Modelling spoken dialogues with state transition diagrams: experiences with the CSLU toolkit

Michael F McTear

University of Ulster

ABSTRACT

The development of a spoken dialogue system is a complex process involving the integration of several component technologies. Various toolkits and authoring environments have been produced that provide assistance with this process. This paper reports on several projects involving CSLU's RAD (Rapid Application Developer) and critically evaluates the applicability of state transition diagrams for modelling different types of spoken dialogue. State transition methods have been recommended for dialogues that involve well-structured tasks that can be mapped directly on to a dialogue structure. However, other significant factors to be considered include the structure of the information to be transacted and the need for verification of the user's input as determined by the system's level of recognition accuracy. Examples of different types of dialogue are presented together with recommendations concerning the advantages and disadvantages of state transition based dialogue control.

1. INTRODUCTION

The success of a spoken dialogue system depends crucially on a carefully designed interface that can overcome the limitations of current spoken language technology [1,2]. Given that speech recognition is a probabilistic process that cannot be guaranteed to be completely accurate, methods for preventing, detecting and recovering from errors are required. At the same time, in order to maximise user satisfaction, the dialogue flow should be natural and efficient [3]. A number of toolkits are available that assist developers with the complex issues involved in constructing a spoken dialogue system. This paper reports on experiences with the CSLU toolkit, which includes a graphically-based authoring environment (RAD, Rapid Application Developer) for constructing simple spoken dialogue systems [4]. Dialogues are created in RAD by linking together a series of dialogue objects to create a finite-state dialogue model. Finite-state models (also known as graph-based models or state-transition diagrams) have been criticised because of their inflexibility as well as their inability to model complex dialogues [1,5]. This paper focuses on the strengths of finite-state dialogue models by examining the types of task for which they are appropriate as well as the technological implications of their use, with the aim of determining the class of dialogues that can be modelled using finite-state methods.

2. APPROACHES TO DIALOGUE MANAGEMENT

A spoken dialogue system can be judged to be successful and usable in an applied context if it provides the correct information within an acceptable transaction time by controlling the flow of the dialogue between the system and the user in as natural a way as possible. Dialogue flow is controlled by the dialogue management module. This module has to determine whether sufficient information has been elicited from the user in order to enable communication with an external application, to engage in communication with the external application to retrieve the required information, and to communicate that information back to the user. The dialogue management module is also responsible for detecting and repairing breakdowns in the dialogue through verifications, confirmations and corrections.

Dialogue management systems can be categorised in terms of the type of control offered and how the control is managed. Dialogue control may be system-led, user-led, or mixed initiative. In a system-led dialogue the system asks a sequence of questions to elicit the required parameters of the task from the user. A user-led dialogue is controlled by the user, who asks the system questions in order to obtain information. In a mixed initiative dialogue control is shared. The user can ask questions at any time, but the system can also take control to elicit required information or to clarify unclear information. The management of dialogue control is not an issue for user-led dialogue as the user decides on which questions to ask. In system-led and mixed-initiative dialogue the control has to be managed in order to determine what questions the system should ask, in what order, and when.

Approaches to dialogue management can be broadly classified into finite-state methods, on the one hand, and self-organising or locally-managed approaches on the other [6]. In a finite-state model the dialogue structure is represented in the form of a state transition network in which the nodes represent the system's questions and the transitions between the nodes determine all the possible paths through the network, thus specifying all legal dialogues. In RAD, as in most other finite-state systems, subdialogues can be used. These can range in complexity from a single state to a nested collection of subdialogues, supporting a more modular approach to dialogue modelling and providing libraries of commonly recurring transactions. There are several variants of self-organising dialogue managers, including frame-based and object-oriented approaches, theorem-proving, plan-based management, and event-driven methods. Self-organising methods provide for greater flexibility. The paths through the dialogue are not pre-determined and the structure evolves dynamically based on some computation of the next dialogue act. Most commercially available dialogue systems use some form of finite-state dialogue modelling, although a more flexible event-driven dialogue control has been successfully deployed in commercial contexts using the Philips SpeechMania™ toolkit, while the DialogBuilder component of the Nuance Developer's Toolkit supports an object-oriented approach to dialogue development [5, 7].

2.1. Criticisms of finite-state dialogue models

Finite-state models have been criticized because of their inflexibility as well as their inability to cope with the requirements of more complex dialogues. Because the dialogue paths are specified in advance, there is no way of managing deviations from these paths. Problems arise, for example, if the user's answer is over-informative, i.e. it provides more information than required by the question. Taking the example of a simple travel inquiry system, a natural order for the system's questions might be: *destination* > *origin* > *date* > *time* . However, when answering the system's question concerning destination the user might reply with a destination as well as the departure time (or indeed other combinations of the four required parameters). A finite-state based system would simply progress through its set of predetermined questions, ignoring or failing to process the additional information and then asking an irrelevant question concerning the departure time [5]. There are two solutions to this problem. The first is to attempt to constrain the user's input to the responses required by the system [3]. The second approach, which aims to permit more flexible user input, is to provide additional questions and transitions to cope with the range of possible user responses. However, as soon as the questions multiply, the number of transitions grows to unmanageable proportions. This problem is further augmented if adequate repair mechanisms are to be included at each node for confirmation or clarification of the user's input. Thus it was estimated that in the Philips system there were about 1,000 system questions. Allowing for flexible adaptation to the user's input - for example, in the case where a user says more than the system expected or provides an unanticipated response - given that almost any question could follow almost any other - the network would require tens of thousands of transitions [5].

A further problem concerns the complexity of the task to be accomplished in the dialogue. Dialogues involving various types of simple inquiry and information transfer, such as travel or financial information, can be modelled as form-filling tasks, in which the system finds values for slots in a query pattern. Other types of dialogue that involve some form of negotiation cannot be modelled in this way. For example, planning a journey may require the discussion of constraints that are unknown by either the system or the user at the outset. In these interactions some form of negotiation and discussion of constraints is required. For example, in the TRAINS project [8], the user and the system collaborate to construct an agreed executable plan that has to be developed incrementally in order to incorporate new constraints that arise during the course of the dialogue.

2.2. Empirical studies of finite-based dialogue systems

The strengths and weaknesses of system-led dialogue control and finite-state models have been investigated in several empirical studies. Hone and Baber [9] examined the relationship between dialogue constraints and transaction times, finding that more constrained dialogues that employed a menu-like interaction style with yes/no confirmation of all user input tended to result in dialogues with longer transaction times, as would be expected. However, this effect depended on the system's level of recognition accuracy, which was manipulated in the experiments. It was found that there was a greater likelihood of errors in the less constrained system as it permitted a larger active recognition vocabulary. In another study two versions of a simple call assistance application were built [10]. The system-led version used isolated word recognition and word spotting, while the mixed initiative version used continuous speech recognition and more complex natural language processing. In the system-led version the user was prompted for the required service in two steps, while in the mixed initiative version the user could request the service in a single utterance. The minimum number of turns per transaction was lower for the mixed initiative system, although more additional turns were required for the mixed initiative system on account of the greater number of recognition errors. Thus the system-led interface was not slower than its mixed-initiative counterpart. Moreover, a subjective analysis of user satisfaction indicated that users were satisfied with both versions. Similar results were found in a study involving train timetable information [11] in which it was found that for simple services a system-driven dialogue using isolated word recognition achieved good user acceptance. This finding was supported in a study of dialogue strategies comparing explicit and implicit recovery from communication breakdowns [12]. The version incorporating explicit confirmation and repair, which made greater use of isolated word recognition and spelling, was found to be robust and safe, even though it increased the number of turns required to complete the transaction.

The conclusion from these studies is that system-led dialogue using state transitions would appear to be suitable for simple tasks with a flat menu structure and a small list of options, bringing also the advantage of less complex spoken language and dialogue modelling technology. The lack of flexibility and naturalness may be justified as a trade-off against these technological demands. However, the definition of what constitutes a simple task is unclear. The examples described in the next section attempt to tease out some of these factors.

3. USING STATE TRANSITION DIAGRAMS FOR DIALOGUE MANAGEMENT

It is generally assumed that a task-oriented dialogue reflects closely the structure of the domain task [4]. The examples to be considered in this section - directory assistance, questionnaires, travel inquiries - could all be considered to be well-structured domain tasks.

3.1. Directory assistance

A number of directory inquiry systems have been developed using spoken dialogue technology [13,14]. We will consider only the part of a directory inquiry dialogue in which the system attempts to elicit the name of the person to be called. To complete this task and identify a unique individual, a first and last name are required. This task can be completed in a single step - *Request First and Last Name* - as in [13], or in a series of

steps - *Request Surname* > *Request Spelling of Surname* > *Request First Name* > *Confirm First and Last Name* - as in [14]. In a system implemented using RAD at the University of Ulster the dialogue was designed to maximise transaction success at the possible expense of transaction duration i.e. the system persisted with attempts to elicit the name until a certain threshold was reached (set arbitrarily at 2 attempts at elicitation or confirmation) before giving up [15]. The dialogue model was structured as follows:

Elicit first and last name > *If successful, confirm* > *Else elicit surname* > *If successful, confirm* >*Else request spelling of surname* >*If successful, confirm and Elicit first name* > *If successful, confirm* > *Else elicit spelling of first name* > *If successful, confirm and continue* > *Else fail.*

This task can be accomplished in a minimum of four system/user turns (if confirmation turns are included), and in a maximum of 30 system/user turns in the worst case scenario before the system gives up. (Additional turns were required for situations involving multiple individuals with the same name, variations on first names, and names that are homophones and that require spelling to disambiguate).

The main characteristic of this task and its dialogue model is that there is a minimal amount of information to be elicited. The elicitation sequence is relatively fixed and the user can be constrained to respond minimally to suitably designed prompts. This task lends itself easily to implementation using a finite-state dialogue model incorporating sub-dialogues for sub-tasks such as Request Surname and repair sub-tasks. A finite-state model could also be used for similarly structured tasks such as obtaining weather forecasts, football scores, ordering items from a catalogue, or making simple bank transactions. In each case the task is well structured (possibly hierarchically), there is a small number of parameters to be negotiated, and a relatively fixed sequence for their elicitation.

3.2. Questionnaires

Dialogues for questionnaires are also highly structured even though a large number of questions may be required to elicit the required information. For questionnaires the user can be constrained through carefully designed prompts to produce an acceptable range of responses [3]. A large scale project involving the US Census was implemented at CSLU using RAD [16]. The information can be elicited in a fixed order, such as *Name* > *Gender* > *Birth date* > *Marital Status, etc.* , with sub-dialogues for the more complex items. Finite-state models can be used for similar tasks such as eliciting a person's personal details for financial transactions or obtaining information for insurance quotes. The key characteristic of this class of dialogues is that they are well-structured. Even though there may be several items of information to be elicited, these can be broken down into well-structured sub-tasks that are independent of one another.

3.3. Travel inquiries

For travel inquiries there is also in the simplest case a fixed set of parameters to be acquired and a natural sequence for their elicitation, e.g. *destination* > *origin* > *date* > *time* . Indeed, the Philips system could have been implemented using a finite-state model, given this task structure, if there had not been additional requirements for flexible and natural interaction, which allowed users to produce over-informative answers. An example of a finite-state dialogue for travel inquiries is described in [4].

4. DISCUSSION

The extent to which well-structured tasks can be modelled with finite-state techniques depends, however, also on additional factors, particularly the structure of the information to be negotiated as well as the type of verification that is required [2]. Where there are dependencies between the items of information to be elicited, finite-state dialogue models soon run into difficulties. A good example is the Flight Reservation System of the Danish Dialogue Project [17]. Although the reservation task was found in a field study to be well-structured, consisting of a series of ordered sub-tasks, there are complex dependencies in this system between various parameters, for example, between discounted fares and flight availability. As a result a client could opt for a discounted fare and go on to confirm several parameters only to have to backtrack to a different dialogue path because the desired departure time was not available at the discounted price. The keyword "change" can be spoken by the user to correct the latest piece of information given to the system, but to correct earlier information "change" has to be used repeatedly to cause the system to backtrack sequentially until the item to be changed is reached. Thus when there are dependencies between the items of information the use of a finite-state dialogue model becomes unwieldy, leading to the combinatorial explosion of states and transition described in [5].

Regarding verification, the simplest way to implement verification in a finite-state model is by using explicit verification as each item of information is elicited. Some form of implicit verification is possible within such a model, as shown in the Danish system, but this requires careful design to enable the user's corrections to be detected and interpreted as new values. The results tend to be unnatural and cumbersome, compared with the more natural implicit verifications possible with the Philips SpeechMania™ system. One approach is to dispense with verifications for all but the most important items of information. This method was used in the CSLU Census system as a result of user studies that indicated that users disliked verification of each response [16]. In this system if the confidence score for the user's response was low, the prompt was repeated. The repeated response was recorded and recognised, and the system proceeded to the next question. The question was then flagged for review if the repeated response received a low confidence score. In this system operators could identify and correct errors at a later date, whereas in other applications using a similar dialogue model the confidence ratings could be used to determine whether a prompt should be repeated (low confidence), whether the system should verify the information (medium confidence), or accept the recognised value and proceed to the next question (high confidence). A similar solution was adopted in the TRAINS system [8] in which, when faced with ambiguity, the system chose a specific interpretation, running the risk of making a mistake, as opposed

to generating a clarification sub-dialogue. Given good recognition this strategy results in shorter transaction times. Similarly, as argued in [2], if the cost of errors is minimal, it may be possible to forgo confirmations and allow the system to proceed with its next question. This strategy requires an analysis of the potential cost of errors. Given good recognition, the problem or errors will occur only infrequently and transaction times can be improved. However the system has to be able to recognise and interpret subsequent corrections if they do arise. Except for highly constrained corrections this would not be possible for finite-state systems that used isolated word recognition or word-spotting rather than more sophisticated natural language processing.

4. CONCLUSIONS

This preliminary analysis of classes of dialogue has indicated that the choice of a finite-state dialogue model for a particular task is determined not only by the nature of the task but also by other factors such as dependencies between the information items in the dialogue, the methods used for verification, and the performance of the speech recognition and understanding modules. Future work will investigate the types of dialogue model that are best suited for other classes of dialogue with the aim of providing principled guidelines to support dialogue engineers when faced with complex decisions involving technological constraints and optimal solutions.

5. REFERENCES

1. van de Burgt, S.P., Kloosterman, H., Andernach, T., Bos, R., and Nijholt, A. "Building dialogue systems that sell", *Proceedings Natural Language Processing and Industrial Applications*, New Brunswick, 41-46, 1996.

2. Kamm, C., "User Interfaces for Voice Applications", In D.B.Roe and J.G.Wilpon (eds.) *Voice communication between humans and machines*. Washington, D.C.: National Academy Press, 34-75, 1995.

3. Hansen, B., Novick, D.G., and Sutton, S. "Systematic Design of Spoken Prompts," *CHI'96*, Vancouver, B157-164, 1996.

4. Novick, D.G., and Sutton, S., "Building on experience: managing spoken interaction through library subdialogues", *Proceedings of TWLT 11: Dialogue Management in Natural Language Systems,* Twente, 51-60, 1989.

5. Aust, H., and Oerder, M. "Dialogue control in automatic inquiry systems", *ESCA Workshop on Spoken Dialogue Systems*, Vigso, Denmark, 121-124, 1995.

6. Fraser, N.M., and Dalsgaard, P. "Spoken Dialogue Systems: A European Perspective", *Proceedings of International Symposium on Spoken Dialogue*, Philadephia, 25-36, 1996.

7. Nuance Developer's Toolkit. WWW document: http://www.nuance.com/products/toolkit.htm.

8. Allen, J.F., Miller, B.W., Ringger, E.K., and Sikorski, T. "A robust system for natural spoken dialogue", *Proceedings of the 34th Annual Meeting of the ACL*, 62-70, 1996.

9. Hone, K.S., and Baber, C. "Using a simulation method to predict the transaction time effects of applying alternative levels of constraint to utterances within speech interactive dialogues", *ESCA Workshop on Spoken Dialogue Systems*, Vigso, Denmark, 209-212, 1995.

10. Potjer, J., Russel, A., Boves, L., and den Os, E. "Subjective and objective evaluation of two types of dialogues in a call assistance service", *IVTTA*, Basking Ridge, New Jersey, 121-124, 1996.

11. Billi, R., Castagneri, G., and Danieli, M. "Field trial evaluation of two different information inquiry systems", *IVTTA*, Basking Ridge, New Jersey, 129-132, 1996.

12. Danieli, M., and Gerbino, E. "Metrics for evaluating dialog strategies in a spoken language system", *AAAI-95 Spring Symposium on Empirical Methods in Discourse Interpretation and Generation*, Stanford, CA., 34-39, 1995.

13. Fraser, N.M., Salmon, B., and Thomas, T. "Call routing by name recognition: field trial results for the Operetta™ system", *IVTTA*, Basking Ridge, New Jersey, 101-104, 1996.

14. Attwater, D.J., and Whittaker, S.J. "Issues in large-vocabulary interactive speech systems", *BT Technology Journal*, 14:1, 177-186, 1996.

15. McKendry, M. "The development of directory inquiry spoken dialogue systems". Project Report, University of Ulster, 1998.

16. Cole, R.A., Novick, D.G., Vermeulen, P.J.E., Sutton, S., Fanty, M., Wessels, L.F.A., de Villiers, J., Schalkwyk, J., Hansen, B. and Burnett, D. "Experiments with a spoken dialogue system for taking the U.S. census", *Speech Communication*, 23, 3, 1997.

17. Dybkjær, L., Bernsen, N.O., and Dybkjær, H. "A methodology for diagnostic evaluation of spoken human-machine interaction", *Int. J. Human-Computer Studies* 48: 605-625, 1998.