Deciding Query Entailment in Fuzzy Description Logic Knowledge Bases

Jingwei Cheng, Z.M. Ma, Fu Zhang, and Xing Wang

Northeastern University, Shenyang, 110004, China
cjingwei@gmail.com, mazongmin@ise.neu.edu.cn

Abstract. Existing fuzzy description logic (DL) reasoners either are not capable of answering conjunctive queries, or only apply to DLs with less expressivity. In this paper, we present an algorithm for answering expressive fuzzy conjunctive queries, which allows the occurrence of both lower bound and the upper bound of thresholds in a query atom, over the relative expressive DL, namely fuzzy $\mathcal{ALCN}$. Our algorithm is specially tailored for deciding conjunctive query entailment of negative role atoms in the form of $R(x, y) \leq n$ or $R(x, y) < n$ which, to the best of our knowledge, has not been touched on in other literatures.

1 Introduction

Description logics (DLs, for short) [1] are the logical foundation of the Semantic Web, which support knowledge representation and reasoning by means of the concepts and roles. However, the reasoning services that aim at accessing and querying the data underlying ontologies, such as *retrieval*, *realisation* and *instantiation*, are only in weak form and do not support complex queries (mainly conjunctive queries, CQs). Conjunctive queries originated from research in relational databases, and, more recently, have also been identified as a desirable form of querying DL knowledge bases (KBs). The first conjunctive query algorithm [2] over DLs was actually specified for the purpose of deciding conjunctive query containment for $\mathcal{DLR}_{reg}$. Recently, query entailment and answering have also been extensively studied both for tractable DLs [3][4] and for expressive DLs[5][6].

In order to capture and reason about vague and imprecise domain knowledge, there have been a substantial amount of work carried out in the context of fuzzy DLs. When querying over fuzzy DL KBs, as in the crisp case, same difficulties emerged in that existing fuzzy DL reasoners, such as fuzzyDL[7] and FiRE [8], are not capable of dealing with CQs either. In [9], A fuzzy extension of CARIN system[5] is provided, along with a decision procedure for answering union of conjunctive queries. Some other work mainly focuses on querying over lightweight ontologies, e.g. in [10][11]. However, these extensions allow only positive role atoms in a query, while the negative atoms are not touched on.

In this paper, we thus present a very first algorithm for answering expressive and fuzzy CQs, allowing in a query both positive atoms and negative atoms, over the relative expressive fuzzy DL, namely f-$\mathcal{ALCN}$.

S.S. Bhowmick, J. Küng, and R. Wagner (Eds.): DEXA 2009, LNCS 5690, pp. 830–837, 2009.

2 f-$\mathcal{ALCN}$

Definition 1. *(syntax) Let N_C, N_R, and N_I be countable infinite and pairwise disjoint sets of concept, role and individual names respectively. f-$\mathcal{ALCN}$ concepts (denoted by C and D) are formed out of concept names according to the following abstract syntax: $C, D \rightarrow \top|\bot|A|C \sqcap D|C \sqcup D|\neg C|\forall R.C|\exists R.C| \geq pR| \leq pR|$, where $A \in N_C$, $R \in N_R$, p is a nonnegative integer.*

An f-$\mathcal{ALCN}$ KB $\mathcal{K}$ can be partitioned into a terminological part called TBox and an assertional part called ABox, denoted by $\mathcal{K} = (\mathcal{T}, \mathcal{A})$. A TBox is a finite set of *general concept inclusion* axioms (GCIs) of the form $C \sqsubseteq D$, where C, D are concepts. An ABox consists of fuzzy assertions of the form $B(o) \bowtie n$, $R(o, o') \bowtie n$, or $o \neq o'$, where $o, o' \in N_I$, $\bowtie \in \{\geq, >, \leq, <\}$. We use $\rhd$ to denote $\geq$ or $>$, and $\lhd$ to denote $\leq$ or $<$. We call ABox assertions defined by $\rhd$ *positive assertions*, while those defined by $\lhd$ *negative assertions*. Moreover, for every operator $\bowtie$, we define (i) its symmetric operator $\bowtie^-$ as $\geq^-=\leq$, $>^-=<$, $\leq^-=\geq$, $<^-=>$, and (ii) its negation operator $\neg \bowtie$ as $\neg \geq=<$, $\neg >=\leq$, $\neg \leq=>$, $\neg <=\geq$.

Definition 2. *(semantics) The semantics of f-$\mathcal{ALCN}$ are provided by an interpretation, which is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$. Here $\Delta^{\mathcal{I}}$ is a non-empty set of objects, called the domain of interpretation, and $\cdot^{\mathcal{I}}$ is an interpretation function which maps different individual names into different elements in $\Delta^{\mathcal{I}}$, concept A into membership function $A^{\mathcal{I}}$: $\Delta^{\mathcal{I}} \rightarrow$ [0,1], role R into membership function $R^{\mathcal{I}}$:$\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \rightarrow$ [0,1]. The semantics of f-$\mathcal{ALCN}$ concepts and roles are depicted as follows.*

- $\top^{\mathcal{I}}(o) = 1 \qquad \bot^{\mathcal{I}}(o) = 0 \qquad (\neg C)^{\mathcal{I}}(o) = 1 - C^{\mathcal{I}}(o)$
- $(C \sqcap D)^{\mathcal{I}}(o) = \min\{C^{\mathcal{I}}(o), D^{\mathcal{I}}(o)\} \quad (C \sqcup D)^{\mathcal{I}}(o) = \max\{C^{\mathcal{I}}(o), D^{\mathcal{I}}(o)\}$
- $(\forall R.C)^{\mathcal{I}}(o) = \inf_{o' \in \Delta^{\mathcal{I}}}\{\max\{1 - R^{\mathcal{I}}(o, o'), C^{\mathcal{I}}(o')\}\}$
- $(\exists R.C)^{\mathcal{I}}(o) = \sup_{o' \in \Delta^{\mathcal{I}}}\{\min\{R^{\mathcal{I}}(o, o'), C^{\mathcal{I}}(o')\}\}$
- $(\geq pR)^{\mathcal{I}}(o) = \sup_{o_1,\ldots,o_p \in \Delta^{\mathcal{I}}} \min_{i=1}^{p}\{R^{\mathcal{I}}(o, o_i)\}$
- $(\leq pR)^{\mathcal{I}}(o) = \inf_{o_1,\ldots,o_{p+1} \in \Delta^{\mathcal{I}}} \max_{i=1}^{p+1}\{1 - R^{\mathcal{I}}(o, o_i)\}$

Given an interpretation $\mathcal{I}$ and an inclusion axiom $C \sqsubseteq D$, $\mathcal{I}$ is a model of $C \sqsubseteq D$, if $C^{\mathcal{I}}(o) \leq D^{\mathcal{I}}(o)$ for any $o \in \Delta^{\mathcal{I}}$, written as $\mathcal{I} \models A \sqsubseteq C$. Similarly, for ABox assertions, $\mathcal{I} \models B(o) \bowtie n$(resp. $\mathcal{I} \models R(o, o') \bowtie n$), iff $B^{\mathcal{I}}(o^{\mathcal{I}}) \bowtie n$(resp. $R^{\mathcal{I}}(o^{\mathcal{I}}, o'^{\mathcal{I}}) \bowtie n$). If an interpretation $\mathcal{I}$ is a model of all the axioms and assertions in a KB $\mathcal{K}$, we call it a model of $\mathcal{K}$. A KB is *satisfiable* iff it has at least one model. A KB $\mathcal{K}$ *entails* (logically implies) a fuzzy assertion φ, iff all the models of $\mathcal{K}$ are also models of φ, written as $\mathcal{K} \models \varphi$.

3 Querying Entailment Problems

3.1 Fuzzy Querying Language

Let N_V be a countable infinite set of variables and is disjoint with N_C, N_R, and N_I. A *term* t is either an individual name from N_I or a variable name from

N_V. Let C be a concept, R a role, and t, t' terms. An *fuzzy query atom* is an expression of the form $\langle C(t) \bowtie n\rangle$ or $\langle R(t,t') \bowtie m\rangle$, where n denotes the lower bound (which corresponds to $\rhd$) or upper bound (which corresponds to $\lhd$) of membership of being the term t a member of the fuzzy set C, m denotes the degree of membership of being the term pair (t,t') a member of the fuzzy role R. We refer to these two different types of atoms as *fuzzy concept atoms* and *fuzzy role atoms* respectively. We call fuzzy query atoms defined by $\rhd$ *fuzzy positive query atoms*, while those defined by $\lhd$ *fuzzy negative query atoms*. A *fuzzy conjunctive query* q is a conjunction of fuzzy query atoms which is of the form $q = \langle\rangle \leftarrow \wedge_{i=1}^{n}\langle at_i \bowtie n_i\rangle$, where $\langle at_i \bowtie n_i\rangle$ denotes the i-th fuzzy query atom of q. We use $\mathbf{Var}(q)$ to denote the set of variables occurring in q, $\mathbf{Ind}(q)$ to denote the set of individual names occurring in q, and $\mathbf{Term}(q)$ for the set of terms in q, where $\mathbf{Term}(q) = \mathbf{Var}(q) \cup \mathbf{Ind}(q)$.

The semantics of a fuzzy query is given in the same way as for the related fuzzy DL by means of interpretations consisting of an interpretation domain and a fuzzy interpretation function. Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a fuzzy interpretation of f-$\mathcal{ALCN}$, q be a fuzzy conjunctive query and t, t' terms in q, and $\pi : \mathbf{Var}(q) \cup \mathbf{Ind}(q) \rightarrow \Delta^{\mathcal{I}}$ a total function (also called an *assignment*) such that $\pi(a) = a^{\mathcal{I}}$ for each $a \in \mathbf{Ind}(q)$. We say $\mathcal{I} \models^{\pi} \langle C(t) \bowtie n\rangle$ if $C^{\mathcal{I}}(\pi(t)) \bowtie n$, $\mathcal{I} \models^{\pi} \langle R(t, t') \bowtie n\rangle$ if $R^{\mathcal{I}}(\pi(t), \pi(t')) \bowtie n$. If $\mathcal{I} \models^{\pi} at$ for all atom $at \in q$, we write $\mathcal{I} \models^{\pi} q$. If there is a π, such that $\mathcal{I} \models^{\pi} q$, we say $\mathcal{I}$ satisfies q, written as $\mathcal{I} \models q$. We call such a π a *match* of q in $\mathcal{I}$. If $\mathcal{I} \models q$ for each model $\mathcal{I}$ of a KB $\mathcal{K}$, then we say $\mathcal{K}$ entails q, written as $\mathcal{K} \models q$. The *query entailment problem* is defined as follows: given a knowledge base $\mathcal{K}$ and a query q, decide whether $\mathcal{K} \models q$.

3.2 Deciding Query Entailment

The idea behind our algorithm is that it tries to decide fuzzy query entailment problem by (i) constructing (a representation of) all the models of an f-$\mathcal{ALCN}$ KB $\mathcal{K}$, then (ii) checks each of them for a match of a given conjunctive query q.

To construct models of an f-$\mathcal{ALCN}$ KB, we work with a data structure called completion forest. A *completion forest* consists of a labelled directed graph, each node of which is the root of a *completion tree*. Each node x in a completion tree is labelled with a set $\mathcal{L}(x) = \{\langle C, \bowtie, n\rangle\}$, and each edge (x, y) is labelled with a set $\mathcal{L}(x, y) = \{\langle R, \bowtie, n\rangle\}$. A node x is called an *R-predecessor* of a node y (and y is called an *R-successor* of x), if for some R, $\mathcal{L}(x, y) = \{\langle R, \bowtie, n\rangle\}$, *ancestor* is the transitive closure of predecessor.

Starting with an f-$\mathcal{ALCN}$ KB $\mathcal{K} = \langle\mathcal{T}, \mathcal{A}\rangle$, the completion forest $\mathcal{F}$ is initialized such that it contains a root node o, with $\mathcal{L}(o) = \{\langle C, \bowtie, n\rangle \mid \langle C(o) \bowtie n\rangle \in \mathcal{A}\}$, for each individual name o occurring in $\mathcal{A}$, and an edge $\langle o, o'\rangle$ with $\mathcal{L}(\langle o, o'\rangle) = \{\langle R, \bowtie, n\rangle \mid \langle R(o, o') \bowtie n\rangle \in \mathcal{A}\}$, for each pair $\langle o, o'\rangle$ of individual names for which the set $\{R \mid R(o, o') \bowtie n \in \mathcal{A}\}$ is non-empty.

An initial completion forest is expanded according to a set of *expansion rules*(see [12] and [13] for details) that reflect the constructors allowed in f-$\mathcal{ALCN}$. The expansion stops when there is a conjugated pair, called a *clash*, occurs within a node label, or when no more rules are applicable. In the latter

case, the completion forest is called *complete*. Termination is guaranteed by a cycle-checking technique called *blocking*. The model that we can build from a complete and clash-free completion graph is called a *canonical model*. The expansion and blocking rules are such that we can build a model for the knowledge base from each complete and clash-free completion forest.

The query can be represented as a directed, labelled graph called a *query graph*. The nodes in the query graph correspond to the terms in q, and are labelled with a triple corresponds to a related fuzzy concept atom. The edges correspond to the role atoms in q and are labelled similarly.

In the following, we show how to check a completion forest for a match of a query graph. The following example may be helpful in illustrating our method. Given a query $q_1 = \langle\rangle \leftarrow \langle C(a) \geq n_1\rangle \wedge \langle R(a, y_1) \geq n_2\rangle \wedge \langle S(y_1, y_2) \geq n_3\rangle \wedge \langle D(y_2) \geq n_4\rangle$ and a KB $\mathcal{K}_1 = \{(\exists R.(\exists S.D) \sqcap C)(a) \geq n\}$ where $n \geq \max(\{n_i\})$ with $1 \leq i \leq 4$. We first build completion forest for $\mathcal{K}_1$. Initially, there is only one node a labelled with a triple $(\exists R.(\exists S.D) \sqcap C, \geq, n)$, then expansion rules are applied to label of nodes, and lead to the completion forest depicted in Fig. 1. The query graph of q_1 is shown in Fig. 2.

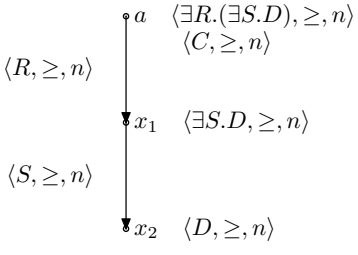

Fig. 1. The completion forest of $\mathcal{K}_1$

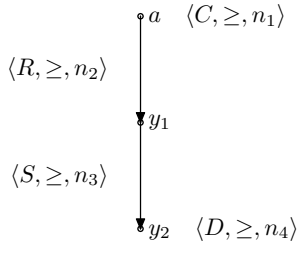

Fig. 2. The query graph of q_1

Now, we can compare the completion forest of $\mathcal{K}_1$ and query graph of q_1 for a match. We start from the root node in the query graph of q_1 and, at the same time, locate in the completion forest of $\mathcal{K}_1$ for a node whose label (a set of triples) *is more general than* the label (also a set of triples) of the node in the query graph. We say a label l_1 is subsumed by a label l_2 if for every triple tr in l_1, there exists a triple tr' in l_2, s.t. (i) tr' share the same concept or role name with tr, (ii) and the range identified by the sign of equality and the membership degrees in tr is *more general than* that of tr'. In Table 1, we list the conditions under which tr is more general than tr'. tr is listed in the leftmost column, while tr' is listed in the topmost row.

Then, we traverse the query graph along the outgoing edge to the next node, while in the completion forest we undeterministically select a branch to proceed (if any). The algorithm terminates when the leaf node in the query graph is encountered. In according to what they represent, we distinguish the nodes in the completion forest and query graph into *constant nodes* which represent individual names and *variable nodes* which represent variable names. During the traversal,

Table 1. The conditions under which tr is more general than tr'

	tr'			
tr	$(C,>,n)$	$(C,\geq,n)$	$(C,<,n)$	$(C,\leq,n)$
$(C,>,m)$	$m \leq n$	$m < n$	NULL	NULL
$(C,\geq,m)$	$m \leq n$	$m \leq n$	NULL	NULL
$(C,<,m)$	NULL	NULL	$m \geq n$	$m > n$
$(C,\leq,m)$	NULL	NULL	$m \geq n$	$m \geq n$

if (i) each constant node occurs in the query graph is mapped to a constant node in the completion forest that represents the same individual, and (ii) the label of every node and edge in query graph is more general than that of in the completion forest, we say that there is a match for a query in the knowledge base.

Given an f-$\mathcal{ALCN}$ knowledge base $\mathcal{K}$ and a query q, the algorithm answers "$\mathcal{K}$ entails q" if a match for q exists in each complete and clash-free completion forest and it answers "$\mathcal{K}$ does not entail q" otherwise.

The aforementioned idea is similar to the tableau algorithm for deciding f-$\mathcal{ALCN}$ KB satisfiability, but there are still three main problems need to be considered.

Firstly, for each concept C that occurs only in one of the fuzzy concept atoms of a query but not in the knowledge base, we must introduce into the knowledge base a concept inclusion axiom of the form $C \sqsubseteq C$. Clearly, this will exert no influence on the logical characteristics of the knowledge base, but it ensures that, for each node x in the completion forest, either $C(x) \rhd n$ or $C(x)\neg \rhd n$ holds.

Secondly, a f-$\mathcal{ALCN}$ KB may have infinitely many infinite models, whereas the tableau algorithm constructs only a subset of the finite models of the knowledge base. It can be shown that inspecting only the canonical models of the knowledge base is sufficient to decide query entailment.

Furthermore, we need to modify the standard blocking condition (for checking KB satisfiability) to make it applicable in the context of deciding query entailment. The standard blocking condition is to stop the cyclic expansion of a branch in the completion forest. For f-$\mathcal{ALCN}$ KBs, if there is a pair of nodes x and y such that x is an ancestor of y and the label of y is a subset of the label of x, then we say that x blocks y and no further expansion rules are applied to y. In the model obtained from the completion forest, a block corresponds to a cyclic path that links the predecessor of the blocked node y to the blocking node x. For query entailment, the blocking condition additionally has to take into account the length of the longest path in the query. We illustrate this by means of an example.

Let $\mathcal{K}_2 = (\mathcal{T}, \mathcal{A})$ be an f-$\mathcal{ALCN}$ KB with $\mathcal{T} = \{C \sqsubseteq \exists R.C\}$ and $\mathcal{A} = \{C(a) \geq n\}$ and let $q_2 = \langle\rangle \leftarrow \langle R(x,x) \geq n\rangle$ be a Boolean fuzzy conjunctive query. It is not hard to check that $\mathcal{K}_2 \not\models q_2$. Figure 3 illustrates the only complete and clash-free completion forest for $\mathcal{K}_2$ that the algorithm would construct. Figure 4 shows a representation of a corresponding canonical model $\mathcal{I}$. The loop in the model $\mathcal{I}$ occurs since the node x_1 blocks the node x_2 in the completion forest.

It is not hard to check that $\mathcal{I} \models \mathcal{K}_2$ and that the mapping $\pi : x \rightarrowtail x_1$ is such that $\mathcal{I} \models^{\pi} q_2$. Since the completion forest shown in Fig. 3 is the only completion forest that the algorithm would construct, we would wrongly conclude that $\mathcal{K}_2$ entails q_2.

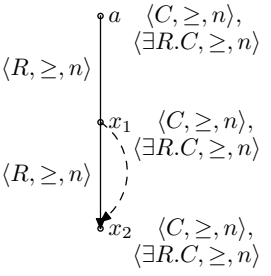

Fig. 3. A complete and clash-free completion forest for $\mathcal{K}_2$. The node x_2 is blocked by x_1, indicated by the dashed line.

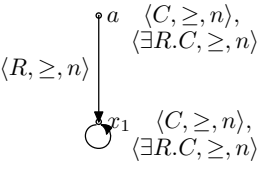

Fig. 4. A canonical model $\mathcal{I}$ for $\mathcal{K}_2$

Instead of using a pair of nodes x and y in the blocking definition such that x is an ancestor of y and the label of y is a subset of the label of x, the new blocking condition, first introduced in [5], requires two isomorphic trees (instead of just two nodes) such that the depth of the trees is equal to the number of fuzzy role atoms in the query. The leaves of the descendant tree are then considered as blocked. In our example, the query contains only one role atom. Hence, the revised blocking condition requires just two trees of depth one.

Figure 5 shows an abstraction of a complete and clash-free completion forest using the modified blocking condition. In the canonical model that corresponds to the completion forest (see Fig. 6), we have a cycle from the element that corresponds to the predecessor of the blocked node to the element that corresponds to the blocking node.

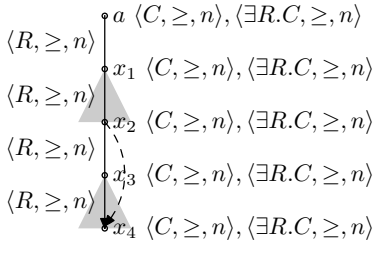

Fig. 5. A complete and clash-free completion forest for $\mathcal{K}_2$ under modified blocking condition

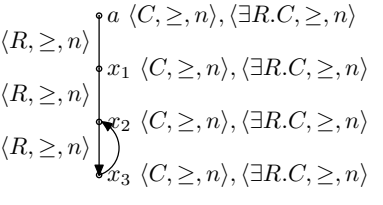

Fig. 6. A canonical model $\mathcal{I}$ for $\mathcal{K}_2$ corresponding to the completion forest in Fig 5

When building a model from a completion forest, one can, instead of building a cycle, also append infinitely many copies of the blocking tree and the path between the blocking and the blocked tree. In our example, this would result in a model that consists of an infinite R-chain.

Thirdly, the method cannot extend directly to the case where negative query atoms are allowed in a query. For example, given a KB $\mathcal{K}_3 = \{\forall R.D(a) \geq 0.6, D(b) < 0.3\}$ and a query $q_3 = \langle\rangle \leftarrow \langle R(a,y) \leq 0.5\rangle \wedge \langle D(y) \leq 0.4\rangle$, we can instinctively recognize that $\mathcal{K}_3$ entails q_3. However, existing algorithm cannot build a completion forest for $\mathcal{K}_3$, in which there exists a match of the query graph of q_3. We thus introduce two new rules for mending this. After applying these rules, the otherwise isolated two nodes are related by a fuzzy role R.

$\forall_{\rhd R}$-rule	$\exists_{\lhd R}$-rule
if 1.$\langle \forall R.C, \rhd, n\rangle \in \mathcal{L}(x)$, x is not blocked,	if 1. $\langle \exists R.C, \lhd, n\rangle \in \mathcal{L}(x)$, x is not blocked.
2.$\langle C, \neg\rhd, m\rangle \in \mathcal{L}(y)$,where $m \leq n$,	2. $\langle C, \neg\lhd, m\rangle \in \mathcal{L}(y)$, where $m \geq n$,
3.there is no $\langle R, \rhd^-, 1-n\rangle \in \mathcal{L}(x,y)$	3. there is no $\langle R, \lhd, n\rangle \in \mathcal{L}(x,y)$
then $\mathcal{L}(x,y) \rightarrow \mathcal{L}(x,y) \cup \langle R, \rhd^-, 1-n\rangle$.	then $\mathcal{L}(x,y) \rightarrow \mathcal{L}(x,y) \cup \langle R, \lhd, n\rangle$.

In our example, before applying $\forall_{\geq R}$ rule, the completion forest of $\mathcal{K}_3$ consists of two isolated nodes a and b labelled with $(\forall R.D, \geq, 0.6)$ and $(D, <, 0.3)$ respectively. The completion forest of $\mathcal{K}_3$ is shown in Fig. 7. The $\forall_{\geq R}$ rule then additionally adds an role assertion $\langle R(a,b) \leq 1 - 0.6\rangle$ (simplified as $\langle R(a,b) \leq 0.4\rangle$) into $\mathcal{K}_2$ and therefore connects the isolated a and b, resulting in the completion forest of $\mathcal{K}_3'$, which is also shown in Fig. 7. Note that the newly introduced role is depicted in Fig. 7 as a dashed line. We can easily find a match for the query graph of q_3(Fig. 8) in the completion forest of $\mathcal{K}_3'$.

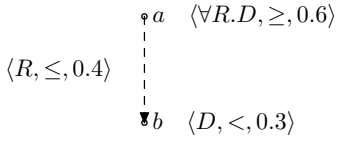

Fig. 7. The completion forest of $\mathcal{K}_3$ and $\mathcal{K}_3'$

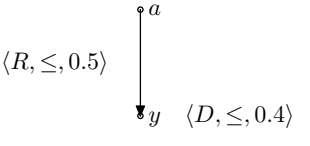

Fig. 8. The query graph of q_3

Theorem 1. *Let $\mathcal{K}$ be a f-$\mathcal{ALCN}$ KB, and $\mathcal{K}'$ a KB obtained by applying the $\forall_{\geq R}$ rule to K. The $\forall_{\geq R}$ rule is sound if a model $\mathcal{I}$ of $\mathcal{K}$ is also a model of $\mathcal{K}'$.*

Proof. $\forall_{\geq R}$: For each $\mathcal{I} \models \mathcal{K}$, we show that $\mathcal{I} \models \mathcal{K}'$, where $\mathcal{K}'$ is obtained by applying $\forall_{\geq R}$ rule to $\mathcal{K}$, i.e. $\mathcal{K}' = \mathcal{K} \cup \{\langle R(x,y) < 1 - n\rangle\}$. For any assertion $\langle \forall R.C(x) \geq n\rangle \in \mathcal{A}$, we have $\mathcal{I} \models \langle \forall R.C(x) \geq n\rangle$, i.e., $\inf_{x' \in \Delta^{\mathcal{I}}}\{\max\{1 - R^{\mathcal{I}}(x,x'), C^{\mathcal{I}}(x')\}\} \geq n$. Since $y \in \Delta^{\mathcal{I}}$, we have $\max\{1 - R^{\mathcal{I}}(x,y), C^{\mathcal{I}}(y)\} \geq n$. Since there is an additional assertion of $\langle C(y) < m\rangle$ with $m \leq n$, the $1 - R^{\mathcal{I}}(x,y) \geq n$ (or its equivalence $R^{\mathcal{I}}(x,y) \leq n$) holds. The $\forall_{>R}$ and $\exists_{\lhd R}$ rules can be proved accordingly.

4 Conclusions

In this paper, we have presented a preliminary result of conjunctive query entailment over an expressive fuzzy DL knowledge base. Further and ongoing research will focus on the query answering or query entailment problems of more expressive DLs and the complexity analysis for them.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (60873010).

References

1. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The description logic handbook: theory, implementation, and applications. Cambridge University Press, New York (2003)
2. Calvanese, D., De Giacomo, G., Lenzerini, M.: On the decidability of query containment under constraints. In: PODS 1998, pp. 149–158 (1998)
3. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The dl-lite family. J. of Automated Reasoning 39(3), 385–429 (2007)
4. Rosati, R.: On conjunctive query answering in EL. In: DL 2007, CEUR Electronic Workshop Proceedings (2007)
5. Levy, A.Y., Rousset, M.C.: Combining horn rules and description logics in carin. Artif. Intell. 104(1-2), 165–209 (1998)
6. Glimm, B., Horrocks, I., Lutz, C., Sattler, U.: Conjunctive query answering for the description logic shiq. In: IJCAI, pp. 399–404 (2007)
7. Bobillo, F., Straccia, U.: fuzzydl: An expressive fuzzy description logic reasoner. In: FUZZ-IEEE 2008, June 2008, pp. 923–930 (2008)
8. Stoilos, G., Simou, N., Stamou, G.B., Kollias, S.D.: Uncertainty and the semantic web. IEEE Intelligent Systems 21(5), 84–87 (2006)
9. Mailis, T., Stoilos, G., Stamou, G.: Expressive reasoning with horn rules and fuzzy description logics. In: Marchiori, M., Pan, J.Z., Marie, C.d.S. (eds.) RR 2007. LNCS, vol. 4524, pp. 43–57. Springer, Heidelberg (2007)
10. Straccia, U.: Answering vague queries in fuzzy dl-lite. In: IPMU 2006, pp. 2238–2245 (2006)
11. Pan, J.Z., Stamou, G.B., Stoilos, G., Taylor, S., Thomas, E.: Scalable querying services over fuzzy ontologies. In: WWW, pp. 575–584 (2008)
12. Stoilos, G., Stamou, G., Pan, J., Tzouvaras, V., Horrocks, I.: Reasoning with very expressive fuzzy description logics. JAIR 30(8), 273–320 (2007)
13. Stoilos, G., Straccia, U., Stamou, G.B., Pan, J.Z.: General concept inclusions in fuzzy description logics. In: ECAI, pp. 457–461 (2006)