

# Implementation of the Healthy Eating Habits Support System Based on User Taste Preferences and Nutritional Balance

Masataka Tokumaru

*Kansai University, Japan, toku@kansai-u.ac.jp*

**Abstract:** This paper proposes a healthy eating habits support system that recommends menu selections based on a user's taste preferences and the requirements of long term nutritional balance. This system is comprised of a nutritional management system (NMS) and a Kansei retrieval system (KRS). NMS adopts a tabu search method to generate a large number of nutritionally balanced menus by combining multiple recipes from a recipe database and stores these menus in a "candidate list." From this candidate list, KRS retrieves menus that are compatible with a user's taste preferences and presents these menus to the user. KRS utilizes Kansei retrieval agents that represent a user's taste preferences and impressions that, subject to a user's evaluation of the presented menus, evolve based on a hybrid model composed of a genetic algorithm and simulated evolution. A simulation using 7,260 actual recipes incorporating 13 types of nutrients demonstrated that the system presented a large number of menus nutritionally balanced over the long term, and predicted a user's taste preferences with more than 80% accuracy after continuous use for eight weeks.

**Keywords:** Eating Habits Support, User's Taste Preference, Nutritional Balance, Interactive Evolutionary Computation.

## 1. INTRODUCTION

It is impractical for people who cook every day for their family to plan a new menu on a daily basis. Thus, they often consult websites when they need new ideas for their menus (Allrecipies.com, CDKitchen.com, FOODILY). The popular recipe retrieval website "Allrecipies" (Allrecipies.com) is accessed by over 24 million unique visitors each month (Allrecipies.com – Newsroom). This


indicates that a large demand for recipe retrieval sites is in evidence. However, contemporary recipe retrieval sites generally retrieve a recipe either by using a keyword or by recommending a popular recipe. In these conditions, a user's taste preferences are not considered, since these sites yield the same retrieval results for all users. In addition, the search results for many recipes do not describe the nutritional details of the recipes. Almost all recipe retrieval sites need a keyword for retrieval and display the nutrients included in a recipe. Therefore, we propose a healthy eating habits support system (HEHSS) that presents recipes based on a user's taste preferences and health needs without requiring the use of keywords.

HEHSS is composed of two main components. The first component is a nutritional management system (NMS) that was introduced in our previous study (Tokumi, Hakamata & Tokumaru, 2013). The second is a Kansei retrieval system (KRS) that was also introduced in another previous study (Hakamata, Tokumi & Tokumaru, 2011). In addition, we employ the "interactive evolutionary computation model," using a Kansei retrieval agent (Tokumaru & Muranaka, 2008) for KRS.

In our previous studies, these two systems were developed independently and their effectiveness was confirmed by numerical simulation. However, HEHSS has not been implemented. Therefore, this study implements HEHSS by integrating NMS and KRS and evaluates its effectiveness, using the real recipe data.

## 2. HEALTHY EATING HABIT SUPPORT SYSTEM

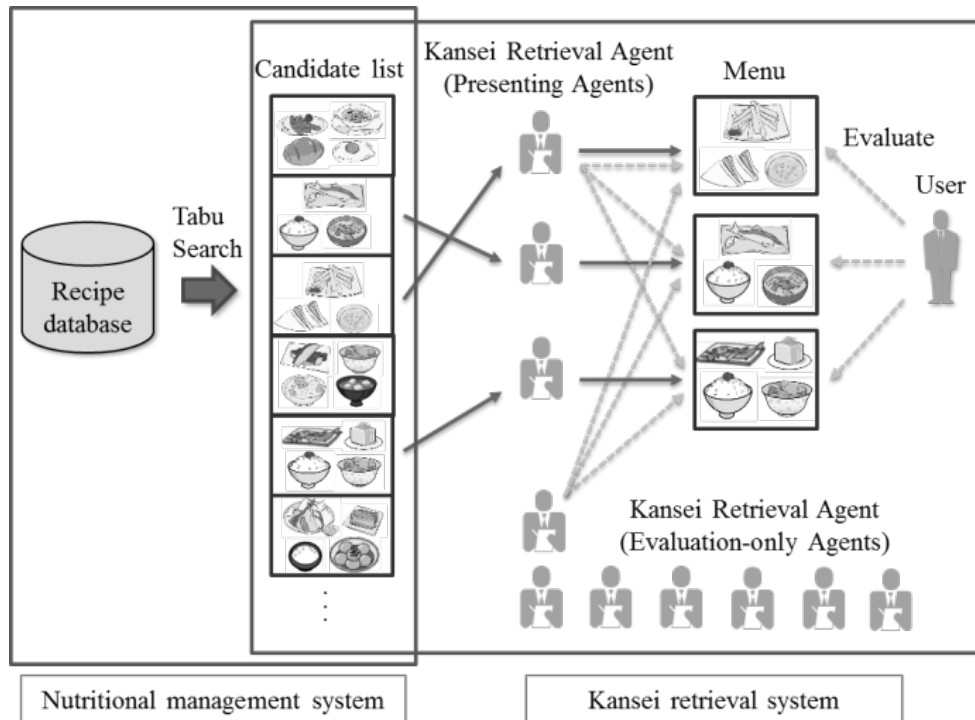
HEHSS considers user taste preferences and health requirements to present recipes. However, when only a single recipe is presented, as shown in Fig. 1, the user must combine the presented recipe with other recipes to create a complete meal. This represents an undesirable burden for the user. Therefore, we propose a system of menu creation that combines multiple recipes for complete meals that are presented to the user, and thus alleviates user burden.

Japanese soup of Cauliflower	
	
Foodstuff	(For one person)
• Cauliflower	62 g
• Carrot	10 g
• Onion	10 g
• Butter	1 g
• Flour	1.5 g
• Miso	6 g
• Soy milk	60 g

**Figure 1:** An example of recipe data used in HEHSS.

HEHSS is comprises NMS and KRS. The structure of this support system is shown in Fig. 2. First, NMS creates nutritionally balanced menus that combine multiple recipes, using the tabu search (TS) method (Glover, 1989 & 1990), and stores the generated menus in a "candidate list." The TS method is a metaheuristics type method and is often used in cases where the problem is one of combinatorial optimization. Next from the candidate list, Kansei retrieval agents (Agents) retrieve

and present menus based on a given user's taste preferences. Agents are used as representations of the user's taste preferences and impressions. The user selects a menu and cooks and consumes the meal. The nutrients the user will consume in this single meal are included in the presentation of the menu selected by the user. Next, the user evaluates the consumed menu, and the Agents evolve using a hybrid model combining a genetic algorithm (GA) and simulated evolution (SimE) scheme (Kling, 1987) on the basis of the user evaluation. GA and SimE refer to a type of metaheuristics that simulates the evolution process of organisms. Through repetition of this process, the taste preferences of Agents evolve to become similar to those of the user. Consequently, an Agent can present the user with a menu that suits their tastes with increasing accuracy.



**Figure 2:** A schematic of the healthy eating habit support system (HEHSS) illustrating NMS on the left and KRS on the right. Agents mediate the retrieval of menus that the user consumes and evaluates; thus, Agent preferences evolve to increasingly correspond to those of the user.

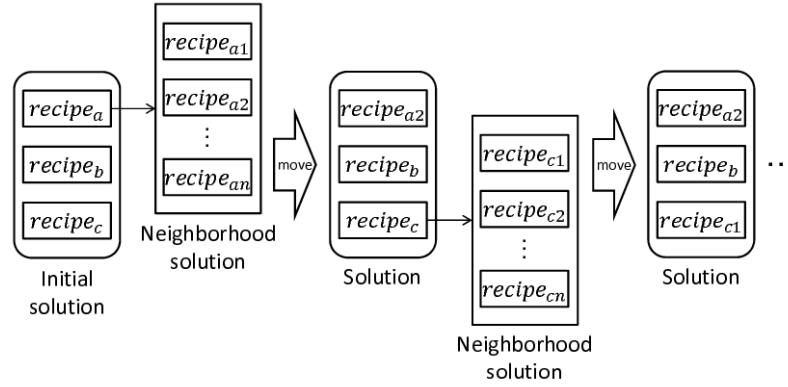
### 3. NUTRITIONAL MANAGEMENT SYSTEM

#### 3.1. Processing

We describe the procedure used by NMS for a case in which the menu is composed of three recipes, as illustrated in Fig. 3. First, NMS randomly selects three recipes,  $recipe_a$ ,  $recipe_b$ , and  $recipe_c$ , from a database. The selected recipes represent the initial solution obtained using the TS method. Next, NMS selects one recipe at random from the initial solution set. Then, NMS forms a neighborhood solutions group based on the selected recipe. The example shown in Fig. 3 indicates that NMS selects  $recipe_a$  and forms a neighborhood solutions group consisting of  $recipe_{a1}$ ,  $recipe_{a2}$ , ...,  $recipe_{an}$  that include nutrients similar to  $recipe_a$ .

Next, NMS evaluates the menu composed of  $recipe_{a1}$ ,  $recipe_b$ , and  $recipe_c$ , and then consecutively substitutes  $recipe_{a1}$  with  $recipe_{a2}$ ,  $recipe_{a3}$ , ...,  $recipe_{an}$  in sequence. NMS similarly evaluates each combination of recipes in sequence. The details of this evaluation are described in Section 3.2. NMS examines the recipe included in the neighborhood solutions group exhibiting the highest evaluation. If the recipe does not exist in the tabu list, NMS forms a new solution. In the

example shown in Fig. 3, this solution is composed of  $recipe_{a2}$ ,  $recipe_b$ , and  $recipe_c$ , because  $recipe_{a2}$  returned the highest evaluation and thus substitutes for  $recipe_a$ . At this time, NMS adds  $recipe_a$  to the tabu list to avoid cycling. While processing TS, menus with an evaluation value larger than the threshold are selected and added to the candidate list if not already present. Standard TS aims to find the single best solution. In contrast, the TS method employed by NMS aims to search numerous menus with evaluation values more than the threshold. Therefore, if the solution does not change, NMS forms a new initial solution and restarts TS. In this case, NMS clears the tabu list (Glover, 1989). NMS repeats the TS process until termination conditions are reached.



**Figure 3:** A schematic of the tabu search (TS) method employed by the nutritional management system (NMS) that seeks to locate numerous menus with evaluation values higher than the established threshold.

In this study, NMS retrieves menus by considering 13 types of nutrient categories, given as Energy, Fat, Saturated fatty acid, Vitamin A, Calcium, Iron, Protein material, Vitamin B1, Vitamin B2, Vitamin C, Alimentary fiber, Kalium, and Sodium chloride equivalent. In “the dietary reference intakes of the Japanese in 2010” (Ministry of Health, Labour and Welfare in Japan, 2010), these 13 types of nutrients are categorized into 5 types according to the preference order of intake. Table 1 shows the preference order of intake of these 13 nutrient categories in detail.

### 3.2. Calculation evaluation

In this section, we describe the method for evaluating a menu by providing a value  $E$ , as expressed by Equation (1).

$$E = \prod_{i=1}^N e_i \quad (1)$$

where  $i$  denotes types of nutrients,  $e$  ( $0 \leq e_i \leq 1.0$ ) denotes evaluation of each nutrient included in the menu, and  $N$  denotes the total nutrient count. NMS considers 13 types of nutrients to create menus; therefore, the value for  $N$  is 13 and NMS considers all nutrients when creating menus. In summary, we cannot overlook the absence of any single nutrient; therefore, we calculate all nutrients in  $E$  by multiplying the values of  $e_i$ . This allows for assessment and correction of excessive or deficient nutrient intake.

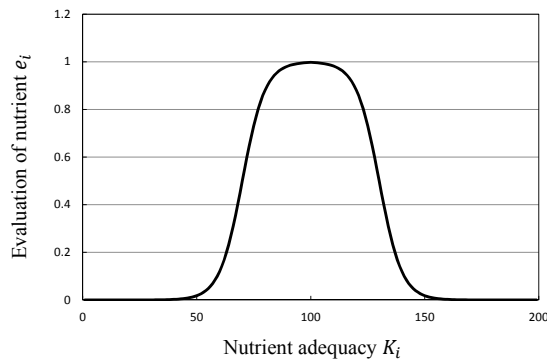
The 13 nutrients considered by NMS are shown in Table 1. The necessary intake for each nutrient is defined by the “the dietary reference intakes of the Japanese in 2010,” and we configured the nutrient evaluation functions to calculate  $e_i$  on the basis of the features of each individual nutrient.

For example, Energy, given as Type 1 in Table 1, has very specific intake requirements, and we

evaluated its intake using the function shown in Fig. 4. We define this function as a Type 1 function.

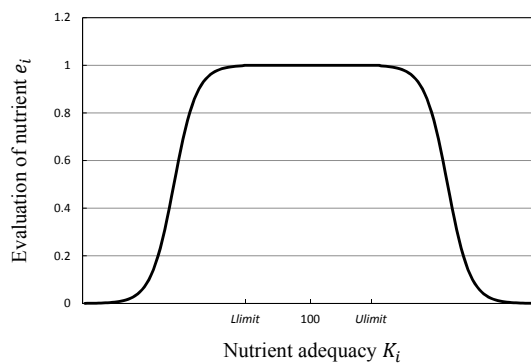
**Table 1:** List of nutrients

evaluation function	Kind of Nutrient
Type1	Energy
Type2	Fat, Saturated fatty acid
Type3	Vitamin A, Calcium, Iron
Type4	Protein material, Vitamin B1, B2, C, Alimentary fiber, Kalium
Type5	Sodium chloride equivalent



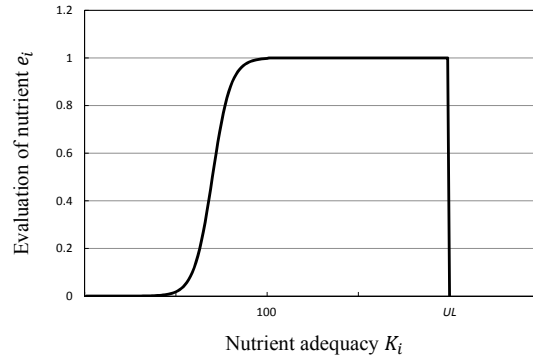
**Figure 4:** Evaluation function used to evaluate menu Energy requirements, given as Type 1 in Table 1.

Similarly, Fat and Saturated fatty acid also have specific intake requirements, which must be satisfied almost exactly. In this context, “almost” means approximately  $\pm 10\%$  of the required intake. We use the evaluation function shown in Fig. 5 to assess Fat and Saturated fatty acid intake, and define this function as a type 2 function. In Fig. 5, *Llimit* is the lower intake requirement and *Ulimit* is the upper intake limit. If a user intakes less than *Llimit* or more than *Ulimit*, the value of  $e_i$  is decreased.



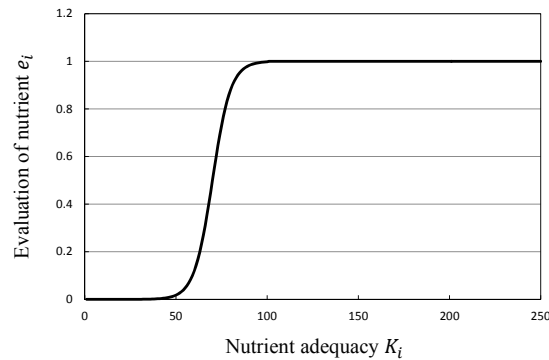
**Figure 5:** Evaluation function used to evaluate menu Fat and Saturated fat requirements, given as Type 2 in Table 1, where *Llimit* is the lower intake requirement and *Ulimit* is the upper intake limit.

To evaluate the nutritional intake of Vitamin A, Calcium, and Iron, we use the evaluation function shown in Fig. 6 and define this function as a Type 3 function. In Fig. 6, *UL* denotes the “Tolerable Upper Intake Levels” (Ministry of Health, Labor and Welfare, 2010) configured for each nutrient’s intake. Unlike *Ulimit* in Fig. 5, if a user intakes a greater quantity of these nutrients than *UL*, the value of  $e_i$  becomes 0.



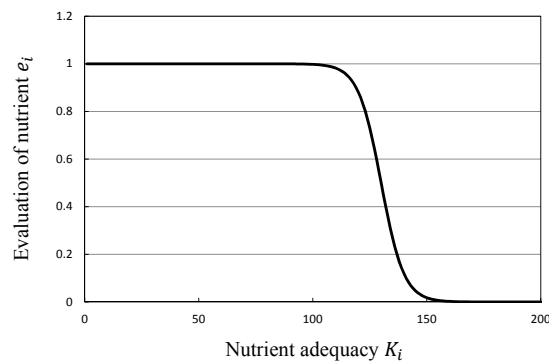
**Figure 6:** Evaluation function used to evaluate menu Vitamin A, Calcium, and Iron requirements, given as Type 3 in Table 1, where the upper limit  $UL$  is configured for each nutrient's intake.

For Protein material, Vitamin B1, Vitamin B2, Vitamin C, Kalium, and Alimentary fiber, an upper nutrient intake limit is not configured. However, the lower limit is configured. Thus, we evaluate these nutrients using the evaluation function shown in Fig. 7 and define this function as a Type 4 function.



**Figure 7:** Evaluation function used to evaluate menu Protein material, Vitamin B1, Vitamin B2, Vitamin C, Alimentary fiber, and Kalium requirements, given as Type 4 in Table 1.

For the Sodium chloride equivalent, the lower limit is also not configured. However, the upper limit is configured. Thus, we use the evaluation function shown in Fig. 8 to assess the nutritional intake of the Sodium chloride equivalent and define this function as a Type 5 function.



**Figure 8:** Evaluation function used to evaluate menu Sodium chloride equivalent requirements, given as Type 5 in Table 1..

By configuring the evaluation functions for  $e_i$  on the basis of the features of each nutrient,  $e_i$  can be calculated using equation (2).

$$e_i = f_j(K_i) \quad (1 \leq j \leq 5) \quad (2)$$

where  $K_i$  denotes the percentage of the required nutrient intake in a meal and  $f_j(K_i)$  denotes the evaluation functions that normalize  $K_i$  into a numerical value from 0 to 1.0. The evaluation function type is represented by the numerical index  $j$  (for  $j = 1-5$ ), which corresponds to the 5 function types.  $K_i$  is expressed by Equation (3), where  $Sum$  represents the sum of the nutrients included in a menu and  $NesIntake_i$  denotes the nourishment a meal must provide.

$$K_i = \frac{Sum_i}{NesIntake_i} \times 100\% \quad (3)$$

### 3.3. Considering nutrient balance for the long-term

The daily nutrient requirement for a person was configured using “the dietary reference intakes of the Japanese in 2010.” However, it is difficult for users to obtain the configured nourishment every day. Therefore, NMS aims to provide for intake of the configured nourishment over the long term.

First, NMS stores a menu chosen by the user that is presented by KRS. Next, *IntakeError* is calculated using Equation (4).

$$IntakeError_{i,t} = FactIntake_{i,t} - NesIntake_{i,t} \quad (4)$$

where *IntakeError* denotes the accumulation of the intake error, *FactIntake* represents the nourishment included in the selected menu, *NesIntake* is the nourishment that the user needs to obtain in a given meal,  $i$  represents any one of the 13 types of nutrients, and  $t$  is the number of meals.  $NesIntake_{i,t+1}$  is calculated using Equation (5).

$$NesIntake_{i,t+1} = BaseIntake_i - IntakeError_{i,t} \quad (5)$$

where *BaseIntake* shows the standard value of intake, as specified according to “the dietary reference intakes of the Japanese in 2010.” *IntakeError* is improved by renewing *NesIntake*. In our study, if  $e_i$  (discussed in Section 3.2) is 1.0, NMS does not calculate *IntakeError*.

## 4. KANSEI RETRIEVAL SYSTEM

### 4.1. Processing

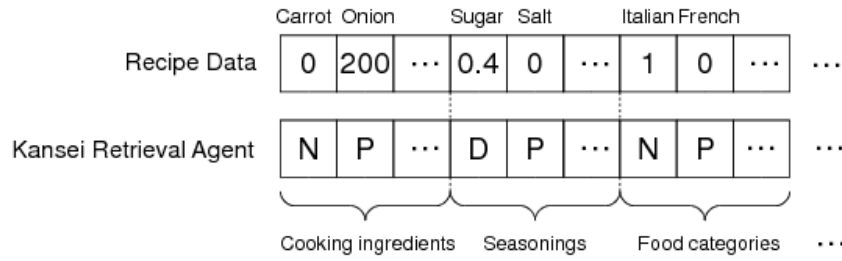
We describe the procedure employed by KRS using Agents. These Agents are models that learn a user’s preferences and use this information to retrieve menus from the candidate list where NMS stores menus. Agents are optimized by consulting user evaluations. KRS uses two types of Agents, given as presenting Agents and evaluation-only Agents. Each presenting Agent selects a menu from the candidate list on the basis of its knowledge of the user’s preferences. The Agent then recommends the selected menu to the user, and the user evaluates the recommendation. The presenting Agent also evaluates menus recommended by other presenting Agents, using their knowledge of the user’s preferences. The evaluation-only Agents, using their knowledge of the user’s preferences, only evaluate menus recommended by the presenting Agents even though they do not themselves present any menus to the user.

Presenting Agents and evaluation-only Agents have the same structure and evaluate menus recommended to the user by presenting Agents. Then, an Agent whose evaluation values are similar to those of the user gets recognized as an appropriate Agent fitting the user. In the optimizing process, the more appropriate Agents become new presenting Agents and recommend menus to the user

## 4.2. Kansei Retrieval Agent

An Agent evaluates a menu on the basis of items contained in the menu. Recipe data include cooking ingredients such as carrots and onions, seasoning such as sugar and salt, food categories such as Italian or French, taste such as sweet or hot, and cookery methods such as fry or boil. The values of cooking ingredients and seasonings, which are real-valued attributes, take the number of grams if the item is used in a recipe, otherwise 0. The values of other categories take a value of 1 if the recipe belongs in the category, otherwise 0.

The Agent codes for the same elements in correspondence with recipe data. An example of Agent and recipe coding is shown in Fig. 9. Each element of the Agent takes the values of P (Positive), N (Negative), or D (Don't care) in correspondence with the recipe data. For example, Fig. 9 indicates that, when an Agent recognizes that the user dislikes carrot, the Agent element corresponding to the recipe element of carrot takes the value of N, and the Agent seeks recipes that exclude carrot. Similarly, when the Agent recognizes the user likes onion, the value for the onion element is P, and the Agent seeks recipes that include onion. A value of D means that the user is unconcerned about the element and, in Fig. 9, the Agent does not discriminate between recipes on the basis of the sugar element.



**Figure 9:** Coding of recipe data and that of the Kansei retrieval agent (Agent), showing a correspondence between the elements of the two groups.

When the Agent searches for a menu in the candidate list that is to be presented to the user, the Agent evaluates menus and assigns a grade to it. The overall grade of the menu is computed by the sum of the evaluated values of recipes that are contained in the menu. The Agent evaluates recipes using Equation (6).

$$V = \sum_{i=1}^S (P_i \times W_i) \quad (6)$$

where  $S$  is the number of elements,  $P_i$  is the value of the fundamental point listed in Table 2, and  $W_i$  is the weight. If an Agent element is denoted by P, and the element of the recipe is denoted by 200, such as is the case of the onion element given in Fig. 9, the fundamental point takes a value of +1.

**Table 2:** F Fundamental points based on correspondence between Agent and recipe elements.

Kansei Retrieval Agent	Recipe	Fundamental Point
P	0	0
P	1	+1
N	0	0
N	1	-1
D	0	0
D	1	0



Similarly, if an element of the Agent is denoted by  $N$ , and the element of the recipe is denoted by 1, which is in the case of the Italian element in Fig. 9, the fundamental point takes a value of  $-1$ . The weight is a degree that indicates the impression an element conveys for a user. In general, favorite and less favorite foods do not change. However, likes and dislikes of a particular food category might change according to mood. Thus, the weight of a desirable food category needs to be higher than that of other items. An Agent estimates that higher the evaluation value of a menu greater is the user preference.

### 4.3. Fitness Function

In KRS, the Agent evolves using a hybrid model combining a GA and SimE, which is described in Section 4.4. The Agent has a fitness function to determine how similar its knowledge of the user's preference is to the user's actual preference. The closer the user's evaluation is to the Agent's evaluation, the higher the fitness given to the Agent. The fitness of the Agent is computed using Equation (7).

$$F = \sum_{i=1}^{N_m} \sum_{j=1}^{N_r} |V_{ij}^{agent} - V_{ij}^{user}| \quad (7)$$

where  $N_m$  is the sum of the presented and recoded menus,  $N_r$  is the number of recipes included in the menu, and  $V_{ij}^{agent}$  and  $V_{ij}^{user}$  are the Agent's and user's evaluations given by Equation (6) for recipe  $j$  of menu  $i$ , respectively.

### 4.4. Hybrid IGA-SimE Learning Technique

KRS uses a hybrid IGA-SimE learning technique proposed in our previous study to evolve the Agents. This technique combines an interactive genetic algorithm (IGA) (Smith, 1991) and SimE. IGAs are among the methods used in interactive evolutionary computing. While a conventional GA uses an evaluation function to determine fitness, IGA obtains fitness by consulting user evaluations.

SimE mimics biological evolution, much like GA, but SimE does not share the problem of converging to localized solutions, as is present in standard GAs. Every element in a SimE scheme is considered to be an individual with some level of fitness. The basic procedure performed by SimE consists of three steps: evaluation, selection, and allocation.

Fig. 10 shows a flowchart for the hybrid IGA-SimE learning technique. Although the method is based on IGA, it uses SimE to process mutations.

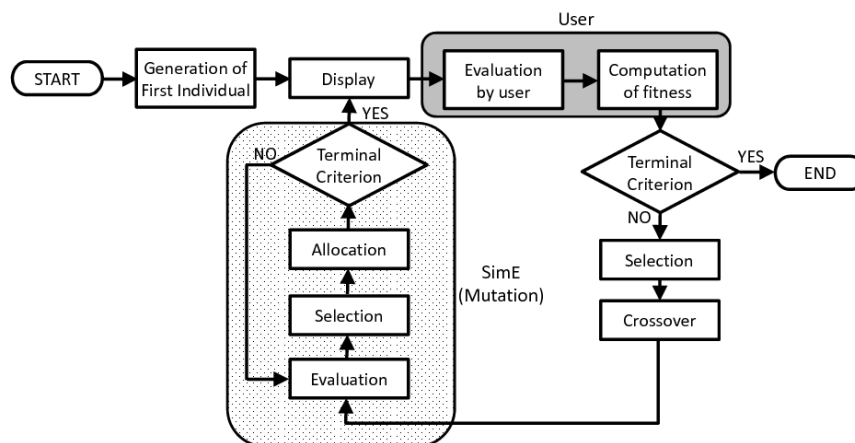


Figure 10: IGA-SimE flowchart

## **5. SIMULATION**

### **5.1. Outline**

In this section, we discuss the simulation study performed to examine the effectiveness of the HEHSS, using real recipe data. This simulation adopted a user interface that allows actual users to easily evaluate menus recommended by the system. This simulation used a simple method in which users rearranged menus in the order of their preference.

For the simulation, we stored 7,260 real recipes utilizing all 13 types of nutrients into the database, 7,048 of which were from two well-liked recipe retrieval websites in Japan (COOKPAD & Calorepi) and 212 of which were from recipe books (TANITA CORPORATION, 2010). We also selected the most popular 100 cooking ingredients affecting user preference for coding Agents and recipes by eliminating cooking ingredients, for example, snails and peanuts, that are seldom used in recipes, or combining several cooking ingredients, such as green peppers and red bell peppers, that give users a similar taste. Thus, the number of elements for Agents and recipes was 100.

### **5.2. Simulation Conditions**

The simulation assumes that HEHSS continues to recommend menus to a user for over 60 days. In this case, the user needs to evaluate menus over 180 times, using the system three times a day for breakfast, lunch, and dinner. Since it is difficult to perform this series of interactions with actual users, in the simulation we used an Agent that has a particular preference to evaluate menus instead of an actual user. We call this Agent a quasi-user. In this simulation, the quasi-user had P, N, or D values for each cooking ingredient, which was randomly selected under the condition where the quasi-user liked 25% of the 100 total cooking ingredients, disliked 25%, and had no preference regarding the remaining 50%. Therefore, the number of elements for the quasi-user having P values was 25, N values was 25, and D values was 50.

In this simulation, NMS retrieved menus by considering that the quasi-user ate a larger portion at lunch, providing for approximately 40% of the daily nutritional requirement, and breakfast and dinner each provided 30%. Each meal had a menu including three recipes, which was recommended by the system and a portion of white rice. NMS held up to 30,000 menus in the candidate list.

KRS utilized 200 Agents in the simulation, five of which were presenting Agents and 195 of which were evaluation-only Agents. Thus, KRS recommended five menus from the candidate list for each meal. The quasi-user evaluated and rearranged those menus in its preferred order.

The hybrid IGA-SimE learning technique requires a sufficiently high number of menus that have been evaluated by a user because the accuracy of the Agent fitness evaluation affects the performance of the optimization positively. Therefore, the quasi-user evaluated 100 menus randomly selected before performing IGA-SimE, and, after this pre-evaluation, KRS optimized the Agents, using evaluation data of those 100 menus and adding evaluation data of new menus presented to the quasi-user in each recommendation. While performing IGA-SimE, the quasi-user repeated menu evaluation 210 times, which means that the system had been, in effect, used for 70 days.

### **5.3. Simulation Results**

Tables 3 and 4, and Fig. 11 show the simulation results. The data presented in Table 3 confirm that NMS could adjust the user's nutritional intake over the course of 70 days while satisfying all 13 nutritional needs.

Fig. 11 shows the average learning performance of KRS in 100 simulation trials, in each of which a quasi-user with different preferences was used. The results show the difference in the codes of the Agent whose fitness is the best and those of the quasi-user. Two types of errors are used, as described below.

1) Errors using the complete consistency criterion

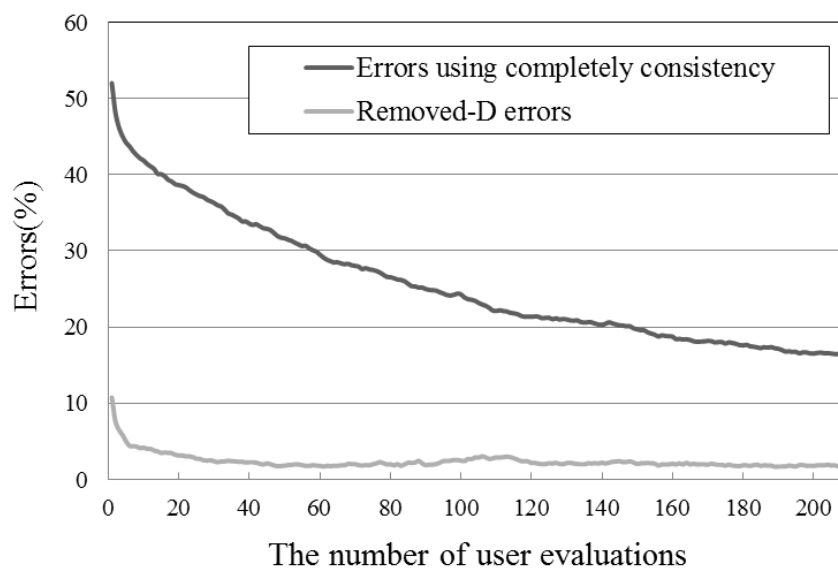
Errors using the complete consistency criterion measure the differences of identical elements between the Agent and quasi-user. These errors are the output of the number of elements that are not completely consistent.

2) Removed-D errors

Removed-D errors output the number of consistent elements unless the element of the Agent and quasi user is D.

**Table 3:** Comparison of nutritional needs and average quantity of nutrients included in a day

Name	nutritional needs during one day	FactIntake for 3 meals
Energy	2,120 — 3,180 kcal	2,643 kcal
Protein material	50.0 g —	102.0 g
Fat	58.9 — 88.3 g	73.5 g
Saturated fatty acid	13.3 — 20.6 g	17.0 g
Vitamin A	600 — 2,700 µg	831 µg
Vitamin B <sub>1</sub>	1.2 mg —	1.6 mg
Vitamin B <sub>2</sub>	1.3 mg —	1.7 mg
Vitamin C	85.0 mg —	257.2 mg
Calcium	650 — 2,300 mg	756 mg
Iron	6.0 — 50.0 mg	13.6 mg
Alimentary fiber	19.0 g —	32.1 g
Sodium chloride equivalent	— 9.0 g	7.7 g
Kalium	2,500 mg —	5,303 mg



**Figure 11:** Output of errors

The errors are reduced as the user progressively evaluates the menus. This also confirms that the system was able to learn about 75% of the user's preferences after 100 evaluations (by reducing the errors to approximately 25%) and over 80% after 200 evaluations. Moreover, the removed-D errors are less than 5% after 20 evaluations. This result indicates that the system could learn the user's preferences rapidly and predict which cooking ingredients the user liked or disliked.

**Table 4:** Example of menus selected to the user

Date	Menu
Day 16 : Breakfast	①Simmered radish ②Lettuce salad ③Stir-fried spinach and pork with oyster sauce
Day 16 : Lunch	①Vinegared vermicelli ②Thai rice ③Fried burdock and chikuwa
Day 16 : Dinner	①Kimchi fried rice ②Japanese style deep-fried chicken ③Fried Japanese mustard spinach and radish Kimchi
Day 17 : Breakfast	①vegetable gelatin ②Fried vegetables ③Okra and grated yam with grated radish
Day 17 : Lunch	①Apple yogurt cake ②Grilled eggplant with grated radish ③Simmered sweet potato and apple
Day 17 : Dinner	①Fried potato and beef ②Japanese mustard spinach and eggplant and bean sprouts with grated radish ③Wholesale ponzu sauce of grilled eggplant
Day 18 : Breakfast	①Plum cake ②Spicy salad of Bean Sprout and Japanese mustard spinach ③Fried eggplant and Japanese mustard spinach
Day 18 : Lunch	①Simmered eggplant ②Sesame salad of steamed chicken and broccoli ③Grilled eggplant and green pepper with onion and bonito
Day 18 : Dinner	①Apple pastry ②Stir-fried sweet and spicy radish leaf ③Fried egg and Chinese chive and whitebait
Day 19 : Breakfast	①Pork and Japanese mustard spinach with grated radish ②Seasoned ground meat rice ③mushrooms with grated yam
Day 19 : Lunch	①Curry pilaf ②Bread of steamed spinach ③Hamburger of spring vegetables
Day 19 : Dinner	①Fried pork and potatoes ②Chicken nugget with curry sauce ③Japanese mustard spinach with mustard

Table 4 shows examples of menus presented to the user for each meal after system learning for 15 days, which means after 45 evaluations. In this example, the quasi-user preferred radish and eggplant. The result shows that HEHSS learned the user's preference and often recommended recipes that used radish (seven recipes) or eggplant (six recipes).

## 6. CONCLUSION

In this study, we developed HEHSS, integrating NMS and KRS. This study focused on a performance evaluation of the system for long term use of real recipe data. Using 7,260 real recipes, the system could recommend various menus adjusting the user's nutritional intake for eight weeks while satisfying all 13 nutritional needs. Moreover, KRS could learn more than 80% of a user's preferences after eight weeks' evaluation and recommend various user preferred menus. However, only quasi-users (Agents of set preferences) were used in this simulation, and we plan to test the effectiveness of the system by using real users in future.

## ACKNOWLEDGEMENT

This research is partially supported by MEXT-Supported Program for the Strategic Research Foundation at Private Universities, 2013-2017 from Ministry of Education, Culture, Sports, Science and Technology.

## REFERENCES

- Allrecipes.com. Available: <http://allrecipes.com/Recipes/main.aspx> [Accessed 10 June 2013]
- CDKitchen.com. Available: <http://www.cdktichen.com> [Accessed 10 June 2013]
- FOODILY. Available: <http://www.foodily.com> [Accessed 10 June 2013]
- Allrecipes.com – Newsroom. Available: <http://allrecipes.com/help/aboutus/newsroom.aspx> [Accessed 24 January 2012]
- Tokumi, Y., Hakamata, J., & Tokumaru, M. (2013). Development of a nutritional management system for a healthy eating habits support system. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 17(2), 324-334.
- Hakamata, J., Tokumi, Y., & Tokumaru, M. (2011). Development of a healthy eating habits support system that presents menus considering a user's taste and health: Optimization of Kansei retrieval system. *12th International Symposium on Advanced Intelligent Systems - ISIS2011*, 479-482.
- Glover, F. (1989). Tabu Search - Part1, *ORSA Journal on Computing*, 1(3), 190-206.
- Glover, F. (1990). Tabu Search - Part2, *ORSA Journal on Computing*, 2(1), 4-32.
- Sumikawa, H., Miyashiro, R., & Nakamori, M. (2007). A Tabu Search Algorithm for the Traveling Tournament Problem. *IPSSJ SIG technical reports*, 64, 5-8.
- Ministry of Health, Labour and Welfare in Japan. The dietary reference intakes of Japanese 2010. Available: <http://www.mhlw.go.jp/shingi/2009/05/s0529-4.html> [Accessed 5 June 2012]
- COOKPAD. Available: <http://cookpad.com> [Accessed 10 June 2013]
- Calorepi. Available: <http://calorepi.com> [Accessed 10 June 2013]
- TANITA CORPORATION. (2010). Canteen of body fat scale TANITA -500 kcal meal satiety-, Daiwa bookstore.
- TANITA CORPORATION. (2010). Sequel to canteen of body fat scale TANITA -500 kcal meal satiety-, Daiwa bookstore.

Smith, J.R. (1991). Designing Biomorphs with an Interactive Genetic Algorithm. Proceedings of the Fourth International Conference on Genetic Algorithm (ICGA'91), San Diego, CA, USA, Morgan Kaufmann Publisher, 535-538.

Horn, J., Nafpliotis, N. & Goldberg, D.E. (1994). A niched pareto genetic algorithm for multiobjective optimization. Proc. of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, 1, 82-87.

Tokumaru, M. & Muranaka, N. (2008). An Evolutionary Fuzzy Color Emotion Model for Coloring Support Systems. 2008 IEEE International Conference on Fuzzy Systems (FUZZ 2008), 408-413.

2008.

Kling, R.M. & Banerjee, P. (1987). A new standard cell placement package using simulated evolution. Proceedings of 24th Design Automation Conference, 60-66.

## **BIOGRAPHY**

Masataka Tokumaru is an Associate Professor of Faculty of Engineering Science at Kansai University in Osaka, Japan since 2008. He received the B.E. and M.E. degrees in Engineering from Kansai University in 1994 and 1997. He received Dr. (Eng.) degree from Osaka City University, Japan in 2001. His research interests are Kansei information processing and evolutionary computation. He is a member of Japan Society of Kansei Engineering, Institute of Electronics, Information and Communication Engineers, Japan Society for Fuzzy Theory and Intelligent Informatics, the Japanese Society for Artificial Intelligence and Human Interface Society.