CookIIS – A Case-Based Recipe Advisor

Norman Ihle, Régis Newo, Alexandre Hanft, Kerstin Bach, Meike Reichle

Intelligent Information Systems Lab
University of Hildesheim
Marienburger Platz 22
31141 Hildesheim, Germany
{lastname}@iis.uni-hildesheim.de

Abstract. The aim of this paper is to present CookIIS[1], a Case-Based Reasoning (CBR) system that provides recipe suggestions. The suggestions are created based on a given set of recipes that the system modifies according to a user's specification. For the adaptation of recipes CookIIS relies on its knowledge model, a set of 342 rules as well as substitution suggestions mined from cooking communities on the WWW. The paper describes how CookIIS processes the competition queries and masters the challenges of the Computer Cooking Contest 2009.

1 Introduction

Retrieving and reusing recipes that are correct and delicious is a difficult task and the participation in last years Computer Cooking Contest (CCC) showed us that the challenges we had to bear are interesting research topics. In this paper we present insights how our system CookIIS is realized and which methodologies and techniques we use to cover the expected areas of competences.

This year the CCC contains three main challenges that we all address with CookIIS. The Compulsory Task focusses on the type of meal and type of cuisine as well as extended dietary practises and the modification of recipes. Therefore we extended and improved last years knowledge models, added new dietary practices and we are now using additional knowledge that has been extracted from cooking communities on the WWW. For participation in the new Adaptation Challenge, CookIIS has an own adaptation component which only contains pasta recipes with more detailed and structured information of the preparation processes. The third task, the Menu Challenge, focusses, like last year, on the composition of a three course menu and we use the findings and further developments made for the Compulsory Task and Adaptation Challenge to also improve the retrieval and adaptation in the Menu Challenge.

Overall CookIIS is a Case-Based Reasoning (CBR), more precisely a structured CBR system that uses techniques like Text Mining and Information Extraction to gather more knowledge and improve the results of our systems. Moreover, we used the experiences we made with last years system to improve

[1] http://cookiis2009.iis.uni-hildesheim.de:8080/ccc

CookIIS, especially its underlying knowledge containers [1]. We realized CookIIS using the industrial strength CBR based tool Information Access Suite (e:IAS) from empolis[2], which employs Case Retrieval Nets [3]. The cases that we use were given by the CCC organizers, so we concentrated on extracting knowledge for recipe transformation, the coverage of our vocabulary and the computation of the similarity measures. Based on last years systems we completely revised the underlying knowledge model: we defined more precise ingredient taxonomies and (semi-)automatically extended the vocabulary aiming that most of the ingredients can be recognized. The more detailed representation of ingredients also results in a better similarity computation.

We decided to use very user-friendly text based graphical interface what makes it easy to use CookIIS. According to the three challenges we have got three components: The *CookIIS Recipe Creator* is developed to compete in the Compulsory Task because its underlying case base contains the according recipes. The *CookIIS Pasta Adaptation* only contains the pasta recipes and focusses in the adaptation task. The *CookIIS Menu Creator* also contains compulsory case base, but has text fields to specify and explicitly exclude ingredients for each of the three courses. CookIIS can be queried in German and in English - however the recipes and the preparation instructions will be in English.

Our paper is structured as follows: Section 2 presents the CookIIS knowledge models and focuses on improvements compared to last year's system. According to the three tasks CookIIS has to perform, section 3 contains the Compulsory Task queries and explains how we extract type of meal and type of cuisine, how we retrieve recipes that fit dietary practises and how we compute the similarity measures. Section 4 presents the adaptation process as well as the integration of community knowledge to provide more reliable substitutions for ingredients. The Menu Challenge queries as well as the creation process of a menu is described in 5. The last section gives a summary of the main features of CookIIS and provides a short outlook on future work.

2 Knowledge Model

Since we use an approach that is based on structured CBR, we implemented a very detailed knowledge model which describes the cooking domain. The model is based on the one of the CCCIIS system that competed in last years Computer Cooking Contest [4]. The main elements of our model are the ingredients that are required to prepare a meal. They are organized in the following eleven classes:

Basic Ingredients	Fish	Meat	Vegetable
Supplement	Fruit	Drinks	Milk
Minor Ingredients	Oil and Fat	Spice and Herb	

Within these classes we modelled about 1000 different ingredients represented as concepts that can be used for describing a meal. The most concepts (177) are contained in the Meat class. Each concept represents one single ingredient and is modeled with synonyms in English and German language. In most cases the

concepts are ordered in one or more taxonomies. These taxonomies are one possibility for the calculation of the similarity between a query and a case as explained in Section 3.4.

For this years contest we manually revised our existing model from last year, cleaned a lot duplicates, added a lot of concepts and synonyms and reworked all of our taxonomies. The remodeling and completion of classes is mainly based on [5]. For example, we split the class 'liquids', which contained drinks as well as non-drinkable fluids into the classes 'Drinks' and 'Oil and Fat' since those two classes have almost nothing in common. Any fluids not fitting in those classes we added to other existing and fitting ones. For example water is now a basic ingredient and only mineral water is a drink. Since there is a special focus on pasta recipes for this years adaptation challenge we also modeled about 40 kinds of different pasta in an additional class.

Besides the ingredients, preparation methods and tools required for the preparation are often mentioned in a recipe. We modeled each in an own class. The preparation methods are not only used to compute the similarity between a query and a case, but it is also used to determine the type of meal (e.g. dessert) as described in the following section. For the Computer Cooking Contest it also important to determine the origin of a recipe. We modeled 48 possible origins and ordered them in a taxonomy to use them for similarity calculation too.

Overall we modelled about 2000 different concepts in 25 different classes. We weighted each class in relation to the importance of the class for finding the right recipe. We built two different case bases using two different aggregates, but the same model: one with the compulsory task recipes and one with the pasta recipes.

3 Compulsory Task

In the Compulsory Task, single recipes have to be retrieved according to automatically computed meta information like the type of ingredients, the type of meal, the type of cuisine while also considering some dietary practices.
The type of ingredients is automatically recognized based on the classes in our knowledge model presented in the previous section. We will explain in the following two sections how we computed other meta information.

3.1 Type of Meal

One type of meta information needed for the recipes is the type of meal, which describes whether a dish is for example a main course or a dessert. We defined a class in which most concepts (i.e. types of meal) are ordered in a taxonomy. We implemented some in order to assign the types of meals to the recipes. We based the implementation on two main aspects:

- indicative keywords in the recipe title,
- indicative ingredients or combination of ingredients.

In the first approach, we extract the type of meal from the title of a recipe. For example, a recipe which contains the indicative word "sherbet" in its title is, with a high probably, a dessert. In the second approach, we analyze the ingredients (and also their types) as well as the preparation methods contained in recipes. The preparation methods are extracted from the given preparation instructions. For example, a frozen meal with milk, fruits and sugar is probably an ice cream. We do not use further information from the preparation instructions yet. We have about 15 rules for the assignment of the type of meal with a precision of about 90 percent.

3.2 Type of cuisine

Another requirement of the CCC is the identification of the type of cuisine (e.g. Italian, Chinese, Mediterranean, etc.), which is not given in the recipe itself. All possible origins are modeled in one class and organized in an taxonomy. For example the concept "Mediterranean" is the parent for "Portuguese", "Spanish" and "Italian", while "Mediterranean" itself is a child node of "Southern European". In order to map the recipes to one of our concepts we implemented three rule-based approaches using rules:

1. identification of the recipe's origin in the recipe title
2. identification of characteristic strings or meals in the recipe title and map them to an origin
3. using the occurrences of spices and herbs and other ingredients to find some elements that are characteristic for a type of cuisine

The first method is based on the finding that many recipes display their origin in their title (e.g. Chinese Chicken Salad). Here the origin is determined right away. The second approach needs some background knowledge. It is based on the fact that some foods or meals are characteristic for a specific type of cuisine (e.g Sauerkraut is typical for the German kitchen). A set of rules maps those occurrences to the according type of cuisine. For the third approach we use ingredients, especially the spices and herbs used in the recipe. For example the use of curry is a strong hint on the Indian cuisine. Overall CookIIS includes about 28 different rules, which are prioritized according to the different approaches and map the recipes with a precision of 80 to 90 percent.

3.3 Dietary Practices

In the CCC tasks and challenges, the participating systems have to be able to retrieve suggestions that are following at least the following two dietary practices (i.e. gout diet and cholesterol diet) and a seasonal food calendar. We also include other dietary practices (i.e. vegetarian, nut free and non alcoholic) which were implemented for our CCCIIS [4] in last year's contest.
To follow the specification of a gout and cholesterol diet, ingredients as well as types of ingredients were given, which should or should not be contained in

the retrieved recipes. For each forbidden ingredient, we set a filter so recipes containing this ingredient are not considered during the retrieval process. We do this by excluding either single ingredients or categories of ingredients (e.g. exclude any kind of meat for a gout diet). We also set filters in order to consider only the recipes that contain at least one of the recommended ingredients. These two kinds of filter ensure that the retrieved recipes always contain some of the recommended ingredients and none of the forbidden ones. We use filters instead of adaptation, because the forbidden ingredients can not be adequately replaced in a recipe while following a diet.
With the seasonal option, the user can specify a month of which the vegetables that are included in the suggested recipe are available. If this option is chosen and some vegetables in the best fitting recipe are not available in the selected month, the system will give a hint. Further we propose, according to the provided seasonal food calendar, adaptation candidates for some unavailable vegetables. The system just proposes candidates that are available in the selected month.

3.4 Similarities (Taxonomies)

As stated before, each recipe is represented as one case consisting of attributes. Each attribute's class is defined by the concepts that we introduced in Section 2. The similarity between two cases is calculated as follows:

- local similarity for each attribute and
- a global similarity measure for recipes.

The taxonomies of our knowledge model are used as a first approach for the computation of the local similarities. By assigning adequate values for the generalization and the specialization step in the taxonomy, similarities between concepts of a class can be computed. In our taxonomies, those values were assigned and adjusted manually. When needed, we defined several taxonomies for given classes. For example, the class meat has two taxonomies. In the first taxonomy the single meat ingredients are ordered according to their kind (e.g. beef, pork, poultry) and in the second taxonomy the ingredients are ordered by the part of the animal they origin from (e.g. fillet, haunch). Since both taxonomies are used for similarity calculation, chicken breast is similar to turkey breast as well as to chicken fillet.
The second approach for the computation of local similarities consists of manually assigning similarity values of some pairs of elements of the class in a similarity matrix. This assignment is only done for pairs of ingredients for which either they are pretty well known to be very similar or one of the ingredients could not be ordered in a taxonomy. We use a combined similarity measure in which the assigned similarity value is the maximal value obtained from both approaches.
The global similarity measure is a weighted sum of the local similarities of the attributes in the case, following the local-global principle for similarity modeling (see [6]). The weights assigned to the local similarities reflects the importance of the corresponding attributes in the cases. The sort of meat used in a meal is, for example, more important than the type of oil used.

4 Adaptation Challenge

Due to the fact that the given recipe base is not large enough to provide recipes for the whole variety of desired and favoured ingredients, an adaptation of the existing recipes to the users needs is necessary. Thereby ingredients which are excluded by a certain diet or explicitly by a user are considered as forbidden ingredients. The intersection of forbidden ingredients and ingredients occurring in a retrieved recipe we call critical and they have to be replaced. Our overall assumption is to replace forbidden ingredients with others from the same ingredient class.

The adaptation of the recipes in CookIIS is mainly done via a set of rules called after the retrieval of similar cases. According to shortcomings of our first adaptation approach used at CCC 2008 we discovered in [7] we restructured our adaptation as a sequence of distinguished steps to pursue different adaptation approaches. At the moment the implemented adaptation steps can be subsumed under two approaches: *community-based adaptation* and *model-based adaptation*. The first executed approach collects concrete pairs of ingredients as adaptation advices from comments inside cooking communities on the WWW and is described in section 4.2. For critical ingredients where no replacements at the first step are found the subsequent adaptation steps are executed (belonging to *model-based adaptation*), which work more general on the designed knowledge model and similarity measures. The reason for using this approach is to avoid handcrafting a single adaptation rule for each ingredient with explicit replacement candidates. It it explained in the next section 4.1. We combine both adaptation approaches and save intermediate results in extra attributes of the case. The main advantage of this sequential adaptation is that we can review and adjust the results of adaptation steps performed before.

4.1 Model-based Adaptation

The adaptation schema bases on a relaxed intersection function from e:IAS [2] Rule Engine which determines similar ingredients for the critical ingredients as replacements. It is explained more in detail in [8].

The replacement of ingredient (concepts) through child concepts of this ingredients is not appropriate (e.g. replacing forbidden tomatoes with cherry tomatoes). Therefore we have to remove the child concepts from the list of replacement candidates. The relaxed intersection function only uses the default similarity measure of a class which is, for CookIIS, the combined similarity measure explained in Section 3.4. It is not possible to explicitly choose one the given approaches used for the computation of similarity values. The relaxed intersection function therefore allows the replacement of forbidden ingredients through parent concepts as well as child concepts, which is not appropriate under certain circumstances. In order to eliminate the child concepts of the forbidden ingredients from the list of replacement candidates, we adjusted some values in the similarity matrices, such that other candidates are more similar than the child concepts. We execute the intersection function twice, once with a threshold equal

to similarity value resulting from a taxonomy and afterwards with a threshold slightly above this first threshold and compare the results to separate and exclude the child concepts.
For desired ingredients which do not appear in the recipe yet we execute an additional adaptation step to replace one of the similar ingredients of the recipe with the desired ingredient. As a last step we count the amount of replacement candidates. If no adequate ingredients remain, we recommend to omit this forbidden ingredient.

4.2 Extraction of Adaptation Knowledge from Cooking Communities

The Internet and the Web 2.0 with its user-generated content is a large source of any kind of knowledge and experience. This includes knowledge and experience about cooking and about adapting recipes, which are discussed in cooking communities. In these communities people upload their favourite recipes for everybody to use and express their opinion about other peoples recipes. Thereby they do not only say what they like or what they do not like about the recipe, they also express the way they adapt the recipe to their needs. This can be for changing the taste, for following a certain diet or just because they did not have an ingredient at hand and took a different one. For this years CCC we implemented an approach that makes this knowledge available for our application.

We collected about 70'000 recipes with more than 280'000 comments from a large German cooking community by crawling the website. This way we got one HTML source-code page for each recipe with the corresponding comments. From this source code we extracted the relevant information entities using filters based on different HTML tags. For the recipe these entities were primarily the recipe title, needed ingredients and the preparation instructions. If users commented the recipe, we extracted the text of the comment, checked if the comment was an answer to another comment and if the comment is marked as a helpful comment or not. All these informations we stored in an database to have an efficient access to the data.

In the next step we used the e:IAS and an extended CookIIS knowledge model to generate two different case bases. One case-base consists of the recipes and one of the comments. Using the e:IAS TextMiner we extracted the mentioned ingredients and stored them as a case for each recipe and each comment. Since our knowledge model is bilingual (English and German) we were able to translate the originally German ingredient names into English terms during this process and also took care of used synonyms. This way had the same terms in the case bases that we use in our CookIIS application.

Having built up the two case bases we first retrieved a recipe, then all of the comments belonging to the recipe, and finally compared the ingredients of the recipe with the ingredients mentioned in the comment. This way we classified the ingredients mentioned in the comments into the following three categories:

- *New*: ingredients that are mentioned in the comment, but not in the recipe

- *Old*: ingredients that are mentioned in the comment as well as in the recipe
- *OldAndNew*: two or more ingredients of one class of our knowledge model, of which at least one was mentioned in the recipe and in the comment and at least one other one was only mentioned in the comment, but not in the recipe

For the CookIIS application the last class is the most interesting. We interpreted ingredients with this classification as either an adaptation (e.g. instead of milk I took cream) or an explanation/specialization (e.g. Gouda is a semi-firm cheese). For each of these ingredients classified as *OldAndNew* we also stored whether it is the new or the old one. We tried to distinguish between adaptation and specialization by looking for hints in the original comment text and by using the taxonomies of our knowledge model. Therefore we tried to find terms in the comment that indicate that an adaptation was described in the comment during the text-mining process (e.g. instead of, alternative, replaced with,...) and stored those terms in the corresponding case. Additionally we looked in the taxonomy of the ingredient class whether the one ingredient is a child of the other (or the other way around). If an ingredient is a child of the other we interpreted this as specialization or explanation, because one ingredient is a more general term than the other. This way we omit to have adaptations like: instead of semi-firm cheese take Gouda.

For each classified ingredient we assigned a specific score, which depends on the following factors:

- the number of ingredients found in the comment text
- whether the comment was marked as helpful or not
- whether a term was found that specifies the classification further or not
- whether a term was found that indicates a different classification or not

After assigning the score we aggregated our classification results. For the CookIIS application we did this in two steps: First we aggregated all classified ingredients of all comments belonging to one recipe. Thereby we counted the number of the same classifications in different comments and subsequently added up the score of the same classifications. Then we aggregated all classifications without regarding the recipe they belong to. This way we can select the most common classifications out of all classifications.

For this years CookIIS application we use the overall aggregation of *OldAndNew*-classified ingredients to generate adaptation suggestions. We look up this ingredient in our "community knowledge DB". If this ingredient is categorized as "old" we use the corresponding "new" ingredient to serve as substitution. If more than one substitution is found, we use the on with the highest score. We usually try to retrieve two adaptation suggestions to be more manifold. Using the approach we got more than 5'300 different adaptation suggestions for about 570 different ingredients of which we only use the most common (regarding the number appearances in the comments and the score). For the future we plan to use the adaptation suggestions with regard to the recipe they belong to. Our idea is to find similar recipes to the recipe the adaptation is done at out of our

pool of 70'000 recipes and find an adaptation suggestion from a similar recipe following the principle that similar recipes need similar adaptations.

The approach described above has a lot of advantages. For finding ingredients we can use our detailed CookIIS knowledge-model. This way we take care of synonyms and have the same terms in the application as well as in the adaptation-database. Since we are not looking for exact semantics but do our own classification we are independent from slang or informal language as often used in bulletin boards in the Internet. By using a large number of recipes and comments we hope to balance wrong classifications out and avoid suggesting false positives. Some more technical details and ideas for the evaluation are described in [9].

5 Menu Challenge

In this challenge CookIIS designs a three-course menu according to given constraints. The user can specify which ingredients should be used, which should be avoided and the *CookIIS Menu Creator* will compose a menu containing a starter, a main dish, and a dessert that fit together. Our underlying assumption is that each dish of a menu should have the same type of cuisine. The technical idea behind the Menu Creator is a two step retrieval [10].

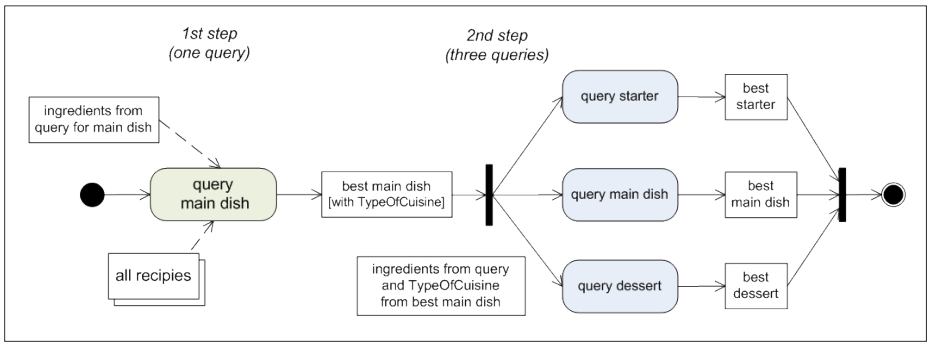

Fig. 1. Menu Design: two-query approach

Figure 1 illustrates our approach, in which a main dish is retrieved first to set the type of cuisine for the whole menu. In the second retrieval step the given ingredients and the type of cuisine are used to retrieve fitting recipes for starter, main dish and dessert. Within the second step we also make sure that the types of meal of the courses differ from each other.

6 Conclusion

In this paper, we presented CookIIS, a case-based system that provides recipe suggestions. We first presented the underlying knowledge model, which contains

25 classes with about 2000 concepts. We improved the knowledge model by taking additional knowledge obtained from cooking literature into account as well as the experience made with our system. In order to meet some of the requirements of the Computer Cooking Contest, various meta information about recipes can be automatically computed in CookIIS. We have a set of rules which is used for the extraction of the needed meta information. We also use a rule based approach to deal with dietary practices by only considering the recipes that can be recommended for a given diet. CookIIS uses a sequential adaptation based on a set of prioritized adaptation rules performing a sequence of steps to pursue two different approaches: on the one hand the model-based adaptation and on the other hand the community-based adaptation.
For the future we still have ideas to improve CookIIS. We are actually trying to consider the amount (i.e. the weight) of ingredients for the retrieval and the computation of meta information. Another improvement we are aspiring consists of learning association rules from the recipes in order to refine the adaptation.

References

1. Richter, M.M.: Introduction. In Lenz, M., Bartsch-Spörl, B., Burkhard, H.D., Wess, S., eds.: Case-Based Reasoning Technology – From Foundations to Applications. LNAI 1400. Springer-Verlag, Berlin (1998)
2. empolis GmbH: Technical white paper e:information access suite. Technical report, empolis GmbH (January 2008)
3. Lenz, M.: Case Retrieval Nets as a Model for Building Flexible Information Systems. Dissertation, Humboldt University of Berlin, Berlin (1999)
4. Hanft, A., Ihle, N., Bach, K., Newo, R., Mänz, J.: Realising a cbr-based approach for computer cooking contest with e:ias. In Schaaf, M., ed.: ECCBR 2008 Workshop Proceedings, Hildesheim, Tharax (2008) 249–258
5. Reinhard Löbbert, Dietlind Hanrieder, U.B., Beck, J.: Lebensmittel: Waren, Lebensmittel, Trends. Verlag Europa-Lehrmittel (2001)
6. Bergmann, R.: Experience Management: Foundations, Development Methodology, and Internet-Based Applications. Volume 2432 of LNAI. Springer-Verlag (2002) Habilitation.
7. Hanft, A., Ihle, N., Newo, R.: Refinements for retrieval and adaptation of the cookiis application. In Hinkelmann, K., Wache, H., eds.: Fifth Conference Professional Knowledge Management, Solothurn, Switzerland. Volume 145 of LNI., GI (2009)
8. Hanft, A., Ihle, N., Bach, K., Newo, R.: Cookiis – competing in the first computer cooking contest. Künstliche Intelligenz **23**(1) (2009) 30–33
9. Ihle, N., Hanft, A., Althoff, K.D.: Extraction of adaptation knowledge from internet communities. In Delany, S.J., ed.: ICCBR 2009 Workshop Proc., Workshop Reasoning from Experiences on the Web. (july 2009) to appear
10. Reichle, M., Bach, K.: Improving result adaptation through 2-step retrieval. In Nalepa, G.J., Baumeister, J., eds.: Proceedings of the 4th Workshop on Knowledge Engineering and Software Engineering (KESE 2008) at the 31st German Conference on Artificial Intelligence (KI-2008). (sep 2008) 73–84