Conceptual Graphs as Cooperative Formalism to Build and Validate a Domain Expertise

Rallou Thomopoulos[1,2], Jean-François Baget[3,2], and Ollivier Haemmerlé[4]

[1] INRA, UMR1208, F-34060 Montpellier cedex 1, France
rallou.thomopoulos@supagro.inra.fr
[2] LIRMM (CNRS & Université Montpellier II), F-34392 Montpellier cedex 5, France
[3] LIG/INRIA Rhône-Alpes, F-38334 St-Ismier cedex, France
jean-francois.baget@inrialpes.fr
[4] IRIT, Université Toulouse le Mirail, F-31058 Toulouse cedex, France
ollivier.haemmerle@univ-tlse2.fr

Abstract. This work takes place in the general context of the construction and validation of a domain expertise. It aims at the cooperation of two kinds of knowledge, heterogeneous by their granularity levels and their formalisms: expert statements represented in the conceptual graph model and experimental data represented in the relational model. We propose to automate two stages: firstly, the generation of an ontology (terminological part of the conceptual graph model) guided both by the relational schema and by the data it contains; secondly, the evaluation of the validity of the expert statements within the experimental data, using annotated conceptual graph patterns.

1 Introduction

Cooperation of heterogeneous knowledge is considered here in the case of different kinds of knowledge that do not have the same statute: one of the sources contains synthetic knowledge, at a general granularity level, it provides generic rules and is considered as intuitive to understand by humans; the other sources, on the contrary, are at a very detailed granularity level, precise and reliable, but too detailed to be directly exploitable by humans. In this study, the representation formalisms used for the different sources are adapted to the kind of knowledge to be represented: (i) expert statements that express generic knowledge rising from the experience of domain specialists and describing commonly admitted mechanisms. This knowledge is represented in the conceptual graph model [1], chosen for its graphical representation of both knowledge and reasoning, relatively intuitive for the experts (see [2,3] for more details). The formalization of simple conceptual graphs and of their extension to rules adopted in this paper is that of [4]; (ii) experimental data from the international literature of the domain, represented in the relational model. These numerous data describe in detail, in a quantified way, experiments carried out to deepen the knowledge of the domain. They may confirm the knowledge provided by the expert statements – or not.

U. Priss, S. Polovina, and R. Hill (Eds.): ICCS 2007, LNAI 4604, pp. 112–125, 2007.

The cooperation of both kinds of knowledge aims at testing the validity of the expert statements within the experimental data, with the longer-term objective to refine them and to consolidate the domain expertise.

Two major differences between the two formalisms are the following. Firstly, the conceptual graphs represent knowledge at a more generic scale than the relational data. Secondly, the conceptual graph model includes an ontological part (hierarchized vocabulary that constitutes the support of the model), contrary to the relational model. We propose as a first stage the generation of an ontology, guided by the structure and the data of the relational model, which in the considered case preexist to the knowledge represented in the conceptual graph model. Some of the questions to answer are the following: how can one identify, within the relational schema and the data it contains, the concepts which can be considered as relevant at a more general granularity level, for the expression of expert statements? How can one organize the identified concepts into a hierarchy, although the relational model does not explicitly take into account the "kind of" relation? Can one go further in the suggestion of relevant complementary concepts? The proposed method is semi-automatic, expert validation is required.

As a second stage, we introduce a process that allows one to test the validity of expert statements within the experimental data, that is, to achieve the querying of a relational database by a system expressed in the conceptual graph formalism. This stage is automatic. Besides the definition of the evaluation of expert statements validity, the problem to solve concerns the automation of the generation of SQL queries on the basis of conceptual graphs whose form and content can vary. The process is based on the use of conceptual graph patterns.

Section 2 describes the generation of an ontology, guided by information about the structure and the data of the relational model. Section 3 presents a method to evaluate the validity of expert statements within the experimental data. Section 4 illustrates the results within a concrete case concerning food quality control, studied at the INRA French institute for agronomical research. In this application the objective is to highlight major trends concerning the impact of food process operations (e.g. milling, storage, extrusion, hydration, etc.) on end-product quality markers (e.g. vitamins, minerals, lipids, etc.).

2 Generation of an Ontology

Our work takes place in the case where a collection of detailed experimental data represented in the relational model preexists to the expression of expert knowledge of a higher granularity level. Our goal is to automate as far as possible the generation of a simple ontology. That ontology is constituted of the set of concept types belonging to the terminological part of the knowledge in the conceptual graph model, by using the existing relational schema and data.

In this section, after a presentation of related work, we describe three steps of the generation of the ontology: the identification of high-level concept types, the organization of these concept types into a hierarchy, and the proposition of complementary concept types.

2.1 Related Work

As the distinction between relevant and non-relevant high-level concept types entails to a large extent human expertise, a completely automatic method to generate the ontology cannot be considered [5]. Our goal differs from concept learning as proposed in the FCA approach – Formal Concept Analysis [6] – which relies on the existence of properties shared by subsets of data in order to group them into new concepts. Here, our main goal is to identify and hierarchize relevant concepts for the expression of expert knowledge, among those which preexist in the data in a non-explicit form or with an inadequate structure. The search for a new structure for specific goals in already structured data, which is our objective here, is not that frequent. Close works are those which concern the cohabitation of heterogeneous vocabularies, like model transformation [7] and ontology alignment [8]. In ontology alignment, mappings are established between pre-existing vocabularies while in our study, the ontology results from the data.

From conceptual graphs to databases. The mapping between simple conceptual graphs and conjunctive queries in databases is well-known [9,4]. Let $\mathcal{V}$ be a vocabulary, and G and Q two simple graphs on $\mathcal{V}$. G and Q are transformed (into G' and Q') in the following way: the concept types are transformed into unary relations, and each concept of type t becomes a non-typed concept incident to a unary relation typed t. For each relation r of type t, for each supertype t' of t, we add a new relation r' of type t' such that $\gamma(r) = \gamma(r')$.[1] Consequently, $G \models_{\mathcal{V}} Q$ iff $\Phi(G') \models \Phi(Q')$ (we do not need the formulas which translate the support anymore since their consequences have been translated in the graphs). Since $\Phi(G')$ and $\Phi(Q')$ are positive conjunctive formulas, we can define $\mathcal{B}$ as the tables having $\Phi(G')$ as associated logical formula and A as the query having $\Phi(Q')$ as associated logical formula. So we have $G \models_{\mathcal{V}} Q$ iff there exists an answer to A in $\mathcal{B}$. Nevertheless, that mapping relies on an identification between the vocabulary of the conceptual graphs and the database schema, which is a too strong hypothesis as we shall see in the following.

Sym'Previus. The Sym'Previus system has been developed in a French research project on the assessment of the microbiological risk in food products [10]. The tool relies on three distinct databases which have been added successively during the evolution of the project: a classic relational database, a conceptual graph database and an XML database. The three bases are queried simultaneously by means of a unique interface based on a single ontologyand on a query language close to the relational formalism (in Section 3 we will deal, on the contrary, with the querying of a relational database by conceptual graph queries). Contrary to the approach proposed in this paper, that ontology was built manually, when the conceptual graph database was added to the system. A relational database schema and its data were pre-existent. To build the ontology, the set of the attributes corresponding to meaningful entities of the application were divided into two parts: the attributes for which the values could be hierarchized according to the "kind of" relation (substrate, pathogenic germ ...)

[1] We denote γ the function that associates, with each relation, a tuple of concepts (its arguments). The size of the tuple is the degree of the relation.

and the attributes for which the values were "flat" sets (the names of the authors of publications for example). All the meaningful attributes were added to the Sym'Previus ontology as concept types. The hierarchized attribute values were inserted as concept subtypes in the ontology. Their precise position in the hierarchy was determined manually by experts.

2.2 Identification of High-Level Concept Types

In this stage, our goal is to identify high-level concept types (concept types located at a general granularity level). We identify two kinds of entities which we consider as potentially relevant high-level concept types: (i) those whose occurrences have a name, that is to say which have an attribute "name" (or "label", or which contain the string "name", etc.). We assume that these entities have a most general nature, by opposition to secondary entities whose occurrences are not named but only identified by a numeric label. Only the first ones are useful to express expert assertions: the experts use notions designated by a name (e.g. process operation names, food names, etc.), they do not use information that are only relevant in specific circumstances (e.g. parameter values that are specific to each experiment); (ii) those which can be divided into subcategories. In order to identify them, we search for entities with an attribute "category" (or "family", "type", etc.). We assume that these entities, due to the classification induced by their subcategories, provide relevant concept types for the ontology.

The border between these two cases is not strict and depends on the kind of modelization used. They will be considered in a homogeneous way in the following. In order to simplify, we do not indicate the exhaustive list of the considered attributes ("name", "category", "family", etc.) but we refer to them under the term of *flag attributes*. Those attributes are of type string.

Definition 1. *We call* **flag attribute** *each attribute whose name belongs to a predefined list composed of terms expressing denomination or classification. Such an attribute is considered to belong to an entity of general granularity level.*

Use of the relational schema. In a first step, we use the schema of the relational database. From a database engineering point of view, after a modelization for example in the entity-association model, we know that a relation (or table) of the relational database schema corresponds: (i) either to an entity of the considered domain – then it contains its attributes. It can also contain the identifiers of other entities (with which it was linked by an association), more rarely association attributes; (ii) or to an association (of type many to many) between entities – it has their identifiers as attributes, more rarely association attributes. The resulting table is generally labelled by the name of the corresponding entity or association. In order to identify high-level concept types, we make the following simplifying assumptions: (i) the entities – rather than the associations – carry the main concepts of the considered domain. Then the high-level concept types must be searched in the names of entities, that is to say among the names of the tables of the relational schema; (ii) the case of an association having a flag attribute is considered as exceptional.

Definition 2. *We consider as* **high-level concept types extracted from the relational schema** *the names of the tables which contain a flag attribute. These identified high-level concept types are added to the ontology.*

Example 1. In our application, examples of high-level concept types extracted from the database schema are the following: *Food product*, *Change*, *Component*, *Method*, *Operation*, *Property*, *Variable*, ... On the other hand, *Experiment*, *Default value*, *Experimental value*, for example, have not been considered as high-level concept types. The case of *Experiment* for instance was validated by the experts as not being a relevant high-level concept type because it refers to an experimental scale – judged too specific – and not to the scale of general mechanisms governing the domain.

Use of the relational data. In a second step, we consider the values taken by the flag attributes. We have made the hypothesis that the flag attributes can take as values subcategories of the entity they belong to. Thus taking into account the relational data allows us to propose the values of the flag attributes as high-level concept types. Their hierarchical organization is specified in Section 2.3.

Definition 3. *The values taken by the flag attributes of the database are considered as* **high-level concept types extracted from the data**. *These identified high-level concept types are added to the ontology.*

Example 2. In our application, the following high-level concept types have been extracted from the data: *Increase*, *Decrease*, *Protein*, *Lipid*, *Vitamin*, *Vitamin B*, *Quality*, *Content*, ...

2.3 Organization of the Concept Types into a Hierarchy

Two organization levels are proposed: (i) between high level concept types extracted from the data and high-level concept types extracted from the schema: the value taken by a flag attribute of a table (high-level concept type extracted from the data) is considered as a specialization of the concept type that has the name of this table (high-level concept type extracted from the schema). For example, *Vitamin* is a specialization of *Component*; (ii) among high-level concept types extracted from the data: this level is based on the inclusion of the labels of the concept types. For example, *Vitamin B* is a specialization of *Vitamin*.

Definition 4 summarizes the stages 2.2 and 2.3, validated by experts.

Definition 4. *The* **generation of a simple ontology** $\boldsymbol{O}$ *from the relational database is processed in the following way. For each table, whose name is denoted* T*, of the database, if table* T *has at least one flag attribute, then:*
– *the high-level concept type extracted from the schema* T *is added to* O*;*
– *for each flag attribute of* T*, that takes a set of values* $v_1, \ldots, v_n$*:*
  • *the high-level concept type extracted from the data* v_i*, subtype of* T*, is added;*
  • *if* v_i *is included in* v_j $(i, j \in [1, n])$*, then* v_j *is a subtype of* v_i*.*

Example 3. For example the table *Component* has the flag attribute *component_name*, whose values are *Protein*, *Lipid*, *Vitamin*, etc. The high-level concept type (extracted from the schema) *Component* is added to O and the high-level concept types (extracted from the data) *Protein*, *Lipid*, *Vitamin*, *Vitamin B* are added to O as subtypes of *Component*. As "Vitamin" is included in "Vitamin B", the concept type *Vitamin B* is a subtype of *Vitamin* (see Figure 1).

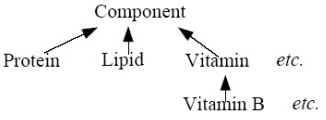

Fig. 1. Example of hierarchical organization between concept types

2.4 Proposition of Complementary Concept Types

The method proposed in this Section so as to complete the ontology with the suggestion of additional relevant concept types, is specific to the form of the expert knowledge considered in the application. We are in the following case. Expert knowledge is expressed by rules of the form "if (hypothesis) then (conclusion)". More precisely, these are causality rules. They express a relation of cause and effect between (i) a set of conditions, described by the hypothesis, interacting to produce and (ii) a resulting effect, described by the conclusion.

For example, a simple expert rule is the following: "if a food product, characterized by a vitamin content, undergoes a cooking in water, then that content decreases". It is represented by the conceptual graph rule of Figure 2.

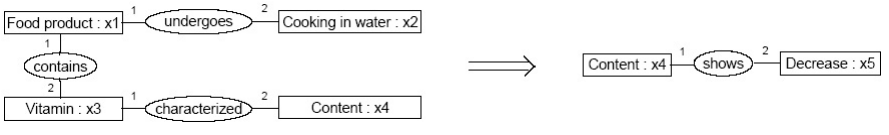

Fig. 2. Example of expert knowledge represented as a conceptual graph rule

The nature of interactions between the concepts that appear in the hypothesis is not always well-known by the experts: these interactions can be due to the interference of other concepts which are not necessarily identified. The objective of this part is to highlight some of these concepts. The method is based on the comparison of textual descriptions of the concepts that appear in the hypothesis. Indeed, the tables of the relational database from which were extracted the concept types that appear in the hypothesis (see Def. 4) sometimes provide textual descriptions, in the values of attributes named for example "description", "comments", etc. For each pair of concept types appearing in the same expert rule hypothesis, and for which such descriptions are available, the proposed method consists in searching for the existence of shared terms in these descriptions.

Example 4. The textual decriptions of some operations (Cooking in water, Steaming, Hydration, Drying) and of some components (Wheat bran, Fiber, Lipid, Vitamin, Polyphenol) share the term "water". Indeed, these unit operations all have an effect on water content (water addition or withdrawal) and these components all have subcategories that have a particular affinity with water (solubility, absorption). Highlighting the shared term "water" led the experts to complete the ontology, firstly, by adding the concept type *Water*, secondly, by specializing existing concept types to reveal categories that have a particular interaction with water: thus *Vitamin* is specialized into *Hydrosoluble vitamin* (super-type, for instance, of *Vitamin B*, which is soluble in water) and *Liposoluble vitamin*.

The obtained results are numerous and must be sorted manually by the experts. The search for shared terms uses techniques from natural language processing, such as the suppression of stopwords and of syntactic variations.

3 Evaluation of the Validity of Expert Statements

Contrary to the previous stage (Section 2) which requires an expert intervention, the method presented in this Section is automatic. The objective is to test whether the expert knowledge expressed as conceptual graph rules (created beforehand manually) is valid within the experimental data of the relational database. This must be achieved without having to define manually, for each rule, the queries to be executed in the database to obtain this information. A validity rate is computed for the tested rule and the data that constitute exceptions to the rule are identified and can be visualized by the user. In this Section, after defining the evaluation of the validity of a rule, we introduce the notions of rule pattern and of rule instance, then we expose the validation of a rule instance.

3.1 Computation of a Validity Rate

Evaluating the validity of an expert rule within the experimental data consists in calculating the proportion of data that satisfy both the hypothesis and the conclusion of the rule, among the data which satisfy the hypothesis of the rule. Let n_H be the number of data that satisfy the hypothesis and $n_{H \wedge C}$ the number of data that satisfy both the hypothesis and the conclusion. The validity rate V of a rule is $V = \frac{n_{h \wedge C}}{n_H} \times 100$, where n_H and $n_{H \wedge C}$ are the results of SQL queries counting the data that respectively satisfy the hypothesis, and both the hypothesis and the conclusion. The problem to solve is the automation of the construction of these queries.

3.2 Rule Patterns, Rule Instances and Associated Properties

Although the expert rules can take various forms, it is possible to group them into sets of rules which follow the same general form.

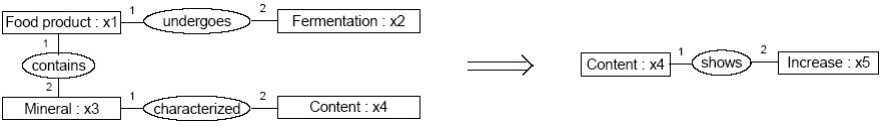

Fig. 3. Example of an expert rule that has the same form than the rule of Figure 2

Example 5. The expert rules given in Figures 2 and 3 have the same form.

The "general form" of a set of expert rules can itself be represented by a rule, called rule pattern. Its structure is identical to that of the expert rules that compose the set, but its concept vertices are more general. In other words, each expert rule of the set has a hypothesis and a conclusion which are specializations (by restriction of the labels) of those of the rule pattern. These expert rules are called rule instances. The hypothesis and conclusion of the rule pattern can thus be projected into those of each of its instances.

Example 6. The rules represented in Figures 2 and 3 are instances of the rule pattern of Figure 4.

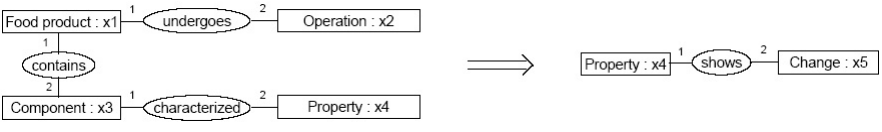

Fig. 4. Example of a rule pattern

The concept types used in a rule pattern have the following generality level: they are high-level concept types extracted from the relational schema. On the contrary, the concept types used in a rule instance can be high-level concept types extracted from the data (the markers can moreover be individual). This characteristic is essential for the validation of a rule instance.

Definition 5. *A* **rule pattern** *is a rule, in the conceptual graph formalism, whose concept vertices have high-level concept types extracted from the relational schema and whose markers are generic. A* **rule instance** *is a rule, in the conceptual graph formalism, obtained by restriction of the labels of the concept vertices of a given rule pattern. The rule instance is said to* **conform** *to this pattern.*

Consequently, the concept types that appear in a rule pattern provide a list of table names of the database (high-level concept types extracted from the schema). The hypothesis (respectively, the conclusion) of a rule pattern can be interpreted, within the database, as the formula of a query that selects the data satisfying the hypothesis (respectively, the conclusion). This formula uses the tables that appear as concept types in the hypothesis (respectively, the conclusion) of the rule pattern. This formula simply specifies a query schema. It does

not specify any particular selection criteria. Such criteria will only appear during the processing of the rule instances, presented in 3.3.

Definition 6. *Let H be the hypothesis of a rule pattern. Let Q be a query on the relational database that selects the data satisfying H. In terms of relational calculus, Q can be written as: $\{t|F(T)\}$, where F is a formula, T a tuple variable of F and $F(T)$ an evaluation of F. The answer to the query Q will be a set of tuples $\{t|F(t)\ true\}$. F is built as the conjunction of the following formulas.*
– Atomic formulas associated with the concepts of H: *Let $s_{c_1}, \ldots, s_{c_n}$ be the concepts of H, of types $c_1, \ldots, c_n$ (these are high level concept types extracted from the relational schema and therefore tables of the relational database). As the concepts of H are generic, each concept s_{c_i} provides the atomic formula: $\exists x_i, c_i(x_i)$.*
– Formulas associated with the relations of H: *Let s_r be a relation vertex of H with $\gamma(s_r) = (s_{c_k}, \ldots, s_{c_l})$. Two cases are possible:*

• *no other tables than those present in H are necessary in the schema of Q to join the tables $c_k, \ldots, c_l$. Each concept $s_{c_k}, \ldots, s_{c_l}$ of $\gamma(s_r)$ provides at least one atomic formula[2] of the form: $x_i.a_i = X_i$, where a_i denotes an attribute of table c_i and X_i a constant or an expression $x_j.a_j$ ($j \in [k, l]$, a_j attribute of c_j).*

• *additional tables to those present in H are necessary in the schema of Q to join the tables $c_k, \ldots, c_l$. Let $t_m, \ldots, t_p$ be these additional tables. Each of them provides an atomic formula $\exists x_i, t_i(x_i)$ and at least one atomic formula $x_i.a_i = X_i$. The relation vertex s_r thus provides a (non-atomic) formula of the form: $\exists x_m, \ldots, x_p, t_m(x_m) \wedge \ldots \wedge t_p(x_p) \wedge x_k.a_k = X_k \wedge \ldots \wedge x_l.a_l = X_l \wedge x_m.a_m = X_m \wedge \ldots x_p.a_p = X_p$.*

– Requested attributes: *Let $attr_1, \ldots, attr_q$ be the requested attributes, respectively from tables $tbl_1, \ldots, tbl_q$ ($attr_i$ not necessarily distinct from a_j, $j \in [k, l] \cup [m, p]$ and tbl_i in $\{c_1, \ldots, c_n, t_m, \ldots, t_p\}$). $F(t)$ is constrained by: $t.attr_i = tbl_i.attr_i$ ($i \in [1, q]$).*

In the general case, $F(t)$ is thus of the form: $\exists x_1, \ldots, x_n, x_m, \ldots, x_p, c_1(x_1) \wedge \ldots \wedge c_n(x_n) \wedge t_m(x_m) \wedge \ldots \wedge t_p(x_p) \wedge x_k.a_k = X_k \wedge \ldots \wedge x_l.a_l = X_l \wedge x_m.a_m = X_m \wedge \ldots x_p.a_p = X_p \wedge t.attr_1 = tbl_1.attr_1 \ldots t.attr_z = tbl_q.attr_q$.

This formula can only partly be generated in an automatic way. Indeed, tables $t_m, \ldots, t_p$, the attributes a_i and the terms X_i cannot always be calculated. The limits of automation are due to the ambiguity of joins between tables and the multiple possibilities that can be encountered in case of intermediate joins between tables. Thus the formula F must be defined by the designer, for the hypothesis of each rule pattern. The formula of the query that selects the data satisfying the conclusion of a rule pattern is built in the same way. Finally, the formula of the query that selects the data satisfying both the hypothesis and the conclusion of a rule pattern is obtained as the conjunction of the formulas associated with the hypothesis and the conclusion.

[2] These atomic formulas are not necessarily distinct from those provided by the other neighbours of s_r, for example a neighbour may provide $x_i.a_i = x_j.a_j$ and another one $x_j.a_j = x_i.a_i$.

To allow the evaluation of an expert rule (see Section 3.1), the two required queries are the one that counts the data satisfying the hypothesis and the one that counts the data satisfying both the hypothesis and the conclusion of a rule pattern. Each rule pattern is associated with those two queries by the designer.

Example 7. The formula of the hypothesis of the rule pattern of Fig. 4 is:
$\exists x_1, x_2, x_3, x_4, x_5, x_6,\ Food\ product(x_1) \wedge Operation(x_2) \wedge Component(x_3) \wedge Property(x_4) \wedge Result(x_5) \wedge Study(x_6) \wedge$
$x_1.id\_foodProduct = x_5.id\_foodProduct \wedge x_2.id\_operation = x_6.id\_\ operation \wedge x_3.id\_component = x_5.id\_subComponent \wedge x_4.id\_property = x_5.id\_\ property \wedge x6.id\_study = x5.id\_study \wedge t.x_4.id\_result = x_5.id\_result.$
The SQL query associated with the hypothesis of the rule pattern of Fig. 4 is:

```
SELECT COUNT(result.id_result) FROM result, food_product, component, study, operation
WHERE result.id_foodProduct = food_product.id_foodProduct AND
study.id_operation = operation.id_operation AND result.id_subComponent = component.id_component
AND result.id_property = property.id_property AND result.id_study = study.id_study.
```

Information is associated with each concept of a rule pattern, intended to inform about the specialization of this concept vertex within the rule instances that conform to this pattern: (i) if the concept type of this vertex has subtypes (high-level concept types extracted from the data), these subtypes are values of an attribute (of the corresponding table): which attribute? It is supposed to be the same for all the instances of a given rule pattern; (ii) if the marker of this vertex can be individual within the rule instances, this marker is then a value of an attribute (of the corresponding table): which attribute? Such an attribute is supposed to exist and to be the same for all the instances of a given rule pattern.

Example 8. In the rule pattern of Figure 4, the type *Component* can be specialized by subtypes which are also values of the attribute *name_component* of table *Component*. Hence in Figures 2 and 3, *Vitamin* and *Mineral*, which are specializations of the concept type *Component*, are also values of the attribute *Component.name_component*.

Definition 7. *An* **annotated** *pattern is a rule pattern P associated with:*
– a hypothesis query, that counts the tuples of the database satisfying the hypothesis of P;
– a hypothesis and conclusion query, that counts the tuples of the database satisfying both the hypothesis and the conclusion of P;
– for each of its concept vertices s_c (of type c), two attributes : (i) a type attribute, which indicates the attribute of table c that contains the specializations (denoted c'_i) of the concept type c expected in the rule instances conforming to P, for an image of s_c (through the projection operation); (ii) a marker attribute, which indicates the attribute of table c that contains the markers of concept types c or c'_i expected in the rule instances conforming to P, for an image of s_c (through the projection operation).

Remark 1. As the formulas of the queries associated with a rule pattern only specify a query schema, the results of both queries should be equal to the number of data in the database. Thus the validity of a rule pattern must be 100 %.

3.3 Validation of a Rule Instance

In order to test the validity of an expert rule, i.e. of a rule instance, which is the researched objective, two new queries will be automatically built: a query that counts the data satisfying the hypothesis of the rule instance (called hypothesis query) and a quary that counts the data satisfying both the hypothesis and the conclusion of the rule instance (called hypothesis and conclusion query).

These queries are composed of two parts: (i) the first part describes the schema of the query to be executed: this first part corresponds to the query associated with the rule pattern that the rule instance to be evaluated conforms to. This part is thus provided by the annotations of the rule pattern; (ii) the second part allows one to select exclusively the tuples which take the attribute values corresponding to the specializations that appear in the rule instance. This part thus specifies selection criteria, which will be automatically built by using, as selection attributes, the annotations of the rule pattern (type attributes and marker attributes) and as selection values, the concept types and the markers of the rule instance to be evaluated.

Definition 8. *Let P be a rule pattern and I an rule instance to be evaluated, that conforms to P. The hypothesis query (respectively the hypothesis and conclusion query) of I, denoted Q_H (resp. $Q_{H\wedge C}$), is the conjunction of:*
– the hypothesis query (resp. the hypothesis and conclusion query) associated with P;
– the set of of selection criteria of the form attribute = value obtained as follows. Let π be a projection of P into I. Let $sc = [c, m]$ be a concept vertex of the hypothesis of P (resp. of whole P) and $sc' = [c', m']$ its image in I through π:

- *if $c' < c$ (with the meaning of the specialization relation) then a selection criterion is created, whose attribute is the type attribute associated with sc and whose value is c' (high level concept type extracted from the data, that corresponds to a value taken by the type attribute associated with sc). If moreover c' has subtypes, in the set of concept types, then for each of these subtypes c'' a selection criterion is created, whose attribute is the type attribute associated with sc and whose value is c'';*
- *if $m' < m$ (with the meaning of the specialization relation) then a selection criterion is created, whose attribute is the marker attribute associated with sc and whose value is m'.*

Remark 2. If there are several projections from P into I, a hypothesis query (resp. a hypothesis and conclusion query) of I is obtained for each of these projections. Only the hypothesis query (resp. the hypothesis and conclusion query) that provides the greatest result (greatest number of data) is retained: it is estimated to correspond to the expected specialization of the rule pattern.

Example 9. SQL query of the hypothesis of the rule instance of Figure 2:

```
SELECT COUNT(result.id_result) FROM result, food_product, component, study, operation
WHERE result.id_foodProduct = food_product.id_foodProduct AND
study.id_operation = operation.id_operation AND result.id_subComponent = component.id_component
```

```
AND result.id_property = property.id_property AND result.id_study = study.id_study
// Part of the query which is added to that of the pattern (Fig. 7)
AND operation.name_operation = 'Cooking in water' AND property.name_property = 'Content'
AND (component.name_component = 'Vitamin'
// Part which corresponds to the subtypes of the concept type Vitamin
OR component.name_component='Liposoluble vitamin' OR component.name_component='Vitamin E' ...)
```

The results of the queries Q_H and $Q_{H \wedge C}$ are respectively n_H and $n_{H \wedge C}$, which allows one to calculate the validity rate of the rule instance. The rules whose validity rate is strictly lower than 100 % have exceptions within the database. These exceptions can be visualized by the user.

Example 10. The validity rate V of the rule of Figure 2 is equal to 97.5 %.

4 Application

The presented methods have been applied within a project concerning food quality. The objective is to better control the parameters that impact the nutritional quality of food products. After a presentation of the work environment, we describe the experimental data and the expert knowledge of the project, then we illustrate the validation of expert knowledge.

4.1 Work Environment

The experimental data are stored in a MySQL database. The data can be entered and consulted by domain specialists through a web browser, using PHP forms. The database is composed of about thirty tables and currently contains the detailed results of approximately 600 experiments.

The expert rules are represented using the interface CoGUI (http://www.lirmm.fr/~gutierre/cogui/). About 150 expert rules are available, about twenty are used to test the proposed methodology, starting with the simplest cases.

The communication between the two systems is based on a JDBC connection.

4.2 Experimental Data Description

Designed for scientists and industrials in the domain of food processing, the experimental data acquisition and consultation tool (in English language) gathers scientific data, as exhaustive as possible, from international publications dealing with nutritional qualities of durum wheat based food products, and describing the impact of food processing on these qualities. Such scientific publications ususally include information about experimental measures concerning nutritional composition analysis, results about the impact of unit operations on nutritional qualities, data concerning the influence of parameters of the unit operation and of other unit operations, bibliographical references, etc.

4.3 Expert Knowledge Description

The concept types of the vocabulary used to express expert knowledge were obtained as presented in Section 2. In the vocabulary, most of semantics is expressed through the concept types. The relation types constitute general connectors, as stable as possible. Expert knowledge, represented as conceptual graph rules, expresses qualitatively, for each unit operation that is part of the process of a durum wheat based food product, and for each nutritional component of interest, the known impact of the considered operation on the considered component. The impact can concern a variation in the component content (increase, decrease, stagnation) but also a modification of qualitative properties of the component, such as digestibility, allergenicity, etc.

4.4 Expert Knowledge Validation

The evaluation of expert knowledge can be visualized in two ways by the user: individually, rule after rule, which allows the user to access the experimental data that constitute exceptions to the considered rule; or as a synthetic table containing all the rules available in the application and their validity rates. Figure 5 illustrates the first case. Validity rates spread out from 73 to 100 %.

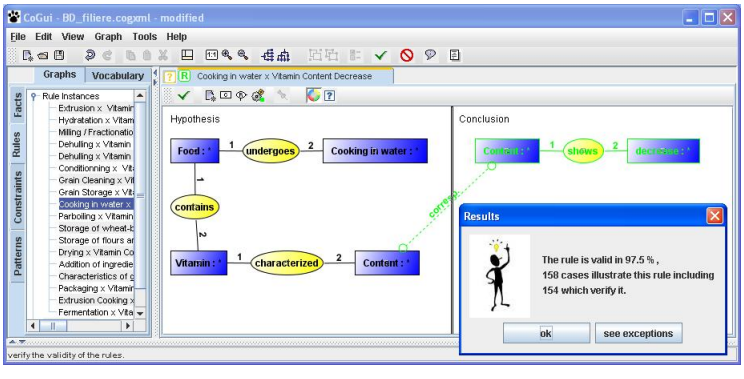

Fig. 5. Evaluation of the validity of an expert rule

5 Conclusion and Perspectives

Given two heterogeneous kinds of information available on a domain (generic expert knowledge expressed as causality rules on the one hand, detailed experimental results on the other hand) represented in two distinct formalisms (respectively the conceptual graph model and the relational model), in this article we proposed two stages for the construction of an expertise on the domain: (i) the generation of an ontology, by identifying high level concept types within the relational diagram and the relational data, and by organizing these concept types into a hierarchy. This stage is automatic but is subject to expert validation;

(ii) the evaluation of the validity of expert knowledge within the experimental data. This stage is based on the notion of rule pattern in the conceptual graph formalism, associated with a corresponding SQL query schema in the relational formalism. The evaluation of a rule instance that conforms to a given rule pattern is then processed by completing the query schema associated with the pattern by selection criteria specific to the considered rule instance. This stage is automatic, which is allowed by annotations of the rule patterns. The proposed methodology thus relies on the cooperation of the two kinds of information and the two heterogeneous formalisms. It is illustrated by a concrete application case.

The longer-term objective of the causality rules is a use for decision-making: given a user's query that expresses a required goal, the issue is to determine which conditions allow one to achieve this goal, by identifying rules whose conclusions would satisfy the required goal, and whose hypotheses would provide sufficient conditions to obtain it.

References

1. Sowa, J.F.: Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley, London, UK (1984)
2. Bos, C., Botella, B., Vanheeghe, P.: Modeling and Simulating Human Behaviors with Conceptual Graphs. In: Delugach, H.S., Keeler, M.A., Searle, L., Lukose, D., Sowa, J.F. (eds.) ICCS 1997. LNCS, vol. 1257, pp. 275–289. Springer, Heidelberg (1997)
3. Genest, D.: Extension du modèle des graphes conceptuels pour la recherche d'informations. PhD thesis, Université Montpellier II (December 2000)
4. Mugnier, M.-L.: Knowledge Representation and Reasoning based on Graph Homomorphism. In: Ganter, B., Mineau, G.W. (eds.) ICCS 2000. LNCS, vol. 1867, pp. 172–192. Springer, Heidelberg (2000)
5. Pernelle, N., Rousset, M.C., Ventos, V.: Automatic construction and refinement of a class hierarchy over multi-valued data. In: Siebes, A., De Raedt, L. (eds.) PKDD 2001. LNCS (LNAI), vol. 2168, pp. 386–398. Springer, Heidelberg (2001)
6. Tilley, T.A., Cole, R.J., Becker, P., Eklund, P.W.: A survey of formal concept analysis support for software engineering activities. In: Ganter, B., Stumme, G., Wille, R. (eds.) Formal Concept Analysis. LNCS (LNAI), vol. 3626, pp. 250–271. Springer, Heidelberg (2005)
7. Sendall, S., Kozaczynski, W.: Model transformation: The heart and soul of model-driven software development. IEEE Software 20(5), 42–45 (2003)
8. Euzenat, J., Le Bach, T., Barrasa, J., Bouquet, P., De Bo, J., Dieng-Kuntz, R., Ehrig, M., Hauswirth, M., Jarrar, M., Lara, R., Maynard, D., Napoli, A., Stamou, G., Stuckenschmidt, H., Shvaiko, P., Tessaris, S., Van Acker, S., Zaihrayeu, I.: State of the art on ontology alignment. deliverable 2.2.3, Knowledge web NoE (2004)
9. Kolaitis, P.G., Vardi, M.Y.: Conjunctive-Query Containment and Constraint Satisfaction. In: Proceedings of PODS'98 (1998)
10. Haemmerlé, O., Buche, P., Thomopoulos, R.: The MIEL system: uniform interrogation of structured and weakly structured imprecise data. Journal of Intelligent Information Systems (2006)