CONSISTENT DIALOGUE ACROSS CONCURRENT TOPICS
BASED ON AN EXPERT SYSTEM MODEL

Bor-shen Lin¹, Hsin-min Wang², and Lin-shan Lee¹,²
¹ Department of Electrical Engineering, National Taiwan University
² Institute of Information Science, Academia Sinica
Taipei, Taiwan, Republic of China
e-mail: hsl@speech.ee.ntu.edu.tw

ABSTRACT

There have been many working spoken dialogue systems, but very few of them are able to handle concurrent topics consistently if the user freely surfs among different topics at any time, because it’s really difficult to resolve semantic ambiguities and check the knowledge consistencies among correlated topics. The issues include not only how to handle the information across multiple topics and domains with shared slots, but how to infer a reasonable dialogue state considering the new information or inconsistencies among knowledge structures after some correlated topics are activated.

In this paper, a plan-based dialogue control mechanism that is capable of handling the very complicated dialogue problem is proposed. This mechanism can keep the knowledge consistent among multiple topics and domains when the topic is switched. Also, by representing the problem-solving procedures of different goals as loadable trees, the dialogue strategies are easy to maintain and modify, and can be dynamically adapted to the users during the dialogue.

1. INTRODUCTION

There have been many working spoken dialogue systems, but very few of them show the ability to handle concurrent topics consistently if the user freely surfs among different topics at any time, because it’s really difficult to resolve semantic ambiguities and check the knowledge consistencies among correlated topics [1,2]. In a travel plan dialogue, for example, the user may temporarily deviate from a multi-stage goal of train ticket reservation, ask for the air flight information and the weather on a specific date in order to make his decision, and then go back to the suspended train ticket reservation topic again. In the meanwhile, some other topics such as hotel reservation and car rental previously activated may still remain unsolved. The difficult issues include not only how to handle the information across multiple topics and domains with shared slots, but how to infer a ‘reasonable’ dialogue state considering the new information or inconsistencies among knowledge structures after some correlated topics and domains are activated.

Conventionally, the finite state networks were used as the dialogue modeling schemes. The advantage is that they are predictable and easy for implementation. However, because they are deterministic, all probable dialogue states should be defined and the conditions for state transitions should be determined manually beforehand. Thus, they are suitable only for those tasks with proper amount of dialogue states and limited complexity of dialogue, such as the system-initiative dialogue systems. When concurrent and correlated topics and arbitrary user initiative are considered in the multi-domain environments, both the dialogue states and knowledge structures swell tremendously, and it becomes infeasible to implement such systems based on such schemes.

On the other hand, the so-called plan-based dialogue modeling schemes were also proposed. They are based on the assumption that persons generally have goals and plans in mind when they interact with other persons or machines, and the purpose of a dialogue is to recognize such goals or plans, and to produce effects corresponding to the purpose of the plans [3]. Several working systems based on similar schemes have been developed [4-8]. However, some problems such as how to handle concurrent topics consistently have not yet been concluded for such modeling schemes [7,8].

In this paper, a plan-based dialogue control mechanism that is capable of handling the very complicated dialogue problem is proposed. This mechanism can keep the knowledge consistent among multiple topics and domains with shared slots by an universal confirmation and correction scheme when the topic is switched. Also, by representing the problem-solving procedures of different goals as loadable trees, the dialogue strategies are easy to maintain and modify, and can be dynamically adapted to the users during the dialogue. The proposed mechanism has been successfully applied to a spoken dialogue system for integrated travel services across correlated topics and domains, including train ticket reservation, weather information, etc., and its capabilities in handling concurrent topics consistently in a complicated dialogue environment have been successfully verified.

2. ISSUES OF CONSISTENCY

To understand the issues of consistency, let’s first consider an example dialogue in a multi-domain multi-topic environment as shown in Figure 1, where the indexes Sx and Cx denote the server and the customer respectively. This example demonstrates how people may acquire multi-domain multi-topic integrated services through a spoken dialogue. In S2, the server keeps going on the reservation topic, but the customer switches the topic actively in C2 in order to decide what date to go. After giving a reply, the server switches back to the reservation topic because the user does not take the initiative. In C4, the customer switches to another topic of ticket price, keeps going on this topic for several dialogue turns, and then actively switches back to the reservation topic in C7. From S9 to S11, the customer happens to find that weather is an important issue, so he changes the departing date after asking about the weather information, and at the same time the server understands this change and helps the customer to correct his goal. From C11 to S12, the customer asks about the arrival time, and the server replies the message that no bus is available in addition to the arrival time. Therefore, in C13, the customer hesitates to make his decision because the arrival time is late, and starts to ask for hotel reservation. The server doesn’t reply
In the following sections, we will introduce a dialogue control mechanism for handling such problems. The related issues include:

- User initiative problem: How should the user’s initiatives be handled consistently?
- Understanding error problem: How may the understanding errors be handled according to the information consistency?
- Initiative taking problem: When will the server take the initiative, and how may the server take the initiative consistently?

In the following sections, we will introduce a dialogue control mechanism for handling such problems.

Figure 1. Example dialogue in a multi-domain multi-topic environment

S1: How may I help you?
C1: I’d like to make a reservation for ticket from Taipei to Tainan.
S2: What date would you like?
C2: What day is it tomorrow?
S3: Tomorrow is Friday. . . . .
C3: Would you like to go tomorrow?
S4: What’s the price of Zhi-chiang type?
S5: Ticket price of Zhi-chiang type for adult from Taipei to Tainan?
C5: Right.
S6: The price of Zhi-chiang type from Taipei to Tainan is ... 729 for adult.
C6: And Fu-hsing type?
S7: The price of Fu-hsing type is 573.
C7: I want two tickets of Fu-hsing type.
S8: How about the train 1012 departing at 7:53 p.m. tomorrow?
C8: A moment, how about the weather in Tainan?
S9: The weather in Tainan is rainy tomorrow.
C9: And the day after tomorrow?
S10: The weather is sunny the day after tomorrow. . . . .
C10: Do you want to change as the day after tomorrow?
S11: How about the train 1016 departing at 8:35 p.m. the day after tomorrow?
C11: What’s the arrival time of this train?
S12: The arrival time is 11:45 at night. There is no bus when you arrive there. Maybe you have to take the taxi.
C12: Would you please reserve some hotel for me?
S13: Would you still like to reserve the train tickets?

Figure 2. Dialogue manager based on an expert system model

3. DIALOGUE MANAGER

Our software platform for developing multi-topic, multi-domain and multi-modal spoken dialogue systems is based on a finite state automaton, described by a dialogue description language, which may integrate all the software modules, including a speech recognizer, a natural language processor, and so on, through the dynamically loadable operators. Each input utterance is first recognized by the speech recognizer, and the output word graph is applied to a hierarchical tag-graph search in the natural language processor so as to generate top N tag sequences with associated parsing trees [9], which are then transcribed into semantic slots sequentially. The semantic slots are passed to a dialogue manager to decide the dialogue state by logical reasoning based on the given knowledge. Afterwards the response sentence is generated in the automaton according to the new dialogue state and associated information, and then synthesized to speech.

The dialogue manager is a prolog-style theorem prover based on the expert system model, consisting of the inference engine, the explanation subsystem, and the dynamic database as shown in Figure 2. It serves as an intelligent consultant, which can infer the proper dialogue state by automatic reasoning based on all available information.

3.1 Dynamic Database

The dynamic database consists of the state of knowledge, the dialogue history, the database for context sensitive help, and the inference trees for different topics in multi-domains. The state of knowledge is decided according to some flags including the correction and contradiction flag, the active topic and focus, the correction scope, and three kinds of semantic slots including the current slots, the goal slots, and the correction slots. The active topic is the current topic under discussion, while the focus is the condition/event of the active topic not yet valid for achieving the goal. According to the (topic, focus) pair, the context sensitive help can be accessed. The correction scope defines the currently correctable slots. Only those slots in this scope are allowed to be overwritten.
Figure 3. Example of inference trees for three topics: ticket reservation, weather information, and ticket price.

Figure 4. A robust confirmation and correction scheme

4.1 Understanding Error
Because of imperfect speech recognition, understanding errors are intrinsic in all spoken dialogue systems. Here the spoken dialogue is in fact very similar to a communication channel. In a conventional communication channel, usually some redundant bits in each packet are used to detect or correct errors, while the protocol may be designed to detect and retransmit the lost packets. In a spoken dialogue, similar mechanisms can also be used to detect or correct understanding errors. The high-level knowledge such as semantic information or knowledge consistency can serve error-check within an utterance, while the confirmation and correction behavior implies the protocol. Assume the current system prompt for confirmation is that “Do you want to go to Taipei on tomorrow morning?”, and the user’s reply is recognized as “No, I want to go tomorrow.”. If the user is rational, such a logic contradiction can be used to judge that understanding errors occur within this utterance and the decoded path should be discarded. However, when inconsistency occurs across utterances, it’s hardly possible to determine this is because the user changed his mind, the system mis-understood the current utterance, or the user wanted to correct the misunderstanding errors in previous utterances. Therefore, a confirmation and correction scheme is designed to handle such situations robustly, as shown in Figure 4. For each utterance, the recognized semantic slots are first saved in the current slots. Afterwards, the consistency between the current slots and the goal slots is checked, and the current slots are merged into the goal slots if no inconsistency occurs. Otherwise, the current slots are copied into the correction slots, and the dialogue enters the correction mode and asks the user to reconfirm the correction slots. If the current slots in the new utterance are consistent with the correction slots, the correction is confirmed and the correction slots are therefore allowed to update the goal slots. Otherwise, the correction slots are simply
discarded. The state transition table for confirmation is shown in Table 1. ‘Yes’ or ‘No’ represents that the word ‘Yes’ or ‘No’ is recognized in the current utterance respectively, while ‘None’ represents that neither ‘Yes’ nor ‘No’ is recognized. The ‘Consistent’, ‘Inconsistent’ or ‘None’ for the confirmation mode represents that the information slots are consistent, inconsistent or no slot is recognized within the correction scope when comparing the current slots with the goal slots, while those for the correction mode are derived by comparing the current slots with the correction slots. For example, if the system prompts “Do you want to go from Tai-Chung to Taipei at 6 o’clock in the afternoon tomorrow?”, and the user responses “No, I’d like to go to Tainan.”, then the transition state is ‘No+Inconsistent’ because the ‘No’ is recognized and the information between ‘Taipei’ and ‘Tainan’ is inconsistent. The dialogue thus enters the correction mode and then prompts the user with “Do you want to go to Tainan?” for reconfirmation. Such a double-checked mechanism can prevent the goal slots from being directly updated regardless of the possible understanding errors in the current utterance. When many slots with errors are to be confirmed, the increasing errors may confuse the user seriously, and perhaps lead the dialogue to complete failure. With the correction mode, the goal slots are updated more smoothly, and the user can therefore correct the misunderstandings slots more easily.

<table>
<thead>
<tr>
<th>mode</th>
<th>Conf.</th>
<th>Inconsistent</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>ConfirmOK</td>
<td>Reject</td>
<td>ConfirmOK</td>
</tr>
<tr>
<td>No</td>
<td>Reject</td>
<td>Correction</td>
<td>Confirm</td>
</tr>
<tr>
<td>None</td>
<td>ConfirmOK</td>
<td>Correction</td>
<td>Repeat</td>
</tr>
<tr>
<td>Yes</td>
<td>Update</td>
<td>Correction</td>
<td>Repeat</td>
</tr>
<tr>
<td>No</td>
<td>Reject</td>
<td>Correction</td>
<td>Repeat</td>
</tr>
</tbody>
</table>

Table 1. State transition table for confirmation/correction.

### 4.2 Initiative Taking

To handle concurrent topics consistently, the state machine for each topic is designed as shown in Figure 5. Initially, all topics are in the idle state. When a topic is activated, it enters the activated state and stays at this state if the goal of this topic is not achieved. When a transaction for the active topic is made, the topic enters the finished state, and the system waits for the user to take the initiative. If the user keeps going the same topic (e.g. C6 and C9 in Figure 1), this topic is reactivated. If the user switches to another topic, this finished topic then enters the suspended state, and the system waits for the user to switch the topic back to the active state.

**Figure 5.** The state machine for each topic

activate another idle topic (C2, C4, C11, and C12), or actively switch to a suspended topic (C7). When the user switches the topic, the system should decide whether or not to activate the desired topic according to topic priority and complexity. For example, in Figure 1, the system decides to enter the new topic directly in S3, S5, and S12, but tries to prevent the user from entering another complicated domain in S13. Though the knowledge consistency between the ticket reservation topic and the hotel reservation topic can be kept in our mechanism, it’s proper to suggest the user not to do so. If the system decides to switch the topic directly, the active topic with the associated state of knowledge is then pushed into the dialogue history, and the desired topic is activated with the probable inherited slots.

### 5. CONCLUSION

We have presented a dialogue control mechanism that is capable of handling issues of knowledge consistency for concurrent topics in a multi-domain multi-topic environment. Such a mechanism has been applied to and verified on integrated travel services using text mode input where the timeout mechanism described in Section 4.2 is implemented by simulation. Dialogue behavior as the example in Figure 1 can all be handled with persistent and consistent state of knowledge while flexible and dynamic dialogue strategies can be also supported. Such a dialogue control mechanism can enhance the spoken dialogue systems in intelligent integrated services.

### 6. REFERENCES

[1] Seneff Stephanie, etc., “Multimodal Discourse Modelling in a Multi-User Multi-Domain Environment”, ICSLP96, A224.PDF.


