Representation of semi-structured imprecise data for fuzzy querying

Patrice Buche, Ollivier Haemmerlé, Rallou Thomopoulos

Representation of semi-structured imprecise data for fuzzy querying

Patrice Buche, Ollivier Haemmerlé and Rallou Thomopoulos

INA P-G, UER d'informatique/INRA BIA, 16 rue Claude Bernard, F-75231 Paris Cedex 5, France
e-mail : {Patrice.Buche,Ollivier.Haemmerle,Rallou.Thomopoulos}@inapg.inra.fr
tel : +33 1 44 08 16 75, +33 1 44 08 72 29, +33 1 44 08 16 79, fax : +33 1 44 08 16 66

Abstract

This work is part of a national project which aims at building a tool for the analysis of microbial risks in food products. As a first step, we propose a unified querying system which simultaneously scans two complementary bases, containing microbiological information : a relational database containing structured information and a conceptual graph knowledge base containing semi-structured information. The unified querying system sends the user's query to both of them. Fuzzy queries and imprecise information are handled in both bases. To achieve this goal, we propose a way of representing fuzzy values, including numerical values, in conceptual graphs.

1.Introduction

Our research takes place in a national project which brings together government institutions and industry to build a tool for the analysis of microbial risks in food products. The first step of this project consists in gathering in a database information available in the scientific bibliography in microbiology and that can be useful for risk assessment.

A fundamental problem in our application is that this information consists of experimental results in a field where knowledge is growing everyday. The integration of this information is a source of irregularity : similar data are often represented in different ways in independent bibliographical references. The term "semi-structured" is used to qualify this kind of information which is not really structured but presents similarities, even if this is implicit. With semi-structured information, it is very difficult to determine a classical database schema in order to store all the useful information.

Microbiological data can be qualitative or quantitative. These data are generally imprecise due to the complexity of the biological processes involved. Storing and handling imprecise information is thus essential. The data are also incomplete : the scientific bibliography does not cover all possible experimental factors and conditions. For this reason it is important to introduce flexible queries, expressing enlarged selection criteria, with preference degrees, in order to avoid empty answers.

In order to store semi-structured information, different approaches have been proposed such as (i) the definition of a new kind of database management system especially designed for semi-structured data [1]; (ii) the definition of viewpoints in the object model [2]; (iii) hybrid approaches combining the use of languages designed for semi-structured information representation such as XML for instance and object-oriented DBMS [3] or semi-structured DBMS [4].

Concerning flexible queries and imprecise information management, the bibliography covers two kinds of problems. In a first category of papers, the fuzzy set framework has been shown to be a sound scientific way of modelling flexible queries [5]. In the second category of papers, the fuzzy set framework has also been proposed to represent imprecise values by means of possibility distributions [6].

The approach we chose consists in designing a unified querying system (called UQS) that scans two separate bases simultaneously : (i) a relational database containing the structured information, processed by the SI engine (for Structured Information), (ii) a conceptual graph knowledge base containing the semi-structured information, processed by the SSI engine (for Semi-Structured Information). The structure of the unified querying system is presented in [7, 8].

The main reason why we chose the relational model to store structured information is the efficiency and the robustness of this technology. A second reason is that it has been widely studied in previous researches as a way of managing fuzzy queries and imprecise information, operated in the SI engine. The structured information subsystem of UQS is presented in [9].

In the semi-structured information subsystem, we chose the conceptual graph model [10, 11] for many reasons. Firstly, its graph structure is well suited for the representation of weakly structured information. Secondly, a conceptual graph knowledge base can be scanned using graph operations already defined in this model. Thirdly, the terminological knowledge can be useful to implement enlarged flexible querying. Fourthly, different software platforms are available, allowing one to realize prototypes easily.

In this paper, we focus on the semi-structured infor-

mation subsystem. We aim at expressing imprecise information and submitting enlarged queries using the conceptual graph model. We are thus concerned with the expression of fuzzy values in a conceptual graph knowledge base. The scanning of the knowledge base by fuzzy queries has also been studied, but is not presented here.

In the second section, we define the UQS unified query language. In section 3, we briefly present the conceptual graph model. In section 4, we focus on the representation of numerical and fuzzy values in this model.

2.UQS query language

In UQS, the queries are expressed in terms of a set of DB-projection[1] attributes and a set of selection criteria using the form attribute/value. These queries are expressed in a given view. A view is a classical concept in databases, e.g. a virtual table in which all the information needed by the user are brought together. The transposition of this notion so as to scan the CG knowledge base has already been presented in [7, 8].

Definition 1 *A query Q in UQS is a set $\{V, a_1, \ldots, a_P, <a_{P+1}, v_{P+1}>, \ldots, <a_{P+S}, v_{P+S}>, nb, t\}$ where V is the name of the view in which the query is asked ; $a_1, \ldots, a_P$ are the attributes of the DB-projection, $<a_{P+1}, v_{P+1}>, \ldots, <a_{P+S}, v_{P+S}>$ are pairs defining the selection criteria, nb is the maximum number of tuples in the result and t is a threshold ($t \in [0,1]$) defining the minimum matching degree accepted for each tuple of the answer. Note that $\{a_1, \ldots, a_P\} \cap \{a_{P+1}, \ldots, a_{P+S}\}$ is not necessarily empty. The pairs defining the selection criteria have the following meaning :*

$\forall s \in [P+1, P+S]$

- *a_s is a selection attribute ;*
- *v_s is the value associated with the selection attribute a_s. This value is a fuzzy set on the underlying domain D_s, defined by its membership function $\mu_{v_s} : D_s \rightarrow [0,1]$. It is refered to by a linguistic label. We distinguish two kinds of fuzzy sets depending on the underlying domain (discrete or continuous).*

Two examples of fuzzy sets are given in fig. 1.

Here is an example of a query Q defined in the fuzzy view *Thermization* in the knowledge base about the behaviour of Listeria.
Q = {*View=Thermization, Id, Substrate, Temperature, Duration, <Substrate, MyMilkProductPreferences>, <Duration, HighDuration>, 10, 0.3*}.

The result of the execution of a query in UQS is a fuzzy relation, e.g. a fuzzy set defined on the cartesian product of the attributes. Each line represents a tuple composed of pairs $<attribute, value>$, resulting from the query. Tuples are ordered according to a matching degree md.

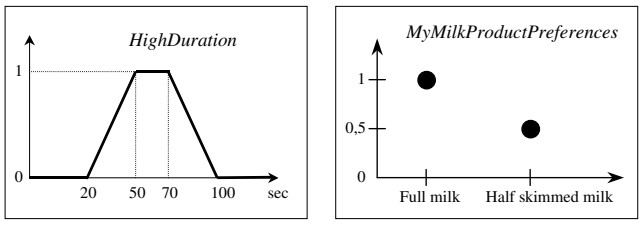

Figure 1: Fuzzy sets *HighDuration* and *MyMilkProductPreferences*

Definition 2 *An answer A to a query Q in UQS is a set of tuples, each of them of the form $\{<a_1, v_1>, \ldots, <a_P, v_P>, md\}$, with the following meaning :*

$\forall p \in [1, P]$

- *a_p is an attribute of the DB-projection ;*
- *v_p is the value associated with the attribute a_p, resulting from the execution of the query.*

  v_p may be an imprecise value represented by a possibility distribution on the underlying domain D_p, defined by $\pi_{v_p} : D_p \rightarrow [0,1]$.

md is the matching degree associated with each tuple.

In the following μ_{v_s} and π_{v_p} are supposed to be normalized.

3.The conceptual graph model

The Conceptual Graph (CG) model is a knowledge representation model based on labelled graphs, introduced by John Sowa [10]. We use the formalization presented in [11]. In the CG model, knowledge is divided into two parts : the terminological part (the support) and the assertional part (the CGs). In this section, we briefly and intuitively present the CG model through the example of our application.

3.1.The support

The *support* provides the ground vocabulary used to build the knowledge base : the types of concepts used, the instances of these types, and the types of relations linking the concepts. It describes the hierarchical organization of these elements.

The *set of concept types* is partially ordered by a *kind of* relation. *Universal* and *Absurd* are its greatest and lowest common elements, as presented in fig. 2.

[1] in order to prevent ambiguities, we use the term "DB-projection" when dealing with the notion used in the relational database model

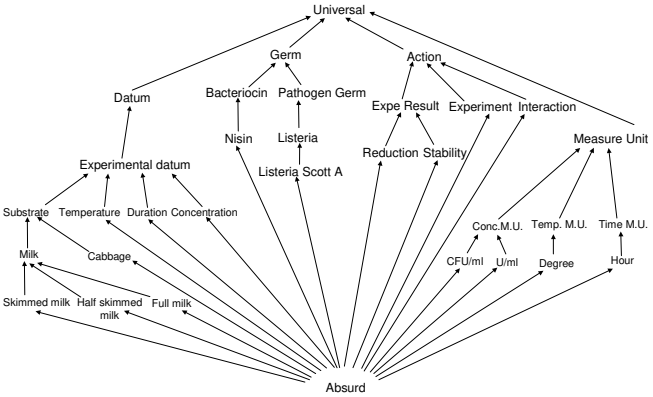

Figure 2: A part of the concept type set for the microbial application

The concepts can be linked by means of relations. The *set of relation types* is partially ordered by a *kind of* relation. Each relation type is characterized by an arity and a signature which specifies the maximal concept types that a given relation can link together. The set of relation types we use contains relation types such as *agt*, which is a binary relation having *(Action, Germ)* as a signature. It means that "an Action has for agent a Germ" (for example an interaction can have a bactery as an agent).

The third set of the support is the *set of individual markers*. Each individual marker represents an instance of a concept. For example, E1 can be an instance of *Experiment*. The generic marker (noted $*$) is a particular marker referring to an unspecified instance of a concept.

3.2.The conceptual graphs

The CGs, built upon the support, express the factual knowledge. The CGs are composed of two kinds of vertices: (i) the *concept vertices* (noted in rectangles or in brackets) which represent the entities, attributes, states, events ; (ii) the *relation vertices* (noted in ovals or in parentheses) which express the nature of the relations between concepts. The *label* of a concept vertex is a couple defined by the type of the concept and a marker (individual or generic) of this type. The label of a relation vertex is its relation type. The CG given in fig. 3 is a representation of the information : "in the experiment E1, interaction I1 between nisin and Listeria Scott A is realized in full milk and leads to reduction as a result".

Definition 3 *The knowledge base* $KB = \{G_1, \ldots, G_p\}$ *containing the semi-structured knowledge of our system is a set of connected, possibly cyclic CGs.*

The set of CGs is partially ordered by the specialization relation (noted $\leq$), which can be computed by the projection operation (a kind of graph morphism). This operation is widely used for the querying of CG knowledge bases.

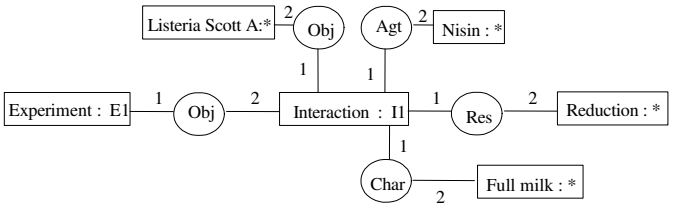

Figure 3: An example of a conceptual graph

4.Adapting the conceptual graph model to quantitative and to fuzzy data

4.1.Representing numerical values in the conceptual graph model

The microbiological data stored, as well as the user's queries, include numerical values, like temperatures, concentrations, durations. However the CG model we use represents symbolic data. To be more precise, two incompatible concept types cannot have a common individual marker. For instance, if '30' is used as a marker for the concept type *Duration*, then it may not be used for the type *Temperature*. For this reason another way of representing numerical values (and values in general) will be adopted in the following, modifying the support in use.

The concept type *NumericalValue* is introduced into the support. This concept type is a subtype of the more general type *Value*. The relation type *NumVal(Datum, NumericalValue)* is introduced into the support. This relation type is a subtype of the more general type *Val(Datum, Value)*.

Remark 1 *The designation of these types, as well as the signatures of the relation types introduced, are given as an example and can be modified and adapted to other applications. In the same way, other subtypes of the concept* Value *and the relation type* Val *may be considered and hierarchically classified, like strings, real numbers, integers and so on.*

Definition 4 *A numerical value is a marker of a specific concept type.*

This concept type is called *NumericalValue* in our application. For example, the set of markers associated with the type *NumericalValue* can be $\mathbb{R}$. This will be assumed in the following.

The conceptual graph of fig. 4 completes fig. 3 with additional information, including numerical values represented on the basis of the new support. It could be translated by "in the experiment E1, interaction I1 between nisin at a concentration of 50 U/ml and Listeria Scott A is realized in full milk during 30 minutes at a

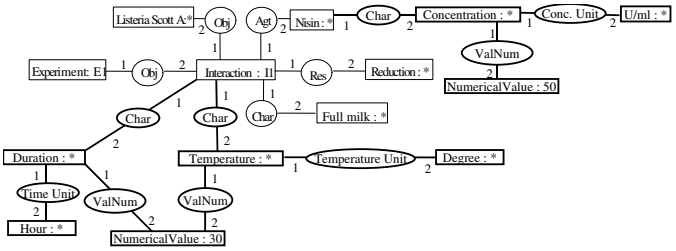

Figure 4: An example of a conceptual graph representing numerical values

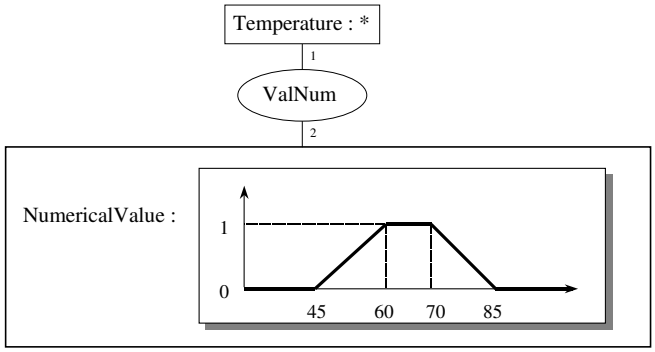

Figure 5: An example of a concept with a fuzzy marker

temperature of 30 degrees and leads to reduction as a result".

4.2.Representing fuzzy values in the conceptual graph model

The theory of fuzzy sets and the possibility theory [12] allow one to deal with imperfect data. In our field of application, we essentially deal with fuzzy values firstly to represent imprecise data, secondly to express preferences with fuzzy sets in order to query an incomplete database.

Fuzzy conceptual graphs were introduced by Morton [14] and extended by several works, e.g. [15, 16].

In comparison with the previous ones, we present a more homogeneous and integrated approach to include fuzzy sets in the CG model : (i) we propose a homogeneous representation of fuzzy types and fuzzy markers ; (ii) the domains of these fuzzy sets are built upon the support.

As presented in section 2, the selection criteria expressed in the unified query language use the form attribute/value, the value being a fuzzy set. This form is directly exploitable in the relational database, whereas in the conceptual graph knowledge base a translation has to be done : an attribute corresponds to a concept type, and a value can either correspond to a concept type or to a marker. For that reason, it is necessary to be able to define both fuzzy concept types and fuzzy markers.

Definition 5 *The* reference domain $Ref(t)$ *associated with the concept type t is the set of individual markers that conform to t.*

$$\forall t \in T_C, Ref(t) = \{m \in I \mid \tau(m) \leq t\}$$

with the following meaning :

- T_C *is the set of concept types defined in the support ;*
- I *is the set of individual markers ;*
- τ *is an application from I to* T_C *that associates each individual marker m with a concept type t.*

The reference domain of a concept type can be finite or infinite, continuous or discrete. For example, if the markers that conform to the concept type *NumericalValue* are the real numbers, then *Ref(NumericalValue)*=$\mathbb{R}$ is continuous and infinite. If there are two individual markers *T1* and *T2* that conform to the concept type *Temperature*, then *Ref(Temperature)* = {*T1, T2*} is discontinuous and discrete.

Definition 6 *Let Ref(t) be the reference domain of a concept type t. A* **fuzzy marker** m_f *of type t is a fuzzy set defined on Ref(t).*

Remark 2 *A "classical" individual marker m* ($m \neq *$) *of type t can be considered as a particular fuzzy marker of t. Its membership function associates the value 1 with m, and the value 0 on the rest of the domain* $Ref(t)$.

Definition 7 *A* **concept with a fuzzy marker** *is a concept vertex whose label is a couple* (t, m_f)*, where t is an element of* T_C *and* m_f *is a fuzzy marker of the concept type t.*

The conceptual graph represented in fig. 5 includes a concept with a fuzzy marker, of type *NumericalValue*.

Definition 8 *A* **fuzzy type** t_f *is a fuzzy set defined on a subset* D_{t_f} *of concept types such that :* $\forall (t_1, t_2) \in D_{t_f}$, t_1 *and* t_2 *are not comparable (i.e.* $t_1 \not\leq t_2$ *and* $t_2 \not\leq t_1$*).*

Remark 3 *A "classical" concept type t can be considered as a particular fuzzy type. Its membership function is defined on one element* $\{t\}$ *of* T_C *and takes the value 1 for this element.*

Definition 9 *A* **concept with a fuzzy type** *is a concept vertex whose label is a couple* (t_f, m)*, where* t_f *is a fuzzy type and m is the generic marker *.*

Remark 4 *In the conceptual graph model, the application* τ *associates each individual marker with a unique concept type. For this reason, m cannot be an individual marker in definition 9.*

For instance, let us suppose that the user's preferences concerning the substrate are *MyMilkProductPreferences* represented in fig. 1. In conceptual graph terms, these preferences correspond to the concept [Full milk : *] with

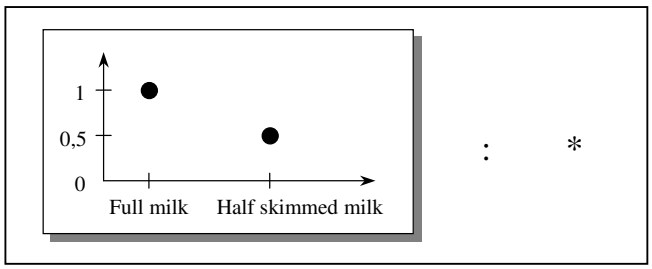

Figure 6: An example of a concept with a fuzzy type

the degree 1, and to the concept [Half skimmed milk : *] with the degree 0.5, which is synthesized by the concept with a fuzzy type of fig. 6.

5.Conclusion and perspectives

In this paper, we have focused on the semi-structured engine of a unified system that queries both a structured relational database and a semi-structured knowledge base represented in terms of conceptual graphs. More precisely, we have presented the representation of fuzzy values (including numerical values) using the conceptual graph model. These fuzzy values can either correspond to fuzzy concept types or to fuzzy markers, they can be interpreted as imprecise data or as queries with expression of preferences. We have also studied the scanning of the knowledge base using fuzzy queries submitted by the unified querying system, but this part of the work has not been presented in this paper. We have implemented the part of this work concerning the CGs as a prototype built on the CoGITo platform [17].

Our very next work will focus on two different points : (i) the extension of the unified query language, (ii) the testing of our prototype on an entire knowledge base, which has to be created in cooperation with the group of microbiologist experts working on our national project.

References

[1] J. McHugh, S. Abiteboul, R. Goldman, D. Quass, and J. Widom. Lore: A database management system for semistructured data. *SIGMOD Record*, 26(3):54–66, September 1997.

[2] S. Coulondre and T. Libourel. Viewpoints handling in a object model with criterium-based classes. In *Proceedings DEXA'99 (Database and EXpert system Application), Lecture Notes in Computer Science #1677*, pages 573–583, Florence, Italy, August 1999. Springer-Verlag.

[3] A. Michard. *XML, langage et applications*, pages 335–344. Eyrolles, Paris, 1999.

[4] R. Goldman, J. McHugh, and J. Widom. From semistructured data to XML: Migrating the lore data model and query language. In *Proceedings of the 2nd International Workshop on the Web and Databases (WebDB'99)*, Philadelphia, USA, June 1999. Springer.

[5] P. Bosc, L. Lietard and O. Pivert. Soft querying, a new feature for database management system. In *Proceedings DEXA'94 (Database and EXpert system Application), Lecture Notes in Computer Science #856*, pages 631–640, 1994. Springer-Verlag.

[6] H. Prade. Lipski's approach to incomplete information data bases restated and generalized in the setting of Zadeh's possibility theory. *Information Systems*, 1(9):27–42, 1984.

[7] P. Buche and O. Haemmerlé. Towards a unified querying system of both structured and semi-structured imprecise data using fuzzy views. In *Proceedings of the 8th International Conference on Conceptual Structures, Lecture Notes in Artificial Intelligence #1867*, pages 207–220, Darmstadt, Germany, August 2000. Springer-Verlag.

[8] P. Buche and O. Haemmerlé. Towards category-based fuzzy querying of both structured and semi-structured imprecise data. In *Proceedings of the Fourth International Conference on Flexible Query Answering Systems (FQAS'2000)*, pages 362–375, Warsaw, Poland, October 2000. Springer-Verlag.

[9] P. Buche and S. Loiseau. Using contextual fuzzy views to query imprecise data. In *Proceedings DEXA'99 (Database and EXpert system Application), Lecture Notes in Computer Science #1677*, pages 460–472, Florence, Italy, August 1999. Springer-Verlag.

[10] J.F. Sowa. *Conceptual structures - Information processing in Mind and Machine*. Addison-Welsey, 1984.

[11] M.L. Mugnier and M. Chein. Représenter des connaissances et raisonner avec des graphes. *Revue d'Intelligence Artificielle*, 10(1):7–56, 1996.

[12] L.A. Zadeh. Fuzzy sets as basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:3–28, 1978.

[13] M.H. Zwietering, T. Wijtzes, J.C. de Wit and K. Van't Riet. A Decision support system for Prediction of the Microbial Spoilage in Foods. *Journal of Food Protection*, 55:973-979, 1992.

[14] S.K. Morton. Conceptual graphs and fuzziness in artificial intelligence. PhD Thesis, University of Bristol, 1987.

[15] V. Wuwongse and M. Manzano. Fuzzy conceptual graphs. In *Proceedings of the 1st International Conference on Conceptual Structures, ICCS'93, Lecture Notes in Artificial Intelligence #699*, pages 430–449, Quebec City, Canada, August 1993. Springer-Verlag.

[16] T.H. Cao. Foundations of Order-Sorted Fuzzy Set Logic Programming in Predicate Logic and Conceptual Graphs. PhD Thesis, University of Queensland, Australia, 1999.

[17] O. Haemmerlé and O. Guinaldo. CoGITo v3.3 : plate-forme de développement d'applications sur les graphes conceptuels. *Technique et Science Informatiques*, 18(9):933–965, November 1999.