TAAABLE: Text Mining, Ontology Engineering, and Hierarchical Classification for Textual Case-Based Cooking

Fadi Badra, Rokia Bendaoud, Rim Bentebibel, Pierre-Antoine Champin, Julien Cojan, Amélie Cordier, Sylvie Després, Stéphanie Jean-Daubias, Jean Lieber, Thomas Meilender, et al.

TAAABLE: Text Mining, Ontology Engineering, and Hierarchical Classification for Textual Case-Based Cooking

Fadi Badra[1], Rokia Bendaoud[1], Rim Bentebitel[2], Pierre-Antoine Champin[3], Julien Cojan[1], Amélie Cordier[3], Sylvie Després[2], Stéphanie Jean-Daubias[3], Jean Lieber[1], Thomas Meilender[1], Alain Mille[3], Emmanuel Nauer[1], Amedeo Napoli[1], Yannick Toussaint[1]

[1]Orpailleur team, LORIA UMR 7503 CNRS, INRIA, Nancy Universities BP 239 54 506 Vandœuvre-lès-Nancy, France, First-Name.Last-Name@loria.fr
[2]LIPN CNRS, UMR 7030, Institut Galilée — Université Paris-Nord 99, avenue Jean-Baptiste Clément 93 430 Villetaneuse, France, First-Name.Last-Name@lipn.univ-paris13.fr
[3]LIRIS CNRS, UMR 5202, Université Lyon 1, INSA Lyon, Université Lyon 2, ECL 43, bd du 11 Novembre 1918, Villeurbanne Cedex, France, First-Name.Last-Name@liris.cnrs.fr

Abstract. This paper presents how the TAAABLE project addresses the textual case-based reasoning challenge of the CCC, thanks to a combination of principles, methods, and technologies of various fields of knowledge-based system technologies, namely CBR, ontology engineering (manual and semi-automatic), data and text-mining using textual resources of the Web, text annotation (used as an indexing technique), knowledge representation, and hierarchical classification. Indeed, to be able to reason on textual cases, indexing them by a formal representation language using a formal vocabulary has proven to be useful.

Keywords: Textual case-based reasoning, case-based cooking, text mining, ontology engineering, hierarchical classification

1 Introduction

The participants in the *Computer Cooking Contest* (CCC[1] are invited to propose a knowledge-based system—e.g., a case-based reasoning (CBR) system—for solving cooking problems using a given set of recipes in a textual form, with a shallow XML structure. The goal of the TAAABLE project is to propose such a system, based on a textual CBR approach and using the recipe book as a case base.[2] The challenges that the project wishes to address are the "compulsory challenge" and the "negation challenge".

Section 2 presents the general notions and notations about CBR used in TAAABLE. The TAAABLE system is composed on an on-line part: a CBR system presented in section 3. This CBR system involves some knowledge requirements; section 4 explains how this requirements are fufilled thanks to knowledge acquisition and extraction. Section 5 presents some preliminary results. Section 6 concludes the paper.

[1] http://www.wi2.uni-trier.de/eccbr08/index.php?task=ccc
[2] The TAAABLE system is on line at the following address: http://labotalc.loria.fr/TaaableCCC/.

2 CBR: basic notions and notations

CBR aims at answering queries Q with the help of a case base, a case being an answer to a previous query. In TAAABLE, a case is a recipe, denoted by R. Retrieval aims at selecting in the case base (i.e., the recipe book) cases *similar* to the query Q. "Similar" corresponds here either to an exact matching (the retrieved recipe answers the query) or to an approximate matching, leading to an adaptation of the retrieved case for answering the query.

3 Structure of the on-line part of TAAABLE

The on-line part of TAAABLE aims at solving cooking problems. It uses an ontology, an indexed case base, and adaptation knowledge. It is based on retrieval and adaptation.

3.1 Modeling and Representation issues

The language used for representing knowledge units in this section is propositional logic. In practice, description logics have also been used, but they are not presented in this paper, for the sake of simplicity.

A cooking ontology. An ontology $\mathcal{O}$ has been built that defines the main concepts and relations about cooking. The way it has been elaborated is presented in section 4.1. The focus has been put on ingredients and recipe types (not on, e.g. kitchen ustensils) that are crucial for this project: the queries are based on these notions and the inferences are not based on other cooking notions (for the current version of the system). Therefore, `Recipe` is the main concept of this cooking ontology. `Recipe` relates to all other main concepts at a higher level in order to describe a cooking procedure. As shown on Figure 1, a recipe has some ingredients and produces a dish. Both ingredients and dishes are food components (`Ingredient` and `Dish` are two subclasses of `FoodComponent`). A food component consists in a given quantity of food, qualified by a given state: an instance of `FoodComponent` uses an instance of `Quantity` to fill its quantity attribute, an instance of `Food` to which the measurement refers, and an instance of `Qualifier` to make precise the state of the food. An instance of `Food` can be replaced in a recipe with an equivalent instance of `Food`.

The main part of the ontology consists in a hierarchy of classes, which corresponds, in propositional logic, to a set of implication axioms $a \implies b$. For example, the axiom $\texttt{apple} \implies \texttt{fruit}$ of $\mathcal{O}$ states that apples are fruits.

Representation of queries and of recipe indexes. A query in TAAABLE specifies recipe types (salad, vegetarian, italian, etc.), desired ingredients, and undesirable ingredients. It is represented by a conjunction of literals based on a vocabulary $\mathcal{V}$, which is a finite set of propositional variables. For example, searching a Chinese dessert recipe with fruits but without ginger corresponds to the query

$$Q_{ex} = \texttt{Chinese} \wedge \texttt{dessert} \wedge \texttt{fruit} \wedge \neg\texttt{ginger}$$

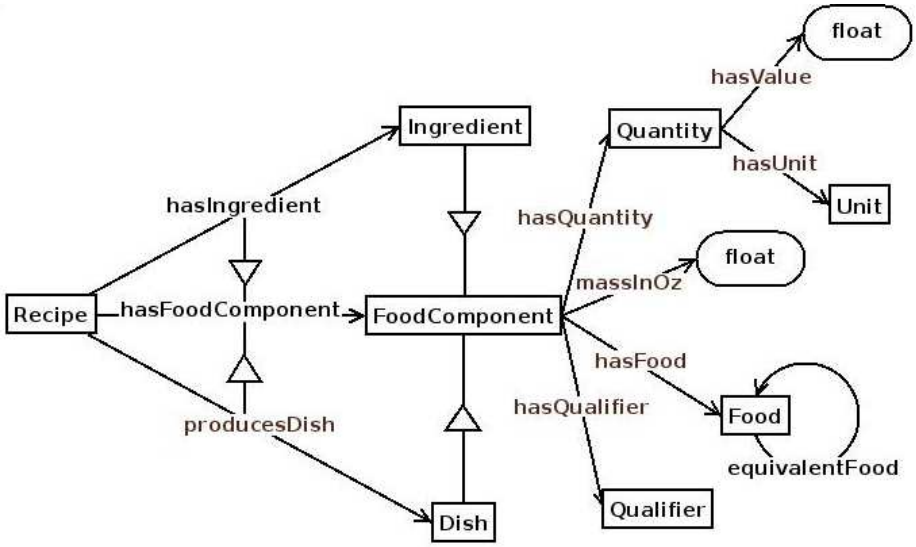

Fig. 1. A cooking ontology (excerpt).

$\texttt{Chinese}, \texttt{dessert}, \texttt{fruit}, \texttt{ginger} \in \mathcal{V}$ standing for "Chinese recipe", "recipe of a dessert", "recipe with fruit", and "recipe with ginger". In these examples, the propositional variables of the domain represents recipe classes (e.g., `ginger` is used as an abbreviation of `RecipeWithGinger`).

A recipe R is also indexed by a conjunction of literals $Idx(R)$. More precisely, the index $Idx(R)$ of a recipe R is the most specific expression in the query language that describes R. For example, if R_{ex} is a recipe of a Chinese dessert with pear, cherry, ananas, sugar, and no other ingredients, then it is indexed by:

$$Idx(R_{ex}) = \texttt{Chinese} \wedge \texttt{dessert} \wedge \texttt{pear} \wedge \texttt{cherry} \wedge \texttt{ananas} \wedge \texttt{sugar} \wedge \textit{Nothing else}$$

where *Nothing else* is a conjunction of the negative litterals $\neg a$, for all the $a \in \mathcal{V}$ such that $\texttt{Chinese} \wedge \texttt{dessert} \wedge \texttt{pear} \wedge \texttt{cherry} \wedge \texttt{ananas} \wedge \texttt{sugar} \not\models_{\mathcal{O}} a$. This kind of "closed world assumption" means that every propositional variable that is not stated in the index recipe, either explicitely or through deduction modulo the ontology, is forced to false. For example, if no ingredient a stated in the recipe is meat ($a \not\models_{\mathcal{O}} \texttt{meat}$), then $Idx(R)$ contains the literals $\neg\texttt{meat}$, $\neg\texttt{chicken}$, $\neg\texttt{beef}$, etc.

A recipe R exactly matches a query Q if

$$Idx(R) \models_{\mathcal{O}} Q \tag{1}$$

i.e., given the knowledge $\mathcal{O}$, every literal stated in Q is satisfied by the index $Idx(R)$.

If, e.g., $\texttt{ananas} \implies \texttt{fruit} \in \mathcal{O}$ and no positive literal a of the query is under `ginger`, then R_{ex} exactly matches Q_{ex}.

With each recipe R is associated an index $Idx(R)$ and, conversely, given an index $Idx(R)$, the corresponding recipe R is accessed through a pointer.

Adaptation knowledge: reformulations and substitutions. The adaptation knowledge is constituted by a set of reformulations. A reformulation is a pair $(\mathtt{r}, \mathcal{A}_\mathtt{r})$ where $\mathtt{r}$ is a binary relation between queries and $\mathcal{A}_\mathtt{r}$ is an adaptation function associated with $\mathtt{r}$ [6]. It has the following semantics: if $Q_1\ \mathtt{r}\ Q$ then, for every recipe R_1 matching the query Q_1, $\mathcal{A}_\mathtt{r}(Q_1, R_1, Q) = R$, a recipe matching the query Q.

In the current version of TAAABLE, a reformulation $(\mathtt{r}, \mathcal{A}_\mathtt{r})$ is given by a substitution σ such that $Q_1\ \mathtt{r}\ Q$ if $Q_1 = \sigma(Q)$ and $\mathcal{A}_\mathtt{r}(Q_1, R_1, Q) = \sigma^{-1}(R_1)$, where σ^{-1} is the inverse of the substitution σ. Thus, the adaptation knowledge is given by a set of substitutions.

For example, let $Q = \mathtt{dessert} \wedge \mathtt{lemon}$ be a query and $\sigma = \mathtt{lemon} \rightsquigarrow \mathtt{orange}$ be a substitution. Then, $Q_1 = \sigma(Q) = \mathtt{dessert} \wedge \mathtt{orange}$. If a recipe R_1 exactly matches Q_1, $\mathit{–Idx}(R_1) \models_\mathcal{O} Q_1$– then $R = \sigma^{-1}(R_1)$, obtained by substitution of orange by lemon in R_1, matches Q $\mathit{–Idx}(R) \models_\mathcal{O} Q$.

A substitution $\sigma = \alpha \rightsquigarrow \beta$ is given by two conjunctions of literals α and β. σ is applicable to a query Q if $Q \models_\mathcal{O} \alpha$ and, when it is the case, $\sigma(Q)$ is obtained from Q by first removing the literals of α and then adding the literals of β: if Q, α, and β are considered as sets of literals closed modulo the ontology $\mathcal{O}$, then $\sigma(Q) = (Q \backslash \alpha) \cup \beta$.

For example, to express that, for Thai cuisine, basil can be substituted by mint, the substitution $\sigma = \mathtt{Thai} \wedge \mathtt{basil} \rightsquigarrow \mathtt{Thai} \wedge \mathtt{mint}$ can be used.

A large part of the substitutions used in the system is simply based on the ontology: if a is a subclass of b in $\mathcal{O}$ (i.e., $(a \implies b) \in \mathcal{O}$), then the *substitution by generalization* $a \rightsquigarrow b$ may be considered. If $a \rightsquigarrow b$ has been accepted as adaptation knowledge, this means that a food of type a may be substituted by any food of type b. For example, $(x \implies \mathtt{citrus}) \in \mathcal{O}$, for any $x \in \{\mathtt{grapefruit}, \mathtt{lemon}, \mathtt{orange}\}$. Thus, if $\mathtt{orange} \rightsquigarrow \mathtt{citrus}$ is in the adaptation knowledge, then substituting orange by lemon or by grapefruit is likely to produce a "correct recipe".

A numerical value $\mathtt{cost}(\sigma) > 0$ is associated with each substitution, that measures the "adaptation effort". This cost represents a preference between adapted recipes: if $\mathtt{cost}(\sigma) < \mathtt{cost}(\tau)$ then, a recipe $\sigma^{-1}(R_1)$ is prefered to a recipe $\tau^{-1}(R'_1)$, where R_1 and R'_1 are two recipes from the recipe book that respectively match $\sigma(Q)$ and $\tau(Q)$.

3.2 Case-based problem-solving process: retrieval and adaptation

Recipe retrieval algorithm is based on the algorithms of strong and smooth classification in an index hierarchy [5].

Strong classification points out the indexes $Idx(R)$ that exactly matches the query Q, as in equation (1). The algorithm is based on a depth-first search of the index hierarchy. If no such index exists, then smooth classification is executed.

Smooth classification aims at finding a modification Q' of the query Q such that there exists at least one index matching exactly Q'. The modification of Q into Q' is based on a *similarity path*, i.e., a composition $\mathit{SP} = \sigma_p \circ \sigma_{p-1} \circ \ldots \circ \sigma_1$ of substitutions: $Q_p = \sigma_p(\sigma_{p-1}(\ldots(\sigma_1(Q))\ldots))$. Let $\mathtt{cost}(\mathit{SP}) = \sum_{i=1}^{p} \mathtt{cost}(\sigma_i)$. The smooth classification algorithm combines an A* search for finding the similarity path *SP* minimizing

$\texttt{cost}(SP)$ such that $Q_p = SP(Q)$ exactly matches at least one of the indexes of the recipe.

Complexity remark. The use of propositional logic for case retrieval, with a large case base and hundreds of propositional variables, may be computationally costly. Actually, it is not the case in our application: only a fragment of propositional logic leading to polynomial problems is used. In practice, strong classification is very efficient and smooth classification depends on the cost of the first similarity path giving a non-empty set of indexes: the higher is this cost, the more smooth classification requires time and memory, and the more adaptation effort is needed.

Recipe adaptation. If strong classification is successful, then there is no need to adapt the retrieved recipe. If smooth classification is successful, it returns a recipe R_p and a similarity path $SP = \sigma_p \circ \sigma_{p-1} \circ \ldots \circ \sigma_1$ such that $Idx(R) \models_{\mathcal{O}} SP(Q)$. Then, adaptation consists simply in applying every σ_i^{-1} successively: $R = SP^{-1}(R_p) = \sigma_1^{-1}(\sigma_2^{-1}(\ldots \sigma_p^{-1}(R_p)\ldots))$ is the adapted recipe.

4 Off-line part of TAAABLE

The previous section has highlighted three ingredients: the ontology, the indexed case base, and the adaptation knowledge. All of them have required a large amount of work that is presented in this section.

4.1 Ontology building

Principles for building manually the general part of the ontology. The ontology $\mathcal{O}$ has been built in order to help the conception of the retrieval and adaptation processes of the TAAABLE system. Therefore, the conceptual choice for the ontology development has been strongly driven by the goal of this particular CBR system. The ontology design process is achieved through the use of the standard methodology that distinguishes the following phases: feasibility study, requirement analysis, conceptualization, and deployment [1]. In this presentation, only the conceptualization phase is considered. Three subtasks have been carried on: development of the cooking conceptual model, formalization of the domain, and implementation in OWL language. The reuse of existing ontologies has been carefully examined but no more considered as the examined ontologies did not cover what was intended to be reached in this project. During the elaboration of the cooking conceptual model, several main classes were identified: `Recipe`, `Ingredient`, `FoodComponent`, `Food`, `Action`, `Amount`, and `Utensil`. These classes were sufficiently independent to decide to build a modular ontology. This conceptual choice should facilitate the enrichment of the different modules (because the concepts included in these modules are disjoint) without changing the content of the others. The main relations linking these modules have been manually defined. Once the choice of a modular ontology was adopted, the building process has been shared out among the different partners in order to take advantage of their skills. The general part of the ontology has been built manually and formalized in OWL. This step has been essential to obtain

the skeleton of the conceptual model. It appeared that some modules play a central role in the retrieval and adaptation processes. Therefore, it was necessary to refine further the design in order to reach a finest granularity level. The parts of the ontology about the dish types, the dish origins, and the food types are finely grained according to existing terminological resources and manual expertise. How these concepts are structured is detailed below.

DishType and DishOrigin hierarchies. These two hierarchies have been built manually. Starting from the organisation of dish types and dish origins in the Recipe Source database[3], a list of general dish types and dish origins has been collected, and hierarchically organised following the specialisation relation.

Concretely, the dish origin hierarchy has only two levels. The first level classifies dishes following their regions, such as *Africa*, *MiddleEast*, *Asia*, *Europe*, etc. Each first level concept is specialised, on the second level, by the country, origin of the dishes. For example, *British*, *French*, *German*, etc. are subconcepts of *Europe*.

In the same way, the DishType hierarchy is mainly organised into two levels. At the first level, there are concepts like *BakedGood*, *Burger*, *Dessert*, *MainDish*, etc. The second level details, if necessary, the first level concepts. For example, *BakedGood* is divided into *Bagel*, *Biscuit*, *Bread*, *Muffin*, *Brownie*, *Cookie*, etc. However, these concepts are no more deeply detailed even if more specific categories exist in Recipe Source, as it is for example the case for *Cookie* which is subdivided into *Apple Cookie*, *Chocolate Chip Cookie*, *Diabetic Cookie*, etc. These concepts could indeed be defined by conjunction of being a *Cookie* and by containing, or not, some specific food.

Food hierarchy. The `Food` hierarchy was also built manually starting from the Cook's Thesaurus[4] and from the recipe book. The Cook's Thesaurus is a cooking encyclopedia that covers thousands of ingredients, including synonyms and suggested substitutions. More than 250 HTML files were examined in order to extract an initial `Food` hierarchy. At the same time, a terminological database is built in order to associate to each food subconcept a linguistically preferred form (e.g., `bok choy`) as well as a set of morphological variants or synonyms (e.g., `pak choy, pak choi, Chinese cabbage, Chinese mustard cabbage`, etc.). The food hierarchy and the terminological database are then manually enriched by adding new concepts and new lexical forms that occur in the recipe book but do not occur in the Cook's Thesaurus. This extension process is iterative and stops when each food component of the recipe book can be linked to a food concept by the annotation process (see below). The current version of the food hierarchy contains 3936 concepts, organised upon 8 levels of specialisation. The terminological base contains 7824 lexical forms, without taking into account singular/plurial variations.

[3] http://www.recipesource.com/
[4] http://www.foodsubs.

4.2 Annotation and indexing process

The case base engine needs a formal representation of a recipe. The annotation process aims at formally representing the content of a recipe as well as defining its categories. This process is in between what is usually called *controlled indexing* [4] where terms come from a predefined terminology and *semantic annotation* [8] where terms (named entities, sequences of words) are explicitly associated with the respective and most specific classes in the ontology. The result of the annotation of a recipe is a set of concepts indexing the recipe. The current state of our prototype does not deal with the preparation part of recipes. Only, the list of ingredients is parsed. First, each ingredient entry in the recipe is parsed and split into the following 4-uple (<quantity>,<unit>,<food-component>,<modifiers>). For example, the entry <IN>1/3 cup milk</IN> is parsed as (1/3,cup,milk,_). The terminological database guides the parsing process. For instance, as light brown sugar is a lexical form associated to the food component concept light brown sugar in the food hierarchy, the entry <IN>1/2 c Packed light brown sugar</IN> is parsed as (1/2,cup,light brown sugar, packed). At present, only the list of food components is kept for indexing.

For example, the recipe entitled Cinnamon Rolls is indexed by the conjunction of the ingredients sweet dough mixture, brown sugar, pecans, dark seedless raisins, ground cinnamon, butter or margarine, and sugar glaze (butter or margarine is a concept in the ontology which is more general than both concepts butter and margarine).

Finally, the annotation process has to index recipes following the region and the country (e.g., Asia, Chinese) and following the dish type (e.g., main dish, dessert). As there is no indication concerning these types in the recipe book, Recipe Source is used again in order to build a corpus where recipes are assigned to types. The process to type a recipe book recipe types takes 3 steps. For a given recipe:

- If there exists a recipe in Recipe Source with the same title, then its types are assigned to the Recipe Book recipe;
- If the title of the recipe (e.g., Chinese Vegetable Soup) contains keywords corresponding to subconcepts of DishType (e.g., soup) or DishOrigin (e.g., Chinese) then these types are assigned to the recipe;
- A set of association rules has also been extracted from the Recipe Source corpora, using the data-mining toolkit CORON [7]. According to exact associations rules (with a 100% confidence) of the form <set of ingredients> $\longrightarrow$ <origin or dish type> (e.g., vanilla bean, banana, chocolate $\longrightarrow$ dessert), type assignations can be done as follows: if part of the recipe matches the left-hand side of the rule, then the type in the right-hand side is assigned to the recipe.

For example, the recipe entitled Cinnamon Rolls is indexed by some ingredients, as presented below, but also by the following recipe types: baked good, bread, pastry, and roll.

4.3 Adaptation knowledge acquisition

The substitution base is a set of equivalence classes of food. The knowledge about food substitution has been acquired from the recipe book and from the Cook's The-

saurus. This process is automatic for the Cook's Thesaurus by extracting the suggested substitution according patterns based on the HTML structure. For the recipe book, the substitution are manually extracted from the food components containing a choice. For example, the possibility to substitute *butter* by *margarin*, or inversely, is acquired from the sequence *butter or margarin* which appears in the food components.

5 Preliminary Results

The core of the TAAABLE system has been implemented in Java.

The first step of the TAAABLE process is initialisation: it consists in parsing the knowledge containers, i.e., the vocabulary $\mathcal{V}$, the ontology $\mathcal{O}$, the set of substitutions, and, finally, the recipe book. The parsing of the recipe book is associated with indexing, that consists in organizing hierarchically the indexes. With the current knowledge containers, the time taken for this initialization is about 30 seconds (on a current PC computer).

The TAAABLE system retrieves cases for all the six exercice queries proposed in the contest (the queries 1 to 5 for the compulsory challenge and the query 6 for the negation challenge). For a given CBR session, let *SP* be the similarity path, p be the number of susbtitutions in *SP* ($p = 0$, means that the similarity path is empty and thus that strong classification succeeds), $c = \texttt{cost}(SP)$, and let n be the number of retrieved recipes R (i.e., the R such that $Idx(R) \models_{\mathcal{O}} SP(Q)$). The queries are recalled below and associated with the information about *SP*, p, c, and n:

$Q1$ "Cook a main dish with meat and cauliflower." ($p = 0$, $c = 0$, $n = 2$)
$Q2$ "I would like to have a nut-free cake." ($p = 0$, $c = 0$, $n = 31$)
$Q3$ "Prepare a Chinese dessert with fruit." ($p = 1$, $c = 3.33$, $n = 15$, $SP = (\texttt{dessert} \rightsquigarrow \texttt{dish\_type})$: the fact that a dessert is required is removed from $Q3$!)
$Q4$ "Cook a main dish with turkey, pistachio, and pasta." ($p = 3$, $c = 7.0$, $n = 1$, $SP = (\texttt{turkey} \rightsquigarrow \texttt{chicken}) \circ (\texttt{pasta} \rightsquigarrow \texttt{grain\_product}) \circ (\texttt{pistachio\_nut} \rightsquigarrow \texttt{fresh\_beans})$)
$Q5$ "I would like to cook eggplant soup." ($p = 0$, $c = 0$, $n = 1$)
$Q6$ "I want to have a salad with tomato but I hate garlic and cucumber." ($p = 0$, $c = 0$, $n = 2$)

Any of these requests has required less than 1 second for retrieval, except $Q4$, that has required less than 4 seconds.

The results of these tests are not fully satisfactory yet: some more work on knowledge acquisition and extraction remains to be done. Let us consider with more details the result for query $Q4$. In the form of a conjunction of literals, this query is

$$Q4 = \texttt{main-dish} \wedge \texttt{turkey} \wedge \texttt{pasta} \wedge \texttt{pistachio}$$

and thus, $SP(Q4) = \texttt{main-dish} \wedge \texttt{chicken} \wedge \texttt{grain\_product} \wedge \texttt{fresh\_beans}$

It returns only one recipe, which is the only recipe R such that $Idx(R) \models_{\mathcal{O}} SP(Q4)$. This is the recipe entitled "Hazelnut Chicken with Orange Thyme Cream". Indeed, this recipe is indexed by

$$Idx(R_{ex}) = \texttt{main-dish} \wedge \texttt{chicken} \wedge \texttt{all-purpose\_flour} \wedge \texttt{hazelnut} \wedge \textit{etc.}$$

where *etc.* is a conjunction of other literals, irrelevant for our example. The ontology contains the following axioms, that explain why the recipe has been retrieved:

$$\begin{aligned} \texttt{all-purpose\_flour} &\Longrightarrow \texttt{wheat\_flour} \\ \texttt{wheat\_flour} &\Longrightarrow \texttt{grain\_product} \\ \texttt{hazelnut} &\Longrightarrow \texttt{fresh\_beans} \end{aligned}$$

The adaptation consists in following SP^{-1} on the recipe. Since

$$\begin{aligned} SP^{-1} = &(\texttt{fresh\_beans} \rightsquigarrow \texttt{pistachio\_nut}) \circ (\texttt{grain\_product} \rightsquigarrow \texttt{pasta}) \\ &\circ (\texttt{chicken} \rightsquigarrow \texttt{turkey}) \end{aligned}$$

Thus, the adaptation takes 3 steps:

- Substituting `hazelnut` (a subclass of `fresh_beans`) by `pistachio_nut`;
- Substituting `wheat_flour` (a subclass of `grain_product`) by `pasta`;
- Substituting `chicken` by `turkey`.

Here, substitution 2 is questionable. In fact, a more sophisticated adaptation than this simple substitution should be carried out in order to obtain an applicable recipe.

6 Conclusion and ongoing work

This paper presents how the TAAABLE project addresses the textual CBR challenge of the CCC, thanks to a combination of principles, methods, and technologies of various fields of knowledge-based system technologies, namely CBR, ontology engineering (manual and semi-automatic), data and text-mining using textual resources of the Web, text annotation (used as an indexing technique), knowledge representation, and hierarchical classification. Indeed, to be able to reason on textual cases, indexing them by a formal representation language using a formal vocabulary has proven to be useful.

The first version of the system relies on simple choices: a propositional representation language (that is not enough expressive for representing, e.g., quantities), a vocabulary restricted to ingredients and recipe types (that does not handle, e.g., the actions of the preparation), and a CBR process relying on simple choices. The ongoing work (until the contest) and the future work (after the contest, if –what we hope– the TAAABLE project goes on) will concentrate on improving these features. In particular, case retrieval returns sometimes a set of more than 5 recipes, and further versions should be able to make a clever choice among them (e.g., thanks to more detailed indexes). Some studies on adaptation knowledge acquisition for TAAABLE will be carried on. In particular, the CABAMAKA system [3] will be used for this purpose. Moreover, the IAKA methodology [2] has to be applied for acquiring adaptation knowledge and elements of the ontology by testing the system with various queries: the comparison of the system results and of what is expected should be useful to point out some missing or faulty pieces of knowledge.

Acknowledgements

The participants of the TAAABLE project wish to thank the organisers of the CCC for providing this benchmark, that entails many interesting problems, and the need to collaborate with researchers in various domains on knowledge engineering.

References

1. P. Cimiano, J. Volker, and R. Studer. Ontologies on Demand? A Description of the State-of-the-Art, Applications, Challenges and Trends for Ontology Learning from Text. *Information, Wissenschaft und Praxis*, 6–7(57):315–320, 2006.
2. Amélie Cordier, Béatrice Fuchs, Léonardo Lana de Carvalho, Jean Lieber, and Alain Mille. Opportunistic Acquisition of Adaptation Knowledge and Cases – The IakA approach. In *To appear in 9th european conference on Case-Based Reasoning (ECCBR-08)*, 2008.
3. M. d'Aquin, F. Badra, S. Lafrogne, J. Lieber, A. Napoli, and L. Szathmary. Case Base Mining for Adaptation Knowledge Acquisition. In M. M. Veloso, editor, *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, pages 750–755. Morgan Kaufmann, Inc., 2007.
4. T. C. Jo, J. H. Seo, and K. Hyeon. Topic Spotting on News Articles with Topic Repository by Controlled Indexing. In Springer Berlin / Heidelberg, editor, *Intelligent Data Engineering and Automated Learning — IDEAL 2000. Data Mining, Financial Engineering, and Intelligent Agents*, volume 1983 of *Lecture Notes in Computer Science*, pages 89–99, 2008.
5. J. Lieber and A. Napoli. Using Classification in Case-Based Planning. In W. Wahlster, editor, *Proceedings of the 12th European Conference on Artificial Intelligence (ECAI'96), Budapest, Hungary*, pages 132–136. John Wiley & Sons, Ltd., 1996.
6. E. Melis, J. Lieber, and A. Napoli. Reformulation in Case-Based Reasoning. In B. Smyth and P. Cunningham, editors, *Fourth European Workshop on Case-Based Reasoning, EWCBR-98*, Lecture Notes in Artificial Intelligence 1488, pages 172–183. Springer, 1998.
7. L. Szathmary and A. Napoli. Coron: A framework for levelwise itemset mining algorithms. In B. Ganter, R. Godin, and E. Mephu Nguifo, editors, *Supplementary Proceedings of The Third International Conference on Formal Concept Analysis – ICFCA'05, Lens, France*, pages 110–113, Feb 2005.
8. V. Uren, Ph. Cimiano, J. Iria, S. Handschuh, M. Vargas-Vera, E. Motta, and F. Ciravegna. Semantic annotation for knowledge management: Requirements and a survey of the state of the art. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, 4(1):14–28, 2006.