TAAABLE 3: Adaptation of ingredient quantities and of textual preparations

Alexandre Blansché, Julien Cojan, Valmi Dufour-Lussier, Jean Lieber, Pascal Molli, Emmanuel Nauer, Hala Skaf-Molli, and Yannick Toussaint

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

Abstract. TAAABLE 3 is a textual case-based cooking system which is a contestant in the third computer cooking contest, inheriting most features of its previous versions (TAAABLE and WIKITAAABLE) and adding new features. The reused features are information retrieval and text mining for building a domain ontology and for annotating the ingredient part of the recipes, a semantic wiki in which the whole knowledge base (recipes, domain knowledge, retrieval knowledge, and adaptation knowledge) is encoded, and an inference engine based on minimal generalisations of the query and adaptation by ingredient substitutions. The two new features are related to adaptation. First, ingredient quantities adaptation is handled using conservative adaptation techniques on numerical constraints, permitting to maximally preserve, for instance, the quantity of sugar in the recipe. Second, adaptation is learned from a set of recipes using text mining techniques. Each recipe is analysed and represented as a tree which identifies ingredients, food components and actions that are performed. Formal concept analysis is used on these data in order to find an appropriate way to insert a new ingredient in the final recipe.

Keywords: textual case-based reasoning, case-based cooking, semantic wiki, conservative adaptation, adaptation of ingredient quantities, natural language processing, formal concept analysis, adaptation of recipe preparations

URL of the system: At `http://taaable.fr`, the reader can find a link to the homepage of the TAAABLE 3 system.

1 Introduction

TAAABLE and WIKITAAABLE have participated in the first and second computer cooking contests [1, 2]. TAAABLE 3 is the 2010's version of TAAABLE. It inherits from features of the two previous versions of the system, as described in Section 2. In particular, it reuses the semantic wiki of WIKITAAABLE and improves it (Section 3). It also has two original features. The first one is related to ingredient quantity adaptation using revision-based adaptation in metric spaces. It is presented in Section 4 and is based on previous work [3]. The second one is related to the textual adaptation of recipe preparations. This feature is based on natural language processing and formal concept analysis. It is presented in Section 5 and detailed in [4].

2 Previous work on the TAAABLE project

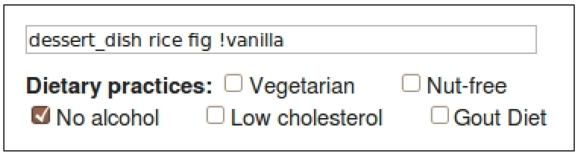
The knowledge-based system TAAABLE is composed of a CBR inference engine, a knowledge base stored in a semantic wiki, and some tools for acquiring or editing knowledge. These tools have a great importance for the different versions of the system, but no major improvements have been made on these tools this year, so they are not detailed here.

The knowledge base stored and edited in a semantic wiki is composed of a cooking domain ontology $\mathcal{O}$, the recipe base indexed by concepts of $\mathcal{O}$ (thanks to an annotation process), and some knowledge useful for retrieval and for adaptation. The ontology $\mathcal{O}$ contains about 5000 concepts organised in several hierarchies, including an ingredient hierarchy and a dish type hierarchy. For example, the concept `mango` is under the concept `tropical_fruit`: every mango is a tropical fruit. Some improvements on the wiki concerning ingredient properties and recipe formalisation are detailed in Section 3.

A query Q to the CBR engine is expressed as a conjunction of literals, where a literal is either a concept of $\mathcal{O}$ or the negation of such a concept. For example, let us consider the following request, expressed in natural language: “I want an alcohol-free dessert with rice and figs, but without vanilla.” This request is expressed by

$$Q = \neg\texttt{alcohol} \wedge \texttt{dessert\_dish} \wedge \texttt{rice} \wedge \texttt{fig} \wedge \neg\texttt{vanilla} \qquad (1)$$

or



in the interface

Since the recipes are indexed by concepts 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 concept `D`, under `C` in $\mathcal{O}$, such that R is indexed by `D`; (2) for each negative literal $\neg$`C` of Q, there exists no concept `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` $\rightarrow$ `B` where `A` is a direct subconcept of `B` in $\mathcal{O}$. The minimality of the generalisation Γ is computed thanks to a cost function, described in [2]. For example, the recipe R named “Glutinous rice with mangoes” is indexed by the conjunction `dessert_dish` $\wedge$ `coconut_cream` $\wedge$ `mango` $\wedge$ `salt` $\wedge$ `sugar` $\wedge$ `rice` $\wedge$ `sesame_seed`.The retrieval process retrieves this recipe for the query (1) with generalisation $\Gamma = \texttt{fig} \rightarrow \texttt{tropical\_fruit}$.

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 concepts 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 “Glutinous rice with mangoes” will consist in (a) substituting `mango` with `tropical_fruit` and (b) substituting `tropical_fruit` with `fig`, which is equivalent to the substitution of mangoes with figs (Figure 1). 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., $\Gamma = a \wedge b \to c$) 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]), though TAAABLE 3 does not use such rules.

Your request is: !vanilla_extract -not_alcoholic_free dessert_dish fig rice

The request used for adaptation is: !vanilla_extract -not_alcoholic_free dessert_dish fig rice

#	Original recipe name (click to open recipe)	Adaptation overview (click to see the details)	Cost
1	Glutinous_rice_with_mangoes	Replace: Mango by Fig	10
Results 1 - 1 on 1 | Processing time: 2.239 seconds			

Fig. 1. Retrieved recipe with the proposed substitution for the query example.

3 New content of the TAAABLE's Semantic Wiki

3.1 Ingredients

In the previous versions of TAAABLE, ingredients were represented by wiki categories. Each ingredient is encoded as a wiki category in which the ingredient is only described according to its relation with more generic categories. For example, the category `Mango` is described as a sub-category of the category `tropical_fruit`.

In the version of the TAAABLE 3 wiki, ingredient categories have been enriched by additional knowledge extracted from the USDA Nutrient database (`http://www.nal.usda.gov/`). An ingredient is now described by its nutritional values (sugar, fat, protein, vitamins, etc.) and some weight conversion equivalences, two types of knowledge required for the adaptation of ingredient quantities (cf. Section 4). Ingredient nutritional values are represented using semantic wiki properties and weight conversions thanks to semantic wiki multi-value properties. Multi-value properties are needed because a weight conversion is based five element: an amount, a unit, a property, an ingredient, and the equivalent weight in grams, (e.g. one cup of sliced mango is 165g). For linking nutritional values and weight conversions to their corresponding wiki categories, a semi-automatic alignment has been processed. This alignment makes the link between the ingredient label in the USDA Nutrient database and its label in the wiki as a category. As the mapping is incomplete, some of the ingredients in the wiki do not possess such information.

Figure 2 shows the content of the category `Mango`. One can see the knowledge about weight conversion and nutritional values organised into tables. Users might also add a picture of the ingredient.

3.2 Recipes

The description of recipes has also been improved. A natural language analysis process, described in Section 5, is used to add a formal description of the preparation of a recipe.

Category:Mango

Weight conversions

1 c, sliced	165 g
1 unit	207 g

Nutritional values

Nutritional value per 100 g (3.5 oz)	
Energy	65 kcal (270 kJ)
Carbohydrates	17 g
Sugars	14.8 g
Dietary fiber	1.8 g
Fat	0.27 g
Protein	0.51 g
Water	81.71 g
Vitamin A (equiv.)	38 μg (4%)
Thiamine (Vit. B1)	0.058 mg (4%)
Riboflavin (Vit. B2)	0.057 mg (4%)
Niacin (Vit. B3)	0.584 mg (4%)
Pantothenic acid (Vit. B5)	0.16 mg (3%)
Vitamin B6	0.134 mg (10%)
Folate (Vit. B9)	14 μg (4%)
Vitamin C	27.7 mg (46%)
Calcium	10 mg (1%)
Iron	0.13 mg (1%)
Magnesium	9 mg (2%)
Phosphorus	11 mg (2%)
Potassium	156 mg (3%)
Sodium	2 mg (0%)
Zinc	0.04 mg (0%)
Percentages are relative to US recommendations for adults. Source: USDA Nutrient database	

Mango

This category currently contains no pages or media.

Category: Tropical fruit

Fig. 2. Mango category.

All the steps of the preparation are inserted in the wiki as a list of items. A graphical representation of the preparation is also included.

Figure 3 shows the final part of the formal preparation of "Glutinous rice with mangoes" as a graph. This graph is included in the wiki page of the corresponding recipe.

4 Adaptation of ingredient quantities with belief revision

4.1 Integration into TAAABLE

The use of TAAABLE remains as presented in Section 2, in particular the queries must be given in the same format. The retrieval and a first adaptation step that computes the set of substitutions ignore ingredient quantities. With the example query given in Section 2 the retrieved recipe is "Glutinous rice with mangoes", and the first adaptation step recommends to substitute mangoes with figs.

The adaptation on ingredient quantities is triggered when calling the adaptation result page, it takes as inputs:

- The source recipe with its list of ingredients with their amounts and units.
- A set of ingredient substitutions of the form $\texttt{from\_ing}_i \rightarrow \texttt{with\_ing}_i$.
- Numerical data about the ingredients: unit conversions (e.g. the weight of a cup of mango) and nutritional data (e.g. how much energy in calories a gram of mango contains).

The result is shown in Figure 4.

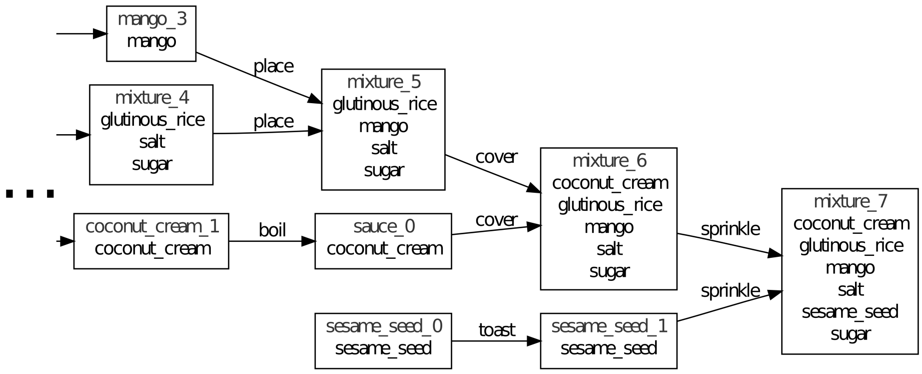


Fig. 3. Excerpt of the formal description of the preparation for "Glutinous rice with mangoes".

Ingredient	Initial Quantity	New Quantity
Coconut cream	2.7 cup (648.0 grams)	2.7 cup (648.0 grams)
Glutinous rice	3.0 cup (522.0 grams)	3.0 cup (522.0 grams)
Granulated sugar	2.5 cup (500.0 grams)	2.4 cup (480.0 grams)
Mango	6.0 whole (1242.0 grams)	0.0 whole (0.0 grams)
Salt	1.2 tsp (7.2 grams)	1.2 tsp (7.2 grams)
Sesame seed	2.0 tbsp (16.0 grams)	0.1 tbsp (0.8 grams)
Fig	0	1242.0 grams

Fig. 4. Ingredient quantity adaptation for the example. The evolution of ingredient quantities are under the influence of several criteria, in particular the preservation of nutritional components, see Section 4.2. This explains, for instance, the reduction of the sugar quantity: figs are slightly sweeter than mangoes. The reduction of the sesame seed quantity can also be explained by the preservation of nutritional components (fibres, calcium, magnesium, . . .), though this justification can be discussed (see the distance definition in Section 4.2).

4.2 How does it work

This adaptation step follows the $\dot{+}$-adaptation approach [3] applied to an attribute-value formalism. The cases are formalised as constraint satisfaction problems over the attributes; solving a case amounts to specifying the value of all its attributes.

Representation space. Only numerical attributes are considered:

– The ingredient quantities in grams and in another unit if they were given in this other unit. The ingredient values will be stored in variables $[\texttt{ingredient}]_{[\texttt{unit}]}$, for instance as mangoes are given in cups, two variables will be used for mangoes: $\texttt{mango}_{\text{cups}}$ and $\texttt{mango}_{\text{grams}}$. In addition to the ingredients expressly given in the recipe source and in the substitutions, their generalisations are considered as well. For instance, as `fig` is classified as a `tropical_fruit` and `tropical_fruit` is a subclass of `fruit` there will be a dimension for each in the representation space.

– All the nutritional components that appear in the wiki (see fig. 2) represented by the variables $\texttt{lipid}_{\text{grams}}$, $\texttt{energy}_{\text{Kcal}}$, etc.

Domain Knowledge. The domain knowledge is expressed by a set of linear constraints:

– Unit conversion, for instance: $\texttt{mango}_{\text{grams}} = 165\ \texttt{mango}_{\text{cups}}$ (a cup of mango contains 165 grams of mango).
– Ingredient hierarchy, for instance: $\texttt{tropical\_fruit}_{\text{grams}} = \texttt{fig}_{\text{grams}} + \texttt{mango}_{\text{grams}}$ (as in this adaptation session no other tropical fruit are considered).
– Nutritional data calculus, for instance: $\texttt{energy}_{\text{Kcal}} = 0.65\ \texttt{mango}_{\text{grams}} + 0.74\ \texttt{fig}_{\text{gram}} + 0.97\ \texttt{glutinous\_rice}_{\text{grams}} + \dots$

Source case. The solution of a case (i.e. its ingredients quantities) is expressed as well as a set of linear constraints. For instance, in the source recipe "Glutinous rice with mangoes": $\texttt{rice}_{\text{cups}} = 3$.

Target Case. For each substitution $\texttt{from\_ing} \rightarrow \texttt{with\_ing}$, a constraint $\texttt{from\_ing} = 0$ is added to the target case. This transcription of the substitutions is based on the fact that the substitution has been obtained by a generalisation-specialisation adaptation path (see Section 2): $\texttt{from\_ing} \xrightarrow{\text{generalisation}} \texttt{parent} \xrightarrow{\text{specialisation}} \texttt{with\_ing}$. The domain knowledge contains then a constraint: $\texttt{parent}_{\text{grams}} = \texttt{from\_ing}_{\text{grams}} + \texttt{with\_ing}_{\text{grams}}$. Stating that $\texttt{from\_ing}_{\text{grams}} = 0$ in the target case while preserving $\texttt{parent}_{\text{grams}}$ will entail the substitution from from_ing with with_ing.

As the target case is also expressed as a set of linear constraints over ingredient quantities, the request could contain constraints of the form $\texttt{fig}_{\text{gram}} \leq 500$ to state that only 500 grams of figs are available. However this is not yet implemented in the interface.

Distance. The adaptation cost is evaluated according to a distance dist between the source case and the proposed solution for target. This distance is taken as a weighted sum of each value change (i.e. $\texttt{dist}(x, y) = \sum_i w_i |y_i - x_i|$ where i stands for one of the dimensions).

Not only ingredient quantities are taken into account in dist (see the representation space definition), in particular, terms corresponding to nutritional components are included as well. This entails some quantities adjustments that are not explicitly related to the substitution, see the sugar and sesame seeds quantity evolution in Figure 4.

The weights are w_i are set to 1 by default, this choice leads to questionable value changes; (see sesame seeds quantity in Figure 4); a more relevant valuation should be considered in future work.

Computation. $\dot{+}$-adaptation relates the CBR principle of minimal change for adaptation to the theory of belief revision [5]. A revision operator $\dot{+}$ encodes some meaning to the expression "minimal change" in the context of making two formulas consistent.

Here, the ingredient quantities given by the source recipe are revised by the target case constraints. $\dotplus$ is defined from `dist`: the adaptation process returns the quantities that are closest to the source recipe quantities according to the `dist` while remaining consistent with target constraints. The computation follows the reduction of $\dotplus$-adaptation computation to linear programming proposed in [3].

5 Adaptation of the preparation: natural language processing and formal concept analysis

At this stage we have a satisfactory glutinous fig rice recipe as far as picking the right ingredients is concerned. But if one simply substitutes "fig" for "mango" wherever it appears in the preparation text, as a user of the system might be led to believe we suggest, the apprentice cook will be facing an insurmountable problem. The recipe text demands that the mangoes be peeled, sliced lengthwise and pitted, all operations ranging somewhere between awkward and utterly impossible to accomplish when it comes to figs.

We introduce a method using natural language processing techniques and formal concept analysis to benefit from information hidden in the preparation text and improve the adaptation stage. The guiding idea behind this work is that, for each ingredient, there exist preparation "prototypes" describing the different ways in which the ingredient can be used. What we then need to adapt our mango rice recipe into a fig rice recipe is a mango prototype, a variety of fig prototypes, and some similarity metric to pick the fig prototype that is closest to our mango prototype. The algorithms that perform those tasks are defined in more details in [4].

5.1 From Natural Language to Formal Representation

Acquiring meaningful case representations from textual sources remain one of the foremost challenges of textual case-based reasoning[6]. Because a recipe basically consists of taking a set of ingredients and combining them in specific ways until we obtain something edible, it seems natural to represent them as trees as shown in Fig. 3. Each node represents the state of a food component at a given time, and each edge $\langle \alpha, \beta \rangle$ represents an action applied on food component α that yields a new food component β. A set of edges sharing the same head (β) represents a "combining action", i.e., an action applied to many food components at once that yield out one new food component.

This representation is used even as it is being built to resolve some of the difficult situations that occur frequently in recipe texts. In a sentence such as "Peel the mangoes, slice [the mangoes] lengthwise and remove the pits [from the mangoes]", it is easy as humans to understand that mangoes are being sliced and pitted, but some heuristic is needed is order for a computer to realise that. We have attributed an arity to each action in order to be able to tell when an argument is missing. Whenever this happens, it is taken that the missing argument is in fact the last node that was added to the tree. In that way, we are able to deal with anaphora, the phenomenon wherein a different word, or no word at all as it might be, is used to represent an object.

Other types of anaphora appear in our mango rice recipe. The expression "seasonings ingredients" clearly refer to some set of food components, so we use the ingredient hierarchy to find all the nodes of the tree that fit under the "seasonings" category. A phrase such as "cover with sauce" is a lot trickier because there is no obvious clue either in the text or in the ontology which food component this "sauce" may be. We built, from the analysis of thousands of recipes, "target sets" of ingredients that usually appear in the food components being referred to by word such as "sauce" or, say, "batter". Because a quantity of recipes include coconut cream-based sauces (and few contain, say, rice-based sauces) we are able to guess rightly that the food component containing coconut cream really is the one the recipe author had in mind when he wrote "sauce".

Once the tree representation is built, it is trivial to identify the sequence of actions being applied to any ingredient of a given recipe. We define the prototype of an ingredient by the actions applied to this ingredient, up to and including the first combining action. In practice this means we only use the leftmost part of the tree. This has the limitation that we may be missing relevant actions, but we are also getting rid of actions that, while being applicable to certain mixtures containing an ingredient, may not make sense when applied directly to this ingredient. In the mango rice, we are keeping the *chill, peel, slice lengthwise, remove pits,* and *place on top* actions, and getting rid of *cover with, sprinkle with,* and *serve* which are indeed too generic to be interesting. As an added benefit, the leftmost part of the tree is the one we are able to build with the higher reliability at this time.

5.2 Recipe Adaptation Using Formal Concept Analysis

The mango prototype used in the mango rice is, then, characterised by the set of actions { `chill`, `peel`, `slice`, `remove-pit`, `place` }. In the recipe base, there are two recipes with figs. Recipe nr. 1163 sports a { `halve`, `sprinkle-over`, `dot-with`, `cook`, `brown`, `place` } fig prototype, and nr. 53 a more modest { `cut`, `combine` }. In order to simplify the processing and avoid discriminating near-synonyms, we can group classes of actions using the action hiearchy. This has the effect of grouping `peel` with `remove-pit`, `cut` with `slice`, and `cook` with `brown`.

To select the more appropriate fig prototype, a concept lattice from the fig prototypes is built with the mango prototype merged in. This is similar to techniques used in document retrieval by [7] wherein a lattice is built according to keywords found in documents and a query is merged in. The formal context contains the set of fig prototypes as its set of objects and its set of attributes is the set of action classes applicable to figs. It is concatenated (see [8]) with a "query" context consisting of the mango prototype from the retrieved recipe as its only object and the set of action classes applied to this mango as its attributes, yielding the formal context shown in Table 1.

Using formal concept analysis [9], the concept lattice shown in Figure 5 is built from Table 1. All prototypes having actions in common will appear together in the extension of at least one concept. The "lower" this concept is in the lattice, the more attributes the prototypes have in common. A set of candidate fig prototypes is taken from the extension of all concepts "immediately above" the lowest concept with `mango` in its extension. Whenever there is more than one candidate, the one that minimises the distance between its attributes and the mango's attributes is selected. In the example, `fig_53`

	cut	halve	remove	place	pour	dot	cook	chill	combine
fig_1163		×		×	×	×	×		
fig_53	×								×
mango	×		×	×				×	

Table 1. Formal context of figs and a mango.

is selected since, while both fig_53 and fig_1163 appear directly above mango, replacing mango with fig_1163 would require removing three actions and adding four, whereas replacing it with fig_53 would require only removing three actions.

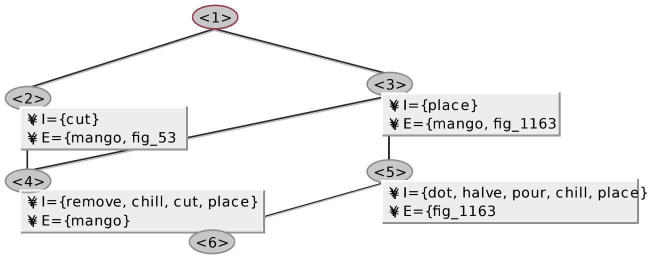


Fig. 5. Concept lattice corresponding to the context of Table 1. Each node (formal concept) is a set of objects O (extension, noted "E") and a set of attributes A (intension, noted "I") such that all objects in O have all attributes in A in common in exclusion to any other. They are ordered according to the inclusion of their extension.

The process is then finalised by replacing the textual parts of the retrieved recipe dealing with mangoes with the parts of recipe nr. 53 dealing with figs:

> [...] Blend the sauce ingredients in a pot and heat until it just reaches the boiling point. Let cool. ~~Peel the mangoes, slice lengthwise and remove the pits.~~ Cut figs into wedges. Divide the rice mixture among 6 plates. [...]

6 Conclusion

TAAABLE 3 is a textual case-based cooking system which is a contestant in the third computer cooking contest, inheriting most features of its previous versions (TAAABLE and WIKITAAABLE) and adding new features. The reused features are information retrieval and text mining for building a domain ontology and for annotating the ingredient part of the recipes, a semantic wiki in which the whole knowledge base (recipes, domain knowledge, retrieval knowledge, and adaptation knowledge) is encoded, and an inference engine based on minimal generalisations of the query and adaptation by ingredient substitutions. The two new features are related to adaptation. First, ingredient quantities adaptation is handled using conservative adaptation techniques on numerical constraints, permitting to maximally preserve, for instance, the quantity of sugar in the recipe. Second, adaptation is learned from a set of recipes using text mining techniques. Each recipe is analysed and represented as a tree which identifies ingredients, food components and actions that are performed. Formal concept analysis is used on these data in order to find an appropriate way to insert a new ingredient in the final recipe.

The evaluations of the whole system and of its components constitute an important ongoing work. Since there is no known computing test asserting whether a recipe is cookable (i.e., giving a satisfactory dish, from a human average taste viewpoint), these evaluations are subject studies.

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. They also wish to thank the reviewers for their work which has impacted the state of the paper and will impact the future work on TAAABLE.

References

1. Badra, F., Bendaoud, R., Bentebitel, R., Champin, P.A., Cojan, J., Cordier, A., Després, S., Jean-Daubias, S., Lieber, J., Meilender, T., Mille, A., Nauer, E., Napoli, A., Toussaint, Y.: Taaable: Text Mining, Ontology Engineering, and Hierarchical Classification for Textual Case-Based Cooking. In: ECCBR Workshops, Workshop of the First Computer Cooking Contest. (2008) 219–228
2. Badra, F., Cojan, J., Cordier, A., Lieber, J., Meilender, T., Mille, A., Molli, P., Nauer, E., Napoli, A., Skaf-Molli, H., Toussaint, Y.: Knowledge Acquisition and Discovery for the Textual Case-Based Cooking system WikiTaaable. In: Workshops of the 8th International Conference on Case-Based Reasoning (ICCBR-09). (2009)
3. Cojan, J., Lieber, J.: Belief Merging-based Case Combination. In David C. Wilson, Lorraine McGinty, eds.: 8th International Conference on Case-Based Reasoning - ICCBR 2009 Case-Based Reasoning Research and Development. Volume 5650 of Lecture Notes in Computer Science., Seattle United States, Springer Berlin (07 2009) 105–119 The original publication is available at www.springerlink.com.
4. Dufour-Lussier, V., Lieber, J., Nauer, E., Toussaint, Y.: Text adaptation using formal concept analysis. In Bichindaritz, I., Montani, S., eds.: Case-Based Reasoning Research and Development. Volume 6176 of Lecture Notes in Artificial Intelligence., Alessandria, Italy, Springer (2010) 96–110
5. Alchourrón, C.E., Gärdenfors, P., Makinson, D.: On the logic of theory change: partial meet functions for contraction and revision. Journal of Symbolic Logic **50** (1985) 510–530
6. Asiimwe, S., Craw, S., Wiratunga, N., Taylor, B.: Automatically acquiring structured case representations: The SMART way. In: Applications and Innovations in Intelligent Systems XV, Springer (2007)
7. Carpineto, C., Romamo, G.: Order-Theoretical Ranking. Journal of the American Society for Information Science **51**(7) (2000)
8. Bendaoud, R., Napoli, A., Toussaint, Y.: Formal Concept Analysis: A unified framework for building and refining ontologies. In Aldo Gangemi, Jérôme Euzenat, eds.: 16th International Conference on Knowledge Engineering and Knowledge Management - EKAW 2008. Volume 5268., Acitrezza, Catania Italie, Springer Berlin / Heidelberg (2008) 156–171
9. Ganter, B., Wille, R.: Formal Concept Analysis. Springer, Heidelberg (1999)