

Partial Matchmaking for Complex Product- and Service Descriptions

Heiner Stuckenschmidt, Martin Kolb
KR&KM Research Group
University of Mannheim

Abstract: Matchmaking between offers and requests is an essential mechanism in electronic market places. Description Logics have been proposed as a appropriate framework for representing offers and requests and standard methods from description logics reasoning have been proposed for computing matches. We argue that classical reasoning is not well suited for realistic scenarios as it does not allow to compute partial matches and qualify the mismatch. We propose a more flexible approach for matching complex product and service descriptions based on the notion of approximate subsumption in expressive description logics and present a prototypical implementation of our matching system.

1 Motivation

As more and more business is performed online, the importance of electronic marketplaces where products and services are traded in a semi-automatic way is increasing steadily. One of the basic problems connected to an efficient use of electronic marketplaces is the problem of matchmaking between offers for products or services and requests by potential customers [Vei03]. Matchmaking tries to find offers that as closely meet the requirements of a requester as possible. This task is non-trivial as in many cases, it is not possible to find perfect matches. In this situation, the matchmaking algorithm has to find 'good enough' offers. Determining which offers are 'good enough' often does not only depend on offer and request it self, but also on the preferences of the customer and the intended use of the product or service. This means that matching has to be customizable towards the specific customer.

Another challenge for successful matchmaking in electronic marketplaces is the need to integrate heterogeneous product and service descriptions. As the representations used by different participants for their internal purposes are often geared towards the specific needs of the company and show little standardization across different enterprises. In order to be able to use this information in matchmaking, open standards for classifying and describing products have been developed that provide a common framework for describing products and services by assigning products to product classes and by defining a number of properties for describing the concrete offer in more details. Examples of such standards are eClass, UNSPSC and RosettaNet [HLS08].

It has been argued that matchmaking in electronic markets can benefit from the use of

semantic web technologies. In particular, OWL has been proposed as a suitable language for representing product classifications [Hep06] and to use logical reasoning in OWL for computing matches between offers and requests [GCTB01, LH04]. This approach has a number of advantages. In particular, the ability to capture complex descriptions in a meaningful way and the ability to make use of background knowledge about the domain of discourse in terms of product and services ontologies support more informed choices. On the other hand, the use of logical reasoning also has some problems. In particular, there are very limited ways of determining particular matches. The abilities boil down to deciding whether an offer is equivalent, more specific, more general or consistent with an offer. These distinctions are often too coarse grained to be useful in matchmaking.

In this paper, we present an approach that combines the benefits of using OWL as a basis for complex product and service descriptions with a more fine-grained way of determining partial matches. The approach is based on more theoretical work on computing approximate subsumption relations between concepts descriptions in Description Logics [Stu07a, Stu07b]. In the following, we first briefly recall proposals for modeling products and services as a basis for logic-based matchmaking as a basis for our work. We then introduce the idea of partial matchmaking based on approximate subsumption. We explain how approximate matches can be computed by extending existing OWL reasoners and present a prototypical implementation of our matchmaking engine and show its application to computing matches in the example domain of hardware components. We close with a brief review of related work and a discussion of the pros and cons of our approach.

2 Product and Service Modeling and Matchmaking

Hepp argues that an adequate description of products consists of the class of a product and of the specification of characteristic properties of the product as well as its intended use. The corresponding descriptions can be based on existing product classification standards such as eClass that provide a hierarchies of product classes and characteristic properties for different classes. A drawback of existing classifications is the lack of a formal foundation that support an automatic comparison of the corresponding descriptions. Different researchers have argued for the benefits of using description logics as a basis for describing products and services [GCTB01, DEDM03, LH04] as description logics provide a well founded framework for talking about class membership and constraints on properties. In our work, we follow the approach proposed by Li and Horrocks [LH04]. They propose to model offers and requests as description logic expressions such as the following example taken from their paper:

$$\begin{aligned}
advert1 &\equiv Sales \sqcap \\
&\quad \forall item.(PC \sqcap (\geq 128 \text{ memorySize})) \sqcap \\
&\quad (\geq 700 \text{ hasUnitPrice}) \sqcap \\
&\quad (\leq 200 \text{ hasQuantity}) \sqcap \\
&\quad (\forall delivery.((\leq 20030501 \text{ date}) \sqcap \\
&\quad (\forall location.Manchester)))
\end{aligned}$$

The example describes a sales service and its properties. Here *Sales* identifies the type of service that can be based on an existing classification and the specifics of the service are represented using constraints on the relevant properties. As an important aspect of a sales service, the offered product is represented as well by assigning it to a product class (in this case the class of personal computers) and restrictions on the properties of the individual product such as the memory size and properties of the whole offer such as the amount available and the unit price. Lei and Horrocks define different notions of exact and partial matches in terms of description logic subsumption. In particular, they define the following degrees of match between a service request R and a service advertisement A [LH04]:

Exact: $R \equiv A$ The request is semantically equivalent with the advertised product. This means that the advertised product exactly matches the request.

Subsume: $A \sqsubseteq R$ The product description is more specific than the request. This means that the advertised product satisfies all requirements formulated in the request and has some additional features.

Plugin: $R \sqsubseteq A$ The request is more specific than the advertised product. This means it is not clear whether the product satisfies all requirements and more information is needed.

Intersection: $(A \sqcap R) \not\sqsubseteq \perp$ This means that the request and the advertised have something in common and that there are no explicit conflicts.

While these types of matches have been widely used in the literature it is meanwhile commonly agreed that these coarse distinctions are only useful for making a pre-selection on the available offers. In particular, using these distinctions, it is possible to filter out offers that are clearly not relevant and to sort relevant offers in different categories and to first concentrate on a more promising class of offers (e.g. first try plugin matches before moving to the intersecting ones). It has also been shown that these different categories can be used to precompute matches between standard requests and available offers and use this information for more efficient matching [SHH07].

3 A new Approach for Partial Matchmaking

A major problem of existing matchmaking approaches that rely on standard description logic inference for computing matches as explained above is the lack of differentiation provided by the four degrees of matching. If an offer only disagrees with the request on a single property it drops out of the class of subsuming matches and ends up in the least attractive class of intersection matches that will contain almost any offer with only the slightest relation to the request. The goal of our work is to provide a more fine-grained approach to matchmaking in the context of description logics that differentiates between different degrees of matching in a meaningful way and is also able to provide feedback to the requester on the specific aspects of an offer that did not match the request.

3.1 Partial Matchmaking

Before we can start to develop a model for partial matchmaking, we first have to define the objectives of the work in more details in terms of fundamental properties an approximate subsumption operator should have to meet the requirements of typical matchmaking tasks. For this purpose, we define a matchmaking as the task of determining from a set of advertisements A_i those that match the requirements of a request R . In the context of description logics it has been argued that an advertisement A matches the request if $A \sqsubseteq B$ holds, because this means that all requirements represented in the request are fulfilled by the advertisement [LH04]. Based on this view, we define partial matchmaking based on an approximate subsumption operator $\tilde{\sqsubseteq}$ as determining all advertisements for which $A_i \tilde{\sqsubseteq} R$ holds. The corresponding operator should be weaker and more flexible than standard subsumption but still have well-defined logical properties. Our approach to partial matchmaking in description logics is based on the idea of a stepwise relaxation the matching conditions by ignoring certain aspects of the descriptions to be matched. The challenge of this approach is to define the relaxation in such a way that the properties mentioned above are met and the relaxation has a meaning in the context of the underlying logic. The approach we have developed is based on the signature - the names of concepts and relations - of the expressions to be matched. The idea is to determine a subset of this signature that can be ignored in the process of computing subsumption relations. This modified subsumption relation can now be used as a basis for computing partial matches. In particular, the statement

$$A \underset{S}{\sqsubseteq} R$$

means that the offer A matches the request R with respect to the restrictions on the classifications and properties mentioned in S . From a practical point of view, this has the advantage that in the case where none of the offers matches our request based on the classical notion of subsumption, we can relax our request by excluding a number of properties from the set S and determining whether any of the offers meets this relaxed request. In an ideal case, this relaxation is done automatically and the requester receives a number

of offers that almost match the request and some information about which of the aspects failed to match the request. Based on this information the requester can decide which of the almost matching offers make sense for him.

This notion of matching has a number of properties that that are desirable for any logic-based matching operator:

completeness In order to allow for partial matches, the operator should drop the requirement of correctness with respect to standard subsumption. On the other hand, we want to preserve completeness in order to ensure that all matching result from standard subsumption are retained.

$$A \sqsubseteq R \Rightarrow A \tilde{\sqsubseteq} R$$

This property directly follows from the monotonicity property of approximate subsumption ([Stu07a], lemma 1).

gradual relaxation Rather than having a single subsumption relation that can or cannot hold between an advertisement and a request we would like to have the possibility to gradually relax the subsumption relation to include less and less relevant advertisements into the result set.

$$\exists i, j \in \{0, \dots, n\} : A \tilde{\sqsubseteq}_i R \Rightarrow A \tilde{\sqsubseteq}_j R, \text{ for } i \leq j$$

This gradual relaxation also provides a basis for ranking results, because advertisements that are results with respect to a lower index can be assumed to be more relevant with respect to the request. For approximate subsumption this gradual relaxation can be achieved by choosing gradually increasing the set S . The property follows from the generalized monotonicity of the approximation ([Stu07a], lemma 2).

generalization of subsumption A rather straightforward but important property is the property, that the approximate subsumption operator should be a generalization of classical subsumption. We formalize this by claiming that the approximate subsumption operation with index zero corresponds to classical subsumption.

$$A \sqsubseteq R \Leftrightarrow A \tilde{\sqsubseteq}_0 R$$

This property trivially holds for our notion of appropriate subsumption if we choose $\tilde{\sqsubseteq}_0$ to be \sqsubseteq .

model-theoretic semantics In order to guarantee that the approximate subsumption operator has a well defined formal meaning, we require the existence of a model-theoretic semantics for this operator as well. In particular, we claim that the operator should be defined on the basis of a non-standard interpretation $\tilde{\mathcal{I}}$ in the following way

$$\forall \mathcal{I} \exists \tilde{\mathcal{I}} : \mathcal{I} \models A \sqsubseteq R \Leftrightarrow \tilde{\mathcal{I}} \models A \sqsubseteq R$$

The existence of a model theoretic semantics is one of the core results of our previous work on approximate subsumption ([Stu07a], definition 3).

sound and complete inference procedures Finally, we want the approximate subsumption operator to behave well with respect to the non-standard interpretation. In particular, we require the existence of a proof procedure $\tilde{\vdash}$ that are sound and complete with respect to the non-standard interpretation.

$$\tilde{\vdash} : \tilde{\mathcal{I}} \models A \sqsubseteq R \Leftrightarrow \tilde{\vdash} A \sqsubseteq R$$

We have shown that there are sound and complete reasoning procedures for the notion of approximate subsumption ([Stu07a], theorem 2 and 3). In the following, we briefly explain the idea of computing approximate subsumption and applying it to the problem of matchmaking.

3.2 Computing Partial Matches

From a technical point of view a nice feature of our approach is that it can actually be implemented by simply performing syntactic modifications on concept expressions. In particular, in order to check whether a statement $C \sqsubseteq_S D$ holds, we take the expression $(C \sqcap \neg D)$ and transform it into a concept expression that simulates the non-standard interpretation. The corresponding transformation $(\cdot)^-$ is defined as follows

$$\begin{aligned} (A)^- &\rightarrow \perp \text{ if } A \in S \\ (\neg A)^- &\rightarrow \perp \text{ if } A \in S \\ (\neg C)^- &\rightarrow \neg(C)^+ \\ (C \sqcap D)^- &\rightarrow (C)^- \sqcap (D)^- \\ (C \sqcup D)^- &\rightarrow (C)^- \sqcup (D)^- \\ (\leq n r.C)^- &\rightarrow (\leq 0 r.(C)^+) \text{ if } r \in S \\ (\leq n r.C)^- &\rightarrow (\leq n r.(C)^+) \text{ if } r \notin S \\ (\geq n r.C)^- &\rightarrow (\geq \max r.(C)^-) \text{ if } r \in S \\ (\geq n r.C)^- &\rightarrow (\geq n r.(C)^-) \text{ if } r \notin S \end{aligned}$$

Here \max is an integer number that is larger than any other number occurring in any qualified number restriction in the concept expression. This is sufficient to model the interpretation that requires less than an infinite number of r-successors. Analogously, we

define a transformation function $(\cdot)^+$ that creates a concept expression that simulates the upper approximation of a concept expression. This transformation is defined as follows:

$$\begin{aligned}
(A)^+ &\rightarrow \top \text{ if } A \in S \\
(\neg A)^+ &\rightarrow \top \text{ if } A \in S \\
(\neg C)^+ &\rightarrow \neg(C)^- \\
(C \sqcap D)^+ &\rightarrow (C)^+ \sqcap (D)^+ \\
(C \sqcup D)^+ &\rightarrow (C)^+ \sqcup (D)^+ \\
(\leq nr.C)^+ &\rightarrow (\leq max - 1r.(C)^-) \text{ if } r \in S \\
(\leq nr.C)^+ &\rightarrow (\leq nr.(C)^-) \text{ if } r \notin S \\
(\geq nr.C)^+ &\rightarrow (\geq 1r.(C)^+) \text{ if } r \in S \\
(\geq nr.C)^+ &\rightarrow (\geq nr.(C)^+) \text{ if } r \notin S
\end{aligned}$$

We again use the number *max* for modeling an infinite number of r-successors. Further, we have to use the condition ≥ 1 instead of < 0 which is equivalent. It can be shown that these rewriting rules provide a way for computing approximate subsumption. We have implemented a prototype of a matching system based on this approach. The system is described in more details in section 4.2

3.3 Illustration of the Approach

We assume a request asking for a Sales service that offers PCs or Laptops with at least 512 MB main memory, at least 256 MB Cache Memory and a price of at most 500 Dollars. The corresponding request can be formulated using the following concept expression:

$$\begin{aligned}
request &\equiv Sales \sqcap \\
&(\forall item.(PC \sqcup Laptop \sqcap (\geq 512 has - memory.Main) \\
&\sqcap (\geq 256 has - memory.Cache) \\
&\sqcap (\leq 500 price.Dollar)))
\end{aligned}$$

We further assume a sales service offering PCs with 256 MB main memory and 256 MB Cache Memory at a price of 450 Dollars and Laptops with 512 MB Main Memory and 256 MB Cache Memory at a price of 650 Dollar. This service can be described using the following concept expression:

$$\begin{aligned}
advert1 \equiv Sales \quad & \sqcap \\
& (\forall item.(PC \quad \sqcap \quad (\geq 256 \text{ has-memory.Main}) \\
& \quad \sqcap \quad (\geq 256 \text{ has-memory.Cache}) \\
& \quad \sqcap \quad (\leq 450 \text{ price.Dollar}))
\end{aligned}$$

$$\begin{aligned}
advert2 \equiv Sales \quad & \sqcap \\
& (\forall item.(Laptop \quad \sqcap \quad (\geq 512 \text{ has-memory.Main}) \\
& \quad \sqcap \quad (\geq 256 \text{ has-memory.Cache}) \\
& \quad \sqcap \quad (\leq 650 \text{ price.Dollar})))
\end{aligned}$$

It is easy to see that neither $advert1 \sqsubseteq request$ nor $advert2 \sqsubseteq request$ holds. The both adverts satisfy the condition of being a sales service, it also offers the right kinds of items - PCs or Laptops, but each of the items offered fails to satisfy one of the requirements. While the PCs do not have enough main memory, the Laptops are too expensive. The reasoner is unable to detect a clash in the expression $advert \sqcap \neg request$. In particular, it fails to detect a clash between the expressions $(\leq 511 \text{ has-memory.Main})$ and $(\geq 256 \text{ has-memory.Main})$ in the case of PCs and between $(\leq 650 \text{ price.Dollar})$ and $(\geq 501 \text{ price.Dollar})$ for the case of Laptops. Using our approach, excluding $has-memory$ from the set S leads to a re-formulation of the problem where $(\leq max-1 \text{ has-memory.Main})$ and $(\geq max \text{ has-memory.Main})$ are compared. As a consequence $advert1 \sqsubseteq_{S-\{has-memory\}} request$ holds¹. In the second case, using our approach will re-write the problem to $(\leq 0 \text{ price.Dollar})$ and $(\geq 1 \text{ price.Dollar})$ which also leads to a clash. Thus we also have $advert2 \sqsubseteq_{S-\{price\}} request$ for the Laptop case. As in the case of information integration, the subset of the vocabulary chosen provides the user with valuable feedback with respect to the relevance of the different matches. In particular, the user can decide whether the original constraint on the price or on the memory should be relaxed.

4 Implementation

We have implemented our method in terms of a research prototype that can be used as a basis for matching experiments. As a first test scenario, we have built an ontology of hardware components and tested the method on the basis of this data. In the following, we briefly describe the test data and explain how our system can be used to compute partial matches for requests for certain hardware products.

¹here S denotes the signature of the concept expression

4.1 An Ontology of Computer Components

As a basis for testing the matcher, we have built an ontology of hardware components, in particular CPUs on the basis of real product data obtained from the Alternate online shop (www.alternate.de) for hardware components. The ontology consists of a hierarchy of classes and relations describing different types of hardware components (CPUs, controllers, harddisks, mainboards, etc.).

Additionally, a number of concrete CPUs offered by the shop are modeled in more details using the scheme proposed by Li and Horrocks. In particular, their are described in terms of properties such as the speed, the amount and type of cache memory available or the number of cores. In total, the current version of the model contains about 200 classes 100 of which represent concrete products being offered, all different kinds of CPUs.

4.2 A Matching Prototype

Figure 1 shows the interface of the prototypical matching system we have implemented. The system is built on top of the Pellet reasoner, an open source implementation of a state of the art description logic reasoner. The prototype allows us to compute approximate matches based on arbitrary OWL-DL ontologies. After loading the ontology, the user selects a class for indicating what kind of objects he is looking for. In our example, we would choose Sales services, in other examples we might be interested in matching requirements on hardware components in the context of computational services offered. After selecting the target class the actual matching is performed using the interface shown in figure 1.

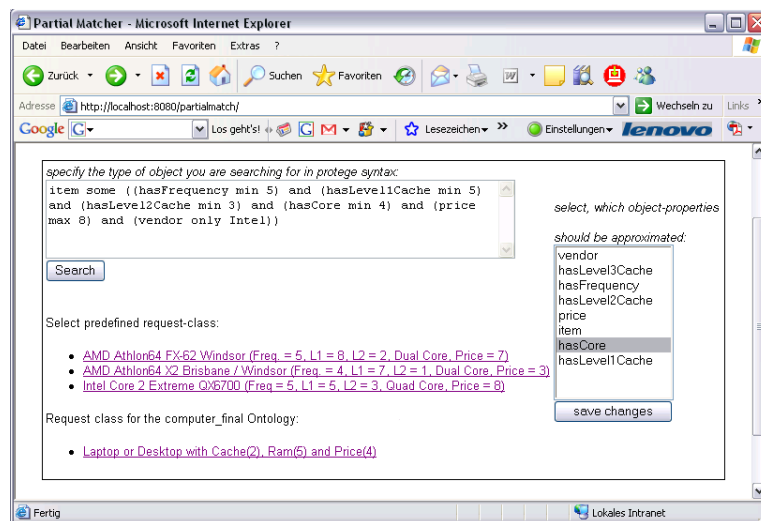


Abbildung 1: Interface of the Product-Matcher Prototype

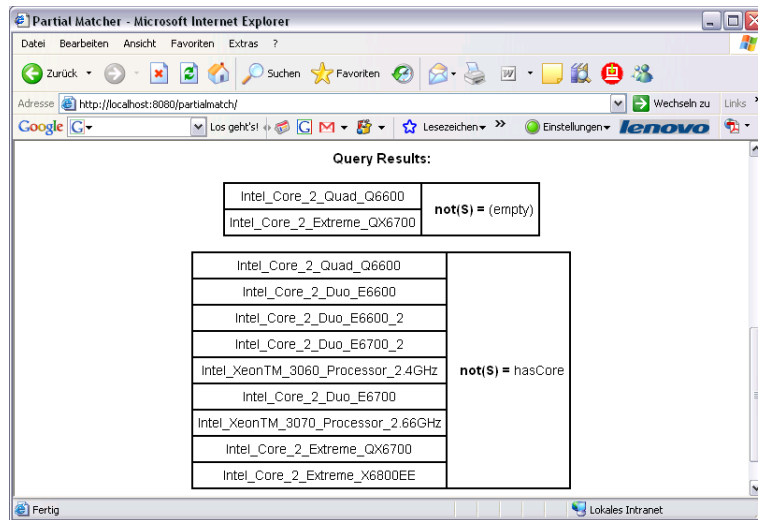


Abbildung 2: Results of an example Matching Process

The interface consists of an input form where the user can formulate a complex request in terms of a description logic expression. The system accepts a simplified syntax for DL expressions adopted from the Protege OWL editor. For demonstration purposes, a number of standard queries can be chosen. The current prototype focusses on the approximation of properties rather than classifications. The user can choose some properties to be approximated from the menu on the right hand side of the screen. Based on the input the system returns all classes in the ontology that are subsumed by the query in the classical sense and the set of all classes that are subsumed disregarding the properties specified by the user. Figure 2 shown the result for the example query in figure 1.

We can see that using the standard notion of subsumption as a matching criterion only leads to two matching products. When using the approximate subsumption operator and relaxing our our requirements on the number of cores the CPUs should have for we see that we get nine matches many of which will also be appropriate in cases.

5 Related Work

Recently, there are some efforts that try to address this problem by combining OWL with numerical techniques for uncertain reasoning, in particular with techniques for probabilistic [GL02] and fuzzy reasoning [Str05]. These approaches are able to compute partial matches by assigning an assessment of the degree of matching to the subsumption relation. This degree of matching normally is a real number between zero and one and therefore allows an absolute ordering of the solutions (An early description of the use of fuzzy values

to compare concepts is discussed in [SWKL02]). Although, in principle this is a solution to the problem of computing the best partial match, there are two principled problems with these approaches

- Defining an interpreting numerical assessments of uncertainty is a difficult problem. Often, it is not clear where the required number come from and what it means that one solution matches with a degree of 0.9 versus a degree of 0.89.
- The reduction to a single numerical assessment of the mismatch does not allow different users to discriminate between different kinds of mismatches. In particular, the user is normally not able to specify which aspects of the request are the most important ones.

In order to avoid these problems alternative ways of computing partial matches are needed, that are only based on the comparison of concept expressions. Another approach that compares the concept structures is reported in [DEDM03], but the degree of matching is again measured in a single value which leaves us with the second problem mentioned above.

6 Discussion

We have presented the prototype of a matching system from complex product and service descriptions following the ideas presented in previous work (i.e. [LH04]). Our approach shares the benefits of this earlier proposal as it allows to match complex descriptions and to take background knowledge in terms of a domain ontology into account. On the other hand, our approach solves some of the problems of earlier work. In particular it provides a more flexible framework for computing and qualifying matches. While previous work only distinguished four different kinds of matches that are rather motivated by the standard reasoning problems in description logics than the actual needs of matchmaking scenarios, our approach can distinguish different degrees of matching based on the set of the concept and relation names that had to be ignored in order to achieve a match. This qualification of the match provides valuable feedback to the user who can make a more informed decision about which offer to accept.

We believe that our approach provides an excellent basis for advanced matching solutions. Before the approach is ready for application in practice there are still a number of steps to be done. First of all, we need a better understanding of the right way of representing product offers in description logics. On the one hand these models have to take existing standards for product classification and description into account, on the other hand, they have to take a form that supports logical reasoning in such a way that matching results meet the users intuition. The approach proposed by Li and Horrocks that we took as a basis for our work emphasizes the second aspect and does not provide insights about how to link to existing standards. Most likely, there is not a single representation that perfectly meets these requirements and we will have to deal with representations at different levels of abstraction that are used for different purposes. The representation of e-commerce

standards in OWL proposed in [Hep06] can serve as a starting point. From there we have to find transformations into a format that uses logical axioms to define the properties of concrete products as proposed by Li and Horrocks. Another important next step is to take user preferences into account when matching. These preferences can easily be expressed in terms of a partial order over subsets of the signature stating which properties the user considers to be important and which not. Based on this information the system will be able to automatically select the most appropriate matches for a specific user.

Literatur

- [DEDM03] Tommaso Di Noia, Francesco Donini Eugenio Di Sciascio und Marina Mongiello. A system for principled matchmaking in an electronic marketplace. In *Proceedings of the Twelfth International World Wide Web Conference*, Seiten 321–330, 2003.
- [GCTB01] Javier Gonzalez-Castillo, David Trastour und Claudio Bartolini. Description Logics for Matchmaking of Services. Technical Report HPL-2001-265, HP Laboratories Bristol, 2001.
- [GL02] Rosalba Giugno und Thomas Lukasiewicz. P-SHOQ(D): A Probabilistic Extension of SHOQ(D) for Probabilistic Ontologies in the Semantic Web. In *Proceedings of the 8th European Conference on Logics in Artificial Intelligence (JELIA'02)*, 2002.
- [Hep06] Martin Hepp. Products and Services Ontologies: A Methodology for Deriving OWL Ontologies from Industrial Categorization Standards. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 2(1):72–99, 2006.
- [HLS08] Martin Hepp, Joerg Leukel und Volker Schmitz. A Quantitative Analysis of Product Categorization Standards: Content, Coverage, and Maintenance of eCI@ss, UNSPSC, eOTD, and the RosettaNet. *Knowledge and Information Systems (KAIS)*, 2008. Accepted for publication.
- [LH04] Lei Li und Ian Horrocks. A Software Framework for Matchmaking Based on Semantic Web Technology. *International Journal of Electronic Commerce*, 8(4):39 – 60, 2004.
- [SHH07] M. Stollberg, M. Hepp und J. Hoffmann. A Caching Mechanism for Semantic Web Service Discovery. In *Proc. of the 6th International Semantic Web Conference (ISWC 2007)*, Busan, South Korea, 2007.
- [Str05] Umberto Straccia. Towards a Fuzzy Description Logic for the Semantic Web (Preliminary Report). In *Proceedings of the 2nd European Semantic Web Conference ESWC-05*, Seiten 167–181, 2005.
- [Stu07a] Heiner Stuckenschmidt. Approximate Subsumption for \mathcal{ALCQ} . In *Proceedings of the 20th International Workshop on Description Logics (DL'07)*, 2007.
- [Stu07b] Heiner Stuckenschmidt. Partial Matching Using Approximate Subsumption. In *Proceedings of the 22nd Conference on Artificial Intelligence (AAAI-07)*, 2007.
- [SWKL02] Katia Sycara, Seth Widoff, Matthias Klusch und Jianguo Lu. Larks: Dynamic Matchmaking Among Heterogeneous Software Agents in Cyberspace. *Autonomous Agents and Multi-Agent Systems*, 5(2):173 – 203, 2002.
- [Vei03] Daniel Veit. *Matchmaking in Electronic Markets*, Jgg. 2882 of *Lecture Notes in Computer Science*. Springer Verlag, 2003.