A tractable DDN-POMDP approach to affective dialogue modeling for general probabilistic frame-based dialogue systems

Trung H. Bui, Mannes Poel, Anton Nijholt, and Job Zwiers
University of Twente
{buith, mpoel, anijholt, zwiers}@cs.utwente.nl

Abstract

We propose a new approach to developing a tractable affective dialogue model for general probabilistic frame-based dialogue systems. The dialogue model, based on the Partially Observable Markov Decision Process (POMDP) and the Dynamic Decision Network (DDN) techniques, is composed of two main parts, the slot level dialogue manager and the global dialogue manager. Our implemented dialogue manager prototype can handle hundreds of slots; each slot might have many values. A first evaluation of the slot level dialogue manager (1-slot case) showed that with a 95% confidence level the DDN-POMDP dialogue strategy outperforms three simple handcrafted dialogue strategies when the user's action error is induced by stress.

1 Introduction

We aim to develop dialogue management models which are able to act appropriately by taking into account some aspects of the user's affective state. These models are called *affective dialogue models*. Concretely, our affective dialogue manager processes two main inputs, namely the user's action (e.g., dialogue act) and the user's affective state, and selects the most appropriate system action based on these inputs and the context. In human-computer dialogue, this work is difficult because the recognition results of the user's action and affective state are ambiguous and uncertain. Furthermore, the user's affective state cannot directly observe and usually changes over time. Therefore, an affective dialogue model should take into account both the basic dialogue principles (such as turn-taking and grounding) and the dynamic aspects of the user's affect. We found that Partially Observable Markov Decision Processes (POMDPs) are suitable for use in designing these affective dialogue models [Bui *et al.*, 2006].

However, solving the POMDP problem (i.e. finding a near-optimal policy) is computationally expensive. Therefore, almost all developed POMDP dialogue management approaches (mainly for spoken dialogue systems, see [Williams *et al.*, 2005] and the earlier work cited in this paper) are limited to toy frame-based dialogue problems with the size of several slots. Recently, William and Young [2006] proposed a scaling-up POMDP method called CSPBVI to deal with the multi-slot problem. The dialogue manager is decomposed into two POMDP levels, a master POMDP and a set of summary POMDPs. However, they have achieved this goal by oversimplifying the user behavior (assuming when the users are asked about a certain slot, they only provide a value for that slot) and reducing the size of the POMDP structure (e.g. approximating the number of values of each slots by only two values $best$ and $rest$). Furthermore, trials with real users, which allow to validate the system response time, were not conducted.

In our research, we opted for another approach which focuses on real-time online belief update for a general probabilistic frame-based (or slot-filling) dialogue system. Each slot is first formulated as a POMDP and then approximated by a set of Dynamic Decision Networks (DDNs). The approach is, therefore, called the DDN-POMDP approach. It has two new features, (1) being able to deal with a large number of slots and (2) being able to take into account the role of the user's affective state in deriving the adaptive dialogue strategies.

In this paper, we first describe our general affective dialogue model using the DDN-POMDP approach. Then, we present a simulated route navigation example and a first evaluation of our method. Finally, we present conclusions and discuss future work.

2 The DDN-POMDP approach for the frame-based affective dialogue problem

Our Affective Dialogue Model (ADM) is composed of two main parts: (1) the slot level dialogue manager and (2) the global dialogue manager. The first part is composed of a set of n slots $f_1, f_2, ..., f_n$ where each slot f_i is formulated as a POMDP (called the slot-POMDP and denoted by SP_i). The second part, the global dialogue manager, is handcrafted. It aims to keep track of the current dialogue information state and to aggregate the system slot actions nominated by the slot-POMDPs. These two parts and the ADM activity process are explained in detail in the next sections.

2.1 Slot Level Dialogue Manager

The state set of each slot-POMDP SP_i is composed of the user's goals for the slot i (Gu_i), the user's affective states (Eu), the user's actions for the slot i (Au_i), and the user's

Proceedings of the 5^{th} Workshop on Knowledge and Reasoning in Practical Dialogue Systems, pages 34–37,
Hyderabad, January 8th, 2007.

grounding states for the slot i (Du_i). The observation set is composed of the observed user's actions for the slot i (OAu_i) and the observed user's affective states (OEu). Eu and OEu are identical for all slots. The action set is the system actions for the slot i.

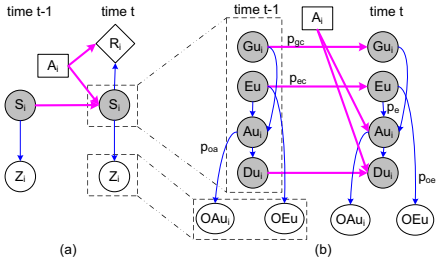

Figure 1: (a) Standard POMDP, (b) Two time-slice of the factored POMDP for slot i

Figure 1b shows a structure of the factored POMDP for slot i of our route navigation example (see Section 3). The features of S_i, A_i, Z_i (Figure 1a), and their correlation form a two time-slice Dynamic Bayesian Network (2TBN). Parameters $p_{gc}, p_{ec}, p_e, p_{oa}$, and p_{oe} are used to produce the transition and observation models in case no real data is available, where p_{gc} and p_{ec} are the probabilities that the user's goal and emotion change; p_e is the probability of the user's action error being induced by emotion; p_{oa} and p_{oe} are the probabilities of the observed action and observed emotional state errors. The reward model depends on each specific application. Therefore it is not specified in our general slot level dialogue manager.

For example, the full-flat slot-POMDP model SP_i of a simplified version of the route navigation example (see Section 3.1) is composed of 61 states (including an absorbing end state), eight actions, and ten observations.

We are interested in finding a solution to directly implement this POMDP model for practical dialogue systems. One intuitive approach is to compute the optimal dialogue strategy using a good approximate POMDP algorithm and use the result for selecting the appropriate system action. We used this approach to find the optimal dialogue policy for the above SP_i [Bui *et al.*, 2006] using Perseus [Spaan and Vlassis, 2005]. However, this approach does not work when the number of slot values and the user's affective states increases (for example, when $|Eu| = 5, m_i = 10$, the full-flat model of SP_i increases up to 1201 states, 22 actions, and 60 observations).

Therefore, to maintain the tractability and allow real-time online belief state update, we approximate each slot-POMDP by a set of $|A_i|$ k-step look-ahead DDNs (kDDNAs) ($k \geq 0$). A kDDNA has $(k + 2)$ slices. The first two slices are similar to the 2TBN showed in Figure 1b, the next k slices are used to predict the user behavior in order to allow the dialogue manager to select the appropriate system action. Figure 2 shows a structure of the kDDNA ($k = 1$) used for SP_i of our route navigation example. The connection from the action nodes to immediate reward nodes in the next slices indicates that when a system slot action is selected that lead to the absorbing `end` state (such as ok or $fail$), the reward in all next slices are equal to `0`.

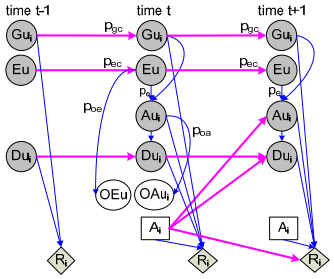

Figure 2: kDDNA with one-step look-ahead ($k = 1$)

2.2 Global Dialogue Manager

The global dialogue manager is composed of two components, the dialogue information state (DIS) and the action selector (AS).

The DIS is considered as the active memory of the dialogue manager, it automatically updates and memorizes the current probability distributions of the user's goal, affective state, action, grounding state of all slots and the recently observed user's action and affective state. The DIS is formally defined by the tuple $\langle P(Gu), p(Eu), P(A_u), P(D_u), oau, oeu \rangle$, where $P(Gu), P(A_u), P(D_u)$ are n dimensional vectors containing the probability distributions of the user's goal, action, grounding state aggregated from Gu_i, Au_i, and Du_i ($i \in [1, n]$), respectively; $p(Eu)$ is the probability distribution of the user's affective state; $oau \in OAu$ and $oeu \in OEu$ are the recent observed user's action and affective state, where OAu is constructed by the user's dialogue act types, slots and slot values [Bui, 2006], Eu and OEu are defined in Section 2.1.

The AS component is responsible for aggregating the system's slot actions nominated by slot-POMDPs. The system action set used by the AS component is constructed by the system's dialogue act types, slots, and slot values [Bui, 2006] and two special actions $giveSolution$ and $stop$. The AS is heuristic and application-dependent. An example of a set of rules to select global system action is described in Section 3.1.

2.3 Affective Dialogue Manager Activity Process

When the dialogue manager is initialized, it loads n slot-POMDP parameter files and creates a set of kDDNAs (m_i kDDNAs are created from the slot-POMDP parameter file i). Depending on each specific application, some slots the values of which can change in time (these slots are called list processing slots, for example the types of food in a selected restaurant [Bui *et al.*, 2004]) can use the same set of kDDNAs. The dialogue manager and the user then only work with a small number of list processing values (i.e. the ordinal numbers), a mapping between these ordinal numbers and the real values is done automatically by the dialogue manager.

The entire process of the affective dialogue manager is explained in this section by a cycle of four steps.

- **Step 1:** When the dialogue manager starts, the kDDNAs nominate greedy actions to the GDM based on the set of prior probability distribution specified in the slot-POMDP parameter files. These actions are combined by the action selector. The output is sent to the user (through the output generation module).

- **Step 2:** The dialogue manager then receives the observed user's action and user's affective state ($oau \in OAu$ and $oeu \in OEu$). The kDDNAs relevant to oau are activated to compute the next slot action. The DIS is also updated.
- **Step 3:** All new actions computed by the selected kDDNAs are sent to the action selector to produce the new system action.
- **Step 4:** The process repeats from step 2 until the GDM selects either $giveSolution$ or $stop$ action.

3 Implementation & Evaluation

The test example is a simulated route navigation in an unsafe tunnel. A serious accident has happened in a huge tunnel. A rescue team (n persons) is sent to the unsafe part of the tunnel to evacuate a large number of injured victims. The rescue members are currently at different locations in the tunnel. The team leader (denoted by "the user") interacts with the dialogue system (located at the operation center) to get the route description for the evacuating task. The system is able to produce the route description when knowing the locations of the rescue members. Furthermore, the system can infer the user's stressful state and use this information to act appropriately.

3.1 Implementation

The above example is formulated as n slots ($f_1, f_2, ..., f_n$) and all slots have the same set of m values which are the locations in the tunnel ($v_1, v_2, ..., v_m$). The user's affective states are five levels of the user's stress: no stress (no), low stress (low), moderate stress ($moderate$), high stress ($high$), and extreme stress ($extreme$). The user's grounding state is composed of two values $notstated$, $stated$. The user's dialogue act type set is $answer$, yes, and no. The system's dialogue act type set is ask, $confirm$, ok, $fail$, $giveSolution$, $stop$ (the two last dialogue act types are only used at the global dialogue manager level as being defined in Section 2.2). The user's goal is to find out the route description for n locations (known by the user). The system aims at showing the user the correct route navigation as soon as possible.

Slot level dialogue manager representation

Slot f_i is represented by $Gu_i = \{v_j | j \in [1, m]\}$, $Eu = \{no, low, moderate, high, extreme\}$, $Au_i = \{answer(v_j), yes, no | j \in [1, m]\}$, $Du_i = \{notstated, stated\}$, $OAu_i = Au_i$, $OEu = Eu$, $A_i = \{ask, confirm(v_j), ok(v_j), fail | j \in [1, m]\}$.

We use two criteria to specify the reward model for each slot, helping the user obtain the correct route description as soon as possible and maintaining the dialogue appropriateness [Williams *et al.*, 2005]. Concretely, if the system `confirms` when the user's grounding state is `notstated`, the reward is `-2`, the reward is `-3` for action `fail`, the reward is `10` for action `ok(x)` where $gu_i =$ `x` ($\texttt{x} \in \{v_j | j \in [1, m]\}$), otherwise the reward is `-50`. The reward for any action taken in the absorbing `end` state is `0`. The reward for any other action is `-1`. The high negative reward for selecting the incorrect slot value (`-50`) is used to force the dialogue manager agent to confirm the information provided by the user when the user's stress level is high.

The probability distributions for each kDDNA are generated using the parameters $p_{gc}, p_{ec}, p_e, p_{oa}, p_{oe}$ defined in Section 2.1 (p_{oa}, p_{oe} can be viewed as the speech recognition error and the stress recognition error) and two new parameters K_{ask} and $K_{confirm}$, where K_{ask} and $K_{confirm}$ are the coefficients associated with the ask and $confirm$ actions (i.e. $p_e(ask) = p_e/K_{ask}; p_e(confirm) = p_e/K_{confirm}$). We assume that when the users are stressful, they make more errors in response to the system ask action than the system $confirm$ action because the number of possible user's action in response to ask is greater than to $confirm$ when the user is not stress.

Global dialogue manager representation

The sets of observed user's actions and system actions are now represented by $OAu = \{answer(I), yes(I), no(I) | I \subseteq \{(f_i = v_i^*) | i \in [1, n]\}, v_i^* \in \{v_i | i \in [1, m]\}\}$, $A = \{ask(I), confirm(J), giveSolution(L), stop | I \subseteq \{f_i | i \in [1, n]\}, J \subseteq \{(f_i = v_i^*) | i \in [1, n]\}, L = \{(f_i = v_i^*) | i \in [1, n]\}, v_i^* \in \{v_i | i \in [1, m]\}\}$

The action selector generates the global system action based on the following rules (applying the first rule that satisfies the set of nominated actions):

1. If all slots nominate ask action then the global action is $ask(f_1, f_2, ..., f_n)$ or $ask(open)$,
2. If all slots nominate $confirm$ action then the global action is $confirm((f_1 = v_1^*), (f_2 = v_2^*), ..., (f_n = v_n^*))$ or $confirm(all)$,
3. If all slots nominate ok action then the global, action is $giveSolution((f_1 = v_1^*), (f_2 = v_2^*), ..., (f_n = v_n^*))$,
4. If some slots $(f_1^*, f_2^*, ..., f_i^*)$ nominate $confirm$ action with the values $(v_1^*, v_2^*, ..., v_i^*)$ then the global action is $confirm((f_1^* = v_1^*), (f_2^* = v_2^*), ..., (f_i^* = v_i^*))$,
5. If some slots $(f_1^*, f_2^*, ..., f_i^*)$ nominate ask action then the global action is $ask(f_1^*)$,
6. Otherwise, the global action is $stop$.

The current version of our implemented dialogue manager prototype is able to handle hundreds of slots, each slot can have many values. When a slot has hundreds or thousands of values (called many-value slot), directly embedding these values into the kDDNAs will lead to a significant delay in the belief update time. One of our solutions in this case is to formulate the many-value slot as a list processing slot as mentioned in Section 2.3. A dialogue example of the 10-slot case ($n = 10, m = 10, p_{gc} = 0, p_{ec} = p_e = p_{oa} = p_{oe} = 0.1, K_{ask} = 1, K_{confirm} = 10, k = 1$) is described in [Bui, 2006].

3.2 Evaluation

The performance of the DDN-POMDP dialogue strategy depends on both the global dialogue manager and the slot level dialogue manager (see Section 2).

Currently a simulated user model for the general n-slot case which is appropriate for a quantitative evaluation of the DDN-POMDP approach has not been available yet, therefore in this section we first evaluate the slot level dialogue manager by comparing the DDN-POMDP dialogue strategy with

a random dialogue strategy and three simple handcrafted dialogue strategies for 1-slot case: (a) SDS-HC1 (first ask and then select ok action if $oau = answer$), (b) SDS-HC2 (first ask, then $confirm$ if $oau = answer$ and then select ok action if $oau = yes$), (c) ADS-HC (first ask, then $confirm$ if $oau = answer$ & $oeu = stress$ and select ok action if $oau = yes$).

The evaluation is conducted by letting each dialogue strategy interact with the same simulated user (the simulated user model is constructed using the 2TBN described in Figure 1b).

Figure 3 shows the average return of 10000 dialogue episodes of five dialogue strategies when the probability of the user's action error being induced by stress p_e changes from 0 (stress has no influence to the user action selection) to 0.8 (stress has high influence to the user action selection). The results of the average return (Figure 3) show that with a 95% confidence level the DDN-POMDP dialogue strategy outperforms all other remaining dialogue strategies when $p_e \geq 0.1$. The DDN-POMDP copes well when the user's action error being induced by stress increases. An example of the interaction between the DDN-POMDP dialogue manager and the simulated user (10 dialogue episodes) is shown in [Bui, 2006].

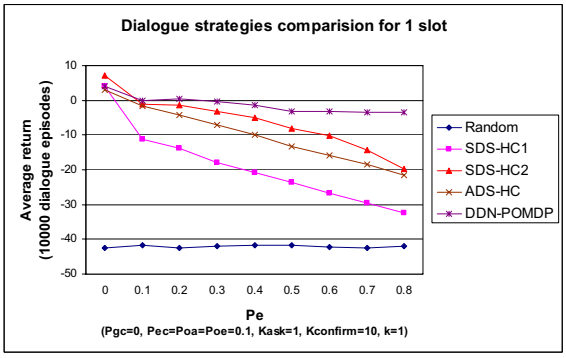

Figure 3: Average return of five dialogue strategies ($p_e \in [0, 0.8]$)

Figure 4 shows that the DDN-POMDP dialogue strategy also copes well with the observed user's action error p_{oa} (for example, the ASR error). The performance of all strategies in Figure 3 and 4 is low when p_e, p_{oa} increases because we set a strong negative reward when the system chooses incorrect solution and when the user's stress is $extreme$, the user acts randomly. When the observed user's action error is too high ($p_{oa} \geq 0.6$), the DDN-POMDP dialogue manager always selects $fail$ action therefore the average return is a constant (equal to -4). One interesting point is that the dialogue strategy SDS-HC2 copes well with the change of p_e (Figure 3) but its performance decreases rapidly when p_{oa} increases (Figure 4).

4 Conclusions and future work

The presented DDN-POMDP approach is shown to be able to handle a large number of slots and keep track of the user's affective state. A first evaluation has been conducted to compare the DDN-POMDP performance with three simple handcrafted dialogue strategies when the user's action error is induced by stress. We plan to evaluate the model with an n slots case by comparing the DDN-POMDP dialogue strategy

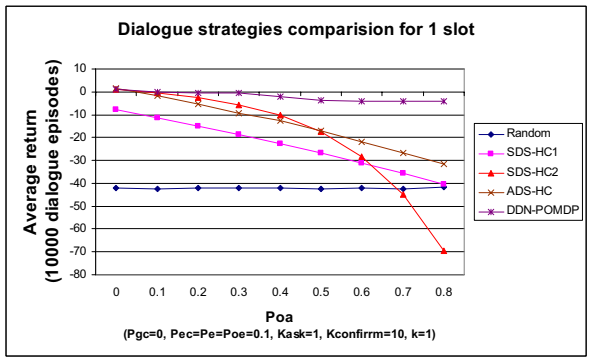

Figure 4: Average return of five dialogue strategies ($p_{oa} \in [0, 0.8]$)

with other well-developed handcrafted dialogue strategies for frame-based dialogue systems such as [Bui *et al.*, 2004]. Another issue is to study the real user behavior in crisis such as in the air traffic control domain in order to improve the user simulator.

Although it is hard to handle really complex dialogue systems using only POMDPs, this approach sheds light to a hybrid solution by combining traditional rule-based and POMDP approaches which hopefully can solve a part of many challenges in developing affective dialogue systems.

Acknowledgments

This work is part of the ICIS program (http://www.icis.decis.nl). ICIS is sponsored by the Dutch government under contract BSIK 03024.

References

[Bui *et al.*, 2004] T.H. Bui, M. Rajman, and M. Melichar. Rapid dialogue prototyping methodology. In *Proceedings of the 7th International Conference on Text, Speech & Dialogue (TSD)*, pages 579–586, Brno, Czech Republic, September 8-11 2004.

[Bui *et al.*, 2006] T.H. Bui, J. Zwiers, M. Poel, and A. Nijholt. Toward affective dialogue modeling using partially observable markov decision processes. In *Proceedings of Workshop Emotion and Computing, 29th Annual German Conference on Artificial Intelligence*, 2006.

[Bui, 2006] T.H. Bui. A tractable ddn-pomdp approach to affective dialogue modeling for general probabilistic frame-based dialogue systems. Technical report, University of Twente, 2006. (to appear).

[Spaan and Vlassis, 2005] M.T.J. Spaan and N. Vlassis. Perseus: Randomized point-based value iteration for pomdps. *Journal of Artificial Intelligence Research*, 24:195–220, 2005.

[Williams and Young, 2006] J.D. Williams and S. Young. Scaling pomdps for dialog management with composite summary point-based value iteration (cspbvi). In *AAAI Workshop on Statistical and Empirical Approaches for Spoken Dialogue Systems*, 2006.

[Williams *et al.*, 2005] J.D. Williams, P. Poupart, and S. Young. Factored partially observable markov decision processes for dialogue management. In *4th Workshop on Knowledge and Reasoning in Practical Dialog Systems*, 2005.