Knowledge extraction for improving case retrieval and recipe adaptation

Julien Cojan, Valmi Dufour-Lussier, Emmanuelle Gaillard, Jean Lieber, Emmanuel Nauer, Yannick Toussaint

Taaable 4: Knowledge extraction for improving case retrieval and recipe adaptation

Julien Cojan, Valmi Dufour-Lussier, Emmanuelle Gaillard,
Jean Lieber, Emmanuel Nauer, and Yannick Toussaint

LORIA (UMR 7503—CNRS, INRIA, Nancy University)
BP 239, 54506 Vandœuvre-lès-Nancy, France,
First-Name.Last-Name@loria.fr

Abstract. Taaable 4 is a case-based cooking system which is a contestant in the fourth *Computer Cooking Contest,* inheriting most of the features of its previous versions as well as adding new ones. Two new features both concerning knowledge acquisition using formal concept analysis (FCA) are proposed this year. The first feature uses FCA in order to enrich the domain ontology (especially the ingredient hierarchy), making the case retrieval more progressive and more precise. The second feature addresses explicitly the adaptation challenge: given a recipe R and some constraints, how can R be adapted. To compute the best way to adapt R, we propose a FCA-based method for extracting adaptation knowledge. These two knowledge extraction processes exploit additional data (73795 recipes) from the *Recipe Source* database.

Keywords: case-based cooking, ontology enrichment, adaptation knowledge acquisition, formal concept analysis
URL of the system: http://taaable.loria.fr

1 Introduction

Taaable has participated in all the previous *Computer Cooking Contests* [1,2,4]. Taaable 4 is the 2011 version of Taaable. Some features are reused: the semantic wiki in which the whole knowledge base (annotated recipes, domain knowledge, retrieval knowledge, and adaptation knowledge) is encoded, and the inference engine. Two features both concerning knowledge acquisition are proposed this year. The first feature, presented in Section 3, addresses the enrichment of the domain ontology (especially the ingredient hierarchy) making the case retrieval more progressive and more precise [8]. The second feature, presented in Section 4, addresses the best way to adapt a recipe R given some constraints, which is the goal of the adaptation challenge. These two new features use a knowledge extraction process based on formal concept analysis applied to additional data coming from the *Recipe Source* database.[1]

[1] http://www.recipesource.com

2 Previous work on the Taaable project

The knowledge-based system TAAABLE is composed of a knowledge base stored and edited in a semantic wiki [6], exploited by a CBR inference engine making several types of adaptation: boolean adaptation for computing ingredient substitutions [1], numerical adaptation for adapting the quantity of ingredients [4], textual adaptation for adapting the recipe preparation [7].

The knowledge base is composed of a cooking domain ontology $\mathcal{O}$ and the recipe base indexed by classes of $\mathcal{O}$. The ontology $\mathcal{O}$ contains about 2800 classes organised in six interrelated hierarchies.[2] The ingredient hierarchy is actually the only one concerned with the refinement process (see Section 3). For example, the class `Ricotta` is under the class `FreshCheese`: every instance of ricotta is an instance of fresh cheese.

A query Q to the CBR engine is expressed as a conjunction of literals, where a literal is either a class of $\mathcal{O}$ or the negation of such a class. For example, asking for "a recipe with tomato and mascarpone" will be expressed by $Q =$ `Tomato` $\wedge$ `Mascarpone`.

Since the recipes are indexed by classes from $\mathcal{O}$, finding the recipes that exactly match Q consists in finding the recipes R such that (1) for each positive literal `C` of Q, there exists a class `D`, under `C` in $\mathcal{O}$ such that R is indexed by `D` and (2) for each negative literal $\neg$`C` of Q, there exists no class `D` under `C` in $\mathcal{O}$ such that R is indexed by `D`. When such an exact match occurs, the matching recipes are returned by the retrieval process and no adaptation is required. Otherwise, the retrieval process aims at finding a minimal generalisation function Γ such that there exists at least one recipe R exactly matching the generalised query $\Gamma(Q)$. Γ is a composition of substitutions `A` $\rightsquigarrow$ `B` where `A` is a direct subclass of `B` in $\mathcal{O}$. For example, the recipe R named "3-step veggie pizza" is indexed by the conjunction `CrescentRoll` $\wedge$ `CreamCheese` $\wedge$ `Mayonnaise` $\wedge$ `Stuffing` $\wedge$ `Broccoli`$\wedge$`Cauliflower`$\wedge$`Tomato`$\wedge$`Onion`$\wedge$`Pepper`$\wedge$`Carrot`.The retrieval process retrieves this recipe for the query Q with generalisation $\Gamma =$ `Mascarpone` $\rightsquigarrow$ `FreshCheese`, because `CreamCheese` is a subclass of `FreshCheese` in $\mathcal{O}$.[3]

Thus, the result of retrieval is a set of recipes R matching a generalisation $\Gamma(Q)$ of the initial query (when there is an exact match, Γ is the identity function). Given R, Γ, and Q, the adaptation of R to Q consists simply in substituting some classes indexing R by (a) following the match from R to $\Gamma(Q)$ and (b) applying the specialisation function Γ^{-1}. For example, the adaptation of the recipe "3-step veggie pizza" will consist in (a) substituting `CreamCheese`

[2] The six hierarchies are about ingredients (2500 classes), dish types (60), dish roles (12), diets (6), geographical locations (60), and cooking actions (150). An example of relation between the ingredient and the diet hierarchies is "is not compatible with" relating, e.g., `Asparagus` to `GoutDiet`.

[3] The current version of TAAABLE is based on an ad hoc formalism that is similar to a (small) fragment of OWL, the web ontology language which is a W3C recommendation. It is planned to use in a further version of TAAABLE a standard representation formalism provided that the complexity of its inferences is low (e.g. a polynomial fragment of OWL).

with FreshCheese and (b) substituting FreshCheese with Mascarpone, which is equivalent to the substitution of cream cheese with mascarpone.

The minimality of the generalisation Γ is computed thanks to a cost function, described in [2]. Function cost is additive for the composition: $\texttt{cost}(\sigma_2 \circ \sigma_1) = \texttt{cost}(\sigma_1) + \texttt{cost}(\sigma_2)$. For a basic substitution $\sigma = \texttt{A} \rightsquigarrow \texttt{B}$ (where A is subsumed by B), the cost is computed as follows:

$$\texttt{cost}(\texttt{A} \rightsquigarrow \texttt{B}) = \mu(\texttt{B}) - \mu(\texttt{A}) \quad \text{with} \quad \mu(\texttt{X}) = \frac{\text{number of recipes with X}}{\text{total number of recipes}}$$

Function cost has the following property: if Γ has a cost computed with respect to a given ontology, and if a new class is added into this ontology (without changing the recipe base), then the cost of Γ will not change. For example, if $\Gamma = \texttt{Mascarpone} \rightsquigarrow \texttt{FreshCheese}$ and if the ItalianFreshCheese class is "added" into the ontology "between" Mascarpone and FreshCheese, the cost of Γ remains unchanged. This property ensures stability for the cost regardless of the introduction of intermediate classes.

An adaptation may be composed of several substitutions but, in the current version of TAAABLE, each substitution of a single ingredient replaces it with a single ingredient. This is linked with the search space of the function Γ: generalisations of conjuncts (e.g. $\Gamma = a \wedge b \rightsquigarrow c$) do not belong to this search space. However, the use of adaptation rules for TAAABLE would lead to substitutions of several ingredients by several ingredients (see [2]). This approach is extended in section 4 for addressing the adaptation challenge.

3 Improving case retrieval by enrichment of the domain ontology

The goal of the refinement process is to insert intermediate classes into the initial hierarchy of the system. These additional classes enable a better distinction of the similarity between classes that were initially immediately subsumed by a common class.

3.1 Ingredient hierarchy refinement

The initial ingredient hierarchy contains classes that are ingredients used in the recipes and more general structuring classes (Vegetable, Fruit, Dairy, Cheese, etc.). All these classes are organised through the specific/generic relation. An excerpt of the FreshCheese subhierarchy is shown in the left-hand side of Fig. 1. From the sole hierarchy, the *similarity* between two *sibling ingredients* (i.e. immediate subclasses of a single class) cannot be ranked. For example, mascarpone is as close to mozzarella as it is to ricotta. Consequently, when a query is generalised for case retrieval, if Mascarpone is generalised to FreshCheese, the set of remaining cases will contain recipes using mozzarella as well as recipes using mascarpone. The refining process aims at gathering initial classes that are similar (i.e. classes that share at least one property). For example Mozzarella and

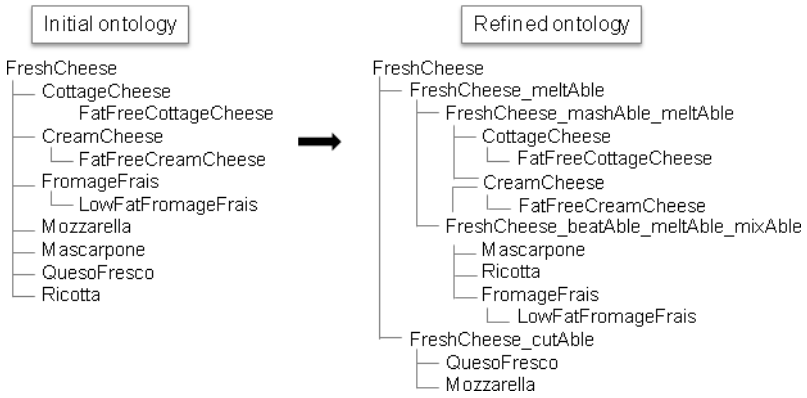

Fig. 1. Initial (left) and refined (right) ontology from the `FreshCheese` class down.

`QuesoFreco` will be gathered under `FreshCheese_cutAble`, the class of fresh cheeses which can be cut (see the right-hand side of Fig. 1). So generalising a class, for example `Mascarpone`, into a parent class will ensure that sibling classes of `Mascarpone` own at least the same properties as `Mascarpone`. The generalisation will be more fine-grained.

3.2 Ingredient hierarchy refinement process

In order to introduce new classes into the initial hierarchy, the initial classes are characterised with additional properties. These properties are exploited using formal concept analysis (FCA) [9]. FCA is a classification method allowing object grouping according to the properties they share.

FCA takes as input what is called a *binary context,* a table in which objects are described by properties. Table 1 shows an example of binary context with 10 objects (which are fresh cheeses), described by two kinds of properties: (1) properties coming from the ontology associating to each ingredient the ingredient classes that subsume it, and (2) the culinary actions that can be performed on them. For example, the object `CottageCheese` has the property of being a `CottageCheese`, which is also a `FreshCheese`. `CottageCheese` is described by 2 (potentially applicable) culinary actions: `mashAble` and `meltAble`, because it can be mashed and melted. To automatise the process, the culinary actions are extracted from 73795 recipes taken from *Recipe Source*. We use the actions both in the ingredient list, which includes some cooking actions (*chopped, diced, crumbled,* etc.), and in the textual preparation, from which the actions applied to the ingredients are extracted using NLP tools [7].

FCA produces *formal concepts* as output. A formal concept is a pair (I, E) where I is a set of properties, called the *intent* of the formal concept, and E is a set of objects, called the *extent* of the formal concept, such that (1) I is the set of all properties shared by objects in E and (2) E is the set of all objects

	FreshCheese	CottageCheese	FatFreeCottageCheese	CreamCheese	FatFreeCreamCheese	FromageFrais	LowFatFromageFrais	Mozzarella	Mascarpone	QuesoFresco	Ricotta	mashAble	meltAble	beatAble	mixAble	sliceAble	crumbleAble	cutAble
CottageCheese	×	×										×	×					
FatFreeCottageCheese	×	×	×									×	×					
CreamCheese	×			×								×	×	×	×			
FatFreeCreamCheese	×			×	×							×	×	×	×			
FromageFrais	×					×							×	×	×			
LowFatFromageFrais	×					×	×						×	×	×			
Mozzarella	×							×								×		×
Mascarpone	×								×				×	×	×			
QuesoFresco	×									×							×	×
Ricotta	×										×		×	×	×			

Table 1. A binary context for cheeses from the `FreshCheese` category, described by generic/specific properties and cooking actions that can be applied to the cheeses.

sharing properties in I. The formal concepts can be ordered by extent inclusion, also called *specialisation* between concepts, into what is called a *concept lattice*. Fig. 2 illustrates the lattice corresponding to the binary context given in Table 1. On this figure, the extents E are given through a reduced form (noted E_r): the objects appear in the most specific concepts, the complete extent can be computed by the union of objects belonging to the subconcepts. So, the top concept (with number 1, in the figure) contains all the objects. In our example, its intent is `FreshCheese`, a property shared by all the objects. By contrast, the bottom concept is defined by the set of all properties. In our example, its extent is empty as none of the objects are described by all the properties.

Many works use FCA to build ontologies (see for example [3,5,11]). As it is done in [3], two types of properties are combined into a unique binary context for creating one lattice. Recipe texts are used in order to characterise the ingredients according to how they are cooked, and the initial ontology is used in order to get the specific/generic relations.

The lattice, presented in Fig. 2, contains the initial classes of the ontology (nodes 6 to 15), and structuring classes have been created (nodes 2 to 5, in grey in the figure). These classes are introduced in the initial ontology, as shown on the right-hand side of Fig. 1, in order to enrich its organisation. Each of them produces a class in the refined ontology, labelled by a concatenation of the intent properties. For example, the node 5 of the lattice produces the `FreshCheese_beatAble_meltable_mixAble` class in the ontology. This class refers to the fresh cheeses that can be beaten, melted, and mixed.

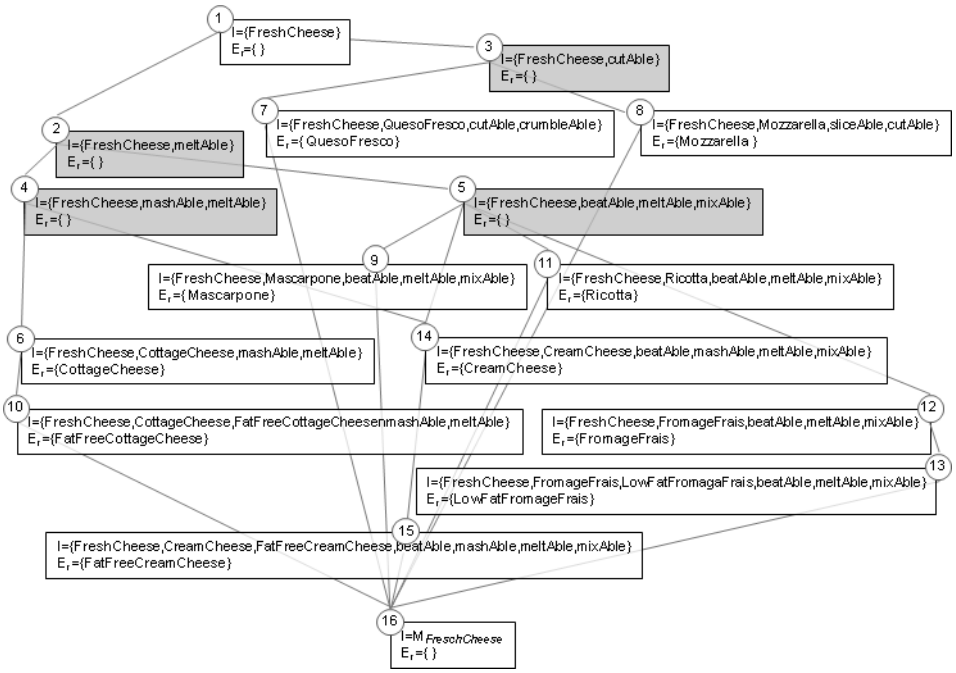

Fig. 2. Concept lattice restructuring the fresh cheese classes.

3.3 Effect on the Taaable system

This section illustrates how the ontology refinement affects the case retrieval and the final answer of the CBR system (see [8] for the details). In order to evaluate these effects, two TAAABLE systems were instantiated: TAAABLE$_3$ using the initial ontology and TAAABLE$_4$ using the refined ontology. The two systems were given the same query to answer. Fig. 3 shows the answer of the two systems to a query asking for a recipe with `tomato` and `mascarpone`. Since no recipe contains both those ingredients in the case base, the two systems search for similar cases.

For TAAABLE$_3$, the retrieval stops after the generalisation `Mascarpone` $\rightsquigarrow$ `FreshCheese`. At this step, 8 recipes are found and adapted by substituting `Mascarpone` for `FreshCheese`. Some of the recipes contain several `FreshCheese`s. In this case, the system proposes to substitute one or more ingredients (this is the case for the recipes numbered 1, 3, and 5). For TAAABLE$_4$, the retrieval is more fine-grained: the generalisation stops on the `FreshCheese_beatAble_meltable_mixAble` class, which is more specific than `FreshCheese`. This class represents the fresh cheeses that can be beaten, melted, and mixed, actions that are applicable to mascarpone.

TAAABLE$_4$ returns only 4 answers, while TAAABLE$_3$ returns 8 answers (including the 4 answers of TAAABLE$_4$ with different substitutions). The generalisation is less sharp in the case of the refined ontology than with the initial one.

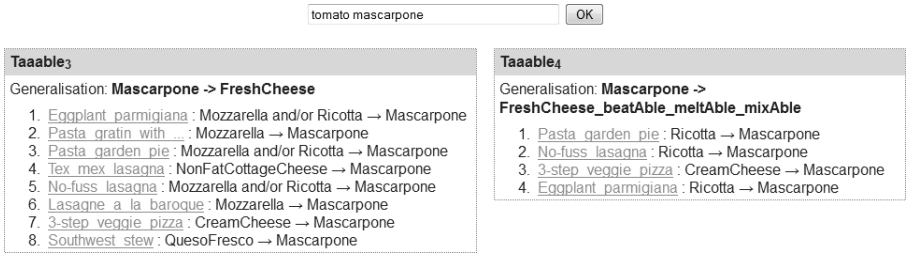

Fig. 3. TAAABLE$_3$ and TAAABLE$_4$'s answers to $Q = \texttt{Tomato} \wedge \texttt{Mascarpone}$.

The substitutions proposed by TAAABLE$_3$ are more various than the ones proposed by TAAABLE$_4$. With TAAABLE$_4$, mascarpone is substituted for ricotta or cream cheese, while TAAABLE$_3$ additionally proposes to substitute mascarpone for mozzarella, non-fat cottage cheese or queso fresco. And the larger the set of substituting ingredients, the more likely the adaptation is to fail, because there is less similarity between the ingredients.

Examining the proposed substitutions, recipe by recipe, allows to evaluate the adaptation. In the following, the adaptation is considered as a failure if the action applied to the substituted ingredient –in our case, the mascarpone– cannot be applied to the substituting ingredient. For the query Q, TAAABLE$_3$ gives 3 clear cases of failure in which the mascarpone should be grated, sliced, or crumbled. There are also 3 *half*-failures for answers in which TAAABLE$_3$ proposes to choose the ingredient that has to be substituted. So, adapting the *Pasta garden* recipe may be considered as a success if the mascarpone replaces the ricotta, but as a failure if the mascarpone replaces the mozzarella and it must be shredded. It is the same for the *No-fuss lasagna* and *Eggplant parmigiana* recipes, where substituting ricotta succeeds while substituting mozzarella fails. To conclude, 6 of the 11 substitutions fail with TAAABLE$_3$ on this example (which is an extreme example case voluntarily chosen for this illustration).

For TAAABLE$_4$, the refined ontology improves the final result as none of the failed adaptations proposed by TAAABLE$_3$ are proposed by TAAABLE$_4$: there is no case of either clear or half-failure. However, one recipe of TAAABLE$_3$ (*Tex-mex lasagna*) for which the adaptation succeeds is not proposed by TAAABLE$_4$ because of the increased distance between `NonFatCottageCheese` and `Mascarpone` in TAAABLE$_4$'s ontology. This results from the stop condition of the algorithm of TAAABLE which runs through the generalisation space. Adaptations which are proposed are those with the lowest adaptation cost. This does not mean that another recipe with a higher adaptation cost does not suit well the query.

A more systematic evaluation remains to be done. However, introducing in the ontology the culinary action viewpoint can be justified according to a minimal adaptation effort principle: two ingredients having the same set of potential culinary actions will not require any change in the textual representation of the recipe (except for the ingredient names). This viewpoint is also defended in [10].

4 Adaptation knowledge acquisition

The other main contribution of this year concerns the acquisition of substitution knowledge explicitly for the adaptation challenge. The goal of the adaptation challenge is to adapt a given recipe with specific constraints, for example: "I want to cook the Chocolate Marble Cake, but I do not have sugar." As TAAABLE returns answers composed of ingredient substitutions, the underlying question is: by which ingredient(s) will sugar be replaced? To compute a successful substitution, FCA is used on a binary context built using *Recipe Source* data.

4.1 Building a binary context for adaptation knowledge acquisition

The goal of the knowledge acquisition process is to find which ingredients must be added or removed in a given recipe R to satisfy the additional constraints. For example, adding/removing an ingredient to R may require adding/removing other ingredients.

To learn such an adaptation knowledge, a binary context is defined where objects are ordered pairs of recipes (R, R') and properties capture the variation between R and R', as introduced by [2] to repair failed substitutions. The variation is exclusively based on ingredients. For all ingredients of R and R, the variation can be expressed by:

- $ING^{=}$: the ingredient ING is used in both R and R';
- ING^{+}: ING is not used in R but is used in R';
- ING^{-}: ING is used in R but not in R'.

For example, the variation of ingredients in (R, R'), represented by $\{Flour^{=}$, $Oil^{=}$, $Water^{-}$, $Salt^{-}$, $Sugar^{+}$, Egg^{+}, $Chocolate^{+}\}$ means that R and R' both use flour and oil, and that R', unlike R, does not use water nor salt but does use sugar, egg, and chocolate.

In order to find how to adapt a given recipe R, a set $\{R_i\}_i$ of recipes *close to* R is retrieved from *Recipe Source*. Using FCA on the ordered pairs of recipes (R, R_i) aims at extracting the variation regularities between R and R_i, variations which can be converted into ingredient substitution rules. For example, if apples must be removed from R and, in several pairs, cinnamon is removed too, we may hypothesise that when removing apples in R, cinnamon must also be removed. Comparing to [2], more criteria are used to select the set of recipes R_i which are the closest to R: (1) a recipe from R_i must satisfy the constraints given by the user, i.e. R_i must contain the ingredients the user wants and must not contain the ingredients the user does not want. (2) R_i must be a recipe of the same dish type (e.g. pie, cake, pizza, etc.) as R.[4] (3) R_i must have a certain amount of ingredients in common with R (at least 50% of the ingredients of R), and must not have too many additional ingredients with respect to R, nor miss too many ingredients

[4] *Recipe Source* contains for each recipe its dish types. For the recipe base provided by the CCC, the recipes were annotated manually by their dish types.

from R. The criteria 1 and 3 are also used for ranking the formal concept according to its intent. The intent of a formal concept is a set of ingredients marked by $=$, $+$, or $-$, which can be transformed in adaptation knowledge: the ingredients marked by $=$ must be kept, and those marked by $+$ (resp. $-$) have to be added (resp. removed). For instance, $I = \{Vanilla^{+}, Egg^{=}, Sugar^{=}, LemonJuice^{-}\}$ means that if vanilla is added, lemon juice must be removed (even if $Egg^{=}$ and $Sugar^{=}$ are present in the intent, they do not imply something in the adaptation).

4.2 An example of result

In the binary context, an ingredient marked with $+$ (resp. $-$, $=$) is an ingredient that has to be added (resp. removed, kept). The set of formal concepts must be filtered and ranked in order to extract the possible substitution ingredients. As we look for the formal concepts whose entailed variations are *minimal* with respect to R, we have chosen the following rules: (1) the intent must satisfy the constraints of the user (e.g. must contain $Banana^{+}$ if the user wants to add banana in R), (2) between 2 intents, the one having, in priority order, the more $=$, the less $-$, and the less $+$ is more relevant. If two intents have the same number of $=$, $-$, and $+$, the intent of the formal concept with the largest extent is considered to be the more relevant.

Let $R = ChocolateMarbleCake$ with the constraint: "I want banana in it". The formal concept with the largest extent (3 objects) is $\{Banana^{+}, Egg^{-}, Milk^{-}, BakingPowder^{=}, BakingSoda^{+}, Chocolate^{=}, Flour^{=}, Sugar^{=}\}$. This intent can be interpreted as follows: when adding banana in the $ChocolateMarbleCake$, baking soda must be added as well, and eggs and milk must be removed. This substitution, while it may look surprising, is valid. Indeed, eggs can be replaced by bananas (!), according to many cooking websites.[5] Removing the milk and adding baking soda are, in this case, two outlying modifications.

5 Conclusion

Taaable 4 proposes two new features, both concerning knowledge acquisition using FCA. The first feature addresses the enrichment of the domain ontology, making the case retrieval more progressive and more precise. The second feature addresses explicitly the adaptation challenge. The best way to adapt a recipe on additional constraints is computed by FCA on recipe variations.

Acknowledgements. The participants of the Taaable project wish to thank the reviewers for their work which has impacted the state of the paper and will impact the future work on Taaable.

[5] See for example `http://www.pioneerthinking.com/eggsub.html`.

References

1. F. Badra, R. Bendaoud, R. Bentebitel, P.-A. Champin, J. Cojan, A. Cordier, S. Després, S. Jean-Daubias, J. Lieber, T. Meilender, A. Mille, E. Nauer, A. Napoli, and Y. Toussaint. Taaable: Text Mining, Ontology Engineering, and Hierarchical Classification for Textual Case-Based Cooking. In *ECCBR Workshops, Workshop of the First Computer Cooking Contest*, pages 219–228, 2008.
2. F. Badra, J. Cojan, A. Cordier, J. Lieber, T. Meilender, A. Mille, P. Molli, E. Nauer, A. Napoli, H. Skaf-Molli, and Y. Toussaint. Knowledge Acquisition and Discovery for the Textual Case-Based Cooking system WikiTaaable. In *Workshops of the 8th International Conference on Case-Based Reasoning*, pages 249–258, 2009.
3. R. Bendaoud, A. Napoli, and Y. Toussaint. Formal Concept Analysis: A unified framework for building and refining ontologies. In Aldo Gangemi and Jérôme Euzenat, editors, *16th International Conference on Knowledge Engineering and Knowledge Management - EKAW 2008*, volume 5268, pages 156–171, Acitrezza, Catania Italie, 2008. Springer Berlin / Heidelberg.
4. A. Blansché, J. Cojan, V. Dufour-Lussier, J. Lieber, P. Molli, E. Nauer, H. Skaf-Molli, and Y. Toussaint. TAAABLE 3: Adaptation of ingredient quantities and of textual preparations. In *18h International Conference on Case-Based Reasoning - ICCBR 2010, "Computer Cooking Contest" Workshop Proceedings*, pages 189–198, Alessandria, Italie, 2010.
5. P. Cimiano, A. Hotho, and S. Staab. Learning concept hierarchies from text corpora using formal concept analysis. In *Journal of Artificial Intelligence Research (JAIR'05)*, volume Volume 24, pages 305–339. AAAI Press, 2005.
6. A. Cordier, J. Lieber, P. Molli, E. Nauer, H. Skaf-Molli, and Y. Toussaint. Wikitaaable: A semantic wiki as a blackboard for a textual case-based reasoning system. In *SemWiki 2009 – 4th Semantic Wiki Workshop*, pages 18–32, Heraklion, Greece, May 2009.
7. V. Dufour-Lussier, J. Lieber, E. Nauer, and Y. Toussaint. Text adaptation using formal concept analysis. In *Case-Based Reasoning Research and Development (proceedings of ICCBR-2010)*, volume 6176 of *Lecture Notes in Artificial Intelligence*, pages 96–110. Springer, 2010.
8. V. Dufour-Lussier, J. Lieber, E. Nauer, and Y. Toussaint. Improving case retrieval by enrichment of the domain ontology. In *19th International Conference on Case Based Reasoning (ICCBR 2011)*, Lecture Notes in Artificial Intelligence. Springer, 2011.
9. B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, Berlin, 1999.
10. Y. Shidochi, T. Takahashi, I. Ide, and H. Murase. Finding replacable materials in cooking recipe texts considering characteristic cooking actions. In *Proc. ACM Multimedia 2009. Workshop on Multimedia for Cooking and Eating Ativities (CEA'09)*, pages 9–14, Beijing, China, 2009.
11. G. Stumme and A. Maedche. FCA-MERGE: Bottom-Up Merging of Ontologies. In *17th International Joint Conferences on Artificial Intelligence (IJCAI'01)*, pages 225–234, San Francisco, CA, 2001. Morgan Kaufmann Publishers, Inc.