

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

Department of Computer Science

{buih, 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. It has two new features: (1) being able to deal with a large number of slots and (2) being able to take into account some aspects of the user’s affective state in deriving the adaptive dialogue strategies. 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 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 [Picard, 1997]. 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 [Bunt, 1994]) 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 can change over time. Therefore, an affective dialogue model should take into account both the basic dialogue principles (such as turn-taking [Sacks *et al.*, 1974] and grounding [Clark and Schaefer, 1989]) and the dynamic aspects of the user’s affective state. 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 the optimal policy) is computationally expensive. Therefore,

almost all developed POMDP dialogue management approaches (mainly for spoken dialogue systems) are limited to toy frame-based dialogue problems with the size of several slots.

Our approach focuses on real-time online belief update for a general probabilistic frame-based dialogue system which is composed of a set of slots¹. The term “probabilistic frame-based” is used because instead of keeping track of the slot values provided by the user, the dialogue manager maintains their probability distributions. Each slot is first formulated as a POMDP and then approximated by a set of Dynamic Decision Networks (DDNs) [Kanazawa and Dean, 1989]. 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 give a short overview of POMDP, DDN, and their applications to the dialogue management and affective user modeling problems. Second, a general affective dialogue model using the DDN-POMDP approach is described. Then, we present a simulated route navigation example and a first evaluation of our method. Finally, we discuss the conclusions and future work.

2 POMDP, DDN, dialogue management, and affective user modeling

A POMDP is defined by the tuple $\langle S, A, Z, T, O, R \rangle$, where S is the set of states (of the environment), A is the set of the agent’s actions, Z is the set of observations the agent can experience of its environment, T is the transition model, O is the observation model, and R is the reward model (Figure 1a).

In a dialogue management context, the agent is the system (i.e., the dialogue manager) and a part of the POMDP environment represents the user’s state. The system uses a state estimator (SE) to compute its internal belief about the user’s current state and a policy π to select actions. SE takes as its input the previous belief state, the most recent action and the

¹when designing a frame-based dialogue system, the application is usually formulated by a set of frames, each frame is composed of a set of relevant slots. The set of frames can easily convert to a set of slots by using standard database normalization procedures [Bui *et al.*, 2004]

most recent observation, and returns an updated belief state. The policy π selects actions based on the system’s current belief state. Two of the main tasks of a POMDP are computing belief states and finding an optimal policy (i.e., optimal dialogue strategy). These two tasks are explained in [Cassandra *et al.*, 1994].

Young *et al.* [2005] have argued that, nearly all existing dialogue management systems, especially the information state approach [Traum and Larsson, 2003] can be considered as a direct implementation of the POMDP-based model with the deterministic dialogue policy. These systems have a number of “severe weaknesses” such as using unreliable confidence measures, having difficulty coping with the dynamic changing of the user’s goal and intention, and tuning the dialogue policy is labor extensive based on off-line analysis of the system logs [Young *et al.*, 2005].

The first work that applied POMDP for the dialogue management problem was the robot home-assistant application [Roy *et al.*, 2000]. The work following this track is [Pineau *et al.*, 2001; Zhang *et al.*, 2001; Williams and Young, 2006]. All these approaches (mainly focused on spoken frame-based dialogue systems) have shown that the performance of POMDP-based dialogue strategies outperform the MDP counterpart (e.g. [Pietquin, 2004]), and furthermore these strategies cope well with various error problems in a spoken dialogue system, especially the speech recognition error problem. However, finding the optimal dialogue strategy either using exact or approximate POMDP algorithms is computationally expensive, therefore all of this work is limited to dialogue problems with the size of several slots (for example, two slots in [Williams *et al.*, 2005], three slots in [Pineau *et al.*, 2001; Zhang *et al.*, 2001], and four slots in [Roy *et al.*, 2000]).

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. Each summary POMDP corresponds to a slot. 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*).

Beside using the POMDP framework, another technique to deal with partially observable situations is to use the DDNs. A DDN is an extended model of the Dynamic Bayesian Network (DBN) with decision and utility nodes. DDNs provide a concise representation for large POMDPs and can be used as inputs for any POMDP algorithm such as value-iteration and policy-iteration based algorithms [Russell and Norvig, 2003]. A special case of the DDNs (selecting actions based on immediate reward or equivalent to a POMDP with the discount factor $\gamma = 0$) was used to model the multimodal dialogue systems [Paek and Horvitz, 2000] and the hidden information state dialogue manager [Young *et al.*, 2005]. DDNs and DDNs are also suitable for use in developing the affective user model [Zhou and Conati, 2003; Li and Ji, 2005; Rosis *et al.*, 2006].

3 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, \dots, 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.

3.1 Slot Level Dialogue Manager

We use the factored POMDP [Boutilier and Poole, 1996] for representing each slot f_i . The state set and observation set are composed of six features. The state set is composed of the user’s goals for the slot i (Gu_i), the user’s emotional states² (Eu), the user’s actions for the slot i (Au_i), and the user’s 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 emotional states (OEu). Two of these six features (Eu and OEu) are identical for all slots. The action set is the system actions for the slot i (A_i).

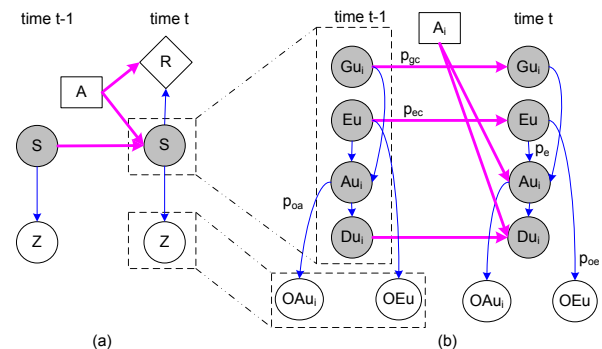


Figure 1: (a) Standard POMDP, (b) Two time-slice of the factored POMDP for slot i , where state set S is factored into four features Gu_i , Eu , Au_i , and Du_i , observation set Z is factored into two features OAu_i and OEu

Figure 1b shows a structure of the factored POMDP for slot i of our route navigation example (see Section 4.1). The features of the state set, action set, observation set, and their correlation form a two time-slice Dynamic Bayesian Network (2TBN). We can easily modify this 2TBN for representing other correlations, for example the correlation between the user’s goal for slot i and the user’s emotional state. 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

²we use the term “emotional state” and “affective state” interchangeably, which is defined in Picard [1997].

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.

Suppose slot f_i has m_i values $\{v_{ij}|j \in [1, m_i]\}$, the above six features and the action set A_i is formally formulated as following ($|X|$ denotes the number of elements of X):

- $Gu_i = \{gu_{ij}|j \in [1, |Gu_i|]\} = \{v_{ij}|j \in [1, m_i]\}$,
- $Eu = \{eu_j|j \in [1, |Eu|]\}$ (we are currently working with a discrete set of the user's emotional states),
- $Au_i = \{au_{ij}|j \in [1, |Au_i|]\} = \{userDAType(v^*)|v^* \in \{\emptyset, v_{ij}|j \in [1, m_i]\}\}$ ($userDAType$ is a set of the user's dialogue act types such as *ask*, *inform*, *answer* and $x(\emptyset)$ is equivalent to x , for example *ask*(\emptyset) is equivalent to *ask*),
- $Du_i = \{du_{ij}|j \in [1, |Du_i|]\}$ (du_{ij} is a grounding state such as *notstated*, *stated*, *confirmed*),
- $OAu_i = \{oau_{ij}|j \in [1, |OAu_i|]\}$ (the value oau_{ij} depends on which input level the observed user's action is sent to the dialogue manager. For example if the observed user's action is sent by the Automatic Speech Recognizer (ASR), the oau_{ij} is the word-graph or N-best hypotheses of the user's utterance. In our model, we assume a high input level for the observed user's action such as the output from the dialogue act recognition module or the intention level in the simulated user model [Eckert *et al.*, 1997], in this case OAu_i has the same set of values as Au_i),
- $OEu = \{oeu_j|j \in [1, |OEu|]\}$ (similar to oau_{ij} , the observed user's emotional state can be represented by a set of observable effects such as response speech, speech pitch, speech volume, posture, and gesture [Ball, 2003]. In our current model, we assume that the observed user's emotional states are the output of the emotion recognition module, and therefore OEu has the same set of values as Eu), and
- $A_i = \{a_{ij}|j \in [1, |A_i|]\} = \{systemDAType(v^*)|v^* \in \{\emptyset, v_{ij}|j \in [1, m_i]\}\}$ ($systemDAType$ is a set of the system dialogue act types, the values of which are similar to $userDAType$).

For example, a simplified version of SP_i of the route navigation example (the detailed version of SP_i is described in Section 4.2) is represented by $S = \langle Gu_i \times Au_i \times Eu \times Du_i \rangle = \langle \{v_1, v_2, v_3\} \times \{answer(v_1), answer(v_2), answer(v_3), yes, no\} \times \{stress, nostress\} \times \{notstated, stated\} \rangle$, $A = \{ask, confirm(v_1), confirm(v_2), confirm(v_3), ok(v_1), ok(v_2), ok(v_3), fail\}$, and $O = \langle OAu_i \times OEu \rangle = \langle \{answer(v_1), answer(v_2), answer(v_3), yes, no\} \times \{stress, nostress\} \rangle$. The full-flat model of this version 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 [Allen *et al.*, 2001]. One intuitive approach is to compute the optimal dialogue strategy using a good approximate

POMDP algorithm (the exact POMDP algorithms such as the Witness [Cassandra *et al.*, 1994] cannot be applied even for the above slot-POMDP) and to use the result (usually in the form of a policy graph and value function table³) 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] which is one of the current best approximate Point-Based Value Iteration POMDP algorithms. However, this approach does not work when the number of slot values and the user's affective states increases (for example, with $|Eu| = 5, m_i = 10$, 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 k -step look-ahead Dynamic Decision Network (kDDN) ($k \geq 0$). A kDDN 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. We observed that the dialogue manager only needs to update its current belief state relevant to the system's last action. Each kDDN (of slot i) is, therefore, further decomposed into a set of $|A_i|$ action DDNs (denoted by kDDNAs). Figure 2 shows a structure of the kDDN and 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.

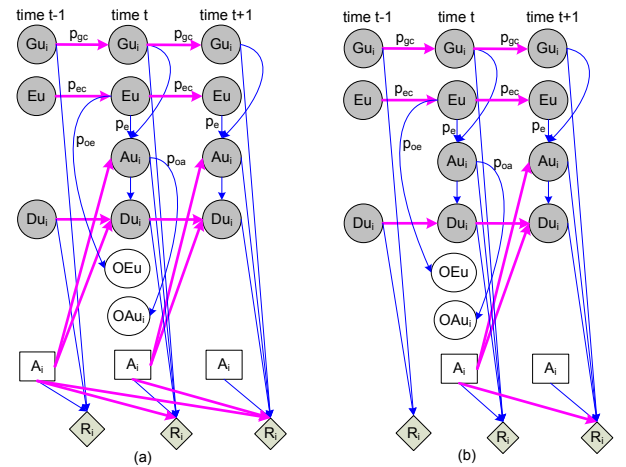


Figure 2: The structure of (a) kDDN and (b) kDDNA with one-step look-ahead ($k = 1$). Both networks have similar structures except the kDDNA does not have the action node in the first slice. In our implemented dialogue manager prototype we use directly the simpler network kDDNA (to gain the computation time for the belief update process) because the ADM can keep track of the last system action, therefore it can update directly on the relevant kDDNA instead of kDDN.

³<http://www.cassandra.org/pomdp/code/index.shtml>

3.2 Global Dialogue Manager

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

The dialogue information state component is considered as the active memory of the dialogue manager, it updates and memorizes the current probability distributions of the user’s goal, emotional state, action, grounding state of all slots and the recently observed user’s action and emotional 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 emotional state; $oau \in OAu$ and $oeu \in OEu$ are the recent observed user’s action and emotional state, where $OAu = \{userDAType(I) | I \subset \{f_i, (f_i = v_i^*) | i \in [1, n]\}, v_i^* \in \{v_j | j \in [1, m_i]\}\}$, Eu and OEu are defined in Section 3.1.

The action selector component is responsible for aggregating the system’s slot actions nominated by slot-POMDPs. The system action set generated by the AS component is represented by $A = \{systemDAType(I), giveSolution(L), stop | I \subset \{f_i, (f_i = v_i^*) | i \in [1, n]\}, L = \{(f_i = v_i^*) | i \in [1, n]\}, v_i^* \in \{v_j | j \in [1, m_i]\}\}$

The action selector is heuristic and application-dependent. An example of a set of rules to select global system action is described in Section 4.2.

3.3 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 that have similar structure such as `list processing slots` [Bui *et al.*, 2004] can use the same set of kDDNAs.

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.

4 Implementation & Evaluation

4.1 Example: Route navigation in an unsafe tunnel

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. In this example, the system can *ask* the user about the location(s) of the rescue member(s), *confirm* a location or a list of locations provided by the user, give the route description (*giveSolution*) to the user, and *stop* the dialogue (and connect the user with the operator).

4.2 Implementation

The route navigation example in Section 4.1 is formulated as n slots (f_1, f_2, \dots, f_n) and all slots have the same set of m values (the locations in the tunnel) (v_1, v_2, \dots, v_m). The user’s affective states are five levels of the user’s stressful situation: 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 3.2). 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

Using the formal formulation described in Section 3.1, 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 $gu_i = x$ ($x \in \{v_j | j \in [1, m]\}$), 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 3.1 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. $p_e = K_{ask} \times p_e(ask) = K_{confirm} \times p_e(confirm)$). We assume that when the users are stressful, they make more errors in response to the system *ask* action than the system *confirm* action (i.e. $K_{ask} \leq K_{confirm}$).

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

Our dialogue manager prototype is a distributed multi-agent system developed using the Java programming language and the middleware iROS platform⁴. The dialogue manager agent exchanges messages with other input and output agents using the iROS Event Heap, a blackboard-like communication mechanism. The kDDNAs are created and integrated with the dialogue manager using the GeNie&SMILE Application Programming Interface⁵. 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 [Bui *et al.*, 2004]. 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. 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 Appendix B.

⁴<http://sourceforge.net/projects/iros>

⁵<http://genie.sis.pitt.edu>

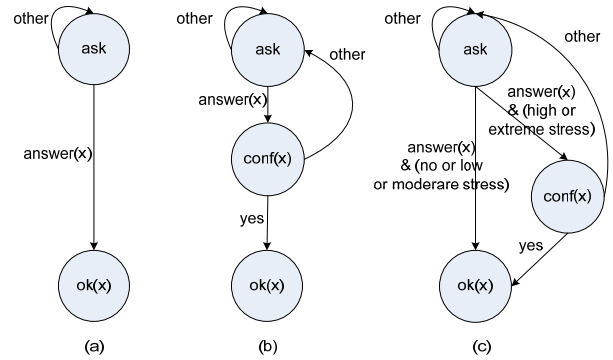


Figure 3: Three handcrafted dialogue strategies for the 1-slot case: (a) SDS-HC1 (first *ask* and then select *ok* action if $userDAType = answer$), (b) SDS-HC2 (first *ask*, then *confirm* if $userDAType = answer$ and then select *ok* action if $userDAType = yes$), (c) ADS-HC (first *ask*, then *confirm* if $userDAType = answer$ & $oeu = stress$ and select *ok* action if $userDAType = yes$). x is the slot value and *conf* means *confirm*.

4.3 Evaluation

The performance of the DDN-POMDP dialogue strategy depends on both the global dialogue manager and the slot level dialogue manager (see Section 3).

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 other handcrafted dialogue strategies SDS-HC1, SDS-HC2, and ADS-HC (Figure 3) for 1-slot case.

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 4 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 4) and the standard deviation (Table 1) 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 Appendix A.

Figure 5 shows that the DDN-POMDP dialogue strategy also copes well with the observed user's action error p_{oa} . 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

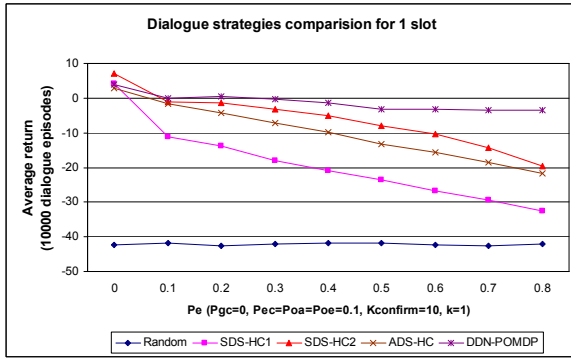


Figure 4: The average return of five dialogue strategies ($p_e \in [0, 0.8]$), the standard deviation is shown in Table 1

p_e	Random	SDS-HC1	SDS-HC2	ADS-HC	DDN-POMDP
0.0	20.73	16.26	2.92	13.81	16.33
0.1	21.28	28.34	20.18	19.09	14.99
0.2	20.72	29.14	20.02	21.38	11.42
0.3	21.27	29.87	21.47	23.19	11.89
0.4	21.30	30.03	22.77	24.68	10.90
0.5	21.29	29.93	25.47	25.90	9.44
0.6	20.92	29.51	26.49	26.77	9.35
0.7	20.59	28.85	29.87	27.31	9.56
0.8	21.00	27.84	34.16	27.66	9.55

Table 1: The standard deviation (10000 dialogue episodes)

copies well with the change of p_e (Figure 4) but its performance decreases rapidly when p_{oa} increases (Figure 5).

5 Conclusions and future work

We have presented a DDN-POMDP approach to affective dialogue modeling and illustrated our affective dialogue model by a simulated route navigation example. Our implemented dialogue manager prototype is able to handle a large number of slots, each slot might have many values. A first evaluation at the slot level dialogue manager (1-slot case) showed that with a 95% confidence level the DDN-POMDP dialogue strategy outperforms three handcrafted dialogue strategies when the user's action error is induced by stress.

The next issues we plan to tackle are: (1) evaluating the model with an n slots case by comparing the DDN-POMDP dialogue strategy with other well-developed handcrafted dialogue strategies for frame-based dialogue systems such as [Bui *et al.*, 2004]; (2) collecting and generating both real and artificial data to build and train the model as well as to validate the model design; (3) extending the model representation, especially by adding more specific features related to the user's goal, emotion, & specifying their correlations; and (4) connecting the dialogue manager with a new version of the talking head being developed in our group [Nguyen, 2006] to allow the system to express emotion.

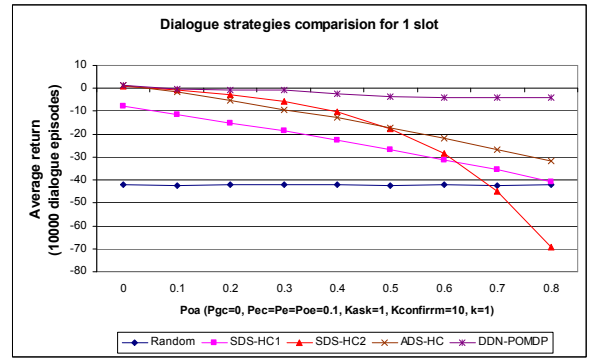


Figure 5: The average return of five dialogue strategies ($p_{oa} \in [0, 0.8]$)

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.

A Interaction between the DDN-POMDP dialogue manager S and the simulated user U (1-slot case, 10 values)

The following example shows 10 dialogue episodes of the interaction between the DDN-POMDP dialogue manager and the simulated user. In all dialogue episodes, the prior probability distributions of G_u and E_u are equally distributed. The initial state of D_u is *notstated*. Based on these probability distributions and the initial state, the dialogue manager selects *ask* action in its first turn. The hidden part is the information generated by the simulated user and it is unobservable to the dialogue manager. The example illustrated some adaptive behavior of the DDN-POMDP dialogue manager such as it chooses the *fail* action when $oeu = extreme$ (in dialogue episodes 1, 3, and 4), the *ok* action when $oeu = no$ or $oeu = low$ (in dialogue episodes 5, 8, and 9), the *confirm* action when $oeu = moderate$ or $oeu = high$ (in dialogue episodes 2, 6, 7, and 10). The average time for each turn is about 93 ms (38 turns in 10 dialogue episodes, the total interaction time is 3531 ms).

Dialogue episode 1

S1: *ask*

U1: $oau=v_6; oeu=extreme; r=-1.0$

[Hidden : $gu=v_6, eu=extreme, au=v_6, du=stated$]

S2: *fail*

End of the episode, episodeReward=-4.0, sumReward=-4.0

Dialogue episode 2

S1: *ask*

U1: $oau=v_9; oeu=high; r=-1.0$

[Hidden : $gu=v_9, eu=high, au=v_9, du=stated$]

S2: *confirm*(v_9)

U2: $oau=yes; oeu=high; r=-1.0$

[Hidden : $gu=v_9, eu=high, au=yes, du=stated$]
S3: *ok*(v_9)
End of the episode, episodeReward=8.0, sumReward=4.0

Dialogue episode 3

S1: *ask*
U1: $oau=v_3; oeu=extreme; r=-1.0$
[Hidden : $gu=v_2, eu=extreme, au=v_3, du=stated$]
S2: *fail*
End of the episode, episodeReward=-4.0, sumReward=0.0

Dialogue episode 4

S1: *ask*
U1: $oau=v_4; oeu=extreme; r=-1.0$
[Hidden : $gu=v_7, eu=extreme, au=v_4, du=stated$]
S2: *fail*
End of the episode, episodeReward=-4.0, sumReward=-4.0

Dialogue episode 5

S1: *ask*
U1: $oau=v_7; oeu=low; r=-1.0$
[Hidden : $gu=v_7, eu=low, au=v_7, du=stated$]
S2: *ok*(v_7)
End of the episode, episodeReward=9.0, sumReward=5.0

Dialogue episode 6

S1: *ask*
U1: $oau=v_1; oeu=high; r=-1.0$
[Hidden : $gu=v_6, eu=extreme, au=v_1, du=stated$]
S2: *confirm*(v_1)
U2: $oau=no; oeu=high; r=-2.0$
[Hidden : $gu=v_6, eu=high, au=no, du=stated$]
S3: *fail*
End of the episode, episodeReward=-6.0, sumReward=-1.0

Dialogue episode 7

S1: *ask*
U1: $oau=v_1; oeu=moderate; r=-1.0$
[Hidden : $gu=v_1, eu=moderate, au=v_1, du=stated$]
S2: *confirm*(v_1)
U2: $oau=yes; oeu=low; r=-1.0$
[Hidden : $gu=v_1, eu=moderate, au=yes, du=stated$]
S3: *ok*(v_1)
End of the episode, episodeReward=8.0, sumReward=7.0

Dialogue episode 8

S1: *ask*
U1: $oau=v_1; oeu=no; r=-1.0$
[Hidden : $gu=v_1, eu=high, au=v_1, du=stated$]
S2: *ok*(v_1)
End of the episode, episodeReward=9.0, sumReward=16.0

Dialogue episode 9

S1: *ask*
U1: $oau=v_6; oeu=no; r=-1.0$
[Hidden : $gu=v_6, eu=no, au=v_6, du=stated$]
S2: *ok*(v_6)
End of the episode, episodeReward=9.0, sumReward=25.0

Dialogue episode 10

S1: *ask*
U1: $oau=v_3; oeu=moderate; r=-1.0$
[Hidden : $gu=v_1, eu=high, au=v_3, du=stated$]
S2: *confirm*(v_3)
U2: $oau=no; oeu=high; r=-2.0$
[Hidden : $gu=v_1, eu=high, au=no, du=stated$]
S3: *fail*
End of the episode, episodeReward=-6.0, sumReward=19.0

End of the simulation

Average return: 1.9, Standard deviation: 6.74, Duration: 3531 ms.

B Interaction between the DDN-POMDP dialogue manager S and a real user U (10-slot case)

S1: *ask*(*open*) (rule 1)
U1: $oau=answer((f_1=v_1), (f_2=v_2)), oeu=high$
S2: *confirm*(($f_1=v_1$), ($f_2=v_2$)) (rule 4)
U2: $oau=yes, oeu=moderate$
S3: *ask*(f_3) (rule 5)
U3: $answer((f_3=v_3), (f_4=v_4), (f_5=v_5)), oeu=low$
S4: *ask*(f_6) (rule 5)
U4: $answer((f_6=v_6), (f_7=v_7), (f_8=v_8), (f_9=v_9), (f_{10}=v_{10})), oeu=no$
S5: *giveSolution*(($f_1=v_1$), ($f_2=v_2$), ($f_3=v_3$), ($f_4=v_4$), ($f_5=v_5$), ($f_6=v_6$), ($f_7=v_7$), ($f_8=v_8$), ($f_9=v_9$), ($f_{10}=v_{10}$)) (rule 3)

The above dialogue example shows the interaction between the DDN-POMDP dialogue manager and the first author of this paper. The prior probability distributions of 10 slot-POMDP parameter files are equally distributed, therefore all slots nominate *ask* action, the AS applies the first rule specified in Section 4.2 (rule 1) and the *ask*(*open*) action is selected. Rules 4, 5, 5, and 3 are used in *S2*, *S3*, *S4* and *S5*, respectively.

References

- [Allen *et al.*, 2001] James Allen, Donna Byron, Myroslava Dzikovska, George Ferguson, Lucian Galescu, and Amanda Stent. Towards conversational human-computer interaction. *AI Magazine*, 2001.
- [Ball, 2003] Eugene Ball. A bayesian heart: Computer recognition and simulation of emotion. In Paolo Petta Robert Trappl and Sabine Payr, editors, *Emotions in Hu-*

- mans and Artifacts*, chapter 11, pages 303–332. The MIT Press, 2003.
- [Boutilier and Poole, 1996] Craig Boutilier and David Poole. Computing optimal policies for partially observable decision processes using compact representations. In *AAAI/IAAI, Vol. 2*, pages 1168–1175, 1996.
- [Bui et al., 2004] Trung H. Bui, Martin Rajman, and Miroslav 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] Trung H. Bui, Job Zwiers, Mannes Poel, and Anton 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.
- [Bunt, 1994] Harry C. Bunt. Context and dialogue control. *THINK Quarterly*, 3:19–31, 1994.
- [Cassandra et al., 1994] Anthony R. Cassandra, Leslie Pack Kaelbling, and Michael L. Littman. Acting optimally in partially observable stochastic domains. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, volume 2, pages 1023–1028, Seattle, Washington, USA, 1994. AAAI Press/MIT Press.
- [Clark and Schaefer, 1989] Herbert H. Clark and Edward F. Schaefer. Contributing to discourse. *Cognitive Science*, 13:259–294, 1989.
- [Eckert et al., 1997] Wieland Eckert, Esther Levin, and Roberto Pieraccini. User modelling for spoken dialogue system evaluation. In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 80–87, 1997.
- [Kanazawa and Dean, 1989] Keiji Kanazawa and Thomas Dean. A model for projection and action. In *Proceedings of the Eleventh International Joint Conferences on Artificial Intelligence (IJCAI-89)*, pages 985–990, August 1989.
- [Li and Ji, 2005] Xiangyang Li and Qiang Ji. Active affective state detection and user assistance with dynamic bayesian networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 35(1):93–105, 2005.
- [Nguyen, 2006] Anh Duc Nguyen. Improving a 3d talking head for using in an avatar of virtual meeting room. Technical report, Human Media Interaction, Department of Computer Science, University of Twente, 2006.
- [Paek and Horvitz, 2000] Tim Paek and Eric Horvitz. Conversation as action under uncertainty. In *UAI '00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 455–464, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.
- [Picard, 1997] Rosalind W. Picard. *Affective Computing*, chapter 1, page 24. The MIT Press, 1997.
- [Pietquin, 2004] Olivier Pietquin. *A Framework for Unsupervised Learning of Dialogue Strategies*. PhD thesis, Universitaires de Louvain, 2004.
- [Pineau et al., 2001] Joelle Pineau, Nicholas Roy, and Sebastian Thrun. A hierarchical approach to pomdp planning and execution. In *Workshop on Hierarchy and Memory in Reinforcement Learning (ICML)*, June 2001.
- [Rosis et al., 2006] Fiorella Rosis, Nicole Novielli, Valeria Carofiglio, Addolorata Cavalluzzi, and Berardina D. Carolis. User modeling and adaptation in health promotion dialogs with an animated character. *Journal of Biomedical Informatics*, In Press, Corrected Proof, 2006.
- [Roy et al., 2000] Nicholas Roy, Joelle Pineau, and Sebastian Thrun. Spoken dialogue management using probabilistic reasoning. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, 2000.
- [Russell and Norvig, 2003] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*, chapter 17, page 630. Prentice Hall, 2nd edition, 2003.
- [Sacks et al., 1974] Harvey Sacks, Emanuel A. Schegloff, and Gail Jefferson. A simplest systematics for the organization of turn-taking for conversation. *Language*, 50(4):696–735, December 1974.
- [Spaan and Vlassis, 2005] Matthijs T. J. Spaan and Nikos Vlassis. Perseus: Randomized point-based value iteration for pomdps. *Journal of Artificial Intelligence Research*, 24:195–220, 2005.
- [Traum and Larsson, 2003] David Traum and Staffan Larsson. The information state approach to dialogue management. In Jan van Kuppevelt and Ronnie W. Smith, editors, *Current and New Directions in Discourse and Dialogue*, chapter 15, pages 325–353. Kluwer Academic Publishers, 2003.
- [Williams and Young, 2006] Jason D. Williams and Steve 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] Jason D. Williams, Pascal Poupart, and Steve Young. Factored partially observable markov decision processes for dialogue management. In *4th Workshop on Knowledge and Reasoning in Practical Dialog Systems*, 2005.
- [Young et al., 2005] Steve Young, Jason D. Williams, Jost Schatzmann, Matt Stuttle, and Karl Weilhammer. The hidden information state approach to dialogue management. Technical Report CUED/F-INFENG/TR.544, University of Cambridge, 2005.
- [Zhang et al., 2001] Bo Zhang, Qingsheng Cai, Jianfeng Mao, and Baining Guo. Spoken dialog management as planning and acting under uncertainty. In *Proceedings of Eurospeech*, 2001.
- [Zhou and Conati, 2003] Xiaoming Zhou and Cristina Conati. Inferring user goals from personality and behavior in a causal model of user affect. In *IUI '03: Proceedings of the 8th international conference on Intelligent user interfaces*, pages 211–218, New York, NY, USA, 2003. ACM Press.