Coordination Mechanism
for Optimized Provision of Services in an Area

Yuki Mori
Graduate school of
Science and Engineering
Ritsumeikan University
1-1-1, Noji-Higashi, Kusatsu, Shiga Japan

Fumiko Harada
College of
Information Science and Engineering
Ritsumeikan University
1-1-1, Noji-Higashi, Kusatsu, Shiga Japan

Hiromitsu Shimakawa
College of
Information Science and Engineering
Ritsumeikan University
1-1-1, Noji-Higashi, Kusatsu, Shiga Japan

Abstract: We are aiming at achievements of the Unconscious Computing to provide users with ubiquitous services even if they are not conscious of ubiquitous computer environments. In the Unconscious Computing, a user’s cellular phone recognizes the actions that the user should do. A server is installed in an area to know the number of resources such as buy-ticket places or ticket reservation counter and to manage the their usage status. When the user enters an area, his cellular phone provides the user with services which let him execute actions to require the resources through the server. However, it is difficult to provide services to all users if they are requesting services for the same resources from the same area.

To solve this problem, this paper proposes the Action Arbitration Function to arbitrate the competition for resources. The Action Arbitration Function integrates the action schedules of the users competing for a resource, reschedules the actions, and provides services to a user who has a tight action in consideration of their future actions.

Key–Words: Ubiquitous Computing, Ubiquitous Services, Unconscious Computing, Scheduling, Tight Action, Action Arbitration Function

1 Introduction

The technological advance in the mobile computing and networking have brought a promising vision for realization of ubiquitous environments consisting of information servers and appliances. Because of these concerns, the user can not devote himself to an action of his main purpose.

Ubiquitous services have been researched for variously. For example, when a user is shopping or sightseeing, the computer provides the recommendation services[6][8]. There is the Personal Agent where a computer recognizes the user situation from the state of in-home appliance’s information, user location or action history and buying information from shops in response to provide the user with services[1]. There is the Matching Services Model[7]. It provides valuable information services to a user. The system compares what the user wants to do with what a provider can do. A sensor is installed in the home, the work environment and so on. The system provides the user with the best service according to the user’s action history and domestic sensor information[2][3][9]. These ubiquitous services are provided to users only when they are conscious of their purposes. However, a user may have actions that the user is not conscious of. If a system reminds a user that the user should do these actions when the user is not conscious of them, the user will appreciate this service.

We are aiming at achievements of the Unconscious Computing to provide users with ubiquitous services even if they are not conscious of ubiquitous computer environments. In the Unconscious Computing, a user’s cellular phone recognizes the actions that the user should do. It is assumed that a server is installed in areas such as station. The server knows the numbers of resources such as buy-ticket places or ticket reservation counters and manages the usage states of the resources in the area. When a user enters the area, the user’s cellular phone provides the user with services which let him execute actions to require the resources. However, when the user reserves the service in the area, another user who requests the same service from the same area might exist. Since the number of resources such as buy-ticket places in an area is limited, it is difficult to provide the services to all users. To solve this problem, this paper proposes the Action Arbitration Function to arbitrate the competition for resources. The advantage of proposed method is
to schedule it in consideration of the actions that each user should be going to do in the future. We evaluated the proposed method with two experiments. In the result, the proposed method was able to process a lot of actions in short processing time.

The rest of this paper is organized as follows. Section 2 describes the Unconscious Computing and sorts of the issue of the Unconscious Computing. Section 3 describes the proposed method of the Action Arbitration Function. Section 4 evaluates the proposed method with experiments. Section 5 concludes this paper.

2 Unconscious Computing

2.1 Services Proposal on The Spot

In this paper, a service is an execution of action which a user should do, and informing that the action is enforceable to the user. The purpose of this research is to provide the services for a user when the user is not conscious of the service.

First of all, each user has to input the schedule to his cellular phone in advance as shown in Figure 1. The cellular phone recognizes the user situation for the schedule that was input. Then the cellular phone calculates the actions required by the user to achieve the schedule items written in the schedule. For example, the user inputs what he want to prepare for a business trip to the schedule of his cellular phone. His cellular phone calculates the required actions of buying tickets, tour reservation, renewal of his commuter ticket and so on, from the schedule of the business trip.

It is assumed that an area server is installed in each area. In this paper, we assume that specified services are provided to a user when he is in an area. The area server knows the number of resources and manages the states of the resources for usage. In this paper, the resources are equipments of the area that a user uses to execute actions such as ticket machines, ticket reservation counter and so on. When a user enters an area, the area server informs the user about the number of the resources and the usage states of the resources in the area. The user makes the action schedule that he wants to do from those resource information in the area[4].

There may be many people in the same area. The action schedule which the user made is not always executed on schedule if many persons want to use the same resource. To solve this problem, the cellular phone of the user sends the action which the user made to the area server. The area server mediates the action schedule to execute the actions well. The area server sends the mediated action schedule to the user.

The user’s cellular phone provides the user with services based on the mediated action schedule. Since those calculations are performed automatically by the cellular phone and the area server, the user can enjoy the services without considering the state of the resources and competition for the resource with other users. This method can provide the service for user when the user is not conscious of the services.

2.2 Classification of Services

We consider three kinds of services to users as following:

**Envisioned service:** To support a user’s behavior to achieve his purpose of which he is conscious.

**Encountered service:** To remind a user to execute an action of which he is not conscious usually.

**Latent service:** To support a user in case of danger.

The envisioned service is the service that supports the user’s behavior to achieve his purpose. For example, when a user goes on a business trip to Japan, an airplane ticket is necessary. The user’s cellular phone recognizes the user’s situation. When the user comes in the area where an airplane ticket can be bought, his cellular phone says to him “You can buy the airplane ticket here! ”.

The encountered service is a service that reminds the user to execute an action. For example, suppose that the user goes to the office by train by using the commutation ticket every day. When the day to renew commutation ticket is coming, his cellular phone says to him ”Please renew the commutation ticket! ”, even if he is not conscious of renewing it.

The latent service provides the user with necessary services in case of the danger such as fire, earthquake. For example, if a disaster occurs, the user’s cellular phone guides him to safe evacuation routes.
2.3 Contention among users

The purpose of our research is to provide services to user when the user is not conscious of the services. However, when the user reserves the service in the area, another user who requests the same service from the same area might exist. The number of resources such as buy-ticket places in the area is limited. Therefore, it is difficult to provide the services to all users.

The method for providing services efficiently in the limited resources is required.

3 Coordination Based on Scheduling

3.1 Action Arbitration Function

When there are many users in the same area, the competition for resources may occur by the limitation of resources in the area. In this paper, we call the function of the area server to arbitrate the competition for resources as the action arbitration function. The action arbitration function knows the number of resources and manages the usage states of the resources in the area. In addition, the action arbitration function provides the appropriate service to each user according to the current usage states of the resources and the situation of him when many users requested the same service in the area.

The action arbitration function always knows the number of resources and the usage states in the area. When a user has entered the area, the action arbitration function informs his cellular phone about the number of resources and the usage state of the resources in the area. The cellular phone manages the actions that the user should do with the resources, the deadlines and the times when they can be begun. The cellular phone makes the action schedule that the user wants to do in the area from information on the area as shown in Figure 2. The action schedule shown in Figure 3, includes three actions whose required resources are resources 1, 2 and 3, respectively. It also sends the following information to the action arbitration function after making the action schedule:

- The user ID
- The action schedule that the user’s cellular phone made
- The sojourn time of the user in the area
- The action list that the user should do in the area in the future

By receiving the action schedule from users, the action arbitration function compares the action schedules and judges the feasibility of each user’s action schedule. After that, the action arbitration function reschedules the action schedules and informs them to the users.

3.2 Action Rescheduling

When the action arbitration function received action schedules of users, the action arbitration function compares the action schedules firstly. When the competition for the resources will not occur, the action arbitration function judge that each user’s action schedule is feasible. After that, the action arbitration function informs each user that the action schedule is feasible. On the other hand, the competition for the resource occasionally will occur because of the restriction of the resource. When the action schedules where users A and B use resource 2 at the same time are obtained as shown in Figure 4, the competition for resource 2 occurs. In this case, the action arbitration function begins rescheduling them. When the action arbitration function begins rescheduling, it considers the resources for executing actions, sojourn time of each user in the area, the deadlines of the actions and the times when the actions can be begun. The action arbitration function rewrites the action schedule.
of user A in the order of resource 1, resource 2 and resource 3 as shown in Figure 4. Similarly, the action arbitration function rewrites the action schedule of user B in the order of resource 3, resource 1 and resource 2. The action arbitration function informs each user about the rewritten action schedule.

Another example of the competition for resource 2 is shown in Figure 5. In this case, the action arbitration function begins rescheduling the action schedules because the competition of the action schedules occurs. However, the action arbitration function cannot reschedule them within the sojourn time of each user because the period in which each user uses the resource 2 is very long. Therefore, either user A or user B must abandon the action with the resource 2. In this case, the action arbitration function provides the resource to the user who has a tight action in consideration of the actions that each user should be going to do in the future. The detail of the tight action is described in the next subsection.

3.3 Tight Action

The action arbitration function begins rescheduling when the competition for resources occurs. If feasible schedules cannot be calculated to resolve the competition, the action arbitration function provides the resource to the user who has a tight action in consideration of the actions that each user should be going to do in the future. A tight action is the action which cannot be executed before its deadline if some of the predecessor actions are postponed to resolve the competition.

As shown in Figure 6, suppose that user A plans to renew his commutation ticket at the station today. Also, user A plans to reserve a tour tomorrow. Similarly, user B plans to renew his commutation ticket at the station today. Users A and B come to the station at the same time and try to renew their commutation tickets according to their schedules. It is assumed that today only user B executes the renewal of the commutation ticket because the commutation tickets can be renewed by only one person in the station today. If user A renews the commutation ticket in today’s sojourn time of the station, he will postpone the renewal of the commutation ticket until tomorrow. The tour reservation is assumed to be able to do from tomorrow. Since the renewal of the commutation ticket is shifted to tomorrow, the tour reservation scheduled on tomorrow will be postponed day after tomorrow. In this case, user A’s tour reservation is a tight action because it is postponed after its deadline by shifting the renewal of the commutation ticket into tomorrow. The action arbitration function provides priority for a resource to the user who has a tight action.
3.4 Priority of a Tight Action

This subsection explains the method of finding the user who has a tight action. We use the following notation for each user.

\( T_n \): The worst execution time of the \( n \)th action which the user should do in the area

\( D_n \): The deadline of the \( n \)th action

\( L \): The past certain fixed period

\( v \): The frequency of visiting the area during the period \( L \)

\( S_a \): The average sojourn time of the user in the area

\( P_n \): The predicted frequency of visiting the area of the user during the period from \( C \) to \( D \)

\[ P_n = \left( \frac{D_n}{C} \right)^{v/L} \]

\( E_n \): The number of times in which the \( n \)th action can be postponed

Each action is characterized by its action name, deadline and the worst execution time as shown in Figure 7. In this case, each \( T_n \) corresponds to \( T_1 = 5 \), \( T_2 = 10 \) and \( T_3 = 15 \). If the user has visited the area where the actions can be executed five times in a week in the past, each \( P_n \) corresponds to \( P_1 = \left( \frac{14}{11} \right)^{2/5} = 2 \), \( P_2 = \left( \frac{16}{11} \right)^{3/5} = 3 \) and \( P_3 = \left( \frac{17}{11} \right)^{4/5} = 4 \). The average sojourn time of this area of the user is assumed to be 20 minutes. The user will visit the area twice by \( D_1 \), because \( P_1 \) is 2. It can be a forecast of the user stays in the area for 20 minutes. The action arbitration function calculates how many times action 1 can be executed in these periods. Action 1 can be executed 7 times by \( D_1 \). Therefore, it corresponds \( E_1 = 7 \). For action 2, the user will visit the area three times by \( D_2 \). It is necessary to execute actions 1 and 2 by the deadline of action 2. Even if 2 actions are executed, action 2 can be executed 4 times more by the deadline of action 2. Therefore, it corresponds \( E_2 = 4 \). Similarly, action 3 corresponds \( E_3 = 2 \). If there is a user who has an action with \( E = 0 \), the action is regarded as a tight action because the number of action is little that can be executed by the action deadline. Therefore, the action arbitration function provides priority for a resource to the user who has an actions with \( E = 0 \). If there are no users who have action with \( E = 0 \), the action arbitration function provide priority for the resource to the users who has the action with smallest of \( E \).

4 Evaluation

4.1 Compared Method

We conduct two experiments to study the effectiveness of the proposed method by comparing it with the method proposed in [4]. Our proposed method checks the tight actions of the users who compete for resources and provides the priorities for resources to the users who have tight actions. The method proposed in [4] checks the actions with short deadlines among users which compete for resources and provides the priorities for resources to the user who has the action with the shortest deadline. It is evaluated as the best scheduling in the scheduling algorithms.

The first experiment evaluates the processing time of scheduling. The second experiment evaluates the number of actions that can be executed by their deadlines.

4.2 Processing Time

The first experiment evaluates the processing times of scheduling by the two methods. We measured the processing time as an experimental methodology assuming that there are 10 users, 20 users, 50 users and 100 users in the area. The worst execution times of actions, the deadlines of actions, action schedule, action list that the users should do in the area, predicted frequency that the users will visit the area and average sojourn time of the area are necessary for finding the tight actions. The experiment assumes that there are 20 actions for each person. The worst execution times of the actions assumed to be 3 minutes in the maximum, 1 minutes in the minimum and 1.5 minutes in the average. It assumes that each user visits the area 60 times in a week. The average sojourn time of the area is assumed to be 20 minutes. The experiment assumes that the competition occurs even if the system is rescheduling. The number of resources in the...
area assumed to be 100. The server has 1.60GHz Intel Celeron M Processor and 512MB RAM. We carried out scheduling by the two methods for 50 times with the same data and evaluated the average and worst processing times. Figure 8 shows the experimental result.

As for the two methods, it is understood to have the almost same average processing time from Figure 8. The worst processing times are shorter than 100 milliseconds. The proposed method takes more processing time than compared method because the proposed method finds tight actions. However, the proposed method and compared method take the almost same processing times even if there are 100 users in the area. Therefore, the proposed method can be scheduled at time as short as the compared method.

### 4.3 Execution Achievement Rate

The second experiment evaluates the number of actions that can be executed by their deadlines by using the two methods. The following equation is used for evaluation of methods:

\[
\text{The Execution Achievement Rate} = \frac{N}{K}.
\]  

(1)

where N and K are the number of actions executed when there are multiple users in the area and the number of actions executed when there is one user in the area, respectively. We calculate the execution achievement rate of the two methods. We compare the numbers of the executable actions of the two methods by using the execution achievement rates. We configure the experiment by eight users. Table 1 shows each user’s parameters.

The experiment has an assumption that eight users come to the area one or five times in five days. We calculated the execution achievement rate of each user’s action. The following restrictions were set in the experiment.

The number of resources that can be used in the area is three.

The eight users visit the area at the same time.

There are 6 kinds of actions of the users as shown in Table 2.

It is considered that the actions to which service was provided were executed.

The actions of each user with 21 actions includes three actions for Action 1,2 and 3 and four actions for Action 4, 5 and 6, respectively. The actions of each user with 7 actions includes one action for Action 1, 2, 3, 5, 6 and 2 actions for Action 4.

We experimented in all combinations of two users among the eight users. The experimental data includes 247 cases in total. Figure 9 shows the experimental result. In Figure 9, the horizontal axis represents the number of people in the area. The figure shows the average of the execution achievement rates. As the number of people increases in the area, the execution achievement rates lower. The execution achievement rate of the proposed method is higher than that of the compared method. Particularly, the more the number of people increases in the area, the more the deviation of the execution achievement rates between the two methods increases. When there are two users in the area, the execution achievement rate of the proposed method is 4% higher than that of the compared method. When there are eight users in the area, the execution achievement rate of the proposed method is 20% higher than that of the compared method. This means that the more the number of people increases in the area, the more the proposed method processes

![Figure 8: Comparison by Processing Time](image)

Table 1: User’s Parameters

<table>
<thead>
<tr>
<th></th>
<th>Number of actions that should be executed in area</th>
<th>Sojourn time in the area (minutes)</th>
<th>Visited days of the area</th>
</tr>
</thead>
<tbody>
<tr>
<td>User A</td>
<td>21</td>
<td>20</td>
<td>all days</td>
</tr>
<tr>
<td>User B</td>
<td>21</td>
<td>20</td>
<td>first day</td>
</tr>
<tr>
<td>User C</td>
<td>21</td>
<td>5</td>
<td>all days</td>
</tr>
<tr>
<td>User D</td>
<td>21</td>
<td>5</td>
<td>first day</td>
</tr>
<tr>
<td>User E</td>
<td>7</td>
<td>20</td>
<td>all days</td>
</tr>
<tr>
<td>User F</td>
<td>7</td>
<td>20</td>
<td>first day</td>
</tr>
<tr>
<td>User G</td>
<td>7</td>
<td>5</td>
<td>all days</td>
</tr>
<tr>
<td>User H</td>
<td>7</td>
<td>5</td>
<td>first day</td>
</tr>
</tbody>
</table>

We experimented in all combinations of two users among the eight users. The experimental data includes 247 cases in total. Figure 9 shows the experimental result. In Figure 9, the horizontal axis represents the number of people in the area. The figure shows the average of the execution achievement rates. As the number of people increases in the area, the execution achievement rates lower. The execution achievement rate of the proposed method is higher than that of the compared method. Particularly, the more the number of people increases in the area, the more the deviation of the execution achievement rates between the two methods increases. When there are two users in the area, the execution achievement rate of the proposed method is 4% higher than that of the compared method. When there are eight users in the area, the execution achievement rate of the proposed method is 20% higher than that of the compared method. This means that the more the number of people increases in the area, the more the proposed method processes
the actions more than the compared method. Therefore, the proposed method efficiently schedules actions when there are a lot of users in the area.

Table 2: Type of Action

<table>
<thead>
<tr>
<th>Action</th>
<th>Deadline</th>
<th>The worst execution time (minutes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action 1</td>
<td>one week</td>
<td>1</td>
</tr>
<tr>
<td>Action 2</td>
<td>one week</td>
<td>2</td>
</tr>
<tr>
<td>Action 3</td>
<td>one week</td>
<td>3</td>
</tr>
<tr>
<td>Action 4</td>
<td>two week</td>
<td>1</td>
</tr>
<tr>
<td>Action 5</td>
<td>two week</td>
<td>2</td>
</tr>
<tr>
<td>Action 6</td>
<td>two week</td>
<td>3</td>
</tr>
</tbody>
</table>

![Figure 9: Ratio to which Action is Processed](image)

5 Conclusion

We are aiming at the achievement of the Unconscious Computing which is a paradigm to provide users with ubiquitous services even if they are not conscious of a ubiquitous computer environment. This paper proposes the Action Arbitration Function to arbitrate the competition of resources in the Unconscious Computing.

The action arbitration function compares the action schedules firstly when the action arbitration function received action schedules of users. When the competition the resource occurs, the action arbitration function begins rescheduling. If the competition for the resource occurs after the rescheduling, the action arbitration function provides the priority for the resource to the user who has a tight action in consideration of the actions that each user should be doing to do in the future.

We have undergone two kinds of experiments to compare our proposed method with the method proposed in [4]. The proposed method checks tight actions among users who compete for resources and provides priority for resources to the user who has a tight action. The compared method checks the actions with short deadlines among users who compete for resources and provides the priorities for the resources to the users who have the action with the shortest deadline.

The first experiment evaluates the processing times of rescheduling by the two methods. In the result, the proposed method can be scheduled at time as short as the compared method. The second experiment evaluates the number of actions that can be executed by their deadlines. In the result, as the number of people increases in the area, the more actions of the proposed method processes than the compared method. The proposed method was able to process a lot of actions at short processing time. Therefore, the proposed method efficiently schedules when there are a lot of users in the area. In the future, we will experiment further by using the proposed method and we will improve the utility of this method.

References:


