A Case-Based Reasoning System for Recommending Cooking Recipes

Pedro Javier Herrera
Department of Computer Architecture and Automatic
Complutense University of Madrid, Spain

Gonzalo Pajares, María Guijarro
Department of Software Engineering and Artificial Intelligence
Complutense University of Madrid, Spain

1 Introduction

Case-Based Reasoning (CBR) is a successful paradigm within Artificial Intelligence (AI) of solving real-world problems based on precedent examples and problems (Aamodt & Plaza, 1994; Kolodner, 1993; Leake, 1996). Instead of relying solely on general knowledge of the problem domain, or making associations in the course of relations between descriptions of the problem and conclusions, this paradigm is capable of using specific knowledge of previous experiences (cases).

All CBR methods have to deal with central tasks that are to identify the current problem situation, find a past case similar to the new one, use that case to suggest a solution to the current problem, evaluate the proposed solution, and update the system by learning from this (Díaz-Agudo et al., 2007). This adaptation can get a new experience in solving problems with certain similarities, which leads to a differentiation CBR with the rest of trends, learning incrementally, as the new adaptations are stored as new cases related, and available for future comparisons.

CBR is required for creating numerous applications in a wide range of domains including diagnosis, help, desk, assessment, decision support, design, etc. Several organizations as IBM, British Airways and NASA are using CBR in applications such as customer support, quality assurance, aircraft maintenance, process planning and many more applications (Voskoglou, 2010).

The underlying ideas of CBR can be applied consistently across application domains although a CBR system designer has to decide from among a range of different methods for organizing, retrieving and reusing the knowledge retained in past cases (Aamodt & Plaza, 1994). Cases may be kept as concrete experiences, or a set of similar cases may form a generalized case (Maximini et al., 2001). Cases may be stored as separate knowledge units, or distributed within the knowledge structure (Bergmann et al., 1996). Cases may be indexed by a prefixed or open vocabulary, and within a flat or hierarchical index structure. The solution from a previous case may be directly applied to the present problem, or adapted according to differences between the two cases. Some CBR methods assume a rather large amount of widely distributed cases in its case base, while others are

based on a more limited set of typical ones. CBR methods may be purely self-contained and automatic, or they may be user guided.

As a general problem-solving methodology intended to cover a wide range of real-world applications, CBR must face the challenge to deal with uncertain, incomplete and vague information. In fact, uncertainty is already inherent in the basic CBR hypothesis demanding that similar problems have similar solutions, as mentioned above.

Choosing and designing recipes is an attractive problem for CBR research. It is an accessible and well understood domain with many individuals believing themselves to be experts in the area. It has proven popular with CBR researchers over the years with CHEF (Hammond, 1986) and Julia (Hinrichs, 1989), being two of the more well-known CBR cookery systems. CHEF is a case-based planner which represents recipes as cases using goals and problems, while Julia is a case-based design system. Both systems include an extensive semantic memory to hold the definitions of the terms needed, using frames organized into a semantic network in CHEF, and a large taxonomy of concepts in Julia.

In this chapter we describe a Java application that implements a CBR system applied to the domain of the cooking recipes and that solves the task of suggesting a recipe given a restricted set of ingredients as the query. We show the three versions of the system carried out so far, explaining the general improvements and new functionality of each one, e.g. using data mining techniques. This system has been implemented using frameworks written in Java, jCOLIBRI (Díaz-Agudo et al., 2007; Recio-García et al., 2009) and WEKA (Witten & Frank, 2005); the library OntoBridge also written in Java that provides management ontologies (OntoBridge, 2010), and JavaServer Pages technology (Fields et al., 2001). Advantages and disadvantages of each new system version are also given, and how new versions have improved the functionality of the system.

1.1 Motivation

The Computer Cooking Contest (CCC) is an annual international competition organized by the International and European Conference on Case-Based Reasoning (ICCBR and ECCBR, respectively). The purpose of this annual event is to provide a challenging, intuitive, and fun application focus for AI research that is open to all individuals and teams. The goal is to encourage the creation of novel development or integrations of AI techniques to solve problems related to cooking, where no restrictions are placed on the types of techniques that can be used. To accomplish this, it is held an open call for software systems that can solve one of a set of cooking challenges, a workshop at which the entrants' solutions are presented, and a live competition in which a chef and the program committee members judge the capabilities of the submitted systems.

The JaDaCook system was developed to participate in the CCC organized by the ECCBR in 2008[1]. The technical characteristics of this system are collaborative menus and queries in free text, describing the knowledge acquisition and reasoning processes.

The JaDaCook 2.0 system was developed to participate in the 2nd CCC organized by the ICCBR in 2009[2]. This system is an improved version of the finalist JaDaCook. We have reengineering the source code making the code more reusable and extensible. JaDaCook 2.0 includes general improvements and new functionality. Namely, a new form based interface, a new version of the ontology including more type of ingredients, dietary practices, types of meal type of cuisine, data mining over the ingredients, textual and IE capabilities.

JADAWeb was developed to participate in the 3rd CCC organized by the ICCBR in 2010[3]. The system has been developed over JaDaCook 2.0. As the main novelties it includes a Web based on natural language

[1] http://www.wi2.uni-trier.de/eccbr08/index.php?task=ccc

[2] http://www.wi2.uni-trier.de/ccc09/index.php

[3] http://vm.liris.cnrs.fr/ccc2010/doku.php

interface with parsing of the input data, a fuzzy similarity function and WordNet reasoning (Fellbaum, 1998) to identify and include new ingredients in the ontology.

The different system versions were also developed as the final evaluation assignment for a graduate course in Machine Learning at the Faculty of Computer Science of the Complutense University of Madrid (Spain).

1.2 Organization of the Chapter

The chapter is organized as follows. Section 2 summarizes the main features of our system giving details for each individual version: knowledge acquisition process, the conceptualization and formalization of the resulting ontology, and the case-based reasoning process focusing on the similarity function. Advantages and disadvantages of each new system version are also given, and how new versions have improved the functionality of the system. Section 3 describes a set of examples tested in the system versions. Finally, Section 4 shows the conclusions and suggests some lines of future work.

2 Modeling Our Problem

2.1 JaDaCook

In this chapter we first describe JaDaCook (Herrera et al., 2008), a Java application that implements a CBR system applied to the domain of the cooking recipes. Case knowledge is originally available from textual sources (Aamodt & Plaza, 1994). JaDaCook has been implemented using the jCOLIBRI framework (Díaz-Agudo et al., 2007; Recio-García et al., 2009). jCOLIBRI supports, among other things, textual processing (Recio-García et al., 2005) and the use of ontologies to enrich reasoning processes in CBR systems (Recio-García et al., 2006). JaDaCook uses an ontology we have formalized to help similarity computation and ingredient substitutions.

This first system version solves the three challenges proposed in the CCC'08. Namely, 'single dish challenge' to answer queries that require the selection and modification of a recipe for a single dish; 'negation challenge' to avoid certain ingredients, and 'menu challenge' to compose a three-course menu based on the available recipes based on collaborative recommending techniques. It allows including new ingredients in the ontology, new menus in the collaborative case base, and new recipes that are automatically processed from the given file. The code of the system is available through the Web site: http://gaia.fdi.ucm.es/grupo/projects/cookingContest/cookingContest .html#jadacook

This section is structured as follows. Section 2.1.1 details the main features of our system. Section 2.1.2 describes the knowledge acquisition process. Section 2.1.3 shows the CBR process.

2.1.1 Main Features

JadaCook integrates in the same GUI (Figure 1) the following challenges: *a)* Single Dish Challenge. Given a query, the result will be the retrieval and adaptation of a recipe for a single dish; *b)* Negation Challenge. This option lets you enter queries with ingredients that the user does not want to appear in the final recommendation for personal reasons; *c)* Menu Challenge. It allows the creation of a three course menu. The collaborative part takes in account the decisions taken by previous users for future recommendations.

We detail the most important aspects of these three features of the system in (Herrera et al., 2008).

2.1.2 Knowledge Acquisition

The knowledge acquisition has been distributed in two steps. First, we looked for existing resources to reuse, and second, formalize the acquired knowledge. We first collect information about types of ingredients, cooking and types of diet from various sources including experts, dictionaries and cooking Web sites (MHLW, 2006). The quest for knowledge was based not only on gathering information in different ways (Hammond, 1986), but the organization itself provided a big base of cases of recipes (900 approximately), to start working and conduct relevant queries.

JaDaCook reasons with an ontology that formalizes the cooking domain knowledge. The ontology organizes ingredients in categories. In the first level there are ingredients of animal origin grouped in some subsets: meat, fish, milk, cheese and eggs. There are also ingredients of plant origin grouped in subsets like cereals, fruits, and vegetables. And finally, there are other specific families of ingredients like sweets, drinks, or basics ingredients like oil or salt. The ontology in its current state is very extensive and complete; it has 206 classes and more than 200 ingredients.

Figure 1: JaDaCook Main Window.

The acquired knowledge is structured, conceptualized and formalized in the Web Ontology Language (OWL) (Herman, 2010) as a reusable ontology. Additionally, a transformation process in the cases of recipes was made in order to work with them in a better way. We use a case structure with the following attributes: case number, title of the recipe, list of ingredients and development process (textual description).

In order to perform an efficient search and the best results, it became necessary to link each case of the base of recipes with the appropriate classes of ontology. Thus, the mapping that emerged was really easy, each ingredient is the type of class ingredients and the number of cases it gives the root class recipe.

2.1.3 CBR Process

The case retrieval process consists of obtaining more similar cases that meet a given query. We focused efforts on finding a good measure of similarity, which put together the most similar to each other and a method for finding the most appropriate cases. The retrieval method we used is the classical *nearest neighbor* (Cover & Hart, 1967) that search comparing the attributes of cases in numerical form. It makes use of global functions to

compare cases and functions of local similarity to compare the attributes of the simple case. The similarity function works as follows. First, the system takes the ingredients for the wish list of the user, and it searches for recipes that contain these ingredients. If a recipe contains an ingredient it increases by 1.0 the similarity value. The similarity value is normalized by the number of ingredients in the query.

The reuse stage is responsible for adapting the solution of the cases recovered to the requirements of the queries, if necessary. In our system, the reuse strategy employed is called *reinstantiation* and it is a kind of strategy based on replacement in which the replacement takes place between elements with the same role but different values. During adaptation, JaDaCook does not reject an ingredient when it is not in the recipe. When it happens, the system searches for ingredients that share the same branch of the original ingredient in the ontology. If it is feasible the algorithm increases by 0.8 the similarity value of this recipe. The system searches in all recipes contained by the case base. Finally, the system gets a list of recipes with the information of how similar each one is in according to the ingredients introduced by the user.

The user can also specify in the query the type of meal he wants, the type of diet, the type of cuisine and if he wants some dietary practices. The system adds some ingredients if the user wants a specific type of meal, i.e. if the user selects a Chinese meal the system will add some ingredients like rice, bamboo, carrot, ginger, chicken, duck, pig, pork or soy sauce.

Importantly, the similarity in the case of an adaptation is not total, because as its name suggests, it adjusts (substitutes) for any similar ingredient.

Lastly, an aspect that has not been discussed is that adaptation is performed within the scope of the brothers; a decision based on the fact that if they are also adapted by cousins and other relations of the tree of ontology, the solution to the query would be quite distant to the ingredients required, giving undesired results.

The stage of learning (retain) has as its primary function storing those cases which have been adapted into the knowledge base. Thus, these new cases may be used in future searches for that recovery is better and more reliable. In this system, is left to the user which cases to store, so in this way we store the most important cases which have better solve the problem.

2.2 JaDaCook 2.0

In this section we describe JaDaCook 2.0 (Herrera et al., 2009), a CBR system that solves the task of suggesting a recipe given a restricted set of ingredients as the query. This version of the system was developed to participate in the CCC'09. It is an improved version of the JaDaCook that participated in CCC'08.

We have reengineering the source code of JaDaCook making the code more reusable and extensible. JaDaCook 2.0 includes general improvements and new functionality. Concretely, a new form based interface (Figure 2), a new version of the ontology including more type of ingredients, dietary practices, types of meal type of cuisine, data mining over the ingredients, textual and IE capabilities. The code of the system is available through the Web site: http://gaia.fdi.ucm.es/projects/cookingContest2/cookingContest.html#JaDaCook

JaDaCook 2.0 reasons using different knowledge sources: *1)* a case base of recipes (provided by the organization and available from textual sources); *2)* a cooking ontology; *3)* a set of association rules of co-apparition of ingredients in the recipes obtained using data mining techniques. These rules are used to proposed substitution ingredients. *4)* A case base of menus built collaboratively using the personal opinion of expert users. This case base is used to build menus from the composition of single dishes.

This version also solves tasks that involves answering queries that require the selection and, where appropriate, modification of recipe for a single dish; and those that requires the composition of a three-course menu based on the available recipes and on collaborative recommending techniques. It adapts the retrieved

recipe by substituting ingredients; however it does not change the recipe step structure. It has also been implemented using jCOLIBRI (Díaz-Agudo et al., 2007; Recio-García et al., 2009) framework written in Java.

This system version also offers a case based collaborative recommender system that bases the menu configuration on the opinion of previous users. When the user queries the menu system, he is asked about his opinion on the result that is recorded and used for future recommendation.

In this section we present a brief review of the technical characteristics of the system, describing the knowledge acquisition, reasoning processes, and the new functionality. This section is structured as follows. Section 2.2.1 details the main features of the graphical interface of the system. Section 2.2.2 describes the knowledge sources of the system and section 2.2.3 briefly explains the CBR process.

Figure 2: JaDaCook 2.0 Main Window. Single dish tab and Ontology tree.

2.2.1 Main Features

JaDaCook 2.0 includes a new form based interface based on the QT multiplatform framework (Qt, 2010). The new interface substitutes the previous natural language interface, makes easiest the query formulation process, and minimizes errors in the communication process with the user. The new interface allows navigating the case base, querying it by selecting from the ontology specific ingredients or types to appear or to avoid and dietary

practices. It also allows querying the system using new ingredients, including new ingredients in the ontology, including new menus in the collaborative case base and new recipes that are automatically processed from the given file.

The new interface integrates in the same GUI (Figure 2) three tabs (from left to right). *1)* 'Single Dish Challenge'. Given a query, the result will be the retrieval and adaptation of a recipe for a single dish. In the single dish tab the user provides the query information as the list of ingredients that the user would like or would not like to include, dietary practices like vegetarian, nut-free or non-alcoholic, plus new dietary practices like gout diet, cholesterol free, etc., and type of cuisine, e.g. Chinese, Mexican, Mediterranean or Italian. *2)* 'Menu Challenge'. It allows the creation of a three course menu. The collaborative part takes in account the decisions taken by previous users for future recommendations. *3)* 'Recipes Inspection' where the user can consult the 1484 recipes in case base.

2.2.2 Knowledge Acquisition

In this version the knowledge of the system has evolved. Apart from the case base, the main source of knowledge is the ontology where we have conceptualized and formalized cooking knowledge from different sources, including experts, dictionaries and cooking Web sites and systems (Hammond, 1986; MHLW, 2006). The quest for knowledge was based not only on gathering information in different ways but the organization itself provided a big base of cases of recipes, about to start working and conduct relevant queries. The acquired knowledge has been incrementally structured, conceptualized and formalized in OWL (Herman, 2010) and in this version includes ingredients from the new recipes, new types of ingredients, new types of cuisine and dietary practices, and classification of recipes and ingredients according to these dietary practices, namely gout diet knowledge, seasonal food knowledge, and cholesterol Diet knowledge.

Like its predecessor, JaDaCook 2.0 reasoning is based on an ontology that captures the terminological knowledge of the cooking domain, organizing objects (individuals) into categories (concepts) what allow inherit properties and to create taxonomies. There are animal origin ingredients, grouped as fish, meat, milk, cheese and eggs, each with other subclasses; plant origin ingredients, like cereals, nuts, fruits and vegetables; and other classes like sweeteners, drinks, basics, like salt and oil. In the new version of the ontology we have including additional knowledge. It has more than 300 ingredients, organized in types and classes that cover the 1484 recipes we have available. In Figures 3 and 4 are shown the 'Animal Origin' and 'Fruits' sub trees of the ontology. The OWL code of the ontology is available through the system Web site. (http://gaia.fdi.ucm.es/grupo/projects/cookingContest2/jadacook2/OntologiaIngredientes.owl).

In this new version of the system we use the case base provided by the organization of the CCC'09. They are in an *xml* file, that is processed and the most relevant information is extracted and stored in case base memory structure in the *precycle* of the application (Recio-García, 2008). Processes of loading and storing cases have also been improved in this version using SAX and DOM (Lam et al., 2008). Each case in the case base has mainly a text title, a list of ingredients and a textual description of the recipe development process. Each case is linked with the corresponding classes of ingredients in the ontology.

We have applied the *Apriori* algorithm (Agrawal et al., 1993) to extract association rules using WEKA (Witten & Frank, 2005) over the case base of recipes. WEKA is a framework written in Java to apply data mining. These rules allow capturing the degree of compatibility between ingredients (Herrera et al., 2009). When the user asks for ingredients to include in the recipes, the system uses these rules to suggest ingredients that appear together with the previously selected ingredients in a bigger number of recipes. This utility is one of the novelties of this version of the system.

The menu case base includes 133 menus acquired in a collaborative way by different users of the system. Each menu is composed of three courses taken from the recipe case base. If a user composes a menu using three single dish queries, then the menu is saved in an *xml* file to be reused in the future. Each menu has a scoring value that can be modified. New menus and new scoring values will be taken into account for future menu recommendations. An *xml* menu example is shown in the following paragraph.

```
<MENU>
<ST>Fusilli Verde with Broccoli and Red Bell Pepper</ST>
<MC>Fast with Five: Garlic Flank Steak with Onion</MC>
<DE>Cocoa Espresso Cooler</DE>
<SC>7</SC>
</MENU>
```

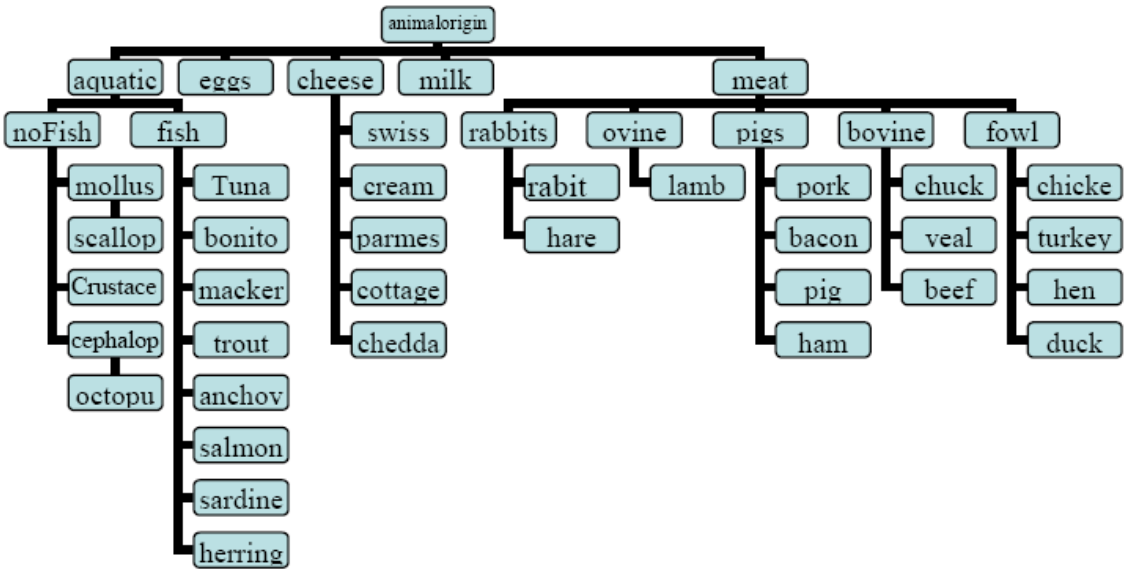

Figure 3: Sub tree which represents the ontology for 'Animal Origin' ingredients.

2.2.3 CBR Process

JaDaCook 2.0 CBR processes take advantage of the ontology. The main usage of the cooking ontology has been centered on similarity assessment and ingredient substitution. The assumption here is that two ingredients are more similar if they are located closer in the ontology. And that one ingredient can be substituted for an ingredient that is classified in the same concept. For example, turkey and chicken are siblings and children of the concept 'Fowl' (like hen, and duck).

The single dish case retrieval process consists of obtaining recipes that are similar to the given query. That means that the retrieved recipe includes the ingredients in the 'would like to' list and it does not include ingredients from the 'would not like to' list, and it has the dietary restrictions and type of cuisine specified in the query through the graphical interface (Figure 2).

The retrieval method is the nearest neighbor (Cover & Hart, 1967) that compares these characteristics and aggregates them obtaining a ranking similarity number to order the candidates. Then the k most similar are shown to the user. In JaDaCook 2.0 we have improved the similarity measure; we have included more knowledge from the ontology. Similarity includes two local similarity functions to compare titles and ingredients of the query and the case. Once all the ingredients have been processed, value computation will be normalized in the range [0,1].

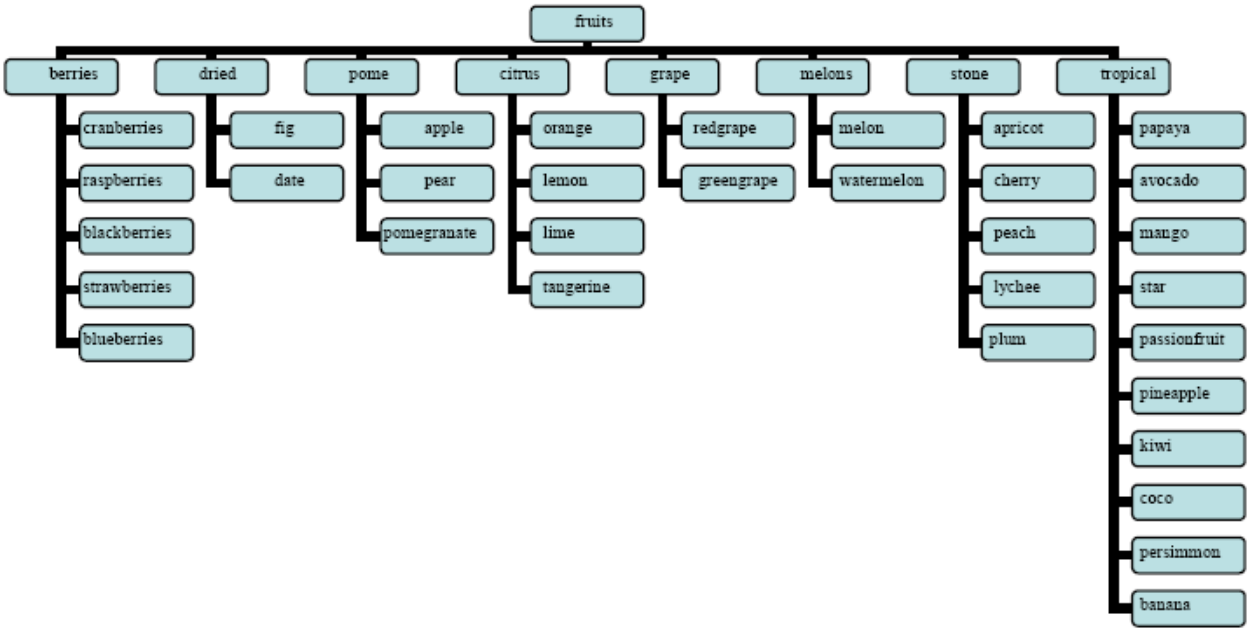

Figure 4: Sub tree which represents the ontology for 'Fruits' ingredients.

We use jCOLIBRI similarity functions, more specifically one of the concept based similarity functions that depends on the location of the cases in the ontology. Details can be found in (Recio-García et al., 2006). If the ingredients in the 'would like to' list are concepts of the ontology that represent type of ingredients, then the children are obtained. To that end, we used the library OntoBridge written in Java that provides management ontologies (OntoBridge, 2010).

The reuse process in JaDaCook 2.0 is a process based on the substitution of ingredients according to their position in the ontology. The reuse strategy is the same one that was employed in the first system version. It is one of the reuse methods included in the library of jCOLIBRI and it is based on substituting one ingredient by one of its siblings in the ontology structure.

The stage of learning (retain) has as its primary function storing those cases which have been adapted into the knowledge base. Thus, these new cases may be used in future searches for that recovery is better and more reliable. In this system, is also left to the user which cases to store, so in this way we store the most important cases which have better solve the problem.

2.3 JADAWeb

In this section we describe JADAWeb (Ballesteros et al., 2010) based on JaDaCook 2.0 that suggests recipes using as input a set of ingredients to include and to avoid, and some optional dietary and cooking restrictions. JADAWeb also retrieves a recipe from the system case base and includes the capability of adapting it by substituting its ingredients by other similar ingredients that appear in the user query. However, the system uses a fuzzy similarity function to determine if two ingredients can be swapped in the recipe. It has also been implemented using the jCOLIBRI framework (Díaz-Agudo et al., 2007; Recio-García et al., 2009) and its facilities to design CBR systems.

This version of the system has a Web interface built using JavaServer Pages (JSP) (Fields et al., 2001) technology where the input of data is a single character string on natural language in English. The online system is accessible via `http://supergaia.fdi.ucm.es:8810/CCCWeb`. It has included an algorithm to detect the negation in a sentence, so the system is able to infer which ingredients the user wants to include and to avoid. The algorithm is based on multilingual dependency parsing (Buchholz & Marsi, 2006). JADAWeb also includes a number of substantial improvements in the decision of which nouns identified in the query are ingredients and which are not. Its predecessors only check if the noun appears in the ingredient ontology. However, this version also searches the noun in WordNet (Fellbaum, 1998) to determine if the noun is an ingredient or not. If an ingredient is identified in the query and it was not defined in the system ontology, the system suggests the area of the ontology is better to classify the new ingredient.

After retrieval, suggestions are ordered by similarity and also according with the characteristics of the season. Hotter meals, like a soup, will be presented before in winter, and fresh meals, like salad, will be presented before if the query is in summer. The information used to make this decision is the presence, or not, of some important words that are present in each recipe, like ‘hot’, ‘oven’, or ‘fresh’.

This section is organized as follows. Section 2.3.1 describes the Web-based graphical user interface and the parsing of the input data. In section 2.3.2 it is defined how the acquisition of knowledge works. Section 2.3.3 describes the CBR process, focusing on the similarity function.

2.3.1 Main Features

The Web interface allows the user writing a natural language sentence with the recipe requirements (Figure 5). The following is an example of a query where the user explains what he wants: “*I want to eat some fruits, like apple, but I don't like kiwi*”. The system connects to the dependency text parser to obtain the information contained in this query. It extracts the list of ingredients to include and to avoid in the recipe.

The system infers the lists of ingredients from the query text and shows the result of the parser. The user also can express if he wants to choose some dietary practices, or type of cuisine using a form type of interface.

2.3.2 Knowledge Acquisition

JADAWeb reasons using two main knowledge sources: the recipe base provided by the contest organization, and the ontology that has been built incrementally and collaboratively from the first versions of the system. The ontology is also conceptualized and formalized in the OWL language (Herman, 2010).

As a novelty, this version uses WordNet (Fellbaum, 1998) to let the user include ingredients that are not previously integrated in the ontology. The textual parsing of the input query searches in WordNet to decide if a not identified noun in the query is an ingredient or not.

The ontology organizes ingredients in categories. The ontology in its current state is very extensive and complete, it has 255 classes, and 202 of them are ingredients. It has been made little changes to make easier the search and the equality between concept names because WordNet has different terms than our previous ontology; e.g. it has been changed 'AnimalOrigin' and 'PlantOrigin' for 'animal' and 'plant'.

Despite the richness of the ontology and the wide range of definitions in the cooking domain, it has been implemented a system to detect new ingredients in the input by searching the nouns in the WordNet lexical database. This lexical database allows searching a concept to get the definition, synonyms and antonyms, among other things.

Figure 5: JADAWeb Web-Based Graphical User Interface.

For every noun in the query that is also in WordNet, it is a candidate to be an ingredient, the system checks if it is already included in the ontology. If the ingredient is not in the ontology, is tried to automatically include and classify it in the ontology. Using WordNet the application checks whether the definition of that noun is feasible to belong to any of the categories present in the ontology, if it is, JADAWeb adds it to the system memory, if the system cannot select a unique branch, it asks the user to select the concept to allocate the ingredient.

In this new version it is used the same case base than its predecessor. The recipe cases are also organized in an *xml* file that is processed using SAX and DOM (Lam et al., 2008) to extract the information to load it in the system memory.

2.3.3 CBR Process

JADAWeb includes new features. The first one is the ability to look for ingredients to adapt not only among the siblings, but also going up in the hierarchy. The similarity function has the same criteria as the degree of consanguinity of two people. Two siblings have a similarity value of 0.6, because both are descendent of the same parents. Therefore, a rise of level, subtract 0.4 of similarity. For two cousins, subtract 0.8: 0.4 because of the rise to the parent and other 0.4 because of the climb to the grandparent. Going down in the level, not penalize. Thus, another example: the children of a sibling of an ingredient have the same similarity as the sibling (0.6), the similarity of a cousin's children are the same as the cousin and also the same as any of the grandfather's descendants (which do not fall for the same branch of the one that is being compared) which is 0.2 in this case. In this way, you can use other similar ingredients. The number of levels can be parameterized and the default value is 2, i.e. the system only searches among siblings and their descendants.

Another novelty in this version is a table of similarity between fuzzy ingredients. It is a table with specific fuzzy values that gives meaning to say that e.g., rice is replaceable by macaroni with a value of 0.4 to add to the similarity function. All these pairs of ingredients are defined in an *xml* file which indicates what ingredients are similar and the associated fuzzy value. The fuzzy table is not only used to replace pairs of ingredients. The system also can decide replacing one ingredient by a group of similar ingredients, e.g., stock instead of water and bouillon cubes with a similarity of 0.9, milk instead of water and cream in a similarity value of 0.6.

3 System Examples

In this section we show a list of examples of the three versions of the system. For each query we describe one recipe proposed by the system. For each query the system returns the best *k* suggestions ordered by the similarity value. *k* is a modifiable parameter by the user.

3.1 JaDaCook

Query 1: "*Cook a main dish with meat and cauliflower*". One example result is shown in Figure 6. In this example, the meat selected is turkey and cauliflower has been recovered successfully. Similarity value is 1. For this query, the system retrieved three recipes with similarity value 1, and three with 0.9. In these cases, cauliflower has been substituted by broccoli.

Details of this and other examples can be obtained from the system Web site: http://gaia.fdi.ucm.es/ grupo/projects/cookingContest/cookingContest.html#jadacook.

3.2 JaDaCook 2.0

Query 2: "*Prepare a low-cholesterol dessert with strawberries and avoid citrus fruits*". The 'would like to' list includes the strawberries ingredient and we want to avoid citrus so we include it in the 'would not like to' list. Dietary practice is set to Cholesterol diet. Figure 7 shows the recipe '*Fruit Cup#1*' with a similarity value of 1. It includes strawberries and none of the citric fruits: orange, lemon, lime y tangerine.

Others examples of this version of the system are available through the Web site: http://gaia.fdi.ucm. es/projects/cookingContest2/cookingContest.html#JaDaCook.

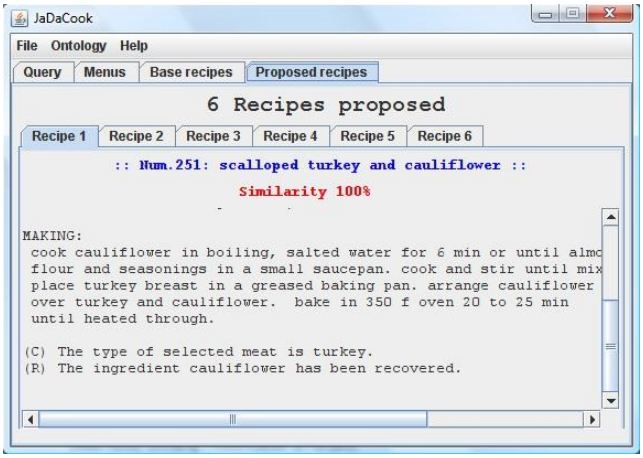

Figure 6: Recipes retrieved for query 1 with the first system version.

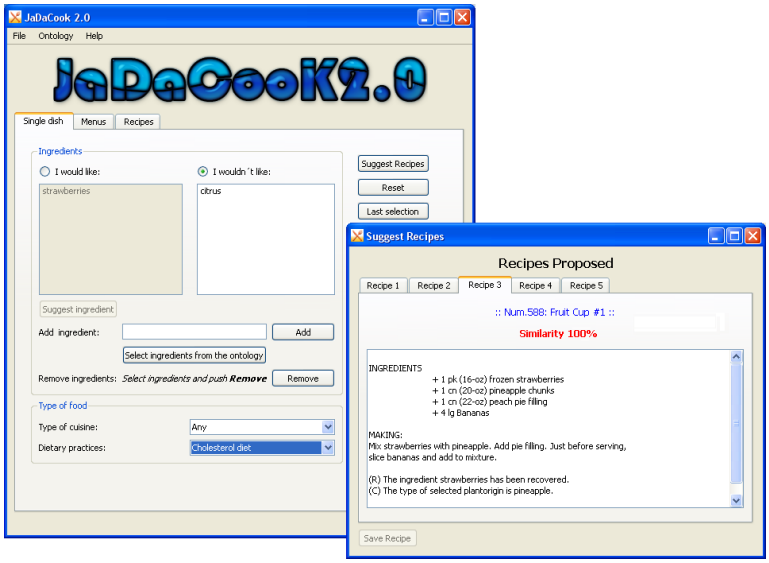

Figure 7: Recipes retrieved for query 2 with the second system version.

3.3 JADAWeb

Query 3: "*I'd rather like pepper, tomato, cheese and onions, but I detest sardines and anchovies*". Figure 8 shows the recipe '*Corona Cheese Chili*' with a similarity value of 1. It includes pepper, tomato, cheese, onion and it does not include sardine and anchovy.

The online system is accessible via http://supergaia.fdi.ucm.es:8810/CCCWeb where other examples can be obtained.

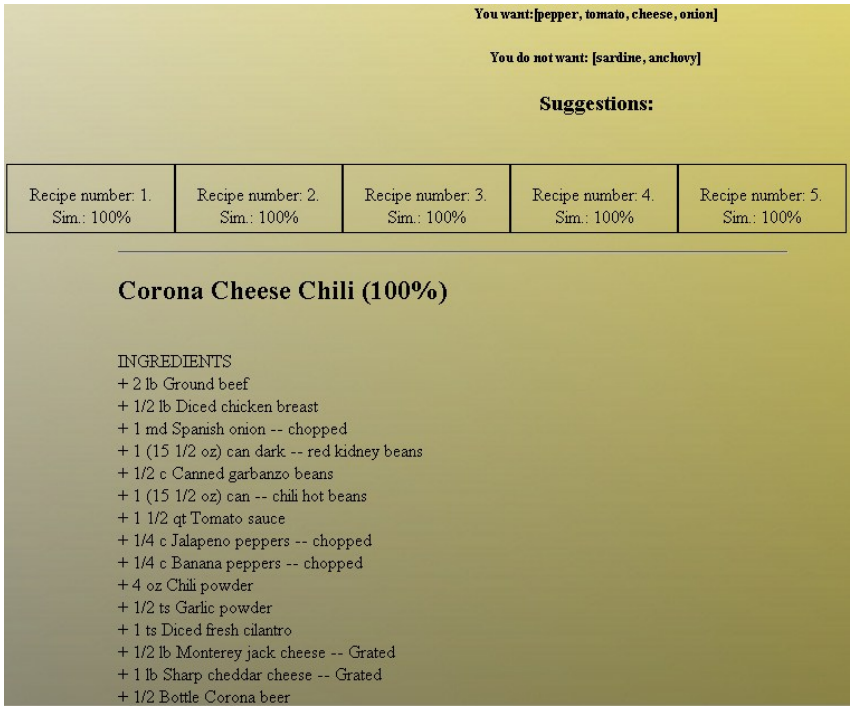

Figure 8: Recipes retrieved for query 3 with the third system version.

4 Discussion and Conclusions

In this chapter we describe a CBR system for recommending cooking recipes. The three versions of the system shown have participated in the final of the Computer Cooking Contest for three consecutive years.

The system reasoning relies on domain ontology of ingredients, types of ingredients, type of meal, dietary practices and type of cuisine; plus a case base of existing given recipes.

The first version of the system is able to learn new ingredients appearing in queries by including them into the hierarchical organization of ingredients. This is a way of supervised learning. The system also learns new menus, such as combinations of existing individual dishes. The collaborative extension of the system is based on the opinion of the final users.

The second version solves first version main drawbacks, includes new functionality, improvements on the ontology, a new interface, and data mining capabilities to capture dependencies between ingredients. Reasoning is based on a case base of recipes and an ontology with reusable knowledge about ingredients, types of ingredients, types of cuisine and dietary practices. The ontology is used as background knowledge to measure similarity between ingredients and single dishes, and to substitute ingredients during adaptation.

The third version improves and extends the system. Its Web-based interface gives a broader access to the system and Web accessibility. It also extends the possibilities by incorporating a fuzzy table which allows similarity between two distant elements or types in the ontology, and the inclusion of various levels in the adaptation algorithm. It should be noted that using WordNet allows more synonyms for some terms and it extends largely the knowledge that it has, avoiding the limitation of the use of a finite ontology of ingredients.

A drawback of the system is its high computational cost when the case base size increases. In the future, this could be improved through parallel implementations.

Another research line will come from using a system that allows voice recognition that could be connected with our dependency parser.

The techniques and concepts of the CBR system proposed could be adapted on solving real-world problems, on domains as varied as diagnosis, help, desk, assessment and decision support.

References

Aamodt, A., & Plaza, E. (1994). Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. *AI Communications*, 7(1), 39-59.

Agrawal, R., Imielinski, T., & Swami, A. N. (1993). Mining Association Rules between Sets of Items in Large Databases. *SIGMOD*, 22(2), 207-16.

Ballesteros, M., Martín, R., & Díaz-Agudo, B. (2010). JADAWeb: A CBR System for Cooking Recipes. In *ICCBR Workshops Proceedings. Computer Cooking Contest* (pp. 179-188).

Bergmann, R., Wilke, W., Vollrath, I., & Wess, S. (1996). Integrating general knowledge with object-oriented case representation and reasoning. In *4th German Workshop Case-Based Reasoning System Development and Evaluation*. vol. 55 (pp. 120–127).

Buchholz, S., & Marsi, E. (2006). Conll-x shared task on multilingual dependency parsing. In *Proceedings of the 10th Conference on Computational Natural Language Learning* (pp. 149-164).

Cover, T., & Hart, P. (1967). Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, 13(1), 21-27.

Díaz-Agudo, B., González-Calero, P. A., Recio-García, J. A., & Sánchez, A. (2007). Building CBR systems with jCOLIBRI. *Special Issue on Experimental Software and Toolkits, Science of Computer Programming*, 69(1-3), 68-75.

Fellbaum, C. (1998). *WordNet: an electronic lexical database*. MIT Press.

Fields, D. K., Kolb, M. A., & Bayern, S. (2001). *Web Development with Java Server Pages*. Manning Publications Co., Greenwich, CT, USA.

Hammond, K. J. (1986). Chef: A model of case-based planning. In *AAAI Proc of the 5th National Conf on Artificial Intelligence* (pp. 267-271).

Herman, I. (2010). Web Ontology Language (OWL). http://www.w3 .org/2004/OWL/ (Available on-line)

Herrera, P. J., Iglesias, P., Romero, D., Rubio, I., & Díaz-Agudo, B. (2008). JaDaCook: Java application developed and cooked over ontological knowledge. In *ECCBR Workshops Proceedings. Computer Cooking Contest* (pp. 209–218).

Herrera, P. J., Iglesias, P., Sánchez, A. M. G., & Díaz-Agudo, B. (2009). JaDaCook 2: Cooking over ontological knowledge. In *ICCBR Workshops Proceedings. Computer Cooking Contest* (pp. 279-288).

Hinrichs, T. R. (1989). Strategies for adaptation and recovery in a design problem solver. In *Proc Second Workshop on Case-Based Reasoning* (pp. 115-118).

Kolodner, J. (1993). *Case-Based Reasoning*. Morgan Kaufmann Publ., San Mateo.

Lam, T. C., Ding, J. J., & Liu, J. C. (2008). XML Document Parsing: Operational and Performance Characteristics. *Computer*, 41(9), 30-37.

Leake, D. B. (1996). *Case Based Reasoning: Experiences, Lessons and Future Directions*. AAAI Press, MIT Press, USA.

Maximini, K., Maximini, R., & Bergmann, R. (2001). An investigation of generalized cases. In *Proc. of the ICCBR 2003* (pp. 261–275).

MHLW, Department of Food Safety, Japanese Ministry of Health, Labour and Welfare (MHLW). (2006). *Introduction of the Positive List System for Agricultural Chemical Residues in Foods*.

OntoBridge. (2010). http://gaia.fdi.ucm.es/grupo/projects/ontobridge/

Qt cross-platform application framework (2010). http://www.qtsoftware.com/.

Recio-García, J. A., Díaz-Agudo, B., Gómez-Martín, M. A., & Wiratunga, N. (2005). Extending jCOLIBRI for Textual CBR. In *Proceedings of Case-Based Reasoning Research and Development, 6th International Conference on Case-Based Reasoning* (pp. 421-435).

Recio-García, J. A., Díaz-Agudo, B., González-Calero, P. A., & Sánchez, A. (2006). Ontology based CBR with jCOLIBRI. In *Applications and Innovations in Intelligent Systems XIV*. *Proceedings of AI-2006, the Twenty-sixth SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence* (pp. 149–162).

Recio-García, J. A. (2008). jCOLIBRI: A multi-level platform for building and generating CBR systems. *Phd Thesis*, Department of Software Engineering and Artificial Intelligence, Faculty of Computer Science, Complutense University of Madrid.

Recio-García, J. A., Díaz-Agudo, B., & González-Calero, P. A. (2009). Boosting the performance of cbr applications with jcolibri. In *ICTAI, IEEE Computer Society* (pp. 276-283).

Voskoglou, M. G. (2010). A Fuzzy Systems Framework for solving real world problems. *WSEAS Transactions on Systems*, 8(9), 875-884.

Witten, I. H., & Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*, Second Edition. Morgan Kaufmann Publishers.