A HYBRID DIALOGUE MANAGEMENT APPROACH FOR A FLIGHT SPOKEN DIALOGUE SYSTEM

XIAO-JUN WU, FANG ZHENG, WEN-HU WU

Center of Speech Technology, State Key Laboratory of Intelligent
Technology and Systems, Tsinghua University, Beijing, 100084, China
E-MAIL: {wuxj, fzheng}@sp.cs.tsinghua.edu.cn, wuwh@mail.tsinghua.edu.cn

Abstract:

The finite-state based dialogue management approaches are powerful for simple tasks with strict and reliable dialogue control, while the plan-based dialogue management approaches are feasible for tasks with several topics or one complex topic. However, neither one is good enough for the dialogue control throughout in a dialogue task providing flight information inquiry and ticket reservation service. To solve this problem, a hybrid dialogue management approach is proposed, in which the plan-based Topic Forest structure is combined with some finite-state control, so that a proper management element can function in different dialogue situations.

1 Introduction

A dialogue manager plays so important a role in a task-oriented spoken dialogue system that it has gained more and more attention. Given the parsing result of the recognized utterances from the user, which is possibly with some errors caused by the imperfectness of the speech recognizer, or the natural language understanding (NLU) module, or both of them, the manager tries to keep the dialogue going smoothly and to exchange necessary and useful information with the user in a friendly and efficient way.

The finite-state dialogue management approaches are relatively easy to be adopted in systems and have been demonstrated to be powerful in specific tasks such as the call center services [1], the telephone directory services [2], the appointment schedules [3], and so on. However, it is found difficult to use these approaches in complex tasks. It may cause great dissatisfaction when the user presents more information items than the system expects, which is so-called an over-informative problem [4].

Some alternative approaches belong to the plan-based dialogue management, and they have been introduced in many recent systems. They work well in cases when the finite-state approaches encounter with difficulties as mentioned above.

A Chinese spoken dialogue system named *EasyFlight* is developed to help the user to query desired flight information and to book tickets. In order to achieve a good system performance, a hybrid dialogue management approach is proposed, which takes the advantages of both the finite-state ones and the plan-based ones.

The reason why both approaches are employed will be given in the next section. It will be shown how they are combined into a hybrid one and thereafter how the new dialogue management approach takes effect. There are some discussions about the system performance, the hybrid essence of the dialogue management, and the approach portability. And finally conclusions will be drawn.

2 Task analysis

The typical dialogues between the user and the system *EasyFlight* are of two phases, one for flight information inquiry and the other for ticket reservation.

The flight information topic is complex. Several parameters are needed to determine a single flight. Some of them are necessary for all flights, while others are just optional. Different users care about different parameters in order to get the desired flight information. Also in some cases, the user may ask for some other information, for example information about planes and about airlines, to help make the decision. This leads to topic shifts possibly occurring in any turn, and gives arise to ellipsis and consistency problems across different topics. In this phase, the plan-based dialogue control should be a good choice.

However, dialogues regarding ticket reservation are typically a list of questions on the ticket amount, the confirmation of the ticket amount, the personal ID, and the confirmation of the personal ID. The finite-state dialogue control will be more appropriate, because it is more reliable for such questionnaire-like or form-filling dialogue tasks.

Based on the above analysis, a hybrid dialogue management approach is proposed to take advantages from

0-7803-7508-4/02/$17.00 ©2002 IEEE

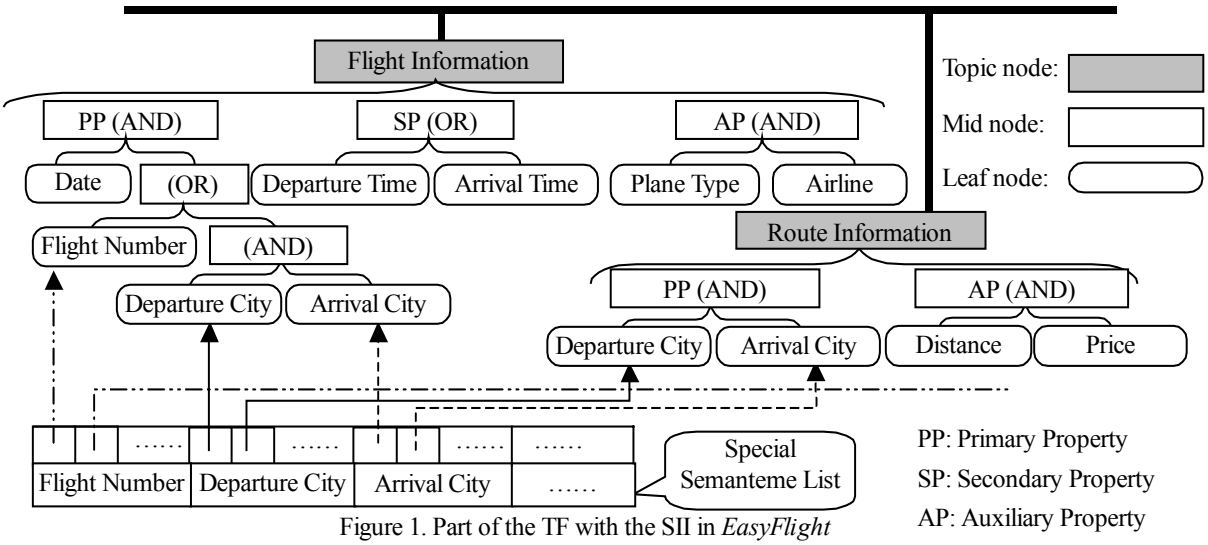

Figure 1. Part of the TF with the SII in *EasyFlight*

both management types, so that a proper management part is selected to act on the dialogue control in either phase.

3 Dialogue modeling

We have proposed a dialogue management model based on the structure called Topic Forest (TF), which is plan-based and well capable of management on dialogues about flight information inquiry [5]. Based on TF, we add the finite-state control, which is responsible for dialogue management in the ticket reservation phase.

3.1 Plan-based Topic Forest

A TF consists of several Topic Trees (TTs) with leaf nodes connected by a Shared Info Index (SII), as depictured in Figure 1.

Every TT (starting with a topic node) represents all information of a certain topic, including all the information items (stored in leaf nodes) with the relationships (represented by mid nodes with the logical relationship "AND" and "OR" and the tree structure) among them. There are three types of branches in a TT, named as Primary Property (PP), Secondary Property (SP) and Auxiliary Property (AP), which are used respectively to store information items with different importance to the topic. Under the PP branch, there are dominant information items of the topic, which can be decided according to the domain database construction. Some detail information items, which are often discussed by the system and the user, are under the SP branch. Other information items are stored under the AP branch.

The SII is a list of semantic items with the indices of related leaf nodes, connecting all the topics in the task as a whole entirety. It helps deal with ellipsis across topics in the dialogue. (More details about TF and SII can be found in [5].)

3.2 Extension of finite-state control

In order to extend the finite-state control to TF without destroying the original structure or disabling the existing operations, we simply embed a special *State* leaf node to the AP branch of the related TT. This node does not store the semantic information that will concern the dialogue content, but the dialogue state, which is the most important in the finite-state control. In other words, this node is a system node that is semantically invisible to the NLU module and only used by the dialogue manager. The real AP branch of the *Flight Information* topic is shown in Figure 2.

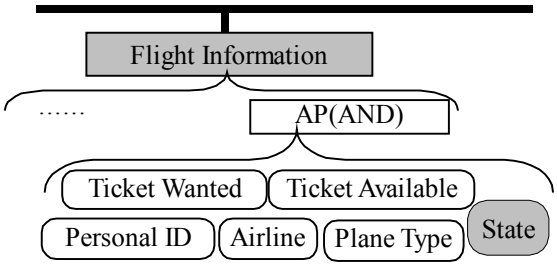

Figure 2. *Flight Information* topic's AP branch

4 Dialogue management

4.1 Input and output

When the user speaks to the dialogue system *EasyFlight*, his/her utterance is recognized as a sequence of key words. A robust parsing module with an extended grammar definition parses the word sequence and then interprets

them to semantemes that forms a semantic frame [6]. The frame is the only input to the dialogue manager. In a word, semantics from the user's utterance, without spoken language phenomena or context recovery, are passed to the dialogue manager.

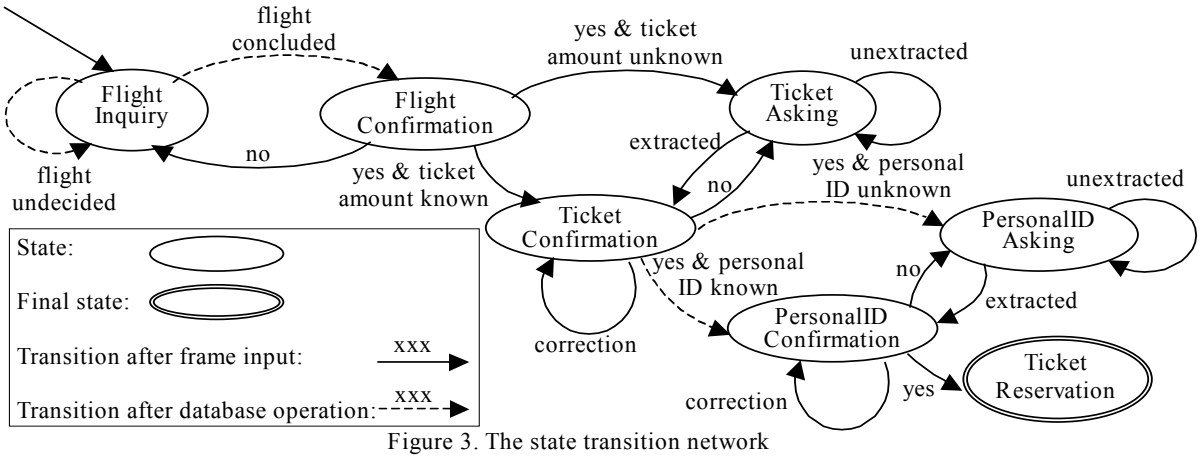

Figure 3. The state transition network

Information in the semantic slots of the input frame will be put into the corresponding leaf nodes respectively. By referring to the structure and content of TTs, the user's requirement and interest in flight information are extracted and collected for the database operation (querying or modifying). According to the operation result and the user's questions if any, the manager can decide what information item is to be fed back to the user, i.e. what is the response focus. Every node in the TF is related to a response generation function. The response focus will be passed to the corresponding function to generate the exact text response.

4.2 Dynamic dialogue management in flight inquiry phase

Since TF is plan-based, the dialogue structure is dynamically decided according to the runtime dialogue situation, which includes the dialogue history, the semantics and the dialogue act of the current user utterance, and the operation result of the task database. In fact, the plan-based dialogue management functions in the first dialogue phase as analyzed.

The dialogue manager controls the dialogue in a mixed-initiative way. The system will try to answer the user's questions, but if information is not sufficient, it will ask the user for it. The user may ignore the system's question, and the system is still cooperative.

When an information item is extracted from the user's utterance, the corresponding leaf node will be assigned TRUE. A mid node will be assigned as the result of the logical operation of all its sub-nodes.

The manager checks whether the PP branch is TRUE in every turn. If not, it will find the unknown information item as the response focus; otherwise, it can query the database to get flight information. If the query result is unique, the system will tell the flight information to the user; otherwise, the response focus will be one information item in the SP branch, and the system will ask the user to give more constraints or to choose from a list. An interesting strategy adopted here is that if the user has mentioned some information item in the SP branch, the system will ask about it first.

The user may have requirements on some items in the AP branch, which will be passed to the database. But the system won't mention them initiatively.

4.3 Dialogue state transitions

The dialogue state of the *Flight Information* topic is stored in the *State* leaf node. The possible dialogue states within the *Flight Information* topic, i.e. the possible values of the *State* node, are *Flight Inquiry*, *Flight Confirmation*, *Tickets Asking*, *Ticket Confirmation*, *PersonalID Asking*, *PersonalID Confirmation* and *Ticket Reservation*. Most state transitions and the transition conditions can be described as in Figure 3. There are still some transitions omitted, for example, those from other states to *Flight Inquiry* with the transition condition being *flight information changed*.

The initial state is *Flight Inquiry*. When a semantic frame is given, some information is put into the TT and the manager tries to determine the dialogue state transition. Most state transitions are determined at this moment, while some transitions such as those represented by dotted-line arrows in Figure 3 are determined after the database operation.

If the dialogue state is *Flight Inquiry*, it will remain the same until at last a certain flight is determined, which is indicated when the database inquiry results in a single entry. This ensures that the plan-based dialogue control functions in the flight inquiry phase as described in section 4.2. Practically, this state cares most dialogue turns.

If the dialogue state is *Flight Confirmation*, the state transition is determined after the semantic frame is given.

All the transition conditions can be determined according to the content of the semantic frame and the *Flight Information* TT. If the flight is confirmed by the user, the dialogue comes to the second phase in which the finite-state dialogue control takes charge.

The transition from state *Ticket Confirmation* is a little different because of the task's specialty. After the ticket reservation amount is confirmed, the manager still has to query the database to see if there are enough tickets available. If no, the dialogue state should be *Ticket Asking* with a message back to the user such as "*Sorry, we don't have enough tickets available now*."

When the personal ID is confirmed, the dialogue state transits to the final state *Ticket Reservation*. At the same time, the ticket reservation records in the database are modified.

4.4 Strategy for remembering and forgetting

When the user gives any information items, the dialogue manager will store them in the corresponding leaf nodes. If there is already some old information, the manager will have to decide to remember or to forget it. The strategy applied here is that if the user's utterance is a declarative sentence, then the old information will be thrown awar (*forgetting*); otherwise, it will be preserved (*remembering*). The new database query will be based on the latest information items. When the dialogue comes to a unique flight and there are several information items kept in the same leaf nodes, the system will initiatively asked the user to decide one.

4.5 Strategy for topic shift

Several topics are involved in the flight task actually and *EasyFlight* enables the user to change topics freely throughout the dialogue. It is assumed that one utterance belongs to only one topic and that topic shifts can be detected only by the contents of utterances.

There are two steps of the topic shift detection. First, the syntactic and semantic parser may detect it, if the utterance is in a special sentence pattern. Second, the manager will check all semantic information items and assure that they belong to one topic. In other cases, the manager assumes the topic keeps the same with the last turn.

When the dialogue topic shifts to another one, the manager just refers and operates on the TT of the new topic. If the new topic shares some information with old ones, the manager will try to get it by consulting to the SII structure, if necessary. The old TT will be kept unchanged when the dialogue manager operates on the new TT.

The state transitions described above are just within the *Flight Information* topic. To be more accurate, the dialogue state discussed above can be referred to as a *topic state*. When the dialogue topic shifts back and forth, the manager still keeps the topic state and it can continue with the determination on the state transition, since the state is seperately stored in the *State* node of the TT.

If there are several topics in the domain, we can design a state transition network for each topic if it is necessary. Because each TT has a *state* node, the manager will keep each topic state respectively. If in some peculiar task, the state change of one topic would influence the state of another one, the state transition determination function should be reinforced.

5 Discussion

We first develop the TF based dialogue manager and it works well for flight information inquiry. When we try to additionally provide ticket reservation service, we just embed an extra node, the *State* node, to the *Flight Information* topic, so that the previous operations of TF can function all the same. The only change after the extension is that a state transition decision function must be called both before and after the database operation.

Figure 4 gives a dialogue example. The extended manager functions all the same for flight information inquiry as before (cf. the example in [5]). On the other hand, because the finite-state control is based on the TF, the user can utter any information items in any dialogue state, i.e. the over-informative problem of finite-state approaches is resolved.

The technologies of speech recognition and natural language understanding are not perfect enough now. Some unrecognized information items will get lost. Some false recognized result will be found nonsense by the syntactic and semantic parser and be elimited. In these cases, the dialogue manager will take the initiative to ask for the necessary items as if the user has not mentioned them yet. In worse cases, the dialogue magager may receive misundertood information as if the user changes his/her mind or has a slip of the tongue. Such situation is just like what happens in human-human interaction when one person mishears another. Our dialogue manager tranverses all the information in the domain TF in every turn, so that it can always assure the consistency of its knowledge. What's more, the system will ask the user for confirmation during the dialouge, which will possiblely correct the errors. In a word, errors may lead to more dialogue turns, but the dialogue still keeps smooth.

S(ystem)1: Welcome to *EasyFlight* the flight information inquiry and ticket reservation system!		(Greeting, in the initial state *Flight Inquiry*)
U(ser)1:	I would fly to New York.	(User initiative, telling the arrival city)
S2:	What day are you going to fly there?	(System initiative, asking for departure time)
U2:	Wednesday.	(User answering)
S3:	There are several flights. When will you prefer to depart?	(System initiative, asking for departure time)
U3:	After 10:00.	(User answering)
S4:	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)
S5:	19:10.	(System answering)
U5:	What's the local time in Beijing then?	(User initiative, topic changed, asking for time difference)
S6:	08:10.	(System answering)
U6:	Fine, I want the flight departing at 16:30.	(User initiative, topic changed, verifying departure time)
S7:	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, entering the state *Flight Confirmation*)
U7:	Yes, exactly. And I want two tickets.	(User confirming, over-information on ticket amount)
S8:	You want two tickets. Is that correct?	(System initiative, entering the state *Ticket Confirmation*)
U8:	Yes.	(User confirming)
S9:	What's your personal ID?	(System initiative, entering the state *PersonalID Asking*)
U9:	123456780912345.	(User answering)
S10:	Your personal ID is 123456780912345. Is that correct?	(System initiative, entering the state *PersonalID Confirmation*)
U10:	Yes.	(User confirming)
S11:	Your reservation is added to the database. Thank you for using *EasyFlight* system!	(System initiative, entering the state *Ticket Reservation*, farewell)

Figure 4. A dialogue example

In *EasyFlight*, every topic is in fact finite-state controlled, while dialogues in the *Flight Inquiry* state in *Flight Information* topic is plan-based controlled and the topic shift resolution is plan-based too. Different from some other finite-state approaches in which a state may be a sub-network of some state transitions, our approach proposed here is really a hybrid one, because the dialogue state transition network cannot be expanded to a complete larger one.

The manager is designed to be task independent. The task specific knowledge is represented by the structure of TF, the database, and the response generation functions. The state decision function for each topic is task-specific too. The dialogue mangement strategies, for both finite-state and plan-based dialogue control, operate on the structure and contents of the TF without realizing the domain semantics. The domain-specific functions are just called in the predefined routine. Therefore the approach proposed here is portable in large measure.

6 Conclusion

The finite-state and the plan-based dialogue management approaches come from different dialogue model theories. They are different in dialogue control and competent for different tasks. However, we manage to combine them into a hybrid one, the TF based dialogue management approach, which is brand-new.

A TF can be viewed as a set of TTs with leaf nodes connected by an SII, and each TT has a state transition network used for state control of the very topic. Topic shifts just lead to the dialogue manager to change the TT in operation and thus to change to the other topic's state, but the former topic condition is kept and ready for the topic to shift back. Dialogue management within and across topics are plan-based.

In certain dialogue situations, an appropriate element of the hybrid dialogue management approach will take over the dialogue control. It helps achieve an excellent system performance throughout in *EasyFlight,* which is impossible or not so good for just using a single type of

approach. With control strategies domain-independent in great deal, this approach is highly portable.

References

[1] K. Georgila, A. Tsopanoglou, N. Fakotakis, et al, "An Integrated Dialogue System for the Automation of Call Center Services," International Conference on Spoken Language Procession, v 2, page 45~48, 1998.

[2] Yen-Ju Yang, Lin-Shan Lee, "A Syllable-based Chinese Spoken Dialogue System for Telephone Directory Services Primarily Trained with a Corpus," International Conference on Spoken Language Procession, v 3, page 619~622, 1998.

[3] H. Brandt–Pook, G.A. Fink, B. Hildebrandt, et al, "A Robust Dialogue System for Making an Appointment," International Conference on Spoken Language Procession, v 2, page 693~696, 1996.

[4] Michael F McTear, "Modeling Spoken Dialogues with State Transition Diagrams: Experiences with the CSLU Toolkit," International Conference on Spoken Language Procession, v 4, page 1223~1226, 1998.

[5] Xiaojun Wu, Fang Zheng, Mingxing Xu, "Topic Forest: A Plan-based Dialog Management Structure," International Conference on Acoustics Speech and Signal Procession, v 1, page 617~620, 2001.

[6] Pengju Yan, Fang Zheng, Mingxing Xu, "Robust Parsing in Spoken Dialog Systems," European Conference on Speech Communication and Technology, v 3, page 2149-2152, 2001.