 Contest and for their help in resolving some technical problems with that data. We would also like to thank them and the other contributors for developing OWLS-TC and making it publicly available. Finally, we would like to thank Vassileios Tsetsos for giving us the graded relevance judgments from [3] to compare them with ours.

References

1. McIlraith, S.A., Son, T.C., Zeng, H.: Semantic web services. IEEE Intelligent Systems **16**(2) (2001) 46–53
2. Klusch, M.: Semantic web service coordination. In M. Schumacher, H.H., ed.: CASCOM - Intelligent Service Coordination in the Semantic Web. Springer (2008)
3. Tsetsos, V., Anagnostopoulos, C., Hadjiefthymiades, S.: On the evaluation of semantic web service matchmaking systems. In: 4th IEEE European Conference on Web Services (ECOWS2006), Zürich, Switzerland (2006)

4. Mea, V.D., Demartini, G., Gaspero, L.D., Mizzaro, S.: Measuring retrieval effectiveness with average distance measure (ADM). Information Wissenschaft und Praxis **57**(8) (2006) 405–416
5. Sakai, T.: New performance metrics based on multigrade relevance: Their application to question answering. In: Fourth NTCIR Workshop on Research in Information Access Technologies, Information Retrieval, Question Answering and Summarization (NTCIR04), Tokyo, Japan (2004)
6. Noia, T.D., Sciascio, E.D., Donini, F.M.: Semantic matchmaking as non-monotonic reasoning: A description logic approach. Journal of Artificial Intelligence Research (JAIR) **29** (2007) 269–307
7. Kekäläinen, J., Järvelin, K.: Using graded relevance assessments in IR evaluation. Journal of the American Society for Information Science and Technology **53**(13) (2002) 1120–1129
8. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of ir techniques. ACM Transactions on Information Systems **20**(4) (2002) 422–446
9. Sakai, T.: On penalising late arrival of relevant documents in information retrieval evaluation with graded relevance. In: First International Workshop on Evaluating Information Access (EVIA), Tokyo, Japan (2007)
10. Sakai, T.: On the reliability of information retrieval metrics based on graded relevance. Information Processing and Management **43**(2) (2007) 531–548
11. Sakai, T.: Ranking the NTCIR systems based on multigrade relevance. In: Revised Selected Papers of the Asia Information Retrieval Symposium, Beijing, China (2004) 251–262
12. Kishida, K.: Property of average precision and its generalization: An examination of evaluation indicator for information retrieval experiments. Technical Report NII-2005-014E, National Institute of Informatics, Tokyo, Japan (2005)
13. Mizzaro, S.: Relevance: The whole history. JASIS **48**(9) (1997) 810–832
14. Klusch, M., Fries, B., Sycara, K.: Automated semantic web service discovery with OWLS-MX. In: 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate, Japan (2006)
15. Kaufer, F., Klusch, M.: Performance of hybrid WSML service matching with WSMO-MX: Preliminary results. In: First International Joint Workshop SMR^2 on Service Matchmaking and Resource Retrieval in the Semantic Web at the 6th International Semantic Web Conference (ISWC2007), Busan, South Korea (2007)
16. Kiefer, C., Bernstein, A.: The creation and evaluation of iSPARQL strategies for matchmaking. In: 5th European Semantic Web Conference (ESWC2008), Tenerife, Canary Islands, Spain (2008) 463–477
17. Dudev, M., Kapahnke, P., Klusch, M., Misutka, J.: International semantic service selection contest s3 - contest participation guideline. online at http://www-ags.dfki.uni-sb.de/∼klusch/s3/s3-contest-plugin.zip (2007)
18. Küster, U., König-Ries, B.: On the empirical evaluation of semantic web service approaches: Towards common SWS test collections. In: 2nd IEEE International Conference on Semantic Computing (ICSC2008), Santa Clara, CA, USA (2008)
19. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.P.: Semantic matching of web services capabilities. In: First International Semantic Web Conference (ISWC2002), Sardinia, Italy (2002) 333–347
20. Preist, C.: A conceptual architecture for semantic web services (extended version). Technical Report HPL-2004-215, HP Laboratories Bristol (2004)
21. Keller, U., Lara, R., Lausen, H., Polleres, A., Fensel, D.: Automatic location of services. In: Second European Semantic Web Conference (ESWC2005), Heraklion, Crete, Greece (2005)

Model-Driven Semantic Service Matchmaking for Collaborative Business Processes

Matthias Klusch, Stefan Nesbigall, and Ingo Zinnikus

German Research Center for Artificial Intelligence (DFKI),
Stuhlsatzenhausweg 3, Saarbrücken
(klusch|stefan.nesbigall|ingo.zinnikus)@dfki.de

Abstract. Business process modelling and execution in a collaborative environment requires a set of methodologies and tools which support the transition from an analysis to an execution level. Integrating the process with a pre-existing IT infrastructure leads to typical interoperability problems. Service-oriented architectures are today's favorite answer to solve these interoperability issues. To tackle them, the recent trend is to use the principles of model driven-design. In this paper, we apply these principles to Semantic Web service technology to assist a business orchestrator finding suitable services at design time, and composing workflows for agent-based execution. We describe a formal approach to preserve the content of the semantic annotations in the model and code transformations.

1 Introduction

Service-oriented architectures (SOA) are today's favorite answer to realize the vision of seamless business interaction across organizational boundaries. It enables enterprises to offer selected functionalities of their business systems via standardized XML-based Web service interfaces (written in WSDL [5]). Complex business application processes can be implemented through appropriate Web service compositions in prior or on-demand each of which functionality is made available to the customer at the respective enterprise portal in the Web. The SOA principle provides a loosely coupled and standardized modular solution to enterprise business application landscapes. One recent trend of developing SOAs is to apply the principles of model-driven software development (MDD) by (i) modelling the overall business process workflows in a more abstract manner, and (ii) providing model transformations that define mappings between the abstract specification and the underlying platform-specific systems. According to [14], business process modelling and execution is commonly performed in a top-down fashion. Since existing standard Web services lack formal semantics, from the point of view of strong AI, the meaningful integration of services realizing the desired business processes exclusively relies on human business domain experts at design time. In contrast, Semantic Web service technology adds expressivity to existing Web service standards by introducing well-formed semantics that simple Web service descriptions are lacking, and envisages intelligent agents to discover and compose complex business services through logic-based reasoning upon their semantic annotations. However, in many real-world cases of business process modelling among contracted and trusted business partners, the fully automated coordination of partly unknown business Web services is neither adequate nor efficient in practice. When service composition is concerned the Semantic Web service approach can be compared to planning from first principles in AI while the model-driven approach can be compared to planning from second principles if the platform-specific engine for executing the models is powerful enough. In this sense, both approaches model-driven process development (MDD) and Semantic Web services (SWS) have their pros and cons when used to integrate external, outsourced business services in SOAs. In the spirit of the model-driven approach, we introduce a metamodel for Semantic Web services, called PIM4SWS, which is an abstraction from most commonly used SWS description languages or so-called platform-specific models, that are OWL-S [18], WSML [26] and standard SAWSDL [22]. That renders semantic service selection and composition for implementing business process workflows in SOAs independent of these models. In particular, we envisage a model-driven Semantic Web service matchmaker, called MDSM (Model-Driven Service Matchmaker), to support human business domain experts and service orchestrators in finding suitable services for this purpose at design time. As a consequence, these experts only need knowledge about the common UML-based metamodel PIM4SWS but not the specific models like OWL-S, WSML or SAWSDL used by different business service developers to describe the semantics of their individual services that are potentiall relevant for implementing the collaborative business process workflow. Syntactic mapping from a metamodel in UML to parts of these specific models are proposed in [16, 11, 1] but without any formal grounding of their transformations. [21] provides a comparison between concepts in OWL-S and WSML. In contrast, we propose to use the formal specification language Z [23], respectively, Object-Z [8] as a common language for provably correct transformations between different SWS models.

The remainder of this paper is structured as follows. We outline the MDSM matchmaking process in section 2. In section 3 we describe the transformation of the service request from the platform-independent to the platform-specific level. Section 4 gives an example of the whole matchmaking process of MDSM, while section 5 concludes the paper.

2 MDSM Overview

The MDSM matchmaker is capable of automated, model-independent semantic service selection to assist business service orchestrators in finding suitable services to realize parts of collaborative business processes as adequate service orchestrations at design time. Consider, for example, the modelling of a complex travel planning process as depicted in figure 1. At a certain point of choice in the planning process the human user, that is the business orchestrator, needs to select a flight booking Web service to realize the respective booking process in the overall workflow of travel planning. For this purpose, the orchestrator models her Web service request in the common metamodel she is familiar with only, that is the platform-independent metamodel PIM4SWS.

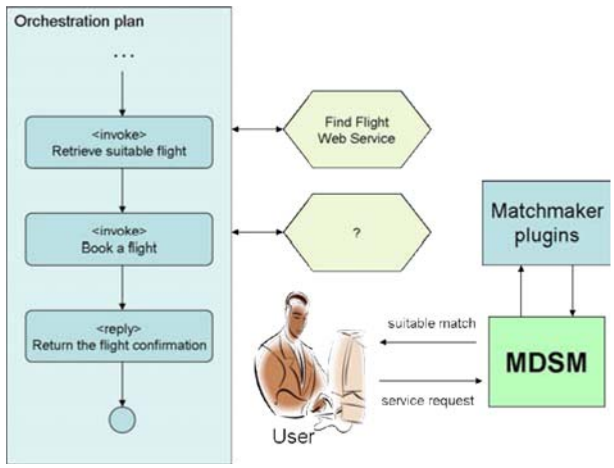


Fig. 1. Orchestration plan

The MDSM, in its first implementation, automatically transforms this request to semantically equivalent service requests in OWL-S, WSML and SAWSDL, and then issues them to relevant platform-specific matchmakers, that are for the implementation of MDSM 1.0, the matchmakers OWLS-MX [13], WSML-MX [12] and SAWSDL-MX. Eventually the MDSM provides the orchestrator with an aggregated and ranked answer set of relevant semantic services [3] together with their grounding in WSDL for invocation (cf. Fig. 2). The retransformation of platform-specific services to the common metamodel PIM4SWS by the MDSM is optional. One crucial step of this model-driven semantic service selection by the MDSM are the semantically equivalent model transformations which we discuss in the following section.

3 Transformation

In order to correctly compile a given service request in PIM4SWS to different platform-specific representations such as OWL-S and WSML, we differentiate between (a) structural transformation of the semantic service description as a whole, and (b) the semantic mapping between corresponding components of the information model of PIM4SWS such as its input, output, preconditions and effects that are described in specific ontology and rule languages like OWL [19], SWRL and WSML [27]. While structural transformations of PIM4SWS

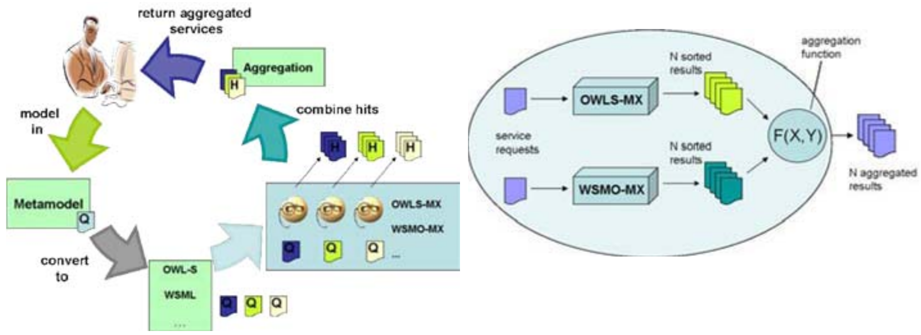


Fig. 2. Overview: Model-driven Semantic Web service matchmaking by the MDSM. (left); Aggregation of the answer sets of platform-specific service matchmakers. (right)

representations in UML to OWL-S and WSML are defined in terms of syntactic mappings between corresponding modelling concepts, we use the standardized formal specification language Z for the latter purpose (cf. Fig. 3).

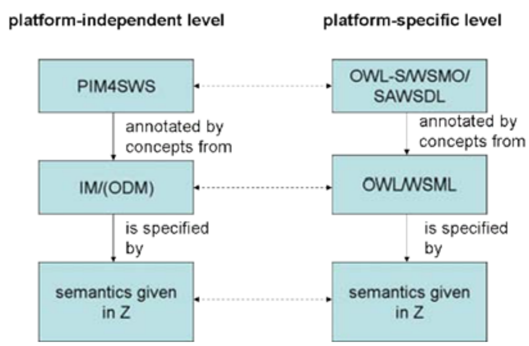


Fig. 3. Overview: Comparison of semantic services across platforms.

3.1 Structural Transformation

The metamodel PIM4SWS is designed as a core model to cover the common parts of the underlying semantic service description languages. It consists of three parts: *InformationModel* (IM; ODMNameSpace), *BlackBox* and *GlasBox* (cf. Fig. 4. The information model is the set service related ontologies described in the standard metamodel ODM [4, 9]) (also called ODMNameSpace), while both its functionality (*Functionals*) in terms of service signature, that are input and output parameters, and specification, that are preconditions and effects, and non-functional parameters (*NonFunctionals*) such as price, service name and developer are described in the service profile or *BlackBox*. The *GlasBox* includes the description of the internal service process and is not considered in this paper.

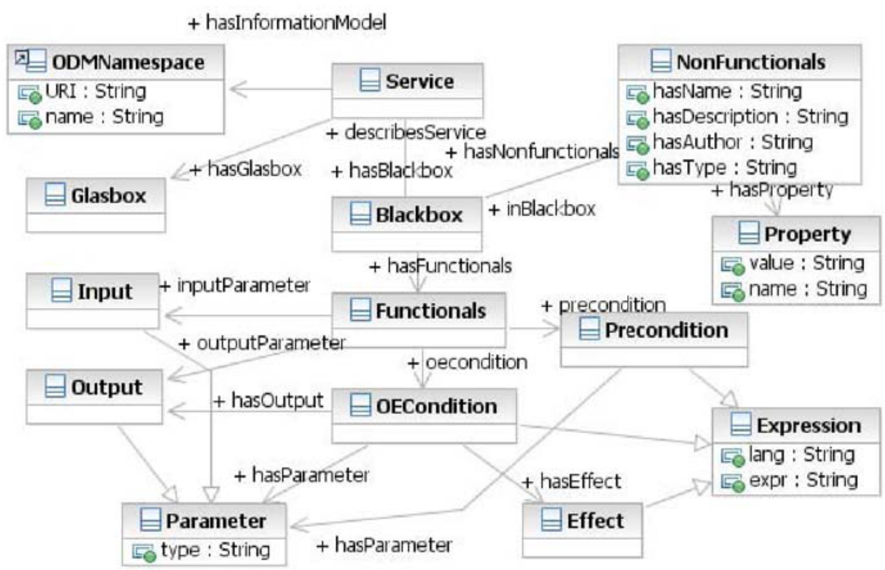


Fig. 4. Platform-independent metamodel for semantic Web services.

We acknowledge that the PIM4SWS metamodel is similar to the OWL-S model which can be to a large extent embedded into the WSML model[15] - which makes the structural transformations from PIM4SWS to both specific models or platforms straight-forward. In particular, the OWL-S service profile is generated from the *Functionals* and *NonFunctionals* of the given PIM4SWS service description, the OWL-S process model for atomic processes is given by the *Functionals* where the "hasResult" construct of the OWL-S service is extracted from the *OECondition*. For structural transformations from PIM4SWS to WSML, the following holds: (a) the *Service* class is related to any service component in WSML; (b) the *NonFunctionals* class is covered by annotations and non-functional properties of the considered service component; and (c) the *Functionals* class is mapped to the capability of the service. Since in PIM4SWS inputs and outputs describe information between a service provider and the requester, these classes are related to pre- and postcondition concepts in WSML. Furthermore, we map preconditions to assumptions. The WSML service result construct is resolved by an implication in the postcondition and effect axiom of WSML, where the antecedent is the semantically transformed *OECondition* expression, and the consequent is the transformed *hasEffect* expression for an effect, and the transformed *hasOutput* for a postcondition in WSML. Each parameter is handled by initializing shared WSML variables. Due to space limitations, we omit further details of the structural transformation from PIM4SWS to WSML and SAWSDL, and rather focus on the semantic mapping between the PIM4SWS information model (in ODM) and different ontology languages (platforms) used for semantic annotation.

3.2 Semantic Transformation using Z

To achieve a verifiably correct mapping between the different DL-based ontology languages used for semantic annotation such as OWL-DL and WSML-DL in our case, we use the formal standard specification language Z as a common basis[1]. Based on [4, 9], our initial version of the information model IM of the PIM4SWS is the description logic *SHIN(D)* related part of the metamodel ODM written in UML; the metamodel of the PIM4SWS-IM is given in [4]. *SHIN(D)* is the intersection of the description logics underlying OWL-DL and WSML-DL, that are *SHOIN(D)*, respectively, *SHIQ(D)*. As such the (PIM4SWS-)IM does not support enumerated classes with nominals (O) nor qualified role cardinality restrictions (Q) for semantic annotations: While WSML-DL does not support the former, the latter cannot fully be covered by OWL-DL [20]. The standard for semantic Web services, SAWSDL, does not provide any specific ontology language, hence, for SAWSDL services, we assume model references to ontologies in OWL-DL and WSML-DL. As a consequence, each platform-specific matchmaker called by the model-driven MDSM matchmaker with service requests in PIM4SWS with annotations in *SHIN(D)* (subsumed by both *SHOIN(D)* and *SHIQ(D)*) is able to match these against any service in OWL-S, WSML and SAWSDL with annotations in OWL-DL or WSML-DL.

Why then using Z? In principle, the information model of the PIM4SWS is not restricted to our initial choice of a description logic (*SHIN(D)*) but shall cover different ontology and rule languages (in different notations) with first-order logic (FOL) semantics. For this purpose, we suggest to use the ISO standard specification language Z (semantically equivalent to FOL) as a common language for specifying semantic annotations of service requests in PIM4SWS by the orchestrator. Please note that the semantic equivalence of PIM4SWS service request annotations in *SHIN(D)* we proposed for our intial version of the PIM4SWS-IM with those in OWL-DL and WSML-DL of the request in OWL-S and WSML compiled by the MDSM matchmaker is trivial: It holds per definition of the PIM4SWS-IM as intersection of OWL-DL and WSML-DL both assumed as sole ontology languages for semantic annotations of service requests in PIM4SWS, OWL-S and WSML. In general, testing the semantic equivalence of pairs of platform-independent and platform-dependent semantic service request

[1] Z is based on Zermelo-Fraenkel set theory and first-order predicate logic. It is widely used by industry for system behaviour specification and verification of properties, and has undergone international ISO standardization. Various tools for formatting, type-checking and aiding proofs in Z are available, e.g. see http://vl.zuser.org/.

annotations in different first-order logic-based ontology languages in different syntactic representation is to convert them into equivalent FOL expressions and use a FOL theorem prover for checking the satisfiability of their mutual logic implication. This semantic equivalence can also be shown using Z as common specification language as shown in figure 5.

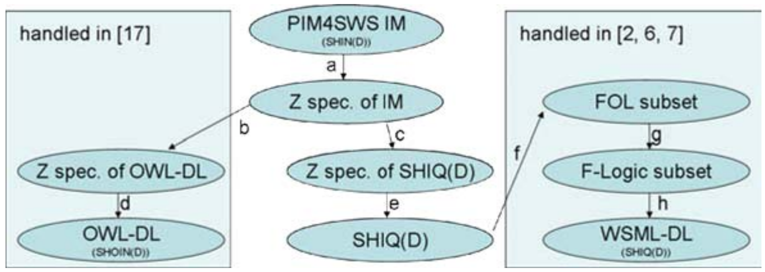


Fig. 5. Semantically equivalent compilation of annotations in PIM4SWS-IM to OWL-DL and WSML-DL using Z.

In particular, having a semantic annotation in the PIM4SWS-IM, we provide its transformation to Z (step a), which corresponds to the *SHIN(D)* related part of the specification of OWL-DL, respectively *SHOIN(D)*, in Z (steps b and d, as reported in [17] with proof of correctness and completeness). Likewise, the specification of the PIM4SWS service request annotation in Z corresponds to the *SHIN(D)* related part of the specification of WSML-DL, respectively *SHIQ(D)* in Z (steps c and d), which, in turn, is a FOL subset (step f, [2]) that can be written in F-Logic (step g) and as such syntactically transformed to an (WSML-DL equivalent) annotation of a WSML service request (step h, as reported in [7, 6]). Due to space limitations, we omit the full specification of the initial version of the PIM4SWS-IM, OWL-DL and WSML-DL in Z but a rough sketch only and show that the given transformations of IM and OWL-DL to Z are semantically equivalent according to the (FOL bases) Z semantics. The latter inherently holds since the IM is defined to be a subset of OWL-DL (and WSML-DL), but the principle of proving the semantic equivalence of a given pair of Z transformations is useful to apply also for cases where this is not the case, e.g., when it is not clear whether additional IM elements can be emulated by those of targeted platform-specific languages.

The universal signature of the (PIM4SWS-)IM is the set of OntologyElements with the interpretation $I_{IM} = (\mathit{IMIndividual}, \cdot^{I_{IM}})$ where IMIndividual is the domain of discourse Δ (according to the set-theoretic first-order semantics of description logics). The function $\cdot^{I_{IM}}$ maps an IMClass c to the subset of the domain (classInstances(c)) and an IMProperty p to a tuple (subject(propVals(p)), object(propVals(p))). The semantics of the initial IM, that is *SHIN(D)*, is then equivalently specified by the following Z axioms.

$[\mathit{OntologyElement}]$

An IMIndividual is modelled as a subset of OntologyElement. IMClass is another subset of OntologyElement disjoint from IMIndividual. The function classInstances maps an IMClass to the IMIndividual set of their instances.

$$\begin{array}{|l} \mathit{IMIndividual}, \mathit{IMClass} : \mathbb{P}\ \mathit{OntologyElement} \\ \mathit{classInstances} : \mathit{IMClass} \rightarrow \mathbb{P}\ \mathit{IMIndividual} \\ \hline \mathit{IMClass} \cap \mathit{IMIndividual} = \varnothing \end{array}$$

IMProperty and IMPropertyValue are again subsets of OntologyElement and disjoint to each other and to the above subsets. An instance of an IMProperty is an IMPropertyValue. The function propVals maps an IMProperty to its set of instances.

$$\begin{array}{|l} \mathit{IMProperty}, \mathit{IMPropertyValue} : \mathbb{P}\ \mathit{OntologyElement} \\ \mathit{propVals} : \mathit{IMProperty} \rightarrow \mathbb{P}\ \mathit{IMPropertyValue} \\ \hline \mathit{IMProperty} \cap \mathit{IMClass} = \varnothing \\ \mathit{IMProperty} \cap \mathit{IMIndividual} = \varnothing \\ \mathit{IMPropertyValue} \cap \mathit{IMClass} = \varnothing \\ \mathit{IMPropertyValue} \cap \mathit{IMIndividual} = \varnothing \\ \mathit{IMPropertyValue} \cap \mathit{IMProperty} = \varnothing \end{array}$$

Every IMPropertyValue has unary relations, subject and object, that returns two IMIndividuals which are related by the property.

$$\begin{array}{|l} \mathit{subject} : \mathit{IMPropertyValue} \leftrightarrow \mathit{IMIndividual} \\ \mathit{object} : \mathit{IMPropertyValue} \leftrightarrow \mathit{IMIndividual} \end{array}$$

To express a class hierarchy in the metamodel generalizations are used. A generalization relates two classes, where the first class is a subclass of the second.

$$\begin{array}{|l} \mathit{imSubClassOf} : \mathit{IMClass} \leftrightarrow \mathit{IMClass} \\ \hline \forall\, c_1, c_2 : \mathit{IMClass} \bullet \mathit{imSubClassOf}(c_1) = c_2 \\ \Leftrightarrow \mathit{classInstances}(c_1) \subseteq \mathit{classInstances}(c_2) \end{array}$$

Universal restricted classes are a subset of an IMClass that are universal restricted to a target IMClass by a given IMProperty.

$$\begin{array}{|l} \mathit{UniRestriction} : \mathbb{P}\ \mathit{IMClass} \\ \mathit{onProperty} : \mathit{UniRestriction} \leftrightarrow \mathit{IMProperty} \\ \mathit{toClass} : \mathit{UniRestriction} \leftrightarrow \mathit{IMClass} \\ \hline \forall\, r : \mathit{UniRestriction};\ a_1, a_2 : \mathit{IMIndividual} \bullet a_1 \in \mathit{classInstances}(r) \\ \Leftrightarrow (\exists\, v : \mathit{IMPropertyValue} \mid v \in \mathit{propVals}(\mathit{onProperty}(r)) \bullet \\ (\mathit{subject}(v) = a_1 \wedge \mathit{object}(v) = a_2) \Rightarrow a_2 \in \mathit{classInstances}(\mathit{toClass}(r))) \end{array}$$

For the Z-specification of semantic annotations in OWL-DL, we refer to [17]. In the following, variables in platform-specific specifications are marked with a prime like x', y'. The interpretation of the Z-specification of OWL-DL expressions is defined as $I_{OWL} = (OWLIndividual, \cdot^{I_{OWL}})$ with the domain OWLIndividual and the interpretation function $\cdot^{I_{OWL}}$ mapping an OWLClass c' to a subset (instances(c')) of the domain and an OWLProperty p' to a tuple (subVal(p')). The corresponding Z axioms for OWL-DL restricted to the IM specification in Z are as follows.

$$[Resource]$$

$$\begin{array}{|l} OWLIndividual, OWLClass, OWLProperty : \mathbb{P}\, Resource \\ \hline OWLIndividual \cap OWLClass = \varnothing \\ OWLProperty \cap OWLClass = \varnothing \\ OWLProperty \cap OWLIndividual = \varnothing \end{array}$$

$$\begin{array}{|l} instances : OWLClass \rightarrow \mathbb{P}\, OWLIndividual \\ subVal : OWLProperty \rightarrow (Resource \leftrightarrow Resource) \end{array}$$

$$\begin{array}{|l} subClassOf : OWLClass \leftrightarrow OWLClass \\ \hline \forall\, c_1', c_2' : OWLClass \bullet \\ subClassOf(c_1') = c_2' \Leftrightarrow \\ instances(c_1') \subseteq instances(c_2') \end{array}$$

$$\begin{array}{|l} allValuesFrom : (OWLClass \times OWLProperty) \leftrightarrow OWLClass \\ \hline \forall\, c_1' : OWLClass;\ p' : OWLProperty;\ c_2' : OWLClass \bullet \\ allValuesFrom(c_1', p') = c_2' \Leftrightarrow (\forall\, a_1', a_2' : OWLIndividual \bullet \\ a_1' \in instances(c_2') \Leftrightarrow ((a_1', a_2') \in subVal(p') \Rightarrow a_2' \in instances(c_1'))) \end{array}$$

Specifying *SHIQ(D)* of WSML-DL for its subset *SHIN(D)* of the IM in Z is the same as we showed for OWL-DL above. That allows to compare the respective elements of WSML-DL and OWL-DL by looking at their representation in Z (with differently renamed elements for better distinction between them) as shown in table 1. The semantically equivalent transformation from WSML-DL to the corresponding F-Logic fragment is given in [6, 7] which means that the Z specification of IM annotations in *SHIN(D)* can be equivalently transformed to F-Logic used to describe semantic annotations (concepts and constraints) in WSML services.

Table 1. Notation of some elements of WSML-DL and OWL-DL in Z

WSML-DL (*SHIQ(D)*) in Z	OWL-DL in Z	Remark
Δ^S	OWLIndividual	set of instances
AC	OWLClass	set of atomic concepts
AR	OWLProperty	set of atomic properties
$\cdot^{I_S}$	instances / subVal	semantic mapping to the domain
subConcept	subClassOf	concept hierarchy
forall	allValuesFrom	universal quantifier

To verify whether a direct model transformation $t(D) = D'$ of a description D in the platform-independent model PIM4SWS-IM to a description D' in platform-specific model or ontology language is semantically correct, one can test whether the semantics of D and D' are equivalent in Z. We show this by example for a description D in IM and D' in OWL-DL both transformed to Z. In Zermelo-Fraenkel set theory, the axiom of extensionality defines the equality of two sets: $\forall A \forall B[\forall x(x \in A \Leftrightarrow x \in B) \Rightarrow A = B]$. Thus, descriptions D and D' are semantically equal ($sem(D) = sem(D')$) iff their interpretations in the domain are the same ($D^I = D'^I$). For reasons of comparability, the domains of discourse of considered ontology languages (models) have to be the same: $MIndividual = OWLIndividual = \Delta$. The element equality in Z is defined as follows: Let $x \in$ IMIndividual, $y' \in$ OWLIndividual$/\Delta$, then elements x, y' are the same $eql(x, y') = true$ iff $sem(x) = sem(y')$. That allows us to compare the semantic equality of different language elements in Z as shown in table 2: Equality of facts (a), sets of instances (b), concepts (c), property values (d), sets of property values according to a given property (e), and properties (f). In fact, we can obtain the semantic equivalence between constructors of the description logics underlying the initial PIM4SWS-IM and those of OWL-DL, respectively WSML-DL denoted in Z. Due to space limitation, we provide only a selection of these Z-equality relations in the following.

Table 2. Equality of facts, concepts, and roles.

a)	**instance:** o	
	$x : IMIndividual$	$= y' : OWLIndividual/\Delta$
		$\Leftrightarrow eql(x, y')$
b)	**instances of class** C**:** C^I	
	$classInstances(x)$	$= instances(y')/y'^{I_S}$
		$\Leftrightarrow (\forall\, i \in classInstances(x)\, \exists\, i' \in instances(y')/y'^{I_S} \mid eql(i, i')) \wedge (\forall\, i' \in instances(y')/y'^{I_S}\ \exists\, i \in classInstances(x) \mid eql(i, i'))$
c)	**class** C**:** C	
	$x : IMClass$	$= y' : OWLClass/AC$
		$\Leftrightarrow classInstances(x) = instances(y')/y'^{I_S}$
d)	**role value:** $\langle o, o' \rangle$	
	$x : IMPropertyValue$	$= (a'_1, a'_2) : OWLIndividual/\Delta \times OWLIndividual/\Delta$
		$\Leftrightarrow eql(subject(x), a'_1) \wedge eql(object(x), a'_2)$
e)	**role values of** R**:** $\langle o, o' \rangle \in R^I$	
	$propVals(p)$	$= subVal(p')/p'^{I_S}$
		$\Leftrightarrow (\forall\, v \in propVals(p)\, \exists (a'_1, a'_2) \in subVal(p')/p'^{I_S} \mid v = (a'_1, a'_2)) \wedge (\forall (a'_1, a'_2) \in subVal(p')/p'^{I_S}\ \exists\, v \in propVals(p) \mid v = (a'_1, a'_2))$
f)	**role** R**:** R	
	$p : IMProperty$	$= p' : OWLProperty/AR$
		$\Leftrightarrow propVals(p) = subVal(p')/p'^{I_S}$

We use these elementary Z-equality relations recursively to prove (by structural induction) the semantic equality for any given DL axiom or expression. For example, consider the DL concept subsumption axiom $c_1 \sqsubseteq c_2$ for two concepts c_1, c_2. The equality of its description *imSubClassOf* (in PIM4SWS-IM) and *subClassOf* (in OWL-DL) can be shown by the equality of their transformation in Z. Let $c_1, c_2 \in$ IMClass, $c_1', c_2' \in$ OWLClass/AC, $c_3 \in$ UniRestriction, $p \in$ IMProperty, $p' \in$ OWLProperty with $c_1 = c_1'$, $c_2 = c_2'$, onProperty$(c_3) = p$, toClass$(c_3) = c_1$ and $p = p'$, then the following holds:

$$\begin{aligned} \mathit{imSubClassOf}(c_1) = c_2 &\Leftrightarrow \mathit{classInstances}(c_1) \subseteq \mathit{classInstances}(c_2) & (1)\\ &\Leftrightarrow \mathit{instances}(c_1') \subseteq \mathit{instances}(c_2') & (2)\\ &\Leftrightarrow \mathit{subClassOf}(c_1') = c_2' & (3) \end{aligned}$$

In line (2) we use the equality relation (b) in table 2 to translate the IM specification of imSubClassOf in Z to OWL-DL, which is the same as *subClassOf* in OWL-DL. Analogously, we provide the (Z-)equality relation for universal quantified role cardinality restrictions ($\forall R.C$ in DL syntax):

$$\begin{aligned} &c_3 \Leftrightarrow \forall\, a_1, a_2 : \mathit{IMIndividual} \bullet a_1 \in \mathit{classInstances}(c_3) \Leftrightarrow\\ &\quad (\exists\, v : \mathit{IMPropertyValue} \mid v \in \mathit{propVals}(p) \bullet\\ &\quad (\mathit{subject}(v) = a_1 \wedge \mathit{object}(v) = a_2) \Rightarrow a_2 \in \mathit{classInstances}(c_1) \end{aligned}$$

Thus, instances of c_3 are equal to instances of the allValuesFrom construct, and determined by the following expression:

$$\begin{aligned} \mathit{classInstances}(c_3) &\Leftrightarrow \{x : \mathit{IMIndividual} \mid \forall\, a_2 : \mathit{IMIndividual} \bullet\\ &\quad \exists\, v : \mathit{IMPropertyValue} \mid v \in \mathit{propVals}(p) \bullet (x = \mathit{subject}(v) \wedge\\ &\quad a_2 = \mathit{object}(v)) \Rightarrow a_2 \in \mathit{classInstances}(c_1)\}\\ &\Leftrightarrow \{x' : \mathit{OWLIndividual} \mid \forall\, a_2' : \mathit{OWLIndividual} \bullet\\ &\quad (x', a_2') \in \mathit{subVal}(p') \Rightarrow a_2' \in \mathit{instances}(c_1')\}\\ &\Leftrightarrow \mathit{instances}(\mathit{allValuesFrom}(c_1', p')) \end{aligned}$$

Based on these Z-equality relations, one can prove that the syntactic model transformation function $(t(D) = D')$ of the MDSM matchmaker from PIM4SWS-IM to OWL-DL and WSML-DL is semantically correct.

4 Example

In the following, we briefly illustrate the principle of model-driven service matchmaking by the MDSM matchmaker. Suppose that a business service orchestrator intends to integrate a flight-booking Web service into her business process implementation. The service shall book one ticket for a given flight and customer, and confirms the booking. This request is formulated in PIM4SWS by the orchestrator and passed to the MDSM which transforms the received request to specific description models, that are, in our case, OWL-S, WSML and SAWSDL.

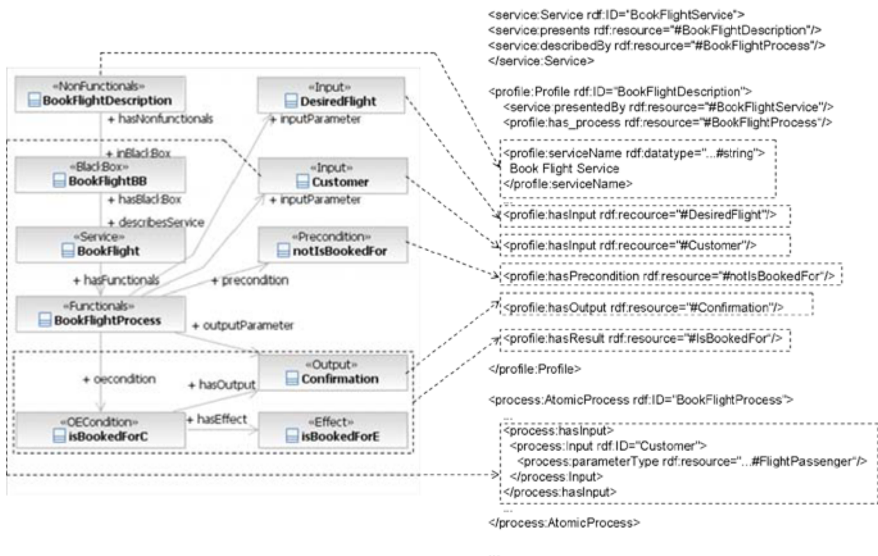


Fig. 6. Bookflight service request transformation from PIM4SWS to OWL-S.

The structural transformations to OWL-S and WSML are depicted in figures 6 and 7.

The semantic transformation of the request concerns all ontological concepts used to describe the service profile parameters (IOPE). We show this by example for the input concept Flight-Passenger (FP) which is described in the PIM4SWS-IM as shown in figure 8. In the following, we use the abbreviations P for Passenger, FP for FlightPassenger, AP for AirPlane, and V for Vehicle.

The MDSM transforms this description (D) directly into an equally named concept FP' described (D') in OWL-DL:

```
subClassOf(restriction travelsBy' allValuesFrom(AirPlane'), Passenger')
```

The semantic equivalence of this transformation ($t(D) = D'$) can be shown using Z as follows. Concept FP in the PIM4SWS-IM is specified in Z by

$$\begin{array}{|l} FP : UniRestriction \\ \hline imSubClassOf(FP) = P \end{array}$$

The denoted equality between concepts in this expression is checked by comparing their extensions: The set of instances of the UniRestriction class FP is given by the set of IMIndividual x which has to be a subset of the instances of concept P:

$$\begin{array}{l} classInstances(FP) = \{x : IMIndividual \mid \\ \forall\, a_2 : IMIndividual \bullet \exists\, v : IMPropertyValue \mid v \in propVals(travelsBy) \bullet \\ x = subject(v) \wedge a_2 = object(v) \Rightarrow a_2 \in classInstances(AP)\} \subseteq classInstances(P) \end{array}$$

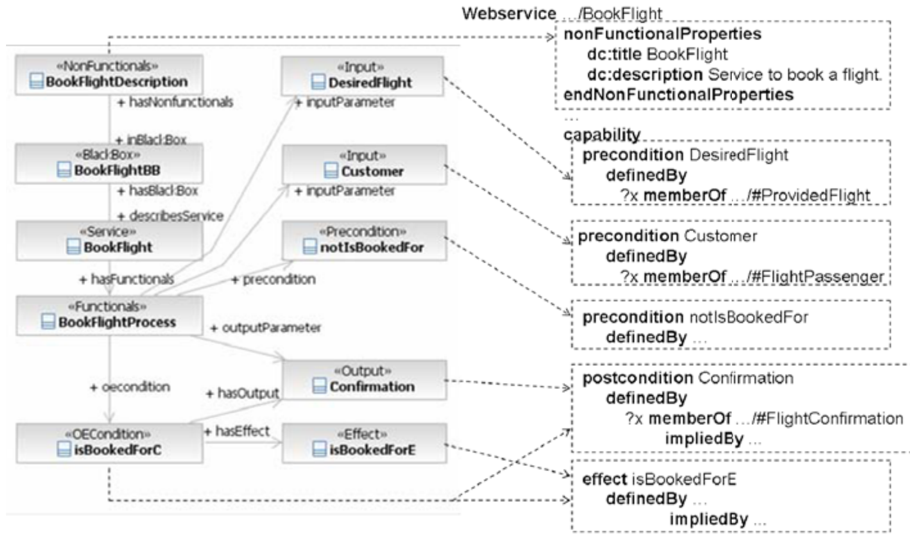


Fig. 7. Bookflight service request transformation from PIM4SWS to WSML.

The Z-specification of the concept FP' in the OWL-DL description D' produced by the MDSM is as follows [17]:

$$\begin{array}{|l} FP' : OWLClass \\ \hline subClassOf(allValuesFrom(AP', travelsBy')) = P' \end{array}$$

According to the Z-element equality definition above, and the semantic Z-equality relations (cf. table 2a-f) of the description logic operations above, the set of instances of FP (in Z) is equal to the OWLClass FP' (in Z):

$$instances(FP') = \{x' : OWLIndividual \mid \forall a_2' : OWLIndividual \bullet$$
$$(x', a_2') \in subVal(travelsBy') \Rightarrow a_2' \in instances(AP')\} \subseteq instances(P')$$

Assuming that $P = P', AP = AP'$ and $travelsBy = travelsBy'$, the semantic equivalence of FP (in PIM4SWS-IM) and its transformation FP' (in OWL-DL) holds. The same is valid for FP' in WSML-DL. FP in *SHIQ(D)* can be specified in Z: `forall travelsBy'.AirPlane'` $\sqsubseteq$ `Passenger'` This expression can be equivalently written in FOL, F-Logic [6, 7] and WSML-DL style.

In FOL: $\forall x(\forall y(travelsBy'(x, y) \supset AirPlane'(y)) \supset Passenger'(x))$;

in F-Logic: $\forall x.(\forall y.x[travelsBy' \twoheadrightarrow y] \supset y : AirPlane') \supset x : Passenger')$;

in WSML-DL:

```
forall ?y(  ?x [travelsBy'  hasValue ?y]  implies ?y memberOf AirPlane')
 implies ?x memberOf Passenger'
```

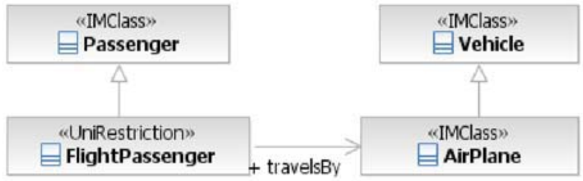


Fig. 8. Part of the PIM4SWS information model for service input concept Flight-Passenger (FP).

The compiled service request in OWL-S, WSML and SAWSDL is passed by the MDSM to its integrated platform-specific matchmakers. Their ranked answer sets are aggregated and eventually presented by the MDSM to the orchestrator.

5 Conclusion

We provided a first approach to model-driven semantic Web service selection to support business process orchestrators at design time. In its inital version, our model-driven service matchmaker MDSM 1.0 is restricted to (a) specific matchmakers for OWL-S, WSML and SAWSDL, and (b) a platform-independent information model defined as intersection of OWL-DL and WSML-DL. However, the principle of model-driven semantic selection applies to other ontology languages and specific matchmakers to be plugged into the MDSM as well. Future work covers the extension of the PIM4SWS information model with OCL constraints, and transformations to SWRL and WSML-Rule [25, 24].

References

1. Acuna, C., Marcos, E.: Modeling Semantic Web Services: A Case Study. *Proc. of the 6th Intl. Conf. on Web Engineering*, ACM Intl. Conf. Proceeding Series; Vol. 263, Palo Alto, USA, 2006.
2. Borgida, A.: On the relative expressiveness of description logics and predicate logics. *Artificial Intelligence*, Vol.: 82, No.: 1-2, p.: 353-367, 1996.
3. Botelho, L., et al.: Service Discovery. *M. Schumacher, H. Helin, H. Schuldt (Eds.) CASCOM - Intelligent Service Coordination in the Semantic Web. Chapter 4.* Birkhuser Verlag, Springer, 2008.
4. Brockmans, S., et al.: Visual Modeling of OWL DL Ontologies Using UML. *In S.A. McIlraith et al., Proc. of the 3rd Intl. Semantic Web Conference*, Springer, Japan, 2004.
5. Chinnici, R., et al.: Web Services Description Language (WSDL) Version 2.0. [Online] Available: http://www.w3c.org/TR/wsdl20/, 2007.
6. de Bruijn, J., Heymans, S.: Translating Ontologies from Predicate-based to Frame-based Languages. *Proc. of the 2nd Intl. Conf. on Rules and Rule Markup Languages for the Semantic Web*, IEEE Computer Society, Washington, DC, USA, 2006.

7. de Bruijn, J., Heymans, S.: WSML Ontology Semantics. *WSML Final Draft*, Available: http://www.wsmo.org/TR/d28/d28.3/v0.1/20061218/, 2006.
8. Duke, R., Rose, G.: Formal Object Oriented Specification Using Object-Z. *Cornerstones of Computing*, Macmillan, 2000.
9. Duric, D.: MDA-based Ontology Infrastructure. *Intl. Journal on Computer Science and Information Systems*, Vol. 1, No. 1, 2004.
10. Gannod, G. C., et al.: Faciliating the Specification of Semantic Web Services Using Model-Driven Development. *Journal of Web Services Research*, 2006.
11. Grønmo, et al.: Transformations between UML and OWL-S. *European Conf. on Model Driven Architecture - Foundations and Applications (ECMDA-FA)*, Nürnberg, Germany, 2005.
12. Kaufer, F., Klusch, M.: WSMO-MX: A Logic Programming Based Hybrid Service Matchmaker. *Proc. of the IEEE Intl. Conf. on Systems, Man and Cybernetics*, Montreal, Canada, 2007.
13. Klusch, M., et al.: OWLS-MX: Hybrid Semantic Web Service Retrieval. *Proc. of the 1st Intl. AAAI Fall Symposium on Agents and the Semantic Web*, Arlington VA, USA, 2005.
14. Koehler, J., et al.: The role of visual modeling and model transformations in business-driven development. *In 5th Intl. Workshop on Graph Transformation and Visual Modeling Techniques*, 2006.
15. Lara, R., Polleres, A.: Formal Mapping and Tool to OWL-S. Available: *http://www.wsmo.org/2004/d4/d4.2/v0.1/20041217/*
16. Lautenbacher, F., Bauer, B.: Creating a Meta-Model for Semantic Web Service Standards. *Proc. of the 3rd Intl. Conf. on Web Information System and Technologies (WEBIST) - Web Interfaces and Applications*, Barcelona, Spain, 2007.
17. Lucanu, D., et al.: Web Ontology Verification and Analysis in the Z Framework. TR 05-01, Faculty of Computer Science, University Alexandru Ioan Cuza, 2005.
18. OWL-S: Semantic Markup for Web Services, *W3C Member Submission*, Available: http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/, 2004.
19. OWL Web Ontology Language Reference, *W3C Recommendation*, Available: http://www.w3.org/TR/2004/REC-owl-ref-20040210/, 2004.
20. Rector, A., Schreiber, G.: Qualified cardinality restrictions (QCRs): Constraining the number of values of a particular type for a property. *W3C Editor's Draft*, Available: http://www.cs.vu.nl/ guus/public/qcr-20060508.html, 2006.
21. Scicluna, J.; et al. (2004): Formal Mapping and Tool to OWL-S. WSMO Working Draft 17 December 2004, Available: http://www.wsmo.org/TR/d4/d4.2/v0.1/.
22. Semantic Annotations for WSDL and XML Schema, *W3C Recommendation*, Available: http://www.w3.org/TR/2007/REC-sawsdl-20070828/, 2007.
23. Spivey, M.: The Z Notation: A Reference Manual, 2nd edition. *Prentice Hall International Series in Computer Science*, 1992.
24. Wang, H. H., et al.: A Formal Semantic Model of the Semantic Web Service Ontology (WSMO). *12th IEEE Intl. Conf. on Engineering of Complex Computer Systems*, Auckland, New Zealand, 2007.
25. Wang, H. H., et al.: Formal Specification of OWL-S with Object-Z. *The First ESWC Workshop on OWL-S: Experiences and Future Directions*, Austria, 2007.
26. Web Service Modeling Ontology (WSMO), *W3C Member Submission*, Available: http://www.w3.org/Submission/2005/SUBM-WSMO-20050603/, 2005.
27. Web Service Modeling Language (WSML), *W3C Member Submission*, Available: http://www.w3.org/Submission/2005/SUBM-WSML-20050603/, 2005.

Combining Boolean Games with the Power of Ontologies for Automated Multi-Attribute Negotiation in the Semantic Web

Thomas Lukasiewicz[1,3] and Azzurra Ragone[2]

[1] Computing Laboratory, University of Oxford
Wolfson Building, Parks Road, Oxford OX1 3QD, UK
thomas.lukasiewicz@comlab.ox.ac.uk

[2] Information Systems Laboratory, Politecnico di Bari
Via E. Orabona 4, 70125 Bari, Italy
a.ragone@poliba.it

Abstract. Recently, multi-attribute negotiation has been extensively studied from a game-theoretic viewpoint. Since normal and extensive form games have the drawback of requiring an explicit representation of utility functions (listing the utility values for all combinations of strategies), logical preference languages have been proposed, which provide a convenient way to compactly specify multi-attribute utility functions. Among these preference languages, there are also Boolean games. In this paper, towards automated multi-attribute negotiation in the Semantic Web, we introduce Boolean description logic games, which are a combination of Boolean games with ontological background knowledge, formulated in expressive description logics. We include and discuss several generalizations, and show how a travel and a service negotiation scenario can be formulated in Boolean description logic games, which shows their practical usefulness.

1 Introduction

During the recent decade, a huge amount of research activities has been centered around the problem of automated negotiation. This is especially due to the development of the World Wide Web, which has provided the means and the commercial necessity for the further development of computational negotiation and bargaining techniques [1].

Another area with an impressive amount of recent research activities is the *Semantic Web* [2,3], which aims at an extension of the current World Wide Web by standards and technologies that help machines to understand the information on the Web so that they can support richer discovery, data integration, navigation, and automation of tasks. The main ideas behind it are to add a machine-readable meaning to Web pages, to use ontologies for a precise definition of shared terms in Web resources, to use knowledge representation technology for automated reasoning from Web resources, and to apply cooperative agent technology for processing the information of the Web.

Only a marginal amount of research activities, however, focuses on the intersection of automated negotiation and the Semantic Web (see Section 6). This is surprising,

[3] Alternative address: Institut für Informationssysteme, Technische Universität Wien, Favoritenstr. 9-11, 1040 Wien, Austria; email: lukasiewicz@kr.tuwien.ac.at.

since representation and reasoning technologies from the Semantic Web may be used to further enhance automated negotiation on the Web, e.g., by providing ontological background knowledge. Moreover, although one important ingredient of the Semantic Web is agent technology, the agents are still largely missing in Semantic Web research to date [4]. This paper is a first step in direction to filling this gap. Towards automated multi-attribute negotiation in the Semantic Web, we introduce Boolean description logic games. The main contributions of this paper are briefly summarized as follows:

– We define Boolean description logic games, which are a combination of n-player Boolean games with description logics. They informally combine n-player Boolean games with ontological background knowledge; in addition, we also introduce strict agent requirements and overlapping agent control assignments.
– We then generalize to Boolean dl-games where each agent has a set of weighted goals, which may be defined over free description logic concepts. We finally propose another generalization, where the agents own roles rather than concepts.
– We provide many examples (from a travel and a service negotiation scenario), which illustrate the introduced concepts related to Boolean description logic games, and which give evidence of the practical usefulness of our approach.

Intuitively, such games aim at a centralized one-step negotiation process, where the agents reveal their preferences to a central mediator, which then calculates one optimal strategy for each agent. Clearly, this is also closely related to service matchmaking and resource retrieval, since the service provider and the service consumer can be both considered as agents having certain service specifications and service preferences, and the result of the negotiation process is then the service where the service specifications are matching optimally the service preferences (see also Example 5.1).

The rest of this paper is organized as follows. In Section 2, we recall the basics of description logics and Boolean games. Section 3 defines Boolean description logic games. In Section 4, we introduce Boolean description logic games with weighted generalized goals. Section 5 generalizes the ontological ownerships. In Section 6, we discuss related work. Section 7 summarizes the main results and gives an outlook on future research.

2 Preliminaries

In this section, we recall the basic concepts of description logics and Boolean games.

2.1 Description Logics

We now recall the description logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$, which stand behind the web ontology languages OWL Lite and OWL DL [5], respectively. Intuitively, description logics model a domain of interest in terms of concepts and roles, which represent classes of individuals and binary relations between classes of individuals, respectively. A description logic knowledge base encodes especially subset relationships between concepts, subset relationships between roles, the membership of individuals to concepts, and the membership of pairs of individuals to roles.

Syntax. We first describe the syntax of $\mathcal{SHOIN}(\mathbf{D})$. We assume a set of *elementary datatypes* and a set of *data values*. A *datatype* is either an elementary datatype or a set of data values (called *datatype oneOf*). A *datatype theory* $\mathbf{D} = (\Delta^{\mathbf{D}}, \cdot^{\mathbf{D}})$ consists of a *datatype domain* $\Delta^{\mathbf{D}}$ and a mapping $\cdot^{\mathbf{D}}$ that assigns to each elementary datatype a subset of $\Delta^{\mathbf{D}}$ and to each data value an element of $\Delta^{\mathbf{D}}$. The mapping $\cdot^{\mathbf{D}}$ is extended to all datatypes by $\{v_1, \ldots\}^{\mathbf{D}} = \{v_1^{\mathbf{D}}, \ldots\}$. Let $\mathbf{A}$, $\mathbf{R}_A$, $\mathbf{R}_D$, and $\mathbf{I}$ be pairwise disjoint (nonempty) denumerable sets of *atomic concepts*, *abstract roles*, *datatype roles*, and *individuals*, respectively. We denote by $\mathbf{R}_A^-$ the set of inverses R^- of all $R \in \mathbf{R}_A$.

A *role* is an element of $\mathbf{R}_A \cup \mathbf{R}_A^- \cup \mathbf{R}_D$. *Concepts* are inductively defined as follows. Every $\phi \in \mathbf{A}$ is a concept, and if $o_1, \ldots, o_n \in \mathbf{I}$, then $\{o_1, \ldots, o_n\}$ is a concept (called *oneOf*). If ϕ, ϕ_1, and ϕ_2 are concepts and if $R \in \mathbf{R}_A \cup \mathbf{R}_A^-$, then also $(\phi_1 \sqcap \phi_2)$, $(\phi_1 \sqcup \phi_2)$, and $\neg\phi$ are concepts (called *conjunction*, *disjunction*, and *negation*, respectively), as well as $\exists R.\phi$, $\forall R.\phi$, $\geqslant nR$, and $\leqslant nR$ (called *exists*, *value*, *atleast*, and *atmost restriction*, respectively) for an integer $n \geqslant 0$. If D is a datatype and $U \in \mathbf{R}_D$, then $\exists U.D$, $\forall U.D$, $\geqslant nU$, and $\leqslant nU$ are concepts (called *datatype exists*, *value*, *atleast*, and *atmost restriction*, respectively) for an integer $n \geqslant 0$. We write $\exists R$ and $\forall R$ to abbreviate $\exists R.\top$ and $\forall R.\top$, respectively. We write $\top$ and $\bot$ to abbreviate the concepts $\phi \sqcup \neg\phi$ and $\phi \sqcap \neg\phi$, respectively, and we eliminate parentheses as usual.

An *axiom* has one of the following forms: (1) $\phi \sqsubseteq \psi$ (called *concept inclusion axiom*), where ϕ and ψ are concepts; (2) $R \sqsubseteq S$ (called *role inclusion axiom*), where either $R, S \in \mathbf{R}_A$ or $R, S \in \mathbf{R}_D$; (3) $\mathrm{Trans}(R)$ (called *transitivity axiom*), where $R \in \mathbf{R}_A$; (4) $\phi(a)$ (called *concept membership axiom*), where ϕ is a concept and $a \in \mathbf{I}$; (5) $R(a, b)$ (resp., $U(a, v)$) (called *role membership axiom*), where $R \in \mathbf{R}_A$ (resp., $U \in \mathbf{R}_D$) and $a, b \in \mathbf{I}$ (resp., $a \in \mathbf{I}$ and v is a data value); and (6) $a = b$ (resp., $a \neq b$) (*equality* (resp., *inequality*) *axiom*), where $a, b \in \mathbf{I}$. A *knowledge base* L is a finite set of axioms. For decidability, number restrictions in L are restricted to simple abstract roles [6]. Since knowledge bases encode ontologies, we also use *ontology* to denote a knowledge base.

The syntax of $\mathcal{SHIF}(\mathbf{D})$ is as the above syntax of $\mathcal{SHOIN}(\mathbf{D})$, but without the oneOf constructor and with the atleast and atmost constructors limited to 0 and 1.

Example 2.1 (travel ontology). A description logic knowledge base L encoding a travel ontology (adapted from http://protege.cim3.net/file/pub/ontologies/travel/) is given by the axioms in Fig. 1. For example, there are some axioms encoding that bed and breakfast accommodations and hotels are different accommodations, and that a budget accommodation is an accommodation that has one or two stars as a rating.

Semantics. An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ w.r.t. a datatype theory $\mathbf{D} = (\Delta^{\mathbf{D}}, \cdot^{\mathbf{D}})$ consists of a nonempty (*abstract*) *domain* $\Delta^{\mathcal{I}}$ disjoint from $\Delta^{\mathbf{D}}$, and a mapping $\cdot^{\mathcal{I}}$ that assigns to each atomic concept $\phi \in \mathbf{A}$ a subset of $\Delta^{\mathcal{I}}$, to each individual $o \in \mathbf{I}$ an element of $\Delta^{\mathcal{I}}$, to each abstract role $R \in \mathbf{R}_A$ a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and to each datatype role $U \in \mathbf{R}_D$ a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathbf{D}}$. We extend $\cdot^{\mathcal{I}}$ to all concepts and roles, and we define the *satisfaction* of an axiom F in an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, denoted $\mathcal{I} \models F$, as usual [5]. We say $\mathcal{I}$ *satisfies* the axiom F, or $\mathcal{I}$ is a *model* of F, iff $\mathcal{I} \models F$. We say $\mathcal{I}$ *satisfies* a knowledge base L, or $\mathcal{I}$ is a *model* of L, denoted $\mathcal{I} \models L$, iff $\mathcal{I} \models F$ for all $F \in L$. We say L is *satisfiable* (resp., *unsatisfiable*) iff L has a (resp., no) model. An axiom F is a *logical consequence* of L, denoted $L \models F$, iff each model of L satisfies F.

$$
\begin{aligned}
&\mathit{BedAndBreakfast} \sqsubseteq \mathit{Accomodation};\\
&\mathit{Hotel} \sqsubseteq \mathit{Accomodation};\\
&\mathit{BedAndBreakfast} \sqsubseteq \neg\mathit{Hotel};\\
&\mathit{BudgetAccommodation} \equiv \mathit{Accomodation} \sqcap \exists \mathit{hasRating}.\{\mathit{OneStarRating}, \mathit{TwoStarRating}\};\\
&\mathit{UrbanArea} \sqsubseteq \mathit{Destination};\\
&\mathit{City} \sqsubseteq \mathit{UrbanArea};\\
&\mathit{Capital} \sqsubseteq \mathit{City};\\
&\mathit{RuralArea} \sqsubseteq \mathit{Destination};\\
&\mathit{NationalPark} \sqsubseteq \mathit{RuralArea};\\
&\mathit{RuralArea} \sqsubseteq \neg\mathit{UrbanArea};\\
&\mathit{BudgetHotelDestination} \equiv \exists \mathit{hasAccomodation}\\
&\qquad\qquad\qquad\qquad\qquad\qquad \sqcap \forall \mathit{hasAccomodation}.(\mathit{BudgetAccommodation} \sqcap \mathit{Hotel});\\
&\mathit{AccommodationRating} \equiv \{\mathit{OneStarRating}, \mathit{TwoStarRating}, \mathit{ThreeStarRating}\};\\
&\mathit{Sightseeing} \sqsubseteq \mathit{Activity};\\
&\mathit{Hiking} \sqsubseteq \mathit{Sport};\\
&\mathit{Sport} \sqsubseteq \mathit{Activity};\\
&\mathit{ThemePark} \sqsubseteq \mathit{Activity};\\
&\mathit{FamilyDestination} \equiv \exists \mathit{hasDestination} \sqcap \exists \mathit{hasAccomodation} \sqcap \geqslant 3\, \mathit{hasActivity};\\
&\mathit{RelaxDestination} \equiv \exists \mathit{hasDestination}.\mathit{NationalPark} \sqcap \exists \mathit{hasActivity}.\mathit{Sightseeing};\\
&\mathit{hasActivity} \equiv \mathit{isOfferedAt}^{-}.
\end{aligned}
$$

Fig. 1. Travel ontology.

Example 2.2 (travel ontology cont'd). It is not difficult to verify that the description logic knowledge base L of Example 2.1 is satisfiable, and that the two concept inclusion axioms *Capital* $\sqsubseteq$ *UrbanArea* and *Capital* $\sqsubseteq$ $\neg$*RuralArea* are logical consequences of L. Informally, L implies that capitals are urban and not rural areas.

2.2 Boolean Games

We now recall n-player Boolean games from [7], which are a generalization of 2-player Boolean games from [8,9]. Such games are essentially normal form games where propositional logic is used for compactly specifying multi-attribute utility functions. We first give some preparative definitions, and then recall n-player Boolean games, including their ingredients, strategy profiles, and important notions of optimality.

We assume a finite set of propositional variables $V = \{p_1, p_2, \ldots, p_k\}$. We denote by $\mathcal{L}_V$ the set of all propositional formulas (denoted by Greek letters $\psi, \phi, \ldots$) built inductively from V via the Boolean operators $\neg$, $\wedge$, and $\vee$.

An *n-player Boolean game* $G = (N, V, \pi, \Phi)$ consists of

(1) a set of n players $N = \{1, 2, \ldots, n\}$, $n \geqslant 2$,
(2) a finite set of propositional variables V,
(3) a *control assignment* $\pi \colon N \to 2^V$, which associates with every player $i \in N$ a set of variables $\pi(i) \subseteq V$, which she controls, such that $\{\pi(i) \mid i \in N\}$ partitions V, and
(4) a *goal assignment* $\Phi \colon N \to \mathcal{L}_V$, which associates with every player $i \in N$ a propositional formula $\Phi(i) \in \mathcal{L}_V$, denoted the *goal* of i.

Example 2.3 (Boolean game). A two-player Boolean game $G = (N, V, \pi, \Phi)$ is given by:

(1) the set of two players $N = \{1, 2\}$,

	b	$\overline{b}$
$a\,c$	(1,0)	(0,1)
$a\,\overline{c}$	(1,0)	**(1,1)**
$\overline{a}\,c$	(0,0)	(0,1)
$\overline{a}\,\overline{c}$	(0,0)	**(1,0)**

Fig. 2. Normal form of a two-player Boolean game.

(2) the set of propositional variables $V = \{a, b, c\}$,
(3) the control assignment $\pi(1) = \{a, c\}$ and $\pi(2) = \{b\}$, and
(4) the goal assignment $\Phi(1) = (a \wedge b) \vee (\neg c \wedge \neg b)$ and $\Phi(2) = (c \wedge \neg b) \vee (a \wedge \neg b)$.

Informally, we have two players 1 and 2, and three propositional variables a, b, and c. Player 1 (resp., 2) controls the variables a and c (resp., the variable b) and has the goal expressed by the propositional formula $\Phi(1)$ (resp., $\Phi(2)$).

A *strategy* for player $i \in N$ is any truth assignment s_i to the variables in $\pi(i)$. A *strategy profile* $s = (s_1, \ldots, s_n)$ consists of one strategy s_i for every $i \in N$. The *utility* to player $i \in N$ under s, denoted $u_i(s)$, is 1, if s satisfies i's goal $\Phi(i)$, and 0, otherwise.

Towards optimal behavior of the players in an n-player Boolean game, we are especially interested in strategy profiles s, called Nash equilibria, where no agent has the incentive to deviate from its part, once the other agents play their parts. More formally, a strategy profile $s = (s_1, \ldots, s_n)$ is a *Nash equilibrium* iff $u_i(s \lhd s'_i) \leqslant u_i(s)$ for every strategy s'_i of player i and for every player $i \in N$, where $s \lhd s'_i$ is the strategy profile obtained from $s = (s_1, \ldots, s_n)$ by replacing s_i by s'_i.

Another important notion of optimality is Pareto-optimality. Informally, a strategy profile is Pareto-optimal if there exists no other strategy profile that makes one player better off and no player worse off in the utility. More formally, a strategy profile s is *Pareto-optimal* iff there exists no strategy profile s' such that (i) $u_i(s') > u_i(s)$ for some player $i \in N$ and (ii) $u_i(s') \geqslant u_i(s)$ for every player $i \in N$.

Example 2.4 (Boolean game cont'd). Consider again the two-player Boolean game $G = (N, V, \pi, \Phi)$ of Example 2.3. Player 1 has all truth assignments to the variables a and c (that is, $a, c \mapsto \mathbf{true}, \mathbf{true}$, $a, c \mapsto \mathbf{true}, \mathbf{false}$, $a, c \mapsto \mathbf{false}, \mathbf{true}$, and $a, c \mapsto \mathbf{false}, \mathbf{false}$, denoted $a\,c$, $a\,\overline{c}$, $\overline{a}\,c$, and $\overline{a}\,\overline{c}$, respectively) as strategies, while player 2 has all truth assignments to b as strategies (that is, $b \mapsto \mathbf{true}$ and $b \mapsto \mathbf{false}$, denoted b and $\overline{b}$, respectively). Any combination of the strategies of two players is a strategy profile. For example, $(a\,c, b)$ is a strategy profile combining the strategy $a\,c$ of player 1 and the strategy b of player 2.

The *normal form* of this two-player Boolean game, using the above strategy profiles $s = (s_1, s_2)$, which combine all strategies s_1 and s_2 of the players 1 and 2, respectively, is depicted in Fig. 2: for every strategy profile $s = (s_1, s_2)$, the matrix has one entry, which shows the pair of utilities $(u_1(s), u_2(s))$ under s to the two players. The utility $u_i(s)$ is equal to 1, when $\Phi(i)$ is satisfied in s, and 0, otherwise.

It is then not difficult to verify that the strategy profile $(a\,\overline{c}, \overline{b})$ is a (pure) Nash equilibrium of this two-player Boolean game G, which is also Pareto-optimal, while $(\overline{a}\,\overline{c}, \overline{b})$ is also a (pure) Nash equilibrium of G, but not Pareto-optimal.

3 Boolean Description Logic Games

In this section, we combine classical n-player Boolean games with ontologies. The main differences to classical n-player Boolean games are summarized as follows (note that many of the new features are also illustrated in Example 3.1):

- Rather than unrelated propositional variables, the agents now control atomic description logic concepts, which may (abbreviate complex description logic concepts and) be related via a description logic knowledge base. In fact, the assumption that the controlled variables are unrelated in classical n-player Boolean games is quite unrealistic; often the variables are related through some background knowledge.
- Rather than having only preferences, the agents may now also have *strict goals*, which have to be necessarily true in an admissible agreement. This reflects the fact that agents accept no agreement where some strict conditions are not true; such strict conditions are very common in many applications in practice.
- Rather than defining a partition, the control assignment may now be overlapping. In fact, such overlapping control assignments are also more realistic.

We first give some preparative definitions as follows. We use a finite set of atomic concepts $\mathcal{A}$ as set of propositional variables V in n-player Boolean games. We denote by $\mathcal{L}_{\mathcal{A}}$ the set of all concepts (denoted by Greek letters $\psi, \phi, \ldots$) built inductively from $\mathcal{A}$ via the Boolean operators $\neg$, $\sqcap$, and $\sqcup$. An *interpretation* I is a full conjunction of atomic concepts and negated atomic concepts from $\mathcal{A}$. We say I *satisfies* a description logic knowledge base L, denoted $I \models L$, iff $L \cup \{I(o)\}$ is satisfiable, where o is a new individual. We say I *satisfies* a concept ϕ over $\mathcal{A}$ under L, denoted $I \models_L \phi$, iff $L \models I \sqsubseteq \phi$. We say ϕ is *satisfiable* under L iff there exists an interpretation I such that $I \models_L \phi$. We are now ready to define n-agent Boolean description logic games.

Definition 3.1 **(n-agent Boolean description logic games).** An *n-agent Boolean description logic game* (or *n-agent Boolean dl-game*) $G = (L, N, \mathcal{A}, \pi, \Sigma, \Phi)$ consists of

(1) a description logic knowledge base L,
(2) a finite set of n agents $N = \{1, 2, \ldots, n\}$, $n \geqslant 2$,
(3) a finite set of atomic concepts $\mathcal{A}$,
(4) a *control assignment* $\pi\colon N \to 2^V$, which associates with every agent $i \in N$ a set of atomic concepts $\pi(i) \subseteq \mathcal{A}$, which she controls,
(5) a *strict goal assignment* $\Sigma\colon N \to \mathcal{L}_{\mathcal{A}}$, which associates with every agent $i \in N$ a concept $\Sigma(i) \in \mathcal{L}_{\mathcal{A}}$ that is satisfiable under L, denoted the *strict goal* of i, and
(6) a *goal assignment* $\Phi\colon N \to \mathcal{L}_{\mathcal{A}}$, which associates with every agent $i \in N$ a concept $\Phi(i) \in \mathcal{L}_{\mathcal{A}}$ that is satisfiable under L, denoted the *goal* of i.

As for the difference between strict and general goals, the agents necessarily want their strict goals to be satisfied, but they only would like their general goals to be satisfied. The following example illustrates n-agent Boolean dl-games.

Example 3.1 (travel negotiation). A two-agent Boolean dl-game $G = (L, N, \mathcal{A}, \pi, \Sigma, \Phi)$, where the *traveler* (agent 1) negotiates with the *travel agency* (agent 2) on the conditions of a vacation, is given as follows:

(1) L is the travel ontology of Example 2.1, depicted in Fig. 1.
(2) $N = \{1, 2\}$, where agent 1 (resp., 2) is the *traveler* (resp., *travel agent*).
(3) $\mathcal{A}$ consists of the following atomic concepts (that are relevant to the negotiation):

$$\begin{array}{ll}
U & \equiv \exists hasDestination \sqcap \forall hasDestination.UrbanArea; \\
R & \equiv \exists hasDestination \sqcap \forall hasDestination.RuralArea; \\
BHD & \equiv BudgetHotelDestination; \\
BA & \equiv \exists hasAccomodation \sqcap \forall hasAccomodation.BudgetAccommodation; \\
H & \equiv \exists hasAccomodation \sqcap \forall hasAccomodation.Hotel; \\
BB & \equiv \exists hasAccomodation \sqcap \forall hasAccomodation.BedAndBreakfast; \\
NP & \equiv \exists hasDestination \sqcap \forall hasDestination.NationalPark; \\
C & \equiv \exists hasDestination \sqcap \forall hasDestination.Capital.
\end{array}$$

(4) Agents 1 and 2 control the following concepts $\pi(1)$ and $\pi(2)$, respectively:

$$\pi(1) = \{U, R, BHD\};$$
$$\pi(2) = \{BA, H, BB, NP, C\}.$$

Informally, agent 1 decides whether the trip takes place to an urban, rural, or budget hotel destination, while agent 2's offers fix the budget, hotel, or bed and breakfast accommodation, and the destination to a national park or a capital city.
(5) Agents 1 and 2 have the following strict goals $\Sigma(1)$ and $\Sigma(2)$, respectively:

$$\Sigma(1) = (U \sqcup R) \sqcap (H \sqcup BB);$$
$$\Sigma(2) = NP \sqcup C.$$

Informally, agent 1 necessarily wants a destination in an urban or a rural area, e.g., she does not like beach destinations, and she also wants an accommodation for her trip in a hotel or a bed and breakfast, so she is excluding e.g. camping grounds. Whereas agent 2 is trying to sell a destination in a national park or a capital city.
(6) Agents 1 and 2 have the following goals $\Phi(1)$ and $\Phi(2)$, respectively,

$$\Phi(1) = (R \sqcap BB) \sqcup (C \sqcap BHD);$$
$$\Phi(2) = (U \sqcap BB) \sqcup (NP \sqcap BHD).$$

Informally, agent 1 would like a destination in a rural area and an accommodation in a bed and breakfast, or a budget hotel accommodation in a capital city. Whereas agent 2 would like to sell a destination in an urban area and an accommodation in a bed and breakfast, or a budget hotel destination in a national park.

We next define the notions of strategies, strategy profiles, and utility functions. In classical n-agent Boolean games, a strategy for agent i is a truth assignment s_i to all the variables she controls, and the utility functions of the agents depend on their goals built from the variables. In our setting, in contrast, atomic concepts are related to each other through a description logic knowledge base L, and each agent may have some strict requirements, and so some truth assignments to the atomic concepts may be infeasible because of L and the strict requirements. We thus exclude such infeasible strategies. In addition, some combinations I of feasible strategies may result in an infeasible strategy profile due to L and the fact that the control assignment may be overlapping. We model

	$U \sqcap \neg R \sqcap BHD$	$\neg U \sqcap R \sqcap BHD$	$U \sqcap \neg R \sqcap \neg BHD$	$\neg U \sqcap R \sqcap \neg BHD$
$BA \sqcap H \sqcap \neg BB \sqcap NP \sqcap \neg C$	$(-1,-1)$	$(\mathbf{0},\mathbf{1})$	$(-1,-1)$	$(0,0)$
$BA \sqcap \neg H \sqcap BB \sqcap NP \sqcap \neg C$	$(-1,-1)$	$(-1,-1)$	$(-1,-1)$	$(\mathbf{1},\mathbf{0})$
$BA \sqcap H \sqcap \neg BB \sqcap \neg NP \sqcap C$	$(\mathbf{1},\mathbf{0})$	$(-1,-1)$	$(\mathbf{0},\mathbf{0})$	$(-1,-1)$
$BA \sqcap \neg H \sqcap BB \sqcap \neg NP \sqcap C$	$(-1,-1)$	$(-1,-1)$	$(\mathbf{0},\mathbf{1})$	$(-1,-1)$
$\neg BA \sqcap H \sqcap \neg BB \sqcap NP \sqcap \neg C$	$(-1,-1)$	$(-1,-1)$	$(-1,-1)$	$(0,0)$
$\neg BA \sqcap \neg H \sqcap BB \sqcap NP \sqcap \neg C$	$(-1,-1)$	$(-1,-1)$	$(-1,-1)$	$(\mathbf{1},\mathbf{0})$
$\neg BA \sqcap H \sqcap \neg BB \sqcap \neg NP \sqcap C$	$(-1,-1)$	$(-1,-1)$	$(\mathbf{0},\mathbf{0})$	$(-1,-1)$
$\neg BA \sqcap \neg H \sqcap BB \sqcap \neg NP \sqcap C$	$(-1,-1)$	$(-1,-1)$	$(\mathbf{0},\mathbf{1})$	$(-1,-1)$

Fig. 3. Normal form of a two-agent Boolean dl-game.

this, exploiting the utility structure: if I is infeasible due to L or the overlapping control assignment, then the utility to all agents is -1; in contrast, if I is feasible, then the utility to agent i under I is equal to 1, if its goal $\Phi(i)$ is satisfied, and 0, otherwise. Therefore, when the agreement I is unsatisfiable, then the utilities are always negative, that is, always less than the utilities when the agreement I is satisfiable. Hence, the unsatisfiable agreement will never be chosen by the agents.

Definition 3.2 (strategies, strategy profiles, utilities). Let $G = (L, N, \mathcal{A}, \pi, \Sigma, \Phi)$ be an n-agent Boolean dl-game. Then, a *strategy* for agent $i \in N$ is an interpretation I_i for the concepts in $\pi(i)$ that satisfies both (i) L and (ii) $\Sigma(i)$ under L. A *strategy profile* $I = (I_1, I_2, \ldots, I_n)$ consists of one strategy I_i for every agent $i \in N$. We say $I = (I_1, I_2, \ldots, I_n)$ is *consistent* iff (i) there exists an interpretation J for $\mathcal{A}$ such that I_i is the restriction of J to $\pi(i)$, for every agent $i \in N$, and (ii) I satisfies L. The *utility* to agent $i \in N$ under I, denoted $u_i(I)$, is defined as follows:

$$u_i(I) \;=\; \begin{cases} -1 & \text{if } I \text{ is inconsistent, or } I \not\models_L \Sigma(i);\\ 1 & \text{if } I \text{ is consistent, } I \models_L \Sigma(i), \text{ and } I \models_L \Phi(i);\\ 0 & \text{if } I \text{ is consistent, } I \models_L \Sigma(i), \text{ and } I \not\models_L \Phi(i). \end{cases}$$

We illustrate the above ideas with the help of a simple example.

Example 3.2 (travel negotiation cont'd). The sets of all strategies $\mathcal{I}_1$ and $\mathcal{I}_2$ of agents 1 and 2, respectively, in the travel negotiation example are given as follows:

$$\begin{aligned} \mathcal{I}_1 = \{ & BA \sqcap H \sqcap \neg BB \sqcap NP \sqcap \neg C,\ BA \sqcap \neg H \sqcap BB \sqcap NP \sqcap \neg C,\\ & BA \sqcap H \sqcap \neg BB \sqcap \neg NP \sqcap C,\ BA \sqcap \neg H \sqcap BB \sqcap \neg NP \sqcap C,\\ & \neg BA \sqcap H \sqcap \neg BB \sqcap NP \sqcap \neg C,\ \neg BA \sqcap \neg H \sqcap BB \sqcap NP \sqcap \neg C,\\ & \neg BA \sqcap H \sqcap \neg BB \sqcap \neg NP \sqcap C,\ \neg BA \sqcap \neg H \sqcap BB \sqcap \neg NP \sqcap C\};\\ \mathcal{I}_2 = \{ & U \sqcap \neg R \sqcap BHD,\ \neg U \sqcap R \sqcap BHD,\ U \sqcap \neg R \sqcap \neg BHD,\ \neg U \sqcap R \sqcap \neg BHD\}. \end{aligned}$$

The set of all strategy profiles is $\mathcal{I}_1 \times \mathcal{I}_2$. The utility pairs $(u_1(I), u_2(I))$ for each strategy profile $I = (I_1, I_2)$ are shown in Fig. 3, which actually depicts the normal form of the two-agent Boolean dl-game G. Note that all inconsistent strategy profiles (due to the description logic knowledge base L) are associated with two negative utilities.

We next define (pure) Nash equilibria of n-agent Boolean dl-games. Informally, as in the classical case, they are strategy profiles where no agent has the incentive to deviate from its part once the other agents stick to their parts.

Definition 3.3 (pure Nash equilibria). Let $G = (L, N, \mathcal{A}, \pi, \Phi)$ be an n-agent Boolean dl-game with $N = \{1, \dots, n\}$. Then, a strategy profile $I = (I_1, \dots, I_n)$ is a *(pure) Nash equilibrium* of G iff $u_i(I \lhd I'_i) \leqslant u_i(I)$ for every strategy I'_i of agent i and for every agent $i \in N$, where $I \lhd I'_i$ is the strategy profile obtained from I by replacing I_i by I'_i.

Another concept of optimality for strategy profiles is the notion of Pareto-optimality. Informally, a strategy profile is Pareto-optimal if there exists no other strategy profile that makes one agent better off and no agent worse off in the utility. Note that, as in the classical case, Nash equilibria are not necessarily Pareto-optimal.

Definition 3.4 (Pareto-optimal strategy profiles). Let $G = (L, N, \mathcal{A}, \pi, \Phi)$ be an n-agent Boolean dl-game with $N = \{1, \dots, n\}$. Then, a strategy profile $I = (I_1, \dots, I_n)$ is *Pareto-optimal* iff there exists no strategy profile I' such that (i) $u_i(I') > u_i(I)$ for some agent $i \in N$ and (ii) $u_i(I') \geqslant u_i(I)$ for every agent $i \in N$.

We illustrate the notions of Nash equilibria and Pareto-optimality in our example.

Example 3.3 (travel negotiation cont'd). The set of all (pure) Nash equilibria of the two-agent Boolean dl-game G of Example 3.1 are given by the bold entries in Fig. 3. It is not difficult to verify that all except for the $(0, 0)$ ones are also Pareto-optimal.

4 Weighted Generalized Goals

In this section, we further extend Boolean dl-games by weighted and generalized goals:

- Instead of one single goal that each agent wants to satisfy, we now assume a set of goals for each agent, where each goal of an agent is associated with a weight. This considers the fact that goals can have different importance, so the best agreement is not necessarily the agreement satisfying the greatest number of goals for each agent. We first define Boolean dl-games with weighted goals, that is, multi-valued preferences. Note that agent utilities are normalized to 1 to make them comparable.
- As another difference to Boolean dl-games, we also do not assume anymore that agent goals are constructed from the controlled atomic concepts.

Definition 4.1 (n-agent Boolean dl-games with weighted goals). An *n-agent Boolean dl-game with weighted goals* $G = (L, N, \mathcal{A}, \pi, \Sigma, \Phi)$ consists of

(1) a description logic knowledge base L,
(2) a finite set of n agents $N = \{1, 2, \dots, n\}$, $n \geqslant 2$,
(3) a finite set of atomic concepts $\mathcal{A}$,
(4) a *control assignment* $\pi \colon N \to 2^V$, which associates with every agent $i \in N$ a set of atomic concepts $\pi(i) \subseteq \mathcal{A}$, which she controls,
(5) a *strict goal assignment* $\Sigma \colon N \to \mathcal{L}_{\mathcal{A}}$, which associates with every agent $i \in N$ a concept $\Sigma(i) \in \mathcal{L}_{\mathcal{A}}$ that is satisfiable under L, denoted the *strict goal* of i, and
(6) a *weighted goal assignment* Φ, which associates with every agent $i \in N$ a mapping Φ_i from a finite set of concepts $\mathcal{L}_i$ that are satisfiable under L (denoted the *weighted goals* of i) to $\Re^+$ such that $\sum_{\phi \in \mathcal{L}_i} \Phi_i(\phi) = 1$.

We give an example of a Boolean dl-game with weighted goals.

Example 4.1 (travel negotiation cont'd). A two-agent Boolean dl-game with weighted goals $G' = (L', N', \mathcal{A}', \pi', \Sigma', \Phi')$ for the travel negotiation example is obtained from the two-agent Boolean dl-game $G = (L, N, \mathcal{A}, \pi, \Sigma, \Phi)$ of Example 3.1 as follows:

(1) $L' = L$.
(2) $N' = N$.
(3) $\mathcal{A}'$ consists of the atomic concepts in $\mathcal{A}$ and the following new ones:

$$\begin{aligned} &\mathit{TP} \equiv \exists \mathit{hasActivity}.\mathit{ThemePark};\\ &\mathit{SS} \equiv \exists \mathit{hasActivity}.\mathit{Sightseeing};\\ &\mathit{HK} \equiv \exists \mathit{hasActivity}.\mathit{Hiking}. \end{aligned}$$

(4) Agents 1 and 2 control the following concepts $\pi(1)$ and $\pi(2)$, respectively:

$$\begin{aligned} \pi(1) &= \{U, R, \mathit{BHD}, \mathit{SS}, \mathit{HK}\};\\ \pi(2) &= \{\mathit{BA}, H, \mathit{BB}, \mathit{NP}, C, \mathit{TP}\}. \end{aligned}$$

More concretely, compared to Example 3.1, the agents now control more variables, namely, *Sightseeing* and *Hiking* for agent 1, and *ThemePark* for agent 2.

(5) Agents 1 and 2 have the following strict goals $\Sigma(1)$ and $\Sigma(2)$, respectively:

$$\begin{aligned} \Sigma(1) &= (U \sqcup R) \sqcap (H \sqcup \mathit{BB}) \sqcap \mathit{BHD};\\ \Sigma(2) &= (\mathit{NP} \sqcup C) \sqcap {\geqslant} 1\, \mathit{hasActivity}. \end{aligned}$$

More specifically, compared to Example 3.1, the agents 1 and 2 now also require *BudgetHotelDestination* and $\geqslant 1$ *hasActivity*, respectively, in the strict goals. Informally, agent 1 also wants a budget hotel destination, while agent 2 wants to include in the travel package that she is trying to sell at least one activity.

(6) Agents 1 and 2 have the following weighted goals Φ_1 and Φ_2, respectively,

$$\begin{aligned} &\Phi_1(\mathit{FamilyDestination}) &&= 0.3;\\ &\Phi_1(\mathit{RelaxDestination}) &&= 0.3;\\ &\Phi_1(\exists \mathit{hasDestination}.(\mathit{Capital} \sqcup \mathit{RuralArea}) \sqcap &&\\ &\qquad\qquad \exists \mathit{hasActivity}.(\mathit{Sport} \sqcap \mathit{ThemePark})) &&= 0.4; \end{aligned}$$

$$\begin{aligned} &\Phi_2(\exists \mathit{hasDestination}.\mathit{RuralArea} \sqcap \exists \mathit{hasActivity}.\mathit{Sightseeing}) &&= 0.3;\\ &\Phi_2(\mathit{FamilyDestination} \sqcap \exists \mathit{hasActivity}.\mathit{ThemePark}) &&= 0.3;\\ &\Phi_2(\mathit{RelaxDestination} \sqcap \exists \mathit{hasActivity}.\mathit{Hiking}) &&= 0.4. \end{aligned}$$

Informally, agent 1 would like either (a) a family destination, or (b) a relax destination, or (c) a capital or rural destination with sports activities in a theme park, the latter with a slightly higher weight. Whereas agent 2 would like to sell either (a) a destination in a rural area with sightseeing, or (b) a family destination with theme park, or (c) a relax destination with hiking, the latter with slightly higher weight.

The notions of strategies and strategy profiles along with the consistency of strategy profiles are defined in the same way as for Boolean dl-games with binary goals. The following definition extends the notion of utility to weighted goals.

	$BA \sqcap H \sqcap \neg BB \sqcap NP \sqcap \neg C \sqcap TP$	$BA \sqcap H \sqcap \neg BB \sqcap \neg NP \sqcap C \sqcap TP$	$BA \sqcap H \sqcap \neg BB \sqcap NP \sqcap \neg C \sqcap \neg TP$	$BA \sqcap H \sqcap \neg BB \sqcap \neg NP \sqcap C \sqcap \neg TP$
$U \sqcap \neg R \sqcap BHD \sqcap SS \sqcap HK$	$(-1,-1)$	$(\mathbf{0.7},\mathbf{0.3})$	$(-1,-1)$	$(0.4,0)$
$\neg U \sqcap R \sqcap BHD \sqcap SS \sqcap HK$	$(\mathbf{1},\mathbf{1})$	$(-1,-1)$	$(0.7,0.7)$	$(-1,-1)$
$U \sqcap \neg R \sqcap BHD \sqcap SS \sqcap \neg HK$	$(-1,-1)$	$(0.4,0)$	$(-1,-1)$	$(0,0)$
$\neg U \sqcap R \sqcap BHD \sqcap SS \sqcap \neg HK$	$(0.7,0.3)$	$(-1,-1)$	$(0.3,0.3)$	$(-1,-1)$
$U \sqcap \neg R \sqcap BHD \sqcap \neg SS \sqcap HK$	$(-1,-1)$	$(0.4,0)$	$(-1,-1)$	$(\mathbf{0.4},\mathbf{0})$
$\neg U \sqcap R \sqcap BHD \sqcap \neg SS \sqcap HK$	$(0.4,0)$	$(-1,-1)$	$(0.4,0)$	$(-1,-1)$

Fig. 4. Normal form of a two-agent Boolean dl-game with weighted generalized goals.

Definition 4.2 (utilities with weighted goals). Let $G = (L, N, \mathcal{A}, \pi, \Phi, \Sigma)$ be an n-agent Boolean dl-game with weighted goals. Then, the *utility* to agent $i \in N$ under I, denoted $u_i(I)$, is defined as follows:

$$u_i(I) \;=\; \begin{cases} -1 & \text{if } I \text{ is inconsistent, or } I \not\models_L \Sigma(i); \\ \Sigma_{\phi \in \mathcal{L}_i,\, I \models_L \phi}\, \Phi_i(\phi) & \text{if } I \text{ is consistent, } I \models L, \text{ and } I \models_L \Sigma(i). \end{cases}$$

We give an example to illustrate the utilities in the case of weighted goals.

Example 4.2 (travel negotiation cont'd). The normal form representation of the two-agent Boolean dl-game with weighted goals G of Example 4.1 is depicted in Fig. 4. Its only (pure) Nash equilibria are given by the bold entries in Fig. 4. Observe that the Nash equilibrium with utility pair $(1, 1)$ is also Pareto-optimal.

5 Controlling Roles

In this section, we present a further generalization of Boolean dl-games where agents control roles instead of concepts. In this case, every strategy is intuitively an instantiation of concepts. We also provide a further application scenario from web service negotiation, along which we sketch this generalization of Boolean dl-games.

Example 5.1 (web service negotiation). Consider a service negotiation scenario, where a service provider (agent 2) and a service requester (agent 1) are negotiating on the conditions of a supply. The description logic knowledge base L is given by the ontology in Fig. 5. We assume the set of two agents $N = \{1, 2\}$. The roles $\pi(1)$ and $\pi(2)$ controlled by agents 1 and 2, respectively, are given as follows:

$\pi(1) = \{\mathit{delivery}, \mathit{hasQuality}\}$;
$\pi(2) = \{\mathit{hasType}\}$.

$EU \sqsubseteq WorldWide$;
$US \sqsubseteq WorldWide$;
$Contract1 \sqsubseteq Contract$;
$Contract2 \sqsubseteq Contract$;
$Contract1 \sqsubseteq \neg Contract2$;
$Cash \sqsubseteq PaymentType$;
$Instalments \sqsubseteq PaymentType$;
$HighQualityService \sqsubseteq \exists assistance \sqcap \forall assistance.Onsite \sqcap =2\, year\_guarantee$;
$LowQualityService \sqsubseteq \exists assistance \sqcap \forall assistance.Phone \sqcap =1\, year\_guarantee$;
$Contract1 \equiv \exists payment \sqcap \forall payment.Instalments \sqcap \exists delivery \sqcap \forall delivery.(US \sqcap EU)$;
$Contract2 \equiv \exists payment \sqcap \forall payment.Cash \sqcap \exists delivery \sqcap \forall delivery.WorldWide$.

Fig. 5. Service ontology.

	$C1$	$C2$
$HQ \sqcap WW$	$(-1,-1)$	$(\mathbf{0},\mathbf{1})$
$HQ \sqcap SE$	$(1,0)$	$(\mathbf{0},\mathbf{1})$

Fig. 6. Normal form of a two-agent Boolean dl-game with controlled roles.

Agents 1 and 2 have the following goals $\Phi(1)$ and $\Phi(2)$, respectively (for ease of presentation, we omit strict and weighted goals here):

$$\begin{aligned}\Phi(1) &= \exists payment \sqcap \forall payment.Instalments;\\ \Phi(2) &= (\exists hasQuality \sqcap \forall hasQuality.LowQualityService \sqcap\\ &\quad \exists hasType \sqcap \forall hasType.Contract1) \sqcup\\ &\quad (\exists hasQuality \sqcap \forall hasQuality.HighQualityService \sqcap\\ &\quad \exists hasType \sqcap \forall hasType.Contract2).\end{aligned}$$

The normal form of the two-agent Boolean dl-game is depicted in Fig. 6, where (for the sake of conciseness) we define the following atomic concepts:

$$\begin{aligned}C1 &\equiv \exists hasType \sqcap \forall hasType.Contract1;\\ C2 &\equiv \exists hasType \sqcap \forall hasType.Contract2;\\ HQ &\equiv \exists hasQuality \sqcap \forall hasQuality.HighQualityService;\\ WW &\equiv \exists delivery \sqcap \forall delivery.WorldWide;\\ SE &\equiv \exists delivery \sqcap \forall delivery.(US \sqcap EU).\end{aligned}$$

Notice that in this approach agents do not have to enumerate all the possible combinations of concepts they control, as before, but, as they control roles instead of concepts, it is enough to consider only concepts that they are interested in, such as e.g. for agent 1 *HighQualityService* or for agent 2 only the type of contracts she wants to offer. This approach is surely more compact than the previous one, even if it could be not exhaustive and give more power w.r.t. some attributes to one agent, the one controlling the role indeed can control an entire set of attributes, e.g., thanks to the control on *hasType*, agent 2 is the only one that can choose what type of contract to offer.

6 Related Work

A large number of negotiation mechanisms have been proposed and studied in the literature. It is possible to distinguish, among others, game-theoretic ones [10,11], heuristic-based approaches [12,13] and logic-based approaches. Although pure game-theoretic and heuristic-based approaches are highly suitable for a wide range of applications, they have some limitations and disadvantages. Often in game-theoretic approaches, it is assumed that no relation exists between agent's strategies and that all the combinations of strategies are possible. Moreover, they usually do not model relations about issues, which is, instead, fundamental in multi-attribute negotiation. On the other hand, heuristic-based approaches use empirical evaluations to find an agreement, which can be sub-optimal, as they do not explore the entire space of possible outcomes.

In the following, we give a brief overview of logic-based approaches to automated negotiation, comparing our approach to existing ones and highlighting relevant differences. There is an extensive literature on argumentation-based negotiation [14,15,16]. In these approaches, an agent can accept/reject/critique a proposal of its opponent, so agents can argue about their beliefs, given their desires and so pursue their intentions. With respect to our framework, these approaches require a larger number of communication rounds in order to exchange information, while our approach is a one-shot negotiation, which ensures the termination after only one round; indeed in argumentation-based frameworks, usually, agent interactions go back and forth for multiple rounds.

Several recent logic-based approaches to negotiation are based on propositional logic. Bouveret et al. [17] use weighted propositional formulas (WPFs) to express agent preferences in the allocation of indivisible goods, but no common knowledge (as our ontology) is present. The use of an ontology allows, e.g., to discover inconsistencies between strategies, as well as attributes, or find out if an agent preference is implied by a combination of strategies (an interpretation) which is fundamental to model a multi-attribute negotiation. Chevaleyre et al. [18] classify utility functions expressed through WPFs according to the properties of the utility function (sub/super-additive, monotone, etc.). We used the most expressive functions according to that classification, namely, weights over unrestricted formulas. Zhang and Zhang [19] adopt a kind of propositional knowledge base arbitration to choose a fair negotiation outcome. However, *common knowledge* is considered as just more entrenched preferences, that could be even dropped in some deals. Instead, the logical constraints in our ontology must *always* be enforced in the negotiation outcomes. Wooldridge and Parsons [20] define an agreement as a model for a set of formulas from both agents. However, Wooldridge and Parsons [20] only study multiple-rounds protocols and the approach leaves the burden to reach an agreement to the agents themselves, although they can follow a protocol. The approach does not take preferences into account, so that it is not possible to compute utility values and check if the reached agreement is Pareto-optimal or a Nash equilibrium. In the work by Ragone et al. [21], a basic propositional logic framework endowed with an ontology was proposed, which is further extended in [22], introducing the extended logic $\mathcal{P}(\mathcal{N})$ (a propositional logic with concrete domains), thus handling numerical features, and showed how to compute Pareto-optimal agreements, by solving an optimization problem and adopting a one-shot negotiation protocol.

For what concerns approaches using more expressive ontology languages, namely, description logics, there is the work by Ragone et al. [23], which although uses a rather inexpressive description logic, $\mathcal{ALEH}(D)$, proposes a semantic-based alternating-offers protocol exploiting non-standard inference services, as concept contraction, and utility theory to find the most suitable agreements. Concept contraction can be useful to provide an explanation of "what is wrong" between request and offer, that is, the reason why agents cannot reach an agreement and *what* has to be given up in order to reach that. Furthermore, differently from our approach, no game-theoretic analysis is provided about Nash equilibria, even if in this framework, agents do not have to reveal their utilities to the opponent. Another work exploits description logics in negotiation scenarios [24], where the more expressive $\mathcal{SHOIN}(\mathbf{D})$ is used to model the logic-based negotiation protocol; a scenario with *fully* incomplete information is studied, where agents do not know anything about the opponent (neither preferences nor utilities). Furthermore, also this framework lacks a game-theoretic analysis about Nash equilibria.

7 Summary and Outlook

Towards automated multi-attribute negotiation in the Semantic Web, we have introduce Boolean description logic games, which combine classical Boolean games with expressive description logics. As further generalizations of classical Boolean games, they also include strict agent requirements and overlapping agent control assignments. We have also considered two generalizations, one with weighted goals, which may be defined over free description logic concepts, and one where the agents own roles rather than concepts. Furthermore, formulations of a travel and a service negotiation scenario have given evidence of the practical usefulness of our approach.

An interesting topic for future research is to more deeply analyze the semantic and the computational properties of Boolean description logic games. In particular, an important issue is the development of algorithms for computing optimal strategy profiles, and the analysis of its computational complexity. Furthermore, it would be interesting to implement a tool for solving Boolean dl-games and testing it on negotiation scenarios. Another topic for future research is a generalization to qualitative conditional preference structures, such as the ones expressed through CP-nets [25]. From a larger perspective, Boolean dl-games aim at a centralized one-step negotiation process, where the agents reveal their preferences to a central mediator, which then calculates one optimal strategy for each agent. In this framework, it is important to study how it is possible to avoid that the agents report untruthful preferences in order to obtain better strategies, which is touching the problem of mechanism design [26].

Acknowledgments. Azzurra Ragone wishes to thank Francesco M. Donini and Tommaso Di Noia for fruitful suggestions and discussions. This work has been partially supported by the Apulia Region funded project PS_092 Distributed Production to Innovative System (DIPIS), by TOWL project (Time-determined ontology based information system for real time stock) EU STREP FP6 (2006-2008), by the Austrian Science Fund (FWF) under the project P18146-N04, and by the German Research Foundation (DFG) under the Heisenberg Programme.

References

1. Jennings, N.R., Faratin, P., Lomuscio, A.R., Parsons, S., Wooldridge, M., Sierra, C.: Automated negotiation: Prospects, methods and challenges. Group Decision and Negotiation **10** (2001) 199–215

2. Berners-Lee, T.: Weaving the Web. Harper, San Francisco, CA (1999)
3. Fensel, D., Wahlster, W., Lieberman, H., Hendler, J., eds.: Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential. MIT Press (2002)
4. Hendler, J.A.: Where are all the intelligent agents? IEEE Intelligent Systems **22**(3) (2007) 2–3
5. Horrocks, I., Patel-Schneider, P.F.: Reducing OWL entailment to description logic satisfiability. In: Proc. ISWC-2003. Volume 2870 of LNCS, Springer (2003) 17–29
6. Horrocks, I., Sattler, U., Tobies, S.: Practical reasoning for expressive description logics. In: Proc. LPAR-1999. Volume 1705 of LNCS, Springer (1999) 161–180
7. Bonzon, E., Lagasquie-Schiex, M.C., Lang, J., Zanuttini, B.: Compact preference representation and Boolean games. Autonomous Agents and Multi-Agent Systems (2008) In press.
8. Harrenstein, P., van der Hoek, W., Meyer, J.J.C., Witteveen, C.: Boolean games. In: Proc. TARK-2001, Morgan Kaufmann (2001) 287–298
9. Harrenstein, P.: Logic in Conflict. PhD thesis, Utrecht University, The Netherlands (2004)
10. Kraus, S.: Strategic Negotiation in Multiagent Environments. MIT Press (2001)
11. Rosenschein, J.S., Zlotkin, G.: Rules of Encounter. MIT Press (1994)
12. Fatima, S., Wooldridge, M., Jennings, N.R.: Optimal agendas for multi-issue negotiation. In: Proc. AAMAS-2003, ACM Press (2003) 129–136
13. Faratin, P., Sierra, C., Jennings, N.R.: Using similarity criteria to make issue trade-offs in automated negotiations. Artificial Intelligence **142**(2) (2002) 205–237
14. Rahwan, I., Ramchurn, S.D., Jennings, N.R., McBurney, P., Parsons, S.: Argumentation-based negotiation. The Knowledge Engineering Review **18**(4) (2003) 343–375
15. Sadri, F., Toni, F., Torroni, P.: Dialogues for negotiation: Agent varieties and dialogue sequences. In: Intelligent Agents VIII. Volume 2333 of LNCS, Springer (2002) 405–421
16. Bentahar, J., Moulin, B., Meyer, J.J.C., Chaib-draa, B.: A modal semantics for an argumentation-based pragmatics for agent communication. In: Argumentation in Multi-Agent Systems. Volume 3366 of LNCS, Springer (2005) 44–63
17. Bouveret, S., Lemaitre, M., Fargier, H., Lang, J.: Allocation of indivisible goods: A general model and some complexity results. In: Proc. AAMAS-2005, ACM Press (2005) 1309–1310
18. Chevaleyre, Y., Endriss, U., Lang, J.: Expressive power of weighted propositional formulas for cardinal preference modeling. In: Proc. KR-2006, AAAI Press (2006) 145–152
19. Zhang, D., Zhang, Y.: A computational model of logic-based negotiation. In: Proc. AAAI-2006, AAAI Press (2006) 728–733
20. Wooldridge, M., Parsons, S.: Languages for negotiation. In: Proc. ECAI-2000, IOS Press (2000) 393–400
21. Ragone, A., Di Noia, T., Di Sciascio, E., Donini, F.M.: A logic-based framework to compute Pareto agreements in one-shot bilateral negotiation. In: Proc. ECAI-2006, IOS Press (2006) 230–234
22. Ragone, A., Di Noia, T., Di Sciascio, E., Donini, F.M.: Logic-based automated multi-issue bilateral negotiation in peer-to-peer e-marketplaces. Autonomous Agents and Multi-Agent Systems **16**(3) (2008) 249–270
23. Ragone, A., Di Noia, T., Di Sciascio, E., Donini, F.M.: Alternating-offers protocol for multi-issue bilateral negotiation in semantic-enabled marketplaces. In: Proc. ISWC-2007. Volume 4825 of LNCS. Springer (2007) 395–408
24. Ragone, A., Di Noia, T., Di Sciascio, E., Donini, F.M.: Description logics for multi-issue bilateral negotiation with incomplete information. In: Proc. AAAI-2007, AAAI Press (2007) 477–482
25. Boutilier, C., Brafman, R.I., Domshlak, C., Hoos, H.H., Poole, D.: CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. J. Artif. Intell. Res. **21** (2004) 135–191
26. Sandholm, T.: Computing in mechanism design. In: New Palgrave Dictionary of Economics. (2008)

Match'n'Date: Semantic Matchmaking for Mobile Dating in P2P Environments

Michele Ruta, Tommaso Di Noia, Eugenio Di Sciascio, and Floriano Scioscia

Politecnico di Bari
via Re David 200, I-70125
Bari, ITALY
email: {m.ruta, t.dinoia, disciascio, f.scioscia}@poliba.it

Abstract. In a generic semantic-based matchmaking process, given a request, it is desirable to obtain a ranked list of compatible services/ resources/ profiles in order of relevance. Furthermore, a match explanation can provide useful information to modify or refine the original request in a principled way. Though the feasibility of this approach has been proved with fixed reasoning engines, it is a challenging subject to perform inference tasks on handheld devices. Here we propose abduction and contraction algorithms in Description Logics specifically devised for applications in mobile environments. A simple interaction paradigm based on Bluetooth protocol stack has also been implemented and tested in a mobile *dating* case study.

1 Introduction

We propose a novel discovery framework whose concrete implementation has been carried out in a mobile *dating* case study even if it is cross-applicable in all discovery scenarios. Knowledge Representation techniques and approaches have been shaped to be effectively suitable in volatile ubiquitous computing contexts. In particular, here we adapt abduction and contraction algorithms used in [4] in order to allow their exploitation in resource-constrained contexts. Building on previous work that enhanced the discovery possibilities offered by standard code-based matching procedures with semantic-based capabilities [15], here we devise a further evolution of matchmaking algorithms allowing to run the proposed reasoning services also on mobile devices. This framework and approach has been tested for profile matchmaking in a p2p environment.

Users equipped with a mobile device expose both their semantically annotated profile and preferences they would like to satisfy encountering another user. An exact match between requester preferences and offered profiles is surely the best possible result, but it is probably too rare to be realistic. It is more feasible to obtain a ranked list of available user profiles even if they do not completely fulfill the request. In the same way, when the user preferences and retrieved profiles are incompatible, it could be interesting to know what are the causes for the incongruence if user is willing to retract some constraints she originally imposed to reach a potential match. The proposed system exploits a revised version of non-monotonic inferences [6] (in particular abduction and contraction) to retrieve compatible profiles arranged in relevance order. A score is computed taking into account the semantic *affinity* between preferences expressed by the user and characteristics found in the available profiles. As explained hereafter, we selected a sublanguage deriving from OWL DL, $\mathcal{AL}(D)$, to model ontologies, preferences and profile annotations whereas the proposed system adopts an enhanced version of DIG 1.1 annotations.

The Bluetooth connectivity of handheld user's device is exploited to allow the data exchange aiming at extending the basic service discovery protocol with semantic capabilities. A "micro-layer" has been integrated within a J2ME[1] application level over the Bluetooth stack in order to enable a simple interchange of semantic annotations between a mobile host performing a query and another one exposing its characteristics. We adopt a simple piconet configuration without stable networked zone servers. Peers are equipped with a Bluetooth interface and they are at the same time able to address requests to other mobile clients as well as to receive and reply to external queries. Each device hosts a semantic facilitator to match on-board user preferences with profiles of users in the neighborhood.

The remaining of the paper is structured as follows: in the next section we motivate the proposed approach and present its background. In Section 3 and Section 4 we move on to the presentation of the theoretical framework. Relevant features of the dating application we implemented are outlined in Section 5 with the aid of a simple illustrative case study. Conclusion closes the paper.

2 Background

Exploiting standard relational databases for resource retrieval, the attributes of the offered and requested resources must exactly coincide to have a match. If requests and offers are simple names or strings, the only possible match would be identity, resulting in an all-or-nothing outcome of the retrieval process. Vague query answering, proposed by [12], was an initial effort to overcome the rigid constraints of relational databases, by attributing weights to several search variables.

Vector-based techniques taken by classical Information Retrieval can be used too, thus reverting the search for a resource matching a request to similarity between weighted vectors of stemmed terms, as proposed in the COINS matchmaker [10] or in LARKS [16]. The need to work in someway with approximation and ranking in DL-based approaches to matchmaking has also recently led to adopting fuzzy-DLs, as in Smart [1] or hybrid approaches, as in the OWLS-MX matchmaker [9].

A further approach structures resource descriptions as set of words. This formalization allows one to evaluate not only identity between sets, but also some

[1] Java 2 Micro Edition: http://java.sun.com/javame/index.jsp

interesting set-based relations between descriptions, such as inclusion, partial overlap, or cardinality of set difference. Anyway, modeling resource descriptions as set of words is too much sensitive to the employed words to be successfully used: the fixed terminology misses meaning that relates to the words. Such a problem can be solved giving to terms a logical and shared meaning through an ontology [8]. Nevertheless set-based approaches already have some properties we believe are fundamental in a resource matchmaking and retrieval process. If we are searching for a resource described through a set of words, we are also interested in sets including the one we search, because they completely fulfill the resource to retrieve. Moreover even if there are characteristics of the retrieved resource not elicited in the description of the searched one, an exact match is still possible because absent information has not to be considered negative. The two statements above may be summarized in the so called *Open World Assumption* (OWA). That is the absence of a characteristic in the description of a resource to be retrieved should not be interpreted as a constraint of absence. Instead it should be considered as a characteristic that could be either refined later or left open if it is irrelevant for the request.

3 Framework and Approach

After discussing the general Knowledge Representation principles that a logical approach to matchmaking may yield, we move on to the Description Logic (DL) setting we adopt[2]. Due to the lack of space, we refer the reader to [6, 4] for several examples and wider argumentation.

3.1 Description Logics and Semantic Matchmaking

From now on we assume that resource descriptions, both requested and offered, in the matchmaking are expressed in a language whose semantics can be mapped to a the Description Logic DL $\mathcal{AL}(D)$, for instance (a subset of) OWL DL or the more compact XML-based DIG language. Such a choice is motivated by several considerations. In [6] it has been proved that there exists a lower bound on the complexity of Concept Contraction, for all DLs that include $\mathcal{AL}$. $\mathcal{AL}(D)$ specifically requires limited computational capabilities to carry out the proposed reasoning services. A simple adaptation of the algorithms reported in the following will allow to report the Concept Contraction and Concept Abduction on an $\mathcal{EL}^{++}$ logic. Formulas (concepts) in $\mathcal{AL}(D)$, we use to represent user profiles and preferences, are built according to the following rules:

$$C, D \rightarrow CN \mid \neg CN \mid \exists R \mid \forall R.C \mid C \sqcap D \mid (\geq_k g) \mid (\leq_k g)$$

where CN represents a concept name. For what concerns the ontology (Terminological Box $\mathcal{T}$ in DL-words) we only allow relations between concept names in the form:

[2] We assume hereafter the reader be familiar with basics of Description Logics formalisms and reasoning [2].

$$CN_1 \sqsubseteq CN_2 \sqcap \ldots CN_n; \quad (1) \qquad CN_1 \equiv CN_2 \sqcap \ldots CN_n; \quad (2) \qquad CN_1 \sqsubseteq \neg CN_2 \sqcap \ldots \neg CN_n; \quad (3)$$

to respectively represent (1) subclass axioms; (2) equivalence axioms; (3) disjoint axioms. Furthermore, given a concept name CN we cannot have more than one equivalence axiom with CN on the left hand side (LHS) and if CN appears on the LHS of an equivalence axiom then it cannot appear on the LHS neither of a subclass axiom nor of a disjoint axiom. In order to avoid cycles within an ontology $\mathcal{T}$, we do not allow a concept name CN appears, directly or indirectly, both on the LHS and on the right hand side of an axiom [2]. Furthermore, for each concrete feature g we impose its range is always explicitely represented by its minimum value and its maximum value. We represent the range of g as:

$$range(g) = (g_{min}, g_{MAX})$$

DL-based systems usually provide two basic reasoning services for $\mathcal{T}$, namely (a) Satisfiability and (b) Subsumption in order to check (a) if a formula C is consistent w.r.t. the ontology $-\mathcal{T} \not\models C \sqsubseteq \bot-$ or (b) if a formula C is more specific or equivalent to a formula D $-\mathcal{T} \models C \sqsubseteq D$.

If we have a *Profile Description* `PD` and a *User Preference* `UP`, we can define at least five different match classes based on subsumption and satisfiability: exact match, subsumption (full) match, plug-in match, intersection (potential) match, disjoint (partial) match [13, 11, 4]. Given a preference, representing a request, and a set of profiles, representing the resources to be retrieved, we can classify the match relation between the preference and each profile according to the above classes. As argued in [4], there is a strong relation among these classes. In particular:

– given a partial match between `UP` and `PD`, solving a Concept Contraction Problem (CCP) [3] one can compute what has to be given up G and kept K in `UP` in order to have a potential match between K (a contracted version of `UP`) and `PD`. Hence, the result of a CCP is a pair $\langle G, K\rangle$ representing respectively elements in `UP` conflicting with `PD` and the (best) contracted `UP` compatible with `PD`.
– given a potential match between `UP` and `PD`, solving a Concept Abduction Problem (CAP) [7] one can compute what has to be hypothesized in `PD` in order to have a full match with `UP` (or its contracted version K). Hence, the result of a CAP is a concept H representing in some way what is *underspecified* in `PD` in order to completely satisfy a preference `UP`. Please note that we say *underspecified* instead of *missing*. This is because we are under a OWA.

Of course, both for Concept Contraction and Concept Abduction we have to define some minimality criteria both on G (give up as few things as possible) and on H (hypothesize as few things as possible). The interested reader may refer to [3, 5] for some minimality criteria in the framework of Description Logics.

An Algorithm for Concept Contraction in $\mathcal{AL}(D)$. An algorithm to solve CAPs for $\mathcal{ALN}$ has been proposed in [6] and it can be easily adapted to deal with $\mathcal{AL}(D)$. In this section we propose a new algorithm to compute a possible solution to CCPs in $\mathcal{AL}(D)$ given two concepts PD, UP both of them satisfiable w.r.t. an ontology $\mathcal{T}$. Before computing solutions to a CCP it is more convenient, from a computational perspective, to reduce both PD and UP to a common normal form. We use here well know techniques [2] to syntactically transform concepts and preserve their formal semantics with respect to $\mathcal{T}$. Given a concept C the normalization process is performed applying recursively the rewriting rules in Fig.1 to each occurrence of the element appearing in the LHS of the rule.

$$
\begin{array}{rl}
CN_1 \mapsto CN_1 \sqcap CN_2 \sqcap \ldots CN_n & \text{if } CN_1 \sqsubseteq CN_2 \sqcap \ldots CN_n \in \mathcal{T} \\
CN_1 \mapsto CN_2 \sqcap CN_3 \sqcap \ldots CN_n & \text{if } CN_1 \equiv CN_2 \sqcap \ldots CN_n \in \mathcal{T} \\
CN_1 \mapsto CN_1 \sqcap \neg CN_2 \sqcap \ldots \neg CN_n & \text{if } CN_1 \sqsubseteq \neg CN_2 \sqcap \ldots \neg CN_n \in \mathcal{T}
\end{array}
$$

$$
\begin{array}{r}
C \sqcap \bot \mapsto \bot; \\
A \sqcap \neg A \mapsto \bot; \\
(\geq_n g) \mapsto \bot \text{ if } n > g_{MAX}; \\
(\leq_m g) \mapsto \bot \text{ if } m < g_{min}; \\
(\geq_n g) \sqcap (\leq_m g) \mapsto \bot \text{ if } n > m; \\
\forall R.C_1 \sqcap \forall R.C_2 \mapsto \forall R.C_1 \sqcap C_2; \\
(\geq_n g) \sqcap (\geq_m g) \mapsto (\geq_n R) \quad \text{if } n > m; \\
(\leq_n g) \sqcap (\leq_m g) \mapsto (\leq_n g) \quad \text{if } n < m;
\end{array}
$$

Fig. 1. Normalization rules

Note that we refer to acyclic terminologies. In case of cyclic terminologies a simple blocking is enough to guarantee the termination of the normalization process. Given a concept $C \in \mathcal{AL}(D)$ and a taxonomy $\mathcal{T}$, we call $norm(C, \mathcal{T})$ the rewriting of C following the rules in Fig.1. If we consider $norm(C, \mathcal{T})$, it can be always represented as the conjunction $C^{CN} \sqcap C^{R} \sqcap C^{(D)}$, where:

C^{CN} is the conjunction of (negated) concept names;
C^{R} is the conjunction of terms involving roles;
$C^{(D)}$ is the conjunction of concrete domain restrictions, no more than two for every role (the maximum and the minimum for each concrete feature).

With $|norm(C, \mathcal{T})|$ we refer to the length of $norm(C, \mathcal{T})$ computed following Algorithm 1 reported in the follwoing.

At this point we have all the elements we need to formalize an algorithm to solve a CCP in $\mathcal{AL}(D)$ given two concepts PD and UP both satisfiable w.r.t. $\mathcal{T}$. In Algorithm $contract(\mathcal{AL}(D), norm(\mathtt{PD}, \mathcal{T}), norm(\mathtt{UP}, \mathcal{T}), \mathcal{T})$ starting from the normalized version of UP and PD we compute a solution $\langle G, K\rangle$ to the corresponding CCP and we also return *penalty*: a numerical value representing the worth associated to G. In other words, we compute the cost for a contraction of

Algorithm 1: How to compute the length of a concept C with respect to a taxonomy $\mathcal{T}$

Algorithm: $|norm(C, \mathcal{T})|$

Input: a $\mathcal{AL}(D)$ concept C and a taxonomy $\mathcal{T}$

Output: the length of $norm(C, \mathcal{T})$

$length := 0$;

if $norm(C, \mathcal{T}) = \bot$ **then**

  return 1;

end

foreach *(negated) concept name* $CN \in norm(C, \mathcal{T})^{CN}$ **do**

  $length := length + 1$;

end

foreach $(\geq_n g) \in norm(C, \mathcal{T})^{(D)}$ *or* $(\leq_m g) \in norm(C, \mathcal{T})^{(D)}$ **do**

  $length := length + 1$;

end

foreach $\exists R \in norm(C, \mathcal{T})^{R}$ **do**

  $length := length + 1$;

end

foreach $\forall R.D$ **do**

  $length := length + |norm(D, \mathcal{T})|$;

end

return *length*;

UP. We will use this value to evaluate the global utility function associated to a profile w.r.t. a set of preferences. Actually, the algorithm can be easily adapted to deal with different penalty functions [6].

Notice that, even though we impose both UP and PD to be satisfiable w.r.t. to $\mathcal{T}$, in lines 1-8 we also consider the case $\mathtt{UP} = \bot$. This is needed because of the recursive nature of the algorithm. In fact, in line 33 we have a recursive call involving the restrictions of a role R. In case this restriction is $\bot$, *i.e.*, $\forall R.\bot$ occurs UP, we have $\mathtt{UP} = \bot$ when we call $contract(\mathcal{AL}(D), norm(\mathtt{PD}, \mathcal{T}), \bot, \mathcal{T})$ in line 33. For the sake of readability of the algorithm let us pose $norm(\mathtt{PD}, \mathcal{T}) = \bar{\mathtt{PD}}$ and $norm(\mathtt{UP}, \mathcal{T}) = \bar{\mathtt{UP}}$.

Algorithm: $contract(\mathcal{AL}(D), \bar{\mathtt{PD}}, \bar{\mathtt{UP}}, \mathcal{T})$

1: $penalty := 0$;

2: **if** $\bar{\mathtt{UP}} = \bot$ **then**

3:   **if** $\bar{\mathtt{PD}} \neq \bot$ **then**

4:     **return** $(\langle \bot, \top \rangle, 1)$;

5:   **else**

6:     **return** $(\langle \bot, \top \rangle, 0)$;

7:   **end if**

8: **else**

9:   $G := \top$;

10:   $K := \top \sqcap \bar{\mathtt{UP}}$;

if $\overline{\mathtt{PD}} = \bot$ **then**
    return $(\langle \overline{\mathtt{UP}}, \top \rangle, |\overline{\mathtt{UP}}|)$;
end if
for each (negated) concept name $CN \in K^{CN}$ **do**
    for each concept name $CN' \in norm(CN, \mathcal{T})^{CN}$ **do**
        if there exists CN'' in $\overline{\mathtt{PD}}^{CN}$ such that $CN'' = \neg CN'$ **then**
            $G := G \sqcap CN$;
            remove CN from K^{CN};
            $penalty := penalty + 1$;
        end if
    end for
end for
for each concept $\exists R \in K^{R}$ **do**
    if there exists $\forall R.\bot \in \overline{\mathtt{PD}}^{R}$ **then**
        $G := G \sqcap \exists R$;
        remove $\exists R$ from K^{CN};
        $penalty := penalty + 1$;
    end if
end for
for each concept $\forall R.E$ in K^{R} **do**
    if either there exists $\exists R \in K^{R}$ or there exists $\exists R \in \overline{\mathtt{PD}}^{R}$ **then**
        for each concept $\forall R.F$ in $\overline{\mathtt{PD}}^{R}$ **do**
            $(\langle G', K' \rangle, penalty') := contract(\mathcal{AL}(D), E, F, \mathcal{T})$;
            $G := G \sqcap \forall R.G'$;
            replace $\forall R.E$ in K with $\forall R.K'$;
            $penalty := penalty + penalty'$;
        end for
    end if
end for
for each concept $(\geq_x g)$ in K **do**
    if there exists $(\leq_y g)$ in $\overline{\mathtt{PD}}$ and $y < x$ **then**
        replace $(\geq_x g)$ with $(\geq_y g)$;
        $G := G \sqcap (\geq_x g)$;
        $penalty := penalty + \frac{x-y}{x}$;
    end if
end for
for each concept $(\leq_x g)$ in K **do**
    if exists $(\geq_y g)$ in $\overline{\mathtt{PD}}$ and $y > x$ **then**
        replace $(\leq_x g)$ with $(\leq_y g)$;
        $G := G \sqcap (\geq_x g)$;
        $penalty := penalty + 1 + \frac{y-x}{x}$;
    end if
end for
end if
return $(\langle G, K \rangle, penalty)$;

4 Dealing with User Preferences

In real dating scenarios it is quite rare to find exactly the profile we are looking for. Often we have to reformulate one or more preferences and to hypothesize some characteristics not specified in the profiles we found. Based on this reformulate/hypothesize process we usually assign a relevance score to the profile representing how good our preferences have been satisfied.

In such a matchmaking process, a user request, can be split often into two separate parts: **strict** requirements and **preferences** [14]. **Strict** requirements represent what, in the request, has to be strictly matched by the retrieved profile description. **Preferences** can be seen as soft user requirements. In other words, the user will accept even a profile whose description does not represent exactly what the user prefers. Usually, a weight is associated to each preference in order to represent its worth (absolute or relative to the other preferences). Hence, for a user preference $\mathtt{UP}$ we distinguish between a concept $\mathtt{UP}_S$ representing strict requirements and a set of weighted concepts $\langle \mathtt{UP}, v \rangle$ where $\mathtt{UP}$ is a DL concept and v is a numerical value representing the preference worth. It should be clear that a matchmaking process has not to be performed w.r.t. $\mathtt{UP}_S$. It represents what the user is not willing to risk on at all. He does not want to hypothesize nothing on it. An approximate solution would not be significant for $\mathtt{UP}_S$. Actually, performing a matchmaking process between preferences and a profile description $\mathtt{PD}$ makes more sense. After all, preferences represent what the user *would like* to be satisfied by $\mathtt{PD}$. Hence, even though a preference is satisfied *with a certain degree* (not necessarily completely) the user will be satisfied *with a certain degree* as well.

Given an ontology $\mathcal{T}$, a profile description $\mathtt{PD}$, a strict requirement $\mathtt{UP}_S$ and a set of preferences $\mathcal{P} = \{\langle \mathtt{UP}_i, v_i \rangle\}$ we compute a global ranking penalty using Algorithm 2. Here we assign a *penalty* $\neq +\infty$ to profiles whose description fully satisfies user strict requirements. We also introduce a penalty threshold ϑ. If the global penalty is higher than ϑ then we discard the selected profile setting *penalty* $:= +\infty$ (line 14). Once we have a profile description such that $\mathcal{T} \models \mathtt{PD} \sqsubseteq \mathtt{UP}_S$, then we compute how much it satisfies user preferences. For each preference we take into account both a numerical evaluation of the characteristics to be given up with $penalty^c$ and a numerical evaluation of those characteristics to be hypothesized in $penalty^a$. The function *abduce* called in line 7 and line 10 is a combination of the algorithms (slightly modified to be used with $\mathcal{AL}(D)$) presented in [6] to compute and rank solutions to CAPs. We do not report here the algorithms for the sake of brevity. In line 12 of Algorithm 2 we combine $penalty^a$ and $penalty^b$ using two parameters h, g representing the worth associated respectively to $penalty^a$ and $penalty^b$.

The value of *penalty* can be easily converted to an affinity value using the following simple transformation:

$$affinity = 1 - \frac{penalty}{|norm(\mathtt{UP}, \mathcal{T})|}$$

Algorithm 2: Algorithm for preference-based semantic retrieval

Algorithm: $preference\_retrieve(\texttt{PD}, \texttt{UP}_S, \mathcal{P}, \mathcal{T}, t)$
penalty $= 0$;
if $\mathcal{T} \models PD \sqsubseteq UP_S$ **then**
  foreach $\langle UP_i, v_i \rangle \in \mathcal{P}$ **do**
    if $\mathcal{T} \models UP_i \sqcap PD \sqsubseteq \bot$ **then**
      $(\langle G, K \rangle, penalty^c) := contract(\mathcal{AL}(D), \texttt{PD}, \texttt{UP}_i, \mathcal{T})$;
      $(H, penalty^a) := abduce(\mathcal{AL}(D), \texttt{PD}, K, \mathcal{T})$;
    else
      $penalty^c := 0$;
      $(H, penalty^a) := abduce(\mathcal{AL}(D), \texttt{PD}, \texttt{UP}_i, \mathcal{T})$;
    end
    $penalty := penalty + v_i \cdot (h \cdot penalty^a + g \cdot penalty^c)$;
  end
  if $penalty > t$ **then**
    $penalty := +\infty$;
  end
  return $(penalty, \langle G, K \rangle, H)$;
end
return $(+\infty, \langle UP, \top \rangle, \bot)$;

5 Case Study: `Match'n'Date`

The mobile dating application `Match'n'Date` has been developed from scratch as a case study for the proposed matchmaking framework and algorithms. The goal is to facilitate acquaintance among people in a given environment. The proposed application is a pure peer-to-peer ubiquitous computing tool, based only on Bluetooth wireless ad-hoc networking. The core is a mobile matchmaker implementing reasoning algorithms for Concept Abduction and Concept Contraction. Note that since Concept Abduction extends Subsumption and Concept Contraction extends Satisfiability [6], the reasoner is also able to perform both consistency and subsumption checks. Each user stores her personal *profile* `PD` and a set of preference $\mathcal{P}$ on her device. They refer to a common domain ontology, which models people's physical appearance and personal interests[3].

A typical use case follows the protocol steps reported hereafter (also illustrated in Fig. 2). We refer to the device of the user looking for a profile as α and to the device hosting a discovered profile as β.

1. The user starts `Match'n'Date` on her mobile device (α). It looks for other devices in the Bluetooth radio range.
2. For each found device β, α checks if `Match'n'Date` is currently running and waiting for a connection.

[3] Due to lack of space, the reference ontology is not reported here.

3. If `Match'n'Date` is running on β, then α asks β to send the profile corresponding to her user. So α sends its profile to β. Profile exchange is performed via the Bluetooth OBEX (OBject EXchange) feature[4].
4. Both α and β run Algorithm $preference\_retrieve$ presented in Section 4 and compute their penalty values. β computes $penalty_{\alpha,\beta}$ while α computes $penalty_{\beta,\alpha}$. If $penalty_{\alpha,\beta} = +\infty$, then α sends a `HALT` message to β. Similarly β sends a `HALT` message to α in case $penalty_{\beta,\alpha} = +\infty$. In both cases the interaction between α and β ends.
5. If no `HALT` messages have been sent, then α sends an invitation to β to start a chat session (over Bluetooth).
6. Now β may visualize the profile sent by α. It may check the $affinity_{\alpha,\beta}$ value (see Fig.5) and it may ask for an explanation of the score looking at the values of $\langle G, K\rangle$ and H returned by $preference\_retrieve$.
7. β may accept or decline the invitation from α.

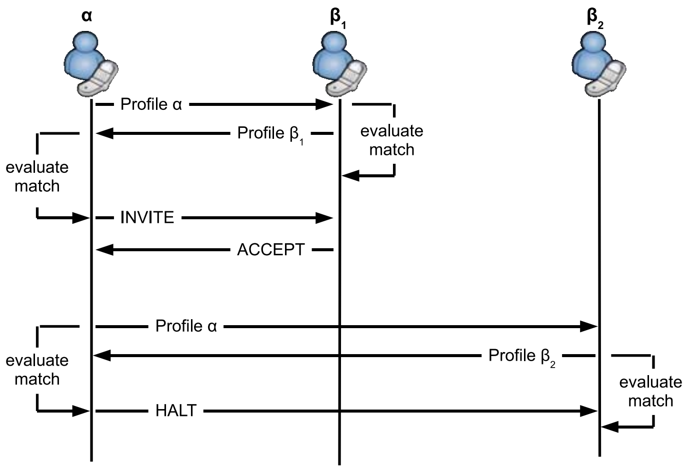


Fig. 2. A typical interaction between `Match'n'Date` devices

5.1 Running Example

Albert has been invited to a party by his room mate Joe, but he is getting quite bored. Joe is spending all the time with his girlfriend and Albert does not know anyone and he cannot find interesting conversation topics with other people. He would like to find a nice and not engaged girl to talk to. After all, Albert does

[4] As the system is at a prototypical state, profiles are now pre-loaded into the handheld. We are developing an intuitive GUI to manage the profile insertion.

Fig. 3. Main application form

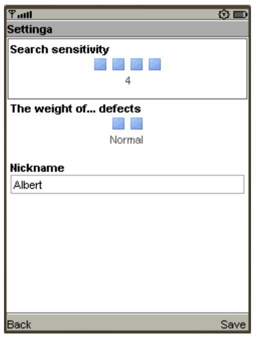


Fig. 4. Settings form

not want to spend all the evening talking with her boyfriend. He would like a woman between 21 and 32 years old and between 160 and 180 cm high, who likes painting and –very important– has not black hair. His former girlfriend had black hair. Currently, he is a little bit biased against black hair girls. So he launches `Match'n'Date` on his mobile phone. The main menu is shown (as in Fig. 3). Albert selects `Search` and `Match'n'Date` searches for other compatible devices in its Bluetooth radio range. Fingers crossed.

A remote device running `Match'n'Date` is found. It belongs to Barbara, who is getting bored too. The party is full of geeks. The most interesting and hot topics tonight seem to be the very last unstable release of the Linux kernel. Luckily she has `Match'n'Date` running on her mobile phone. Albert's device retrieves Barbara's profile and sends his profile to Barbara. The matchmaking process starts.

Hereafter we report the Albert's preferences in logic formalism. Using the graphical interface presented in Fig.4, Albert is able to set the value of the threshold t and the values for h and g used in line 12 of Algorithm 2. In the current implementation of `Match'n'Date` we use a single parameter and always assume $h = g$.

$\mathtt{UP}_S^{Albert}$: $\exists hasMaritalStatus \sqcap \forall hasMaritalStatus.Free$

$\mathtt{UP}_1^{Albert}$: $\langle (\geq_{age} 21) \sqcap (\leq_{age} 32) \sqcap (\geq_{height} 160) \sqcap (\leq_{height} 180), 0.3 \rangle$
$\mathtt{UP}_2^{Albert}$: $\langle \exists hasHobby \sqcap \forall hasHobby.Painting, 0.2 \rangle$
$\mathtt{UP}_3^{Albert}$: $\langle \exists hasHairColor \sqcap \forall hasHairColor.\neg Black, 0.5 \rangle$

Barbara is 28 years old and 172 cm high. She has red hair and currently she is not engaged. She likes art and she does not like swimming. She usually listens to pop-rock music and she watches romantic movies but not science fiction ones.

$\mathtt{PD}^{Barbara}$: $(\geq_{age} 28) \sqcap (\leq_{age} 28) \sqcap (\geq_{height} 172) \sqcap (\leq_{height} 172) \sqcap \exists hasHairColor$
$\sqcap \forall hasHairColor.Red \sqcap \exists hasMaritalStatus \sqcap \forall hasMaritalStatus.Free$
$\sqcap \exists hasHobby \sqcap \forall hasHobby.Art$
$\sqcap \exists hasSportPassion \sqcap \forall hasSportPassion.\neg Swimming$

Fig. 5. Matchmaking score form

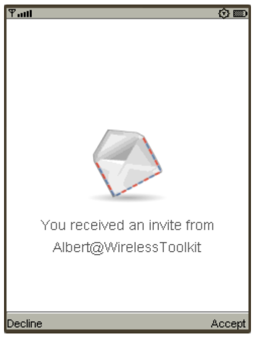


Fig. 6. Invite notification form

$$\sqcap \exists favoriteMusicGenre \sqcap \forall favoriteMusicGenre.Pop - Rock$$
$$\sqcap \exists favoriteMovieGenre \sqcap \forall favoriteMovieGenre.(Romantic \sqcap \neg Sci - Fi)$$

Albert is satisfied with the match outcome and wishes to invite Barbara to a chat. The dating application allows the user to contact the remote device for a chat session.

A simple text-based protocol was developed on top of Bluetooth OBEX for this purpose. Upon reception of an invite from α, β displays a notification to Barbara (see Fig. 6), who can either accept or decline the invitation. If β accepts, the chat session starts.

5.2 Experimental Results

One of the main issues in adapting Semantic Web technologies to mobile scenarios is to cope with computational costs. Matchmaking tasks usually need a heavy use of computational resources. This is the most significant reason why we developed our framework limiting the full expressiveness of OWL DL so using its $\mathcal{AL}(D)$ subset. Note that the reasoning algorithms we propose can be executed in polynomial time and they do not need highly optimized data structures.

In what follows we report some performance evaluation tests. In Fig.7, the time (in milliseconds) needed to calculate the affinity value for 100 pairs Preference-Profile randomly generated is shown. The simulation have been conducted exploiting the *Sun Java (TM) Wireless Toolkit 2.5.2 for CLDC*[5] allowing to emulate Virtual Machines (VMs) with different speeds (ranging from 100 to 1000 *bytecode/ms*). In order to cope with limited computational capabilities and reduced memory availability of handhelds, we fixed the speed of VM to 100 bytecode/ms as reference value for our simulations. In the Fig.8, the time (in milliseconds) needed for Concept Contraction –varying the number of concepts and restrictions in each list of preferences– is reported. Finally, Fig.9 shows the time (in milliseconds) needed for Concept Abduction w.r.t. the number of concepts and restrictions in the component *to keep* –K– of each list of preferences.

[5] http://java.sun.com/products/sjwtoolkit/

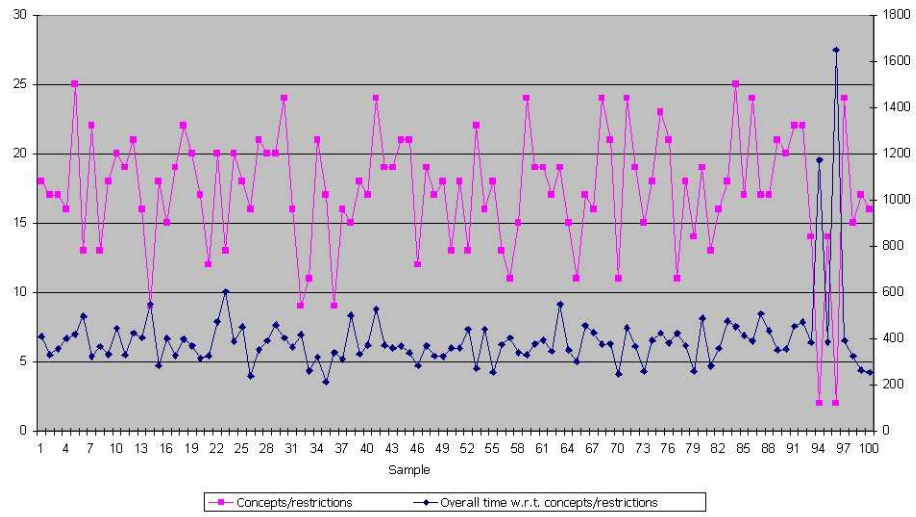


Fig. 7. Overall calculation time w.r.t. concepts and restrictions

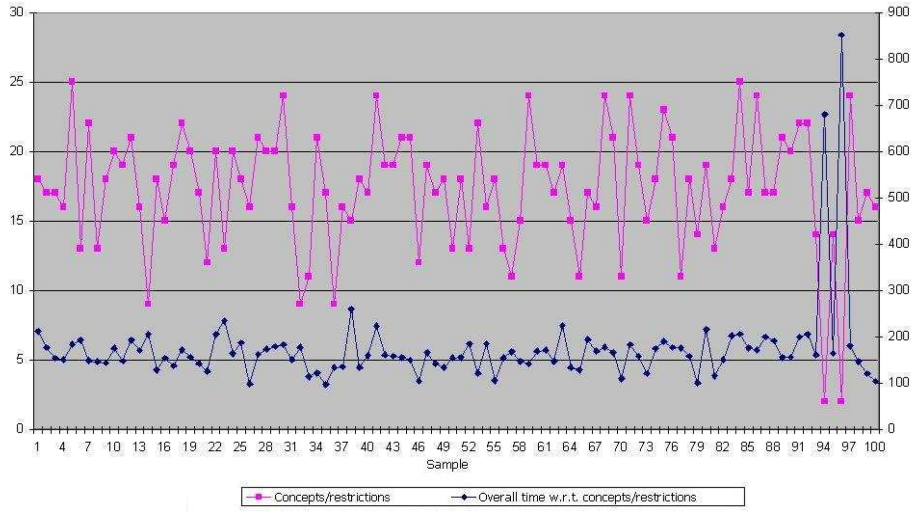


Fig. 8. Execution time for Concept Contraction

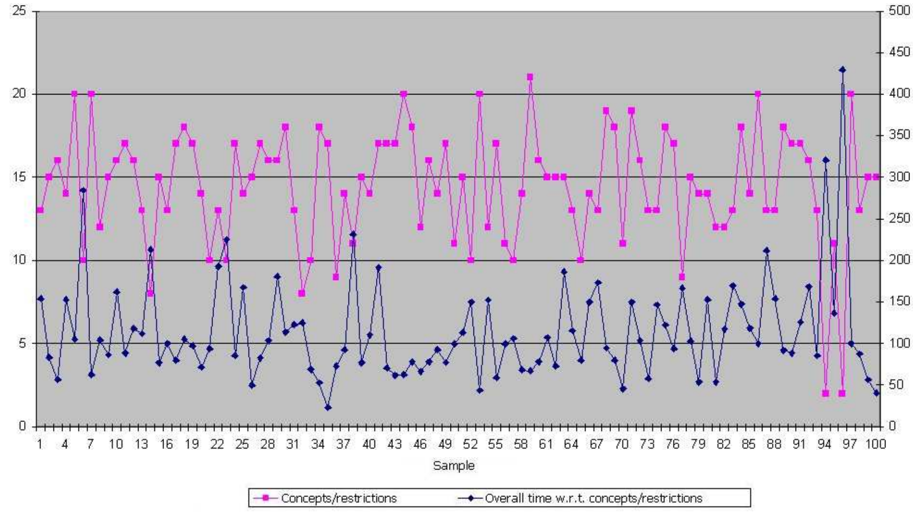


Fig. 9. Execution time for Concept Abduction

6 Conclusion

We have proposed a novel discovery framework for mobile ad-hoc contexts without stable and fixed network infrastructures. Abduction and contraction algorithms presented in [6] have been adapted to allow an exploitation in wireless and p2p scenarios. The proposed approach has been validated in a dating case study where users –equipped with a Bluetooth device– search for semantically annotated profiles compatible with their preferences (also expressed by means of a logic annotation). Framework and approach are general purpose as they are fully re-usable in different contexts and applications.

Future work is aimed at enhancing the expressiveness of the managed logic attempting to remove some constraint actually imposed (as for example the possibility to use the ∃ construct for profile definitions). We are currently working on a thorough evaluation of the approach basically measuring the response times of the system in different use cases and with different hardware and network configurations.

Acknowledgments

The authors wish to thank Nicola Caragnano for fruitful discussions and for the implementation of `Match'n'Date`. The authors acknowledge partial support of Apulia Region Strategic Project PS_121 and PS_092.

References

1. S. Agarwal and S. Lamparter. smart - a semantic matchmaking portal for electronic markets. In *Proceedings of the 7th International IEEE Conference on E-Commerce Technology 2005*, 2005.
2. F. Baader, D. Calvanese, D. Mc Guinness, D. Nardi, and P. Patel-Schneider. *The Description Logic Handbook*. Cambridge University Press, 2002.
3. S. Colucci, T. Di Noia, E. Di Sciascio, F.M. Donini, and M. Mongiello. Concept Abduction and Contraction in Description Logics. In *Proceedings of the 16th International Workshop on Description Logics (DL'03)*, volume 81 of *CEUR Workshop Proceedings*, September 2003.
4. S. Colucci, T. Di Noia, A. Pinto, A. Ragone, M. Ruta, and E. Tinelli. A non-monotonic approach to semantic matchmaking and request refinement in e-marketplaces. *International Journal of Electronic Commerce*, 12(2), 2007.
5. T. Di Noia, E. Di Sciascio, and F.M. Donini. Extending Semantic-Based Matchmaking via Concept Abduction and Contraction. In *Proceedings of the 14th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2004)*, pages 307–320. 2004.
6. T. Di Noia, E. Di Sciascio, and F.M. Donini. Semantic matchmaking as non-monotonic reasoning: A description logic approach. *Journal of Artificial Intelligence Research*, 29:269–307, 2007.
7. T. Di Noia, E. Di Sciascio, F.M. Donini, and M. Mongiello. Abductive matchmaking using description logics. In *IJCAI 2003*, pages 337–342, Acapulco, Messico, August 9–15 2003. MK.
8. D. Fensel, F. van Harmelen, I. Horrocks, D. McGuinness, and P. F. Patel-Schneider. OIL: An Ontology Infrastructure for the Semantic Web. *IEEE Intelligent Systems*, 16(2):38–45, 2001.
9. M. Klusch, B. Fries, and K. Sycara. Automated semantic web service discovery with owls-mx. In *In AAMAS 2006*, pages 915–922. ACM Press, 2006.
10. D. Kuokka and L. Harada. Integrating Information Via Matchmaking. 6:261–279, 1996.
11. R. Lara, M.A. Corella, and P. Castells. A flexible model for service discovery on the web. *International Journal of Electronic Commerce – Special Issue on Semantic Matchmaking and Resource Retrieval*, 12(2):11–41, 2007.
12. A. Motro. VAGUE: A User Interface to Relational Databases that Permits Vague Queries. *ACM Transactions on Office Information Systems*, 6(3):187–214, 1988.
13. M. Paolucci, T. Kawamura, T. Payne, and K. Sycara. Semantic matching of web services capabilities. In *Proceedings of the First International Semantic Web Conference (ISWC-02)*, pages 333–347. Springer-Verlag, 2002.
14. A. Ragone, T. Di Noia, E. Di Sciascio, and F.M. Donini. Logic-based automated multi-issue bilateral negotiation in peer-to-peer e-marketplaces. *Autonomous Agents and Multi-Agent Systems Journal*, 16(3):249–270, 2008.
15. M. Ruta, T. Di Noia, E. Di Sciascio, and F.M. Donini. Semantic based collaborative p2p in ubiquitous computing. *Web Intelligence and Agent Systems*, 5(4):375–391, 2007.
16. K. Sycara, S. Widoff, M. Klusch, and J. Lu. LARKS: Dynamic Matchmaking Among Heterogeneous Software Agents in Cyberspace. *Autonomous agents and multi-agent systems*, 5:173–203, 2002.

Look Ma, No Hands: Supporting the semantic discovery of services without ontologies

George A. Vouros, Fragkiskos Dimitrokallis, Konstantinos Kotis

AI-Lab, University of the Aegean, Karlovassi, Samos, Greece
{georgev, icsdm06002, kotis}@aegean.gr

Abstract. The work reported in this article aims to the discovery of WSDL specifications that are assessed to match to specific data requirements[1]: Going beyond the syntactic level, we aim at exploiting the human-intended semantics of WSDL specifications At the core of the proposed method lies the Latent Semantic Indexing (LSI) method, which automatically maps data requirements specified in a query to part elements of WSDL input and output messages. We study extensively the performance of the proposed method for different types of experiments' configurations. Experiments have been performed over an extended number of services for various domains, with very encouraging results.

Keywords: Latent Semantic Analysis, Semantic discovery of services

1 Introduction

The infrastructure for Web services is based mostly on standards such as UDDI [4, 16], SOAP [5] and WSDL [17]. WSDL (Web Service Description Language) is an XML-based language for the specification of web services. WSDL is mainly focusing on operational and syntactic details regarding the implementation and execution of Web services. The lack of explicit semantics in WSDL makes services' specifications insufficient to satisfy the requirements for effective Web service 'manipulation' (e.g. discovery, composition, invocation etc.) tasks, forcing the relevant mechanisms to be based on keyword matches. While we may relate different types of semantics to the web services (protocol semantics, execution semantics, non-functional semantics, e.g. security, QoS, and others), we can distinguish two widely recognized types: (a) Data semantics (introducing the semantic signature of services i.e. semantics of input/output messages of service operations), (b) Functional Semantics (function of operations and of the service itself). This paper focuses on data semantics, which is generally accepted to be one of the most critical aspects regarding services' semantic description.

The key technology to the "semantic discovery" (and thus to the "semantic matchmaking") of Web services is ontologies. The key idea here is that the use of

[1] This work is supported partially by the European Commission project Grid4All under grant number FP6, project IST-Grid4All-2006-034567

ontologies for the specification of web services shall support agents to exploit the semantics of services via logic-based inference mechanisms, gaining flexibility to the service manipulation tasks. Three main approaches have been proposed for bringing semantics to web services: OWL-S [1], WSDL-S [2] and WSMO [3]. However, given a) the use of WSDL for the specification of existent services, as well as the use of UDDI registries, b) the lack of commonly agreed domain ontologies, c) the lack of ontologies for specific domains, and d) the high cost of semantically annotating WSDL specifications by engineers, we have been motivated towards the approach proposed in this paper.

Focusing on data semantics, the aim of this paper is to support the semantic matchmaking of WSDL specifications, by exploiting the human-intended semantics of specifications, without the use of ontologies. Towards this target, our approach exploits the human-intended semantics of WSDL input/output messages and parts captured mainly via commentary and descriptive textual information. Such textual information regarding WSDL elements may be “fetched” from the service code, or they may be given by the service providers during service advertisement. This information is being exploited for the computation of the position that WSDL specifications can obtain in a semantic space, where the data semantics of WSDL specifications are expressed by means of a number of latent features. Given a query (this is considered to be a specification for the requested service’s data semantics) the objective of the proposed method is to represent this query using the computed latent features and find the registered WSDL specifications that are close enough – and thus similar – to the query, in the semantic space. Doing so, one may consider that these latent features are the concepts of an ontology, which serve as intermediates for the matching of WSDL specifications.

At this point we need to emphasize that although latent features are being used for expressing the data semantics of requested and advertised services, the lack of ontologies prohibit logic-based inferences, leading to the inability of the method to explicitly identify inferred subsumption relations between services’ specifications.

The rest of the paper is structured in the following way: Section 2 provides the related work, Section 3 provides a description of the problem and the background technology, and outlines the matchmaking method. Section 4 presents the experiments conducted towards evaluating the proposed method and finally, section 5 concludes the paper.

2 Related Work

Although several approaches introduce the semantic matchmaking of web services (e.g. [6, 7, 8, 9, 10 and 12]), most of them require the use of a *shared* and *well-agreed* ontology: Although this constraint may be relaxed by exploiting ontology mapping techniques, these techniques need to be further enhanced towards their generic deployment. Furthermore, the difficulties mentioned in Section 1 regarding the semantic description of web services need to be tackled.

In this section we emphasize on the approaches that are closely related to the proposed approach, i.e. to approaches that exploit the content of WSDL specifications, and/or exploit textual information related to these specifications.

The OWLS-MX matchmaker [7] performs hybrid semantic matching that complements logic based reasoning with syntactic IR based similarity metrics for services specified in OWL-S. OWLS-MX aims to exploit the implicit semantics of any part of OWL-S service description by representing it as a weighted category-index term vector. Index terms are stemmed lexical items from a shared minimal vocabulary. Concerning the similarity metrics, authors study four different token-based string metrics: the cosine, the loss of information, the extended Jacquard and the Jensen-Shannon information divergence. In contrast to OWLS-MX we go one step "back": we deal with WSDL specifications of services signatures, rather than with their semantic counterparts in OWL-S, aiming to support the content-based discovery of such specifications to a full extend, rather than in combination to logic-based approaches. This is a rather hard problem given the scarceness of the information, the dependency of specifications on the developers' whim, and the small size of the textual descriptions/comments. In addition, we aim at retrieving specifications that match exactly to a query (rather than retrieving specifications that are merely "neighbors" to the query).

Dealing also with WSDL specifications, aiming to address the challenges involved in searching for web services, authors in [11] present Woogle, a web-service search engine. In addition to simple keyword searches, Woogle supports similarity search for web services. The key ingredient of the approach is a refinement of the agglomerative clustering of parameters' terms in the collection of web services into semantically meaningful concepts. By comparing the resulting concepts, this work reports good similarity measures. In contrast to this approach, rather than exploiting parameters' terms co-occurrences, we aim at exploiting the human-intended meaning of WSDL input/output messages' part elements, "describing" them using a number of latent features, and positioning them in a latent space. This approach, in combination to the exploitation of services' textual comments/descriptions, as our experiments show, proves to be very precise.

Another web-service search engine (combining folksonomies and Semantic Web Services technologies) [20] presents a method for semantic indexing and approximate retrieval of Web services. It relies on graph-based indexing in which connected services can be approximately composed, while graph distance represents service relevance. A query interface translates a user's query into a virtual semantic Web service, which in turn is matched against indexed services. The approach is based on the association of classes (ontology conceptualizations or folksonomy tags) with WSDL service specifications: Thus it can be considered only as complimentary to our approach.

In [12], close enough to the approach proposed in this paper, authors discuss a set of WSDL similarity-assessment methods that can be used in conjunction with the current UDDI API to support the service-discovery process. This method utilizes the textual descriptions of the services, the identifiers of WSDL descriptions and the structures of their operations, messages and types to assess the similarity of two WSDL specifications. Given only a textual description of the desired service (as a whole), this approach uses an information-retrieval method to identify and order the most similar service description files. This step assesses the similarity of the query (requested-service description) - extended to include semantically similar words according to WordNet [14] - with the available services. A (potentially partial) specification of the desired service behavior may further refine discovery by a structure-matching computation step, exploiting mainly the lexical similarity of the identifiers. Rather than relying on the lexical appearance of the identifiers, we aim towards expressing their "meaning" by means of latent features, independently of any external resource: Then, matches between service signatures are determined based on semantic matches of WSDL input/output messages and parameters.

3 Service Discovery with Latent Semantics

3.1 Problem specification

This paper deals with the following problem:

"*Given, (a) a repository R of WSDL specifications and (b) a specification of a query Q that specifies the signature of the requested service, provide those services in R that match with Q".*

As already pointed, we deal only with services' signatures specified in WSDL (i.e. with data semantics). More specifically:

- The signature of a service *s* specifies a set of input/output messages. These are denoted <*s,X,i*>, where *s* is the service id, *X* is the type of the message, *input* or *output*, and *i* is the id of the specific message.
- Each message <*s,X,i*> is associated with textual annotations: <*s,i,Y,text*>, where *s* and *i* are as above, *Y* is the type of annotation provided, *description* or *comment*, and *text* is the actual annotation text (possibly *null*). Each message may be associated with more than one annotation.
- Each message <*s,X,i*> has one or more parameters (or parts): <*s,X,i,name,data_type*>, where *s,X,i* are as above, *name* is the name of the parameter, and *data_type* specifies a (atomic or complex) data type.
- Each parameter is associated with textual annotations: <*s,i,name,Y,text*>, where *s*, *i, name*, *Y* and *text* are as specified above. Each parameter may be associated with more than one annotation.

The way WSDL specifications are associated with annotations is thoroughly explained in the following paragraphs.

Each WSDL specification in the registry includes a specification of the corresponding service signature, as specified above. A query *Q* is also a specification of the requested service signature using the above constructs.

The aim of the discovery mechanism is to find the service *s* in the registry that matches the query *Q*. Formally, a service *s* matches a query *Q* , iff

$$\begin{array}{c} \forall < s, input, i > \exists < Q, input, j >: match(< s, input, i >, < Q, input, j >) \\ \wedge \\ \forall < Q, output, i > \exists < s, output, j >: match(< Q, output, i >, < s, output, j >) \end{array} \quad \textbf{(1)}$$

Given the above formula, for every input parameter of the advertised service there must be a matching input parameter of the required service. Also, for each output parameter of the required service there must be a matching output parameter of the advertised service. We have to notice that alternative definitions, maybe allowing greater matching flexibility, may be provided.

The function *match* determines whether the corresponding parameters match, and provides their similarity degree. In the proposed approach the *match* function is being computed by means of the Latent Semantic Indexing method.

3.2 Background information: Latent Semantic Indexing

The Latent Semantic Indexing (LSI) is a vector space technique originally proposed for information retrieval and indexing [15]. It assumes that there is an underlying latent semantic space that it estimates by means of statistical techniques using an association $N \times M$ matrix of terms-documents. Latent Semantic Analysis (LSA) computes the arrangement of a k-dimensional semantic space to reflect the major associative patterns in the data. This is done by deriving a set of k uncorrelated indexing factors (latent features). As already pointed out in the introduction, these factors may be thought of as artificial concepts whose lexicalization is not important.

Given these factors, each term and document is represented by a vector of values, indicating its strength of association with each of these underlying concepts. In other words, the meaning of each term and document is expressed by k factor values, or equivalently, by the location of a vector in the k-space defined by the factors. Then, a document is the (weighted) sum of its component term vectors. The similarity between two documents is computed by means of the dot product between the corresponding representation vectors.

Concerning our problem, each document corresponds to each input/output message or a message part, and each term is a distinct word in any of these "documents" (as it will be explained, these *pseudo*-documents are being constructed by means of annotations and message/part names).

For the computation of the k factors LSI employs a two-mode factor analysis by decomposing the original association matrix into three other matrices of a very similar form. This is done by a process called "Singular Value Decomposition (SVD)". This results in a breakdown of the original term-document relationships into linearly independent factors. Some of these factors are not significant and are ignored. The resulting k factors specify the dimensionality of the semantic space. By virtue of dimension reduction from the N terms space to the k factors space, where $k < N$, terms that did not actually appear in a document may still end up close to the document, if this is consistent with the major patterns of association in the data.

When one searches an LSI-indexed database of documents, it provides a query (i.e. a pseudo-document), which is a list of terms. The similarity between the query and any document is computed by means of the dot product between the corresponding representation vectors. Doing so, LSI returns a ranked list of documents, according to their similarity to the query.

3.3 The Matchmaking Method

The proposed matchmaking method assumes as input a specification of the desired web service (i.e. the query *Q*) and a repository *R* of registered services' WSDL specifications. All these services are "accompanied" by annotation files that associate descriptive and commentary information to services' specifications. After applying the matchmaking method to the advertised web services, the output is a ranked list of services according to their semantic similarity to the query. Although in our experiments (for implementation convenience, only) queries are being specified in WSDL, the proposed approach does not necessarily require the use of WSDL for the syntax of the query: Given that the queries are transformed in "bags of words" (as required by LSI), the proposed approach can also be used with keyword-based or template-driven querying forms as in Web-services search engines (e.g. Opossum in http://dori.technion.ac.il/, SeekDa in http://seekda.com/).

The proposed matchmaking method combines multiple sources of evidence to determine similarity between the signatures of two web services. In particular it considers the similarity between input and output messages, and between their parameters (i.e. input parts and output parts, respectively).

More precisely our approach is divided into the following stages: a) the matching of input messages, b) the matching of output messages, c) the matching of input parts and d) the matching of output parts. The algorithm determines the matching of each of these elements individually. The results are linearly combined to a single similarity measure between WSDL specifications (Figure 2).

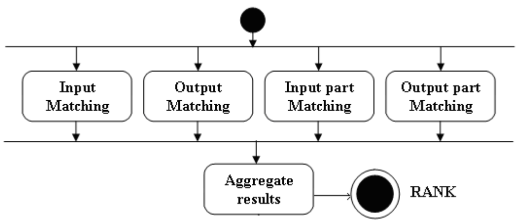


Fig. 2. The combination of stages for the computation of the overall similarity measure.

3.3.1 Annotation Files

The external annotation files (EAF) provide "slots" for the textual annotation of WSDL elements: "Comments" and "description" slots for the service itself, for each service's interface, operation and input/output messages, and for each of the corresponding parameters. It also provides support for specifying mappings between WSDL elements and ontologies. The EAF is an XML file aligned with the WSDL specification via XPATH expressions. Currently, comments and descriptions are considered to aggregate any type of textual information.

Although we plan to incorporate SAWSDL [18] into our framework, we do not commit to the use of SAWSDL at this stage, emphasizing mostly on the use of textual descriptions/comments for WSDL elements.

3.3.2 Input/Output Messages' similarity

Let us consider a message <*s,X,i*>, where *s* is the service *id*, *X* is the type of the message (input or output) and *i* is the id of the specific message; as well as its associated textual annotations <*s,i,Y,text*>, where *Y* is the type of annotation provided (description or comment). We construct a bag of words including all words comprising the message name and all words in the texts that annotate this specific message: This constitutes the pseudo-document corresponding to the message. Each word may be considered as a single term, or as a set of words (in case it is being constructed by concatenating a number of simple terms, e.g. getinputdate_of_arrival): We deal with both cases in our experiments. To improve the precision of our method we eliminate words with little substantive meaning, i.e. stop-words, and we consider only words that appear more than a certain number of times in the specific bag.

We compute the semantic similarity between each input (output) message of the query *Q* and the input (respectively, output) messages of all the web services in the registry by means of the LSI method: The method computes the semantic space by building the association matrix, associating words occurring in the web services' inputs (or outputs) and in their annotations, with the input (respectively, output) messages' pseudo-documents.

Similarly, each query to LSI is being constructed by means of the words in the corresponding messages of the query specification *Q*: In other words, for each message in *Q,* the proposed method constructs a separate LSI query. For each input/output message of the query *Q*, LSI returns a matrix of registered services' messages, together with a matching degree: Since each input/output message of *Q* can match only with an input/output message of each *s* in the registry, the proposed method keeps the highest ranked message per service *s* (i.e. the most similar one). For each registered service *s* whose input/output messages match with the messages of *Q*, a ranking (matching) degree d_s is computed by averaging the similarities of all input and output messages.

3.3.3 Input/Output part semantic similarity

Let us consider a parameter (message part) <*s,X,i,name,data_type*>, of a message *i* of a service *s*, where *name* is the name of the parameter, and *data_type* specifies its data type. This parameter is associated with textual annotations of the form <*s,i,name,Y,text*>, where *Y* denotes the type of annotation and *text* the annotation text. In this case, we construct a bag of words including all words in name and all words in each text annotating this specific parameter: This constitutes the pseudo-document corresponding to this parameter. As in messages, each word may be considered as a single term, or as a set of words (in case it is being constructed by concatenating a number of simple terms): We deal with both cases in our experiments. The pre-processing stage of eliminating stop-words, and words that do not occur many times in the specific bag, applies here as well.

We compute the semantic similarity between each input (output) parameter of the query *Q* and the input (respectively, output) parameters of all the web services in the registry by means of the LSI method: The method computes the semantic space by building the association matrix, associating words that occur in the web services' input (output) parameters and in their annotations, with the pseudo-documents corresponding to the parameters. Similarly, each query to LSI is being constructed by means of the words corresponding to an input/output parameter of the query *Q*.

a	0.5
b	0.7
c	0.1
d	0.001
e	0.5
f	0.99

(a)

b	0.7
d	0.001
f	0.99

(b)

a	0.8
b	0.05
c	0.2
d	0.4
e	0.999
f	0.2

(c)

a	0.8
d	0.4
e	0.999

(d)

a	0.75
b	0.2
c	0.995

(e)

Fig. 3. Ranking services based on input parts' similarities

For each input/output parameter of the query *Q*, LSI returns a matrix of registered services' parameters, together with a matching degree: Since each input/output parameter of *Q* can match only with an input/output parameter of each *s* in the registry, the proposed method keeps the highest ranked parameter per service *s.* For each registered service *s* whose input/output parameters match with the messages of *Q*, a ranking degree d_s is computed by averaging the matching degree of all input and output parameters.

Since the method of ranking registered services using input/output messages, as well as the method of using input/output parameters, is the same, let us exemplify the computations with a simple example concerning input parameters. Suppose we have the following advertised services: i) service s_1 with three input parts: a, b, c, ii) service s_2 with one input part: d, and iii) service s_3 with two input parts: e, f. The service request *Q* has two input parts: x, y. The method uses LSI to compute the similarity between the input part x of *Q* and all input parts of the advertised services, i.e. a, b, c, d, e and f. The results are shown in table (a) of Figure 3. The method keeps for each service only the input part that has the maximum degree of match. The result after this step is shown in table (b) of Figure 3. Then, LSI computes the similarity between the other input part *y* and all the input parts of the advertised services. The results are shown in table (c) of Figure 3. Again, the method keeps for each service only the input part that has the maximum degree of match. The result after this step is shown in table (d) of Figure 3. Finally, these two tables (b) and (d) are summed and the result is divided by the number of input parts of the request, as shown in table (e). This final matrix associates each advertised service with a similarity to the query *Q*, based on the computation of their input parts' similarities.

3.3.4 Input/Output messages' similarity by means of input/output parts similarity

In addition to the above described approach, we have been experimenting with an alternative method for the computation of input/output messages' similarities, so as to enrich the information consulted for the matching of messages: We construct the bag of words for each input/output message not only using the message names with their textual annotations, but adding also the associated input/output parts with their annotations. In particular, we identify the input/output messages of a web service operation together with the corresponding input/output parts of it. Then, the method computes the matching similarity of the advertised services with the query in the same way as we have described in the previous paragraphs.

3.3.5 Combining individual similarities

The result of the matching algorithm is a list of registered services matching to the query *Q*, with their rankings. These ranking degrees result from averaging the degrees computed by each of the individual stages described: a) the matching of input messages, b) the matching of output messages, c) the matching of input parts and d) the matching of output parts.

4 Experiments

To evaluate our approach, we have used service specifications from the OWL-S Service Retrieval Test Collection (OWLS-TC) version 2 [19] as well as an additional set of nine (9) Web services for Network Simulation (NS). From the OWLS-TC collection, we have translated a set of 70 services to WSDL, due to problems faced with the OWL-S-to-WSDL translation tool and due to services' description duplicates concerning the WSDL part elements. We have been experimented with 7 OWLS-TC domains and the additional NS domain. Table 1 summarizes information concerning the characteristics of the services in our repository *R,* i.e. the different domains of the services (column 1), the number of services for each domain (column 2), and the total number of WSDL input/output messages (column 3) and WSDL input/output distinct parts (column 4) for each domain.

Table 1. Information about the experimental domains and services

	Domain	# Services	# WSDL messages	# WSDL Parts
1	Weapon	3	6	7
2	Education	9	9	82
3	Economy	14	14	31
4	Travel	9	9	27
5	Portal	22	8	63
6	Books	8	22	19
7	Medical	5	5	39
8	Network Simulation	9	21	41
	Total	**79**	**94**	**309**

Concerning the WSDL specifications for domains 2 to 6, message part names are mainly composed by a single-word term capitalized and an underscore character as a prefix (e.g. _COUNTRY). For the domain 7, the messages part names are composed by multi-word terms, either separated with an underscore or with no separator, or using a combination of these (e.g. GetPatientMedicalRecords_AuthorizedMedicalRecords). We handle individual, distinct terms of multi-word terms separately, only in cases these are separated by an underscore separator, which is one of the most generic case considered.

WSDL specifications have been manually annotated by human annotators that have adequate knowledge of the related domains and services. They have been advised to carefully choose the annotations in order to indicate as close as possible the intended meaning of the input/output messages and of their parameters. Where possible, annotators have been advised to get feedback from xsd-schema complex types included in the WSDL specifications. Services in the “Network Simulation” domain have been annotated by their developers. More specifically, we can identify 3 different annotation cases that have been applied in the corpus:

- Annotations that are formed by “free text including terms from xsd-schema types and from the WSDL message part name”.
- Annotations that are formed by “free text including terms only from the WSDL message part name”.
- Annotations that are formed by “a single term”. In this case, annotators choose a single term without considering xsd-schema types or message’s part names. For instance, for the “wsdl:part name="Capital_City"”, the annotator has chosen the single-term-description “*<description> Capital </description>*”, which “captures” the intended meaning of the entity “Capital City”.

The result of this process is a set of annotations with terms belonging in one of the following three categories: a) terms from xsd-schema types, b) terms from WSDL message part names, or c) terms chosen by human annotators.

Table 2. Information concerning annotations per domain

Domain	**Annotation type**
Travel, Portal	Descriptions: category (b), Comments: category (a)
Books, Education	Descriptions: category (b), Comments: category (a)
Economy	Descriptions: category (b), Comments: category (a)
Weapon	Descriptions: category (b), Comments: category (a)
Medical	Descriptions: category(c), Comments: category (a)
Network Simulation	Descriptions: category (c), Comments: category (c)

The use of free text with terms from the xsd-schema types and/or the WSDL message part names, means that the annotator is allowed to form a natural language sentence using these terms: E.g. “*<description> The service requests accommodation using country information </description>*”. Although the use of free text may distract the matching method (due to the incorporation of noisy terms), the freedom that the approach gives to the annotator is important and realistic. In this example of annotation, the annotator has combined terms (e.g. “country”) from messages’ part names (category b). As another example, in the comment “*<comment> The country name is a string. A country is described with its capital, its currency, and its government </comment>*” the annotator has used terms (terms “capital”, “currency”, “government”) from the xsd-schema type that corresponds to the specific message part name (category a). Table 2 summarizes information concerning annotations per domain. It must be noticed that all annotations contain 5 to 7 “significant” terms; i.e. non stop-words that may drive the computation of elements’ intended mappings.

Given a query Q, we consider that the repository R includes one matching service. For evaluation purposes we measured the *precision* (p) of the method (i.e. the percentage of times that the method returns the correct service at the top of the ranked list), as well as the *Top-k precision* (p_k) of the method (i.e. the percentage of times that the correct service is among the top k in the ranked list). Specifically, we have measured p_5 and p_{10}.

The evaluation of the approach has been extensively conducted with a large number of variations of the queries, so as to test the robustness of the proposed approach, even in cases where annotations are misleading, or missing. According to these variations, the words that are being used for the construction of the queries for the input/output messages and parts vary significantly.

Specifically, words concerning the query Q may be fetched from:

- WSDL input/output message and part names without splitting them into single words (this is considered to be the default case and it is not denoted in the experiments).
- Splitting all words in the bag (this case is denoted by "split")
- WSDL input/output messages' annotations (this case is denoted by "in/out")
- WSDL input/output parts' annotations (this case is denoted by "inP/outP")
- WSDL input/output messages' annotations in conjunction with information from the corresponding input/output parts (this case is denoted by "inWP/outWP")

Combinations of the above cases give the different configurations of the method. For instance, the cases where the method considers only the matching of input/output parts, constructing queries using WSDL input and output parts' annotations together with part names are denoted as LSA-inP-outP. As another example, the cases where the method considers (a) the matching of input/output messages, constructing queries using words from the WSDL input and output messages' annotations in combination with words from input and output parts' annotations (i.e. the last case above), in combination with (b) the matching of messages parameters, constructing queries using WSDL input and output parts' annotations, together with (c) splitting all words, is denoted as LSA-inWP-outWP-inP-outP-split. Conclusively, as shown in the Figures 4, 5, 6, we have run experiments with eight (8) configurations of the proposed method.

For each case, we have considered alternatives for the filtering of words, based on the times (t) of words' appearance in the bag. Due to the very low numbers of words' appearances we distinguished two cases: (a) Filtering out words that appear exactly one time ($t>1$) and (b) including all words ($t>=1$).

First, we evaluated the proposed approach by placing each of the 79 WSDL specifications included in the repository R as a query. This may be considered to be the best case for our method since each query matches exactly with an advertised service (Figure 4). In addition, we evaluated the method using each of the 79 services in R as a query, but without considering their textual annotations. The results of these cases are also presented in Figure 4. As shown, annotations play a significant role towards increasing the precision of our method: This is particularly true for cases with $t>1$. However, when all words are included in the bag of words ($t>=1$), even if generally the results are better when considering annotations (even better from the cases where $t>1$), some configurations fail to achieve better results from the corresponding cases with no annotations: This is due to the incorporation of "noisy" words.

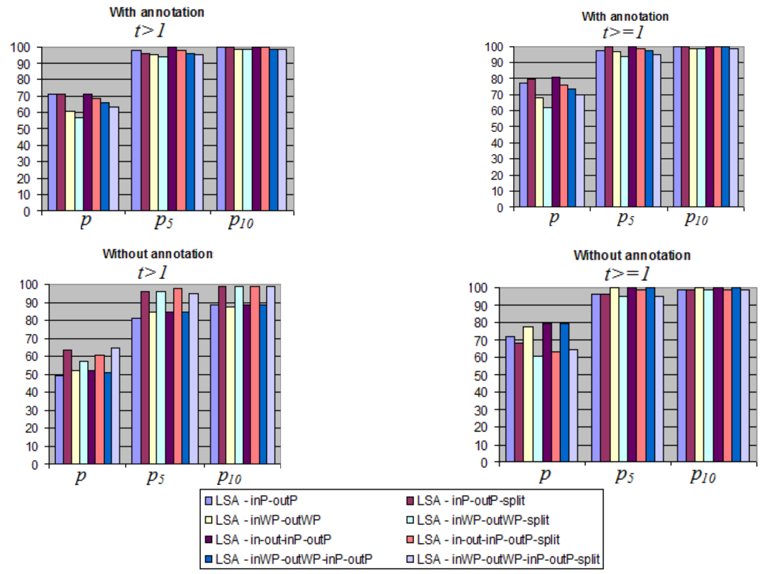


Fig. 4. Cases where Q is in R.

To test the robustness of the proposed method in the presence of noisy words and in cases where queries do not contain many "useful" terms, we have been experimenting with a variety of cases for the queries. Thus, we have conducted experiments by adding extra *new* words (let us call them noise-words) in the bag of words of Q. Noise-words are randomly chosen from the textual annotations of services included in the experimental corpus, given that these services belong to the same domain with the domain of the query service. We have been experimenting with one, three, and five noise-words per query. Figure 5 shows the results of these experiments, together with two more graphs presenting results of the proposed method when WSDL messages and part names are not included in the queries (and without adding noise-words). In addition to the above, we have conducted further experiments with noise-words: This time noise-words replace (one, three, or five) words included in the query. Noise-words are new words and being chosen as in the previous cases. Figure 6 presents the results of these additional experiments, in two different sets of cases: a) Replacement of words with noise-words in queries that include WSDL messages and part names, and b) replacement of words with noise-words in queries that do not include WSDL messages and part names.

In Figures 5 and 6, it can be shown that the precision of the method scales proportionally to the number of noise-words. The best results are achieved in cases where all words are taken into account (t>1). In these cases, when the method considers the input/output messages and their input/output parts separately, together with the part name (split or not) (LSA-in-out-inP-outP-{split}) it achieves the best results, even in cases with increased noise: It manages to present the correct service among the top 5 (respectively, top 10) ranked services with precision greater than

85% (respectively, 93%). It must be pointed that the addition of 5 noise-words with/without replacement, incorporates a major (if not radical) change in the specification of the queries.

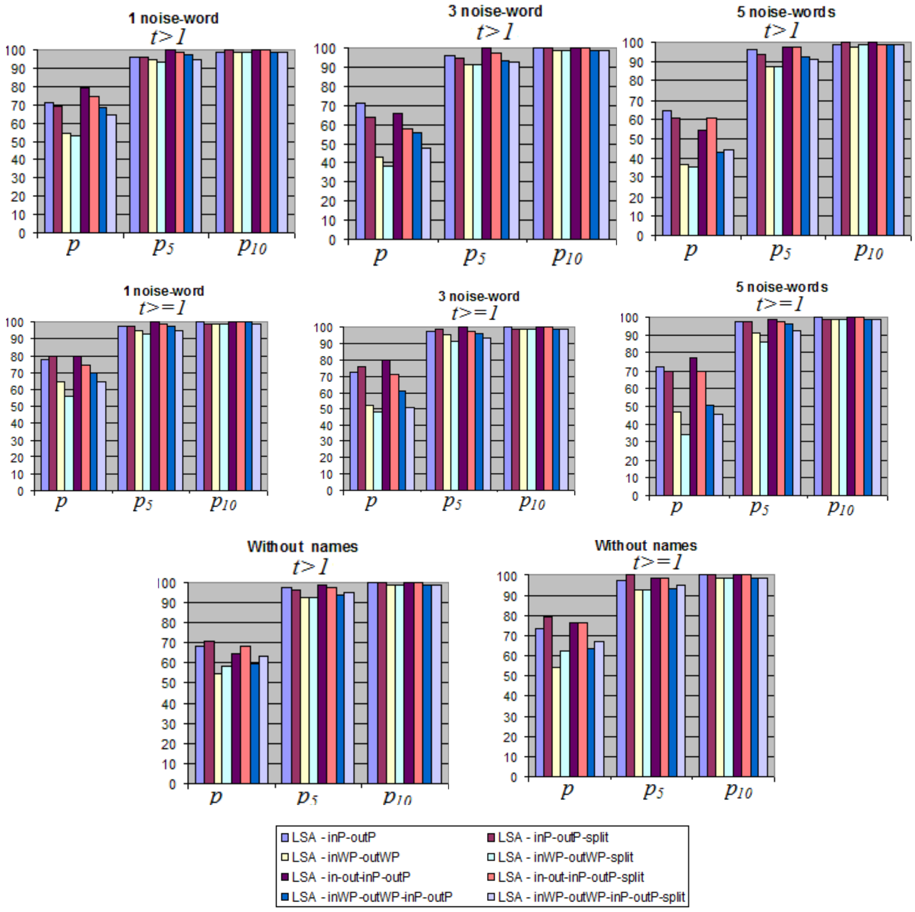


Fig. 5. Results of the proposed method a) adding noise-words in the queries, b) omitting WSDL messages and part names without adding noise-words

5. Concluding Remarks

The work reported in this article aims to the discovery of WSDL specifications that match specific data requirements by computing the intended semantics of part elements of input/output messages.

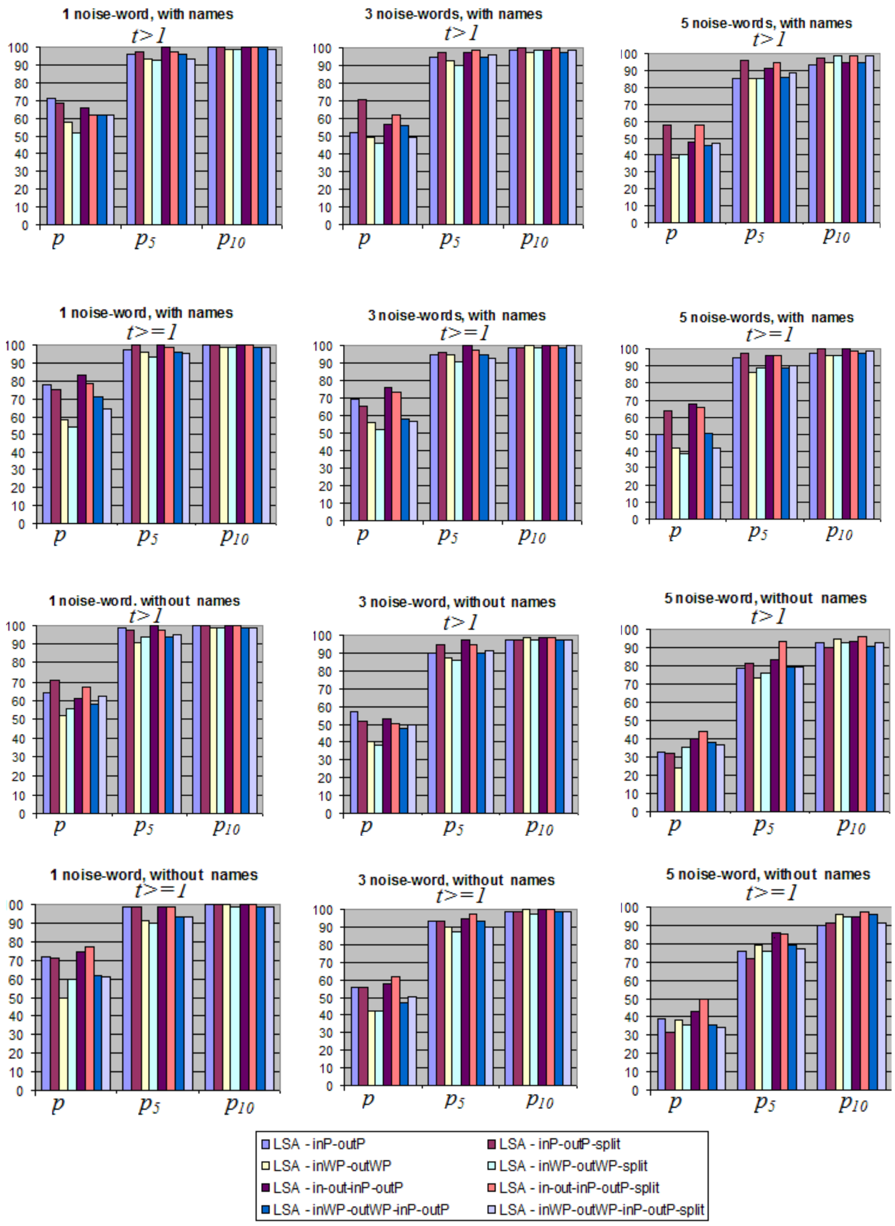


Fig. 6. Results of experiments with replacement of query words with noise-words

Going beyond the syntactic level, we aim at exploiting the human-intended semantics of WSDL specifications captured by means of comments and descriptions been associated with these elements. The basic constituent of the proposed method is the Latent Semantic Indexing (LSI) method, which maps data requirements specified in a WSDL query to part elements of WSDL input and output messages. Preliminary results for different types of experiments' configurations are very encouraging: This "basic" method manages to achieve quite high precision. Further work includes testing the method in a large repository of WSDL specifications, and studying extensively its dependency on the quality of annotations, also in combination with other methods.

References

1. M. Burstein, et al. OWL-S: Semantic Markup for Web Services. W3C Member Submission, Nov. 2004.
2. R. Akkiraju, et al. Web Service Semantics - WSDL-S. W3C Member Submission, Nov. 2005.
3. D. Roman, et al. WSMO - Web Service Modeling Ontology. In DERI Working Draft 14, vol. 1, pp. 77–106. DERI, IOS Press, 2005.
4. OASIS http://www.oasis-open.org/home/index.php
5. Simple Object Access Protocol (SOAP) http://www.w3.org/TR/2000/NOTE-SOAP-20000508
6. M. Paolucci, T. Kawamura, T. Payne, and K. Sycara. Semantic Matching of Web Service Capabilities. In Proc. of ISWC 2002, pp. 333–347. Springer Verlag, 2002.
7. M. Klusch, B. Fries, M. Khalid, and K. Sycara. "OWL-S: Hybrid OWL-S Service Matchmaking". In Proceedings of 1st Intl. AAAI Fall Symposium on Agents and the Semantic Web, 2005.
8. Fernandez, A. Polleres, and S. Ossowski. "Towards Fine-grained Service Matchmaking by Using Concept Similarity". In Proc. of SMR2 2007, vol. 243 of CEUR-WS, pp. 31–45, November 2007.
9. d' Amato, S. Staab, N. Fanizzi, and F. Esposito, "Efficient discovery of services specified in description logics languages", In Proc. of SMR2 2007, vol. 243 of CEUR-WS, .pp. 15-29, Busan, Korea, November 2007.
10. M. C. Jaeger, G. Rojec-Goldmann, G. Muhl, C. Liebetruth, and K. Geihs,"Ranked Matching for Service Descriptions using OWL-S", In Proc. of KiVS, p. 91-102 2005.
11. X. Dong, A. Halevy, J. Madhavan, E. Nemes, J. Zhang, "Similarity Search for Web Services", In Proc. of the 30th VLDB Conf., pp. 372-383 , Toronto, Canada, 2004,
12. D. Skoutas, A. Simitsis, T. Sellis, "A Ranking Mechanism for Semantic Web Service Discovery", in IEEE Congress on Services, pp. 41-48, Salt Lake City, UT, USA, 2007
13. Y. Wang and E. Stroulia, "Semantic Structure Matching for Assessing Web-Service Similarity", In Proceeding of ICSOC 2003, pp. 194-207, Trento, Italy, 2003.
14. Miller, G. (1995). WordNet: A lexical database for English. Communications of the ACM, 38(11) pp. 39-41
15. S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, R. Harshman. Indexing by Latent Semantic Analysis. Journal of the American Society of Information Science (1990), 41(6), 391-407
16. UDDI technical paper, http://www. uddi.org/pubs/Iru_UDDI_Technical_White_Paper.pdf
17. Web Services Description Language (WSDL) http://www.w3.org/TR/wsdl
18. Semantic Annotations for WSDL and XML Schema (SAWSDL) http://www.w3.org/TR/2007/ REC-sawsdl- 20070828/
19. OWLS-TC, http://projects.semwebcentral.org/projects/ owls-tc/
20. E. Toch, A. Gal, I. Reinhartz-Berger, and D. Dori. A Semantic Approach to Approximate Service Retrieval, to appear at ACM Transactions on Internet Technology, February 2008

Closing the Service Discovery Gap by Collaborative Tagging and Clustering Techniques

Alberto Fernandez[3], Conor Hayes[1], Nikos Loutas[1, 2], Vassilios Peristeras[1], Axel Polleres[1], Konstantinos Tarabanis[2]

[1]National University of Ireland, Galway
Digital Enterprise Research Institute
{firstname.lastname}@deri.org

[2]University of Macedonia, Thessaloniki, Greece
kat@uom.gr

[3]CETINIA, Universidad Rey Juan Carlos, Móstoles, Spain
alberto.fernandez@urjc.es

Abstract. Whereas the number of services that are provided online is growing rapidly, current service discovery approaches seem to have problems fulfilling their objectives. Existing approaches are hampered by the complexity of underlying semantic service models and by the fact that they try to impose a technical vocabulary to users. This leads to what we call the service discovery gap. In this paper we envision an approach that allows users to query or browse services using free text tags, thus providing an interface in terms of the users' vocabulary instead of the service's vocabulary. Unlike simple keyword search, we envision tag clouds associated with services themselves as semantic descriptions carrying collaborative knowledge about the service that can be clustered hierarchically, forming lightweight "ontologies". Besides tag-based discovery only describing the service on a global view, we envision refined tags and refined search/discovery in terms of the concepts that are common to all current semantic service description models, i.e. input, output, and operation. We argue that Service matching can be achieved, by applying tag-cloud-based service similarity on the one hand and by clustering services using case based indexing and retrieval techniques on the other hand.

Keywords: service discovery, tag, clustering

1 Introduction

The Service Oriented Architecture (SOA) paradigm [34, 35] has recently become the prevalent way of building enterprise Information Systems (IS). The main idea behind SOA is the ability to use, reuse and share services from different sources. Nowadays, Web Services (WS) are the prevailing paradigm for implementing SOA, supported by significant industry investment [20].

IBM's initial reference architecture for SOA [12] identified three basic players: service providers, service requestors and service brokers. This reference architecture supports four key functionalities - discovering, composing, publishing and invoking a service. A common use case is where the service provider publishes services using the service broker, while the service requestor uses the service broker to discover services.

Complementing the SOA/WS paradigm, the Semantic Web model [4] offers a means of enhancing the brokerage model using machine-readable annotations, which could be used by agents for automated discovery and composition, even at run-time. This led to the definition of various Semantic Web Service (SWS) models. The first SWS model proposed was DAML-S [1] followed by OWL-S [31], WSMO [40], SWSF [50], WSDL-S [1] and finally the SAWSDL [26] standard recommendation. Common to these service models is the separation of aspects to describe a service in terms of *inputs*, *outputs*, and *operations* (plus often real-world *preconditions* and service execution *effects*). To describe these aspects, SWS models rely on the existence of respective domain ontologies which can be referenced in actual service descriptions.

However, SWS efforts have struggled to achieve adoption [52], due to the complexity in providing meaningful service descriptions and the lack of pervasive domain ontologies for service descriptions. Despite the fact that all OWL-S, WSMO, and SWSF were submitted to W3C, these comprehensive frameworks have not become standards [51]. Instead, the more lightweight WSDL-S framework has made a significant contribution to the recently published SAWSDL standard by W3C and has led to other lightweight "versions" of SWS description frameworks such as SA-REST [45], or WSMO-Lite [27].

Another important obstacle to the success of SWS technologies is that all existing frameworks have the assumption that the service description (semantic or not) is a publishing task that will be handled by the service providers. As such, current frameworks do not capture information about the way, the reason or the context in which services are used and do not take into account how actual users perceive services.

In the present paper, we aim to promote a paradigm where services are semantically annotated both by service providers, who may use formal SWS descriptions as well as free text tags to describe their services, and by users, who will generally use free text tags to annotate services. Our approach is influenced by the way that content is annotated in Web 2.0 platforms.

The Web 2.0 paradigm can be viewed as a minimalistic, bottom-up approach to semantic annotation, emphasising on the ease-of-use and on the need for participation over formal correctness [33,32]. In the rapidly growing Web 2.0 realm, formal description frameworks to describe services or mash-ups, or repositories for storing service descriptions have not yet emerged. Services are generated and published in a completely decentralized and uncontrolled way. Web 2.0 service discovery is enabled by lightweight annotation (tagging) services provided by third parties, such as del.icio.us or digg.com, where the annotation is completely decoupled from the actual service provision.

In turn, attempts to reconnect the "anarchic" annotation/resource description practice of Web 2.0 to the Semantic Web world are already underway with efforts such as the Meaning of a Tag (MOAT) project [37], that help in assigning machine-readable meaning (namely URIs, possibly defined in an ontology) to tags.

This paper is a first attempt to mix existing SWS description and discovery models with the Web 2.0 tag- and user-centric collaborative solutions to resource discovery.

Our approach suggests that Web 2.0 descriptions provide surface level or user-centric descriptions of services, enabling users to draw upon the perceptions of other users to narrow the discovery search space. More formal service descriptions from the matching subset of services can then be presented to the users, enabling them to identify the best-matching service to their query context. As such, service descriptions should ideally cater for both formal semantic descriptions from one of the existing frameworks as well as tag-clouds produced by collaborative annotation.

Our proposal of a mixed service discovery model consists of two main ideas. Firstly, we encourage users to provide tags upon service usage, forming tag-clouds per service. These tag-clouds can be matched using standard similarity measures against user requests. As an ignition step and refinement we propose clustering techniques in order to first generate initial tag-clouds by clustering provider descriptions based on both formal descriptions and provider-based tag sets. Secondly, we hierarchically cluster existing service tag-clouds, in order to achieve lightweight, browsable service ontologies, represented by discriminating tags per cluster.

The first step of our approach addresses the cold start problem, i.e. even without user-provided tags, we generate synthetic tag-clouds from provider descriptions. We expect precision to gradually increase with uptake of the system and the addition of user tags. We believe that this approach could be a viable alternative for facilitating the discovery of services.

The remainder of this paper is organized as follows: Section 2 discusses previous work on service discovery and matchmaking, and introduces the notion of the service discovery gap. Section 3 discusses our approach in detail. Finally, Section 4 concludes the paper and presents our future directions.

2 The Service Discovery Gap

The area of service discovery and matchmaking has been a very active research area in recent years. Nonetheless a vast majority of approaches has focused on partially complex description frameworks defining matchmaking algorithms that rely on exact logical reasoning capabilities, see [13, 36, 28, 48] for approaches for OWL-S and its predecessor DAML-S, or [21, 46] for WSMO. In acknowledgement of the infeasibility of exact logical matchmaking by sheer complexity of the involved reasoning, Stollberg et al. [46] propose a stepwise refinement of the service discovery process. There keyword-based matching serves as pre-filtering, followed by abstract matching of high-level service capabilities, and exact logical matching being the last step. D'Amato et al. [8] propose a service retrieval method based on a conceptual clustering approach, where services are specified as description logic concepts.

Catering for the infeasibility of exact logical matchmaking in general, Klusch et. al [25] present a hybrid matchmaker for OWL-S that complements logic based reasoning with approximate matching techniques from Information Retrieval. The work is inspired by earlier work for similarity-based matchmaking among software agents in LARKS [47]. In our own previous work, we have further discussed similarity-based matchmaking for semantic service descriptions [9].

Bernstein et al. suggested precise logical matching in order to increase precision of keyword only based models [5] matching descriptions of a process ontology. Since then Bernstein and colleagues have presented several non-logical approaches to enhance precise logical matchmaking for service discovery including similarity measures from IR, machine-learning, data-mining [23, 24, 22]. Still, while not expecting user requests being specified in terms of complex semantic service descriptions, these works rely on a query language with a relatively high learning curve using a SPARQL-based language for describing user requests and search terms. Somewhat orthogonal, [42] promotes the idea for using SPARQL as a "description language" i.e., an expression language for OWL-S process result's pre- and post-conditions and effects.

In summary, we have observed that, within SWS discovery, exact logical matchmaking is being superseded by similarity-based matchmaking, but still at a level of logical descriptions. Some approaches propose multi-step filtering/or selection, where simple keyword matching or IR methods may precede logical matching. However, although [21] already envisions "*way for requesters to easily locate pre-defined goals e.g. keyword matching*" concrete methods to obtain the relevant keyword set for matching services and associating them with more formal descriptions of a service are rarely found.

For real users, who provide their request in the form of free text, service discovery is hampered by what has been called the *vocabulary problem* [11]: the user requires a service but is unsure of what terms he needs to find it. This problem has previously been described in HCI [11], IR [7] and case-based reasoning [3]. Indeed, it regularly appears in retrieval systems where humans are required to guess an underlying system vocabulary, indexing or reasoning that may be non-intuitive or non coincident with human understanding. In the domain of multimedia retrieval, for example, the gap between the computational representation and the human interpretation of an image is referred to as the 'Semantic Gap' [15]. In the context of this paper, we may term this problem the *Service Discovery Gap* - caused by the breakdown between user vocabulary and expectations and service description. For example, the user's mental model may stress the usefulness of the service outputs in terms of the inputs to a common task, whereas the similarity function may primarily use the service input and operation features. This mismatch means that the user may not have a suitable vocabulary to formulate queries to retrieve relevant services.

This gap has many more faces than only the discrepancy between technical descriptions of services and the vocabulary that might be used for keyword search by potential requesters.

Language Gap: there is a lack of common semantic descriptions of available services by their providers. Where available at all, service descriptions may use descriptions in varying formats, levels of granularity and underlying (logical) languages. Despite this most service frameworks contain the same service aspects contained in WSDL: *Inputs, Outputs, Operations,* (and more rarely *Preconditions* and *Effects,* which are for instance missing in SA-WSDL).

Provider/User Gap: This gap occurs where providers have different intentions for the use of their service than the users who consume the service.

3 Web 2.0 Approaches to Service Discovery

To address this problem we look at how user-defined tags might help. Tags are short informal descriptions, often one or two words long, used by Web users to describe online resources. There are no techniques for specifying "meaning" or inferring or describing relationships between tags. Tag-clouds refer to aggregated tag information, in which a taxonomy or "tagsonomy" emerges through repeated collective usage of the same tags. Part A of Figure 1 illustrates a tag-cloud in the blog domain.
The advantage of using tags in the context of service discovery is that they supply a user-defined vocabulary based on a consensus of how the service is perceived or used in the world. For the needs of our research, we assume that each service that is made available via the Web has its own tag-cloud.

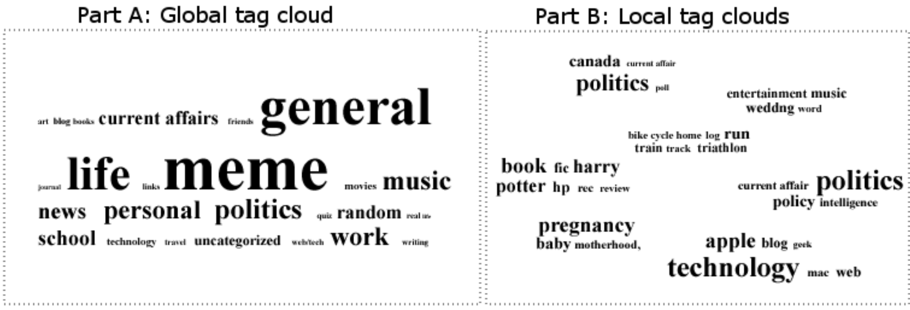


Figure 1: A global tag cloud and several local 'synthetic' tag clouds, which were produced by clustering the associated blog content [17].

3.1 MAC/FAC Similarity Matching

In this section we introduce several approaches to service discovery using Web 2.0, formal service descriptions and similarity matching. The conceptual framework for the similarity matching that we draw upon is from case-based reasoning (CBR), a well-established domain in which similarity-based retrieval is a fundamental part. The key observation we take from this domain is that similarity-based retrieval can be viewed as a process of at least two steps, rather than a single-shot retrieval. An influential theory of similarity-based retrieval in this domain is the *Many Are Called/Few Are Chosen* (MAC/FAC) model of Gentner, Forbus and Law [10].

The first or MAC stage uses a process called *surface similarity*, a relatively inexpensive matching function that uses simple surface level features to return a subset of items from the search space. The second stage or FAC stage involves a deeper, more powerful and (more expensive) matching process carried out on this subset using richer, structured feature information. The MAC/FAC procedure has been used for many years in several forms in case-based reasoning. A typical example is a MAC stage where incremental query-expansion is performed on a database followed by a FAC stage where similarity matching is performed on the resultset [44]. An alternative approach, closer to the approach we propose, involves a MAC stage using collaborative filtering followed by FAC stage using similarity-based matching

[19]. The essential observation is that retrieval can take place incrementally, in at least two steps, where the initial steps involve the retrieval of a subset of item by means of an inexpensive function on surface-level features followed by a refinement of this subset using more sophisticated techniques on structural features.

Approach 1: Matching Tag clouds

This model allows us to incorporate tag information and formal service descriptions into a user-centric service discovery model. If we take a step back and re-think which "semantic descriptions" would best suit typical free text service requests, the obvious solution are simply *tag-clouds* to describe and classify services. By a tag-cloud-based service description C here we mean a set of pairs $<t_i,n_i>$, where t_i is a free text tag and n_i is the frequency of tag t_i in the tag cloud C. Tag clouds are used in faceted browsing and often displayed graphically there, emphasizing the weight n_i by the font size of a tag (see Figure 1). A service tag-cloud is obtained by aggregating the free text tag annotations provided by the users of that service, being the frequency n_i the number of users that included the tag t_i in their annotation of that service.

Such tag-clouds immediately solve the language gap, since there is no more formal language involved which needs mediation. Matching user requests (i.e. a tag set) to tag-clouds is obvious: Tag-clouds have a natural correspondence to the typical vector space model used in standard document classification and information retrieval, and indexing methods. Assuming we had tag-clouds describing each service in place, which properly describe the meaning of a service, matching itself would be an almost straightforward task. The user specified keywords would just be used as a "filter" to mask each service tag cloud, dropping all tags that are not of interest and the weighted sum of this masked tag cloud would denote the degree of match.

To compare tag-clouds, weights are usually normalised. The *normalised tag frequency* r_i of tag t_i in C, where k is the number of tags in C, is defined as:

$$r_i = \frac{n_i}{\sum_k n_k}$$

Let T be a normalised tag-cloud and Q a user specified tag set, then the similarity between T and Q is defined as:

$$sim(T,Q) = \sum_{t_i \in T} \delta(tp_i, Q)$$

where tp_i stands for the normalised tag pair (<tag, weight>), and δ is defined as:

$$\delta(<t_1, r_1>, Q) = \begin{cases} r_1 & if\ t_1 \in Q \\ 0 & otherwise \end{cases}$$

This naïve approach of tag-cloud matching could obviously be refined by lessons learned from the traditional semantic service matchmaking realm. Typical proposals to service matching use asymmetric measures in the degree of match between request and offer, which takes into account the subsumption relation between them (e.g. *plug-in* vs. *subsumes* in Paolucci's [36] approach). For example, suppose two service descriptions s_1 = *buy book*, and s_2 = *buy fiction book*, the degree of match is usually different if the s_1 is the user request and s_2 is the provider, or vice versa. This asymmetry is lost in the tag cloud (there is no subsumption relation between concepts), but might be emulated by using the subset operator between tag sets (e.g. $\{buy, book\} \subseteq \{buy, fiction, book\}$). However, asymmetry might not make sense when the intended use of the similarity measure is clustering (as detailed below), since this process is done without a reference request. Thus, these approaches cannot be directly applied in a clustering context.

The tag-cloud description metaphor may be further refined by dividing each service description according to the aspects common to current SWS description frameworks. For example, instead of a single tag cloud, we could envision separate tag-cloud descriptions per service for *inputs*, *outputs* and *operations*. Requests could be grouped likewise, e.g. providing separate search fields for these different aspects in a tag-based service discovery engine. In that case, service similarity must combine the similarity value for each of these fields. Different options can be considered here. If we consider those fields as a conjunctive set (i.e. all are expected to be matched) then a triangular norm (e.g. the *minimum*) can be used. A more general approach is a weighted sum of each similarity, where the weighting parameters can be established a priori (e.g. equally distributed: 1/3) or defined by the user.

Approach 2: Browsing a Tag-Cloud Concept Hierarchy

Approach 1 provides a means for querying service descriptions using simple natural language query terms. Although a tag-cloud describes the most frequently used terms by other users of a service, a new user may still have trouble formulating a query that would match it, particularly if the tag-cloud is sparse. An alternative approach is to provide a visual browsing mechanism where the various concepts represented in the service space are described using representative tags. To achieve this, we propose a type of ontology that is automatically built by matching similar tag-clouds in order to allow users to browse service descriptions at different levels of granularity. To realise this, we draw upon three techniques. The first is in the area of hierarchical clustering [53], the second is in the area of tag analysis in blogs [17]. The third is in centroid-based classification [14].

In the context of this work, hierarchical clustering allows us to produce a browseable interface of service descriptions at different levels of granularity using tag data. Furthermore, a clustering approach provides the means for implementing recommendation mechanisms: when a user finds a service, other services that belong to the same cluster can be recommended to her.

Hierarchical clustering can be further subdivided into two approaches: agglomerative or bottom-up approaches where data objects are initially assigned to their own clusters and then pairs of similar clusters are repeatedly merged until a whole tree is formed; and partitional or top-down approaches where the entire corpus is initially divided into two clusters and these clusters are repeatedly sub-divided until a whole tree is formed. Conventional wisdom has it that agglomerative approaches, while computationally more expensive, tend to outperform partitional approaches in clustering accuracy. Recent analysis on large, high dimensional data sets has suggested that this is not necessarily the case [53]. Furthermore, a hybrid approach called *constrained agglomerative clustering*, where an initial partitional approach provides constraints for the subsequent agglomerative process, has demonstrated improved clustering performance with small increase in computational cost over partitional approaches [53].

Thus, in approach 2 we use constrained agglomerative clustering to cluster our service descriptions into a concept tree. Each service is represented in terms of its tag-cloud and similarity between services is calculated based on similarity between tag-clouds. Note that semantic concept similarity techniques ([38, 29, 39]) cannot be applied in this context, since they assume that concepts are defined and related to each other in some ontology. In general clustering rests upon a fundamental hypothesis in information retrieval: Van Rijsbergen's cluster hypothesis, which proposes that similar documents are likely to be more relevant to an information requirement than less similar documents [49]. In the context of this paper, we consider a tag-cloud to be a document and an information requirement to be service discovery requirement. We believe that this is a reasonable assumption as tag-cloud data can be pre-processed, stemmed and weighted as input data for clustering in exactly the same way as text document data.

Using the vector-space model, each tag-cloud is represented as a vector in term space and each term in the vector is weighted according to the standard *tf-idf* weighting scheme [41]. In the vector-space model, similarity is calculated using the cosine measure. To prevent large clouds having undue influence during similarity matching, each vector is normalised so that it is of unit length on the hypersphere. The corpus of tag-cloud vectors can then be used as input on the clustering algorithm.

For a detailed account of the constrained agglomerative clustering, we refer the reader to the work of Zhao et al. [53]. The output of the algorithm is a dendrogram that can be browsed from its root nodes (containing all tag cloud documents) to its leaves, where each leaf represents a single tag-cloud (and its associated service). At each concept node, the *extensional* description of the concept is represented in terms of the services associated with the tag-clouds in that node. The *intensional* and thus browsable, description of the concept at each node is easily extracted from the cluster *centroid* at the node.

The cluster centroid at each node is produced by firstly producing a *composite vector* of the tag-cloud vectors contained in the cluster at the node and then normalising each term of the composite vector by the number of tag clouds (at the node). For a node *p*, containing a set *N* of tag cloud vectors, the centroid vector C_p is defined by

$$C_p = \sum_{n \in N} n \Big/ |N|$$

The cluster centroid is a vector that contains a weighted representation of the tags most representative of the concept in cluster. A synthetic tag-cloud can be extracted from the centroid using a threshold to filter lowly weighted terms and using the tag term weights as an input to any tag cloud presentation algorithm.

The overall output is a browseable dendogram, where each node is represented by a synthetic tag cloud representing service description at different levels of specificity (see Figure 2).

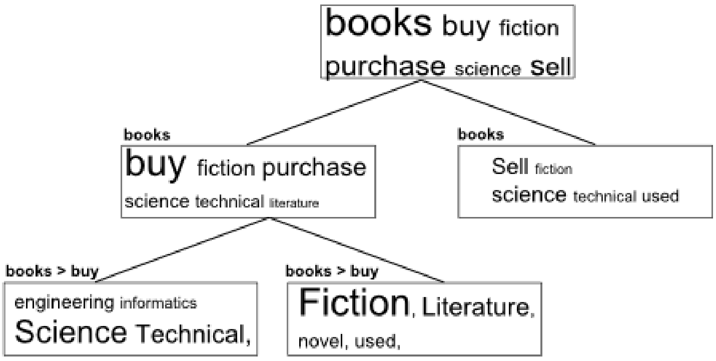


Figure 2: An illustration of a dendrogram induced from tag cloud data. At each node a threshold controls how many tags are displayed. For each sub-node, the single mostly highly weighted tag in each parent node is displayed as a part of a path summary. E.g. books>buy

The Cold Start Problem

A fundamental problem with all online services that rely upon collecting social data is the *cold-start problem* [30, 43] which refers to the difficulty in offering a service when there is yet no user data and the difficulty in collecting user data when there is no service. Although, the problem has generally been defined in terms of collaborative recommendation systems, it is equally relevant to services that rely upon user submitted tag data, such as the one that we have just defined.

A typical solution to the cold-start problem in collaborative recommendation is to deploy what is termed a *content-based* service along side the collaborative service [43, 6]. The key idea is that the content-based service can be deployed where there is insufficient social data to make a socially-based recommendation. The terms 'content' loosely refers to any non-socially derived descriptive data that can be used for retrieval purposes, typically using IR inspired matching algorithms.

In the context of this work, we plan to leverage a 'content' based approach to the 'cold start' problem by clustering the semantic descriptions that the service providers add to their services. These may both consist of well-structured formal descriptions that follow one of the SWS frameworks discussed earlier, i.e. OWL-S, WSMO, SAWSDL, textual descriptions, or tag sets provided directly by the provider. Note that, in the case of SWS descriptions, similarity-based matchmaking techniques for semantic service descriptions (as described in section 2) must be used.

Our technique draws upon the work of Hayes et al. which uses content clustering and tags to produce interpretable tag-based summaries of data in the blog domain [16] (See Figure 1). The essential observation of this work is that where tag data is sparse, the underlying content data can be clustered, producing synthetic tag-clouds as by-process. These tag clouds are shown to be strong indicators of the cluster semantics and coherence. Currently we are collecting service provider descriptions, which will act as input for a cold-start clustering process. The cluster concepts will be represented by synthetic tag clouds extracted from the available service descriptions, providers' tags, as well as tag sets extracted from service descriptions by traditional information extraction techniques.

The cold start approach raises the question as to when we know when there is enough social data or what to do when there is social data for some services and not for others. Although we do not pretend to have an answer at this point, we do acknowledge that a substantial amount of research has been directed to the question of interleaving outputs from different models in the field of recommender systems [6]. Furthermore, the success of a system must be measured in terms of use satisfaction and, in this regard, we plan to exploit our previous experience in evaluating online whether one recommender algorithm improves upon another [18]. Another question is how often the clustering process needs to be carried out, given that tag-clouds evolve over time. We propose that experimental analysis will provide the answer to this question and direct the reader to our initial work on this subject in the area of clustering blog data [17].

4 Conclusions and Future Work

In this paper we started by describing the current state in service provision and by outlining the problems that are raised during service discovery by introducing the service discovery gap. Afterwards, we made the assumption that in our world of services, services are semantically described both by service providers and by users. The semantic descriptions of the services providers follow some SWS framework, while those of the users are expressed via tag clouds.

Our solution allows users, at a first step, to query or browse services using free text tags, thus providing an interface in terms of the users' vocabulary instead of the service's vocabulary. Then, in a second step, the users can query deeper in this narrowed result set by using concepts that are common between different service models, i.e. input, output, and operation. A description of the second step is beyond the scope of this paper. However, we have recently developed a common mapping language between service descriptions that allows their input, output and operation tasks to be compared. After step one, where the user has found a candidate set of services using the tag cloud method, he/she can make an informed choice of services available without having to learn the syntax of each service type. Further populating this set of common concepts is one of the most important future steps in our research agenda.

Two approaches to service discovery have been suggested: a similarity-based approach for querying tag clouds and a browsing approach using hierarchical clustering. In addition, we proposed a content-based approach to solve the cold-start problem.

As part of our future work, we plan to implement and evaluate these different approaches using standard evaluation techniques from machine learning and information retrieval, as well as on-line approaches to measuring user satisfaction [18].

Acknowledgments

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

References

1. Akkiraju, R. et. al.: Web Service Semantics - WSDL-S, W3C Member Submission, http://www.w3.org/Submission/WSDL-S/
2. Ankolekar, A. et. al: DAML-S: Semantic Markup For Web Services. Proceedings of the International Semantic Web Workshop (2001).
3. Avesani, P., Hayes, C., Cova, M. (2005). Language Games: Solving the Vocabulary Problem in Multi-Case-Base Reasoning in proceedings of the 6th International Conference on Case-Based Reasoning, ICCBR 05. Hector Munoz-Avila, Francisco Ricci (eds), Springer
4. Berners-Lee, T., Hendler, J., Lassila, O.L.: The semantic web. Scientific American, vol. 284(5), pp. 34–43 (2001)
5. Bernstein, A., Klein, M.: Towards high-precision service retrieval. IEEE Internet Computing, vol. 8(1): 30-36 (2004)
6. Burke, R. 2002. Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-Adapted Interaction 12, 4 (Nov. 2002), 331-370.
7. Chen, H., Ng, T. D., Martinez, J., and Schatz, B. R. 1997. A concept space approach to addressing the vocabulary problem in scientific information retrieval: an experiment on the worm community system. J. Am. Soc. Inf. Sci. 48, 1 (Jan. 1997), 17-31.
8. d'Amato, C., Staab, S., Fanizzi, N., and F. Esposito. *Efficient discovery of services specified in description logics languages*. In Proceedings of the ISWC-2007 Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web (SMR2), 2007.
9. Fernandez, A., Polleres, A., and Ossowski, S. Towards Fine-grained Service Matchmaking by Using Concept Similarity, In Proceedings of the ISWC-2007 Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web (SMR2), 31-45. 2007.
10. Forbus, K. D., Gentner D., and Law, K. 1994. MAC/FAC: A Model of Similarity-based Retrieval. Cognitive Science 19, 141-205.
11. Furnas, G. W., Landauer, T. K., Gomez, L. M., and Dumais, S. T. 1987. The vocabulary problem in human-system communication. Commun. ACM 30, 11 (Nov. 1987), 964-971.
12. Gisolfi, D., IBM: *An introduction to dynamic e-business*. http://www.ibm.com/developerworks/webservices/library/ws-arc1/ (2001).
13. Grimm, S., Motik, B., Preist, C.: *Variance in e-business service discovery*. In Proceedings of the ISWC Workshop on Semantic Web Services. (2004).

14. Han, E. and Karypis, G. 2000. Centroid-Based Document Classification: Analysis and Experimental Results. In Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery (September 13 - 16, 2000). D. A. Zighed, H. J. Komorowski, and J. M. Zytkow, Eds. Lecture Notes In Computer Science, vol. 1910. Springer-Verlag, London, 424-431.
15. Hare, J. S., Lewis, P. H., Enser, P. G. B. and Sandom, C. J. (2006) Mind the Gap: Another look at the problem of the semantic gap in image retrieval. In: Multimedia Content Analysis, Management and Retrieval 2006, 17-19 January, San Jose, California, USA.
16. Hayes, C., Avesani, P. Using Tags and Clustering to Identify Topic-Relevant Blogs. In Proceedings of the 1st International Conference on Weblogs and Social Media (ICWSM 07): 67—75 Boulder, Colorada, U.S.A, 2007
17. Hayes, C., Avesani, P., and Bojars, U. An Analysis of Bloggers, Topics and Tags for a Blog Recommender System. In Bettina Berendt, Andreas Hotho, Dunja Mladenic, Giovanni Semeraro (ed.), LNAI: Advances in Web Mining Springer, 2007.
18. Hayes, C., Massa, P., Avesani, P., Cunningham, P. (2002) An On-line Evaluation Framework For Recommender Systems in Proceedings of the Workshop on Recommendation and Personalization in eCommerce at AH2002, University of Malaga Press, 2002. F. Ricci and B. Smyth (editors). ISBN 699-8196-7
19. Hayes, C., Cunningham, P. (2003) Context Boosting Collaborative Recommendations in Applications and Innovations in Intelligent Systems XI, proceedings of the Twenty-Third Annual International Conference of the British Computer Society's Specialist Group on Artificial Intelligence (SGAI), Cambridge. Springer Verlag.
20. InfoWorld: http://www.infoworld.com/article/08/02/25/news-soa-adoption_1.html (2008)
21. Keller, U., Lara, R., Lausen, H., Polleres, A., Fensel, D.: Automatic Location of Services. Proceedings of the 2nd ESWC 3532 (2005)
22. Kiefer, C., and Bernstein, A. The Creation and Evaluation of iSPARQL Strategies for Matchmaking. 5th ESWC, 2008.
23. Kiefer, C., Bernstein, A. and Locher, A. Adding Data Mining Support to SPARQL via Statistical Relational Learning Methods. 5th ESWC, 2008.
24. Kiefer, C., Bernstein, A., Lee, H., Klein, M., and Stocker, M. Semantic Process Retrieval with iSPARQL. 4th ESWC, 2007.
25. Klusch, M., Fries, B., Sycara, K.: Automated semantic web service discovery with owls-mx. Proceedings of the AAMAS, pp. 915-922 (2006)
26. Kopecky, J., Vitvar, T., Bournez, C., Farrell, J.: SAWSDL: Semantic Annotations for WSDL and XML Schema, IEEE Internet Computing, vol. 11 (6) (2007)
27. Kopecky, J., Vitvar, Zaremba, M., Fensel, D.: WSMO-Lite: Lightweight Semantic Descriptions for Services on the Web Full, Proceedings of 5th ECOWS, pp. 77-86 (2007)
28. Li, L. and Horrocks, I. A software framework for matchmaking based on semantic web technology. Int. J. of Electronic Commerce, 8(4):39–60, 2004.
29. Li, Y., Bandar, Z., and McLean, D. An approach for measuring semantic similarity between words using multiple information sources. IEEE Trans.

Knowl. Data Eng., 15(4):871–882, 2003.
30. Maltz, D. and Ehrlich, K. 1995. Pointing the way: active collaborative filtering. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Denver, Colorado, United States, May 07 - 11, 1995). I. R. Katz, R. Mack, L. Marks, M. B. Rosson, and J. Nielsen, Eds. Conference on Human Factors in Computing Systems. ACM Press/Addison-Wesley Publishing Co., New York, NY, 202-209.
31. Martin, D. et. al.: OWL-S: Semantic Markup for Web Services, W3C Member Submission, http://www.w3.org/Submission/OWL-S/ (2004)
32. Murugesan, S.: Understanding Web 2.0. IT Professional, vol. 9(4), pp. 34-41 (2007)
33. O'Reilly, T.: What Is Web 2.0, Design Patterns and Business Models for the Next Generation of Software, http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html (2005)
34. OASIS Reference Model for Service Oriented Architecture, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm
35. OPEN Group, http://www.opengroup.org/projects/soa/doc.tpl?gdid=10632
36. Paolucci, M., Kawamura, T., Payne, T., and Sycara, K. Semantic Matching of Web Service Capabilities. In ISWC, pages 333–347. Springer Verlag, 2002.
37. Passant, A. MOAT Project: moat-project.org/
38. Rada, R., Mili, H., Bicknell, E. and Blettner, M. Development and application of a metric on semantic nets. IEEE Transactions on Systems, Man and Cybernetics,, 19(1):17–30, 1989.
39. Resnik, P. Using information content to evaluate semantic similarity in a taxonomy. In IJCAI, pages 448–453, 1995.
40. Roman, D., Lausen, H., Keller, U. (eds.), Web Service Modeling Ontology (WSMO), Technical report, WSMO Final Draft, http://www.wsmo.org/TR/d2/v1.2/ (2005)
41. Salton, G. 1989. Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley.
42. Sbodio, M. L., Moulin. C.: SPARQL as an expression language for OWL-S. Proceedings of OWL-S: Experiences and Directions, 4th ESWC (2007)
43. Schein, A. I., Popescul, A., Ungar, L. H., and Pennock, D. M. 2002. Methods and metrics for cold-start recommendations. In Proceedings of the 25th Annual international ACM SIGIR Conference on Research and Development in information Retrieval (Tampere, Finland, August 11 - 15, 2002). SIGIR '02. ACM, New York, NY, 253-260
44. Schumacher, J., Bergmann, R. (2000) An effective approach for similarity-based retrieval on top of relational databases, in: Fifth European Workshop on Case-Based Reasoning. Springer-Verlag.
45. Sheth, A., Gomadam, K., Lathem, J.: SA-REST: Semantically Interoperable and Easier-to-Use Services and Mashups, IEEE Internet Computing, vol. 11 (6): pp. 91-94 (2007)
46. Stollberg, M., Keller, U., Lausen, H., Heymans. S.: Two phase web service discovery based on rich functional descriptions. Proceedings of 4th ESWC (2007)
47. Sycara, K., Klusch, M., Widoff, S., Lu, J., Larks, K.: Dynamic matchmaking

among heterogeneous software agents in cyberspace. Autonomous Agents and Multi-Agent Systems, 5(2), Kluwer (2002)
48. Sycara, K., Paolucci, M., Anolekar, A., Srinivasan, N.: Automated discovery, interaction and composition of semantic web services. Web Semantics, 1(1), Elsevier (2003).
49. Van Rijsbergen, C. J. Information Retrieval. Butterworths, second edition, 1979.
50. W3C: SWSF Submission, www.w3.org/Submission/SWSF/
51. W3C: Team Comment on Web Service Modeling Ontology (WSMO) Submission, http://www.w3.org/Submission/2005/06/Comment (2005)
52. Xuan S.: Semantic Web Services: An Unfulfilled Promise. IEEE IT Professional, vol. 9(4), pp. 42 – 45 (2007)
53. Zhao, Y., Karypis, G., and Fayyad, U. 2005. Hierarchical Clustering Algorithms for Document Datasets. Data Min. Knowl. Discov. 10, 2 (Mar. 2005), 141-168.

ISWC 2008

The 7th International Semantic Web Conference

October 26 – 30, 2008

Congress Center, Karlsruhe, Germany