Expressive Querying over Fuzzy DL-Lite Ontologies(*)

Jeff Z. Pan[1], Giorgos Stamou[2], Giorgos Stoilos[2], and Edward Thomas[1]

[1] Department of Computing Science, University of Aberdeen, AB24 3UE, UK

[2] Department of Computer Science, National & Technical Univ. of Athens, Zographou 15780, GR

Abstract. Fuzzy Description Logics (f-DLs) have been proposed as formalisms capable of capturing and reasoning about imprecise and vague knowledge. The last years, research in Description Logics, and also in f-DLs, is largely focused on the development of languages where complexity of query answering is as efficient as query answering in data bases. One such example is the DL-Lite language and its fuzzy extension f-DL-Lite. In the current paper we present various a variety of query languages by which we can query a fuzzy DL-Lite knowledge base. Then, we present a prototype implementation for querying f-DL-Lite ontologies.

1 Introduction

Recently there have been quite a few work on DL-based fuzzy ontology languages [11, 9, 10, 8, 12], which have been proposed as formalisms capable of capturing and reasoning about imprecise and vague knowledge. In particular, Straccia [12] extended the DL-Lite ontology language [2], which enables highly efficient query answering procedures, to fuzzy DL-Lite. He showed that conjunctive query answering in f-DL-Lite is quite similar to query answering in crisp DL-Lite, although some technical details like top-k answering and a modification on the knowledge base consistency algorithm of crisp DL-Lite need to be considered.

In this paper, we propose two novel query languages, which provide one with different ways on querying fuzzy DL-Lite ontologies. More precisely, we allow the users to specify threshold queries and general fuzzy queries. Comparing with Straccia's query language, the threshold query language is flexible as it allows one to specify a threshold for each query atom (such as "tell me e-shops that are popular [with degrees at least 0.8] and sell good books [with degrees at least 0.9]"), while the general fuzzy query language is a general form of Straccia's query language. Furthermore, we present algorithms for answering these queries and report implementations as well as preliminary but encouraging evaluation based on the ONTOSEARCH2, which is a query engine for both DL-Lite and fuzzy DL-Lite.

(*) This extended abstract is accompanied with an online technical report (http://www.ontosearch.org/TR/f-DL-Lite.pdf), which contains more details including algorithms and the proofs.

2 f-DL-Lite

In the current section we will briefly introduce the fuzzy DL-Lite (which we call f-DL-Lite) language [12], which extends DL-Lite core with fuzzy assertions of the forms $B(\mathtt{a}) \geq n, R(\mathtt{a}, \mathtt{b}) \geq n$, where B is basic class, R is a property, $\mathtt{a}$ and $\mathtt{b}$ are individuals and n is a real number in the range $[0, 1]$. We assume that the reader is familiar with DL-Lite; in a different case [1, 2] are the standard sources. We only remind the form of *conjunctive queries* which we will use in the following. A conjunctive query (CQ) q is of the form

$$q(X) \leftarrow \exists Y.conj(X, Y) \tag{1}$$

where $q(X)$ is called the head, $conj(X, Y)$ is called the body, X are called the distinguished variables, Y are existentially quantified variables called the non-distinguished variables, and $conj(X, Y)$ is a conjunction of atoms of the form $\mathsf{A}(v)$, $R(v_1, v_2)$, where A, R are respectively *named* classes and *named* properties, v, v_1 and v_2 are *individual* variables in X and Y or individual names in $\mathcal{O}$. The semantics of f-DL-Lite ontologies is defined in terms of *fuzzy interpretations* [11]. A fuzzy interpretation is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where the domain $\Delta^{\mathcal{I}}$ is a non-empty set of objects and $\cdot^{\mathcal{I}}$ is a fuzzy interpretation function, which maps:

- an individual $\mathtt{a}$ to an element of $\mathtt{a}^{\mathcal{I}} \in \Delta^{\mathcal{I}}$,
- a named class A to a membership function $\mathsf{A}^{\mathcal{I}} : \Delta^{\mathcal{I}} \rightarrow [0, 1]$, and
- a named property R to a membership function $R^{\mathcal{I}} : \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \rightarrow [0, 1]$.

Using the fuzzy set theoretic operations [6], fuzzy interpretations can be extended to interpret f-DL-Lite class and property descriptions. Following Straccia [12], we use the Lukasiewicz negation, c(a)=1-a and the Gödel t-norm for interpreting conjunctions, $t(a, b) = \min(a, b)$. The semantics of f-DL-Lite class and property descriptions, and f-DL-Lite axioms are depicted in Table 1. Given the above semantics, it is obvious that crisp assertions $B(a)$, $R(a, b)$ are special forms of fuzzy assertions where $n = 1$.

Syntax	**Semantics**
$\exists R$	$(\exists R)^{\mathcal{I}}(o_1) = \sup_{o_2 \in \Delta^{\mathcal{I}}} \{R^{\mathcal{I}}(o_1, o_2)\}$
$\neg B$	$(\neg B)^{\mathcal{I}}(o) = 1 - B^{\mathcal{I}}(o)$
$C_1 \sqcap C_2$	$(C_1 \sqcap C_2)^{\mathcal{I}}(o) = t(C_1^{\mathcal{I}}(o), C_2^{\mathcal{I}}(o))$
R^-	$(R^-)^{\mathcal{I}}(o_2, o_1) = R^{\mathcal{I}}(o_1, o_2)$
$B \sqsubseteq C$	$\forall o \in \Delta^{\mathcal{I}}, B^{\mathcal{I}}(o) \leq C^{\mathcal{I}}(o)$
$\mathsf{Func}(R)$	$\forall o_1 \in \Delta^{\mathcal{I}}, \sharp\{o_2 \mid R^{\mathcal{I}}(o_1, o_2) > 0\} = 1$
$B(\mathtt{a}) \geq n$	$B^{\mathcal{I}}(\mathtt{a}^{\mathcal{I}}) \geq n$
$R(\mathtt{a}, \mathtt{b}) \geq n$	$R^{\mathcal{I}}(\mathtt{a}^{\mathcal{I}}, \mathtt{b}^{\mathcal{I}}) \geq n$

Table 1. Semantics of f-DL-Lite class and property descriptions, and f-DL-Lite axioms

3 Querying f-DL-Lite Ontologies

In this section, we introduce two query languages for f-DL-Lite ontologies. The first language extends conjunctive queries with thresholds for atoms in queries. This is motivated by the extension of the instance checking problem which considers a single fuzzy assertion. The second language is a general fuzzy query language, which is a general form of query languages, such as the fuzzy threshold query language and the query language proposed in [12].

3.1 Two New Query Languages

Threshold Queries As noted in [1] in DL-Lite the instance checking problem is a special case of conjunctive queries. Since f-DL-Lite extends DL-Lite with fuzzy assertions, it would be natural for a query language to allow users to specify fuzzy assertions for atoms in queries. Thus, we define *conjunctive threshold queries* (CTQ) which extend atoms $\mathsf{A}(v), R(v_1, v_2)$ in conjunctive queries of the form (1) into the following forms $\mathsf{A}(v) \geq t_1, R(v_1, v_2) \geq t_2$, where $t_1, t_2 \in (0, 1]$ are thresholds.

Example 1. We can query models who are tall with a degree no less than 0.7 and light with a degree no less than 0.8 with the following conjunctive threshold query:

$$q(v) \leftarrow \mathsf{Model}(v) \geq 1, \mathsf{Tall}(v) \geq 0.7, \mathsf{Light}(v) \geq 0.8.$$

It is obvious that threshold queries are more flexible than queries of the form (1) in that users can specify different thresholds for different atoms in their queries.

Formally, given an f-DL-Lite ontology $\mathcal{O}$, a conjunctive threshold query q_T and an evaluation $[X \mapsto S]$, we say $\mathcal{O}$ entails q_T (denoted as $\mathcal{O} \models_{\mathrm{T}} q_T$) if every interpretation $\mathcal{I}$ of $\mathcal{O}$ satisfies the following condition: for each atom $\mathsf{A}(v) \geq t_1$ $(R(v_1, v_2) \geq t_2)$ of q_T, we have $\mathsf{A}^{\mathcal{I}}(v)_{X \mapsto S} \geq t_1$ (resp. $R^{\mathcal{I}}(v_1, v_2)_{X \mapsto S} \geq t_2$). In this case, S is called a *solution* of q_T. A disjunctive threshold query (DTQ) is a set of conjunctive threshold queries sharing the same head.

General Fuzzy Queries Since f-DL-Lite associates assertions with degrees of truth, another useful feature for its query language is to associate degrees of truth with answers in answer sets of queries over f-DL-Lite ontologies. In threshold queries, an evaluation $[X \mapsto S]$ either satisfies the query entailment or not; hence, answers of such queries are crisp. In this subsection, we introduce general fuzzy queries which allow fuzzy answers. Syntactically, general fuzzy conjunctive queries (GFCQ) extend the atoms $\mathsf{A}(v), R(v_1, v_2)$ of conjunctive queries of the form (1) into ones with the following form $\mathsf{A}(v) : k_1, R(v_1, v_2) : k_2$, where $k_1, k_2 \in (0, 1]$ are degrees. These syntactic extensions are similar with the ones proposed for fuzzy-SWRL in [8]. Thus, the existential quantifier is interpreted as sup, while we leave the semantics of the conjunction (G) and that of the degree-associated atoms (a) open. To simplify the presentation of the semantics, we use a unified representation $atom_i(\bar{v})$ for atoms in general fuzzy conjunctive queries.

Given an f-DL-Lite ontology $\mathcal{O}$, an interpretation $\mathcal{I}$ of $\mathcal{O}$, a general fuzzy conjunctive query q_F and an evaluation $[X \mapsto S]$, the degree of truth of q_F under $\mathcal{I}$ is $d = \sup_{S' \in \Delta^{\mathcal{I}} \times \ldots \times \Delta^{\mathcal{I}}} \{G_{i=1}^{n}\ a(k_i, atom_i^{\mathcal{I}}(\bar{v})_{[X \mapsto S, Y \mapsto S']})\}$, where k_i ($1 \leq i \leq n$) and $atom_i$ are as mentioned above, G is the semantic function for conjunctions and a is the semantic function for degree-associated atoms. $S : d$ is called a *candidate solution* of q_F. When $d > 0$, $S : d$ is called a *solution* of q_F. Furthermore, the semantic functions should satisfy the following condition: If $atom_i^{\mathcal{I}}(\bar{v})_{[X \mapsto S, Y \mapsto S']} = 0$ for all possible $S', d = 0$.

A general fuzzy disjunctive query (GFDQ) is a set of general fuzzy conjunctive queries sharing the same head. The disjunction is interpreted as the s-norm (u) of disjuncts. In what follows, we give some example of the semantic functions for conjunctions and degree-associated atoms.

1. Fuzzy threshold queries: If we use t-norms (t) as the semantic function for conjunctions and R-implications (ω_t) as the semantic function for degree-associated atoms, we get fuzzy threshold queries, in which the degree of truth of q_F under $\mathcal{I}$ is $d = \sup_{S' \in \Delta^{\mathcal{I}} \times \ldots \times \Delta^{\mathcal{I}}} \{t_{i=1}^{n}\ \omega_t(k_i, atom_i^{\mathcal{I}}(\bar{v})_{[X \mapsto S, Y \mapsto S']})\}$. Given some S', if for all atoms we have $atom_i^{\mathcal{I}}(\bar{v})_{[X \mapsto S, Y \mapsto S']} \geq k_i$, since $\omega_t(x, y) = 1$ when $y \geq x$ [6], we have $d = 1$; this corresponds to threshold queries introduced earlier.
2. Straccia's query language [12]: It is a special case of fuzzy threshold query language, where all $k_i = 1$. Since $\omega_t(1, y) = y$ [6], the degree of truth of q_F under $\mathcal{I}$ is $d = \sup_{S' \in \Delta^{\mathcal{I}} \times \ldots \times \Delta^{\mathcal{I}}} \{t_{i=1}^{n}\ atom_i^{\mathcal{I}}(\bar{v})_{[X \mapsto S, Y \mapsto S']}\}$.
3. Fuzzy aggregation queries: if we use fuzzy aggregation functions [6], such as $G(x) = \sum_{i=1}^{n}\ x_i$, for conjunctions and $a(k_i, y) = \frac{k_i}{\sum_{i=1}^{n}\ k_i} * y$ as the semantic function for degree-associated atoms, we get fuzzy aggregation queries, in which the degree of truth of q_F under $\mathcal{I}$ is $d = \sup_{S' \in \Delta^{\mathcal{I}} \times \ldots \times \Delta^{\mathcal{I}}} \frac{\sum_{i=1}^{n}\ k_i * atom_i^{\mathcal{I}}(\bar{v})_{[X \mapsto S, Y \mapsto S']}}{\sum_{i=1}^{n}\ k_i}$.
4. Fuzzy weighted queries: If we use generalised weighted t-norms as the semantic function for conjunction, we get fuzzy weighted queries, in which the degree of truth of q_F under $\mathcal{I}$ is $d = \sup_{S' \in \Delta^{\mathcal{I}} \times \ldots \times \Delta^{\mathcal{I}}} \{\min_{i=1}^{n}\ u(\overline{k} - k_i t(\overline{k}, atom_i^{\mathcal{I}}(\bar{v})_{[X \mapsto S, Y \mapsto S']}))\}$, where $\overline{k} = \max_{i=1}^{n} k_i$ and u is a t-conorm (fuzzy union). The main idea of this type of queries is that they provide an aggregation type of operation, on the other hand an entry with a low value for a low-weighted criterion should not be critically penalized. Moreover, lowering the weight of a criterion in the query should not lead to a decrease of the relevance score, which should mainly be determined by the high-weighted criteria. For more details see [3].

3.2 Query Answering

This sub-section provides algorithms to answer the two kinds of queries (presented in the previous sub-section) over f-DL-Lite ontologies.

Algorithms for answering queries in f-DL-Lite mainly consist of four steps (like the algorithm for crisp DL-Lite [1]): (i) normalisation of the set $\mathcal{T}$ of the class axioms of $\mathcal{O}$ by the procedure $\mathsf{Normalise}(\mathcal{T})$, which returns the normalised set $\mathcal{T}'$ of class axioms; (ii) normalisation and storage of the set $\mathcal{A}$ of individual axioms in $\mathcal{O}$ by the procedure $\mathsf{Store}(\mathcal{A})$ that normalise $\mathcal{A}$ and returns the relational database $\mathrm{DB}(\mathcal{A})$ of $\mathcal{A}$, as well as checking the consistency of $\mathcal{O}$ by the procedure $\mathsf{Consistency}(\mathcal{O}, \mathcal{T}')$; (iii) reformulation of the input query q against the normalised set $\mathcal{T}$ of the class axioms by the procedure $\mathsf{PerfectRef}(q, \mathcal{T}')$, which returns a set Q of (conjunctive) queries; (iv) transformation of the set Q of (conjunctive) queries into SQL queries by the procedure $\mathsf{SQL}(Q)$, as well as the evaluation of $\mathsf{SQL}(Q)$ by the procedure $\mathsf{Eval}(\mathsf{SQL}(Q), \mathrm{DB}(\mathcal{A}))$.

Answering Threshold Queries Given an f-DL-Lite ontology $\mathcal{O}$, a conjunctive threshold query q_T, the procedure $\mathsf{Answer_T}(\mathcal{O}, q_T)$ computes the solutions of q_T w.r.t. $\mathcal{O}$, following the above steps (i) - (iv).

Algorithm A-1: $\mathsf{Answer_T}(\mathcal{O}, q_T)$

1: $\mathcal{T}$ = Class-Axioms($\mathcal{O}$)
2: $\mathcal{T}'$ = Normalise($\mathcal{T}$) //normalisation of class axioms
3: $\mathcal{A}$ = Individual-Axioms($\mathcal{O}$)
4: DB($\mathcal{A}$) = Store($\mathcal{A}$) //normalisation and storage of individual axioms
5: **if** Consistency($\mathcal{O}, \mathcal{T}'$) = **false then**
6: **return** *inconsistent* //$\mathcal{O}$ is inconsistent
7: **end if**
8: **return** Eval($\mathsf{SQL_T}(\mathsf{PerfectRef_T}(q_T, \mathcal{T}'))$, DB($\mathcal{A}$))

Algorithm A-2: $\mathsf{SQL_T}$(Q)

1: $QS := \emptyset$
2: **for** every query q in Q **do**
3: sc:=Select-Clause(q) //construct the select-clause of q
4: fc:=From-Clause(q) //construct the from-clause of q
5: wc1:=WC-Binding(q) //construct the part of the where-clause about binding
6: wc2:=WC-Threshold(q) //construct the part of the where-clause that relates to thresholds
7: $QS := QS \cup$ Construct-SQL(sc,fc,wc1,wc2)
8: **end for**
9: **return** QS

Theorem 1. *Let $\mathcal{O}$ be an f-DL-Lite ontology, q_T a conjunctive threshold query and S a tuple of constants. S is a solution of q_T w.r.t. $\mathcal{O}$ iff $S \in \mathsf{Answer_T}(\mathcal{O}, q_T)$.*

See the online TR$^{(*)}$ for detailed explanations of the algorithms A-1 and A-2, as well as the proof of Theorem 1.

Answering General Fuzzy Queries Similarly, given an f-DL-Lite ontology $\mathcal{O}$, a general fuzzy conjunctive query q_F, the procedure $\mathsf{Answer_F}(\mathcal{O}, q_F)$ computes the solutions of q_F w.r.t. $\mathcal{O}$.

Algorithm A-3: $\mathsf{Answer_F}(\mathcal{O}, q_F, a, G)$

1: $\mathcal{T} = \mathsf{Class\text{-}Axioms}(\mathcal{O})$
2: $\mathcal{T}' = \mathsf{Normalise}(\mathcal{T})$ //normalisation of class axioms
3: $\mathcal{A} = \mathsf{Individual\text{-}Axioms}(\mathcal{O})$
4: $\mathrm{DB}(\mathcal{A}) = \mathsf{Store}(\mathcal{A})$ //normalisation and storage of individual axioms
5: **if** $\mathsf{Consistency}(\mathcal{O}, \mathcal{T}') =$ **false then**
6: **return** *inconsistent* //$\mathcal{O}$ is inconsistent
7: **end if**
8: $q = \mathsf{Remove\text{-}Degrees}(q_F)$ //q is transformed from q_F by removing the degrees from q_F
9: **return** $\mathsf{Cal}(q_F, \mathsf{EvalSQL}(\mathsf{PerfectRef}(q, \mathcal{T}')), \mathrm{DB}(\mathcal{A})), a, G)$

Algorithm A-4: $\mathsf{Cal}(q_F, SS, a, G)$

1: $ANS := \emptyset$
2: **for** every tuple $S \in SS$ **do**
3: $ANS := ANS \cup \mathsf{CalSoln}(q_F, S, a, G)$ //Calculate the solution $S : d$ based on the semantic functions a and G
4: **end for**
5: **return** ANS

Theorem 2. *Let $\mathcal{O}$ be an f-DL-Lite ontology, q_F a general fuzzy conjunctive query and $S : d$ a pair of a tuple of constants together with a truth degree, a a semantic function for conjunctions and G a semantic function for degree-associated atoms. $S : d$ is a solution of q_F w.r.t. $\mathcal{O}$ iff $(S : d) \in \mathsf{Answer_F}(\mathcal{O}, q_F, a, G)$.*

See the online TR$^{(*)}$ for detailed explanations of the algorithms A-3 and A-4, as well as the proof of Theorem 2.

4 Implementation and Evaluation

The fuzzy DL-Lite knowledge base was implemented by extending the DL-Lite knowledge base system used in ONTOSEARCH2 [7]. In ONTOSEARCH2 we slightly extend SPARQL to make fuzzy queries. In this section, we mainly focus on the evaluations of the performance. Please refer to the online TR $^{(*)}$ for more details of the fuzzy extension of SPARQL.

We evaluated the performance of our system by modifying the Lehigh University Benchmark [5] to include fuzzy concepts, and restricted the semantic complexity of the underlying ontology to that of DL-Lite. This allowed us to create data sets of arbitrary size. Comparative benchmarking was performed with ONTOSEARCH2, this is a non-fuzzy implementation of DL-Lite which allowed us to determine the overhead of a fuzzy query compared to a normal DL-Lite query. We added two fuzzy concepts to the ontology, "Busy" and "Famous". The first of these was determined by the number of courses taught or taken by a member of staff or student, the second is determined by the number of papers published. The values are calculated using the s-shaped curve functions $k_f(n)$ to calculate the fuzzy value for fame given n papers published, and $k_b(n)$ to calculate the fuzzy value for busyness given n courses taken:

$$k_f(n) = \frac{2}{1+exp(-0.1n)} - 1 \qquad k_b(n) = \frac{2}{1+exp(-0.4n)} - 1$$

For GFCQs the we used a summation operation for a, a multiplication operation for G giving a semantics of weights.

To test the system we created dl-lite datasets containing 1, 10 and 50 universities, and processed these to include the fuzzy concepts described above. Two queries were created, the first simple instance retrieval of all "Famous" members of staff. In CTQ form this had a threshold of ≥ 0.5 in GFCQ form the query returned all staff members in order of Fame, and in DL-Lite form, this query simply returned all members of staff. The second query found all busy students which were taught by famous members of staff. The CTQ returned all students with a busyness ≥ 0.5 who were taught by staff with fame ≥ 0.5, the GFCQ returned a list of students sorted by a weight function based on their business and the fame of any members of staff who taught them, the DL-Lite query simply returned a list of all students who were taught by any member of staff. The results are shown in 2.

Table 2. Results of the fuzzy Lehigh University Benchmark queries

Query	$T[1]$ (ms)	$T[10]$ (ms)	$T[50]$ (ms)
CTQ-1	179	536	1061
GFCQ-1	220	683	1887
DL-Lite-1	152	422	891
CTQ-2	532	845	2922
GFCQ-2	520	973	3654
DL-Lite-2	494	892	2523

The performance of the fuzzy reasoner is in all cases close to the performance of the crisp case reasoner for query answering. With small data sets, it is has almost identical performance, particularly on more complex queries. As more data must be evaluated, the performance drops slightly.

5 Conclusion and Outlook

DL-based fuzzy ontology languages have attracted much attention the last years. That is mainly due to the fact that compared to other fuzzy formalisms, fuzzy ontology languages provide an expressive and yet efficient way to perform reasoning over a fuzzy knowledge. In this paper, we report on evaluations of efficient conjunctive query answering over fuzzy DL-Lite ontologies.

Although there have been quite a few work on fuzzy SQL, such as [4], the closest work to ours is Straccia's work on f-DL-Lite [12], since DL-Lite itself goes beyond relational databases. Our paper builds on Straccia's work [12], to further propose two new expressive query languages accompanied with query answering algorithms over f-DL-Lite not proposed in [12]. The first one is a simple generalization of the instance checking (entailment) problem for fuzzy DLs, while the second one consists of a very expressive fuzzy query language

which goes beyond traditional conjunctive queries used in [12]. Actually this query language can work as a framework and by giving different semantics to its parts we can create different fuzzy query languages. Finally, our preliminary evaluations indicate that the performance of the fuzzy query engine is at least in many cases close to the performance of the crisp query engine.

Acknowledgements

This research was partially supported by (1) the FP6 Network of Excellence EC project Knowledge Web (IST-2004-507842), (2) the Advanced Knowledge Technologies (AKT6) project which is sponsored by the UK Engineering and Physical Sciences Council under grant number GR/N15764/01 and (3) the FP6 EC project X-Media (FP6-26978). Giorgos Stoilos was also partially supported by project PENED 03ED475 2003, which is cofinanced 75% of public expenditure through EC - European Social Fund, 25% of public expenditure through Ministry of Development - General Secretariat of Research and Technology and through private sector, under measure 8.3 of OPERATIONAL PROGRAMME "COMPETITIVENESS" in the 3rd Community Support Programme.

References

1. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, , and R. Rosati. DL-Lite: Tractable description logics for ontologies. In *Proc. of AAAI 2005*, 2005.
2. D. Calvanese, G. De Giacomo, M. Lenzerini, R. Rosati, and G. Vetere. DL-Lite: Practical Reasoning for Rich DLs. In *Proc. of the DL2004 Workshop*, 2004.
3. A. Chortaras, Giorgos Stamou, and Andreas Stafylopatis. Adaptation of weighted fuzzy programs. In *Proc. of the International Conference on Artificial Neural Networks ICANN 2006*, pages 45–54. Springer, 2006.
4. J. Galindo, M. armen Aranda, J.L. Caro, A. Guevara, and A. Aguayo. Applying fuzzy databases and fsql to the management of rural accommodation. *Tourism Management*, 23:623–629(7), 2002.
5. Y. Guo, Z. Pan, and J. Heflin. LUBM: A Benchmark for OWL Knowledge Base Systems. *Journal of Web Semantics*, 3(2):158–182, 2005.
6. G. J. Klir and B. Yuan. *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Prentice-Hall, 1995.
7. J. Z. Pan, E. Thomas, and D. Sleeman. ONTOSEARCH2: Searching and Querying Web Ontologies. In *Proc. of WWW/Internet 2006*, pages 211–218, 2006.
8. J.Z. Pan, G. Stoilos, G. Stamou, V. Tzouvaras, and I. Horrocks. f-SWRL: A fuzzy extension of SWRL. *Journal on Data Semantics, special issue on Emergent Semantics*, 4090:28–46, 2006.
9. G. Stoilos, G. Stamou, V. Tzouvaras, J.Z. Pan, and I. Horrocks. The fuzzy description logic f-$\mathcal{SHIN}$. In *Proc. of the International Workshop on Uncertainty Reasoning for the Semantic Web*, 2005.
10. G. Stoilos, G. Stamou, V. Tzouvaras, J.Z. Pan, and I. Horrocks. Fuzzy OWL: Uncertainty and the semantic web. In *Proc. of the International Workshop on OWL: Experiences and Directions*, 2005.
11. U. Straccia. Reasoning within fuzzy description logics. *Journal of Artificial Intelligence Research*, 14:137–166, 2001.
12. U. Straccia. Answering vague queries in fuzzy DL-Lite. In *Proceedings of the 11th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, (IPMU-06)*, pages 2238–2245, 2006.