Spoken Dialog Management for Robots

Nicholas Roy and Joelle Pineau and Sebastian Thrun
Robotics Institute, Carnegie Mellon University
Pittsburgh, PA 15217
{nicholas.roy | joelle.pineau | sebastian.thrun} @cs.cmu.edu

Abstract

Spoken dialog managers have benefited from stochastic planners such as MDPs. However, so far, MDPs do not handle well noisy and ambiguous utterances from the user. We address this problem by inverting the notion of dialog state; the state represents the *user*'s intentions, rather than the system state. This approach allows for simple and intuitive dialog description at the sacrifice of state observability.

We use a POMDP-style approach to generate dialog policies; however, the intractability of POMDP solutions requires an approximate solution. We instead augment the state representation of the MDP by providing the system with the maximum likelihood state and a compressed representation of the belief state. In this way, the system can approximate the optimal POMDP solution, but at MDP-like speeds.

Introduction

The development of automatic speech recognition has made possible more natural human-computer interaction, and more recently, rich interaction between humans and robots. Speech recognition and speech understanding, however, are not yet at the point where a computer can reliably extract the intended meaning from every human utterance. Human speech can be both noisy and ambiguous, and many real-world systems, such as on a robot in a workplace, must also be speaker-independent. Regardless of these difficulties, any system that manages human-robot dialogs must be able to perform reliably even with noisy and stochastic speech input.

Recent research in dialog management has shown that Markov Decision Processes can be useful for generating policies (Young 1990; Levin, Pieraccini, & Eckert 1998); the actions are the speech productions from the system, the state is represented by the dialog as a whole, and the goal is to maximize some reward, such as fulfilling some request of the user. However, the correct way to represent the state of the dialog is still an open problem (Singh *et al.* 1999). A common solution is to restrict the system to a single goal, for example booking a flight in an automated travel agent system; the system state is described in terms of how close the agent is to being able to book the flight.

Such systems suffer from three principal problems. Firstly, most existing dialog systems are built for a single purpose over a single task domain, for example retrieving e-mail or making travel arrangements (Levin, Pieraccini, & Eckert 1998). While it is not impossible to extend these systems to multiple domains, the interaction between different task domains is difficult to model. Secondly, system-initiated strategies perform best with these approaches, however mixed-initiative strategies are a more natural model for human-robot interaction. Finally, most existing dialog systems do not model confidences on the human utterances, and therefore do not account for the reliability of utterances while performing a strategy. Some systems do use the log-likelihood values for speech utterances, however, these values are only thresholded as to whether or not the utterance needed confirmation or not (Niimi & Kobayashi 1996; Singh *et al.* 1999).

The real problem that lies at the heart of these three issues is that of observability. The ultimate goal of a dialog system is to satisfy a user request; however, what the user wants is only partially observable at best. A conventional MDP demands to know the current state at all times, therefore the state has to be wholly contained in the system. System-initiated dialogs are successful because there is no uncertainty in determining when the user wants something, and a single task-domain system works well, because the relevant goals and actions are easily specified.

In this paper, we invert the conventional notion of state in a dialog. The world is viewed as partially unobservable – the underlying state is the intention of the user with respect to the dialog. The only observations about the user's state are the speech utterances given by the speech recognition system. By accepting the partial observability of the world, the dialog problem becomes one that is addressed by Partially Observable Markov Decision Processes (POMDPs).

There are several POMDP algorithms that are the natural choice for policy generation (Kaelbling, Littman, & Cassandra 1998; Cassandra, Littman, & Zhang 1997). However, solving real world dialog scenarios is computationally intractable for full-blown POMDP solvers, as the complexity is doubly exponential in the number of states. We therefore will use an algorithm for finding approximate solutions to POMDP-style problems and apply it to dialog manage-

ment. This algorithm, the Augmented MDP, was developed for mobile robot navigation in the context of Coastal Navigation (Roy & Thrun 1999), and operates by augmenting the state description with a compression of the current belief state. By representing the belief state succinctly with its entropy, belief-space planning can be approximated without the expected complexity.

In the first section of this paper, we develop the model of dialog interaction, including the effects of speech recognition and speech production on the state representation. This model allows for a more natural description of dialog problems, and in particular allows for intuitive handling of noisy and ambiguous dialogs. Few existing dialogs can handle ambiguous input, typically relying on natural language processing to resolve semantic ambiguities (Aust & Ney 1998). Secondly, we describe the Augmented MDP method used to find the optimal policy for a given dialog system. While this algorithm was initially developed for mobile robot navigation, the dialog management problem has several features that require modification to the original algorithm. We conclude with a description of an example problem domain and compare the performance of the Augmented MDP to conventional MDP and POMDP dialog strategies.

The application that we are developing this dialog model for is on a mobile robot with knowledge from several domains, and interacting with many people over time. While there habe been some development of speech recognition and understanding on robots (Torrance 1994; Asoh, Hara, & Matsui 1998), most research has been restricted to the desktop or telephone domains. The reliability of the audio signal on a mobile robot, coupled with the expectations of natural interaction that people have with more anthropomorphic robots increases the demands placed on the dialog manager.

Dialog Systems and POMDPs

A Partially Observable Markov Decision Process (POMDP) is a natural way of modelling dialog processes, especially when the state of the system is viewed as the state of the user. The partial observability capabilities of a POMDP policy allows the dialog planner to recover from noisy or ambiguous utterances in a natural and autonomous way. At no time does the machine interpreter have any direct knowledge of the state of the user, i.e, what the user wants. The machine interpreter can only infer this state from the (noisy) speech of the user. The POMDP framework provides exactly the right mechanism for modelling uncertainty about what the user is trying to accomplish.

The POMDP consists of an underlying, unobservable Markov Decision Process. The MDP is specified by:

- a set of states $\mathcal{S} \epsilon \{s_1, s_2, \ldots_n\}$
- a set of actions $\mathcal{A} \epsilon \{a_1, a_2, \ldots, a_m\}$
- a set of transition probabilities $T(s', a, s) = P(s'|s, a)$
- a set of rewards $R : \mathcal{S} \times \mathcal{A} \mapsto \Re$
- an initial state s_o

The transition probabilities form a structure over the set of states, connecting the states in a directed graph with arcs between states with non-zero transition probabilities.

The POMDP adds:

- a set of observations $\mathcal{O} \epsilon \{o_1, o_2, \ldots, o_l\}$
- a set of observation probabilities $O(o, s, a) = P(o \mid s, a)$

and replaces

- the initial state s_o with an initial belief, $P(s_o : s_o \epsilon \mathcal{S})$
- the set of rewards with rewards conditioned on observations as well: $R : \mathcal{S} \times \mathcal{A} \times \mathcal{O} \mapsto \Re$

The policy is one that generates an action a based on the current belief $P(s : s_o \epsilon \mathcal{S})$. The optimal strategy for a POMDP is one that maximizes the expected reward (either to a finite or infinite horizon, possibly with discounting). Unfortunately, POMDPs for all but the most trivial problems are computationally intractable.

We can, however, simplify the problem by noticing that the uncertainty, or belief state of the system, tends to have a certain structure. The uncertainty that the system has is usually domain-specific. For example, it may be likely that an office-delivery system can mis-hear the name of a package recipient, but it is unlikely that the system will confuse the name of the mail recipient for a request to get coffee. In this manner, the beliefs that the system instantiating the POMDP is likely to have are typically well-localised and often uni-modal. We can strengthen the uni-modal assumption by imposing a topology on the model that connects states that are easily confused.

By making the uni-modal, localised assumption about the uncertainty, it becomes possible to summarise the belief state as its most likely state, and the entropy of the belief state. The important point to note is that the entropy of the belief state approximates a sufficient statistic [1]. Given this assumption, we can do MDP style planning with this representation.

The Augmented MDP

We represent the belief state $\mathbf{x}$ of the system as an ordered pair, where s is the most likely state in the MDP, and H is the entropy of the current belief.

$$\mathbf{x} = (argmax_s P(s); H(P(s)) \tag{1}$$

The second half of the state pair $(s; H(P(s)))$ is the entropy of the probability distribution over the states. This is computed using:

$$H(P(s)) = -\sum_s P(s) \log P(s) \tag{2}$$

During planning, we can model the effect of actions and observations by reconstructing the full belief $P(s)$ from the state $\mathbf{x}$ (under the uni-modal assumption above), apply the actionse and observations, and recompress the full belief back down to its state representation $\mathbf{x}'$.

[1] Although sufficient statistics are usually moments of continuous distributions, our experience has shown that the entropy serves equally well.

Modelling Actions and Observations

We model the effect of an action a (in the case of dialog management, taking actions usually corresponds to saying something to the user) as

$$P(s'|a) = \sum_{\mathcal{S}} P(s'|a,s) \cdot P(s) \quad (3)$$

$$\Rightarrow \mathbf{x}' = (\operatorname*{argmax}_s P(s'|a); H(P(s'|a)) \quad (4)$$

where $P(s)$ is the prior distribution over the states $\mathcal{S}$ before action a, and $P(s'|a)$ is the posterior. We summarize the posterior as $\mathbf{x}'$, computed from prior $\mathbf{x} = (P(s); H(P(s))$ and action a.

We compute the effect of receiving an utterance o:

$$P(s'|o) = \alpha P(o|s) \cdot P(s) \quad (5)$$

$$\Rightarrow \mathbf{x}' = (P(s'|o); H(P(s'|o))) \quad (6)$$

where $P(s'|s,o)$ is the posterior of the state distribution after the observation o is made and $P(o|s')$ is the probability of an utterance o given a state s, given by the POMDP model of the dialog system. α is simply a normalizer to ensure the probabilities sum to 1.

In order to generate the optimal policy using the Augmented MDP framework, two modifications have to be made to the model specification. One is to maximise the reward for state-action-observation triplets with minimum belief entropy. We modify the reward as following:

$$R(\mathbf{x},a,o) = R(s,a,o,H(P(s))) = \frac{R(s,a,o)}{(H(P(s))+1)} \quad (7)$$

Ideally, we would maximise the expected reward for the belief state $\mathbf{x}$, but the simplification provided by the approximation in equation 7 provides similar behaviour, in that the goal state is achieved with minimal entropy in the belief.

A second modification is to add the effect of actions and observations on the entropy of the belief state. While taking an action in a dialog system does not immediately change the entropy of the belief state (saying something does not make the system more or less sure about the user's intentions), the subsequent observation will change the entropy. It is impossible to predict deterministically which observation will be received after an action, so the posterior entropy after an action is a weighted sum of the observations:

$$H(P(s'|s,a)) = \sum_{\mathcal{O}} p(o|s',a) H(P(s''|s',o) \cdot P(s'|s,a)) \quad (8)$$

These posterior entropies were precomputed for all possible entropy levels before value iteration. By pre-computing the entropy levels in this way, it becomes possible to identify which actions have the most utility in reducing the entropy, for any given state-entropy pair.

Policy Generation To generate the optimal policy, we perform undiscounted, infinite-horizon value iteration, using the Bellman equations:

$$J(\mathbf{x}_i) = \max_a [R(\mathbf{x}_i) + \sum_{j=1}^{N} p(\mathbf{x}_j|\mathbf{x}_i,a) \cdot J(\mathbf{x}_j)] \quad (9)$$

$$\pi(\mathbf{x}_i) = \operatorname*{argmax}_a [R(\mathbf{x}_i) + \sum_{j=1}^{N} p(\mathbf{x}_j|\mathbf{x}_i,a) \cdot J(\mathbf{x}_j)] \quad (10)$$

Equation 9 computes the value for each state $\mathbf{x}$. The value $J(\mathbf{x},a)$ is defined as the immediate reward, $R\mathbf{x}_i$, plus the expected reward from the future states j achieved by taking action a, weighted by the transition probabilities $p(j|i,a)$. The value $J(\mathbf{x})$ is taken to be the maximum $J(\mathbf{x},a)$ over all a. Similarly, equation 10 gives the policy for state$\mathbf{x}$, and is the action a which maximises $J(\mathbf{x},a)$. The values are updated iteratively until the value for each states converges to a stable value, and then the policy is extracted. The value iteration occurs over all state-entropy pairs, with the entropy chosen empirically as 20 entropy levels in the range $[0, 2.0]$.

Dialog Management Specifics

For the purposes of dialog management, a number of modifications are made. In particular, the observation probabilities $P(o|s,a)$ given by the model are typically artificial and only used for policy generation. During the actual policy execution, the probabilities for the observations are non-static, and are given by the speech recognition system. In this way we can adjust the belief state "on-the-fly" depending on how certain the speech recognition system is of any particular utterance. The effect of *ambiguous* utterances remains unchanged from policy generation to policy execution.

Secondly, the reward modification as given in equation 7 was only carried out for state-action-observation triplets that led into the terminal (`request_done`) state; this has the effect of allowing arbitrary uncertainty (entropy) along the path, but minimising the uncertainty of maximising the reward along the goal. This modification is specific to the kinds of dialogs we are expecting and was implemented to minimise potential unnecessary entropy-reductions along the path from start to goal.

The Example Domain

The system that was used throughout these experiments is based on a mobile robot, Florence Nightingale (Flo), developed as a prototype nursing home assistant. Flo uses the Sphinx II speech recognition system (Ravishankar 1996), and the Festival speech synthesis system (Black, Taylor, & Caley 1999). Figure 1 shows a picture of the robot.

Since the robot is a nursing home assistant, we use task domains that are relevant to everyday life. Table 1 shows a list of the task domains the user can ask about: the time, the weather, what is on different TV stations, and can also ask to deliver a video message (called *facemail*) to another person. These abilities have all been implemented on Flo,

Figure 1: *Florence Nightingale*, the prototype nursing home robot used in these experiments.

and the information about the weather and TV schedules is downloaded on request from the web.

Time
Weather (Current and for Tomorrow)
TV Schedules for different channels (ABC, NBC, CBS)
Delivering video messages (facemail)

Table 1: The task domains for Flo.

If we translate these tasks into the MDP framework that we have described, the decision problem has 17 states, and the state transition graph is given in figure 2. The different tasks have varying levels of complexity, from simply saying the time, to delivering facemail, in which the user moves through four different states before the task is completed.

The Model

For the planning stage, the observation model was as follows [2]: each state emits an observation probability that corresponds to the state name. The observation is the correct state with 60% probability, and is an incorrect state name with 2.5% probability for each of the 16 incorrect states. There are 22 possible actions, such as `say_time` or `ask_which_station`. The transitions probabilities $P(s'|s,u)$ are largely sparse and deterministic. Finally, the reward is structured such that taking an incorrect action (such as `say_hello` when no greeting has been given, or `deliver_mail` when no mail has been given) had a reward of -10, but the reward for taking the correct action is +20. Some special cases exist: the reward for transitioning to the `request_done` state is +200, the reward for asking a question or confirming a response is -1, and the reward for delivering a message to the wrong person is -200.

The reason for the additional negative reward in delivering to the wrong person is to highlight the difference between a costly action that is merely irritating (giving an inappropriate response) and an action that can be much more serious (having the robot leave the room at the wrong time, or travel to the wrong destination).

[2] Recall that during policy execution, the observation probability is given by the speech recognition system itself.

An Example Dialog

Table 2 shows an example dialog from the Augmented MDP system. The left hand columns are the emitted observation and the entropy of the system that resulted. The most interesting lines are the fourth and fifth lines of the dialog: although the true state of the user is `want_facemail`, the observation made is that the user is in state `want_time`. The conventional MDP would take the observation as given, but the observation from the speech recognition system was highly uncertain, so the entropy of the belief becomes quite high (20). The entropy stays high before getting a more certain observation. The system confirms the desired state, before moving on. We see a similar action a little later; here, because the cost of mis-delivering a facemail is very high, the system takes confirmation action even when the entropy is relatively low.

Experimental Results

We compared the performance of the three algorithms (conventional MDP, Augmented MDP and full POMDP) over the example domain. The metric used was to look at the total reward accumulated over the course of an extended test. In order to perform this full test, the observations and states from the underlying MDP were generated stochastically from the model and then given to the policy. The action taken by the policy was returned to the model, and the policy rewarded based on the state-action-observation triplet. The experiments were run for a total of 100 dialogs, where each dialog is considered to be a sequence of observation-action utterances from the start state `no_request` to the goal state `request_done`. Once the final state was reached, the system transitioned back to the start state deterministically.

The Restricted State Space Problem

The POMDP policy was generated using the Incremental Improvement algorithm (Cassandra, Littman, & Zhang 1997), using code provided by Tony Cassandra. The solver was unable to complete a solution for the full state space, so we used a restricted problem, with only 7 states, and 2 task domains from table 1: the current time and the current weather.

Figure 3 shows the performance of the three algorithms, over the course of 100 dialogs. Notice that the POMDP strategy outperformed both the conventional and Augmented MDP; it accumulated the most reward, and fastest. This good performance of the POMDP is not suprising, but time to compute this strategy is high – 729 secs. In terms of time to execute the strategies, although the Augmented MDP eventually accumulates more reward than the conventional planner, it takes longer to do so, because it is delayed by low-cost confirming actions.

In order to get a fairer picture of the three algorithms, the time was normalized by the length of each dialog iteration (a sequence from `no_request` to `request_done`). By

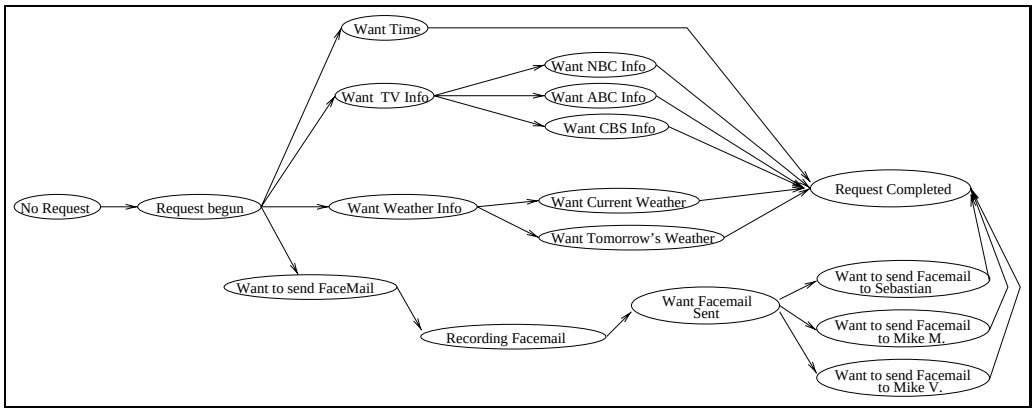

Figure 2: A graph of the basic Markov Decision Process underlying the dialog manager.

Observation	Belief Entropy	Action	Reward	True State
<silence>	0	do_nothing	20	none
flo hello	2	ask_repeat	-1	request_begun
flo hello	0	say_hello	20	request_begun
was on a	20	ask_repeat	-1	want_facemail
flo time is	20	ask_repeat	-1	want_facemail
flo i want to deliver please	6	confirm_facemail	-1	want_facemail
flo i want to deliver facemail	0	say_begin_recording	20	want_facemail
<recording_facemail>	0	record_facemail	20	recording_facemail
flo send it	1	ask_repeat	-1	want_facemail
flo sent it	0	ask_which_facemail	20	want_facemail
flo i want to send it to mike	3	ask_repeat	-1	want_facemail_mike_m
flo i want to send it to mike m	0	deliver_facemail_mike_m	200	want_facemail_mike_m
thanks flo goodbye	0	do_nothing	20	request_done

Table 2: An example dialog, from the Augmented MDP.

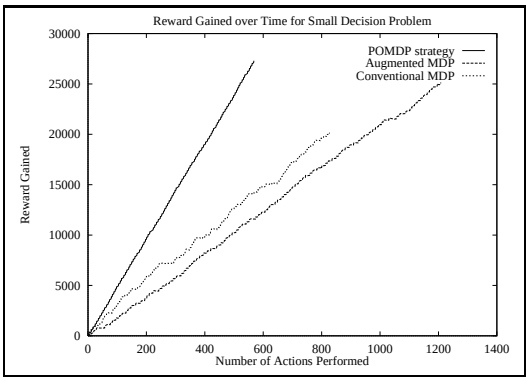

Figure 3: A comparison of the reward gained over time for the Augmented MDP vs. the conventional MDP vs POMDP for the 7 state problem. The time is measured in number of actions.

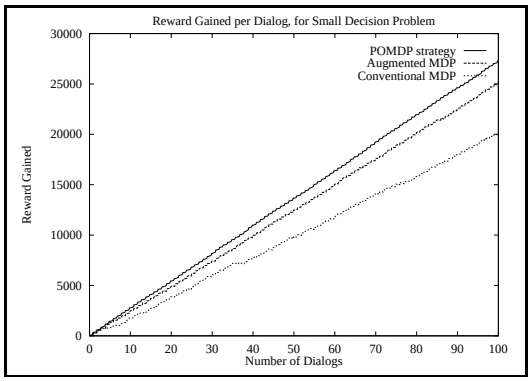

Figure 4: A comparison of the reward gained over time for the Augmented MDP vs. the conventional MDP for the 7 state problem. In this case, the time is measured in dialogs, or iterations of satisfying user requests.

disregarding how *long* it takes each strategy to fulfill a user request, the Augmented MDP is no longer penalized as heavily for too many confirming actions.

Figure 4 shows the performance of the three algorithms, as a function of time measured in dialogs.

The POMDP strategy still outperforms the other two, but the ability of the Augmented MDP is more evident here; the Augmented MDP is accumulating reward at a rate that is closer to the POMDP strategy than the conventional MDP. The sub-optimality of the Augmented MDP compared to the POMDP algorithm is shown in that it is too conservative – the POMDP algorithm does not spend as much time confirming responses.

The Full State Space Problem

Figure 5 demonstrates the algorithms on the full dialog model as given in figure 2. Because of the number of states, no POMDP solution could be computed for this problem.

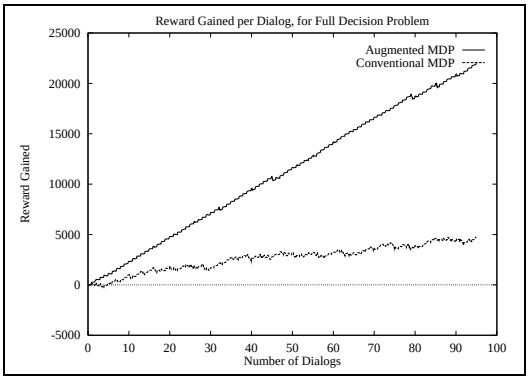

Figure 5: A comparison of the reward gained over time for the Augmented MDP vs. the conventional MDP for the 17 state problem. Again, the time is measured in number of actions.

The Augmented MDP clearly outperforms the conventional MDP strategy, as it more than triples the total accumulated reward over the lifetime of the strategies, although at the cost of taking longer to reach the goal state in each dialog. Table 3 breaks down the numbers in more detail. The average reward for the Augmented MDP is 18.6 per action, which is the maximum reward for most actions, suggesting that the Augmented MDP is taking the right action about 95% of the time. Furthermore, the average reward per dialog for the Augmented MDP is 230 compared to 49.7 for the conventional MDP, which suggests that the conventional MDP is making a large number of mistakes in each dialog.

Finally, the standard deviation for the Augmented MDP is much narrower, suggesting that this algorithm is getting its rewards much more consistently than the conventional MDP.

Augmented MDP	
Average Reward	18.6 +/- 57.1
Average Dialog Reward	230.7 +/- 77.4
Conventional MDP	
Average Reward	3.8 +/- 67.2
Average Dialog Reward	49.7 +/- 193.7

Table 3: A comparison of the rewards accumulated for the two algorithms using the full model.

Figure 6 demonstrates the speed for the Augmented MDP algorithm. The graph shows the time to compute the optimal policy as a function of state space size. The number of possible actions and observations also changes with the state space size and is roughly equal to it. As the graph depicts, the conventional MDP solves the problems almost instantaneously. However, the POMDP takes orders and orders of magnitude longer. After 24 hours of computation on a 400MHz Pentium II, only 2 episodes of value iteration had occurred. While the Augmented MDP did not have the near-instant solution time for a policy, it only took 8 seconds for the full 17 state policy, and figure 6 suggests that the algorithm may scale well to larger domains.

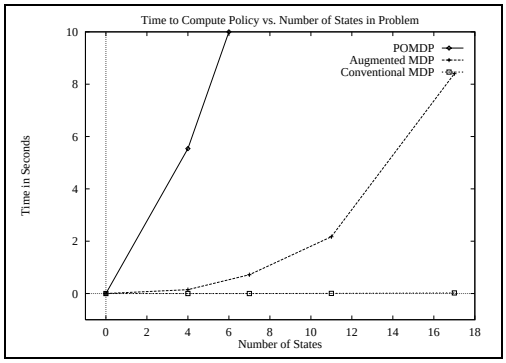

Figure 6: A comparison of times to find the policy, for the conventional MDP, Augmented MDP and POMDP solvers. The POMDP was unable to find a solution for the 7 and 11 state problems.

Conclusion

This paper discusses a novel way to view the dialog management problem. The domain is represented as the partially observable state of the user, where the observations are speech utterances from the user. This formulation clearly requires a POMDP-style solution, however, finding a POMDP for any but the most trivial dialog problem is not computationally feasible. Therefore, we use instead of a full belief state an augmented MDP state representation for approximating the optimal policy. This representation consists of the maximum-likelihood state augmented with a compression of the belief state represented by the entropy of the belief. This approach, the Augmented MDP, was developed for mobile robots in the context of Coastal Navigation. The Augmented MDP finds a solution that quantitatively outperforms the conventional MDP, while dramatically reducing the time to solution compared to a full POMDP (linear vs. exponential in the number of states). The Augmented MDP allows mixed-initiative and user-initiative dialogs, and handles noisy observations from the user appropriately. Particularly, ambiguous observations are also handled without full-blown natural language processing. The policies generated by the Augmented MDP are sub-optimal compared to the POMDP in terms of reward gained per action, however, the Augmented MDP substantially outperforms conventional MDPs without the computational cost associated with POMDP solutions.

While the results of the Augmented MDP approach to the dialog system are promising, a number of improvements are needed. The Augmented MDP is overly cautious, refusing to commit to a particular course of action until it is completely certain. This could be avoided by having some non-linear reward structure, where achieving the goal with some arbitrarily low entropy is equivalent to achieving the

goal with zero entropy.

Secondly, the compression of the belief state as the single entropy statistic is overly optimistic about the kinds of belief states that are likely to be encountered. In particular, it very poorly models ambiguities between two distinct task domains, as compared with ambiguities between two states in a single domain. The right way to compress the belief state is still an open question; multiple statistics are almost certainly needed, but which and how many is not clear.

Another problem is that the policy is relatively sensitive to the parameters of the model. At the moment, the model parameters are set by hand. While learning the parameters from scratch for a full POMDP is probably unnecessary, automatic tuning of the model parameters would definitely add to the utility of the model. Furthermore, the mechanism for computing the changes in entropy as a result of observations could also be learned. By generating synthetic speech data, it should be possible to learn the effects of different observations on the entropy, and dispense with a number of assumptions about the prior belief state.

Acknowledgements

The authors would like to thank Tom Mitchell for his advice and support of this research.

Kevin Lenzo and Mathur Ravishankar made our use of Sphinx possible, answered requests for information and made bug fixes willingly. Tony Cassandra was extremely helpful in distributing his POMDP code to us, and answering promptly any questions we had. The assistance of the Nursebot team is also gratefully acknowledged, including the members from the School of Nursing and the Department of Computer Science Intelligent Systems at the University of Pittsburgh.

This research was supported in part by Le Fonds pour la Formation de Chercheurs et l'Aide à la Recherche (Fonds FCAR).

References

Asoh, H.; Hara, I.; and Matsui, T. 1998. Structured dynamic multi-agent architecture for controlling mobile office-conversant robot. In *Proc. IEEE ICRA*, volume 1.

Aust, H., and Ney, H. 1998. Evaluating dialog systems used in the real world. In *Proc. IEEE ICASSP*, volume 2, 1053–1056.

Black, A.; Taylor, P.; and Caley, R. 1999. *The Festival Speech Synthesis System*, 1.4 edition.

Cassandra, A.; Littman, M. L.; and Zhang, N. L. 1997. Incremental pruning: A simple, fast, exact algorithm for partially observable Markov decision processes. In *Proc. 13th Ann. Conf. on Uncertainty in Artificial Intelligence (UAI–97)*, 54–61.

Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. R. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101:99–134.

Levin, E.; Pieraccini, R.; and Eckert, W. 1998. Using Markov decision process for learning dialogue strategies. In *Proc. ICASSP*.

Niimi, Y., and Kobayashi, Y. 1996. Dialog control strategy based on the reliability of speech recognition. In *Proc. ICSLP*.

Ravishankar, M. 1996. *Efficient Algorithms for Speech Recognition*. Ph.D. Dissertation, Carnegie Mellon.

Roy, N., and Thrun, S. 1999. Coastal navigation with mobile robots. In *NIPS*, volume 11.

Singh, S.; Kearns, M.; Litman, D.; and Walker, M. 1999. Reinforcement learning for spoken dialog systems. In *NIPS*.

Torrance, M. C. 1994. Natural communication with robots. Master's thesis, MIT Dept of E.E.and C.S.

Young, S. 1990. Use of dialogue, pragmatics and semantics to enhance speech recognition. *Speech Communication* 9(5-6).