# ≪CooCo, what can I cook today? Surprise me.≫

Karen Insa Wolf, Stefan Goetze, and Frank Wallhoff

Fraunhofer Institute for Digital Media Technology IDMT
Marie-Curie-Strasse 2, 26129 Oldenburg, Germany
{insa.wolf,stefan.goetze,frank.wallhoff}@idmt.fraunhofer.de
http://www.idmt.fraunhofer.de

**Abstract.** In this paper a heuristic computer-based approach is described to vary cooking recipes by replacing ingredients. Conceptually, the approach is integrated in a speech dialogue system. The approach is based on a scoring system. The score value is used to rate different ingredients as candidates to substitute a specific ingredient of a recipe. This substitution score depends on different factors: 1) rating of the similarity between the ingredient which has to be replaced and the substitution candidate 2) rating how well the substitution candidate fits the recipe 3) gustatory preferences of the user. The substitution candidate with the highest score is proposed to the user.

**Keywords:** speech dialogue system, cooking coach, recipe variation

## 1 Motivation

The task in the open challenge of the Computer Cooking Contest [1] is a computer-based adaptation of cooking recipes. The present contribution proposes an approach to substitute ingredients of recipes. The approach is integrated in a speech dialogue system, called CooCo (Cooking Coach), introduced in [13]. CooCo is currently being further developed. A speech dialogue system is a suitable framework for this task:

– Speech input and output is a natural and convenient way to interact with technical devices or systems.
– A speech dialogue system is particularly suitable in scenarios in which the user cannot use his or her hands for interaction. Keyboard, mouse or touchscreen are not convenient user interfaces while cooking.
– Assuming a flexible dialogue management, spontaneous utterance of the user (like e.g. ≪Oops, I do not have ...≫) can be processed.
– The user can be involved in a unobtrusive manner to improve the recipe variation result and tailor the recipe to her/his personal gusto.

## 2 Concept of CooCo

CooCo is designed to assist users in different scenarios: The user can ask for recipes while doing the dishes or can get reminders regarding timing and next steps while cooking. Both tasks require a context-based dialogue system including modules for interpreting, planning and re-planning, as well as memorizing and learning. Different approaches to realize a speech dialogue manager exist, e.g. [9]. Lison distinguishes between hand-crafted and statistical approaches and proposes the toolkit OpenDial to combine both [8]. The dialogue manager of CooCo is based on OpenDial [10]. CooCo's assistance while cooking is conceptually based on a dynamic planning module to actively manage the cooking process. This goes beyond simply reading out the cooking steps aloud when the user asks for this [11]. CooCo formulates an action plan considering active and passive time of the user (e.g. cutting vs. simmering) and dependencies of the cooking steps [13]. The recipe advice mode includes generic models of gustatory preferences (e.g. hot or sweet depending on typical amount of ingredients like chili or sugar) which will be adapted based on the feedback of the user. A new feature, presented in this paper, is the variation of the cooking recipes.

## 3 Computer-based variation of cooking recipes

The computer-based variations of cooking recipes addresses topics of artificial intelligence and machine learning approaches. The task to derive the consequences of the substitution of an ingredient on the textual description of the preparation steps requires techniques of natural language understanding, e.g. [2]. Other approaches aim at replacing ingredients, e.g. by randomizing recipe items [3], by using cognitive super computing (based on IBM's computer system WATSON, [6]) or by just enlarging the database (by the help of a community) to find a matching recipe for every combination of ingredients [12].

The approach presented here addresses the replacement of ingredients. Thereby, $I_{db}$ is the set of all ingredients ($I$) of a specific database. A subset $I_{rc} \subseteq I_{db}$ with ingredients, which belong to one recipe, is defined as $I_{rc} = \{i_{rc,1}, \ldots, i_{rc,m}\}$ with maximum number $m$ of ingredients. The subset $I_{sb} = \{i_{sb,1}, \ldots, i_{sb,h}\}$ with $h \leq m$ and $I_{sb} \subseteq I_{rc}$ comprises all ingredients which will be substituted. The food items which are candidates ($C$) to substitute one element of $I_{sb}$ belong to the set $C_{sb} = \{c_{sb,1}, ..., c_{sb,n}\}$ with maximum number $n$ of known food items. The set of the remaining ingredients of the recipe without the elements of $I_{sb}$ is defined as $I_{rm} = I_{rc} \setminus I_{sb}$. The approach is based on the computation of a substitution score $s$ ranging from 0 to 120 indicating the fit of a specific substitution pair $i_{sb,j} \in I_{sb}$ and $c_{sb,k} \in C_{sb}$. The substitution score is based on statistical information derived from a recipe database and general food knowledge. The approach can also be regarded as one module of a case-based reasoning process of a recipe advisor, as it is described e.g. in [7], to include the substitution of ingredients.

## 4 Use cases

The central task in the following two use cases is to propose a tasty recipe based on the user's input by replacing ingredients. The intention of the user differs in the scenarios. Both use cases can be extended by including the question of undesired ingredients. In order to enlarge the number of possible recipe candidates, the proposed recipe variation approach can be applied in this case additionally to substitute undesired ingredients. Users differ in their gustatory preferences, one likes more traditional recipes, while the other is more open to new tastes. To adjust these individual preferences, two user parameters are introduced referred to as experimental levels. The experimental level $e_{cd}$ influences how common or uncommon a substitution candidate should be. The level $e_{cb}$ regulates how common or uncommon the combination of a substitution candidate and all elements of $I_{rm}$ is. For both levels three adjustment steps can be chosen by the user, ranging from 1 = very common to 3 = very uncommon.

### 4.1 Use case 1: ≪Suprise me.≫

Based on one chosen recipe the user asks for a variation of this recipe. A similar scenario would be that the user realizes that one ingredient is missing but s/he still wants to cook the chosen recipe accepting variations. In both cases, CooCo can choose freely possible substitution candidates. In the first case, the ingredient $i_{sb,j}$ is not defined by the user. In the second case, $i_{sb,j}$ is the missing ingredient.

### 4.2 Use case 2: ≪Work with what I have.≫

The user specifies some ingredients $I_{us}$, s/he wants to work with, but no recipe can be found in the database which uses all desired ingredients. The task for CooCo is now to propose one recipe which matches by replacing missing elements ($I_{ms}$) of $I_{rc}$ with those of $I_{us}$. For this scenario, a plausibility check is necessary since not each combination of ingredients presents a suitable option for a recipe.

## 5 CooCo's Recipe Variation Approach

The central aim of the approach is to compute substitution scores $s$ for different substitution candidates of $C_{sb}$ in relation to one element $i_{sb,j}$ of $I_{sb}$. The candidate $c_{sb,k}$ with the highest score is finally proposed to the user. Considering the abbreviations $i_{sb,j} = i$ and $c_{sb,k} = c$ the substitution score $s(i, c)$ is derived as

$$s(i, c) = s_b(i, c) + s_{sp}(i, c) + s_n(i, c) + s_{cd}(c) + s_{cb}(c, i_{rm} | i_{rm} \in I_{rm}), \qquad (1)$$

with $s_b$ as basic substitution score, $s_{sp}$ as special substitution score, $s_n$ as substitution score based on nutrition facts, and $s_{cd}$ and $s_{cb}$ as substitution scores derived from a statistical analysis of the ingredients and their combination frequency based on the recipe database. The derivation of each summand of Eq. 1 is explained in the following. The substitution of more than one ingredient can be done by repeating the algorithm, up to now without considering the results of subsequent substitution steps. As starting point a recipe database with 1.222 recipes is chosen [5]. Additionally, a semantic net is created representing food

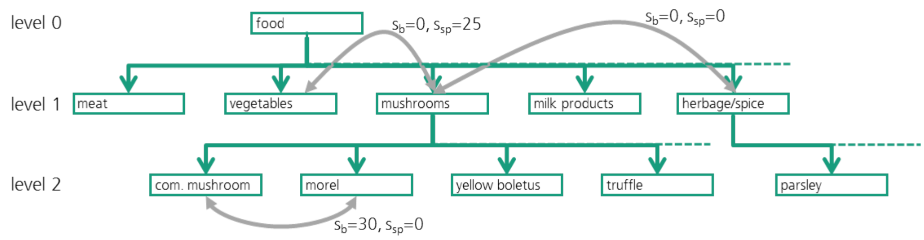


**Fig. 1.** Part of the semantic net with substitution scores $s_b$ and $s_{sp}$.

items in a structured way, cf. Fig. 1. Each item is represented as class within a relationship network of currently 120 classes starting from the level 0 up to 3.

Besides the parent-children constellation different properties of each food class are stored. These properties are grouped in (a) those properties considering only the class itself and (b) those properties related to other classes. For group (a), the following properties are introduced:

**nutrition facts** $n_g$, **with** $g = \{c, f, p, e\}$**:** Nutrition facts are stored for different food classes of level 2 or higher. In the first version of CooCo, the variable $n_c$ contains carbohydrates, $n_f$ fat,$n_p$ protein, and $n_e$ energy per 100 g.

**relative frequency** $f_{cd}$**:** For each food class its relative frequency is derived based on the recipe database. The number of recipes in which the class occurs as ingredient is divided by the total number of recipes. This frequency value describes how common or uncommon a certain ingredient is.

**substitution score** $s_{cd}$**:** Based on the relative frequency $f_{cd}$ the score $s_{cd}$ is derived, considering the experimental level $e_{cd}$. The relative frequencies $f_{cd}$ are classified in five categories. The first category $D_{cd,1}$ contains rarely used and the last category $D_{cd,5}$ frequently used ingredients, assuming $D_{cd} := [0 \ldots 0.005 \ldots 0.01 \ldots 0.03 \ldots\ 0.08 \ldots 1.0]$. The index of the category in combination with the experimental level $e_{cd}$ defines the magnitude of the substitution score $s_{cd}$. This is implemented using a weighting matrix

$$W = \begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{14} & w_{15} \\ w_{21} & w_{22} & w_{23} & w_{24} & w_{25} \\ w_{31} & w_{32} & w_{33} & w_{34} & w_{35} \end{bmatrix} = \begin{bmatrix} -2 & -1 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & -1 & -2 \end{bmatrix} \qquad (2)$$

and by taking one of its elements to derive

$$s_{cd} = 10 w_{e_{cd},p}, \qquad (3)$$

with row number $e_{cd}$ and column number $p$ as index of $D_{cd,p}$.

The group (b) of properties considers the relation between two food classes to indicate how good they can substitute each other or how good they can be combined in one recipe.

**basic substitution score $s_b$:** It is assumed that food classes at a low semantic level (e.g. common mushrooms, morel or truffel in the class "mushrooms", level 2) are similar to each other. Therefore, the score $s_b$ is introduced depending on the level of class in the semantic net, cf. Fig. 1.

**special substitution score $s_{sp}$:** A few explicitly defined scores $s_{sp}$ are stored (e.g. vegetables as substitution candidate for mushrooms or the milk product "tofu" as good substitution candidate for children of the class "meat").

**substitution score based on nutrition facts $s_n$:** It is assumed that one food class of level 2 or higher is a good substitution candidate for another food class if they have similar nutrition facts. Therefore, the similarity factor $f_n$ of two ingredients $i_A$ and $i_B$ is derived based on their nutrition facts $n_g$ as

$$f_{n,g}(i_A, i_B) = \frac{|n_g(i_A) - n_g(i_B)|}{(n_g(i_A) + n_g(i_B))}, \tag{4}$$

with $g = \{c, f, p, e\}$. The mean value $\mu_{f_n}$ and the standard deviation $\sigma_{f_n}$ derived from all $f_{n,g}$ is used as measure of the similarity of the nutrition facts - being aware of the roughness and simplicity of this approach. The substitution score based on nutrition facts is derived as

$$s_n = \min(2/\mu_{f_n}, 15) + \min(2/\sigma_{f_n}, 15). \tag{5}$$

**relative combination frequency $f_{cb}$:** The frequency value $f_{cb}(i_A, i_B)$ expresses how often an ingredient $i_A$ is used in combination with a specific ingredient $i_B$ of $I_{db}$. Therefore, the number of recipes $n_{A\&B}$, in which both ingredients $i_A$ and $i_B$ are included, is determined. This yields $f_{cb}(i_A, i_B) = n_{A\&B}/n_A$. As the denominator usually differs numerically for $f_{cb}(i_A, i_B)$ and $f_{cb}(i_B, i_A)$, the frequencies differ correspondingly. Following this approach, it has to be considered that uncommon ingredients can get $f_{cb}$ values close to 1.0 as $n_A$ is close to $n_{A\&B}$.

**substitution score $s_{cb}$:** The score $s_{cb}$ is derived in a similar way as it is done for $s_{cd}$, but now considering the relative frequency $f_{cb}$ and the experimental level $e_{cb}$. All frequencies $f_{cb}(c, i)$ of the substitution candidate $c \in C_{sb}$ and the ingredients $i \in I_{rem}$ have to be calculated first. The mean value of all these frequencies is then derived as $f_{cb}(c, I_{rm})$. This value is finally assigned to one of the categories $B_{cb,1}$ up to $B_{cb,5}$, heuristically defined as $B_{cb} := [0 \ldots 0.30 \ldots 0.35 \ldots 0.40 \ldots 0.50 \ldots 1.0]$. Using the weighting matrix $W$ from Eq. 2 the score is derived as

$$s_{cb} = 10 w_{e_{cb},q}, \tag{6}$$

with row number $e_{cb}$ and column number $q$ as index of $B_{cb,q}$. The effect is, that if the user wants uncommon combinations, expressed as $e_{cb} = 3$, a set of ingredients with a low $f_{cb}(c, I_{rm})$ get a high score.

## 6 Implementation

The knowledge base containing the recipe database and the semantic net are implemented in Prolog. Standard request functions are implemented, so that recipes including or excluding specific ingredients can be looked up. The approach described above is conceptually tested, further implementation and evaluation is on going work. The procedures to handle both use cases (cf. Section 4) are described in the following based on one test example. The starting point is a simple mushroom soup recipe:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 250 | g | common mushrooms, | 40 | g | butter, | 40 | g | flour, |
| 5 | dl | bouillon, | 5 | dl | milk, | 1 | tb | parsley, minced, |
| - | - | salt, | - | - | pepper | | | |

### 6.1 Procedure for use case 1: ≪Surprise me.≫

In the following description only some examples of the possible substitution candidates are listed.

1. Choose the ingredient with the largest proportion relative to all ingredients of $I_{rc}$ as $i_{sb}$ [*common mushrooms*].
2. Compute $(s_b + s_{sp})$ for all $c_{sb} \in C_{sb}$ [class "mushrooms": *yellow boletus, morel, truffle*; extract of class "vegetables": *red pepper, tomato, cucumber*]
3. Compute $s_n$ for all pairs of $c_{sb}$ and $i_{sb}$ [listing *parsley, cauliflower, morel, yellow boletus* as the one with a high score $s_n$].
4. Derive $s_{cd}$ for all $c_{sb}$.
5. Derive $s_{cb}$ for all $c_{sb}$ with respect to the elements of $I_{rem}$.
6. Sum up $s$ for each pair of $c_{sb}$ and $i_{sb}$ following Eq. 1.

Numerical results for the different experimental levels $e_{cd}$ and $e_{cb}$ are listed in Tab. 1. Parsley is left out as it is already part of the recipe. The results show, that a user with a low $e_{cd}$ of 1 and a medium or high $e_{cb}$ of 2 or 3 will be recommended a tomato soup. In case a very common combination of ingredients is wanted ($s(1,1)$), morel soup is proposed instead. Reason for this is that the recipes with common mushrooms and morel often share the basic combination of ingredients. A user who wants uncommon ingredients in a uncommon combination gets truffle as substitution candidate ($s(3,3)$). Elements of the class mushrooms are mostly preferred. A whole class like "mushrooms" could also be excluded, resulting in recommendations of cauliflower as substitution candidate as a less common ingredient than tomatoes. As the terms $s_{cd}$ and $s_{cb}$ are based on the statistical analysis of the recipe database, the result depends strongly on the size and the quality of the recipe database.

### 6.2 Procedure for use case 2: ≪Work with what I have.≫

In use case 2 the user desires a recipe with the ingredient set $I_{us} = \{$*butter, flour, parsley, bouillon, red pepper*$\}$. Firstly, for all elements of $I_{us}$ the frequencies $f_{cb}$ to each other are checked heuristically: If all $f_{cb} > 0$ and the mean $\mu_{f_{cb}} > 0.1$, then CooCo accepts the set $I_{us}$. Otherwise the dialogue with the user is reopened to ask for other set members. In the example, the set is accepted. The recipe that matches best $I_{us}$ is mushroom soup, based on the simple rule to look for

**Table 1.** Numerical results of use case 1. The result $s(j, k)$ means $s$ based on $e_{cd} = j$ $e_{cb} = k$. The respective candidate with the largest score $s$ is marked in bold letters.

| | y. boletus | morel | truffle | red pepper | tomato | cucumber | cauliflower |
|---|---|---|---|---|---|---|---|
| $s_b + s_{sp}$ | 30 | 30 | 30 | 25 | 25 | 25 | 25 |
| $s_n$ | 29.2 | 29.2 | 29.0 | 9.4 | 5.6 | 12.8 | 33.0 |
| $f_{cd}$ | 0.007 | 0.004 | 0.002 | 0.107 | 0.142 | 0.010 | 0.010 |
| $f_{cb}$ | 0.397 | 0.543 | 0.286 | 0.347 | 0.346 | 0.236 | 0.367 |
| $s(1,1)$ | 49 | **59** | 19 | 44 | 51 | 8 | 48 |
| $s(1,2)$ | 49 | 39 | 39 | 54 | **61** | 28 | 48 |
| $s(1,3)$ | 49 | 19 | 59 | 64 | **71** | 48 | 48 |
| $s(2,1)$ | 59 | **79** | 39 | 24 | 31 | 18 | 58 |
| $s(2,2)$ | 59 | **59** | **59** | 34 | 41 | 38 | 58 |
| $s(2,3)$ | 59 | 39 | **79** | 44 | 51 | 58 | 58 |
| $s(3,1)$ | 69 | **99** | 59 | 4 | 11 | 28 | 68 |
| $s(3,2)$ | 69 | **79** | **79** | 14 | 21 | 48 | 68 |
| $s(3,3)$ | 69 | 59 | **99** | 24 | 31 | 68 | 68 |

those recipes with the smallest number of missing ingredients $I_{ms} = \{i_{ms} | (i_{ms} \in I_{rc}) \wedge (i_{ms} \notin I_{us})\}$. However, red pepper is not part of the original recipe. The algorithm now attends to compute based on Eq. 1 as criteria whether red pepper is a suitable substitution candidate $c_{sb}$ for one of the missing ingredients. Some of the missing ingredients {*pepper, salt, bouillon*} are marked as standard ingredients in the database. CooCo assumes as first guess that they are available also in case the user did not mentioned them explicitly. If this is confirmed by the user, the only missing ingredients left are $I_{ms}$ = {*common mushrooms, milk*}. Considering the experimental levels, the score $s$ is derived for all pairs of $c_{sb}$ with one of the elements of $I_{ms}$. The computation result looking at the substitution pair *red pepper - common mushrooms* differs slightly compared to the result of use case 1 because *milk* is left out in the computation of $s_{cb}$ as it is missing. As consequence, $f_{cb}$ is classified here in $B_{cb,3}$ resulting in $s_{cb} = 0$, independently of $e_{cb}$ as the weight factor in Eq. 6 is zero. Therefore, for all $e_{cb}$ levels the score $s$ is identical to $s(j, 2)$ with $e_{cd} = j$, cf. Tab. 1. The highest score $s = 54$ is reached for $e_{cd} = 1$. Considering a threshold scheme of $[120 \ldots 80]$ (very good), $]80 \ldots 40]$ (acceptable), $]40 \ldots 0]$ (not recommended) for $s$, the substitution pair *red pepper - common mushrooms* is evaluated as "acceptable". In no case it is an option to replace *milk* with *red pepper*, the highest score is $s = 29$. This is reasonable, but a rule should be added in future versions to avoid the substitution of liquid and solid ingredients in any case. This is possible by adding an appropriate property in the semantic net. Milk remains here as missing candidate. Two different last options are possible: (1) Ask the user explicitly whether there is after all a potential substitution candidate. If yes, repeat the procedure. (2) Evaluate how well the missing ingredient could be omitted. Therefore, $s_b + s_{sp} + s_n$ is computed in relation to all ingredients of $I_{rm}$ to get a hint if one of them could make up for the omission by increasing its quantity. In this specific example, the result of 17.5 for *milk* in relation to *butter* is not promising enough to propose this as solution. As final step, the amount of liquid within the recipe ingredients is checked leading here to an increase of the amount of bouillon to recover the original amount of liquid. The final solution with appropriate comments based on the score $s$ is presented to the user.

## 7 Conclusion and future work

A new feature of the currently developed application CooCo is presented. An approach to derive recipe variations by replacing ingredients is introduced. Two different use cases are addressed. The introduced examples provide reasonable results. This first proposed version of the approach has to be further improved and expanded in future work. An evaluation of the substitution results is planned based on feedback of users integrated in the speech dialogue system. The mechanism how to choose the best starting recipe in use case 2 can be ameliorated, including e.g. more information of the gustatory preferences of the user. The present approach prefers recipes with a small number of ingredients. In summary, the approach is a first step for computer-based tasty cooking recipe variations.

**Acknowledgement.** The authors would like to thank the anonymous reviewers for their helpful comments to improve the final version of the paper and their suggestions for future work in this research field.

## References

1. CCC: Computer cooking contest (2015), `cc2015.loria.fr/index.php`, 3.3.2015
2. Dufour-Lussier, V., Le Ber, F., Lieber, J., Nauer, E.: Automatic case acquisition from texts for process-oriented case-based reasoning. Inf Systems 40, 153–167 (2014)
3. Easier Baking, `www.easierbaking.com`, 9.7.2015
4. Gamrad, D.: Modeling, simulation, and realization of cognitive technical systems. Ph.D. thesis, Universität Duisburg-Essen (2011)
5. Herz, J.: Kalorio (2015), `www.kalorio.de/`, 27.2.2015
6. IBM, Inst of Culinary Education: Cognitive Cooking with Chef Watson: Recipes for Innovation from IBM & the Institute of Culinary Education. Sourcebooks (2015)
7. Ihle, N., Newo, R., Hanft, A., Bach, K., Reichle, M.: CookIIS - A Case-Based Recipe Advisor. In: Proc 8th Int Conf on CBR 2009, Seattle, USA, pp. 269-278 (2009)
8. Lison, P.: Structured Probabilistic Modelling for Dialogue Management. Ph.D. thesis, Dep Informatics, University of Oslo (2014)
9. Morbini, F., Audhkhasi, K., Sagae, K., Artstein, R., Can, D., Georgiou, P., Narayanan, S., Leuski, A., Traum, D.: Which ASR should I choose for my dialogue system? In: Proc SIGDIAL 2013, Metz, France. pp. 394 – 403 (2013)
10. Open Dial, `www.opendial-toolkit.net`, 9.7.2015
11. Schäfer, U., Arnold, F., Ostermann, S., Reifers, S.: Ingredients and recipe for a robust mobile speech-enabled cooking assistant for german. In: KI 2013: Advances in Artificial Intelligence, pp. 212–223. No. 8077 in LNCS, Springer (2013)
12. SousChef, `www.acaciatreesoftware.com/`, 9.7.2015
13. Wolf, K.I., Goetze, S., Wellmann, J., Winneke, A., Wallhoff, F.: Concept of a nutrition consultant application with context based speech recognition. In: Proc Workshop Kognitive Systeme 2015, Bielefeld (2015)