Handling Multiple Topics in Spoken Dialogue Systems Using Inference Trees

Bor-shen Lin
Department of Electrical Engineering, National Taiwan University,
Taipei 106, Taiwan, Republic of China
E-mail: bsl@speech.ee.ntu.edu.tw

Hsin-min Wang
Institute of Information Science, Academia Sinica,
Taipei 115, Taiwan, Republic of China
E-mail: whm@iis.sinica.edu.tw

Abstract

Recently developed spoken dialogue systems usually use the deterministic finite state network for dialogue modeling, which has not good domain portability and high flexibility for handling multiple topics. In this paper, a flexible plan-based dialogue control mechanism based on the expert system model is proposed. This mechanism not only has good domain portability and high flexibility for handling multiple topics, but is capable of dynamically adapting dialogue strategies during the dialogue. Several issues specific to spoken dialogue systems such as the robust confirmation/correction scheme and the timeout mechanism for taking initiatives are considered and can be well handled in this mechanism.

1 Introduction

Conventionally, the dialogue modeling in spoken dialogue systems is based on the deterministic finite state network [1], in which the transitions of dialogue states, determined by the speech act types and current states of knowledge, should be hard-coded in the script file using the dialogue description language. Good portability for handling multiple topics and multi-domains can not be achieved based on such dialogue control mechanism because, considering the complicated task structure with various confirmation and correction schemes necessary for spoken dialogue systems, it's really difficult to modify the dialogue control flow for new domains or new topics. On the other hand, the dialogue control based on stochastic modeling [2], in which the dialogue states are defined according to a large dialogue corpus while the transitions are determined by stochastic models, has also been suggested. Neither does this modeling scheme show good domain portability, because the dialogue corpora for training stochastic models in different domains are difficult to acquire. Some spoken dialogue systems based on so-called plan-based dialogue modeling schemes, in which the task structure is usually expressed as and-or graphs or dialogue trees representing the problem solving conditions, were also proposed [3-5]. However, some problems such as how to handle concurrent topics consistently have not yet been concluded for such modeling schemes [6,7].

In this paper, a flexible plan-based dialogue modeling scheme based on the expert system model [8] is proposed. This scheme, called dialogue expert, utilizes the modular features of the expert system to separate the domain/state dependent dynamic database from the domain/state independent inference engine and explanation subsystem. The dynamic database can be further decomposed into the state-dependent history stack and state parameters, and the domain-specific inference trees that represent the dialogue strategies for different topics. Domain portability can be achieved in this scheme because, in the dialogue expert, only the domain-specific inference trees need to be modified when the domain is switched. Also, by representing the dialogue states as the tree structures, the number of topics can be increased simply by increasing the number of trees while the topic-switching can be well manipulated by the domain/state independent inference engine, so flexibility for handling multiple topics is achieved, too. Furthermore, multiple dialogue strategies can be easily supported because the tree structures of different strategies can be easily modified, maintained and dynamically adapted even during the dialogue. All the details of this scheme will be illustrated in the following sections.

2 System Overview

Our software platform for developing multi-topic, multi-domain and multi-modal spoken dialogue systems is shown in Figure 1. The kernel of this platform is a finite state automaton, described by a dialogue description language denoted as DDL, which integrates all the

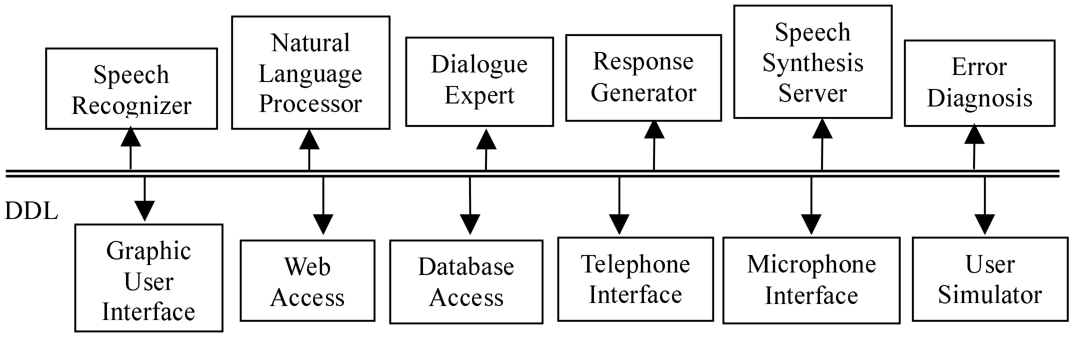

Figure 1. Software platform for developing spoken dialogue systems

software modules, including speech recognizer, natural language processor, and so on, through the dynamically loadable operators defined by the system developers. Each input utterance is first recognized by the speech recognizer, and the output word graph is applied to a hierarchical tag-graph search in natural language processor so as to generate top N tag sequences with associated parsing trees [9]. These tag sequences are then transcribed into semantic slots sequentially by the DDL. The semantic slots are then passed to the dialogue control mechanism, called dialogue expert as shown in Figure 1, to decide the next dialogue state by logical reasoning based on the given knowledge. Afterwards the response sentence is generated by the response generator according to the next dialogue state with associated information, and then synthesized to speech. This integration platform and the carefully partitioned software modules make the independent development and quick integration possible, and various programming languages with associated techniques can also be used for different modules respectively. The detailed operations of the dialogue expert will be further described in the following section.

3 Dialogue Expert

The dialogue expert, which consists of the inference engine, the explanation subsystem, and the dynamic database as shown in Figure 2, is built based on the idea of the expert system model. It serves as an independent and intelligent consultant, which can infer the proper dialogue state by automatic reasoning based on available information from the natural language processor.

3.1 Dynamic Database

The dynamic database consists of the state parameters, the history stack, and the inference trees for different topics. The state parameters include some flags, such as the correction flag or the contradiction flag, the active topic and focus, the correction scope, and three kinds of semantic slots, such as the current slots recognized in the current utterance, the goal slots containing consistently accumulated slots, and the correction slots to be used for correction if any inconsistency between current slots and goal slots occurs. Necessary state parameters are pushed

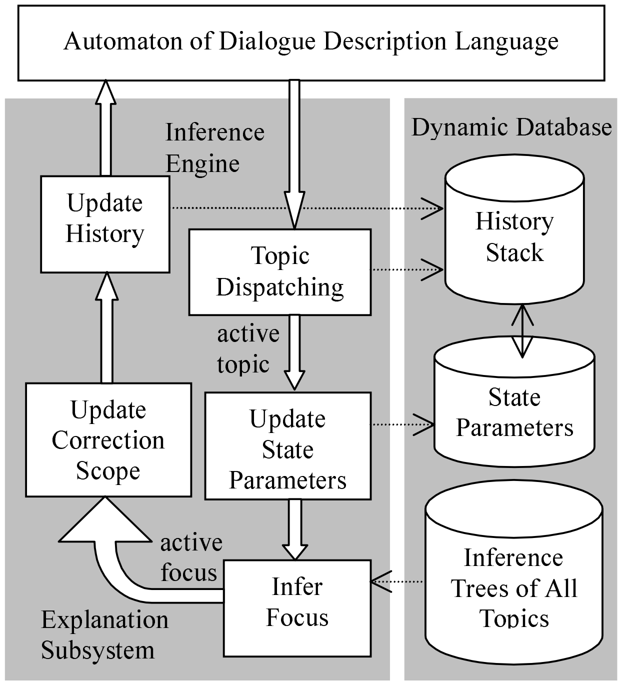

Figure 2. Dialogue manager using expert system model

into the history stack in each dialogue turn, so both the state parameters and history stack are state-dependent. The domain-specific inference trees, expressed as and-or graphs, represent the problem solving procedures of all the probable communication goals. The ticket reservation, for example, requires the information about from where to where, which train to take, and the kind and number of the tickets to be reserved, and all the retrieved information should be confirmed before a reservation is made. So, in an inference tree of the ticket reservation domain as shown in Figure 3, the root node has four children nodes representing the necessary conditions respectively, and is therefore marked as an and-type node. On the other hand, 'which train to take' is decided by either the desired time/type or the train number, so the 'which_train' node, with two children nodes 'by_time' and 'by_train_no', is marked as an or-type node. Besides the and/or types, some other attributes such as 'confirm', 'optional', or 'slot' attributes, can also be bound to the

Knowledge in dynamic database:
and(ticket_reserv,[from_to, which_train, conf_reserv]).
and(from_to,[from_station, to_station, conf_from_to]).
or(which_train,[by_time, by_train_no]).
and(by_time, [date, time, type, conf_by_time]).
slot(date).
slot(time).
confirm(conf_from_to).
confirm(conf_by_time).
.
.

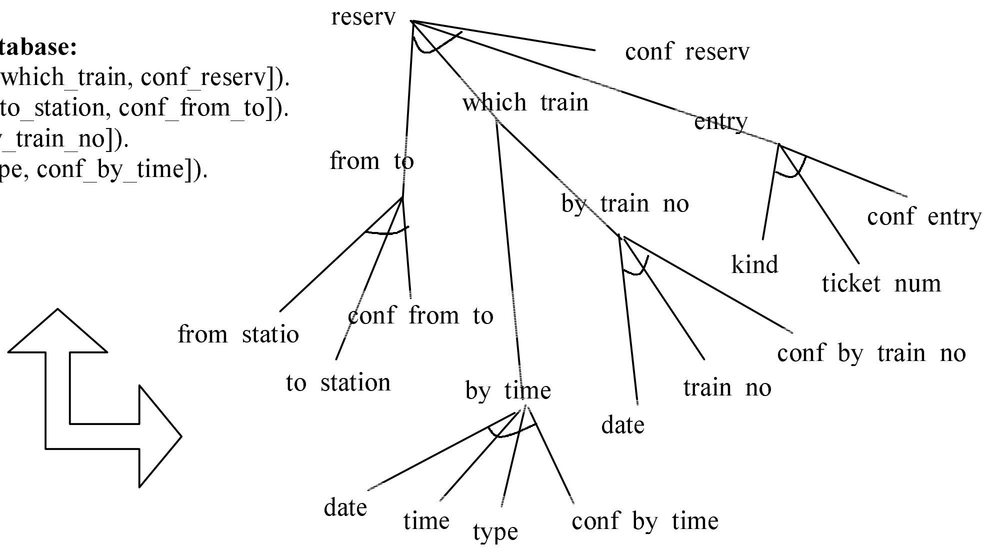

Figure 3. An example of inference tree of the ticket reservation task

node with different functions controlled by the inference engine. With the inference trees and various node attributes, all the topics with complicated logical structures can be well represented, and thus dialogue strategies can be modified flexibly. Also, it is possible to dynamically choose various strategies according to different inference trees even during the dialogue.

3.2 Inference Engine and Explanation Subsystem

After each utterance is recognized, the inference engine should first choose the active topic according to the semantic slots and speech act types, and then update the state parameters. Afterwards, the inference engine further derives the truth/falsity states for all nodes of trees, and then infers the focus, i.e., the condition node not yet solved, in the explanation subsystem so as to decide how the dialogue state transits. Then, the correction scope for semantic slots is dynamically generated according to the active focus and the current slots, and the necessary state parameters are pushed into the history stack. Several interesting points about the inference engine are discussed in the following. First, according to the active topic and focus, the on-line help sensitive to the dialogue context can be easily achieved. In fact, the topic and focus here are actually analogous to the dialogue box and active edit item in the Windows environments, so the concept of context-sensitive help can be easily applied. Second, during the inferring, knowledge consistency across dialogue turns should be checked and kept, which is quite complicated but very important for spoken dialogue systems because understanding errors due to recognition errors often occur. For example, assume the current dialogue state is confirming several semantic slots, such as “Do you want to go to Taipei tomorrow morning?”, and the user’s reply is recognized as ‘no’ but with consistent information within the correction scope, such as ‘no’ with ‘tomorrow morning’. Then, the dialogue expert should judge from such a logic contradiction that there must be understanding errors coming from recognition errors, therefore reject the current tag sequence, and ask for transcribing the next tag sequence. However, when inconsistency across dialogue turns occurs, it’s usually very hard to determine this is because the user changed his mind or the system mis-understood what the user said. An universal confirmation and correction scheme integrated in the inference engine that can robustly handle such situations is therefore necessary, which will be discussed in the next section.

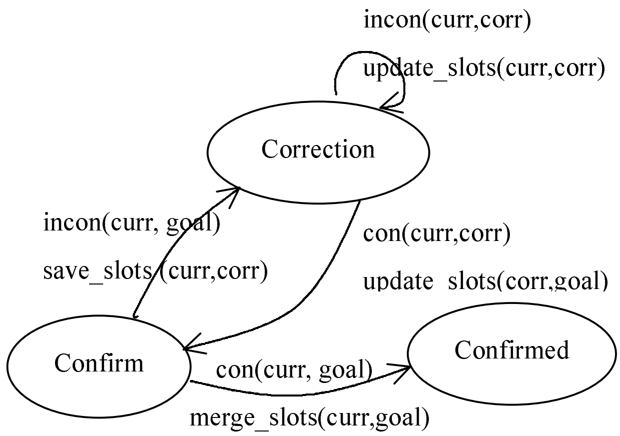

Figure 4. A robust confirmation and correction scheme

4 Confirmation and Correction Scheme

A double-checked confirmation and correction scheme is specially designed 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 goal slots are 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 the confirmation node is shown in Table 1. 'Yes' or 'No' represent the words 'Yes' or 'No' is recognized in current utterance respectively, while 'None' represents neither 'Yes' nor 'No' is recognized. The 'Consistent', 'Inconsistent' or 'None' for the confirmation mode represents the information are consistent, inconsistent or no slot is recognized within the correction scope when comparing current slots with goal slots, while those for the correction mode are derived by comparing current slots with 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. The design idea here is to update the goal slots through a double-checked mechanism. Without the correction mode, the goal slots will be updated directly regardless of possible recognition errors, and the user may be confused due to many errors in slots. For example, if "No, I'd like to go to Tainan" is mis-recognized as "No, I want to go today morning", then the system may prompt the user with "Do you want to go from Tai-Chung to Taipei at 6 o'clock in the morning today?". Such a prompt imposes heavy load on the user to remember all the information and too many inconsistencies between the prompt and the user's goal, which usually leads the dialogue to complete failure, especially in spoken dialogue. On the other hand, with the correction mode, instead of the very complicated prompt as the above example, the system will first prompt "Do you want to go today morning?", the user can easily correct the mis-recognized slots and the dialogue can successfully progress to achieve the user's goal

5 Topic Switching

When the user switches the topic actively, the topic dispatching in the inference engine pushes the active topic with associated state parameters into the history stack, and activates the new topic with inherited semantic slots according to the topic priority. When the transaction for the active topic is accomplished, the timeout mechanism waits for the user to take the initiative, otherwise, the system will pop the topic according to the topic priority. For example, the user can keep going on the same topic and provide some different values of semantic slots for the next transaction by taking the initiative within the timeout constraint. If the user doesn't decide what to do for a proper period of time, the system then pops the suspended topic according to the topic priority. Such a timeout mechanism may make the spoken dialogue progress more naturally and smoothly when the topic switches.

		Consistent	Inconsistent	None
conf. mode	Yes	Confirmed	Reject	Confirmed
	No	Reject	Correction	Confirm*
	None	Confirmed	Correction	Repeat
corr. mode	Yes	Update	Reject	Update
	No	Reject	Correction	Confirm*
	None	Update	Correction	Repeat

Table 1. State transition table for confirmation/correction (* denotes the same node but with a clearer prompt).

6 Conclusion

We have presented a dialogue control mechanism with good domain portability and high flexibility for handling multiple topics. Several issues specific to spoken dialogue systems such as the robust confirmation and correction scheme are discussed and included in our design. Our recent efforts are to extend this dialogue control mechanism to handle the consistency among correlated concurrent topics.

References:

[1] H. Aust, M. Oerder, F. Seide, and V. Steinbiss, "A Spoken Language Inquiry System for Automatic Train Timetable Information", *Philips Journal of Research*, Vol. 49, No. 4, 399-418, 1995.

[2] Yang-ju Yang and Lin-shan Lee, "A Syllable-Based Chinese Spoken Dialogue System for Telephone Directory Services Primarily Trained with A Corpus", *ICSLP*98, SL980528.PDF.

[3] Jerry Wright, etc., "Spoken Language Understanding within DIALOGS Using a Graphic Model of Task Structure", *ICSLP98*, SL980385.PDF.

[4] Kuansan Wang, "An Event Driven Model for Dialogue Systems", *ICSLP98*, SL980888.PDF.

[5] MasahiroARAKI, and ShujiDOSHITA, "A Robust Dialogue Model for Spoken Dialogue Processing", *ICSLP98*, SL980729.PDF.

[6] Paul C., etc., "A Schema Based to Dialogue Control", *ICSLP98*, SL980637.PDF.

[7] Didier Pernel, "User's Multiple Goals in Spoken Dialogue", *Eurospeech97*, Vol. 4, 2239-2242.

[8] George F. Luger and William A. Stubblefield, "Rule-based Expert Systems", *in "Artificial Intelligence, structures and strategies for complex problem solving"*, Chap 8, 1993.

[9] Bor-shen Lin, etc., "Hierarchical Tag-Graph Search for Spontaneous Speech Understanding in Spoken Dialogue Systems", *ICSLP98,* SL980449.PDF.