Approximate Querying of XML Fuzzy Data

Patrice Buche[1], Juliette Dibie-Barthélemy[1,2], and Fanny Wattez[1,2]

[1] Mét@risk MIA INRA
16, rue Claude Bernard,
F-75231 Paris Cedex 05
[2] UMR MIA INA P-G/INRA
16, rue Claude Bernard,
F-75231 Paris Cedex 05
{Patrice.Buche, Juliette.Dibie, Fanny.Wattez}@inapg.inra.fr

Abstract. The MIEL++ system integrates data expressed in two different formalisms: a relational database and an XML database. The XML database is filled with data semi-automatically retrieved from the Web, which have been semantically enriched according to the ontology used in the relational database. These data may be imprecise and represented as possibility distributions. The MIEL++ querying system scans the two databases simultaneously in a transparent way for the end-user. To scan the XML database, the MIEL query is translated into an XML tree query. In this paper, we propose to introduce flexibility into the query processing of the XML database, in order to take into account the imperfections due to the semantic enrichment of its data. This flexibility relies on fuzzy queries and query rewriting which consists in generating a set of approximate queries from an original query using three transformation techniques: deletion, renaming and insertion of query nodes.

1 Introduction

Numerous approaches have been proposed in the bibliography to introduce flexibility in the comparison between an XML tree query and XML data trees. The first one is based on the encoding of the XML data trees [6]. This method only permits the introduction of intermediate nodes in the tree query structure in order to carry out the comparison with the data trees. The second approach [1] is based on the rewriting of the XML tree query. It permits the introduction, renaming and deletion of nodes in the query. The third one [12] is a combination of the previous two: the data are encoded and the query is rewritten. It provides an accurate computation of the transformation cost between the query and the data. But, it is very difficult to use because it requires one to redefine the management of the index and data encoding in the XML Database Management System. In a fourth approach [2], fuzzy predicates are introduced into the query to express flexible selection conditions and to perform fuzzy subtree matching. But it does not take into account suppression and renaming of nodes. In this paper, we propose a new XML querying system. It combines the flexibility provided by the

H. Legind Larsen et al. (Eds.): FQAS 2006, LNAI 4027, pp. 26–38, 2006.

use of fuzzy sets to represent the user's preferences in an XML query and the flexibility of XML query rewriting (including insertion, deletion and renaming of nodes) to perform an approximate comparison between an XML tree query and an XML data tree. Moreover, it supports XML imprecise data expressed as possibility distributions which is also an original contribution because few research has been done in modeling and querying imprecise XML data. To the best of our knowledge (see [10] for a recent synthesis), only imprecise data and probabilistic data modeling in XML have been proposed. Our querying system is fully compatible with XML querying standards since the final rewriting of the queries is performed in XQuery language (http://www.w3.org/XML/Query/).

This work is realised in the framework of a system development whose aim is to integrate heterogeneous data sources. Two approaches are generally considered to solve this problem: the data warehouse approach [14] in which data are transformed to be stored in one global schema and mediated architectures [15] where the data remain stored in the original sources, the mapping between the global integration schema and the schemas of original sources being carried out by wrappers. In our system, we propose data integration based on a *mediated architecture*. More precisely, we use a global schema to integrate data expressed in two different formalisms: a relational database and an XML database. This architecture, called MIEL++ [4], is close to a *Global as Views* approach, in which the global schema is defined in terms of the local schemas to be integrated, as in the TSIMMIS [13] system. An original aspect of our approach is that our XML database is comparable to a data warehouse, since it contains data, semi-automatically retrieved from the Web, which have been modified in order to be expressed in the same vocabulary and semantic relations as the ones used in the relational database [9]. These data, called SML data, may be imprecise [3] and represented as possibility distributions [17]. Moreover, in order to avoid empty answers, MIEL++ querying system proposes to the end-user to express selection criteria by means of fuzzy sets used as expression of preferences [3]. In [5], we have defined a wrapper which translates a MIEL query into an XML tree query to scan the XML database. We made the assumption that the XML data trees retrieved by the MIEL++ system have to fit exactly the structure of the XML tree query. This assumption does not permit the imperfections of the SML data to be taken into account. Therefore, we cannot make the assumption as in [11] that we know precisely the schema of the Web data sources we want to integrate. In this paper, we study the way of introducing more flexibility into the MIEL query processing of an XML database. This work is done in the framework of the development of a real data warehouse in an actual application domain: risk assessment in food safety.

In section 2, we briefly present firstly the fuzzy set framework that we use to represent imprecise data and preferences in the queries, secondly the MIEL query language and thirdly the way imprecise data are represented in an XML database. In section 3, we define a new wrapper which translates a MIEL query into a set of approximate XML queries. In section 4, we present the implementation and preliminary test results of this wrapper in a real application.

2 Backgrounds

2.1 Fuzzy Set Theory

In this article, we use the representation of fuzzy sets proposed in [16, 17].

Definition 1. A *fuzzy set* f on a definition domain $Dom(f)$ is defined by a membership function μ_f from $Dom(f)$ to $[0, 1]$ that associates the degree to which x belongs to f with each element x of $Dom(f)$.

The fuzzy set formalism can be used in two ways: (i) in the databases, in order to represent imprecise data expressed in terms of possibility distributions or (ii) in the queries, in order to represent fuzzy selection criteria which express the preferences of the end-user. A fuzzy set can be defined on a continuous or discrete definition domain (see Fig. 1). In order to answer queries in databases involving fuzzy sets, we must be able to compare fuzzy sets. Two scalar measures are classically used in the fuzzy set theory to evaluate the compatibility between an imprecise datum and end-user's preferences: a possibility degree of matching [17] and a necessity degree of matching [8]. In this paper, for simplicity reasons, we will only use the *possibility degree* of matching which is defined below.

Definition 2. Let f and g be two fuzzy sets defined on the same definition domain Dom, representing respectively a selection criterion and an imprecise datum, μ_f and μ_g being their membership functions. The *possibility degree of matching* between f and g is $\Pi(f, g) = sup_{x \in Dom}(min(\mu_f(x),\ \mu_g(x)))$. *The selection criterion is satisfied if and only if* $\Pi(f, g) > 0$.

2.2 The MIEL Query Language

In the MIEL++ system, the query processing is done through the MIEL query language. This query processing relies on the ontology of the databases, called *MIEL++ ontology*, which contains the vocabulary used by the end-users to express their queries. We first present the MIEL++ ontology and then, we introduce the MIEL query language by presenting the queries and their answers.

The MIEL++ ontology. The ontology is notably composed of: (1) a taxonomy of terms including the set of attributes which can be queried on by the

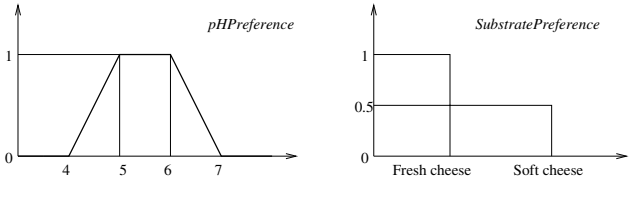

Fig. 1. An example of a continuous fuzzy set pHPreference noted [4,5,6,7] and a discrete fuzzy set SubstratePreference noted (1/Fresh cheese + 0.5/Soft cheese)

end-user, and their corresponding definition domains. Each attribute has a definition domain which can be numeric, "flat" symbolic (unordered constants) or hierarchized symbolic (constants partially ordered by the "kind-of" relation); (2) a relational schema, which corresponds to the schema of the relational database of the MIEL++ system. That relational schema is composed of a set of signatures of the possible relations between the terms of the taxonomy.

The queries. In the MIEL query language, a query is asked in a view, which is a pre-written query allowing the system to hide the complexity of the database schema. A view is characterized by its set of queryable attributes and by its actual definition. A query is then an instanciation of a given view by the end-user, by specifying, among the set of queryable attributes of the view, which are the selection attributes and their corresponding searched values, and which are the projection attributes of the query.

Definition 3. A *query* Q asked on a view V defined on n attributes $\{a_1, \ldots, a_n\}$ is defined by $Q = \{V, S, C\}$ where $S \subseteq \{a_1, \ldots, a_n\}$ represents the set of the projection attributes and where $C = \{c_1, \ldots, c_m\}$ is the set of conjunctive selection criteria. Each selection criterion c_i is restricted to an equality $< a_i = v_i >$ between an attribute $a_i \in \{a_1, \ldots, a_n\}$ and its searched value v_i which can be crisp or fuzzy and must be defined on a subset of the definition domain of a_i.

When the fuzzy value of a selection attribute has a hierarchized symbolic definition domain, the fuzzy set used to represent the fuzzy value can be defined on a subset of this definition domain. We consider that such a fuzzy set defines degrees implicitly on the whole definition domain of the selection attribute. In order to take those implicit degrees into account, the *fuzzy set closure* has been defined in [3]. Intuitively, the degrees are propagated to more specific values of the hierarchized symbolic domain. The fuzzy set closure is systematically used when a comparison involves two fuzzy sets (an expression of end-users' preferences and an imprecise datum) defined on a hierarchical definition domain.

The answers. An answer to a query Q must (1) satisfy all the selection criteria of Q in the meaning of definition 4 given below and (2) associate a constant value with each projection attribute of Q.

Definition 4. Let $< a = v >$ be a selection criterion and v' a value of the attribute a stored in the databases. The selection criterion $< a = v >$ is satisfied with the possibility degree $\Pi(cl(v), cl(v'))$ in the meaning of definition 2 where the cl function corresponds to the fuzzy set closure.

As the selection criteria of a query are conjunctive, we use the *min* operator to compute the adequation degree associated with the answer.

Definition 5. An *answer* to a query $Q = \{V, S, C\}$ is a set of tuples, each of the form $\{v_1, \ldots, v_l, ad_\Pi\}$, where $v_1, \ldots, v_l$ correspond to the crisp or fuzzy values associated with each projection attribute $a_i \in S$, where all the selection criteria $c_1, \ldots, c_m$ of Q are satisfied with the possibility degrees $\Pi_1, \ldots, \Pi_m$, and where ad_Π is the possibility degree of the answer to Q defined by: $ad_\Pi = min_{i=1}^{m}(\Pi_i)$.

2.3 The XML Database

The XML database has been built in the MIEL++ system in order to store information retrieved from the Web. It is a set of XML documents which may contain fuzzy values. We propose to model XML documents as fuzzy data trees [5, 4] which are data trees that allow fuzzy values to be represented. According to the definition of [7], a data tree is a triple (t, l, v) where t is a finite tree, l a labelling function that assigns a label to each node of t and v a partial value function that assigns a value to nodes of t. The couple (t, l) is called a labelled tree. The representation of fuzzy values relies on the fuzzy set formalism. For readability reasons in this paper, we only deal with discrete fuzzy sets. However, the contributions of this paper can easily be extended to continuous fuzzy sets.

Definition 6. A discrete fuzzy set f is represented by a data tree which is composed of a root labelled *DFS* and such that for each element x of $Dom(f)$, there exists a node labelled *ValF* that has two children labelled *Item* and *MD* (for Membership Degree) of respective values x and $\mu(x)$.

In a fuzzy data tree, the partial value function v can assign a crisp or a fuzzy value to a node, which is then called *crisp* or *fuzzy value node*.

Definition 7. A *fuzzy data tree* is a triple (t, l, v) where (t, l) is a labelled tree and v is a partial value function that assigns a value to the crisp and fuzzy value nodes of t. The value assigned to a crisp value node is an atomic value and the one assigned to a fuzzy value node is a data tree which conforms to definition 6.

A fuzzy data tree (t, l, v) is an *instance* [7] of a type tree (t_T, l_T) [1] if there exists a *strict type homomorphism* h from (t, l) to (t_T, l_T): (i) h preserves the root of t: $\text{root}(t_T) = h(\text{root}(t))$, (ii) h preserves the structure of t: whenever node m is a child of node n, $h(m)$ is a child of $h(n)$ and (iii) h preserves the labels of t: for each node n of t, $l(n)=l_T(h(n))$.

Example 1. *Figure 2 gives two examples of fuzzy data trees representing two lines of a Web data table. The crisp value node* originalVal *represents the original value of a* Product *in the Web data table and the fuzzy value node* finalVal *is a discrete fuzzy set representing the terms of the MIEL++ taxonomy associated with the original value. See subsection 4.1 for more details.*

3 The Approximate Query Processing of the XML Base

The XML database is filled with data semi-automatically extracted from the Web. Its integration into the MIEL++ system is possible thanks to the semantic enrichment of its data according to the ontology (see subsection 4.1 for more details). This section presents the XML subsystem which realizes the query processing of the XML database by means of the MIEL query language. In order

[1] A type tree is a labelled tree such that no node has two children labelled the same.

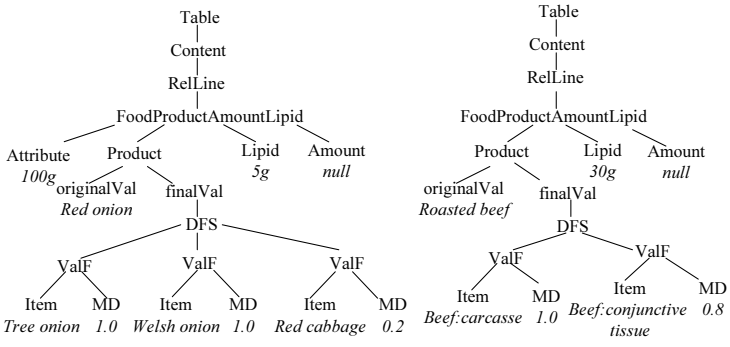

Fig. 2. Two fuzzy data trees

to take into account the imperfections coming from the semantic enrichment of the XML database, we propose to introduce flexibility into the query processing of the XML database using the following three techniques: deletion, renaming and insertion of query nodes. In this section, we define the notions of views, queries and answers of the MIEL query language in the XML subsystem.

3.1 The Views

The XML subsystem contains a set of views, which are built from the terms and the relations of the MIEL++ ontology and allow one to query the XML database. The information needed to perform approximate query processing is encoded at the view level. For each node of each view, except for the root of each view, we propose to define a deletion cost, renaming values and their costs.

Definition 8. A *view that conforms to a type tree* (t_T, l_T) is a 4-tuple $V=(t_V, l_V, w_V, c_V)$ where (t_V, l_V) is an instance of (t_T, l_T), w_V is a partial function that assigns the mark *ql* to crisp and fuzzy value nodes of t_V which are then queryable nodes, c_V is a partial function that assigns transformation information to each node and value node –except to the root– of t_V. The transformation information of a node n are represented by a couple of the form $(dc_n, \{(r_n^1, rc_n^1), \ldots, (r_n^p, rc_n^p)\})$ where dc_n is the deletion cost of the node n and $r_n^1, \ldots, r_n^p$ are the possible renamings of the node n with their corresponding costs $rc_n^1, \ldots, rc_n^p$.

Example 2. *The left side of figure 3 shows a view using the relation* FoodProductAmountLipid *involving three queryable attributes: the* product*, the* amount of the product *and the* quantity of lipid*. A deletion cost of* 1000 *(resp.* 100*) is associated with the node* FoodProductAmountLipid *(resp.* Amount*). A possible renaming in Relation (resp. Attribute) is associated with the node* FoodProductAmountLipid *(resp.* Amount*) with a renaming cost of* 50 *(resp.* 10*).*

3.2 The Queries

A query is built from a given view, where the end-user specifies: (i) among the set of queryable value nodes of the view, the selection and the projection value

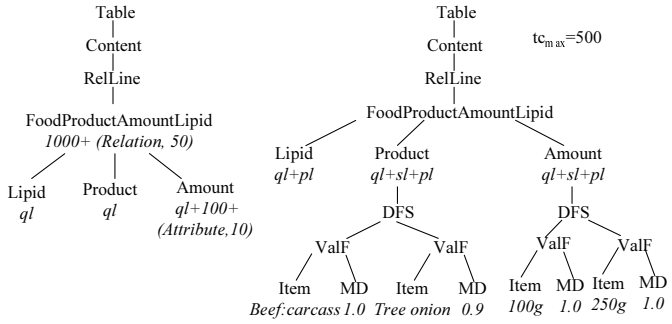

Fig. 3. An example of a view and a query defined on that view

nodes of the query and (ii) a transformation cost threshold which allows some transformations to be forbidden. The end-user can also modify the transformation information defined in the view.

Definition 9. A *query* that conforms to a type tree (t_T, l_T) is a 8-tuple $Q=(t_Q, l_Q, w_Q, c_Q, tc_{max}, p_Q, s_Q, ws_Q)$ where:

- (t_Q, l_Q, w_Q, c_Q) is a view that conforms to (t_T, l_T);
- tc_{max} is the maximum acceptable transformation cost of the query;
- p_Q is a partial function that assigns the mark *pl* to the queryable value nodes of the view, which are considered as the *projection value nodes*;
- s_Q is a partial function that assigns the mark *sl* to the queryable value nodes of the view, which are considered as the *selection value nodes*, also called selection criteria;
- ws_Q is a partial value function that assigns a value to the selection value nodes of the query, such that the value assigned to a crisp value node is an atomic value and the value assigned to a fuzzy value node is a data tree with a root labelled DFS which conforms to definition 6.

As defined in definition 3, the value v of a selection criterion $< a = v >$, a being a value node of the query, must be defined on a subset of the definition domain of a. When this value is fuzzy, the fuzzy selection criterion which expresses the end-user's preferences is represented by a fuzzy set.

Example 3. *The query Q of figure 3 (right side) expresses that the end-user wants to obtain the product, the amount of product and the quantity of lipid from the view using the relation* FoodProductAmountLipid*. The fuzzy value assigned to the selection criterion* Product *can be interpreted as "the end-user wants Beef: carcass as a product, but he/she also accepts Tree onion with a lower interest". The associated maximum transformation cost* tc_{max} *is 500.*

3.3 The Approximate Queries

The search for approximate answers to an XML query Q is done in two steps. The first step consists in generating potential approximated queries from the query

Q by means of combinations of the following two transformation rules: (i) every node and value node of Q with a deletion cost can be deleted, (ii) every node and value node with renaming values and costs can be renamed. Each generated approximate query must have at least one projection value node and one selection value node, non necessarily distinct. In the second step, a transformation cost is computed with each potential approximate query generated as the sum of the deletion costs of the deleted nodes and the renaming costs of the renamed nodes from the query Q. Only the approximate queries with a transformation cost lower than the transformation cost threshold of the query Q are kept and will be executed to find the approximate answers to the query Q.

Definition 10. An approximate query generated from a query Q=(t_Q, l_Q, w_Q, c_Q, tc_{max}, p_Q, s_Q, ws_Q) is a 7-tuple A=(t_A, l_A, w_A, p_A, s_A, ws_A, tc_A) where:

- there exists a *weak structural homomorphism* h from the nodes of t_A into nodes of t_Q: (i) h preserves the root of t_A: root(t_A) = h(root(t_Q)) and (ii) h preserves the ancestor-descendant relationship of t_A: whenever node m is a descendant of node n, $h(m)$ is a descendant of $h(n)$;
- l_A is a labelling function that assigns to each node n in t_A either the label assigned by l_Q to the image node $h(n)$ in t_Q or a renaming value belonging to the transformation information $c_Q(h(n))$ of the image node $h(n)$ in t_Q;
- for every value node n of t_A such that $w_Q(h(n)) = ql$, we have $w_A(n) = ql$ and, for the other value nodes of t_A, w_A is not defined;
- for every value node n of t_A such that $p_Q(h(n)) = pl$, we have $p_A(n) = pl$ and, for the other value nodes of t_A, p_A is not defined. Moreover, there must exist at least one value node n in t_A such that $p_A(n) = pl$;
- for every value node n of t_A such that $s_Q(h(n)) = sl$, we have $s_A(n) = sl$ and, for the other value nodes of t_A, s_A is not defined. Moreover, there must exist at least one value node n in t_A such that $s_A(n) = sl$;
- for every value node n of t_A such that $ws_Q(h(n))$ is defined, we have $ws_A(n) = ws_Q(h(n))$ and, for the other value nodes of t_A, ws_A is not defined;
- tc_A is the transformation cost of the query Q into the approximate query A: $tc_A = \sum_{i=1}^{l} dc_{n_d^i} + \sum_{i=1}^{q} rc_{n_r^i}$ where $c_Q(n)$=(dc_n, {(r_n^1, rc_n^1), …, (r_n^p, rc_n^p)}) is the transformation information of a node n of t_Q (see definition 8), {n_d^1, …, n_d^l} is the list of deleted nodes[2] of t_Q in t_A and {n_r^1, …, n_r^q} is the list of renamed nodes[3] of t_Q in t_A. Moreover, tc_A must be lower or equal to the maximum acceptable transformation cost tc_{max} of the query.

Example 4. *Figure 4 gives two examples of approximate queries generated from the query Q of figure 3. The first approximate query (left side) has been generated by renaming the node* Amount *to* Attribute*, the second one (right side) by deleting the node* Amount*.*

[2] The list of nodes of t_Q such that n_d^i has no antecedent in t_A by h.

[3] The list of nodes of t_Q such that n_r^i has an antecedent in t_A by h and $l_Q(n_r^i) \neq l_A(h^{-1}(n_r^i))$.

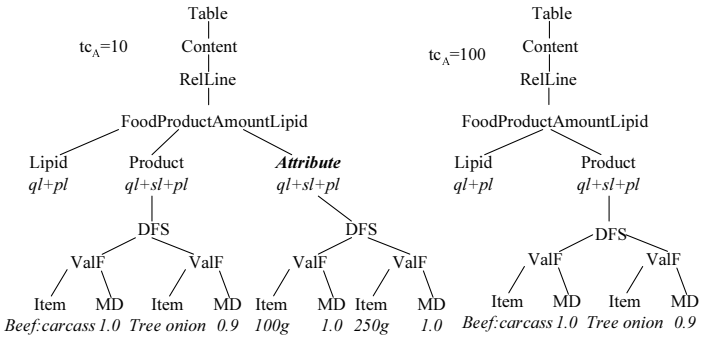

Fig. 4. Two examples of approximate queries generated from the query Q of figure 3

Remark 1. *Thanks to the representation of fuzzy values in a fuzzy data tree (see definition 7), there is no distinction between the deletion of a fuzzy value node and of a crisp value node: in both cases, the node and its value are deleted.*

3.4 The Answers

The approximate answer to an XML query Q is the union of the answers to the generated approximate queries from Q. While the generation of approximate queries from Q relies on the deletion and the renaming of nodes of Q, the computation of the answers to an approximate query A allows the insertion of nodes into A. An answer to an approximate query A (i) satisfies all the selection criteria of A in the meaning of definition 4 and (ii) associates a constant value with each projection value node of A. The search for the answers to an approximate query in an XML database is done through the valuation of the query on the fuzzy data trees of the database as defined below.

Definition 11. Let A=(t_A, l_A, w_A, p_A, s_A, ws_A, tc_A) be an approximate query generated from Q=(t_Q, l_Q, w_Q, c_Q, tc_{max}, p_Q, s_Q, ws_Q) which conforms to a type tree T=(t_T, l_T) and D=(t_D, l_D, v_D) be a fuzzy data tree instance of T. A *valuation* of A with respect to D is a mapping σ_D from t_A into t_D such that:

- σ_D is a *weak type homomorphism* from (t_A, l_A) into (t_D, l_D): σ_D is a weak structural homomorphism (see definition 10) and preserves the labels of t_A;
- σ_D satisfies each selection criterion n_s^i, $i \in [1, m]$, of A with the possibility degree $\Pi(ws_A(n_s^i), v_D(\sigma_D(n_s^i)))$.

The adequation degree of the fuzzy data tree D to the approximate query A through the valuation σ_D is $ad_{\Pi(D)}$=$min_{i\in[1,m]}(\Pi(ws_A(n_s^i),\ v_D(\sigma_D(n_s^i))))$.

An answer to an approximate query in the XML database is a set of tuples, where each tuple is composed of (i) a set of values given to each projection node, (ii) an adequation degree of the answer to the approximate query and (iii) a cost of transforming the initial query into the approximate query.

Definition 12. An *answer* to an approximate query A=(t_A, l_A, w_A, p_A, s_A, ws_A, tc_A) composed of m projection value nodes $n_p^1, \ldots, n_p^m$ in an XML database $\mathcal{W}$ is a set of tuples, each tuple being defined as follows: $\{ \cup_{i=1}^{m} v_D(\sigma_D(n_p^i)) \cup ad_{\Pi(D)} \cup tc_A \mid D$ is a fuzzy data tree of $\mathcal{W}$ and σ_D is a valuation of A w.r.t. $D\}$.

Example 5. *To facilitate result interpretation, answers are ordered first by ascending transformation cost (tc_A) and second by descending adequation degree ($ad_{\Pi(D)}$). The answer to the query Q of figure 3 according to the left approximate query of figure 4 in the fuzzy data trees of figure 2 is the following: {(Product.originalVal=Red onion, Product.finalVal=1.0/Tree onion+1.0/Welsh onion+0.2/Red cabbage, Attribute=100g, Lipid=5g, ad_Π=0.9, tc_A=10)}. The answer to the query Q of figure 3 according to the right approximate query of figure 4 in the fuzzy data trees of figure 2 is the following: {(Product.originalVal= Roasted Beef, Product.finalVal=1.0/Beef:carcass+0.8/Beef:conjunctive tissue, Lipid=30g, ad_Π=1.0, tc_A=100), (Product.originalVal=Red onion, Product. finalVal = 1.0/Tree onion+ 1.0/Welsh onion+ 0.2/Red cabbage, Lipid=5g, ad_Π =0.9, tc_A=100) }.*

Remark 2. *Note that when we compute the approximate answer to a query Q which is the union of the answers to the generated approximate queries from Q, we only keep tuples coming from distinct data trees of the XML database.*

4 Application

The approximate query processing detailed below has been applied to the XML database of the MIEL++ system. Subsection 4.1 briefly presents the SML process which permits this database to be filled. Subsection 4.2 presents the implementation of the approximate query processing and first experimental results.

4.1 The XML Database Filling

The XML database of the MIEL++ system is filled with data semi-automatically extracted from the Web. The search is focussed on pdf and html documents which contain data tables and concern the MIEL++ domain application. The Web data tables are extracted and translated into a generic XML representation of data table, called XTab. Those XTab documents are then transformed into SML (for Semantic Markup Language) documents, by a semantization process based on the MIEL++ ontology. Then it becomes possible to query the SML documents through the MIEL uniform query language. The SML process [9] achieves three kinds of semantic enrichment: (i) it associates terms of a Web data table with their corresponding terms in the MIEL++ taxonomy, (ii) when enough terms are identified in a given column of a Web data table, it becomes possible to identify the "title" of the column, otherwise the generic title *attribute* is associated with the column; (iii) it instanciates semantic relations of the ontology which appear in the Web table schema. That instanciation is done by comparing the previously identified columns with the signatures of the semantic relations of the ontology.

A semantic relation can be completely or partially represented in a Web table schema. Moreover, in [5], we propose a fuzzy semantic enrichment of the terms of a Web data table: each association between a term of a Web data table and a term belonging to the MIEL++ taxonomy is weighted by a possibility degree depending on their syntactic closeness. The SML documents thus contain fuzzy data: for a given term of a Web data table, its associated terms belonging to the taxonomy are represented by a discrete fuzzy set.

Example 6. *Figure 5 presents two examples of Web data tables and the corresponding tables obtained by semantic enrichment. In the left table, the first column has been identified as having the type* product *and the third one as having the type* lipid. *But the second one, which actually corresponds to the amount, has not been identified because the title of the column is an abbreviation (Qty). So it is associated with the generic type* attribute. *The semantic relations* FoodProductAmountLipid *of the MIEL++ ontology is only partially instanciated, because the attribute* Amount *is not in the Web table schema. The remaining column having the type* attribute *is added to the instanciation of the relation* FoodProductAmountLipid *in order to avoid bad interpretation of the data. The left fuzzy data tree of figure 2 corresponds to the third line of this Web data table. In the right table, the first column has been identified as having the type* product *and the second one as having the type* lipid. *Consequently, the relation* FoodProductAmountLipid *is only partially instanciated, because the attribute* Amount *is not in the Web table schema. The right fuzzy data tree of figure 2 corresponds to the second line of this Web data table.*

In example 5, the tuple of the first answer (tc_A=10) shows that the renaming of the node Amount *of the query Q to* Attribute *allows the value 100g to be retrieved even if the corresponding column (*Qty*) in the left Web data table of figure 5 has not been identified as an amount. The first tuple of the second answer shows that the deletion of the node* Amount *in the query Q allows pertinent information to*

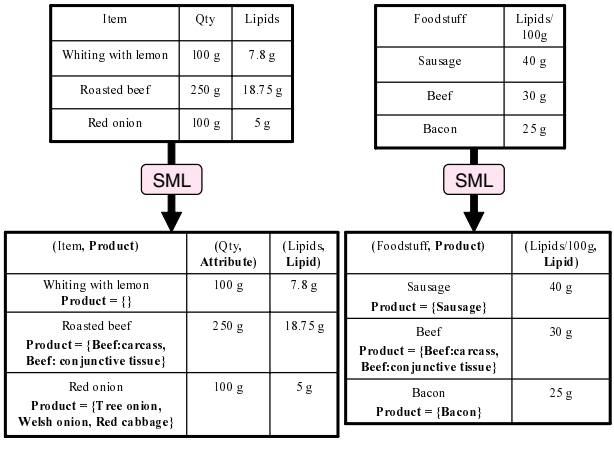

Fig. 5. Two examples of Web data tables and their associated semantic enrichment

be retrieved from the right Web data table of figure 5 even if the column amount has not been identified. In all the tuples of both answers, the insertion of the nodes originalVal *and* finalVal *into the query Q has been used.*

4.2 Implementation and Preliminary Tests

We have implemented a prototype of this querying system in Java using the Xquery processor Saxon (www.saxonica.com). Given an XML initial query, the program runs in the following three steps. In the first step, approximate queries are generated in XML documents from the initial query by means of deletion and renaming of nodes. In the second step, each distinct approximate query is translated into an XQuery query and executed in ascending order of its transformation cost. The result is built from the union of the answer to the initial query and the answers to the generated approximate queries, the duplicates being removed. Encouraging preliminary tests have been done on a database composed of 196 SML documents. Three different queries involving three different semantic relations have been tested. The first (resp. second and third) query contains 3 (resp. 2 and 2) selection and 3 (resp 3 and 2) projection attributes. Among the 93 results obtained, 66 are pertinent results obtained from the approximate queries, 4 of which being also obtained from the initial query.

5 Conclusion

In this paper, we propose a new XML querying system which harmoniously combines two kinds of complementary flexibility management. These two kinds of flexibility concern on the one hand the structure of the query tree and on the other hand the values stored in the data trees. Firstly, the calculation of a set of approximate queries from an initial one allowing insertion, renaming and deletion of nodes permits the retrieval of XML data trees whose structure is close to that of the initial query. Secondly, the expression of preferences in selection criteria of the query, by means of fuzzy sets, allows the end-user to enlarge the set of answers. Finally, this new querying system is able to manage imprecise data, represented by possibility distributions, stored in the XML database using fuzzy pattern matching techniques. In the very near future, we will realise experimentations of our approach in different application domains using different ontologies in order to evaluate the genericity of our approach. We will also study the way of introducing a "degree of uncertainty" in the partial instanciation of semantic relations of the ontology into a Web data table.

References

1. S. Amer-Yahia, S. Cho, and D. Srivastava, *Tree pattern relaxation*, EDBT, 2002.
2. D. Braga, A. Campi, E. Damiani, G. Pasi, and PL. Lanzi, *FXPath: Flexible querying of xml documents*, Proc. of EuroFuse 2002 (2002).

3. P. Buche, C. Dervin, O. Haemmerlé, and R. Thomopoulos, *Fuzzy querying of incomplete, imprecise and heterogeneously structured data in the relational model using ontologies and rules*, IEEE Trans. Fuzzy Systems **13** (2005), no. 3, 373–383.
4. P. Buche, J. Dibie-Barthélemy, O. Haemmerlé, and G. Hignette, *Fuzzy semantic tagging and flexible querying of xml documents extracted from the web*, Journal of Intelligent Information Systems **26** (2006), 25–40.
5. P. Buche, J. Dibie-Barthélemy, O. Haemmerlé, and M. Houhou, *Towards flexible querying of xml imprecise data in a data warehouse opened on the web*, FQAS 2004 (Lyon, France), LNAI #3055, Springer, June 2004, pp. 28–40.
6. E. Damiani and L. Tanca, *Blind queries to xml data*, DEXA'00, 2000, pp. 345–356.
7. C. Delobel, M. C. Rousset C. Reynaud, J.-P. Sirot, and D. Vodislav, *Semantic integration in xyleme: a uniform tree-based approach*, DKE. **44** (2003), no. 3, 267–298.
8. D. Dubois and H. Prade, *Possibility theory: an approach to computerized processing of uncertainty*, New York: Plenum Press, 1988.
9. H. Gagliardi, O. Haemmerlé, N. Pernelle, and F. Sais, *A semantic enrichment of data tables applied to food risk assessment*, DS'05, LNCS #3735, 2005, pp. 374–376.
10. Zongmin Ma, *Fuzzy database modeling with xml*, Springer., 2005.
11. Lucian Popa, Yannis Velegrakis, Renée J. Miller, Mauricio A. Hernández, and Ronald Fagin, *Translating web data.*, VLDB, 2002, pp. 598–609.
12. T. Schlieder, *Schema-driven evaluation of approximate tree-pattern queries*, Proceedings of EDBT, 2002.
13. Jeffrey D. Ullman, *Information integration using logical views.*, Theor. Comput. Sci. **239** (2000), no. 2, 189–210.
14. J. Widom, *Research problems in data warehousing*, Proceedings of the International Conference on Information and Knowledge Management, 1995.
15. G. Wiederhold, *Mediation in information systems*, ACM Computing Surveys **27** (1995), no. 2, 265–267.
16. L. Zadeh, *Fuzzy sets*, Information and control **8** (1965), 338–353.
17. ———, *Fuzzy sets as a basis for a theory of possibility*, Fuzzy Sets and Systems **1** (1978), 3–28.