Considering Nutrients during the Generation of Recipes by Process-Oriented Case-based Reasoning

Christian Zeyen, Maximilian Hoffmann, Gilbert Müller and Ralph Bergmann

Business Information Systems II
University of Trier
54286 Trier, Germany
[zeyen][s4mnhoff][muellerg][bergmann]@uni-trier.de,
http://www.wi2.uni-trier.de

Abstract. This paper investigates the generation of recipes in consideration of user-defined nutrient contents. For this purpose, we extend our previous case-based reasoning approach that already covers the formulation of user queries with various dietary practices. More precisely, this work augments the domain ontology with nutritional information and introduces a novel nutrition concept fulfillment into the retrieval and adaptation process. An experimental evaluation with real cooking recipes demonstrates the applicability of the approach and systematically investigates the influence of various adaptation methods on the query fulfillment with multiple constraints. It is shown, that all adaptation methods are able to optimize generated recipes according to certain nutritional constraints as well as ingredient and cooking step preferences and that the adaptation outperforms the sole retrieval of available recipes.

Keywords: Process-Oriented Case-Based Reasoning · Workflow Adaptation · Recipe Generation · Nutrient Content

1 Introduction

A healthy and balanced diet is of high importance for many people. In some cases, allergies or intolerances require particularly a nutrition that is tailored to individual needs. Thus, numerous different dietary practices exist such as avoiding specific kinds of foods when following a lactose-free or vegetarian diet, preparing food in a certain way for a raw food diet, or reaching a certain amount of nutrients while being on a high-protein diet. Cooking portals make it convenient for amateur chefs to find desired recipes by providing diverse search capabilities such as a faceted search for different diets or local specialties. While many portals enable the adaptation of ingredient quantities for a desired number of servings, the vast majority is not able to customize available recipes according to more comprehensive user queries, e.g., by exchanging ingredients or adapting preparation steps in a recipe. Since 2008, the computer cooking contest (CCC) has been organized almost every year and provided varying cooking challenges some of which demanded the creation of recipes for certain dietary practices. In recent years, various applications emerged mainly utilizing case-based reasoning (CBR) [1] for the experience-based creation of recipes by retrieval and adaptation of available

recipes. For example, JadaWeb [2,7] is a textual CBR system that provides desired and undesired ingredients as query capabilities and does also consider pre-defined diet restrictions such as vegetarian, nut-free, or non-alcoholic using the knowledge encoded in an ontology. The CookIIS system [6] provides similar capabilities and utilizes predefined lists of recommended and not recommended ingredients for certain diets such as vegetarian, gout, or low-cholesterol diets. Not recommended ingredients are used as filters during retrieval and the subsequent rule-based adaptation tries to insert recommended ingredients to further optimize the retrieved recipe. Taaable [4,5] is another textual CBR system that is able to substitute ingredients and adapt their quantities based on a semantic wiki and a domain ontology that captures dietary practices and nutritional information about ingredients and dishes. Our contribution to this contest is a process-oriented CBR (POCBR) [9] system named CookingCAKE [12]. We represent recipes as cooking workflows in order to enable queries containing ingredients in conjunction with preparation instructions. In the process-oriented query language (POQL) [15] the user can specify desired and undesired ingredients and/or preparation steps as workflow fragments. Analogous to other approaches, the user may specify queries for recipes that comply with different dietary practices. For instance, vegetarian dishes can be requested by inserting meat as a restriction and a diet with raw food can be achieved by adding a generalized preparation step increase temperature as a restriction to the query. However, we are not aware of any approach that covers the specification of queries for dishes having a certain distribution of nutrients. Thus, in this paper, we propose an extension of the recipe generation that additionally considers nutrient contents.

The contribution of this paper is twofold. From an application-oriented point of view, we augment the case representation with ontological knowledge about nutrition, nutrients, and conversions of ingredient quantities. Furthermore, besides the POQL query, we introduce a second query component for the specification of desired nutrient contents. This also requires the extension of the retrieval and adaptation with a novel nutrient content fulfillment. Due to the additional query component, the creation of recipes becomes a multi-objective optimization problem and thus the system may not be able to maximize the fulfillment of both query parts at the same time. Hence, from a methodological perspective, we provide an experimental evaluation that systematically examines the influence of adaptation methods on the solution of this problem.

This paper is organized as follows: In section 2 we briefly summarize relevant previous work in the field of POCBR. Subsequently, section 3 describes our approach. The experimental evaluation is presented in section 4 while section 5 summarizes our findings and discusses future work.

2 Generating Recipes with Process-Oriented CBR

We now briefly give some background information about our previous work on the generation of recipes represented as workflows with POCBR.

2.1 Similarity-based Workflow Retrieval

In POCBR, cases are often represented as processes or workflows that describe the logical or chronological order of tasks and the exchange of physical products or data

to produce a certain outcome [17]. In cooking workflows, tasks represent preparation instructions and exchange ingredients in order to produce a certain dish. We represent workflows as semantically labeled directed graphs [3]. In the graph representation, nodes and edges have semantic descriptions consisting of pairs of attributes and values. Figure 1 gives an example of a simple cooking workflow for the preparation of a tomato

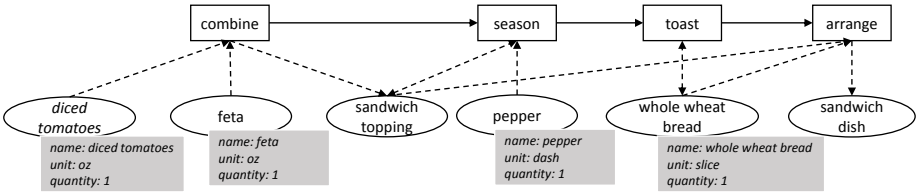

Fig. 1. Example of a cooking workflow

cheese sandwich. Each task and data node has a *name* attribute used as the label. The values for this attribute are structured hierarchically in a data and task taxonomy in order to enable the assessment of the semantic similarity among them. The data nodes that represent real ingredients have an extended semantic description specifying their *unit* and *quantity*.

Queries in POCBR are used to describe the users' requirements for retrieving the most useful workflows. In previous work [15], we proposed a process-oriented query language (POQL) to specify such queries. One part of a POQL query $Q = (Q^+, Q^-)$ is a query workflow representing desired properties of a workflow. The other part consists of several restriction workflows each defining one undesired situation that should be avoided. Figure 2 illustrates such a POQL query. In this example, the query workflow

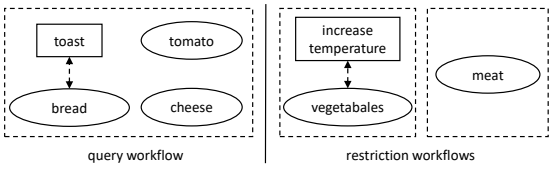

Fig. 2. Example of a POQL query

specifies the desired properties toasted bread, tomato, and cheese. The first restriction workflow restricts the desired dish to be a raw food dish while the second restriction workflow favors vegetarian dishes. Comparing the query with the cooking workflow depicted above, one can see that all properties defined by the query are fulfilled in the workflow when considering the subsumption of concepts defined by the taxonomies. Please note that the POQL query in the example does not specify any quantities of ingredients, which are thus ignored during the retrieval and adaptation of workflows.

To compute the query fulfillment between a POQL query $Q = (Q^+, Q^-)$ and a workflow W we propose the following formula:

$$\mathrm{QF}_{poql}(Q, W) = \frac{\mathrm{sim}(Q^+, W) + \mathrm{RS}(Q^-, W)}{2} \tag{1}$$

The similarity between a query workflow Q^+ and a workflow W is determined by a semantic similarity measure $\mathrm{sim}(Q^+, W) \rightarrow [0, 1]$. We use a measure by Bergmann and Gil [3] that applies graph-matching combined with a heuristic search to find the best possible mapping between the nodes and edges of both workflows. Each mapping is rated with local similarity measures by comparing the semantic descriptions of mapped nodes and edges. Following the local-global principle, the overall similarity is computed by aggregating all the local similarity according to an aggregation function. The restriction workflows $q^- \in Q^-$ are compared with a workflow W using a restriction satisfaction function $\mathrm{RS}(Q^-, W) \rightarrow [0, 1]$ that is also based on the semantic similarity measure:

$$\mathrm{RS}(Q^-, W) = 1 - \frac{|\{q^- | q^- \in Q^- \wedge \mathrm{sim}(q^-, W) = 1\}|}{|Q^-|} \tag{2}$$

A restriction component is not satisfied and thus reduces the overall restriction satisfaction if the similarity between both workflows is 1.0. Consequently, matching desired data items or tasks increase the query fulfillment while matching undesired data items or tasks reduce the query fulfillment between a POQL query and a workflow.

2.2 Automatic Workflow Adaptation

If the best matching workflow from the case base does not entirely fulfill the user query, we aim to adapted the workflow according to the restrictions and requirements specified in the query by adding missing desired ingredients/preparation steps and removing undesired ingredients/preparation steps. For this purpose, we proposed several automatic workflow adaptation methods which we now briefly describe (for a more detailed description of these methods we refer to our previous works [14,11,13,12]). Since such adaptation methods usually require a significant amount of domain-specific adaptation knowledge, we additionally developed new methods that allow automatic learning of the required knowledge from the workflow repository. Hence, we distinguish between a learning phase of adaptation knowledge and a problem solving phase in which adaptations are performed. In all approaches, the adaptation of the workflow is performed by chaining several adaptation steps that iteratively transform the retrieved workflow towards an adapted workflow. This process solves an optimization problem aiming to maximize the query fulfillment (as specified in formula 1). This optimization process is implemented as a heuristic search procedure with the goal of achieving an adapted workflow with the highest possible query fulfillment.

The *adaptation operator adaptation* [14] performs individual transformation steps which are denoted in a STRIPS-like manner. An operator consists of two workflow subgraphs that we call *streamlets*: A DELETE-streamlet specifies a workflow fragment to be deleted from the workflow and an ADD-streamlet represents a workflow fragment

to be added to the workflow. The overall adaptation is implemented as a search process that aims to incrementally modify the workflow with the goal of increasing the query fulfillment. The required workflow adaptation operators can be learned from the workflow repository by analyzing pairs of highly similar workflows (selected by using a similarity threshold). For each pair, the difference is determined and workflow operators are generated, whose ADD and DELETE-streamlets basically cover those differences.

The *workflow stream adaptation* [11] decomposes workflows into meaningful sub-workflows named workflow streams in the learning phase. Workflow streams represent valuable adaptation knowledge and can be exchanged during compositional adaptation. More precisely, a workflow stream can be replaced by a stream learned from another workflow that produces the same partial output but in a different manner, i.e., with other task or data items. Workflow streams can only be replaced, if their data nodes indicate that they represent the same kind of sub-workflow. This ensures that replacing an arbitrary stream does not violate the syntactic correctness of the workflow. This adaptation is also implemented as a search process that aims to increase the query fulfillment, but it replaces larger portions of a workflow than the *adaptation operator adaptation*.

The adaptation by *generalization and specialization* [13] first produces a generalized workflow that is structurally identical to the base workflow but its semantic descriptions of task and data items are generalized. The approach generalizes a workflow by considering a set of similar workflows as training samples and selects generalized semantic descriptions from the domain taxonomy as generalization hierarchy. The *generalization* produces a generalized workflow repository which is then used for workflow retrieval. The retrieved workflow is adapted by the *specialization* such that the query fulfillment is maximized.

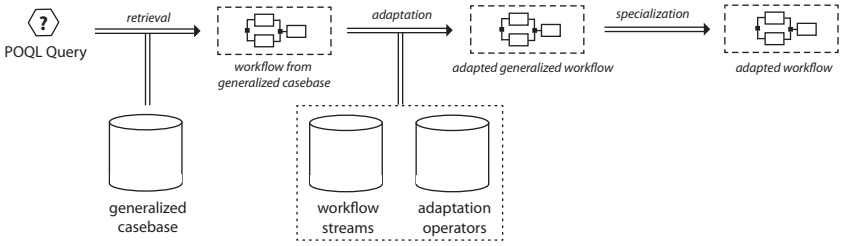

Fig. 3. Combined adaptation approach

The three approaches can be integrated to form a *combined adaptation approach* [12]. This integration involves the actual adaptation process as well as the learning phase. First of all, during the learning phase, adaptation operators and workflow streams can be learned not only from the available specific workflows, but also from workflows resulting from generalization. Thus, we first apply generalization to the workflows in the repository and then we learn adaptation operators and workflow streams from the generalized workflows. The adaptation process itself then uses the three adaptation meth-

ods in combination (see Fig. 3). First, similarity-based retrieval selects the best matching generalized workflow from the generalized case base. Then, the *workflow stream adaptation* replaces entire sub-workflows (e.g., the preparation of the *sandwich sauce*) by matching sub-workflows (e.g., other *sauces*) learned from other workflows. Next, the *adaptation operator adaptation* performs additional modifications on the workflows (e.g., the ingredient *tomato* is replaced by *mushrooms*). Finally, the *specialization* adapts workflows (if necessary) by replacing single generalized data items or tasks with more specific ones by means of the respective taxonomy (e.g. the generalized ingredient *meat* is replaced by *chicken*).

3 Generation of Recipes in Consideration of Nutrients

Generally speaking, there are three basic components in a diet providing food energy to humans: protein, fat, and carbohydrates. Unlike these macronutrients, the micronutrients such as vitamins and minerals are a group of essential nutrients that are present in relatively low amounts in foods. The National Academy of Sciences (NAS) provide comprehensive reports (see [8]) recommending reference values for nutrient intakes for U.S. and Canadian individuals. One aspect of the recommendations includes the *Acceptable Macronutrient Distribution Ranges* for healthy diets, which are expressed as a percentage of total energy intake (i.e. calories). Many online sources as well as literature in the health and fitness sector provide various recommendations for diets (e.g. to count macronutrients instead of calories) and do also propose numerous different ratios of nutrients as a percentage of the daily intake of calories. We follow these recommendations by considering the absolute amounts of desired nutrient contents during the generation of recipes and we take macronutrients as well as micronutrients into account. In order to enable the user to pose a query for a customized dish, we define a *nutrition concept* as follows:

Definition 1. *A nutrition concept* $N = \{(n_1, min_1, max_1), \ldots, (n_k, min_k, max_k)\}$ *is a set of* k *triplets each defining a nutrient* n_i *with a lower and upper limit (*$0 \leq min_i \leq max_i$*) for the absolute amount in grams of the nutrient in a desired dish.*

According to this definition, Table 1 exemplifies some nutrition concepts for different goals. The nutrient contents (except for the last concept *least fat*) are based on the rec-

Table 1. Example of Nutrition Concepts for a single Meal

nutrition concept (goal)	nutrient	desired amount [g]
moderate (lose weight)	protein	16.67
	total lipid (fat)	9.26
	carbohydrate, by difference	45.83
major minerals (healthy diet)	calcium	0.17
	magnesium	0.06
least fat (lose weight)	total lipid (fat)	0.0

ommendations by the NAS and the portion size of one serving of the available recipes. The last nutrition concept is not based on a recommendation and zero fat can be considered as an unrealistic goal. However, it can be suitable to obtain dishes with the least amount of fat. Since this concept defines a minimization problem w.r.t. the creation of a recipe, it is assumed that the need of adaptations for reaching this nutrient content is increased. Please note that, for the sake of simplicity, the example concepts only define the desired nutrient amount as a single value (i.e., $min = max$). However, retrieval and adaptation also support more elaborate queries according to Definition 1.

In the following, this section describes the extensions made to the existing domain ontology, the novel nutrition concept fulfillment, and the assessment of nutrient contents as prerequisites for the nutrient-aware generation of recipes.

3.1 Enriching the Case Representation and Domain Model

In order to consider nutrient contents, the domain ontology containing the taxonomic relations between ingredients is enriched with knowledge about nutrition concepts, nutrients, as well as quantity and unit conversions. We propose a knowledge model similar to the one presented by the *Taaable* project [4] but with the addition of global conversions that are broadly applicable and user-defined nutrition concepts as specified in Definition 1. Figure 4 illustrates the core components of our ontology. According to the definition of nutrition concepts, each concept specifies several nutrients with a minimum and maximum quantity. We implemented the nutrition concepts depicted in Table 1 in the ontology in order to make them available to users and to use them in the evaluation of the approach.

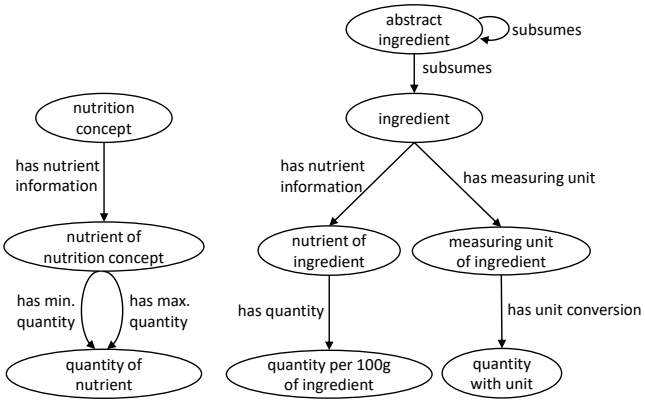

Fig. 4. Domain Ontology with Ingredients, Nutrients, and Nutrition Concepts

The ontology assigns nutrient information and measuring units to each non-abstract ingredient. The nutrient information consists of a name and an indication of quantity per 100 grams. The measuring units of ingredients state different kinds of kitchen measurements such as slices, cups, knife tips, or tablespoons. The ontology further defines

ingredient-specific unit conversions to obtain gram equivalents (e.g. *4 tomato slices* $\widehat{=}$ *100g*) and it also includes global unit mappings and conversions (e.g. *1 tablespoon* $=$ *1 tbsp*, *1 tbsp* $=$ *3 tsp*, or *1 tbsp* $=$ *10g*). In the event where a given ingredient quantity in a (new) recipe cannot be converted into grams since there is no matching ingredient-specific conversion available, the global conversions ensure that either a unit can be mapped to another unit for which a ingredient-specific conversion exists or that a quantity can be directly converted into a gram equivalent. The global conversions are prioritized in order to prefer transformations between semantically similar notations over conversions of different measuring units. For instance, a global conversion such as *1 tbsp* $=$ *10g* has a very low priority and is only applied to an ingredient quantity (given in *tbsp*) in the event where no matching ingredient-specific or higher prioritized global conversion has been found.

For the nutrient-aware creation of recipes, the specific amounts of nutrients contained in each available dish must be considered. Hence, in a pre-processing step, the quantities of ingredients indicated in the recipes are converted into gram equivalents by means of the ontology. This computation differs for concrete and abstract ingredients. For concrete ingredients, the ontology provides unit conversions that are used to convert all ingredient quantities into gram equivalents. Subsequently, the nutrients contained in each ingredient and their masses are determined with the help of the ontology. Abstract ingredients (such as *meat*) are represented as inner nodes in the taxonomy and the ontology does not provide concrete nutrient information. In this event, all subsumed specific ingredients (e.g. *salami*, *ham*, ...) are determined and their amounts of nutrients are assessed. Then, the arithmetic mean of all the masses is calculated as an approximation for the nutrient contents of the abstract ingredient.

3.2 Nutrition Concept Query Fulfillment

Diets that require certain amounts of nutrients cannot be defined in POQL queries. Thus, we combine the existing query with a new nutrition concept query. In such a query, the user may specify an arbitrary number of nutrients according to Definition 1: $N = \{(n_1, min_1, max_1), \ldots, (n_k, min_k, max_k)\}$. In order to evaluate to which extent a given cooking workflow $W \in CB$ from the case base fulfills the desired nutrient distribution, we define a novel *nutrition concept query fulfillment* as a function $\mathrm{QF}_{nc}(N, W) \rightarrow [0, 1]$:

$$\mathrm{QF}_{nc}(N, W) = \sum_{s_i \in N} \frac{\mathrm{sim}(s_i, W)}{|N|} \tag{3}$$

The similarity function $\mathrm{sim}(s, W) \rightarrow [0, 1]$ (see formula 4) assesses a similarity score between a nutrient specification $s = (n, min, max)$ and a workflow W. Moreover, the function $\mathrm{amount}(n, W) \rightarrow \mathbb{R}_0^+$ determines the absolute amount of the nutrient n in a workflow W. The function $\mathrm{maxAmount}(n, CB) \rightarrow \mathbb{R}_0^+$ retrieves a cached value (for normalization purpose) of the nutrient's maximum amount that has ever occurred in a workflow $W \in CB$. Initially, the maximum amounts are set based on the nutrient contents in the original workflows. A maximum amount is updated, whenever

successful adaptations increase the overall nutrient content to a novel highest maximum value. Consequently, if $\text{amount}(n, W) > \text{maxAmount}(n, CB)$ holds true then $\text{maxAmount}(n, CB) \coloneqq \text{amount}(n, W)$.

$$\text{sim}(s, W) = \begin{cases} 1 - \frac{s.min - \text{amount}(s.n, W)}{\max(s.min, \text{maxAmount}(s.n, CB) - s.max)} & \text{if amount}(s.n, W) < s.min, \\ 1 - \frac{\text{amount}(s.n, W) - s.max}{\max(s.min, \text{maxAmount}(s.n, CB) - s.max)} & \text{else if amount}(s.n, W) > s.max, \\ 1 & \text{else.} \end{cases} \quad (4)$$

For the overall similarity computation, a combined query fulfillment $\text{QF}(Q, N, W)$ is computed that contains the conventional POQL query fulfillment $\text{QF}_{poql}(Q, W)$ (see formula 1) as well as the novel *nutrition concept query fulfillment* $\text{QF}_{nc}(N, W)$:

$$\text{QF}(Q, N, W) = \alpha \cdot \text{QF}_{poql}(Q, W) + (1 - \alpha) \cdot \text{QF}_{nc}(N, W) \quad (5)$$

Both criteria are weighted by a parameter $\alpha \in [0, 1]$. This combined query fulfillment measure replaces the conventional POQL query fulfillment. Thus, the retrieval process and subsequent adaptations now aim at optimizing recipes with regard to both criteria. However, this procedure results in a multi-objective optimization problem and thus the approach may not be able to maximize both query fulfillments at the same time. As described in section 2, the best matching workflow w.r.t. the query is retrieved during the retrieval phase. In a linear retrieval, each workflow from the case base is evaluated concerning both query components: $\text{QF}_{poql}(Q, W)$ is assessed by performing a heuristic search for the best possible mapping between the elements in the query and the workflow and $\text{QF}_{nc}(N, W)$ is assessed by examining each nutrient content constraint. The best workflow is then used as a starting point for subsequent adaptations. In the combined adaptation approach, all adaptations are performed in a sequence while each adaptation step aims to maximize the combined query fulfillment analogous to the retrieval phase. For this purpose, the adaptations with workflow streams and the adaptation operators perform a heuristic search to find applicable adaptation steps that produce an adapted workflow with the highest possible query fulfillment. Subsequently, the adaptation by specialization specializes task and data items one by one in consideration of maximization of the query fulfillment. Since the constraint satisfaction is assessed implicitly by similarity scores, neither the retrieval nor the adaptation approaches have to be adjusted to new constraints. However, new types of constraints (e.g. desired percentage of nutrient content) require the definition of measures that are suitable to assess their satisfaction properly.

3.3 Generating Recipes with CookingCAKE

We implemented the approach as an extension to our CookingCAKE system [12], which is part of the CAKE framework[1]. CookingCAKE uses a case base of 70 cooking workflows that describe the preparation of sandwich recipes. The workflows are manually extracted from the cooking portal allrecipes.com and the *Taaable* project [5]. The semantic descriptions of data nodes (representing the ingredients) are extended with quantities

[1] See cake.wi2.uni-trier.de

and units in a semi-automatic process. During this process, all ingredient quantities are normalized to a single portion of a dish and unit terms are standardized. The domain ontology is enriched with data extracted from the *USDA Food Composition Databases*[2] of the United States Department of Agriculture, which have also been used by the *Taaable* project. In particular, these databases contain nutritional information such as name of the nutrient, measuring unit, and the amount per 100 grams of food for many thousands of foods. They also include food-specific weight indications and conversions in grams. In total, our domain ontology includes 33 nutrients for 154 ingredients.

Fig. 5. The CookingCAKE User Query Interface

Figure 5 illustrates the user interface[3] of the prototypical implementation.[4] In the upper third the user can state the first query part, i.e., desired ingredients and preparation steps as well as undesired ones. In the middle, arbitrary nutrition concepts can be created

[2] See https://ndb.nal.usda.gov/ndb/

[3] Please note that this is a simplified user interface that does not allow the definition of workflow fragments. However, a full-featured interface is also available online.

[4] A running demo is available under cookingcake.wi2.uni-trier.de/diet

by specifying nutrients with minimum and maximum amounts as the second query part. At the bottom, the user can set a weight for the fulfillment of both query parts. However, a user may also define a query that consists of only one part.

Table 2. Example of two sandwich recipes created for a query with *croissant* as a desired ingredient and *least fat* as a nutrition concept

	Generation with QF_{poql}	Generation with QF_{poql} and QF_{nc}
ingredients	croissant (0.5 slice) spam (1.5oz) Emmental cheese (1 slice) italian dressing (1 tsp) banana (0.5 slice)	bread (1 piece) zucchini (0.25 piece) red onion (0.25 piece) yellow squash (0.25 cup) red bell pepper (0.25 cup) basil (4 leaf)
nutrients	total lipid (fat) (24g) ...	total lipid (fat) (3g) ...
QF_{poql}	1,0000	1,0000
QF_{nc}	0,9137	0,9879

Table 2 exemplifies two sandwich recipes that are created for a query with *croissant* as a desired ingredient and *least fat* as a nutrition concept. The first sandwich is created using the conventional approach and the second one is created with the nutrient-aware approach. Comparing the POQL query fulfillment, one can see that both sandwiches fulfill the requirement although croissant is generalized to bread in the second sandwich. However, in the domain ontology, croissant is a child node of bread and thus the similarity measure rates the similarity with a score of 1.0. Regarding the nutrition concept fulfillment, the amount of fat is significantly reduced in the second dish.

4 Experimental Evaluation

We now present the evaluation of our approach, which is based on a previous evaluation [10] by Gilbert Müller. The evaluation shows that the adaptation methods summarized in section 2.2 are capable of increasing the POQL query fulfillment significantly while maintaining the overall quality of the workflows at an adequate level. The evaluation also determines a ranking of adaptation algorithms w.r.t. the increase of the POQL query fulfillment. The combined adaptation has the largest leverage followed by the adaptation operators, the adaptation by generalization and specialization, and the workflow stream adaptation. We investigate the following hypotheses in this evaluation:

H1 The approach is able to consider nutrition concepts during the generation of cooking recipes.

H2 The adaptation methods outperform the sole retrieval in terms of the nutrition concept fulfillment.

H3 The ranking of the adaptation methods w.r.t. the increase of the nutrition concept fulfillment is identical to the order obtained w.r.t. the POQL query fulfillment.

4.1 Evaluation Setup

Based on the previous evaluation [10], we now extend the setup in this work. For the evaluation, we use 51 POQL queries for sandwich recipes, which were defined by nine amateur chefs. Based on the domain vocabulary, the users specified arbitrary queries using POQL as well as plain text, which was used for plausibility checks. Furthermore, each POQL query is combined with each of the nutrition concepts listed in Table 1 resulting in 153 queries in total.

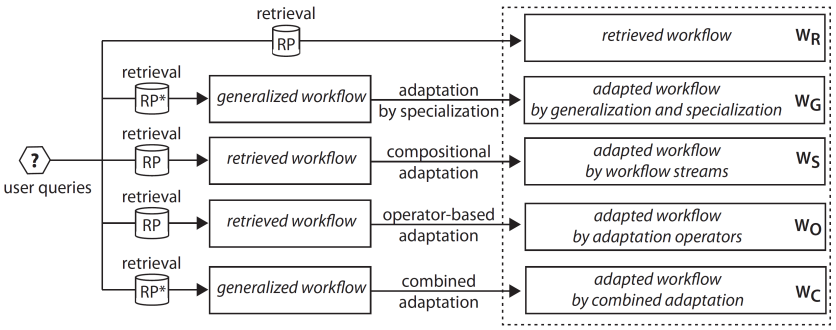

Fig. 6. Evaluation setup taken from [10]

With each of these queries, the experiments are performed using the setup as illustrated by Figure 6. In a first experiment, the best-matching workflow W_R is retrieved from the repository without any adaptation. In a second test run, the retrieval is performed on the generalized workflow repository and a specialization is performed subsequently, resulting in an adapted workflow W_G. In the next two runs, the retrieval is based again on the original repository and the workflow stream adaptation or the adaptation operator adaptation is applied to the retrieved workflow, producing an adapted workflow W_S or W_O, respectively. In a last experiment, the retrieval is performed using the generalized repository. Subsequently, a combined adaptation with workflow streams, adaptation operators, and specialization in that sequence is applied in order to obtain an adapted workflow W_C.

All the experiments described above are performed twice. In a first run, the recipes are generated using the conventional approach without the consideration of nutrition concepts. Consequently, only the POQL query fulfillment QF_{poql} is considered as the optimization criterion during retrieval and adaptation. In the other run, the nutrition concepts are taken into consideration and the combined, equally weighted query fulfillment $\mathrm{QF} = 0.5 \cdot \mathrm{QF}_{poql} + 0.5 \cdot \mathrm{QF}_{nc}$ is used to control the creation process.

4.2 Evaluation Results

Table 3 summarizes the average results over all 153 queries. For each sample type obtained from the experiments (see Figure 6), the average query fulfillments are listed

for the workflows resulted from the generation while considering nutrition concepts. The difference to the query fulfillments obtained from the conventional generation are denoted in brackets. QF_{poql} denotes the POQL query fulfillment, QF_{nc} is the nutrition concept fulfillment, and the combined query fulfillment is stated by QF.

Table 3. Evaluation results: Average values over all queries

	query fulfillment (difference to conventional approach)		
sample type	QF_{poql}	QF_{nc}	QF
W_R	0.8285 (-0.0155)	0.9610 (0.0534)	0.8948 (0.0190)
W_G	0.8639 (-0.0132)	0.9761 (0.0816)	0.9200 (0.0342)
W_S	0.8503 (-0.0122)	0.9667 (0.0754)	0.9085 (0.0316)
W_O	0.8906 (-0.0060)	0.9694 (0.0859)	0.9300 (0.0400)
W_C	0.9295 (-0.0035)	0.9831 (0.0867)	0.9563 (0.0416)

Comparing the average query fulfillments over all queries and samples between the nutrient-aware approach and the conventional approach, one can see that the nutrient-aware approach is able to increase the fulfillment of nutrition concepts about 8 percentage points while QF_{poql} is slightly lower with an decrease of about 1 percentage point. All differences in the table are statistically significant (except for the difference for the combined adaptation W_C w.r.t. QF_{poql}) as determined by a paired t-test ($p < 0.005$). Thus, hypothesis H1 can be confirmed.

Regarding hypothesis H2, we already investigated in previous work [10] the capability of the adaptation methods to optimize QF_{poql}. The ranking of the adaptation methods w.r.t. the increase QF_{poql} can be confirmed by this evaluation and the combined adaptation W_C does also outperform the other approaches. Especially the fulfillment of W_C is about 10 percentage points higher than the fulfillment of W_R. Comparing QF_{nc} between W_C and W_R, it can be seen that the fulfillment is also increased about 2 percentage points. The application of the sole adaptation methods does also produce adapted workflows (W_G, W_S, and W_O) that have a higher QF_{nc} than the retrieved workflow W_R. Consequently, hypothesis H2 is confirmed, too.

When looking closer at the nutrition concept query fulfillments produced by the different adaptation approaches, one can see that each adaptation method slightly increases the fulfillment and that the order assumed for this leverage in hypothesis H3 does hold true for the most part. As predicted, the combined adaptation has the highest potential to optimize the nutrient distribution as desired by the user. In this study, QF_{nc} is increased about 2.2 percentage points. The adaptation by generalization and specialization reached the second highest increase with 1.5 percentage points and outperforms (in contrast to QF_{poql}) the adaptation operators, which achieved an increase of about 0.8 percentage points in average. The adaptation with workflow streams performed slightly worse with an increase of about 0.6 percentage points. Based on the results, we assume hypothesis H3 at least to be partially confirmed.

The ranking observed can be explained by the function principle of the adaptation approaches. Concerning the generalization and specialization no structural changes are applied to the workflows. However, single ingredients can be replaced by semantically similar ones. The operator adaptation is suitable to perform more fine-grained adaptations than the workflow stream adaptation, since its workflow fragments are normally smaller. However, in both adaptations, it may happen that the substitution of workflow fragments also inserts or removes ingredients with nutrients that do not comply with a given nutrition concept.

5 Conclusions and Future Work

This paper presents a means for the consideration of nutrient contents during the creation of recipes. The definition of nutrition concept queries enables the user to specify desired dishes with certain minimum and maximum amounts of nutrients. The automatic generation of recipes provides customized solutions for the user by optimizing the recipes w.r.t. the combined query (consisting of the POQL and nutrient content component). Maximizing the combined query fulfillment is a multi-objective optimization problem that can be solved adequately by our approach. The experimental evaluation demonstrated that each adaptation method is able to optimize the workflows w.r.t. the POQL query while also optimizing the fulfillment of the nutrient contents. While the single adaptation methods only achieve a rather slight increase, the combined adaptation approach is able to significantly elevate the fulfillment of both query components. This result does also confirm the results obtained from a previous evaluation [16] that investigated the performance of the combined adaptation with a different combined query fulfillment consisting of the POQL query fulfillment and a complexity-aware query fulfillment. Consequently, we have shown that changing the underlying similarity measures for a newly composed query type ensures that retrieval and adaptation are able to handle multiple, potentially conflicting constraints by optimizing the overall query fulfillment. Thus, we assume that this framework will be applicable beyond recipe generation, which we would like to investigate in future work.

Regarding the cooking domain, future work could extend the definition of nutrition concepts to support other common specifications of nutrient contents such as the desired macronutrient distribution expressed as a percentage of the daily intake of calories. Furthermore, the adaptation of ingredient quantities should be integrated in the creation process of recipes. By this means, we assume that the nutrients can be better adapted according to the user query since in the current approach only fixed amounts of ingredients (normalized to produce a single serving of a dish) can be substituted by the adaptations. The current repository does only contain 70 cooking workflows. Hence, it is desirable to have some more recipes that comply with certain nutrient distributions and thus are particularly suitable for this approach.

Acknowledgments. This work was funded by the German Research Foundation (DFG), project number BE 1373/3-3.

References

1. Aamodt, A., Plaza, E.: Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Communications **7**(1), 39–59 (1994)
2. Ballesteros, M., Martın, R., Dıaz-Agudo, B.: JADAWeb: A CBR System for Cooking Recipes. In: Bichindaritz, I., Montani, S. (eds.) Case-Based Reasoning. Research and Development, ICCBR 2010. LNCS, vol. 6176, pp. 179–188. Springer (2010)
3. Bergmann, R., Gil, Y.: Similarity Assessment and Efficient Retrieval of Semantic Workflows. Inf. Syst. **40**, 115–127 (2014)
4. Blansché, A., Cojan, J., Dufour-Lussier, V., Lieber, J., Molli, P., Nauer, E., Skaf-Molli, H., Toussaint, Y.: Taaable 3: Adaptation of ingredient quantities and of textual preparations. In: Bichindaritz, I., Montani, S. (eds.) ICCBR 2010 Workshop Proceedings. pp. 189–198 (2010)
5. Cordier, A., Lieber, J., Molli, P., Nauer, E., Skaf-Molli, H., Toussaint, Y.: WIKITAAABLE: A semantic wiki as a blackboard for a textual case-base reasoning system. In: Lange, C., Schaffert, S., Skaf-Molli, H., Völkel, M. (eds.) 4th Semantic Wiki Workshop (SemWiki) at the 6th ESWC, 2009. CEUR Workshop Proceedings, vol. 464. CEUR-WS.org (2009)
6. Hanft, A., Newo, R., Bach, K., Ihle, N., Althoff, K.D.: CookIIS - A successful Recipe Advisor and Menu Creator. In: Mointani, S., Jain, L.C. (eds.) Successful Case-based Reasoning Applications - 1. Studies in Comput. Intell., vol. 305, pp. 187–222. Springer (2010)
7. Herrera, P.J., Iglesias, P., Romero, D., Rubio, I., D-Agudo, B.: JaDaCook: Java Application Developed and Cooked Over Ontological Knowledge. In: Schaaf, M. (ed.) The 9th ECCBR 2008, Workshop Proceedings. pp. 209–218 (2008)
8. Institute of Medicine: Dietary Reference Intakes for Energy, Carbohydrate, Fiber, Fat, Fatty Acids, Cholesterol, Protein, and Amino Acids. The National Academies Press, Washington, DC (2005)
9. Minor, M., Montani, S., Recio-Garcia, J.A.: Process-oriented case-based reasoning. Information Systems **40**, 103 – 105 (2014)
10. Müller, G.: Workflow Modeling Assistance by Case-based Reasoning. Ph.D. thesis, University of Trier (2018)
11. Müller, G., Bergmann, R.: Workflow Streams: A Means for Compositional Adaptation in Process-Oriented CBR. In: Lamontagne, L., Plaza, E. (eds.) Case-Based Reasoning Research and Development, ICCBR 2014. LNCS, vol. 8765, pp. 315–329. Springer (2014)
12. Müller, G., Bergmann, R.: CookingCAKE: A Framework for the Adaptation of Cooking Recipes Represented as Workflows. In: Kendall-Morwick, J. (ed.) Workshop Proceedings from (ICCBR 2015). CEUR, vol. 1520, pp. 221–232. CEUR-WS.org (2015)
13. Müller, G., Bergmann, R.: Generalization of Workflows in Process-Oriented Case-Based Reasoning. In: Russell, I., Eberle, W. (eds.) Proceedings of FLAIRS 2015. pp. 391–396. AAAI Press (2015)
14. Müller, G., Bergmann, R.: Learning and Applying Adaptation Operators in Process-Oriented Case-Based Reasoning. In: Hüllermeier, E., Minor, M. (eds.) Case-Based Reasoning Research and Development, ICCBR 2015. LNCS, vol. 9343, pp. 259–274. Springer (2015)
15. Müller, G., Bergmann, R.: POQL: A New Query Language for Process-Oriented Case-Based Reasoning. In: Bergmann, R., Görg, S., Müller, G. (eds.) Proceedings of the LWA 2015. CEUR Workshop Proceedings, vol. 1458, pp. 247–255. CEUR-WS.org (2015)
16. Müller, G., Bergmann, R.: Complexity-Aware Generation of Workflows by Process-Oriented Case-Based Reasoning. In: Kern-Isberner, G., Fürnkranz, J., Thimm, M. (eds.) KI 2017: Advances in Artificial Intelligence. LNCS, vol. 10505, pp. 207–221. Springer (2017)
17. Van Der Aalst, W.M.: Business process management: A comprehensive survey. ISRN Software Engineering **2013** (2013)