The iSeM Matchmaker: A Flexible Approach For Adaptive Hybrid Semantic Service Selection

P. Kapahnke[a], M. Klusch[a]

[a]*German Research Center for Artificial Intelligence (DFKI), Saarbruecken, Stuhlsatzenhausweg 3, Germany.*

Abstract

We present iSeM (intelligent Service Matchmaker), a precise hybrid and adaptive matchmaker for semantic Web services, which exploits functional service descriptions in terms of logical signature annotations as well as specifications of preconditions and effects. In particular, besides well-known strict logical matching filters and non-logic-based textual and structural signature matching, it adopts approximated reasoning based on logical concept abduction and contraction for the description logic subset SH with information-theoretic valuation for matching inputs and outputs. In addition, it uses a stateless logical specification matching approach, which applies the incomplete but decidable θ-subsumption algorithm for preconditions and effects. The optimal aggregation strategy of all those aspects is learned off-line by means of a binary SVM-based service relevance classifier in combination with evidential coherence-based pruning to improve ranking precision with respect to false classification of any such variant on its own. We demonstrate the additional benefit of the presented approximation and the adaptive hybrid combination by example and by presenting an experimental performance analysis.

Keywords: Semantic Web Service Selection, Matchmaker

1. Introduction

Semantic service selection is commonly considered key to the discovery of relevant services in the semantic Web, and there are already quite a few matchmakers available for this purpose as for example the summary given in [13] and the annual international *Semantic Service Selection* (S3) contest[1] show. In this paper, we present the first adaptive semantic service IOPE (inputs, outputs, preconditions and effects) matchmaker[2]. In essence, its innovative features are (a) approximated logical signature (IO) matching based on non-monotonic concept

Email addresses: `patrick.kapahnke@dfki.de` (P. Kapahnke), `klusch@dfki.de` (M. Klusch)

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

[2]This article is a revised and extended version of [11]

abduction and contraction in the description logic subset SH[3] together with information-theoretic similarity and evidential coherence-based valuation of the result to avoid strict logical false negatives, (b) stateless strict logical specification (PE) plug-in matching to avoid failures of signature matching only, and (c) SVM (support vector machine)-based semantic relevance learning adopted from [10] but extended to full functional service profile (IOPE) matching and use of approximated IO matching results to prune the feature space for precision. Performance evaluation results particularly indicate that this kind of approximated logical matching performs close to its logical and non-logical counterparts (text and structural matching) and significantly improves ranking precision in adaptive hybrid combination with those.

The remainder of the article is structured as follows. We motivate our matchmaker iSeM and give an overview of our approach in Section 3. A detailed description of its signature matching filters with focus on approximated logical matching is given in Section 3.1, while Section 3.2 discusses its stateless, logical specification matching filter. Section 3.3 describes the SVM-based service relevance learning for selection, which is followed by implementation details including an introduction to the S2M2 framework for semantic matchmaker development in Section 4. Performance evaluation and result discussion are provided in Section 5, after which we comment on related work in Section 6. Finally, we conclude in Section 7.

2. Motivation

The specific problems of semantic service selection the matchmaker iSeM has been particularly designed to cope with are motivated by the following service example, which is used throughout the paper.

Example 1: Consider the semantic profiles of service request R and offer S in Figure 1, taken from the standard test collection OWLS-TC4 according to which S is relevant to R.

The desired service R is supposed to purchase a book for a given person by debiting his own debit account, shipping the book to him and eventually acknowledging the completed deal. The e-shopping service S, like amazon.com, offers arbitrary articles including books that are requested by some customer whose own credit card account gets respectively charged while sending an invoice for and pricing information about the deal. Both services are written in OWL-S with semantic signature concept definitions in description logic OWL-DL and their logical preconditions and effects in SWRL. In the following, we assume the matchmaker to have an appropriate shared ontology and a service registry available over which semantic service selection is performed. ∘

[3]http://www.cs.man.ac.uk/ ezolin/dl/

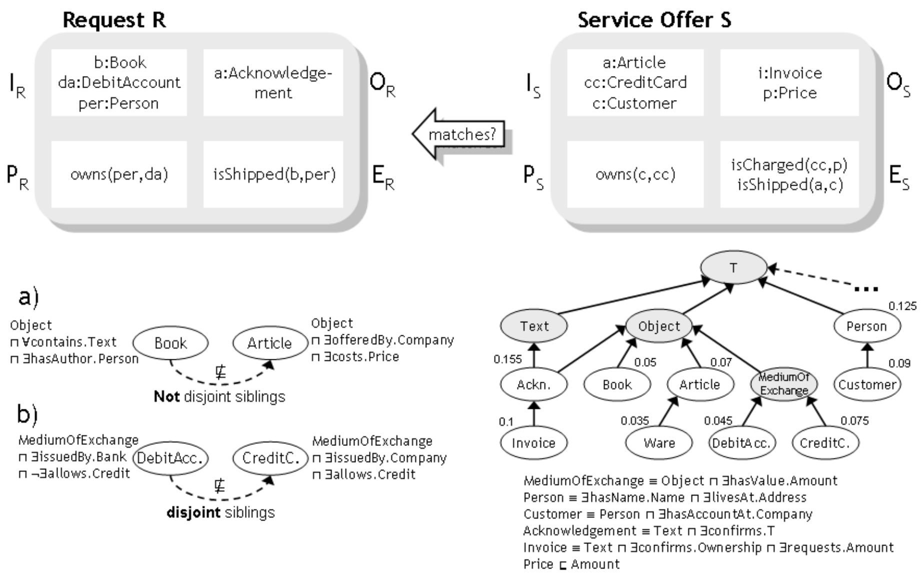


Figure 1: Service request (book purchasing) and relevant service offer (article purchasing).

False negatives of strict logical signature matching. The majority of semantic service matchmakers perform logical signature matching (see [13] and S3). One prominent set of strict logical matching filters for this purpose is provided below [18, 9]. Each of these filters requires (a) each service input concept to be more generic than or equal to those provided in the request and (b) the complete requested output to be covered by that of the service in terms of different types of logical subsumption relations.

Definition 1: *Strict logical signature matching.*
Let S, R be semantic service offer and request and $S_{in}, S_{out}, R_{in}, R_{out}$ the multisets of input and output concepts of the semantic signatures of S and R defined in a shared OWL ontology O. The logical filter definitions used in iSeM are inspired by OWLS-MX3 [10] but are more strict with respect to the mapping of matching request and offer parameters. While each element of R_{in} (S_{out}) could have more than one matching part in S_{in} (R_{out}) in OWLS-MX3, iSeM requires strictly injective concept assignments for logicl matching filters. This refinement was based on previous observations regarding cases of false positives in OWLS-MX3 due to this characteristic. To accomplish this, $BPG_{\sqsubseteq}(\bar{C}, \bar{D})$ is defined as the set of concept assignments (C, D) with $C \in \bar{C}$ and $D \in \bar{D}$ that form an injective mapping as valid solution of bipartite graph matching on a graph with $\bar{C}$ and $\bar{D}$ as nodes and weighted edges between them. The weights v between concepts C and D indicate whether $C \sqsubseteq D$ holds: $v = 1$ iff $C \sqsubseteq D$ else $v = 0$. That is, $BPG_{\sqsubseteq}(\bar{C}, \bar{D})$ is the mapping that maximizes the sum of binary weights v for the given logical relation type. For the definition of filters analo-

gous to OWLS-MX3, $BPG_X(\bar{C},\bar{D})$ with $X \in \{(\equiv, \sqsubseteq_1, \sqsupseteq_1\}$ are introduced, with $\sqsupseteq_1$ denoting *direct* parent/child relations in a subsumption graph. Moreover, if no assignment is possible, i.e. $|\bar{D}| < |\bar{C}|$, it holds that $BPG_X(\bar{C},\bar{D}) = \emptyset$. The degree MatchIO$_{Logic}(R,S)$ of strict logical signature matching is then finally defined as follows:
MatchIO$_{Logic}(R,S) \in \{$**Exact, Plug-in, Subsumes, Subsumed-by, LFail**$\}$ with

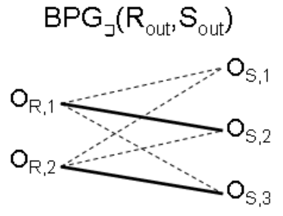
Exact: $BPG_{\equiv}(S_{in},R_{in}) \neq \emptyset \wedge \forall (I_S,I_R) \in BPG_{\equiv}(S_{in},R_{in}) : I_S \equiv I_R$
$\wedge\, BPG_{\equiv}(R_{out},S_{out}) \neq \emptyset \wedge \forall (O_R,O_S) \in BPG_{\equiv}(R_{out},S_{out}) : O_R \equiv O_S$

Plug-in: $BPG_{\sqsupseteq}(S_{in},R_{in}) \neq \emptyset \wedge \forall (I_S,I_R) \in BPG_{\sqsupseteq}(S_{in},R_{in}) : I_S \sqsupseteq I_R$
$\wedge\, BPG_{\sqsupseteq_1}(R_{out},S_{out}) \neq \emptyset \wedge \forall (I_S,I_R) \in BPG_{\sqsupseteq_1}(R_{out},S_{out}) : O_R \sqsupseteq_1 O_S$

Subsumes: $BPG_{\sqsupseteq}(S_{in},R_{in}) \neq \emptyset \wedge \forall (I_S,I_R) \in BPG_{\sqsupseteq}(S_{in},R_{in}) : I_S \sqsupseteq I_R$
$\wedge\, BPG_{\sqsupseteq}(R_{out},S_{out}) \neq \emptyset \wedge \forall (I_S,I_R) \in BPG_{\sqsupseteq}(R_{out},S_{out}) : O_R \sqsupseteq O_S$

Subsumed-by: $BPG_{\sqsupseteq}(S_{in},R_{in}) \neq \emptyset \wedge \forall (I_S,I_R) \in BPG_{\sqsupseteq}(S_{in},R_{in}) : I_S \sqsupseteq I_R$
$\wedge\, BPG_{\sqsubseteq_1}(R_{out},S_{out}) \neq \emptyset \wedge \forall (I_S,I_R) \in BPG_{\sqsubseteq_1}(R_{out},S_{out}) : O_R \sqsubseteq_1 O_S$

LFail: None of the above logical filter constraints are satisfied. $\diamond$



Edge	*v*
$(O_{R,1},O_{S,1})$	0
$\mathbf{(O_{R,1},O_{S,2})}$	**1**
$(O_{R,1},O_{S,3})$	1
$(O_{R,2},O_{S,1})$	0
$(O_{R,2},O_{S,2})$	0
$\mathbf{(O_{R,2},O_{S,3})}$	**1**

Edge	*v*
$(O_{R,1},O_{S',1})$	0
$\mathbf{(O_{R,1},O_{S',2})}$	**1**
$(O_{R,1},O_{S',3})$	1
$\mathbf{(O_{R,2},O_{S',1})}$	**0**
$(O_{R,2},O_{S',2})$	0
$(O_{R,2},O_{S',3})$	0

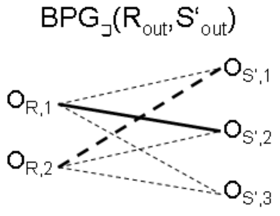


Figure 2: BPG for output part of Subsumes filter, positive and (partially) negative example.

Figure 2 shows two examples of the application of $BPG_{\sqsupseteq}$: on the left hand side, the algorithm is able to find an assignment for each parameter in R_{out} for which the required subsumption relation holds; the assignment on the right is (one of) the best possible solutions but does not satisfy the requirement $O_{R,2} \sqsupseteq O_{S,1}$ (i.e. $v = 0$), which causes the filter to fail.

Applying these strict logical matching filters to the running example introduced above produces a logical fail (LFail), hence a false negative. The reasons are that (a) the inputs book and article are not strictly logically disjoint siblings in the ontology, that is ($Book \sqcap Article \not\sqsubseteq \bot$), and (b) the inputs debit account and credit card are strictly logically disjoint, that is ($DebitAccount \sqcap CreditCard \sqsubseteq \bot$).

Such cases of logical signature mismatches may appear quite often, in fact, applying the above filters to the de facto standard collection OWLS-TC4 yields a relatively high number of strict logical false negatives for each request in the order of 45% of the size of its relevance set in average. As shown, for example, in [10, 9, 12] and the contest S3, some hybrid combination of strict logical with non-logic-based approximated signature matching methods may avoid failures of strict logical signature matching filters defined above in practice[4]. But how can logical matching itself be improved by what kind of complementary approximation (cf. Section 3.1), and how well does this perform compared to and in combination with its non-logic-based counterparts in practice (cf. Section 5.1)?

Failures of signature matching only. It is well known that matching of semantic signatures only may fail in many cases, since they do not capture the functional behavior commonly encoded in logical service preconditions and effects (PE). There are different approaches to logical PE-matching [13] - but which one can most effectively be used in a third-party matchmaker that usually has no access to concept instances describing the states of service providers and requesters (cf. Section 3.2)?

Best combination of semantic matching filters. How to best combine different kinds of semantic service matching filters in terms of precision? One option proposed, for example, in [8, 10, 12] is to let the matchmaker learn the optimal aggregation of different matching results for its semantic relevance decision - rather than to put the burden of finding and hard-coding the solution by hand on the developer. Though this turned out to be quite successful in the S3 contest restricted to semantic signatures, how can approximated logical matching be used to improve the learning for better precision of service selection (cf. Section 3.3)?

3. iSeM Matchmaker: Overview

Before delving into the technical details of the matchmaker iSeM, we shall first provide an overview of its functionality.

Matchmaking algorithm in brief. For any given service request R and service offers $S \in SR$ described in OWL-S or SAWSDL, with SR being the service registry of iSeM, the matchmaker returns a ranked set of relevant services as its answer set to the user. For this purpose, it first learns the weighted aggregation of different kinds of service IOPE matching results off line over a given training set of positive and negative samples by means of SVM-based binary relevance classification with ranking. These different kinds of matching approaches include strict and approximated logical, text similarity-based and structural semantic

[4]Avoidance and higher (lower) ranking of false negatives (positives) increases average precision of ranked result lists

matching of service signatures (IO) in SH[5], as well as stateless, logical plug-in matching of service preconditions and effects (PE) in SWRL, if they exist. Once learning has been done, the same filters are used by the learned relevance classifier for selecting relevant services for previously unknown requests. iSeM may be classified as an adaptive, hybrid semantic service IOPE matchmaker [13].

Hybrid signature (IO) matching. Logical signature matching of iSeM comes in two complementary flavors: Strict logical matching and approximated logical matching. For every service pair (R, S) for which strict logical signature matching $\text{MatchIO}_{Logic}(R, S)$ as defined above (Section 2, Def. 1) fails, iSeM computes the approximated logical matching degree $\text{MatchIO}_{ALogic}(R, S)$ based on approximated subsumption relations $(C \sqsubseteq_{AC} D)$ between I/O concepts C, D via contraction and structured abduction together with their information-theoretic valuation. This leads to two hypotheses of approximated logical signature matching, that are approximated logical plug-in (H_1) and subsumed-by (H_2), both of which weighted by their averaged informative quality $v \in [-1, 1]$. Eventually, the degree $\text{MatchIO}_{ALogic}(R, S) = (H,\ v)$ of approximated logical service signature matching is determined as the hypothesis H with maximal valuation v. The approximated logical matching results are used in the learning process over a given training set of service pairs to prune the respective feature space restricted to logic-based matching to compensate for strict logical false negatives. In addition, iSeM performs non-logic-based approximated matching, that are text and structural semantic similarity-based signature matching for which purpose it applies the respective filters of OWLS-MX3 [10] (cf. Section 3.1).

Logical specification (PE) matching. To cope with failures of signature matching only and to allow for third-party matchmaking without having access to service concept instances, iSeM performs stateless, logical plug-in matching $\text{MatchPE}(S, R)$ of service preconditions and effects by means of approximated theorem proving, that is theta-subsumption, of required logical PE-implications like in LARKS[18] (cf. Section 3.2).

Learning of full service profile (IOPE) selection. To combine the results of its different IOPE matching filters for optimal precise service selection, iSeM performs binary SVM-based semantic relevance learning off line over a given training set of positive and negative samples (S, R) each of which is represented as a vector x in the 10-dimensional feature space of different matching filters. This space gets particularly pruned by exploiting the approximated logical signature matching results to compensate for strict logical false negatives. Once that has been done, the learned binary classifier d with ranking r is applied by iSeM to any service pair (S, R) with unknown request R to return the final

[5]Restriction to annotation in SH is due to respective limitation of the adopted concept abduction reasoner [4]; its extension to $SHOIN$ is ongoing.

result: MatchIOPE$(S, R) = (d, r)$ (cf. Section 3.3, 5.1).

3.1. Hybrid Semantic Signature Matching

Semantic signature matching in iSeM is performed by means of both logic-based and non-logic-based matching. While the first type basically relies on strict logical (cf. Definition 1) and approximated logical concept subsumptions (cf. Section 3.1.1), the second exploits text and structural similarities of signature concepts (cf. Section 3.1.2). Both kinds of approximated logical and non-logic-based matching are performed by iSeM in particular to compensate for strict logical signature matching failures in due course of its relevance classification learning (cf. Section 3.3).

3.1.1. Approximated Logical Matching

Inspired by [3, 4, 16], approximated logical signature matching of a given service pair (S, R) relies on the combined use of logical contraction and abduction of signature concepts for approximated concept subsumption (cf. Definition 2) which is valuated in terms of the information gain and loss induced by its construction (cf. Definition 3). Eventually, we extend both means of approximation and valuation on the concept level to its application on the signature level (cf. Definition 4).

Definition 2: *Logical concept contraction and abduction.*
Let C, D be concepts of an ontology O in SH. The *contraction* of C with respect to D is $CCP(C, D) = (G, K)$ with $C \equiv G \sqcap K$ and $K \sqcap D \not\sqsubseteq \bot$.[6] The abducible concept K^h is derived from concept K through rewriting operations [4]:
$K^h = h_0 \sqcap rew(K)$, $rew(A) = A$, $rew(\neg A) = \neg A$, $rew(C_1 \sqcap C_2) = rew(C_1) \sqcap rew(C_2)$, $rew(\exists R.C) = \exists R.(h_i \sqcap rew(C))$ and $rew(\forall R.C) = \forall R.(h_i \sqcap rew(C))$; where i is incremented per application of rew, A is a primitive component (in the logical unfolding of K in O), C_i are concepts in SH, and $\bar{H} = (h_0, \ldots, h_n)$ denotes variables in the resulting concept structure, where additional definitions are added for approximation. The *Structural abduction* of concept K with respect to D is $SAP(K, D) = H = (H_0, \ldots, H_n)$ with $\sigma[\bar{H}, H](K^h) \sqsubseteq D$ and $\sigma[\bar{H}, H](K^h) \not\sqsubseteq \bot$. The *approximated concept* $C' := \sigma[\bar{H}, H](K^h)$ of C with respect to D is constructed by applying $\sigma[\bar{H}, H] = \{h_0 \mapsto H_0, \ldots, h_n \mapsto H_n\}$ to the abducible concept K^h. The *approximated logical concept subsumption* $C \sqsubseteq_{AC} D$ is defined as follows: $C \sqsubseteq_{AC} D \Leftrightarrow C' \sqsubseteq D$ with $(G, K) = CCP(C, D)$, $H = SAP(K, D)$ and $C' = \sigma[\bar{H}, H](K^h)$. $\diamond$

To avoid strict logical false negatives leading to lower average precision, iSeM assumes the user to be willing to give up those parts of logical signature concept definitions that cause strict logical subsumption failures and keeping the

[6] K ("‘keep"’) denotes the compatible part of C with respect to D, while G ("‘give up"’) denotes the respectively incompatible part.

remaining parts instead. The latter are used to compute approximated concept subsumption relations and the respectively approximated signature matching. A tableau algorithm for computing near-optimal solutions to the problem is given by Di Noia et al. in [4]. Figure 3 provides a schematical overview of the approximation process: given the incompatible concept definitions C and D, the *contraction* is computed to establish compatibility in terms of a less specific definition K based on C (step 1). Based on this result, *structural abduction* is applied to construct the approximation C', for which concept subsumption $C' \sqsubseteq D$ holds (step 2).

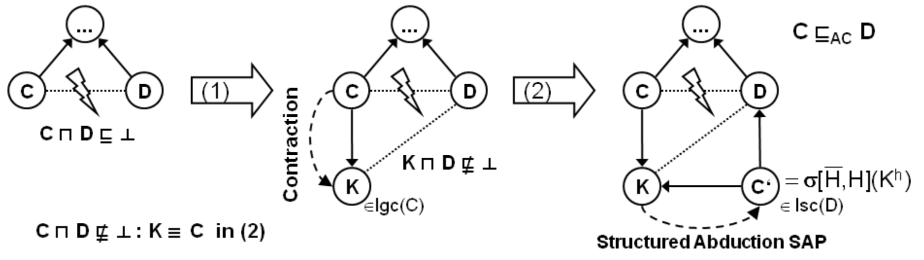


Figure 3: Approximated logical concept subsumption. Arrows denote subsumption relations, dashed lines concept (in-)compatibility.

Example 2: Consider Example 1. The approximated logical subsumption between strict logically disjoint siblings $DebitAccount, CreditCard$ is computed as follows:
$(G, K) = CCP(DA, CC) = (\neg\exists allows.Credit^P, MOE \sqcap \exists issuedBy.Bank^P)$, i.e. the restriction of not allowing credit of the debit account is given up, which establishes compatibility with the $CreditCard$ definition.
$K^h = h_0 \sqcap Object^P \sqcap \exists hasValue.(h_1 \sqcap Value^P) \sqcap \exists issuedBy.(h_2 \sqcap Bank^P)$. The abducible concept K^h determines the positions for concept refinement in the structure of the remaining concept definition K.
$\bar{H} = (h_0, h_1, h_2)$, $H = SAP(DA, CC) = (\exists allows.Credit^P, \top, Company^P)$ then is the solution computed according to [4].
The approximated concept $DebitAccount'$ is then constructed using the following mapping function: $\sigma[\bar{H}, H] = \{h_0 \mapsto \exists allows.Credit^P, h_1 \mapsto \top, h_2 \mapsto Company^P\}$,
$DA' = \sigma[\bar{H}, H](K^h) = \exists allows.Credit \sqcap MOE \sqcap \exists issuedBy.(Bank \sqcap Company)$. DA, CC and MOF are abbreviations for concept names $DebitAccount$, $CreditCard$ and $MediumOfExchange$ respectively. It holds that $DebitAccount'$ $\sqsubseteq CreditCard$, hence $DebitAccount \sqsubseteq_{AC} CreditCard$. ∘

It is worth mentioning that for the special case, where $C \sqsubseteq D$ initially holds, the algorithm presented in [4] that is used by iSeM yields a trivial solution, which later on causes the overall aproximate logic-based matching filter to be redundant to strict logic-based matching in terms of (true or false) positive classification, i.e. it can only be used to remedy strict logical matching failures. This fact is formalized by the following lemma and will be used in subsequent analysis of this behaviour:

Lemma 1: *Trivial approximated concept subsumption.*
Given two concepts C and D in *NNF* and satisfiable in SH with $C \sqsubseteq D$ and a T-box $T \subset SH$. The application of the tableaux-algorithm for approximated concept subsumption given in [4] to the problem $SAP(C, D)$ always yields the trivial solution $H = \langle H_0, \ldots, H_n \rangle = \langle \top, \ldots, \top \rangle$.

Proof: SAP is applicable per Definition 2, since $C \sqsubseteq D$ implies $C \sqcap D \not\sqsubseteq \bot$. $H = \langle \top, \ldots, \top \rangle$ is a solution for $SAP(C, D)$, since $\sigma[\bar{H}, H](C^h) = C$ for C in *NNF*. Thus, because of the original assumption $C \sqsubseteq D$, it holds that $\sigma[\bar{H}, H](C^h) \sqsubseteq D$ and $\sigma[\bar{H}, H](C^h) \not\sqsubseteq \bot$. A tableau τ for the proposition $T \models C \sqsubseteq D$ is already *closed* per definition. Algorithm 1 given in [4] then yields the following substitution because of the conditional given in line 4: $\sigma = \{h_0 \mapsto \top, \ldots, h_n \mapsto \top\}$.

In order to rank the computed approximations, we valuate them by means of their informative quality. Roughly, the informative quality of approximated logical subsumption between signature concepts C, D is the difference between the information gain and loss induced by its construction. That is, the utility of the respectively approximated concept C' is the trade off between its information-theoretic similarity [16] with the original concept C and the targeted one D. The similarity is based on the probabilistic information content of concepts with respect to the frequency of their occurrence in semantic service signatures.

Definition 3: *Informative quality of approximated subsumption.*
Let SR be the set of service offers registered at the matchmaker (service registry), S_{in}, S_{out} the multi-sets of concepts used for signature parameter annotation of service S, $SAC(SR)$ the set of all concepts used for annotating services in SR. We define the *informative quality v of approximated concept subsumption* $C \sqsubseteq_{AC} D$ (cf. Definition 2) as:

$$v(C, D) = sim_{inf}(C', D) - (1 - sim_{inf}(C', C))$$

with the information-theoretic similarity of concepts C and D proposed by Lin [16]:

$$sim_{inf}(C, D) = 2 \cdot IC(maxdcs(C, D))/(IC(C) + IC(D)),$$

where $maxdcs(C, D) = argmax_{c \in dcs(C,D)}\{IC(c)\}$ is the direct common subsumer (dcs) of C and D in ontology O with maximum information content $IC(c)$. The *information content* of concept $C \in SAC(SR)$ is $IC(C) = -\log P(C)$, else $IC(C) := max_{D \in SAC(SR)}\{IC(D)\}$. We define the *probability of concept C* being used for semantic service annotation as the frequency of its occurrence in semantic signatures of services in service registry SR:

$$P(C) = \frac{1}{|IO_{SR}|} \cdot \sum_{S \in SR} |\{D \in S_{in} \cup S_{out} : D \sqsubseteq C\}|,$$

where IO_{SR} is the multiset of all parameters used in SR. Please note that we adapted the original notion introduced by Resnik [17] for our approach based on description logics to account for implicit subsumption relationships.$\diamond$

Example 3: The informative quality of $DebitAccount \sqsubseteq_{AC} CreditCard$ given in Example 2 is computed as follows:
$IC(DA) = -\log P(DA) = -\log 0.045 \approx 1.348$ is the information content of the original concept $DebitAccount$ and $IC(CC) = -\log P(CC) = -\log 0.075 \approx 1.125$ the information content of target concept $CreditCard$ accordingly. For the approximated concept $DebitAccount'$, it holds that $IC(DA') = -\log 0.035 \approx 1.456$, since $DA' \notin SAC(SR)$. $sim_{inf}(DA', CC) = \frac{2 \cdot 1.125}{1.456+1.125} \approx 0.872$ is the information gain from using the approximated concept instead of the original one and $sim_{inf}(DA', DA) = \frac{2 \cdot 1.348}{1.456+1.348} \approx 0.962$ the information loss of the approximation. The valuation then is computed as follows: $v(DA, CC) = 0.872 - (1 - 0.962) = 0.834$. $\circ$

For each service pair, depending on the computed type of their approximated signature concept subsumption relations, one can determine two hypotheses of approximated logical service signature matching: approximated logical plug-in and approximated logical subsumed-by. For both, the maximal informative quality is computed using bipartite concept graph matching.

Definition 4: *Approximated logical signature match.*
Let S, R be semantic service offer and request, $S_{in}, S_{out}, R_{in}, R_{out}$ multisets of their signature concepts and $BPG_{\sqsubseteq_{AC}}(\bar{C}, \bar{D})$ the concept assignment via bipartite graph matching as in Definition 1 but with approximated subsumption $\sqsubseteq_{AC}$ and informative quality of edge weights $v(C, D)$ for $C \in \bar{C}$, $D \in \bar{D}$; $BPG_{\sqsupseteq_{AC}}(\bar{C}, \bar{D})$ analogously with edge weights $v(D, C)$.
Approximated logical plug-in matching hypothesis $H_1(R, S)$ holds iff:

$$\begin{gathered} \forall I_S \in S_{in} : \exists I_R \in R_{in} : (I_S, I_R) \in BPG_{\sqsupseteq_{AC}}(S_{in}, R_{in}) \\ \wedge \forall O_R \in R_{out} : \exists O_S \in S_{out} : (O_S, O_R) \in BPG_{\sqsubseteq_{AC}}(S_{in}, R_{in}). \end{gathered}$$

Approximated logical subsumed-by matching hypothesis $H_2(R, S)$ holds iff:

$$\begin{gathered} \forall I_S \in S_{in} : \exists I_R \in R_{in} : (I_S, I_R) \in BPG_{\sqsupseteq_{AC}}(S_{in}, R_{in}) \\ \wedge \forall O_R \in R_{out} : \exists O_S \in S_{out} : (O_S, O_R) \in BPG_{\sqsupseteq_{AC}}(S_{in}, R_{in}). \end{gathered}$$

Informative quality $val_{(S,R)} : \{H_1, H_2\} \rightarrow [-1, 1]$ *of an approximated signature matching hypothesis* is the average of informative qualities of its respective approximated concept subsumptions:

$$\begin{gathered} val_{(S,R)}(H_1) = \\ \frac{1}{2 \cdot |S_{in}|} \cdot \sum_{(I_R, I_S) \in BPG_{\sqsupseteq_{AC}}(R_{in}, S_{in})} v(I_R, I_S) \\ + \frac{1}{2 \cdot |R_{out}|} \cdot \sum_{(O_S, O_R) \in BPG_{\sqsubseteq_{AC}}(S_{out}, R_{out})} v(O_S, O_R). \end{gathered}$$

$$val_{(S,R)}(H_2) =$$

$$\frac{1}{2\cdot|S_{in}|}\cdot\sum_{(I_R,I_S)\in BPG_{\sqsupseteq_{AC}}(R_{in},S_{in})} v(I_R,I_S)$$
$$+\frac{1}{2\cdot|R_{out}|}\cdot\sum_{(O_S,O_R)\in BPG_{\sqsupseteq_{AC}}(S_{out},R_{out})} v(O_S,O_R).$$

The *approximated logical signature matching degree* is the approximation hypothesis with maximum informative quality: MatchIO$_{ALogic}(S,R) := (H,v)$ with $H = argmax_{x\in\{H_1,H_2\}} val(x)$ and $v = val_{(S,R)}(H)$. Semantic relevance ranking of services S bases on MatchIO$_{ALogic}(S,R)[2]\in$[-1,1]. *Binary relevance classification by approximated logical matching*: MatchIO$_{ALogic}(S,R)^{*} = 1$ iff MatchIO$_{ALogic}(S,R)[2] \geq 0$, else MatchIO$_{ALogic}(R,S)^{*} = 0$. $\diamond$

Example 4: Consider Examples 1 − 3. The approximated logical signature match of S, R is computed as follows:
$BPG_{\sqsupseteq_{AC}}(R_{in},S_{in}) = \{(Book, Article), (DA, CC), (Person, Customer)\}$ is the assignment based on concept approximation and information-theoretic valuation for inputs and $BPG_{\sqsubseteq_{AC}}(S_{out},R_{out}) = \{(Invoice, Ack)\}$ the assignment for outputs accordingly, both assuming approximated signature matching hypothesis H_1. The informative quality valuation for H_1 is $val_{(S,R)}(H_1) = \frac{1}{2\cdot 3}\cdot(0.829+0.834+0.927)+\frac{1}{2\cdot 1}\cdot 0.895 = 0.879$. In this example, the same valuation holds for H_2, wich can be easily seen considering the fact that computation of $val_{S,R}(H_1)$ and $val_{S,R}(H_2)$ only differ regarding the outputs, for which only one assignment is possible and concepts already subsume. The overall matching result then is MatchIO$_{ALogic}(S,R) := (H_1, 0.851)$ $\circ$

Obviously, the approximated logical matching relation MatchIO$_{ALogic}(R,S)$ always exists, and we will show in the following, that its binary decision variant MatchIO$_{ALogic}(R,S)^{*}$ is redundant to its logical counterpart MatchIO$_{Logic}(R,S)$ with respect to positive service classification. That is, their true and false positives are the same, but not vice versa. This can be easily seen by considering that strict logical positives already provide parameter assignments based on subsumption relations and approximation is trivial in those cases (cf. Lemma 1).

Theorem 1: *Redundance of strict and approximated logical signature matching positives.* Given a service offer S and service request R described in terms of sets of satisfiable input and output concepts S_{in}, S_{out}, R_{in}, R_{out}, it holds that:

$$\text{MatchIO}_{Logic}(S,R) \neq \text{LFail} \Rightarrow \text{MatchIO}_{ALogic}(S,R)^{*}= 1.$$

In particular, it holds that all strict logical *true positives* are also approximated logical *true positives* and all strict logical *false positives* are also approximated logical *false positives*.

Proof: $LM(R,S)$ holds iff one of the following 4 cases applies: MatchIO$_{Logic}(R,S)$ $\in$ {Exact, Plug-in, Subsumes Subsumed-by}. For the *Subsumes* case the following holds: There exists a solution $BPG_{\sqsupseteq}(S_{in},R_{in})$ for which for every contained pair (I_R,I_S) it holds that $I_R \sqsubseteq I_S$. $CCP(I_R,I_S) = \langle\top, I_R\rangle$ (since $I_R \sqcap I_S \not\sqsubseteq \bot$),

$SAP(I_R, I_S) = \langle \top, \cdots, \top \rangle$ (Lemma 1). Hence, for each pair (I_R, I_S), the concept constructed for approximation is as follows: $I'_R = \sigma[\bar{H}, H](K^h) = \sigma[\bar{H}, \langle \top, \cdots, \top \rangle](I^h_R) = I_R$. Moreover, $sim_{inf}(I'_R, I_R) = (2 \cdot IC(I_R) = /(IC(I_R) + IC(I_R)) = 1$. From this, one can clearly see that $v(I_R, I_S) \geq 0$ for each such pair. The analoguous series of explanations can be applied for all output pairs (O_S, O_R). Therefore, $val_{(S,R)}(H_1) \geq 0$ (every summand is ≥ 0), which implies MatchIO$_{ALogic}(S, R)$*= 1 per last paragraph of Definition 4 (valuation for hypothesis $H = argmax_{x \in \{H_1, H_2\}} val(x)$ must be ≥ 0 because at least $H_1 \geq 0$). Cases MatchIO$_{Logic}(R, S)$ = Exact and MatchIO$_{Logic}(R, S)$ = Plug-in are special cases (with more restricted operators $\equiv$ and $\sqsupseteq_1$ respectively) of what has just been shown and the applied arguments also hold. For MatchIO$_{Logic}(R, S)$ = Subsumed-by, argumentation is equivalent wrt. to inputs, for outputs it can be easily shown analogously with approximation hypothesis H_2 instead of H_1.

This fact is used in iSeM to restrict its computation of approximated logical signature matches in the learning phase to cases of strict logical false negatives only and use the evidential coherence of the matching results to heuristically prune the feature space for precision (cf. Section 3.3.2).

3.1.2. Text and Structural Signature Matching

Non-logic-based approximated signature matching can be performed by means of text and structural similarity measurement. For iSeM, we adopted those of the matchmaker OWLS-MX3, since they have been experimentally shown to be most effective for this purpose [10]. For text matching of signature concepts in the classical vector space model, their unfoldings in the shared ontology are represented as weighted keyword vectors for token-based similarity measurement. Structural semantic similarity of concepts relies on their relative positioning in the subsumption graph, in particular on the shortest path via their direct common subsumer and its depth in the taxonomy [15].

Definition 5: *Approximated non-logic-based signature matching*
Let SR be the service registry of the matchmaker, I the text index of service signature concepts, O the shared ontology and $\overrightarrow{S_{in}}$, $\overrightarrow{S_{out}}$, $\overrightarrow{R_{in}}$, $\overrightarrow{R_{out}}$ the TFIDF weighted keyword vector of the conjunction of unfolded input or output concepts of S and R respectively. *Text similarity-based signature matching* is the average of the respective signature concept similarities:

$$\text{MatchIO}_{Text}(S, R) = \frac{1}{2} \cdot (sim_{text}(\overrightarrow{S_{in}}, \overrightarrow{R_{in}}) + sim_{text}(\overrightarrow{S_{out}}, \overrightarrow{R_{out}}))$$

with Tanimoto coefficient (alternatively Cosine similarity) $sim_{text}(\overrightarrow{C}, \overrightarrow{D}) \in [0, 1]$. *Structural semantic signature matching* is the averaged maximal structural similarity of their signature concepts:

$$\text{MatchIO}_{Struct}(S, R) = \frac{1}{2} \cdot (sim_{struct}(S_{in}, R_{in}) + sim_{struct}(S_{out}, R_{out}))$$

with

$$sim_{struct}(A,B) = \frac{1}{|A|}\sum_{a\in A} max\{sim_{csim}(a,b) : b \in B\} \in [0,1],$$

and structural concept similarity adopted from [15]:

$$sim_{csim}(C,D) = \begin{cases} e^{-\alpha l}\cdot \frac{e^{\beta h}-e^{-\beta h}}{e^{\beta h}+e^{-\beta h}} & ,C \neq D \\ 1 & ,C = D \end{cases},$$

with l shortest path via direct common subsumer between given concepts and h its depth in O, $\alpha = 0.2$ and $\beta = 0.6$ weighting parameters manually adjusted to structural features of ontology O based on results of [15]. ⋄

Example 5: Applied to Example 1, we obtain a high score for text-based signature matching MatchIO$_{text}(S,R) = 0.71$ which correctly accounts for semantic relevance of S to R, hence avoids the strict logical false negative. The same holds for the structural semantic matching MatchIO$_{struct}(S,R) = 0.69$. For example, text and structural similarities of the strict logically disjoint input concept siblings *DebitAccount* and *CreditCard* are high ($sim_{text}(DA,CC)$= 0.94, $sim_{csim}(DA,CC) = 0.63$) which indicates their semantic proximity. Please note, that we do not apply a threshold value do determine relevance but perform semantic relevance learning (cf. Section 3.3). However, matching pairs tend to get higher results for MatchIO$_{text}$ and MatchIO$_{struct}$ than irrelevant pairs. ◦

While text matching of signatures may avoid strict logical matching failures, structural semantic matching may also compensate for text matching failures, in particular when mere is-a ontologies with inclusion axioms only are used for semantic annotation of service signatures. For reasons of space limitation, we refer to [10] for more details and examples.

3.2. Stateless Logical Specification Matching

As mentioned above, semantic signatures of services do not cover functional service semantics usually encoded in terms of logical service preconditions and effects. This may cause signature matching only to fail, for example if signatures are equivalent for a book selling service offer and a book borrowing request. Though semantic service descriptions rarely contain such specifications in practice [14], we equipped the implemented iSeM matchmaker with the most prominent PE-matching filter adopted from software retrieval: logical specification plug-in matching.

Definition 6: *Stateless, logical specification plug-in matching.*
Let (S,R) be services with preconditions (P_R,P_S) and effects (E_R,E_S) defined in SWRL. Service S *logically specification-plugin matches* R:

$$\text{MatchPE}(S,R) \text{ iff } \models (P_R \Rightarrow P_S) \wedge (E_S \Rightarrow E_R).$$

Stateless checking of MatchPE(S, R) in iSeM 1.0 is adopted from LARKS [18]: Preconditions and effects specified as SWRL rules are translated into PROLOG as in [19] and then used to compute the required logical implications by means of θ-subsumption checking stateless, that is without any instances (ABox), as given in [20]:

$$(\forall p_S \in P_S : \exists p_R \in P_R : p_R \leq_\theta p_S) \Rightarrow (P_R \Rightarrow P_S)$$
$$(\forall e_R \in E_R : \exists e_S \in E_S : e_S \leq_\theta e_R) \Rightarrow (E_S \Rightarrow E_R).$$

A clause C θ-subsumes D, written $C \leq_\theta D$, iff there exists a substitution θ such that $C\theta \subseteq D$ holds; θ-subsumption is an incomplete, decidable consequence relation [6]. $\diamond$

Example 6: If applied to Example 1, this PE-matching filter succeeds, hence avoids the respective false negative of strict logical signature matching only. Further, consider a service pair (S, R') having the identical or strict logically equivalent semantic signatures as (S, R) given in Example 1 - but with the requested effect of R' to only register a book at a given local index such that service S is irrelevant to R': The false positive S of (strict or approximated) logical signature matching only can be avoided by an additional specification plug-in matching filter, which, in this case, would correctly fail. $\circ$

3.3. Off-Line Service Relevance Learning

In order to find the best combination of its different matching filters for most precise service selection, iSeM learns their optimal weighted aggregation by using a support vector machine (SVM) approach. In particular, the underlying feature space is pruned by evidential coherence-based weighting of approximated against strict logical signature matching results over the given training set to improve precision.

3.3.1. Overview: Learning and Selection

The training set TS is a subset (5%) drawn uniformly at random from the service test collection OWLS-TC4. It contains user-rated service pairs (S, R) each of which is equipped with a 10-dimensional matching feature vector x_i for positive and/or negative service relevance samples $(x_i, y_i) \in X \times \{1, -1\}$ in the possibly non-linearly separable[7] feature space $X = \{0,1\}^5 \times [-1,1]^2 \times [0,1] \times [0,1] \times \{0,1\}$. The different matching results for (S, R) are encoded as follows: x[1] ... x[5]$\in \{0,1\}^5$ for MatchIO$_{Logic}(R, S)$ in decreasing order; x[6] $= val_{(S,R)}(H_1)$ and x[7] $= val_{(S,R)}(H_2) \in [-1,1]$ for MatchIO$_{ALogic}(R, S)$; x[8]$\in [0,1]$ for MatchIO$_{Text}(R, S)$; x[9]$\in [0,1]$ for MatchIO$_{Struct}(R, S)$; and x[10]$\in \{0,1\}$ for MatchPE(R, S). For example: $x = (0, 0, 0, 0, 1, 0.85, 0, 0.4, 0.6, 1)$ encodes a strict logical fail but approximated logical plugin with informative quality of 0.85, text (structural) match of 0.4 (0.6) and plugin specification match.

[7]E.g. feature space for OWLS-TC4 is non-linearly separable

The SVM-based classification learning problem of iSeM then is to find a separating hyperplane h in X such that for all samples $(x, y) \in TS$ for (S, R) with minimal distances (these particular samples are also called support vectors) to h these distances are maximal. It is defined as follows:

$$\text{minimize in } w,b,\zeta\text{: } \frac{1}{2}w^T w + C\sum_{i=1}^{N}\zeta_i$$

$$\text{subject to } \forall 1 \leq i \leq N : y_i(w^T\phi(x_i) + b) \geq 1 - \zeta_i, \zeta_i \geq 0,$$

where w and b define the optimally separating hyperplane as the set of points satisfying $w^T\phi(x) + b = 0$. Furthermore, w is the *normal vector* which specifies the orientation of the plane, b is called *bias* and indicates the offset of the hyperplane from the origin of the feature space X. The error term $C\sum_{i=1}^{N}\zeta_i$ is introduced to allow for outliers in a non-linear separable training set, where the error penalty parameter C must be specified beforehand. The predefined function ϕ maps features into a higher, possibly infinitely dimensional space in which the SVM finds a hyperplane that allows a classification of non-linear separable data (more precise with respect to the original dimension of X)[8].

Since $w = \sum_{i=1}^{N} y_i\alpha_i\phi(x_i)$ is a linear combination of training sample feature vectors the dual formulation of the SVM classification problem that is actually solved by OWLS-MX3 is as follows:

$$\text{maximize in } \alpha\text{: } \frac{1}{2}\sum_{i,j=1}^{N} y_i y_j \alpha_i \alpha_j K(x_i, x_j) - \sum_{i=1}^{N}\alpha_i$$

$$\text{subject to } \sum_{i=1}^{N} y_i\alpha_i = 0, \forall 1 \leq i \leq N : 0 \leq \alpha_i \leq C.$$

The *kernel* function $K(x_i, x_j) = \phi(x_i)^T\phi(x_j)$ implicitly defines ϕ in the scalar product, while problem is solved by finding a set of Lagrange multipliers α_i representing the hyperplane for which training samples x_i with $\alpha_i \neq 0$ are called support vectors (of the hyperplane). As kernel, the RBF (Radial Basis Function) $K(x_i, x_j) = e^{-\gamma\|x_i - x_j\|^2}$ is used as suggested in [7]. Unlike polynomial kernels, it only introduces a single parameter γ which keeps the complexity of model selection low. Besides, for specific parameter settings it can behave like a linear or sigmoid kernel.

The searching of an optimal SVM paramter setting (C, γ) with respect to average classification accuracy has been achieved by means of grid search and 6-folded cross-validation. Binary classification of samples $x \in X$ for service pair (S, R) with the above mentioned parameters is defined as follows: $d(x) = \sum_{i=1}^{N} y_i\alpha_i K(x_i, x) + b$ with bias b satisfying the Karush-Kuhn-Tucker

[8]The fraction $\frac{1}{2}$ is introduced for computational reasons only, and does not affect the classification result.

condition (KKT)[2], such that S is classified as relevant iff $d(x) > 0$. Please note, that w is not a direct output of the dual optimization but computed using the objective value o of the dual optimization and the coefficients α_i based on the relation between the primary and dual problem: $||w||^2 = w^T w = \sum_{i,j=1}^{N} y_i y_j \alpha_i \alpha_j K(x_i, x_j) = 2 \cdot (o + \sum_{i=1}^{N} \alpha_i)$. Since we are not only interested in binary classification but also want to compute a service ranking, the distance of the sample to the hyperplane is also computed: $dist(x) = \frac{d(x)}{|w|}$. The matching function then returns a tuple of classification result and distance using the following definition: MatchIOPE$(S, R) = (d(x) > 0,\, dist(x))$.

3.3.2. Evidential Coherence-Based Feature Space Pruning

To improve the performance of the binary SVM-based relevance classier to be learned by iSeM, iSeM exploits information available from the given training set TS to prune the feature space X based on the classification results of strict vs. approximated logical signature matching. Due to redundance of both logical matching types for (true and false) positive classification (cf. Theorem 1), it restricts the pruning of feature vectors $x \in X$ to cases of strict logical matching failures ($MatchIO_{ALogic}(R, S) = LFail$). The respective set $Ev = \{(x, y) : x[5] = 1\}$ of classification events is partitioned with respect to binary classification results of approximated logical matching (MatchIO$_{ALogic}(R, S)^*$) for these cases as follows:

$$E_1 = \{(x, y) \in Ev : y = 1 \wedge (x[6] > 0 \vee x[7] > 0)\},$$

$$E_2 = \{(x, y) \in Ev : y = -1 \wedge x[6] \leq 0 \wedge x[7] \leq 0\},$$

$$E_3 = \{(x, y) \in Ev : y = 1 \wedge x[6] \leq 0 \wedge x[7] \leq 0\},$$

$$E_4 = \{(x, y) \in Ev : y = -1 \wedge (x[6] > 0 \vee x[7] > 0)\}.$$

For example, E_1 denotes all relevant samples $(x, y) \in Ev$ classified correctly as (true) positives by $MatchIO_{ALogic}$ while E_2 contains all irrelevant samples $(x, y) \in Ev$ classified correctly as (true) negatives by $MatchIO_{ALogic}$. The set E_3 contains wrong negative classifications of approximated matching, hence is redundant to its strict logical counterpart and deleted from the respectively pruned feature space for learning. In contrast, E_4 contains those cases, where approximated logic-based matching itself classifies as false positives contrary to strict logic-based matching.

Inspired by the work of Glass [5], the feature space X is pruned further by modification of logical matching results of feature vectors $x \in X$ of samples in E_1, E_2 or E_4 based on evidential coherence-based weighting of approximated matching results as follows:

$$E_1, x[6] \geq x[7] \mapsto x[5] := 0, x[6] := w_1 \cdot x[6], x[7] := 0,$$

$$E_1, x[6] < x[7] \mapsto x[5] := 0, x[6] := 0, x[7] := w_2 \cdot x[7],$$

$$E_2, x[6] \geq x[7] \mapsto x[6] := w_3 \cdot x[6], x[7] := 0,$$

$$E_2, x[6] < x[7] \mapsto x[6] := 0, x[7] := w_4 \cdot x[7],$$
$$E_4, x[6] \geq x[7] \mapsto x[6] := (1 - w_1) \cdot x[6], x[7] := 0,$$
$$E_4, x[6] < x[7] \mapsto x[6] := 0, x[7] := (1 - w_2) \cdot x[7].$$

In case of true positive approximated logical matching, the encoded strict logical misclassification in $x \in X$ is displaced ($x[5] = 0$); in any case, the better approximation (H_1 or H_2) is weighted with the evidential coherence value (one of $w_1 \ldots w_4$) of one of the following hypotheses (A1, A2) of relevance explanation: (A1) $MatchIO_{ALogic}$ is a correct explanation of semantic *relevance* (avoids logical false negatives), and (A2) $MatchIO_{ALogic}$ is a correct explanation for semantic *irrelevance* (avoids introduction of false positives). For events in E_4, the coherence for the contrary of hypothesis A1 ($MatchIO_{ALogic}$ is *not* a correct explanation of semantic relevance) is used for weighting corresponding features to alleviate support for false positives of approximate logic-based matching in the aggregation strategy learning step.

Which of both hypotheses of semantic relevance explanation is best with respect to a given test collection? Following [5], iSeM determines the quality of an explanation by measuring the impact of evidence E on the probability of explanation H (with coherence or confirmation measures) rather than measuring its posterior probability with Bayes. In other words, it determines the most plausible explanation H instead of the most probable measured in terms of its coherence with evidence E over given training set. While hypothesis A1 (A2) is represented by special case set H_i^+ (H_i^-), the set E^+ (E^-) provides cases of observed evidence for relevance (irrelevance) in the test collection. The coherence overlap measure $Co(H, E) = \frac{P(H \cap E)}{P(H \cup E)}$ performed best in practice [5], and is used by iSeM to compute the weights of approximated logical signature matching results for respective feature space pruning: $w_1 = Co(H_1^+, E^+)$, $w_2 = Co(H_2^+, E^+)$, $w_3 = Co(H_1^-, E^-)$ and $w_4 = Co(H_2^-, E^-)$.

Example 7: Consider training set TS with $|Ev| = 20$, $|E_1| = 10$ and $|E_4| = 1$. E_1 contains 8 events (cases) of approximated plug-in matching ($x[6] \geq x[7]$), the only event in E_4 is also an approximated plug-in match. Required posterior probabilities for $w_1 = Co(H_1^+, E^+)$ are computed as follows: $P(H_1^+) = \frac{|\{x \in E_1 \cup E_4 : x[6] \geq x[7]\}|}{|Ev|} = \frac{9}{20}$,
$P(E^+) = \frac{|E_1 \cup E_3|}{|Ev|} = \frac{14}{20}$, $P(H_1^+|E^+) = \frac{|\{x \in E_1 : x[6] \geq x[7]\}|}{|E_1 \cup E_3|} = \frac{8}{14}$. The resulting evidential coherence-based weight of approximated logical matching is: $Co(H_1^+, E^+)$ $= \frac{P(E^+) \cdot P(H_1^+|E^+)}{P(E^+) + P(H_1^+) - P(H_1^+ \cap E^+)} \approx 0.5333$. That is, the coherence value of the hypothesis of approximation H_1 (represented by feature $x[6]$) being a correct explanation for semantic relevance (A1) is $w_1 = 0.5333$. Analogously, to compute the weights w_2, w_3 and w_4, the following posterior probabilities can be computed by observing the set of all relevant events Ev: $P(H_1^-) = \frac{|\{x \in E_2 \cup E_3 : x[6] \geq x[7]\}|}{|Ev|}$,
$P(H_2^+) = \frac{|\{x \in E_1 \cup E_4 : x[6] < x[7]\}|}{|Ev|}$, $P(H_2^-) = \frac{|\{x \in E_2 \cup E_3 : x[6] < x[7]\}|}{|Ev|}$, $P(E^-) = \frac{|E_2 \cup E_4|}{|Ev|}$,
$P(H_1^-|E^-) = \frac{|\{x \in E_2 : x[6] \geq x[7]\}|}{|E_2 \cup E_4|}$, $P(H_2^+|E^+) = \frac{|\{x \in E_1 : x[6] < x[7]\}|}{|E_1 \cup E_3|}$, $P(H_2^-|E^-) =$

$\frac{|\{x \in E_2 : x[6] < x[7]\}|}{|E_2 \cup E_4|}$. To compute the remaining coherence values, the probabilities have to be inserted into the following formulas: $w_2 = \frac{P(E^+) \cdot P(H_2^+|E^+)}{P(E^+)+P(H_2^+)-P(H_2^+ \cap E^+)}$, $w_3 = \frac{P(E^-) \cdot P(H_1^-|E^-)}{P(E^-)+P(H_1^-)-P(H_1^- \cap E^-)}$, $w_4 = \frac{P(E^-) \cdot P(H_2^-|E^-)}{P(E^-)+P(H_2^-)-P(H_2^- \cap E^-)}$. $\circ$

4. Implementation

In the following, details regarding the implementation of the current version 1.1 of iSeM are given. Selected algorithms are presented and an architectural overview based on the *Semantic Service MatchMaker* (S2M2) framework is provided, followed by details on the flexible model-driven approach used to define matching filters. iSeM 1.1 is implemented in Java and publicly available at http://www.semwebcentral.org/projects/isem/.

4.1. Algorithms

Algorithm 1 (plase note that all algorithms described here are located in the Appendix to not disrupt the text flow) shows the training phase of iSeM as described formally in the previous sections. After the feature vectors have been computed for each request/candidate pair (R, S) with relevance y of training set TS (lines 2–5), the event sets as described in Section 3.3.2 are filtered (lines 10–13). After that, the adaptation of feature vectors based on coherence computations for the previously described cases is performed (lines 15–31) and the SVM is trained.

Algorithm 2 technically describes the approximate service matching procedure in detail, which is used during the training phase[9]. At first, the explanations including local valuation on concept level for all relevant concept combinations are computed (lines 2–18), which then serve as basis for possible approximate matching variants H_1 and H_2. The next lines (19–21) then describe the use of bipartite graph matching, which finds the best assignment for the signature components. After that, the local valuations of both variants on service level are computed (lines 22-23). Finally, the two approximation results are returned with their valuation. Please note, that the result tuple of this function may be used to compute MatchIO$_{ALogic}$*; however, this is not required for the training and matching process using the SVM-based result aggregation.

Approximation on concept level is described in Algorithm 3 and follows the formal definitions given in Section 3.1.1. Given a concept C and target concept D, the contraction problem is solved first to establish compatibility ($K \sqcap D \not\sqsubseteq \bot$), followed by structural concept abduction to create a (near-)optimal (wrt. to minimization of applied changes) subsumed concept definition. Both results are taken into account for the constructed concept C', which is then returned together with its information-theoretic valuation v.

[9]It is also used for matching, which is not covered here, because it is quite similar to training in terms of feature computation and only differs in using the SVM for prediction instead of training.

4.2. S2M2 Framework

The *S2M2* (Semantic Service MatchMaker) framework is designed to support various aspects of development of a semantic matchmaker while being independent of specific formalisms for service description and semantic annotation as well as inference mechanisms. To accomplish this, adequate interfaces for *extraction* of relevant service information, *matching* of service candidates with a given request and creation of *rankings* based on results of the matching process are provided.

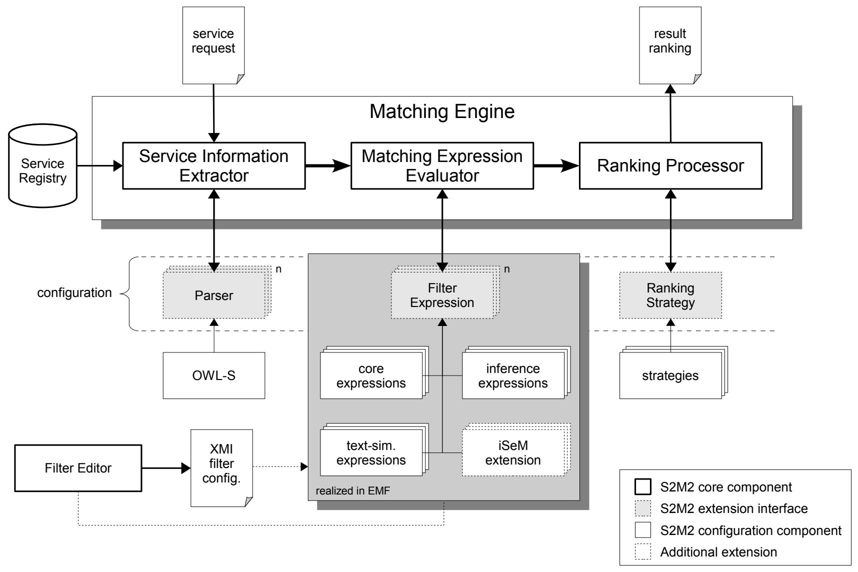


Figure 4: S2M2 architecture overview.

Figure 4 provides a broad overview of the S2M2 framework. Given a set of service offer candidates stored in a service registry and a request, service information is parsed and mapped to an internal representation as preparation step of the matching process inside the *Service Information Extractor* core component. A parser interface offers capabilities for extension to other formats not provided by default. The step of mapping functional or non-functional properties enables the implementation of matchmakers independent of the used formalism to describe services. For example, a matchmaker developer may intend to map OWL-S input parameters and SAWSDL model references of input message parts to a unified view on service inputs in the internal representation.

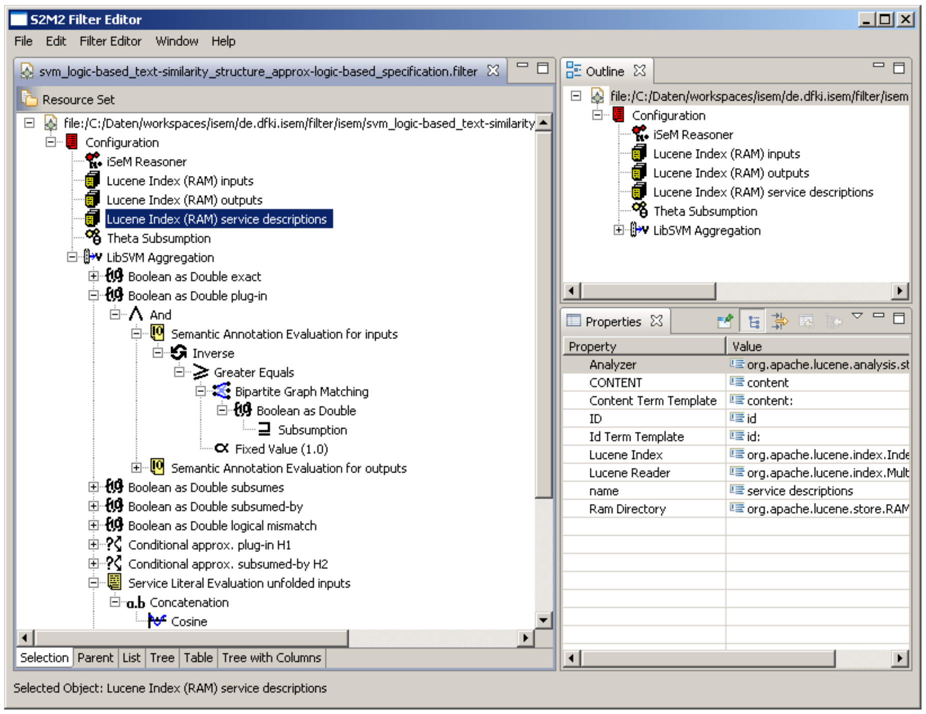
Each service offer/request pair is then matched inside the *Matching Expression Evaluator* according to a set of matching feature *expressions* (filter definitions). For this, a very flexible approach based on the Eclipse Modeling Framework (EMF) has been adopted, which describes filter expressions in terms of a meta-model based on a generic expression interface definition. This allows (a) to easily extend the set of expressions provided with S2M2 by creating new packages extending existing features or the generic interface as has been done for example in iSeM 1.1, (b) to automatically generate code for a graphical editor to create and edit filter definition instances for S2M2 core functionality as well as additional packages and (c) to make use of the persistence functionality provided by EMF in terms of the XMI (XML Metadata Interchange) format, which is interpreted by the generic matching engine implementation of S2M2 and thus allows for straight-forward integration of own filter definitions in the matchmaker. Figure 5 shows the generated stand-alone tree-editor application based on RCP (Rich Client Platform), which is contained in the iSeM distribution. Based on EMF, other editors can be implemented, such as for example a syntax-highlighting text editor with auto-completion using XText. The set of matching expressions provided with S2M2 range from logical connectors like *and*, *or* to more complex numerical operations and specific similarity computations on concept- or textual level. Besides the *core* package, S2M2 provides a package for *inference* mechanisms, which is further sub-divided into description-logic-based operations and reasoners and theorem-proving, as well as a *text-similarity* package providing sample implementations for VSM-based (vector space model) similarity computations. Some of them are directly used for implementing iSeM 1.1 as presented in previous sections, while some functionality has been added in terms of a new package, which provides specification and implementation of approximate matching and SVM-based feature aggregation among others. An exhaustive listing of supported expressions of S2M2 and the iSeM extension can be found in the Appendix in Table B.1.

After evaluation of matching expressions, the result ranking is prepared by the *Ranking Processor*. The ordering is given by a ranking strategy specification that incorporates the results of the previously performed evaluation. Currently, different ranking production strategies have to be implemented in Java directly given a set of interfaces. Moreover, the generic matching engine factory implementation of S2M2 is currently hard-coded to allow for decreasing ranking in multiple feature dimensions with decreasing priority only. However, for a first stand-alone version of S2M2, it is planned to enable complete matchmaker configurations using EMF as described for filter definitions, which includes ranking strategies and service information extraction.

5. Performance Analysis

The performance evaluation of iSeM (namely iSeM 1.1) has been conducted using the service retrieval test collection OWLS-TC4[10], which consists of 1083 service offers in OWL-S 1.1 and 42 queries including binary and graded relevance assignments from nine different application domains. Since version 4, it also includes definitions of preconditions and effects for a subset of services (180

[10]Publicly available at http://www.semwebcentral.org/projects/owls-tc


S2M2 Filter Editor
File Edit Filter Editor Window Help
svm_logic-based_text-similarity_structure_approx-logic-based_specification.filter
Resource Set
Outline
Properties
Selection Parent List Tree Table Tree with Columns

```
file:/C:/Daten/workspaces/isem/de.dfki.isem/filter/isem/svm_logic-based_text-similarity
  Configuration
    iSeM Reasoner
    Lucene Index (RAM) inputs
    Lucene Index (RAM) outputs
    Lucene Index (RAM) service descriptions
    Theta Subsumption
  LibSVM Aggregation
    Boolean as Double exact
    Boolean as Double plug-in
      And
        Semantic Annotation Evaluation for inputs
          Inverse
            Greater Equals
              Bipartite Graph Matching
                Boolean as Double
                  Subsumption
              Fixed Value (1.0)
        Semantic Annotation Evaluation for outputs
    Boolean as Double subsumes
    Boolean as Double subsumed-by
    Boolean as Double logical mismatch
    Conditional approx. plug-in H1
    Conditional approx. subsumed-by H2
    Service Literal Evaluation unfolded inputs
      Concatenation
        Cosine
```

```
file:/C:/Daten/workspaces/isem/de.dfki.isem/filter/isem
  Configuration
    iSeM Reasoner
    Lucene Index (RAM) inputs
    Lucene Index (RAM) outputs
    Lucene Index (RAM) service descriptions
    Theta Subsumption
    LibSVM Aggregation
```

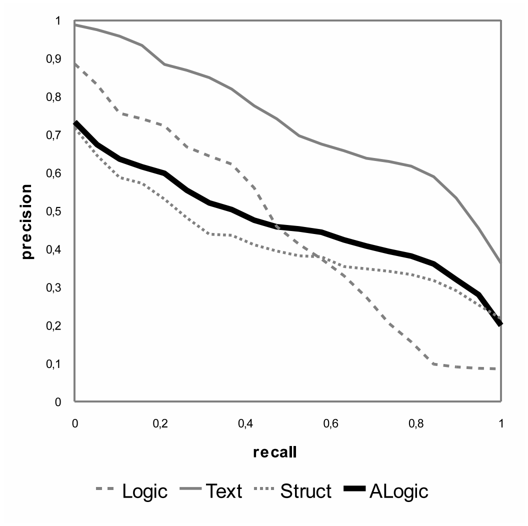
Property	Value
Analyzer	org.apache.lucene.analysis.st
CONTENT	content
Content Term Template	content:
ID	id
Id Term Template	id:
Lucene Index	org.apache.lucene.index.Inde
Lucene Reader	org.apache.lucene.index.Mult
name	service descriptions
Ram Directory	org.apache.lucene.store.RAM

Selected Object: Lucene Index (RAM) service descriptions

Figure 5: EMF-based S2M2 filter editor.

offers, 18 queries) which enables us to apply the full-fledged iSeM including specification matching as presented above. However, since only a subset of definitions actually contain PE and therefore the used θ-subsumption algorithm trivially yields MatchPE(S, R) (because $\top \Rightarrow \top$) for a larger portion of the test collection, we added another binary dimension to feature space X of our learning algorithm that allows iSeM to identify those cases: $x[11] \in \{0, 1\}$ with $x[11] = 1$ iff request R contains a non-trivial precondition or effect ($P_R \neq \top \vee E_R \neq \top$). Moreover, service parameter annotations in OWLS-TC4 are not restricted to the DL subset SH, which is a requirement for the current version of the approximated logic-based signature matching algorithm used in iSeM 1.0. To overcome this, we implemented a trivial and non-optimal approximation for cases where the algorithm is not applicable: given concepts C and D with one of them not in SH, the result of concept contraction is $CCP(C, D) = \langle C, \top \rangle$ and concept abduction yields $C' = SAP(C, D) = D$, i.e. every aspect of the original definition of C is neglected to derive the most obvious solution such that $C' \sqsubseteq D$. However, even considering this weak approximation for those cases, we observed a significant improvement with respect to ranking precision as we will show in the following. For evaluation, we used the public tool SME2 v2.1[11] on a WinXP SP3 32bit machine with Intel Core2Duo T9600 (2,8GHz) processor and 3GB RAM. We measured macro-averaged precision at 20 equidistant recall levels (MARP graph), averaged *average precision* and average query response time.

5.1. Evaluation Results



		AvgP	AvgQRT
Basic Variants	Logic	.7	0.07s
	Text	.8	0.86s
	Struct	.75	0.18s
	ALogic	.69	0.44s
SVM	Logic, Text, Struct	.83	1.84s
	All w/o pruning	.88	2.28s
	All w. pruning	**.92**	**2.34s**

Figure 6: Macro-averaged recall/precision (MARP), average precision (AvgP) and average query response time (AvgQRT) of basic and adaptive signature matching by iSeM 1.0.

In summary, the evaluation results shown in Figure 6 reveal that (a) approximated logical matching via abduction and informative quality performs similarly to its strict logical counterpart with respect to average precision, but differs to a large extent at fixed recall levels, (b) this kind of matching performs close to but still worse than its non-logic-based approximated counterparts (text and structural matching), and (c) adaptive hybrid combination outperforms all other variants in terms of precision.

As expected, due to the redundance of strict and approximated logical signature matching positives (cf. Theorem 1), approximated logic-based matching alone was not able to outperform its non-logic-based counterparts but performed close to strict logical matching. As already hinted in statement (a) before, the rankings of approximated logic-based and strict logic-based matching vary to a large extent, which provides evidence that both variants are mutually independent to a satisfactory degree. This fact has been validated using a Friedman test conducted on average precisions per query, which revealed that there is no

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

significant difference at 5% level ($p \approx 0.088$), i.e. neither was able to outperform the other for a majority of queries. As has already been shown in context of OWLS-MX3 (cf. [10]), this also holds for the other basic matching variants leading to the conclusion that each of the basic signature matching filters of iSeM contributes to an overall increase of performance for some cases of strict logical false classification, i.e. none of the tested variants outperformed the others for almost all service requests in the test collection.

The adaptive hybrid aggregation of all matching filters as done by iSeM (cf. Section 3.3) significantly increases the retrieval performance compared to that of its individual matching filters. While the combination of strict logic-based, text similarity and structure matching already yields good results as expected from previous observations with OWLS-MX3, the additional consideration of approximated logical matching and stateless logical specification matching (in the learning process) performs best. Moreover, the comparative performance analysis conducted for the annual S3 contest in 2010[12] resulted in iSeM being among the top contestants regarding precision-based measures. In fact, it won the OWL-S track with the best average precision (0.92) ever measured in this contest so far. For our specific use case, the proposed feature space pruning for relevance learning performed best, but arguably not in general [1].

Regarding query response times, the adaptive hybrid aggregated variants performed significantly slower than the basic matching filters (cf. Figure 6), which is not surprising considering the fact that each of the presented filters has to be evaluated per offer/query pair in case of hybrid matching. For the basic variants, it is worth mentioning that the approximated logic-based matching performs significantly slower than strict logic-based matching, but performance is still reasonable looking at the benefit with respect to ranking precision of the full-fledged iSeM system.

5.2. Discussion

As the evaluation results show, hybrid matchmakers may benefit from integration of the presented approximated logic-based matching approach and thus increase ranking precision significantly. As motivated in Section 2, the reason for this is the improved *avoidance of strict logical false negatives* by additionally considering approximated logic-based matching, which shows sufficient mutual independence of all other approaches presented here.

The proposed feature space model as basis for the applied machine-leaning approach allows for easy integration of arbitrary matching filters and similarity functions and facilitates straightforward matchmaker tool configuration without manually setting up weights for a fixed result aggregation. This approach implicitly solves the question of how to find the *best combination of semantic matching filters* to achieve high precision on average. Moreover, the application

[12]S3 2010 summary report is available at http://www-ags.dfki.uni-sb.de/ klusch/s3/html/2010.html

of SVM using RBF-kernel nicely fits the non-linear separable nature of the feature space in context of semantic service retrieval on the Web as presented in this chapter. For example, in OWLS-TC4 it is not possible to strictly perform linearly-weighted separation based on the various features into matching true positives and true negatives.

Failure of signature matching only has been minimized satisfactorily using the presented stateless specification matching step whenever preconditions or effects were available. Besides the increased overall average precision achieved by the fully-fledged iSeM matchmaker, this has also been shown exemplarily in the S3 2010 summary report. However, as already stated before, OWLS-TC4 only consists of few services and request documents with full IOPE profile. A majority of examples does not provide preconditions and effects yet, which alleviates the overall benefit in precision for the experimental performance analysis to some degree.

On the other hand, as shown above, the presented approach results in a linear increase of query response times, because every filtering and similarity computation is performed for each service request and offer pair, even in cases where the adapted aggregation strategy gives very low weights on some features for certain circumstances. Moreover, the adaption is done such that precision is optimized on average for the whole training set and thus is rather coarse-grained. Even though this turned out not to be a problem for OWLS-TC4, there may be cases where some basic approaches (features) are better than others given certain contexts and vice versa. For example, the presented approximated logic-based matching may perform better in domains described in terms of rich-detailed ontologies like medical domains, where it may prove inferior for coarse-grained domain descriptions. Finally, the presented adaption strategy is strictly off-line with a distinct training and exploitation phase. Adapting iSeM over time, for example while collecting more and more user feedback, would require to drop the current aggregation function and train a new one from scratch after some period. This could be controlled by a threshold value for the resulting precision based on the retrieved user feedback, but training the SVM itself is a costly process.

6. Related Work

There exists a large body of work in the area of semantic service matchmaking, and the field has been surveyed extensively for example in [13]. However, to the best of our knowledge, iSeM is the first fully-fledged adaptive, hybrid semantic service IOPE matchmaker. Therefore, we will focus on the adaptivity aspect of hybrid matching and approximative logic-based filtering in the following.

The strict logical and the non-logic-based semantic signature matching filters as well as the SVM-based learning process of iSeM are adopted from the adaptive signature matchmaker OWLS-MX3 [10]. However, unlike iSeM, OWLS-MX3 neither performs approximated logical signature matching, nor PE-matching, nor is its adaptive process applicable to IOPE matching results and the feature space is not evidentially pruned. The same holds for the adaptive hybrid semantic signature matchmaker SAWSDL-MX2[12]. Besides, SAWSDL-MX2 performs structural matching on the WSDL grounding level only which significantly differs from the semantic structural matching performed by iSeM. The use of abduction for approximated logical signature matching is inspired by DiNoia et al.[4, 3]. However, their non-adaptive matchmaker MaMaS performs abduction for approximated matching of monolithic service concept descriptions in SH, while iSeM exploits it for significantly different approximated structured signature matching and its use for learning. Besides, MaMaS has not been evaluated yet. OWL-S iMatcher [8] performs hybrid Semantic Web service matchmaking based on a flexible approach for user-defined matching strategies named iSPARQL. It also adopts a wide range of machine-learning aggregation strategies and presented evaluation results were promising. In contrast to iSeM, it does not consider approximated logic-based matching and specification matching in any way.

7. Conclusion

We presented the first adaptive, hybrid and full semantic service profile (IOPE) matchmaker that, in particular, performs approximated logical reasoning and respectively evidential coherence-based pruning of learning space to improve precision over strict logical matching. The evaluation results for iSeM revealed, among other things, that its adaptive hybrid combination with non-logic-based approximated signature matching improves on each of them individually. Moreover, it proved to be among the top contestants regarding precision-based measures and won the OWL-S track of the 2010 edition of the annual S3 contest.

The presented approach for adaptive combination of matching variants based on a feature space representation provides a flexible basis for detailed experiments with different machine-learning approaches. Besides well-known off-line algorithms, we intend to also implement *on-line* approaches to add flexibility at runtime as described above. Moreover, the approximated logical matching results of iSeM can also be exploited for explanation-based interaction with the user during the selection process. Though in its initially implemented version iSeM is non-obtrusive in this respect, such interaction together with extending the abductive approximated reasoning to OWL2-DL annotations is subject to future work.

References

[1] Blum, A. L.; Langley P. (1997): Selection of Relevant Features and Examples in Machine Learning. *Artificial Intelligence*, 97:245-271.

[2] Chang, CC.; Lin, CJ. (2001): LIBSVM: a library for support vector machines. Available at http://www.csie.ntu.edu.tw/ cjlin/libsvm

[3] Colucci, S; et al. (2005): Concept Abduction and Contraction for Semantic-based Discovery of Matches and Negotiation Spaces in an E-Marketplace. *Electronic Commerce Research and Applications*, 4(4)

[4] Di Noia, T.; Di Sciascio, E.; Donini F. M. (2009): A Tableaux-based Calculus for Abduction in Expressive Description Logics: Preliminary Results. Proceedings of 22nd International Workshop on Description Logics (DL).

[5] Glass, D. H. (2009): Inference to the Best Explanation: A comparison of approaches. Proceedings of the AISB 2009 Convention, Edinburgh, UK. www.aisb.org.uk/convention/aisb09/Proceedings/

[6] Idestam-Almquist, P. (1995): Generalization of Clauses under Implication. *Artificial Intelligence Research*, 3:467-489.

[7] Keerthi, SS.; Lin, CJ. (2003): Asymptotic behaviour of support vector machines with Gaussian kernel. Journal on Neural Computation, Vol. 15, Issue 7, 1667–1689.

[8] Kiefer, C.; Bernstein, A. (2008): The Creation and Evaluation of iSPARQL Strategies for Matchmaking. Proceedings of the 5th European Semantic Web Conference (ESWC), Springer.

[9] Klusch, M.; Fries, B.; Sycara, K. (2009): OWLS-MX: A Hybrid Semantic Web Service Matchmaker for OWL-S Services. *Web Semantics*, 7(2), Elsevier.

[10] Klusch, M.; Kapahnke, P. (2009): OWLS-MX3: An Adaptive Hybrid Semantic Service Matchmaker for OWL-S. Proceedings of 3rd International Workshop on Semantic Matchmaking and Resource Retrieval (SMR2), USA; CEUR 525.

[11] Klusch, M.; Kapahnke, P. (2010): iSeM: Approximated Reasoning for Adaptive Hybrid Selection of Semantic Services. Proceedings of 4th IEEE International Conference on Semantic Computing (ICSC), USA.

[12] Klusch, M.; Kapahnke, P.; Zinnikus, I. (2009): Hybrid Adaptive Web Service Selection with SAWSDL-MX and WSDL Analyzer. Proceedings of 6th European Semantic Web Conference (ESWC), Heraklion, Greece, IOS Press.

[13] Klusch, M. (2008): Semantic Web Service Coordination. In: M. Schumacher, H. Helin, H. Schuldt (Eds.) CASCOM - Intelligent Service Coordination in the Semantic Web. Chapter 4. Birkhäuser Verlag, Springer.

[14] Klusch, M.; Xing, Z. (2008): Deployed Semantic Services for the Common User of the Web: A Reality Check. Proceedings of the 2nd IEEE International Conference on Semantic Computing (ICSC), Santa Clara, USA, IEEE Press.

[15] Li, Y.; Bandar, A.; McLean, D. (2003): An approach for measuring semantic similarity between words using multiple information sources. Transactions on Knowledge and Data Engineering 15, p. 871–882.

[16] Lin, D. (1998): An Information-Theoretic Definition of Similarity. Proceedings of the 15th International Conference on Machine Learning, USA.

[17] Resnik, P. (1999): Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. Journal of Artificial Intelligence Research 11, p. 95–130.

[18] Sycara, K.; Widoff, S.; Klusch, M.; Lu, J. (2002): LARKS: Dynamic Matchmaking Among Heterogeneous Software Agents in Cyberspace. *Autonomous Agents and Multi-Agent Systems*, 5:173-203, Kluwer.

[19] Samuel, K. et al. (2008): Translating OWL and semantic web rules into prolog: Moving toward description logic programs. *Theory and Practice of Logic Programming*, 8(03):301-322.

[20] Scheffer,T.; Herbrich, R.; Wysotzki, F. (1996), Efficient theta-Subsumption based on Graph Algorithms. *Lecture Notes In Computer Science*, 1314, Springer.

Appendix A. Algorithms

```
Input: Training set $TS'$
Output: Trained $SVM$
// compute features for each training sample
1  $TS \leftarrow \emptyset$
2  foreach $(R, S, y) \in TS'$ do
3      $x \leftarrow (\texttt{MatchIO}_{Logic}(R,S) = Exact, \ldots, \texttt{MatchIO}_{Logic}(R,S) =$
       $LFail, \texttt{val}_{(S,R)}(H_1), \texttt{val}_{(S,R)}(H_2), \texttt{MatchIO}_{Text}(R,S),$
       $\texttt{MatchIO}_{Struct}(R,S), \texttt{MatchPE}(R,S))$
4      $TS \leftarrow TS \cup \{(x, y)\}$
5  end
// filter different relevant cases
6  $LP \leftarrow \emptyset$
7  $E_1 \leftarrow \emptyset, \ldots, E_4 \leftarrow \emptyset$
8  foreach $(x, y) \in TS$ do
       // irrelevant samples for approx.
9      if $x[5] \neq 1$ then $LP \leftarrow LP \cup \{(x, y)\}$
10     if $y = 1 \wedge (x[6] > 0 \vee x[7] > 0)$ then $E_1 \leftarrow E_1 \cup \{(x, y)\}$
11     if $y = -1 \wedge x[6] \leq 0 \wedge x[7] \leq 0$ then $E_2 \leftarrow E_2 \cup \{(x, y)\}$
12     if $y = 1 \wedge x[6] \leq 0 \wedge x[7] \leq 0$ then $E_3 \leftarrow E_3 \cup \{(x, y)\}$
13     if $y = -1 \wedge (x[6] > 0 \vee x[7] > 0$ then $E_4 \leftarrow E_4 \cup \{(x, y)\}$
14 end
// improve feature vectors based on approximation
15 $w_1 \leftarrow \texttt{coherence}(\{(x, y) \in E_1 \cup E_4 : x[6] \geq x[7]\}, E_1 \cup E_3)$
16 $w_2 \leftarrow \texttt{coherence}(\{(x, y) \in E_1 \cup E_4 : x[6] < x[7]\}, E_1 \cup E_3)$
17 $w_3 \leftarrow \texttt{coherence}(\{(x, y) \in E_2 \cup E_3 : x[6] \geq x[7]\}, E_2 \cup E_4)$
18 $w_4 \leftarrow \texttt{coherence}(\{(x, y) \in E_2 \cup E_3 : x[6] < x[7]\}, E_2 \cup E_4)$
19 foreach $(x, y) \in E_1$ do
20     $x[5] \leftarrow 0$
21     if $x[6] \geq x[7]$ then $x[6] \leftarrow w_1 \cdot x[6]$, $x[7] \leftarrow 0$
22     if $x[6] < x[7]$ then $x[6] \leftarrow 0$, $x[7] \leftarrow w_2 \cdot x[7]$
23 end
24 foreach $(x, y) \in E_2$ do
25     if $x[6] \geq x[7]$ then $x[6] \leftarrow w_3 \cdot x[6]$, $x[7] \leftarrow 0$
26     if $x[6] < x[7]$ then $x[6] \leftarrow 0$, $x[7] \leftarrow w_4 \cdot x[7]$
27 end
28 foreach $(x, y) \in E_4$ do
29     if $x[6] \geq x[7]$ then $x[6] \leftarrow (1 - w_1) \cdot x[6]$, $x[7] \leftarrow 0$
30     if $x[6] < x[7]$ then $x[6] \leftarrow 0$, $x[7] \leftarrow (1 - w_2) \cdot x[7]$
31 end
32 $TS \leftarrow LP \cup E_1 \cup E_2 \cup E_4$
// train SVM on improved features
33 return $\texttt{trainSVM}(TS)$
```

Algorithm 1: Training Phase

```
Input: Request/offer pair $(R, S)$, weighting parameter $\alpha$
Output: Approximation explanation and valuation $(H, v)$
// compute approximation for all parameter combinations
$M_{in} \leftarrow \emptyset$
foreach $I_S \in S_{in}$ do
    foreach $I_R \in R_{in}$ do
        $M_{in} \leftarrow M_{in} \cup \{(I_S, I_R) \mapsto \texttt{approximate(}I_R, I_S\texttt{)}\}$
    end
end
$M_{out,\sqsubseteq} \leftarrow \emptyset$
foreach $O_R \in R_{out}$ do
    foreach $O_S \in S_{out}$ do
        $M_{out,\sqsubseteq} \leftarrow M_{out,\sqsubseteq} \cup \{(O_R, O_S) \mapsto \texttt{approximate(}O_S, O_R\texttt{)}\}$
    end
end
$M_{out,\sqsupseteq} \leftarrow \emptyset$
foreach $O_R \in R_{out}$ do
    foreach $O_S \in S_{out}$ do
        $M_{out,\sqsupseteq} \leftarrow M_{out,\sqsupseteq} \cup \{(O_R, O_S) \mapsto \texttt{approximate(}O_R, O_S\texttt{)}\}$
    end
end
// bipartite graph matching for approximation
$H_I \leftarrow \texttt{BPG(}M_{in}\texttt{)}$
$H_{O,\sqsubseteq} \leftarrow (H_I, \texttt{BPG(}M_{out,\sqsubseteq}\texttt{)})$
$H_{O,\sqsupseteq} \leftarrow (H_I, \texttt{BPG(}M_{out,\sqsupseteq}\texttt{)})$
// compute valuation for both overall explanations
$val_{(S,R)}(H_1) \leftarrow$
$\frac{1}{2\cdot|S_{in}|} \cdot \sum_{(I_S,I_R)\in H_I} M_{in}(I_S, I_R) + \frac{1}{2\cdot|R_{out}|} \cdot \sum_{(O_R,O_S)\in H_{O,\sqsubseteq}} M_{out,\sqsubseteq}(O_R, O_S)$
$val_{(S,R)}(H_2) \leftarrow$
$\frac{1}{2\cdot|S_{in}|} \cdot \sum_{(I_S,I_R)\in H_I} M_{in}(I_S, I_R) + \frac{1}{2\cdot|R_{out}|} \cdot \sum_{(O_R,O_S)\in H_{O,\sqsupseteq}} M_{out,\sqsupseteq}(O_R, O_S)$
// return approximations and their valuation
return
$\{(H_1 = (H_I, H_{O,\sqsubseteq}), val_{(S,R)}(H_1)), (H_2 = (H_I, H_{O,\sqsupseteq}), val_{(S,R)}(H_2))\}$
```

Algorithm 2: Approximate Logic-based Matching

```
   Input: Original concept $C$ and target concept $D$
   Output: Approximation explanation and valuation $(H(C,D), v)$
1  $(G,K) \leftarrow \texttt{CCP}(C,D)$                         // concept contraction
2  $H \leftarrow \texttt{SAP}(K,D)$                             // concept abduction
3  $C' \leftarrow \sigma[\bar{H},H](K^h)$                        // approximated concept
4  $H(C,D) \leftarrow (G,H,C')$                                 // explanation
5  $v \leftarrow \texttt{sim}_{inf}(C',D) - (1 - \texttt{sim}_{inf}(C',C))$   // valuation
6  return $(H(C,D),v)$
```

Algorithm 3: Concept approximation

Appendix B. S2M2/iSeM Filter Expressions

expression	domain	range	# of sub-expr.	description
basic				
and	any	boolean	2..n	conjunction
or	any	boolean	2..n	disjunction
negation	any	boolean	1	negates sub-expression
inverse	any	any	1	switches request and candidate paramter
true	any	boolean	–	always returns $true$
greater equals	any	boolean	2	checks if first sub-expression is greater or equal than second
average	any	float	2..n	computes average value
maximum	any	float	2..n	computes maximum value
fixed value(α)	any	float	–	always returns α
boolean as float	any	float	1	converts boolean subexpression result to one of $\{0,1\}$
conditional	any	any	3	checks first subexpression and executes second or third based on result (if-then-else)
semantic annotations	service	any	1	extracts collection of concepts (inputs or outputs) and applies subexpression to it
literals	service	any	1	extracts collection of string values (descriptions, unfolded I/O) and applies subexpression to it
specification	service	any	1	extracts specification (P/E) and applies subexpression to it
forall exists	collection	boolean	1	checks if there exists a candidate value for each request value, where subexpression holds (surjection)
bipartite graph matching	collection	float	1	applies bipartite graph matching and returns average valuation (injection)
concat	string coll.	any	1	concatenates strings and evaluates subexpression on result
inference				
equivalence	concept	boolean	–	concept equivalence $\equiv$
subsumption	concept	boolean	–	concept subsumption $\sqsubseteq$
least generic concept	concept	boolean	–	checks for $\sqsubseteq_1$
implication	spec.	boolean	–	checks if logical expressions imply $\Rightarrow$
text similarity				
loss of information	string	float	–	computes loss of information
cosine	string	float	–	computes VSM-based cosine measure
iSeM				
approximated subsumption	concept	float	–	computes valuation of approximated logical matching $v(C,D)$
direct common subsumers	concept	concept coll.	–	computes direct common subsumers
structural similarity	concept	float	–	computes structural similarity $sim_{csim}(C,D)$
svm	any	float	2..n	aggregates sub-expressions using SVM

Table B.1: S2M2 and iSeM filter expressions.