*International Conference on Acoustics, Speech and Signal Processing, Vol. I., May 7-11, Salt Lake City, USA*

# TOPIC FOREST: A PLAN-BASED DIALOG MANAGEMENT STRUCTURE

*Xiaojun Wu, Fang Zheng and Mingxing Xu*

Center of Speech Technology, State Key Laboratory of Intelligent Technology and Systems
Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, China
[*wuxj, fzheng, xumx*]*@sp.cs.tsinghua.edu.cn*, *http://sp.cs.tsinghua.edu.cn*

## ABSTRACT

There are many task-oriented dialog systems, but few of them can cope with the issues such as the multiple-topic issue, the topic changing issue, the information sharing among different topics, and the difference in importance for different information items. To provide efficient solutions, a plan-based dialog management structure named Topic Forest is proposed, which makes the mixed-initiative dialog control easier. The Topic Forest based reasoning engine with a certain strategy for both remembering and forgetting is also described. The reasoning strategy is designed to be domain-independent; therefore it makes the dialog management model easy to be ported to any other different domains.

## 1. INTRODUCTION

The human-machine dialog interface is an integration of speech recognition and language understanding technologies [1,2]. Generally speaking, a human-machine spoken dialog system consists of four modules: spontaneous speech recognition, syntactic and semantic analysis, dialog management, and response generation. Most dialog systems are task-oriented and they try to interactively understand the user's intention according to the dialog context and finally provide the required information services.

Given semantic information and the user's intention by the language-understanding module, the main task of dialog manager is to query or update the system's database and give the user reasonable response according to dialog status. If the elicited information is not sufficient enough, the dialog manager is expected to prompt the user to give more information. And if misunderstanding occurs, the manager is to complete the task all the same by a certain strategy.

Many existing spoken dialog systems adopt the finite-state based or local-managed based approaches.

In the finite-state dialog management model, all dialog states and transitions are predefined as a finite transition network, thus it specifies all the valid dialog paths [3,4], but the system will run into difficulties if the user provides more information than required according to the system's question in a certain dialog state. However, considering all the possible answers will lead to proliferation of dialog states. The model will also get into trouble if there are complicated constraints among parameters of the task for the user has to negotiate with the system.

The local-managed approach does not try to pre-determine the dialog paths and the model contents evolve dynamically. Without the predefinition of dialog states or transitions as in the finite-state model, it is suitable for complex tasks. The dialog control is flexible and it is easy to achieve mixed-initiative. The plan-based modeling is one of such modeling methods; it defrags the specific task to small goals and plans, controls the dialog interaction to accomplish them and completes the whole task finally. This method has great representation ability if there is a good structure and an appropriate management strategy.

In this paper a well-structured plan-based dialog management model is proposed which can handle the multiple-topic issue. And topics can be changed freely and the information items can be adapted discriminatingly to the user's interest.

## 2. TOPIC FOREST

Many existing dialog systems are designed for a single topic. However in daily life, it is more complicated. For example, in a flight reservation dialog system, topics can include flight information, weather conditions, time difference and so on. Furthermore, different topics may share some common information that is often elided when the topics are from one to another. Also many information items in a complex topic are of different importance due to users' different interests. Fro instance in the flight reservation topic, some users care about the plane types and airlines while some do not.

All these problems are not taken into consideration currently. Systems in [5-7] deal with a single topic. Lin's system and Chu-Carroll's operate on necessary information items only [6,7]. Wright's system will take the same action, in case the system knows information item A and discusses with the user about information item B, and in case of vice versa [5]. Lin introduces a dialog model handling multiple topics and topic changing, but sharing information is not considered [8].

The dialog model described here uses a tree-like structure called the Topic Tree (TT) to represent topics of the dialog task. All TTs for topics in the dialog form the Topic Forest (TF). TF, together with its Shared Information Index (SII), can well solve the problems mentioned above. A flight reservation system named *EasyFlight* will be taken as an example to demonstrate the structure and operations.

Figure 1 shows a part of the TF in *EasyFlight*. There are three TTs representing the flight information topic, the airlines information topic and the time difference topic.

### 2.1 Topic Tree Structure

TT is a tree-like structure, representing information items of a single topic, describing relations between items and keeping track of the status of topic information.

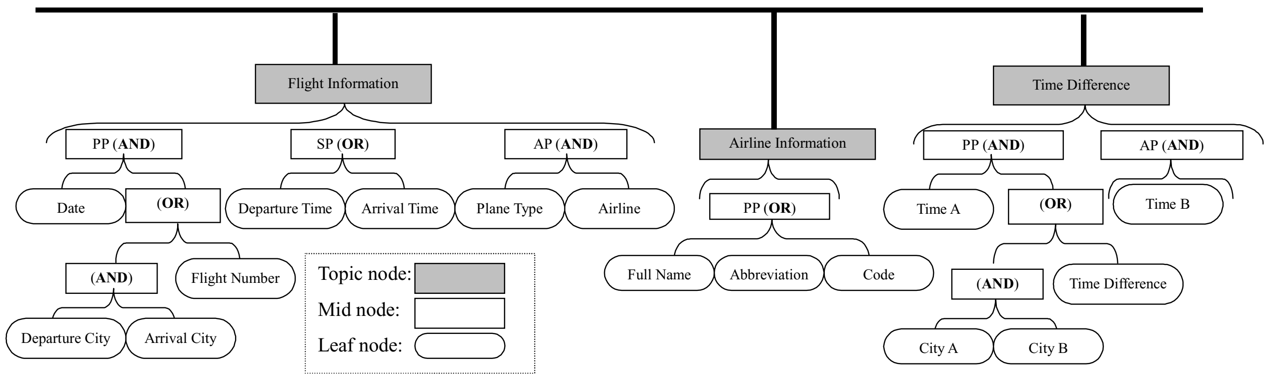

**Figure 1**. Part of a Topic Forest (PP: Primary Property, SP: Secondary Property, AP: Additional Property)

There are three types of nodes in a TT: the topic node, the mid node and the leaf node. (1) Each topic node is the root of a tree, indicating the type of the topic and the corresponding relation in system's database. (2) A mid node labels the logical relation ("AND" or "OR") of all its child nodes. There are three special types of mid nodes, corresponding to the Primary Property (PP), the Secondary Property (SP) and the Additional Property (AP). All leaf nodes under a PP node store dominant information items of the topic, whose values help to decide whether to query the database or not. Nodes under a SP node store some detail information, and the system will take initiative to discuss about them with the user. Other information items are stored in nodes under an AP node. If the user doesn't mention them, neither will the system. The classification for different information items makes it possible for the system to treat items differently in terms of the user's interest. (3) A leaf node is used to store one information item extracted from some semanteme. The content of a leaf node remarks the status of the information item and all leaf nodes of the tree add up to represent the status of the topic.

Each node has a one-bit flag. That of a leaf node indicates whether the user has verified the information item or whether it contains valid value. That of a mid node is the logical operation result of all its child nodes, recursively. A true value for a topic node flag means all information items can be decided and there is no need to ask the user.

Every node is associated with a response generation function (RGF) to generate the text responses to the user, which, according to the current dialog status, may provide knowledge of the information item, or ask the user for desired information, or prompt the user to confirm. Every function is only responsible for the associated node and the designing and the modification of one will not affect any others.

## 2.2 Topic Forest and Shared Info Index

The TF will have been set up when all TTs have been set up. Afterwards we can build the SII so as to deal with shared information items among topics.

The SII can be viewed as a collection of all one-to-many mappings from every special semanteme to all its corresponding leaf nodes. After TF has been set up, the dialog-reasoning engine will traverse all nodes and record those nodes sharing the same semantemes. When an information item is required but it is not valid, the engine will refer to the shared information in other topics with the help of the SII structure. SII connects all the topics in domain as a whole entirety as shown in Figure 2.

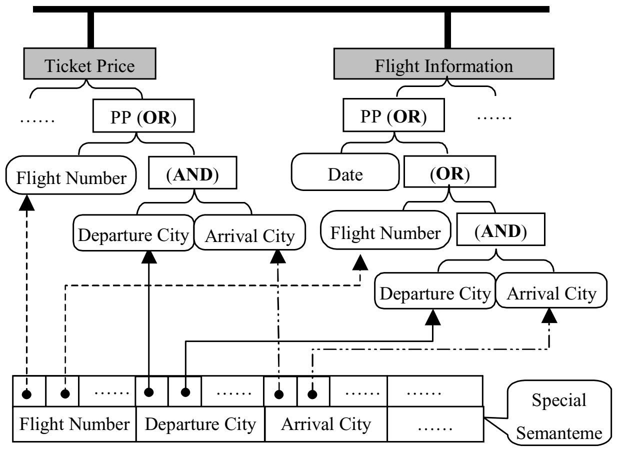

**Figure 2**. TF and SII

## 3. DIALOG MANAGEMENT

Dialog management operations are introduced in this section, including the setup of both TF and SII, and the reasoning engine's strategy.

### 3.1 Preparing TF and SII

When a dialog task is determined, we can analyze and decide all topics in the task transaction for the TF. TTs are then set up according to the properties and relations among information items. The following is an example of setting up the TT for the "*Flight Information*" topic.

First of all, we set up the topic node, which is tagged with the relation in database for query. After that, we consider which information items should be regarded as PP, SP or AP. Thinking of possible flight information queries, all users know the *Departure City* and the *Arrival City* and they also have some idea about the *Travel Date*. These three items are dominant, so they belong to PP. For an experienced user, the *Flight Number* and the *Date* may be provided directly. The two items are enough to determine a flight; hence the *Flight Number* should belong to PP too. In both cases, *Date* is necessary. *Cities* or *Flight Number* is optional. Thus, "*Departure City*" and "*Arrival City*" are connected by an "(**AND**)" node, which is connected with "*Flight Number*" by an "(**OR**)" node. They together with "*Date*" are under the "PP (**AND**)" node. It is not possible that each user knows the exact departure time or the exact arrival time, though both time are important to determine which flight to take and they are usually discussed by the user and the system. So we place them under the "SP (**OR**)" node. Not all users care about the plane types or airlines. These two items belong to the "AP (**AND**)". The following work is to fill the content of each leaf node. We mark each node with the corresponding semanteme, the column name of database relation, value type and default value (e.g. the city name where the system works for "*Departure City*"). At last we connect every node with its RGF. The TT for "*Flight Information*" topic is completed.

When every TT has been set up, we get the TF. Therefore given a task, the setup of the TF is off-line and it can be loaded when the dialog system is being initialized.

After the TF has been loaded, the reasoning engine will traverse all nodes, collect their corresponding semanteme info and set up the SII automatically. It will take little time.

## 3.2 Reasoning Strategy

Here the reasoning means summarizing the status of topic information and deciding which dialog action should be taken. The reasoning engine is closely related to the dialog model's structure and representation ability. The engine proposed in this paper is based on the TF structure.

The engine has two input parameters. One is the semantic frame given by the semantic analysis module according to the automatic speech recognition results while the other is the utterance's topic. The semantic frame consists of semantic slots. Each slot indicates a certain semanteme that will be filled into a proper leaf node. Some non-topic information is in the frame too. The engine keeps a set of status parameters and some auxiliary variables. They are used for representing topic information status and context, which help to do semantic analysis and reasoning.

The engine's output is a dialog action, i.e. the calling to a certain RGF in the dialog system. The function refers to status parameters and auxiliary variables to produce proper text response. The domain knowledge lies in the definition of the semantic frame and semantic slots, the design of TF, the RGFs and the domain's database. But it has nothing to do with the engine's reasoning strategy.

### *3.2.1 Remembering and forgetting*

Information items are stored in leaf nodes. They come from semantic slots extracted from the user's utterance, other leaf nodes indexed by the SII, or the query result from database.

It is called the Topic Tree Filling (TTF) to fill information items into corresponding leaf nodes. The operation can be an appending or a replacing. An appending operation adds new values to leaf nodes so that there are more choices for topic decision. A replacing is to replace old values with new ones so that the manager can forget the obsolete knowledge. The strategy of appending or replacing helps the system to remember or to forget information in the dialog interaction.

In the dialog depicted in Figure 3, the U2 utterance indicates that the user wants to make a decision between "tomorrow" and "the day after tomorrow", so an appending operation is preferred. The U3 utterance verifies date and hence a replacing is proper.

| | |
|---|---|
| U1: Is there any flight to Shanghai tomorrow please? | |
| S1: Yes there is. | |
| U2: How about the day after tomorrow? | (Appending) |
| S2: There is too. | |
| U3: I'd like the day after tomorrow. | (Replacing) |

**Figure 3**. Dialog Example 1

```
 O1: IF PP is invalid
 O2: THEN find an invalid leaf node X under PP, and
 O3:       call X's RGF
 O4: ELSE generate query sentence and perform query
 O5:      IF query fails
 O6:      THEN call a system RGF
 O7:      ELSE perform TTF
 O8:           IF the user asks questions
 O9:           THEN find the answer node Y, and
O10:                 call Y's RGF
O11:           ELIF query result is not unique
O12:                THEN select a node Z under SP, and
O13:                      call Z's RGF
O14:                ELIF all nodes under PP have unique values
O15:                     THEN call topic node's RGF
O16:                     ELSE call a multivalued node's RGF
```

**Figure 4.** Reasoning strategy

### *3.2.2 Reasoning strategy*

After the TTF, the reasoning engine searches the tree structure for an action node whose related RGF will be called later, or to directly call system functions to produce proper response to the user. The reasoning strategy is exhibited in Figure 4 (where **O***x* means Operation *x*).

As shown in Figure 4, O2 searches for an invalid node *X* under the PP, an unknown information item. However, in O12, a node that was once mentioned takes higher priority. Such rule ensures the system to discuss with the user about what he is interested in. And O14 checks the value status of nodes so that the system can help the user to confirm information items even if they have been corrected for several times.

### *3.2.3 Topic changing*

During the interaction, if the system realizes that the user changes the topic, the engine will also change to the new TT and perform operations with the same strategy. If the system decides to change the current topic, status parameters and auxiliary variables are modified in the RGF with topic changing response output to the user. The modification will take effects when processing the user's next utterance.

In brief, the reasoning engine's behavior is independent of domains, because it only works upon the structure of TF and the information status of nodes in spite which topic on earth is under discussion or what the information item means.

## 4. DIALOG EXAMPLE

Figure 5 shows a practical dialog example.

According to the semantic result of U1, the engine fills the "*Arrival City*" node in the "*Flight Information*" TT. The PP is invalid, and it finds the "*Date*" node to take action and the RGF prompts the user a question. When U2 being processed, the PP is valid (the "*Departure City*" node has a default value). The engine traverses the TT to generate a query sentence. The query result isn't unique, so it selects the "*Departure Time*" node to call its RGF, which produces S2. The same thing happens when processing U3, and the RGF produces S3. In U4, the "*Arrival Time*" node is found to answer the user's question. The user changes topic in U5, and the engine fills the "*Time Difference*" TT, getting some information from the "*Flight Information*" TT by SII. Topic is changed back in U6, and this time the query result is unique. The topic node's RGF is called.

During the dialog, the user doesn't take initiative to talk about departure time, plane type or airline. The system takes initiative to ask departure time, but not the plane type or airline, because these information items are of different importance and stored in different parts of TT. It is only if the user initiatively mentions plane type or airline, that system will try to ask about them.

| | |
|---|---|
| U1：I would fly to New York. | (User initiative, telling the arrival city) |
| S1：What day are you going to fly there? | (System initiative, asking for departure time) |
| U2：Wednesday. | (User answering) |
| S2：There are several flights. When will you prefer to depart? | (System initiative, asking for departure time) |
| U3：After 10:00. | (User answering) |
| S3：Which flight do you prefer, 10:10 or 16:30? | (System initiative, asking for departure time) |
| U4：What's the arrival time for the flight at 16:30? | (User initiative, asking for arrival time) |
| S4：19:10. | (System answering) |
| U5：What's the local time in Beijing then? | (User initiative, topic changed, asking for time difference) |
| S5：08:10. | (System answering) |
| U6：Fine, I want the flight departing at 16:30. | (User initiative, topic changed, verifying departure time) |
| S6：You want Flight CA 981 on 14th June, departing from Beijing at 16:30 and arriving in New York at 19:10, is that correct? | (System initiative, asking for user confirmation) |
| U7：Yes, exactly. | (User confirming) |

**Figure 5**. Dialog Example 2

## 5. CONCLUSION

A plan-based dialog management structure named Topic Forest is presented. It is able to deal with multiple topics and make full use of shared information. It is also able to treat information items discriminatingly in terms of users' interests. The reasoning engine based on Topic Forest is domain-independent and it achieves mixed-initiative dialog control. The model has been applied in a Chinese spoken dialog system *EasyFlight*, providing the flight information and reservation services.

The finite-state model has advantage in dialog tasks such as directory assistance and questionnaires, because of its simplicity and reliability. If it can be integrated with the Topic Forest, the model will be more powerful, for the system may control dialog transaction by different models according to current dialog status. How to define status parameters and auxiliary variables needs more work to do so that dialog status can be described more precisely. Also we will try to find a good RGF design method and consider the user's model in order to improve the system performance.

## 6. REFERENCES

[1] Jeremy Peckham, "VODIS - A voice operated data base inquiry system", *Speech Technology*, v3 n3 p56-61, 1986

[2] Victor Zue, Stephanie Seneff, James R. Glass, *et al,* "JUPITER: A telephone-based conversational interface for weather information", *IEEE Transactions on Speech and Audio Processing*, v8 n1 p85-96, 2000

[3] Dinghua Guan, Min Chu, Quan Zhang*, et al*, "The research project of man-computer dialogue system in Chinese", *International Conference on Spoken Language Processing*, p245-248, 1998

[4] Michael F. McTear*, "*Modeling spoken dialogues with state transition diagrams: experiences with the CSLU Toolkit", *International Conference on Spoken Language Processing*, p545-548, 1998

[5] Yi-Chung Lin, Tung-Hui Chiang, Heui-Ming Wang, *et al,* "The design of a multi-domain Mandarin Chinese spoken dialogue system", *International Conference on Spoken Language Processing*, p230-233, 1998

[6] Jennifer Chu-Carroll, "Form-based reasoning for mixed-initiative dialogue management in information-query systems", *European Conference on Speech Communication and Technology*, v4 p1519-1522, 1999

[7] Jerry H. Wright, Allen L. Gorin, Alicia Abella, "Spoken language understanding within DIALOGS using a graphic model of task structure", *International Conference on Spoken Language Processing*, p385-388, 1998

[8] Bor-shen Lin, Hsin-min Wang, Lin-shan Lee, "Consistent dialogue across concurrent topics based on an expert system model", *European Conference on Speech Communication and Technology*, v3 p1427-1430, 1999