Probabilistic Dialog Management

Dirk Schnelle-Walka, Stefan Radomski, Stephan Radeck-Arneth
Telecooperation Group
Darmstadt University of Technology
Hochschulstraße 10
D-64283 Darmstadt, Germany
[dirk|radomski|stephan.radeck-arneth]@tk.informatik.tu-darmstadt.de
phone: +49 (6151) 16-64231

October 11, 2013

Abstract

Modeling user interfaces as dialogs provides a conceptual framework to address global coherence and efficiency of interactions. While non-probabilistic approaches provide convincing results and transparent dialog behavior, probabilistic techniques can help to account for inherent uncertainties in user input. In this paper, we present three patterns for probabilistic dialog management or support thereof.

1 Introduction

Describing graphical user interfaces is still, predominantly, achieved by applying the Model-View-Controller pattern [5] or one of its variations. Wherein the graphical widgets of an application provide the means to input data as the view, processed by a controller component to adapt the model of an application, which in turn, updates the view. This pattern works very well for graphical user interfaces in the absence of recognition errors or with inexpensive error correction.

There are two problems with the MVC pattern with regard to generic user interfaces. First, it does not ensure a coherent global dialog behavior between the user and the system. Second, it assumes unambiguous user input, recognized without any errors, which is not the case for interfaces employing spoken or gesture input. Therefore, we identified MVC as an anti-pattern [6] as far as dialog management is concerned.

This paper is aimed at developers of multimodal interfaces and extends our pattern language introduced in the aforementioned earlier paper. We describe patterns to support global dialog coherence by probabilistic approaches, accounting for inherent recognition uncertainty with some modalities.

2 Patterns

The patterns described in this paper integrate into the language of patterns for dialog management that we introduced in [6]. An overview of the pattern language and their relations is shown in figure 2. The earlier patterns

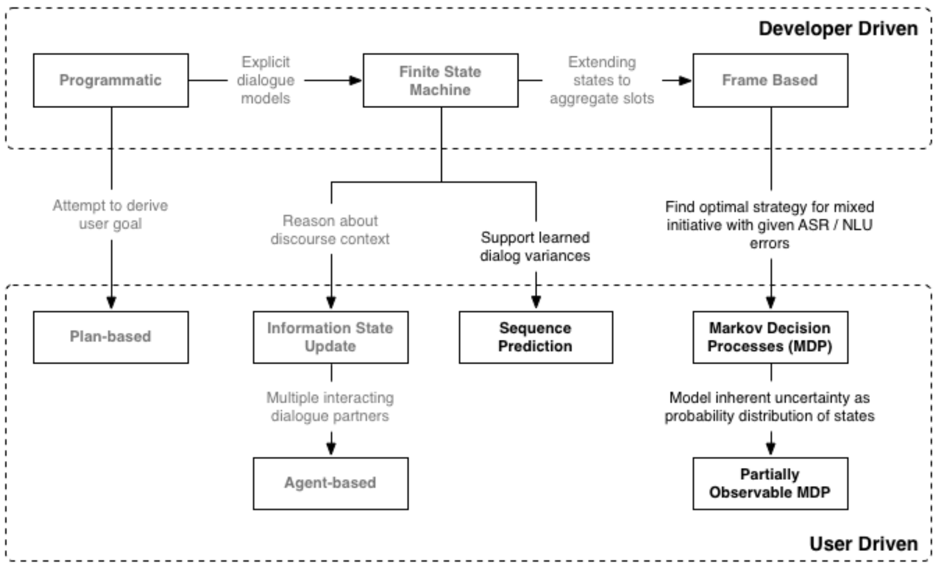

Developer Driven
Programmatic
Explicit dialogue models
Finite State Machine
Extending states to aggregate slots
Frame Based
Attempt to derive user goal
Reason about discourse context
Support learned dialog variances
Find optimal strategy for mixed initiative with given ASR / NLU errors
Plan-based
Information State Update
Sequence Prediction
Markov Decision Processes (MDP)
Multiple interacting dialogue partners
Model inherent uncertainty as probability distribution of states
Agent-based
Partially Observable MDP
User Driven

Figure 1: Overview of the pattern language

of our language are shown as gray. Table 2 summarizes these patterns as (External) Pattern Thumbnails [3].

In this paper we extend our language by three more patterns: Markov Decision Processes (MDP), Partially Observable MDP and Sequence Prediction. While the first two have been established in research around dialogue management in the past decade, the latter is more of an explorative nature. This means that we use the pattern format to explore the domain. We aim for extending existing research in sequence prediction algorithms mainly discussed in [2] to discuss their applicability to dialog management. Consequently, there are no known uses.

The following terms are defined in more detail in our original paper [6] and only included here for completeness.

Dialog (-strategy): A recursive sequence of inputs and outputs necessary to achieve a goal [4].

Dialog Turn: A single input or output within a dialog.

(Information) Slot: Storage to hold a single atomic piece of information.

Name	**Intent**
PROGRAMMATIC DIALOG MANAGEMENT	Implement an interactive application with unimodal interaction and no need for explicit dialog management.
FINITE STATE DIALOG MANAGEMENT	Provide an interactive application with an easy way to adapt the dialog structure later on.
FRAME BASED DIALOG MANAGEMENT	Allow for adaptations of dialog structure without altering application logic, but try to ease the verbosity of finite state dialog models.
INFORMATION STATE UPDATE	Allow for more flexible dialogs with a certain amount of *intelligence* in the dialog structure.
PLAN BASED DIALOG MANAGEMENT	Uncover the user's underlying goal to guide the actual dialog management.
AGENT BASED DIALOG MANAGEMENT	Model interaction with distinct subsystems as agents with their own beliefs, desires, intentions (and obligations).

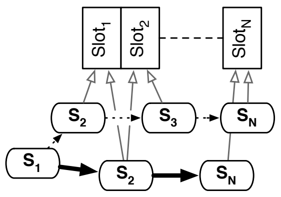
Table 1: Thumbnails of patterns described in [6]

Mixed Initiative: The possibility for a user to provide compound information during her turn, as opposed to a single information; Fills multiple information slots at once.

Patterns are in a custom format which is based on the Coplien format [1] which we find useful to talk about human computer interaction.

Markov Decision Processes (MDP)

Intent



Find an *optimal* dialog strategy to fill a set of information slots with a sequence of mixed-initiative dialog turns, where a single user action can potentially fill multiple slots.

Context

A mixed-initiative dialog needs to sufficiently instantiate a template as a set of information slots. Different (compound) user actions yield different recognition accuracies depending on their complexity and the performance of a recognizer and an eventual natural language understanding unit. Frame Based Dialog Management can be applied to allow for an arbitrary order to fill the slots but does not take the different recognition rates with compound user actions into account.

Problem

There are multiple, possible sequences of state transitions, each state prompting the user to perform an action that will potentially fill multiple information slots. The recognition accuracies differ with regard to the amount and kind of information the user action contains for a single prompt. How to account for those different recognition accuracies to find an optimal and consistent dialog strategy to sufficiently instantiate a template?

Forces

- A sufficiently large corpus of example dialogs exist or users can be simulated.
- Dialog is limited in scope to prevent state explosion.
- The goal of the user can be conceived as the instantiation of a template.

Solution

The solution tries to find the optimal dialog strategy to instantiate templates as a set of information slots as defined by an objective cost / reward function [3]. By modeling the dialog as a Markov Decision Process (MDP), approaches from reinforced learning can be applied to minimize these cost / reward function with regard to the dialog turns. Paek and Chickering analyzed in [4] different reward models and their suitability to constrain the state space when its structure is unknown.

A MDP is formally defined as the quadruple:

$$\begin{aligned} MDP :=& \quad S, A, T, R \\ S_{tates} :=& \quad \{s \in all\ dialog\ states\} \end{aligned}$$

$$\begin{aligned} A_{ctions} :=& \quad \{a \in \textit{all output actions}\} \\ T_{ransitions} :=& \quad P(s_{t+1}|s_t, a) \\ R_{eward} :=& \quad r(s, a) \end{aligned}$$

With S_{tates} being the cartesian product of all the information slots with their possible values. A_{ctions} as a set of dialog acts the machine can perform to prompt the user to provide input, $T_{ransitions}$ as the probabilities to transition from state s_t to s_{t+1} should the action a be selected and R_{eward} as a cost / reward function, associating a cost for performing an action in a given state.

The dialog starts in the state where all information slots are unfilled. The transition with the highest probability is taken and the associated action is performed (e.g. open prompt greeting). The user's input is processed by a semantic interpretation unit and the new state determined. The process continues until one template is sufficiently instantiated for the system to satisfy the users goal. To apply a Markov Decision Process for dialog modeling, consider the following:

1. Establish the state-set S_{tates} as the cartesian product of all possible input field values.
2. Identify all possible actions A_{ctions} that are relevant to perform the dialog.
3. Provide a cost function to account for e.g. overall dialog length, cost of database queries, number of unfilled information slots, cost of rendering a new prompt.
4. Use reinforced learning to train the $T_{ransition}$ probabilities from the dialog corpus or user simulation.

The resulting MDP provides the basis for a dialog strategy, satisfying the optimality criterion implicit in the cost function.

Consequences

☺ Resulting dialog strategy is learned from real data.

☺ All dialog strategies are (nearly) optimal with regard to cost function.

☺ Accuracy of recognition and language understanding is taken into account.

☹ The cartesian product of all possible input field values tends to be huge and an optimal solution often intractable.

☹ Actual dialog behavior is opaque as it is encoded in the MDP.

☹ Adapting the dialog requires retraining.

☹ Recovery strategies, to realign the users interaction intent with the systems interpretation [5], are difficult to implement, as all the different approaches would need to be formalized in the cost function.

Known Uses

Lemon showed in [2] that dialog management and natural language generation are closely related and that a joint and automated training result in a significantly better reward.

Boyer et al. introduce in [1] a tutorial dialog system based on this pattern.

Related Patterns

Markov Decision Processes (MDP) extends Frame Based Dialog Management to find an *optimal* dialog strategy to fill a set of information slots with a sequence of mixed-initiative dialog taking into account the respective recognition rates for such compound input.

Partially Observable MDP helps to model this uncertainty as a probability distribution of states.

References

[1] Kristy Elizabeth Boyer, Eun Young Ha, Robert Phillips, Michael D Wallis, Mladen A Vouk, and James C Lester. Inferring tutorial dialogue structure with hidden markov modeling. In *Proceedings of the Fourth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 19–26. Association for Computational Linguistics, 2009.

[2] Oliver Lemon. Learning what to say and how to say it: Joint optimisation of spoken dialogue management and natural language generation. *Computer Speech & Language*, 25(2):210–221, 2011.

[3] E. Levin, R. Pieraccini, and W. Eckert. Using markov decision process for learning dialogue strategies. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 1, pages 201 –204 vol.1, may 1998.

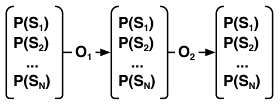
[4] Tim Paek and David Maxwell Chickering. The markov assumption in spoken dialogue management. In *6th SIGDIAL Workshop on Discourse and Dialogue*, 2005.

[5] Dirk Schnelle-Walka. A pattern language for error management in voice user interfaces. In *Proceedings of the 15th European Conference on Pattern Languages of Programs*, EuroPLoP '10, pages 8:1–8:23, New York, NY, USA, 2010. ACM.

Partially Observable MDP

Intent

Provide a dialog management system that implicitly copes with the uncertainty related to the recognition of user input.



Context

Markov Decision Processes (MDP) has been applied to find an *optimal* dialog strategy to fill a set of information slots in a mixed initiative dialog. Still, a huge class of problems, especially with multimodal applications, stems from the fact that user input cannot be recognized with absolute certainty.

Problem

How to model dialogs with inherent uncertainty in the user input?

Forces

- User input intend cannot be derived with certainty from employed modalities.
- A sufficiently large corpus of example dialogs exists or user input can be simulated.
- Dialog is limited in scope to prevent state explosion.

Solution

Implicitly modeling uncertainty as a partially observable Markov decision process with a probability distribution among all the dialog states can help to arrive at concise and effective dialogs in the absence of robust recognition. In order to implement this strategy consider the following:

Model the dialog as a partially observable Markov Decision Process - an extension of the MDP quadruple defined above as:

$$\begin{aligned}
POMDP &:= S, A, T, R, O, Z, \lambda, b_0 \\
S_{tates} &:= \{s \in all\ dialog\ states\} \\
A_{ctions} &:= \{a_m \in all\ output\ actions\} \\
T_{ransitions} &:= P(s_{t+1}|s_t, a_m) \\
R_{eward} &:= r(s, a_m) \\
O_{bservations} &:= \{o \in all\ user\ input\ a_u\} \\
Z &:= P(o_{t+1}|s_{t+1}, a) \\
\lambda &:= geometric\ discount\ factor\ with\ 0 \le \pi \le 1 \\
b_0 &:= initial\ state\ probability\ distribution(belief)
\end{aligned}$$

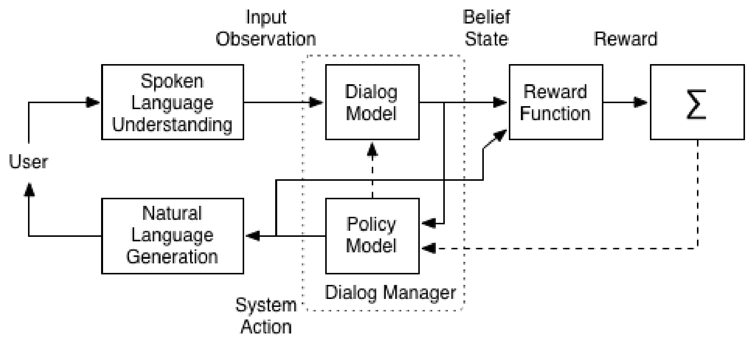
With S, A, T, R as defined in the MDP pattern and $O_{bservations}$ as the set of input actions a user might perform, Z as the probability of observing user action o_{t+1} after performing a_m as part of the previous transition. λ as a discounting factor to optionally emphasize late rewards and b_0 as the initial state probability distribution.

The major difference when compared to other state-based dialog managers is that POMDPs will maintain a probability distribution of all states in b and employ dynamic programming to determine the most likely user goal and system action [3].

When performing the dialog, the system will receive the users input and perform *belief monitoring* to update the state probability distribution in b for each dialog turn. This distribution gets mapped to actions (e.g. a voice prompt). Several stochastic models are needed in order to operationalize the approach and would need to be trained from a dialog corpus or by user simulation.

As it may be computationally intractable to process the whole belief state and associated actions, Young proposes a grid-based approach [3], where actions are points in the belief state and a distance metric can be employed to find the best action. Another extension is the mapping of all state into a summary state space, containing the most likely states corresponding to user goals.

An overview of the principal components in a POMDP dialog system, based on [3] is shown in the following figure:



Consequences

- ☺ Resulting dialog strategy is learned from real data.
- ☺ Robust with respect to recognition or understanding errors
- ☺ Implicitly models and takes into account uncertainty in user recognition
- ☹ The state space tends to get huge as it has to model all dialog states and mappings from beliefs to actions.

- ☹ Users goal is assumed to change infrequently to keep belief monitoring manageable.
- ☹ Recovery strategies are difficult to implement as they have to be part of the original corpus.

Known Uses

Wiliams et al. showed in [2] that the performance of this dialog management is comparable to hand-crafted dialog managers.

The Trainbot system [4] uses this dialog manager to make appropriate dialog turns in a given situation.

Tsiakoulis et al. presented in [1] a voice-based in-car system for providing information about local amenities (e.g. restaurants).

Related Patterns

Partially Observable MDP extends Markov Decision Processes (MDP) by providing the means to model uncertainty as a probability distribution of states.

References

[1] Pirros Tsiakoulis, M Gašic, Matthew Henderson, J Planells-Lerma, Jorge Prombonas, Blaise Thomson, Kai Yu, Steve Young, and Eli Tzirkel. Statistical methods for building robust spoken dialogue systems in an automobile. In *4th International Conference on Applied Human Factors and Ergonomics*, 2012.

[2] Jason D Williams and Steve Young. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21(2):393–422, 2007.

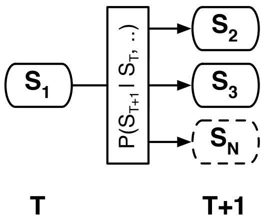
[3] S. Young. Using pomdps for dialog management. In *Spoken Language Technology Workshop, 2006. IEEE*, pages 8 –13, dec. 2006.

[4] Weidong Zhou and Baozong Yuan. Trainbot: A spoken dialog sytem using partially observable markov decision processes. In *Wireless, Mobile and Multimedia Networks (ICWMNN 2010), IET 3rd International Conference on*, pages 381–384. IET, 2010.

Sequence Prediction

Intent

Support an actual dialog strategy by making predictions about future user input.



Context

A state-based multimodal dialog system exists, e.g. Finite State Dialog Management, where 1) multiple strategies can satisfy the users interaction intent or 2) knowledge about potential future steps can be applied to increase the interaction efficiency.

Problem

How to support the dialog manager with knowledge about previously observed interaction strategies employed by a user?

Forces

- System- or user-actions can be performed in several ways.
- Users are likely to stick to the approach they identified first [?].
- The system can support the user in an unobtrusive way by using knowledge about probable future input.

Solution

By learning the observed sequences, a system can provide support to arrive at more concise and effective dialogs. To implement this strategy consider the following:

Establish an N-Gram $P(s_{t+1}|s_t, s_{t-1}..s_{t-N-1}, t(s_t, s_{t+1}) \in T)$ to determine the probability of a state given a history of states and use a sequence prediction algorithms to take a guess at the next state. Support the user by unobtrusively offer short-cuts and interaction support using this knowledge.

1. Count all occurrences of a state in a dialog corpus or online while performing the dialog as the 1-Gram model.
2. Maintain a history and establish the 2..N-Gram models as well.
3. Look-up the N most likely states given the history in the N-Gram and unobtrusively support transitions to these states in the interface.

Consequences

☺ Can support actual dialog managers with their strategy by hinting at future transitions.

☹ Is not suited to perform actual dialog management, but rather a complimentary approach.

☹ The N-Grams need to be established per user as their interaction patterns may differ.

An evaluation and overview of available sequence prediction algorithms is available in e.g. [1].

Related Patterns

Sequence Prediction extends Finite State Dialog Management by supporting an actual dialog manager with predictions about future user input. In contrast to the previous two patterns, this is a complimentary approach to dialog management and not an actual dialog management technique.

References

[1] Melanie Hartmann and Daniel Schreiber. Prediction algorithms for user actions. In Ingo Brunkhorst, Daniel Krause, and Wassiou Sitou, editors, *15th Workshop on Adaptivity and User Modeling in Interactive Systems*, 2007.

3 Conclusion

In this paper we continued the work on our pattern language for dialog management with patterns about probabilistic dialog management. Specifically, we described the following three patterns:

Markov Decision Processes (MDP) extends Frame Based Dialog Management to find an *optimal* dialog strategy to fill a set of information slots with a sequence of mixed-initiative dialog turns with given uncertainty in recognizing the user's actions.

Partially Observable MDP helps to model this uncertainty as a probability distribution of states.

Sequence Prediction is an extension to Finite State Dialog Management by supporting an actual dialog strategy by making predictions about future user input. In contrast to the previous two this is rather a guidance to dialog management.

The major disadvantage of probabilistic approaches, from our point of view, is the opaqueness of the learned dialog strategies, making it hard or even impossible to make small adaptations to the dialog or employ error recovery strategies. Nevertheless, being able to learn an optimal strategy or implicitly modeling the uncertainty can provide very useful when embedded as probabilistic sub-dialogs.

In the future we will further extend our language by describing more recent variances of the described patterns.

Acknowledgements

The authors would like to thank our shepherd Johannes Koskinen for his valuable input. We would also like to thank the members of the writer's workshop at the VikingPLoP 2013 conference for their additional thoughts that helped us improving our work.

References

[1] James O. Coplien. *A generative development-process pattern language*, pages 183–237. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1995.

[2] Melanie Hartmann and Daniel Schreiber. Prediction algorithms for user actions. In Ingo Brunkhorst, Daniel Krause, and Wassiou Sitou, editors, *15th Workshop on Adaptivity and User Modeling in Interactive Systems*, 2007.

[3] Gerard Meszaros and Jim Doble. Pattern languages of program design 3. chapter A pattern language for pattern writing, pages 529–574. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1997.

[4] J Nielsen. Classification of dialog techniques. *ACM SIGCHI Bulletin*, 1987.

[5] T. Reenskaug. Models - views - controllers. Technical report, Xerox Parc, 1979.

[6] Dirk Schnelle-Walka and Stefan Radomski. A pattern language for dialogue management. In *Proceeding of VikingPLoP 2012*, Apr 2012.