PAPER

A Cooking-Step Scheduling Algorithm with Guidance System for Homemade Cooking

Yukiko MATSUSHIMA[†], ***Student Member and*** **Nobuo FUNABIKI**[†a)], ***Member***

SUMMARY *Homemade cooking* plays a key role for a healthy and cost-efficient life. Unfortunately, preparing multiple dishes is generally time-consuming. In this paper, an algorithm is proposed to minimize the cooking time by scheduling the cooking-step of multiple dishes. The cooking procedure of a dish is divided into a sequence of six types of *cooking-steps* to consider the constraints in cooks and cooking utensils in a kitchen. A *cooking model* is presented to optimize the cooking-step schedule and estimate the cooking time for a given starting order of dishes under various constraints of cooks and utensils. Then, a high-quality schedule is sought by repeating the generation of a new order and the model application based on exhaustive search and simulated annealing. Our simulation results and cooking experiments confirm the effectiveness of our proposal.

key words: *homemade cooking, cooking model, cooking-step scheduling, algorithm, exhaustive search, simulated annealing*

1. Introduction

Homemade cooking can play a key role for a healthy and cost-efficient life. However, for people unaccustomed to cooking, preparing multiple dishes for a dinner is not an easy task. Even for people experienced in cooking, homemade cooking in every day is actually a hard job due to the time limitation, since they generally work or study during day time. As a result, a lot of people are eating at restaurants or buying lunch boxes at stores. On the other hand, they may fail in keeping well-balanced nutritional diets recommended in *Food Balance Guide* by the government [1]. Furthermore, there have been public fears about the metabolic syndrome [2]. Thus, more and more people have desired to enjoy healthy and cost-effective diets by cooking at home.

One solution to this situation can be that for a given set of dishes, we provide an optimal cooking scheduling and the resulting cooking time. Then, users can reselect dishes so that they can complete cooking within the available time. If the estimated cooking time is too long, they may give up some dishes or choose other dishes that need shorter cooking time. Conversely, they can add more dishes or choose time-consuming dishes. Furthermore, by having the information of cooking time, people can start cooking at the proper time so that they can eat dishes just when they need.

In this paper, we propose an algorithm that schedules the *cooking-steps* for multiple dishes with the estimated cooking time, so as to minimize the total cooking time. For this algorithm, the cooking procedure for each dish is divided into a sequence of cooking-steps to consider the constraints in cooks and cooking utensils in a kitchen. A *cooking model* is presented to optimize the schedule of the cooking-steps and estimate the cooking time for a given order of starting dishes under various conditions in the kitchen layout and the number of cooks, such that the constraints in cooks and cooking utensils are satisfied at any cooking-step in the schedule.

For this cooking model, we define a *kitchen layout* that is composed of cooks, cutting boards, ranges, microwave ovens, and a sink, where the number of cooking utensils can be specified flexibly. For the cooking process, we define six types of cooking-steps, namely, *Cut-step*, *Mix-step*, *Fry-step*, *Boil-step*, *Microwave-step*, and *Stand-step*. To check the constraints in applying each cooking-step, we define the states of utensils and cooks for a given kitchen layout.

To find the cooking schedule and estimate the cooking time for a given starting order of cooking dishes, the cooking model arranges the cooking-steps at the earliest possible time such that the constraints are satisfied. This model can consider two types of cooks, namely, a *main-cook* and a *sub-cook*. A *main-cook* can handle any cooking-step, whereas a *sub-cook* can handle a subset of the cooking-steps, assuming a helper for a main-cook such as a partner or a child. Actually, the Japanese government has supported projects such as "Equal employment and work and family harmonization" [3] and "Ikumen (child-rearing men) project" [4], to increase opportunities of cooking together by family members. One goal of our algorithm is to encourage a partner or a child to participate in cooking.

Using this cooking model for a given set of multiple dishes, our cooking-step scheduling algorithm seeks a schedule that minimizes the total cooking time. For a small number of dishes, we apply exhaustive search to identify the optimal order of cooking dishes, while for a large number of dishes, *simulated annealing (SA)* [5] is adopted to find a satisfying solution within acceptable time. Furthermore, the algorithm can be used to plan menus for homemade cooking under the limited cooking time constraint [6].

In addition, we implement a *cooking guidance system* on an Android tablet as a standalone system using Java, to navigate the cooking-step applications by the schedule during real cooking in a kitchen. This system can record the accurate time required for each cooking-step, because the

Manuscript received February 9, 2015.
Manuscript revised May 4, 2015.
Manuscript publicized May 18, 2015.

[†]The authors are with the Graduate School of Natural Science and Technology, Okayama University, Okayama-shi, 700–8530 Japan.
a) E-mail: funabiki@okayama-u.ac.jp

DOI: 10.1587/transinf.2015EDP7048

feedback of the real cooking time is essential to improve the accuracy of the algorithm.

To evaluate our proposal, we executed two simulations and two cooking experiments. In the first simulation, the reduction of the total cooking time was investigated for various number of dishes ranging from two to ten. In the second simulation, the robustness of the algorithm schedule was verified by fluctuating required time for cooking-steps. Then, in the first cooking experiment, four dishes were cooked by a main-cook only. In the second one, four different dishes were cooked by main and sub-cooks. In both experiments, the cooking time difference between the schedule and the real cook was less than four min.

The rest of this paper is organized as follows: Sects. 2–4 present the cooking model, cooking-step scheduling algorithm, and cooking guidance system respectively. Section 5 shows evaluation results. Section 6 introduces some related works. Section 7 concludes this paper with future studies.

2. Cooking Model

In this section, we present the cooking model to optimize the arrangement of applying the cooking-steps for multiple dishes and estimate the cooking time.

2.1 Kitchen Layout

Figure 1 illustrates the kitchen layout in the cooking model. The kitchen consists of cooks, cutting boards, ranges, microwave ovens, and a sink. The number of cooks may be either one or two. When two cooks are considered, they are designated as a main-cook and a sub-cook. The main-cook can execute any cooking-step and complete any dish by himself/herself. On the other hand, the sub-cook, assuming a partner or a child, can execute a part of the cooking-steps to help the main-cook. The cooking utensils in the kitchen, e.g., the number of cutting boards, ranges, and microwave ovens, can be specified by the user. For simplicity, our model assumes that these plural utensils have the same performance such as heating ability and functions. Besides, we assume a sufficient number of pans and pots, and sufficient kitchen space.

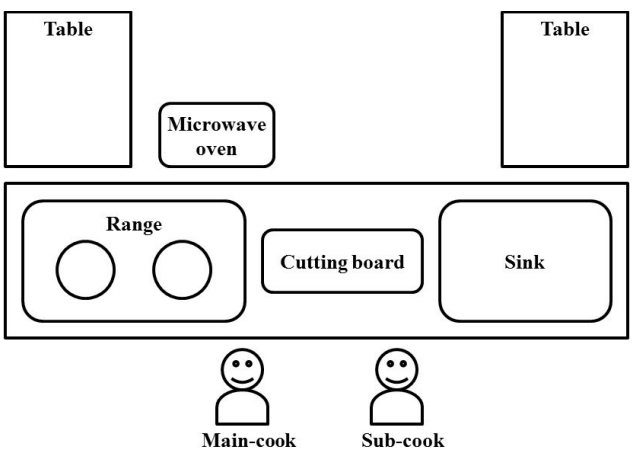

Fig. 1 Kitchen layout.

2.2 Cooking-Steps for Cooking Process

In this model, the cooking process for a dish is regarded as a sequence of cooking-steps. Empirically, we define six different cooking-steps depending on the natures, namely, *Cut-step*, *Mix-step*, *Fry-step*, *Boil-step*, *Microwave-step*, and *Stand-step*. We consider that for the first three cooking-steps, a cook must always spend his/her labor during its processing, whereas for the remaining steps, a cook needs to do so only at the beginning and end of the process.

In *Cut-step*, a cook processes ingredients by peeling, cutting, and slicing them on a cutting board. In *Mix-step*, a cook mixes ingredients by blending, kneading, and wrapping them. In *Fry-step*, a cook heats ingredients by frying, deep-frying, or grilling them using a pan on a range where a cook is always engaged to avoid scorching. In *Boil-step*, a cook can heat ingredients using a range without a load by boiling, stewing, and steaming them. In *Microwave-step*, a cook can heat ingredients using a microwave oven. In *Stand-step*, a cook keeps ingredients in the current shape for a certain time.

The setting-up procedures before and after the main one, e.g., washing utensils and ingredients, preheating oil, or moving ingredients from a pan to a plate, usually do not take long time and thus can be included in each cooking-step.

Here, we note that *Cut-step*, *Mix-step*, and *Fry-step* require a cook load, which can be executed only when a cook is free. On the other hand, *Boil-step*, *Microwave-step*, and *Stand-step* do not require a cook load, which can be executed in parallel with other steps without considering a free cook. Thus, by applying the cooking-steps properly, the reduction of the cooking time can be expected even for one cook.

A dish recipe can be converted into a sequence of the abovementioned cooking-steps in a straightforward way by finding a keyword corresponding to each cooking-step type. We have developed this conversion algorithm using a list that contains the set of keywords representing each cooking-step [7], and verified the effectiveness through applications to randomly selected recipes from two Web sites [8] and [9].

2.3 State Transitions

The state of a cooking dish starts from *Ready* and ends at *Completion* after transiting six cooking-steps. *Ready* represents the state where the first cooking-step of the dish is ready to start. *Completion* does the state where all the cooking-steps have completed for the dish. Two or more cooking-steps of different dishes can appear at the same time, such as cutting ingredients (*Cut-step*) while boiling water (*Boil-step*). The transition to each cooking-step must satisfy the following conditions:

1. A *cook* and a *cutting board* must be available for the

transition to *Cut-step*.
2. A *cook* must be available for the transition to *Mix-step*.
3. A *cook* and a *range* must be available for the transition to *Fry-step*.
4. A *range* must be available for the transition to *Boil-step*.
5. A *microwave oven* must be available for the transition to *Microwave-step*.

A cook has two states of *Available* and *Busy*. *Available* represents the state where he/she can start a new cooking-step requiring the load such as *Cut-step*, *Mix-step*, and *Fry-step*, while *Busy* represents the state where the cook is occupied. In the cooking model, the number of cooks and the roles of the sub-cook can be specified by the user.

A cooking utensil, e.g., a cutting board, a range, and a microwave oven, also has *Available* and *Busy* states and the number of each utensil can be specified by the user.

2.4 Preferential Cooking-Steps

Some dish recipes require processing two or more cooking-steps as continuously as possible. For example, cooking a soup often requires boiling ingredients immediately after frying them. To consider this kind of recipe, we introduce the *preferential cooking-step flag*. When this flag is *ON*, the corresponding dish is always moved to the top of the *dish order*, and thus, the next cooking-step is selected with the first priority when the conditions are satisfied. Here, if the next step cannot be started because of unavailability of a cook or a utensil, some gap time appears until the conditions are satisfied.

2.5 Role Limitation and Dish Preference of Sub-Cook

We assume that a sub-cook is a partner or a child who can execute a part of the six cooking-steps. For example, *Cut-step* may be too hard or dangerous for a sub-cook and thus should be prohibited. If the sub-cook is a child, only the *Mix-step* should be assigned.

Besides, if a sub-cook is assigned all of the possible cooking-steps of a certain dish, he/she can enjoy cooking more, and may cook this dish by himself/herself in the next time by understanding the whole cooking process. To consider this preference, we adopt the *sub-cook preference flag* for the dish. When this flag is *ON*, the possible cooking-steps of the corresponding dish are assigned to the sub-cook preferentially.

2.6 Cooking-Step Schedule Generation

In this subsection, we present the procedure to generate a schedule of applying the cooking-steps for a given starting order of dishes through cooking simulations.

2.6.1 Input

The inputs to this procedure are given as follows:

- the number of servings for cooking: m
- the number of cooks, cutting boards, stoves, and microwave ovens
- the possible cooking-steps for the sub-cook
- the skill levels for the cooks
- the cleaning chance threshold (min.)
- the list of n dishes for cooking: $\mathcal{V} = \{1, \ldots, n\}$
  - the name of dish i ($i \in \mathcal{V}$)
  - the sub-cook preference flag
  - the cooking-step of dish i at step j
    - the cooking-step type
    - the cooking time (min.)
    - the preferential cooking-step flag

2.6.2 Output

The output is the cooking-step schedule σ for all the dishes, which describes *when*, *who* should do *what*.

2.6.3 Objective

Two objective functions can be defined for this procedure. The first one represents the *maximum cooking time to* complete all the dishes in Eq. (1), which intends that the cooking can be finished as fast as possible. The second one represents the *cooking time difference* between the first and last dish in Eq. (2), which intends that every dish can be served at the same time as best as possible. C_i represents the elapsed time since starting the cooking when the state dish i becomes *Completion*. In our evaluations, we use the first objective function as a clearer index to evaluate the effectiveness of our proposal.

$$f_1(\sigma) = \max_{i \in \mathcal{V}} C_i. \tag{1}$$

$$f_2(\sigma) = \max_{i \in \mathcal{V}} C_i - \min_{i \in \mathcal{V}} C_i. \tag{2}$$

2.6.4 Procedure

Our cooking model finds a schedule of applying the cooking-steps for the dishes, and estimates the cooking time to cook the dishes by simulating their cooking processes, for a given order for starting the first steps of the dishes. The starting order of dishes, called the *dish order*, is represented by ν in this paper. Any dish with the *ON* preferential cooking-step flag for the current step is always moved to the top of the order so as to be selected with the first priority. The procedure for the cooking simulation in the model is as follows:

1. Initialize T by 0.
2. End the current cooking-step if its cooking time is elapsed.
3. Transit the state to *Completion* of a dish and record the completion time when the last cooking-step is ended.
4. Terminate the procedure when the state of every dish

becomes *Completion*.
5. Select a next cooking-step for a main-cook or a sub-cook when the state is *Available*. Record the starting time if it is selected.
6. Select a next cooking-step for a range or a microwave oven when the corresponding state is *Available*. Record the starting time if it is selected.
7. Record the starting time when the cooking-step of a dish becomes *Stand-step*.
8. Increment T by 1, and go to 2.

The following subsections describe the details of the next cooking-step selections in 5. and 6. in this procedure.

2.6.5 Next Cooking-Step Selection for Main-Cook or Sub-Cook

The next cooking-step of a dish for an available cook is selected by the following procedure:

1. The first dish in v that satisfies the following four conditions is found:
   a. The state is not *Completion*.
   b. The next cooking-step of the dish is none of *Boil-step*, *Microwave-step*, and *Stand-step* because they do not need a cook.
   c. If the cook is a sub-cook, he/she is permitted for the next step.
   d. If the cook is a main-cook, the *sub-cook preference flag* for the dish is *OFF* or the sub-cook is prohibited from the next step.
2. If the next cooking-step of the dish is *Cut-step*, the state of the cutting board is checked:
   a. If the state of one cutting board is *Available*, this next cooking-step is selected for the cook, and the state of this cutting board is changed to *Busy*.
   b. If every cutting board is *Busy*, this dish is given up, and 1. is repeated to check the next dish candidate.
3. If the next cooking-step of the dish is *Mix-step*, this cooking-step is selected for the cook.
4. If the next cooking-step of the dish is *Fry-step*, the state of the range is checked:
   a. If the state of one range is *Available*, this next cooking-step is selected for the cook, and the state of this range is changed to *Busy*.
   b. If every range is *Busy*, this dish is given up, and 1. is repeated to check the next dish candidate.
5. If a new cooking-step is selected for a cook, the state of this cook is changed to *Busy*.

2.6.6 Next Cooking-Step Selection for Range or Microwave Oven

The next cooking-step of a dish for an available range or microwave oven is selected by the following procedure:

1. The first dish in v that satisfies the following two conditions is found:
   a. The state is not *Completion*.
   b. The next cooking-step of the dish is none of *Cut-step*, *Mix-step*, and *Fry-step* because they need a cook.
2. When the next cooking-step of the found dish is *Boil-step*, the state of a range is changed to *Busy*.
3. When the next cooking-step of the found dish is *Microwave-step*, the state of a microwave oven is changed to *Busy*.

2.7 Cooking Time Estimation

In addition to the schedule optimization, our cooking model can be used to estimate the total cooking time and the starting time of each cooking-step in a given schedule to evaluate it through simulations. In this subsection, we describe the modifications in the cooking-step schedule generation for this use.

2.7.1 Input and Output

In the inputs to this cooking time estimation, a cooking-step schedule and the updated time for cooking-steps are used. The updated time can be obtained from real cooking or be generated by randomly fluctuating the original one for simulations. In the output, only the starting/ending time of a cooking step may be different from that in the input schedule.

2.7.2 Cooking Time Estimation Procedure

In the cooking time estimation, the schedule or the sequence of applying the cooking-steps is fixed as that in the input one, whereas the starting/ending time may be changed. Thus, only the end of the current cooking-step and the conditions of starting the next cooking-step are checked here as in the following procedure:

1. Initialize T by 0.
2. End the current cooking-step if the elapsing time from the start reaches the given time.
3. Transit the state to *Completion* of a dish and record the completion time when the last cooking-step is completed.
4. Terminate the procedure when the state of every dish becomes *Completion*.
5. Start the next cooking-step in the given schedule and record the time if the conditions are satisfied.
6. Increment T by 1, and go to 2.

3. Cooking-Step Scheduling Algorithm

In this section, we present the cooking-step scheduling al-

gorithm to seek a *dish order* ν that minimizes the objective function in Eq. (1) or Eq. (2). Specifically, the cooking schedule is obtained by repeating the generation of ν and the calculation of the objective function using the cooking model in Sect. 2. For the generation of ν, we adopt exhaustive search when the number of dishes is small. For large number of the dishes, simulated annealing (SA) is applied.

3.1 Cooking Time Derivation

First, the required time for each cooking-step of a dish is derived from the recipe or from the table where the standard cooking time is defined for each ingredient. When the latter one is used, it is then adjusted by considering the volume, the way of cooking, and the skill level of the cook.

In the table, the standard time has been specified for each cooking-step of each ingredient with the unit volume. If the standard time is not given in the table for the ingredient under cooking, the default standard time for each cooking-step is used. For this purpose, the ingredient name is extracted from the recipe in our algorithm.

Then, the standard time is adjusted by the volume of the ingredient. Because the time specified in the table is normalized with respect to the unit volume, it is multiplied by the value of the actual volume in cooking.

Then, for *Cut-step*, this time is further adjusted by considering the way of cutting because the time is strongly affected by it. Also, for *Fry-step* and *Boil-step*, the time is adjusted by the way of frying or boiling. Here, the time is multiplied by the specified adjustment factor that has been defined for each way of cutting, frying, or boiling in another table.

Furthermore, the time depends on the skill of a cook. Thus, it is further multiplied by the skill factor that has been defined for each cooking-step and each skill level in the same table.

3.2 Exhaustive Search for Smaller Dish Number

When the number of dishes is seven or less, exhaustive search on the dish order is applied to find an optimal solution in terms of the objective function given from the cooking model. Because the number of dishes for one supper is usually up to five in our daily lives, even exhaustive search does not take long time. We note that the number of possible dish orders is 5,040 for seven dishes. Here, each possible dish order is fed into the cooking model, and the corresponding objective function is calculated. Then, the cooking-step schedule providing the minimum objective function is selected.

3.3 Simulated Annealing for Larger Dish Number

When the number of dishes becomes larger, SA is applied to find a near-optimal solution within acceptable time. Initially, ν is randomly generated. Then, ν is slightly modified by randomly swapping two adjacent dishes. The transition to the new ν is accepted with probability $e^{-\frac{\Delta}{t}}$, where Δ is the difference between the current and previous objective functions and t is the temperature. The temperature decreases gradually by multiplied with 0.95 [5], and ν is generated in constant times at each temperature. Note that if $\Delta \leq 0$, the transition probability is set to 1. Thus, the following three parameters, k_1, k_2, and k_3, should be adjusted in SA, where $k_1 = 5$, $k_2 = 20$, and $k_3 = 10$ are used in our simulations [10]. In our simulations, this SA can find an optimal or near-optimal solution for any of 100 instances with different combinations of dishes where the number of dishes increases from two to ten.

- the initial temperature: $k_1 \times n$
- the number of temperature changes: $k_2 \times n$
- the number of dish order generations at each temperature: $k_3 \times n$.

3.4 Cleaning Chance Suggestion

The generated cooking-step schedule may give a cook free time that can be used for cleaning or washing used pots, plates, or others. Thus, to suggest such timing, our algorithm outputs the *clearance chance suggestion* message when the idling time of a cook is longer than the given threshold in the schedule.

4. Cooking Guidance System

In addition to the algorithm, we also develop a *cooking guidance system* on an Android tablet to navigate the cooking procedure.

4.1 Overview

The cooking guidance system uses the inputs/outputs of the *cooking-step scheduling algorithm* by copying the corresponding CSV files manually. This system consists of the functions for the *menu display*, the *cooking guidance*, and the *cooking-time measurement*. The following subsections will discuss their details.

4.2 Menu Display

The *menu display* shows the list of the dishes in a menu and the list of the ingredients for each dish, so that the user can confirm the details of the dishes that he/she will cook. Figures 2 and 3 illustrate examples of the dish list and the ingredient list for a dish respectively.

4.3 Cooking Guidance

The *cooking guidance* navigates the cooking process for each cook (main-cook or sub-cook) by showing the sequence of the cooking-steps in the schedule. This function can switch the display for both cooks, for the main-cook

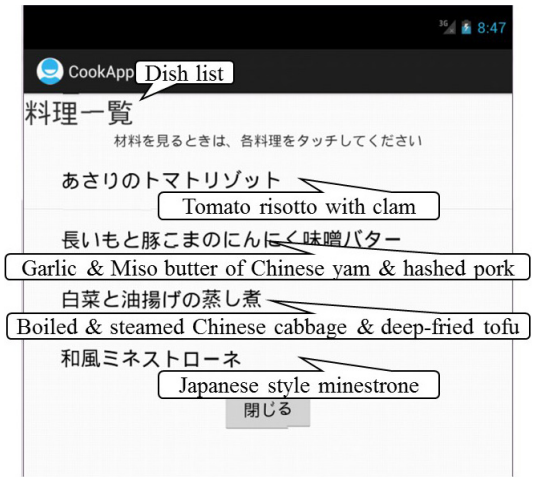

Fig. 2 Dish list display.

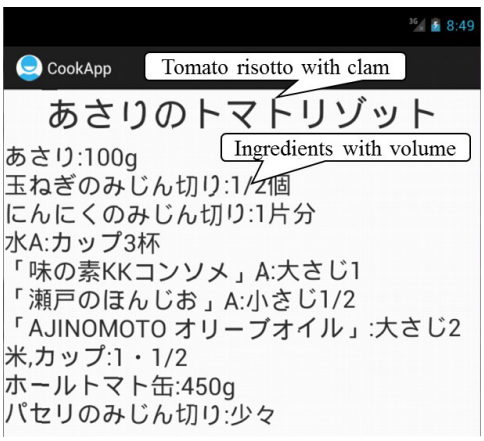

Fig. 3 Ingredient list display.

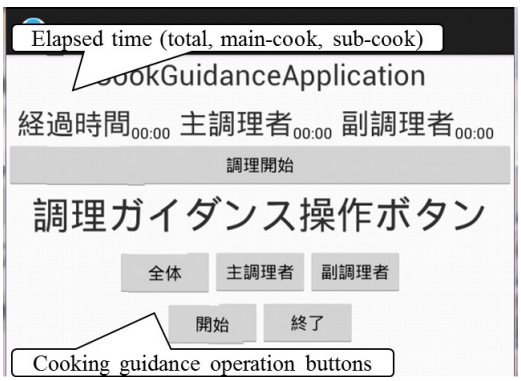

Fig. 4 Cook selection interface.

only, and for the sub-cook only, by clicking the corresponding button in the interface. Figure 4 illustrates the cook selection interface.

Figures 5, 6, and 7 illustrate the interface of displaying the sequence of the cooking-steps for the both cooks, for the main-cook, and for the sub-cook respectively. In each interface, the current cooking-step for the main-cook

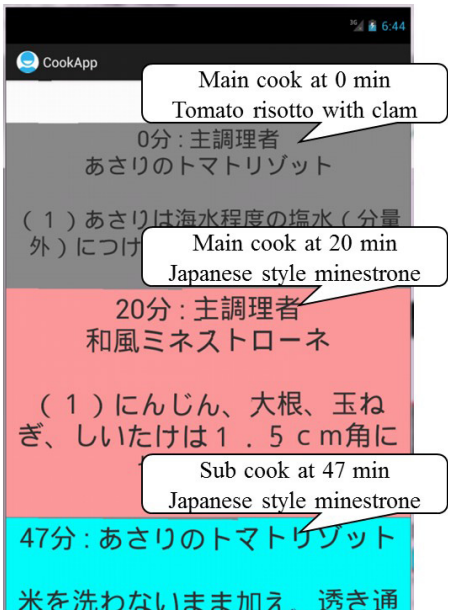

Fig. 5 Cooking guidance interface for both cooks.

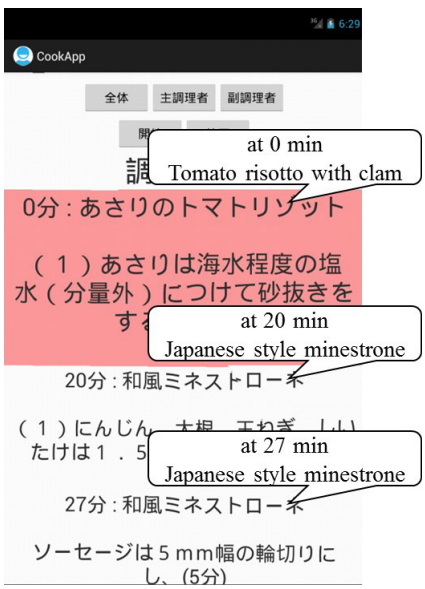

Fig. 6 Cooking guidance interface for main-cook.

is highlighted by red, and that for the sub-cook is by blue to avoid misrecognitions. Besides, the color of the completed cooking-steps is changed from white to gray. The start button and the end button in the interface are prepared so that the user can signal the transition of the cooking-step to the function. When either button is clicked, the function changes the color of the corresponding cooking-step.

4.4 Cooking-Time Measurement

The interface in Fig. 4 shows the elapsed time for each cooking-step and the entire cooking process. The elapsed time for a cooking-step starts when the corresponding start

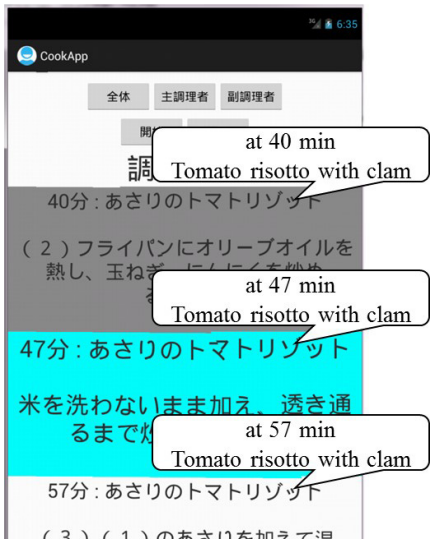

Fig. 7 Cooking guidance interface for sub-cook.

button is clicked and ends when the end button is clicked. The elapsed time for the entire cooking process starts when any start button is first clicked and ends when the end button for the last cooking-step is clicked. The measured elapsed time is saved in the database to improve the accuracy of time parameters in the cooking model by reflecting the real time of the user.

5. Evaluations

In this section, we verify the effectiveness of our proposal by simulations and cooking experiments.

5.1 Evaluations by Simulations

First, we evaluate the effectiveness of the cooking-step scheduling algorithm by simulations in terms of the computation time, the cooking time reduction, and the robustness of generated schedules.

5.1.1 Conditions for Evaluations

For our simulations, we randomly selected 100 dishes from the recipes provided in "Ajinomoto Park" [8] and "me:new" [9], and converted each recipe into a sequence of six types of cooking-steps as the input data set to the algorithm using the Web recipe conversion algorithm in [7] with slight modifications for the cooking time derivation in Sect. 3.1. Then, we randomly selected dishes from this input data set and applied our algorithm. This simulation is repeated 100 times to obtain the average result.

We note here that we excluded the recipes that could not be converted automatically by the conversion algorithm. Actually, the recipe text processing is one of the hot topics in information retrieval, case-based reasoning, and natural language processing fields. Many works have been reported for solving the problem of confusing and ambiguous expressions in recipes [11], [12].

In simulations, a main-cook and a sub-cook participated, and one cutting board, two ranges, and one microwave oven were available in the kitchen. As expected users of our algorithm, we assumed a housewife as the main-cook who can handle any cooking-step, and a college student as the sub-cook who cannot use a knife well. Thus, *Cut-step* was excluded from his cooking-steps. The skill levels of the both cooks are set to be normal.

Table 1 Solution quality and computation time.

number of dishes n	optimal solution percentage with SA (%)	CPU time with SA (sec.)	CPU time with exhaustive (sec.)
2	100	0.0769	0.0004
3	100	0.1269	0.0004
4	99	0.1934	0.0025
5	94	0.2747	0.0145
6	93	0.3705	0.1182
7	89	0.4627	0.9936
8	87	0.5800	9.8695
9	88	0.6908	106.2833
10	91	0.8635	313.4160

5.1.2 Solution Quality and Computation Time

First, we evaluate the solution quality and the CPU time by exhaustive search and SA of the algorithm on the platform Windows7 Home Premium for OS, Intel Core i5-2450M 2.50GHz with 4.00GB memory. Table 1 shows the optimal solution percentage found by SA among 100 different combinations of dishes for each number of dishes, and the average CPU time by the two methods. Here, an *optimal solution* means the cooking-step schedule by the cooking model that gives the minimum value of the total cooking time $f_1(\sigma)$ in Eq. (1).

Table 1 shows that exhaustive search is actually faster than SA until $n = 6$. Thus, we recommend the use of exhaustive search when the number of dishes is seven or less and SA otherwise.

5.1.3 Cooking Time Reduction

Then, we evaluate the cooking time reduction of the proposal algorithm. Figure 8 compares the total cooking time in the three cases for the number of dishes ranging from two to ten. For the *sequential cooking*, the total cooking time is given by simply adding the time for every dish where no parallel processing is applied. For the *proposal with random order*, the proposed cooking model is adopted and the dish order ν is randomly generated. For the *proposal with algorithm order*, we apply the proposed cooking model and the cooking order generated by the proposed algorithm. The *proposal with random order* can reduce the cooking time by 34% compared with the sequential cooking. Then, the *proposal with algorithm order* can further reduce it by 9.6%, demonstrating the effectiveness of our proposed cooking

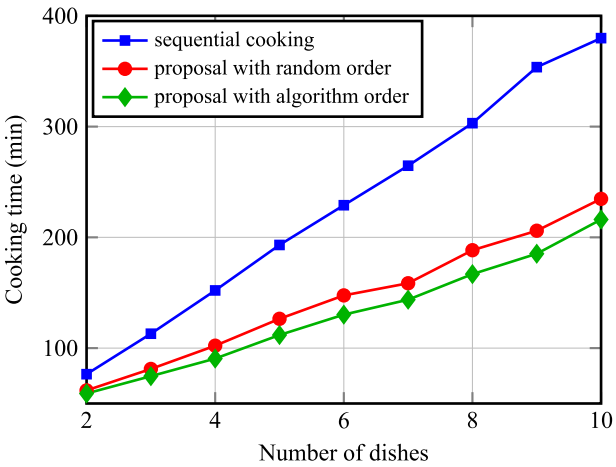

Fig. 8 Comparisons of total cooking time by three scheduling methods with respect to different number of dishes.

Table 2 Total cooking time for randomly fluctuated actual time (min.).

	original case		real case		ideal case	
algorithm	original		original		fluctuated	
estimation	original		fluctuated		fluctuated	
number of dishes n	avg.	s.d.	avg.	s.d.	avg.	s.d.
3	74.26	23.22	75.06	24.04	74.67	23.96
4	92.16	22.53	92.4	22.74	91.72	22.27
5	115.75	31.91	117.10	32.23	116.44	32.27
6	133.16	30.27	133.39	30.00	132.82	30.28
7	148.23	33.68	149.29	32.98	148.38	33.59

model and algorithm.

5.1.4 Robustness of Generated Schedule

Furthermore, we evaluate the robustness of the cooking-step schedule generated by the algorithm through simulations under the condition that the actual cooking time for each cooking-step is fluctuated. Here, the conditions in Sect. 5.1.1 are used, and the actual cooking is generated by applying a 30% fluctuation on the original time.

For this purpose, we compare the total cooking time in three cases by adopting either the original time or the fluctuated time for the algorithm in Sect. 3 and for the cooking time estimation in Sect. 2.7. The first case adopts the original time for both, which is called the *original case*. The second case adopts the original time for the algorithm and the fluctuated time for the estimation, which is called the *real case*. The third one adopts the fluctuated time for both, which is called the *ideal case*, because it assumes the time fluctuation is predictable.

Table 2 shows the average (avg.) and standard deviation (s.d.) of the total cooking time for 100 combinations of dishes for the number of dishes ranging from three to seven, which shows the negligible difference among for all number of dishes. Thus, the cooking-step schedule by the algorithm is robust in terms of fluctuations of real cooking time for each cooking-step.

Table 3 Cooking-step schedule of Main-Cook.

start time	who	dish	step	time (min.)
0	main	Tomato salad	cut	5
0	range1	Consommé	boil	6
5	main	Tomato salad	cut	2
7	main	Consommé	mix	1
8	main	Tomato salad	mix	2
10	main	Pickled cucumber	cut	3
13		Pickled cucumber	stand	10
13	main	Fried eggplant	cut	5
18	main	Fried eggplant	cut	7
25	main	Pickled cucumber	mix	3
28	main	Pickled cucumber	mix	2
30	main	Pickled cucumber	mix	1
31	main	Fried eggplant	fry	4
35	main	Fried eggplant	fry	2
37	main	Fried eggplant	fry	1
38	main	Fried eggplant	fry	2
scheduled total cooking time (min.)				40
actual total cooking time (min.)				38

5.2 Evaluations by Cooking Experiments

In this section, we evaluate the effectiveness of the algorithm through real cooking experiments in kitchens where the same number of utensils in Sect. 5.1.1 are used. We adopted the *cooking guidance system* on an Android tablet to navigate the cooking-step applications in the schedule [13].

5.2.1 Cooking Experiment by Main-Cook

In the first experiment, only a main-cook who has the normal skill level participated. She cooked four dishes, namely *Fried eggplant with chili source*, *Tomato salad*, *Consommé*, and *Pickled cucumber*. Table 3 shows the cooking-step schedule by the algorithm. The real cooking time (38min.) of this cook is only two minutes shorter than the simulation (40min.).

5.2.2 Cooking Experiment by Main and Sub-Cooks

In the second experiment, a female main-cook and a male sub-cook both with the normal level participated. They also cooked four dishes, namely *Chinese-style fried pork and white radish*, *Bean sprout and tuna salad*, *Tapped cucumber and tuna salad*, and *Enokidake mushroom and egg soup*. Table 4 shows the schedule by the algorithm. The real cooking time (35min.) of this cook is four minutes longer than the simulation (31min.). The reason why the time difference becomes larger than the first experiment can be that the sub-cook was unfamiliar with the kitchen and often had to interrupt the main-cook by asking the location of a seasoning in the kitchen.

6. Related Works

In [14], Hamada et al. presented a method to generate a

Table 4 Cooking-step schedule of Main and Sub-Cooks.

start time	who	dish	step	time (min.)
0	sub	Enokidake soup	mix	6
0	main	Chinese-style fried pork	cut	3
3	main	Chinese-style fried pork	mix	3
6	sub	Bean sprout salad	mix	2
6	main	Chinese-style fried pork	cut	3
8	m-oven	Bean sprout salad	microwave	6
9	main	Chinese-style fried pork	cut	2
11	sub	Chinese-style fried pork	fry	3
11	main	Enokidake soup	cut	3
14		Bean sprout salad	stand	2
14	range1	Enokidake soup	boil	5
14	sub	Chinese-style fried pork	fry	3
14	main	Tapped cucumber salad	cut	5
17	sub	Chinese-style fried pork	mix	3
19	main	Enokidake soup	mix	2
20	sub	Chinese-style fried pork	mix	1
21	sub	Enokidake soup	mix	2
21	main	Tapped cucumber salad	cut	1
22	main	Tapped cucumber salad	mix	1
23	sub	Tapped cucumber salad	mix	5
23	main	Bean sprout salad	mix	5
28	sub	Tapped cucumber salad	mix	1
28	main	Bean sprout salad	mix	1
29	main	Bean sprout salad	cut	2
scheduled total cooking time (min.)				31
actual total cooking time (min.)				35

flow graph automatically from textbooks for cooking programs by creating a domain specific dictionary using statistical methods and applying structural analysis methods using the dictionary. In this graph, the cooking process in a recipe is divided into a sequence of primitive operations such as "break an egg" and "bake an egg" (*action unit* in [15]).

In [15], Hamada et al. presented a cooking navigation system called *Cooking Navi* to allow users to cook several recipes in parallel by providing information to users with texts, videos, and audio. A recipe is represented by a *flow graph* that describes the order of processing ingredients and the constraints [14]. An action block is introduced to represent a series of *action units* that handle the same ingredient at the same resource. It can be used to represent the same role of the *preferential cooking-step flag* in our paper.

Then, by considering the human resource and the kitchen resource, an optimal cooking schedule is searched by the list scheduling algorithm [16]. The schedule for the remaining process can be dynamically rescheduled. Although their goal is similar to ours and may be actually more advanced in providing information with a segmented video corresponding to each cooking-step [17], it has several disadvantages. 1) The conversion of a recipe to a flow graph is not easy for conventional users, where it needs an algorithmic conversion. On the other hand, the conversion in our proposal can be completed in a straightforward way by finding a keyword corresponding to each cooking-step type. 2) The sub-cook cannot be prohibited from dangerous cooking-step.

In [18], Freyne et al. presented a preliminary study into the suitability of recommender algorithms for recipe recommendation based on preferences provided by 512 users on a corpus of recipes. They examined the accuracy of collaborative and content-based filtering algorithms, and compared them to hybrid recommender strategies.

In [19], Ueda et al. presented a method for extracting the user's preferences from his/her recipe browsing and cooking history for a personalized recipe recommendation method based on the food preferences in [20].

In [21] Hu et al. presented a tabletop dish recommendation system for multiple participants dining together named the Group FDT (Future Dining Table), which always recognizes the dining status of the users by image processing, and recommends dishes timely based on the investigation of real dining, literature, and the experimental result.

7. Conclusion

To help homemade cooking, this paper presented a *cooking-step scheduling algorithm* that minimizes the cooking time by scheduling the cooking-step for multiple dishes. A *cooking guidance system* was also implemented on an Android tablet to navigate the cooking-step applications in the schedule. The effectiveness of our proposal was verified through simulations and real cooking experiments as well. Our future works include further experiments using the proposal, the construction of recipe database containing the necessary information for the algorithm, and the development of the Web-based cooking assistant system.

References

[1] Ministry of Health, Labour and Welfare (MHLW), "Food balance guide," 2005, http://www.mhlw.go.jp/bunya/kenkou/eiyou-syokuji.html
[2] Ministry of Health, Labour and Welfare (MHLW), "Prevention of metabolic syndrome," 2007, http://www.mhlw.go.jp/bunya/kenkou/metabo02/
[3] Ministry of Health, Labour and Welfare (MHLW), "Equal employment," 2011, http://www.mhlw.go.jp/bunya/koyoukintou/index.html
[4] Ministry of Health, Labour, and Welfare (MHLW), "Ikumen (child-rearing men) project," 2011, http://www.ikumen-project.jp/index.html
[5] M.S. Sadiq and H. Youssef, Iterative computer algorithms with applications in engineering, Wiley-IEEE Computer Society Press, 2000.
[6] N. Funabiki, S. Taniguchi, Y. Matsushima, and T. Nakanishi, "A proposal of a menu planning algorithm for two-phase cooking by busy persons," Proc. 3rd International Workshop on Virtual Environment and Network-Oriented Applications (VENOA 2011), pp.668–673, 2011.
[7] Y. Zhang, N. Funabiki, and T. Nakanishi, "A Web recipe conversion algorithm for cooking-step scheduling," IEICE Tech. Report (Japanese), vol.113, no.421, pp.11–16, Jan. 2014.
[8] Ajinomoto Park, http://park.ajinomoto.co.jp
[9] me:new, http://menew.jp/
[10] N. Funabiki and Y. Matsushima, "Cooking-step scheduling algorithm for simultaneous cooking of multiple dishes," Intelligent control and innovative computing, eds. S.I. Ao, O. Castillo, and X. Huang, Lecture Notes in Electrical Engineering, vol.110, pp.123–136, Springer US, 2012.
[11] DEIM 2015, http://db-event.jpn.org/deim2015/
[12] Cooking with Computers, http://liris.cnrs.fr/cwc/

[13] T. Okada, Y. Matsushima, N. Funabiki, T. Nakanishi, and K. Watanabe, "An android application of cooking guidance function in homemade cooking assistance system," IEICE Tech. Report (Japanese), vol.112, no.379, pp.91–96, Jan. 2013.
[14] R. Hamada, I. Ide, S. Sakai, and H. Tanaka, "Structural analysis of cooking preparation steps in Japanese," Proc. 5th Int. Workshop on Information Retrieval with Asian Languages, pp.157–164, 2000.
[15] R. Hamada, J. Okabe, I. Ide, S. Satoh, S. Sakai, and H. Tanaka, "Cooking Navi: Assistant for daily cooking in kitchen," Proc. 13th Annual ACM International Conference on Multimedia, pp.371–374, 2005.
[16] Y. Srikant and P. Shankar, The compiler design handbook: Optimization and machine code generation, CRC Press, 2002.
[17] R. Hamada, K. Miura, I. Ide, S. Satoh, S. Sakai, and H. Tanaka, "Multimedia integration for cooking video indexing," Proc. Pacific-Rim Conf. Multimedia., vol.II, pp.657–664, Dec. 2004.
[18] J. Freyne and S. Berkovsky, "Recommending food: Reasoning on recipes and ingredients," User Modeling, Adaptation, and Personalization, Lecture Notes in Computer Science, vol.6075, pp.381–386, Springer, Berlin, Heidelberg, 2010.
[19] M. Ueda, M. Takahata, and S. Nakajima, "User's food preference extraction for personalized cooking recipe recommendation," Proc. Second Workshop on Semantic Personalized Information Management: Retrieval and Recommendation, 2011
[20] M. Ueda, M. Takahata, and S. Nakajima, "Recipe recommendation method based on user's food preferences," Proc. IADIS Int. Conf. e-Society, pp.591–594, 2011.
[21] J. Hu, Y. Otsuka, and T. Inoue, "Automatic dish recommendation system for people dining together: The group FDT," Proc. Int. Conf. Collabo. Technol., pp.42–47, 2012.

Yukiko Matsushima received the B.S. degree in information science from Tokushima University, Japan, in 2001, and the M.S. degree in communication network engineering from Okayama University, Japan, in 2010, respectively. In 2001, she joined Be-Max Professional School as a lecturer. In 2015, she moved to the Department of Computer and Information Engineering at National Institute of Technology, Tsuyama College as an assistant professor. She is currently a Ph.D. candidate in Graduate School of Natural Science and Technology at Okayama University, Japan. Her research interests include educational technology and Web service systems.

Nobuo Funabiki received the B.S. and Ph.D. degrees in mathematical engineering and information physics from the University of Tokyo, Japan, in 1984 and 1993, respectively. He received the M.S. degree in electrical engineering from Case Western Reserve University, USA, in 1991. From 1984 to 1994, he was with Sumitomo Metal Industries, Ltd., Japan. In 1994, he joined the Department of Information and Computer Sciences at Osaka University, Japan, as an assistant professor, and became an associate professor in 1995. He stayed at University of Illinois, Urbana-Champaign, in 1998, and at University of California, Santa Barbara, in 2000–2001, as a visiting researcher. In 2001, he moved to the Department of Communication Network Engineering (currently, Department of Electrical and Communication Engineering) at Okayama University as a professor. His research interests include computer networks, optimization algorithms, educational technology, and Web technology. He is a member of IEEE and IPSJ.