Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics
(ACL 2000). Hong Kong, Oct. 2000

Spoken Dialogue Management Using Probabilistic Reasoning

Nicholas Roy and Joelle Pineau and Sebastian Thrun
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
{*nicholas.roy*|*joelle.pineau*|*sebastian.thrun*}@*cs.cmu.edu*

Abstract

Spoken dialogue managers have benefited from using stochastic planners such as Markov Decision Processes (MDPs). However, so far, MDPs do not handle well noisy and ambiguous speech utterances. We use a Partially Observable Markov Decision Process (POMDP)-style approach to generate dialogue strategies by inverting the notion of dialogue state; the state represents the *user*'s intentions, rather than the system state. We demonstrate that under the same noisy conditions, a POMDP dialogue manager makes fewer mistakes than an MDP dialogue manager. Furthermore, as the quality of speech recognition degrades, the POMDP dialogue manager automatically adjusts the policy.

1 Introduction

The development of automatic speech recognition has made possible more natural human-computer interaction. 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 must also be speaker-independent. Regardless of these difficulties, any system that manages human-machine dialogues must be able to perform reliably even with noisy and stochastic speech input.

Recent research in dialogue management has shown that Markov Decision Processes (MDPs) can be useful for generating effective dialogue strategies (Young, 1990; Levin et al., 1998); the system is modelled as a set of states that represent the dialogue as a whole, and a set of actions corresponding to speech productions from the system. The goal is to maximise the reward obtained for fulfilling a user's request. However, the correct way to represent the state of the dialogue is still an open problem (Singh et al., 1999). A common solution is to restrict the system to a single goal. For example, in 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 a principal problem. A conventional MDP-based dialogue manager must know the current state of the system at all times, and therefore the state has to be wholly contained in the system representation. These systems perform well under certain conditions, but not all. For example, MDPs have been used successfully for such tasks as retrieving e-mail or making travel arrangements (Walker et al., 1998; Levin et al., 1998) over the phone, task domains that are generally low in both noise and ambiguity. However, the issue of reliability in the face of noise is a major concern for our application. Our dialogue manager was developed for a mobile robot application that has knowledge from several domains, and must interact with many people over time. For speaker-independent systems and systems that must act in a noisy environment, the user's action and intentions cannot always be used to infer the dialogue state; it may be not be possible to reliably and completely determine the state of the dialogue following each utterance. The poor reliability of the audio signal on a mobile robot, coupled with the expectations of natural interaction that people have with more anthropomorphic interfaces, increases the demands placed on the dialogue manager.

Most existing dialogue systems do not model confidences on recognition accuracy of the human utterances, and therefore do not account for the reliability of speech recognition when applying a dialogue strategy. Some systems do use the log-likelihood values for speech utterances, however these values are only thresholded to indicate whether the utterance needs to be confirmed (Niimi and Kobayashi, 1996; Singh et al., 1999). An important concept lying at the heart of this issue is that of observability – the ultimate goal of a dialogue system is to satisfy a user request; however, what the user really wants is at best partially observable.

We handle the problem of partial observability by inverting the conventional notion of state in a dialogue. The world is viewed as partially unobservable – the underlying state is the intention of the user with respect to the dialogue task. The only observations about the user's state are the speech utterances given by the speech recognition system, from which some knowledge about the current state can be inferred. By accepting the partial observability of the world, the dialogue problem becomes one that is addressed by Partially Observable Markov Decision Processes (POMDPs) (Sondik, 1971). Finding an optimal policy for a given POMDP model corresponds to defining an optimal dialogue strategy. Optimality is attained within the context of a set of rewards that define the relative value of taking various actions.

We will show that conventional MDP solutions are insufficient, and that a more robust methodology is required. Note that in the limit of perfect sensing, the POMDP policy will be equivalent to an MDP policy. What the POMDP policy offers is an ability to compensate appropriately for better or worse sensing. As the speech recognition degrades, the POMDP policy acquires reward more slowly, but makes fewer mistakes and blind guesses compared to a conventional MDP policy.

There are several POMDP algorithms that may be the natural choice for policy generation (Sondik, 1971; Monahan, 1982; Parr and Russell, 1995; Cassandra et al., 1997; Kaelbling et al., 1998; Thrun, 1999). However, solving real world dialogue 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 dialogue management. This algorithm, the Augmented MDP, was developed for mobile robot navigation (Roy and 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 dialogue interaction. This model allows for a more natural description of dialogue problems, and in particular allows for intuitive handling of noisy and ambiguous dialogues. Few existing dialogues can handle ambiguous input, typically relying on natural language processing to resolve semantic ambiguities (Aust and Ney, 1998). Secondly, we present a description of an example problem domain, and finally we present experimental results comparing the performance of the POMDP (approximated by the Augmented MDP) to conventional MDP dialogue strategies.

2 Dialogue Systems and POMDPs

A Partially Observable Markov Decision Process (POMDP) is a natural way of modelling dialogue 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 dialogue 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 user's noisy input. The POMDP framework provides a principled 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 actions represent the set of responses that the system can carry out. 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 rewards define the relative value of accomplishing certain actions when in certain states.

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 observations consist of a set of keywords which are extracted from the speech utterances. The POMDP plans in belief space; each belief consists of a probability distribution over the set of states, representing the respective probability that the user is in each of these states. The initial belief specified in the model is updated every time the system receives a new observation from the user.

The POMDP model, as defined above, first goes through a planning phase, during which it finds an optimal strategy, or policy, which describes an optimal mapping of action a to belief $P(s : s_o \epsilon \mathcal{S})$, for all possible beliefs. The dialogue manager uses this policy to direct its behaviour during conversations with users. The optimal strategy for a POMDP is one that prescribes action selection that maximises the expected reward. Unfortunately, finding an optimal policy exactly for all but the most trivial POMDP problems is computationally intractable. A near-optimal policy can be computed significantly faster than an exact one, at the expense of a slight reduction in performance. This is often done by imposing restrictions on the policies that can be selected, or by simplifying the belief state and solving for a simplified uncertainty representation.

In the Augmented MDP approach, the POMDP problem is simplified by noticing that the belief state of the system tends to have a certain structure. The uncertainty that the system has is usually domain-specific and localised. For example, it may be likely that a household robot system can confuse TV channels ('ABC' for 'NBC'), but it is unlikely that the system will confuse a TV channel request for a request to get coffee. By making the localised assumption about the uncertainty, it becomes possible to summarise any given belief vector by a pair consisting of the most likely state, and the entropy of the belief state.

$$p(\mathbf{s}) \cong < \operatorname*{argmax}_{\mathbf{s}} p(\mathbf{s}); H(p(\mathbf{s})) > \quad (1)$$

$$H(p(\mathbf{s})) = -\sum_{i=1}^{N} p(\mathbf{s}) \log_2 p(\mathbf{s}) \quad (2)$$

The entropy of the belief state approximates a sufficient statistic for the entire belief state [1]. Given this assumption, we can plan a policy for every possible such {*state, entropy*} pair, that approximates the POMDP policy for the corresponding belief $p(\mathbf{s})$.

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

3 The Example Domain

The system that was used throughout these experiments is based on a mobile robot, Florence

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

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 et al., 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 assisted living in a home environment. Table 1 shows a list of the task domains the user can inquire about (the time, the patient's medication schedule, what is on different TV stations), in addition to a list of robot motion commands. These abilities have all been implemented on Flo. The medication schedule is pre-programmed, the information about the TV schedules is downloaded on request from the web, and the motion commands correspond to pre-selected robot navigation sequences.

Time
Medication (Medication 1, Medication 2, ..., Medication n)
TV Schedules for different channels (ABC, NBC, CBS)
Robot Motion Commands (To the kitchen, To the Bedroom)

Table 1: The task domains for Flo.

If we translate these tasks into the framework that we have described, the decision problem has 13 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 going through a list of medications. For simplicity, only the maximum-likelihood transitions are shown in Figure 2. Note that this model is handcrafted. There is ongoing research into learning policies automatically using reinforcement learning (Singh et al., 1999); dialogue models could be learned in a similar manner. This example model is simply to illustrate the utility of the POMDP approach.

There are 20 different actions; 10 actions correspond to different abilities of the robot such as going to the kitchen, or giving the time. The remaining 10 actions are clarification or confirmation actions, such as re-confirming the desired TV channel. There are 16 observations that correspond to relevant keywords as well as a nonsense observation. The reward structure gives the most reward for choosing actions that satisfy the user request. These actions then lead back to the beginning state. Most other actions are penalised with an equivalent negative amount. However, the confirmation/clarification actions are penalised lightly (values close to 0), and the motion commands are penalised heavily if taken from the wrong state, to illustrate the difference between an undesirable action that is merely irritating (i.e., giving an inappropriate response) and an action that can be much more costly (e.g., having the robot leave the room at the wrong time, or travel to the wrong destination).

3.1 An Example Dialogue

Table 2 shows an example dialogue obtained by having an actual user interact with the system on the robot. The left-most column is the emitted observation from the speech recognition system. The operating conditions of the system are fairly poor, since the microphone is on-board the robot and subject to background noise as well as being located some distance from the user. In the final two lines of the script, the robot chooses the correct action after some confirmation questions, despite the fact that the signal from the speech recogniser is both very noisy and also ambiguous, containing cues both for the "say_hello" response and for robot motion to the kitchen.

4 Experimental Results

We compared the performance of the three algorithms (conventional MDP, POMDP approximated by the Augmented MDP, and exact 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 was rewarded based on the state-action-observation triplet. The experiments were run for a total of 100 dialogues, where each dialogue is considered to be a cycle of observation-action utterances from the start state `request_begun` through a sequence of states and back to the start state. The time was normalised by the length of each dialogue cycle.

4.1 The Restricted State Space Problem

The exact POMDP policy was generated using the Incremental Improvement algorithm (Cassan-

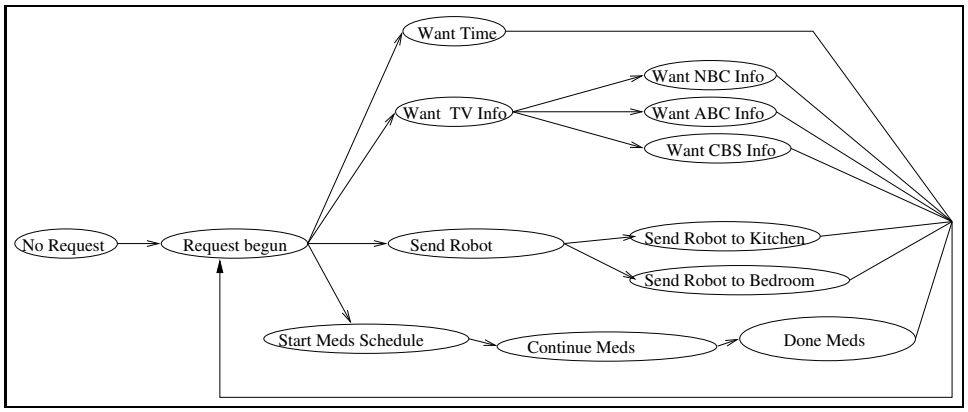

Figure 2: A simplified graph of the basic Markov Decision Process underlying the dialogue manager. Only the maximum-likelihood transitions are shown.

Observation	True State	Belief Entropy	Action	Reward
flo hello	request_begun	0.406	say_hello	100
flo what is like	start_meds	2.735	ask_repeat	-100
flo what time is it for will the	want_time	0.490	say_time	100
flo was on abc	want_tv	1.176	ask_which_station	-1
flo was on abc	want_abc	0.886	say_abc	100
flo what is on nbc	want_nbc	1.375	confirm_channel_nbc	-1
flo yes	want_nbc	0.062	say_nbc	100
flo go to the that pretty good what	send_robot	0.864	ask_robot_where	-1
flo that that hello be	send_robot_bedroom	1.839	confirm_robot_place	-1
flo the bedroom any i	send_robot_bedroom	0.194	go_to_bedroom	100
flo go it eight a hello	send_robot	1.110	ask_robot_where	-1
flo the kitchen hello	send_robot_kitchen	1.184	go_to_kitchen	100

Table 2: An example dialogue. Note that the robot chooses the correct action in the final two exchanges, even though the utterance is both noisy and ambiguous.

dra et al., 1997). The solver was unable to complete a solution for the full state space, so we created a much smaller dialogue model, with only 7 states and 2 task domains: time and weather information.

Figure 3 shows the performance of the three algorithms, over the course of 100 dialogues. Notice that the exact POMDP strategy outperformed both the conventional MDP and approximate POMDP; it accumulated the most reward, and did so with the fastest rate of accumulation. The good performance of the exact POMDP is not surprising because it is an optimal solution for this problem, but time to compute this strategy is high: 729 secs, compared with 1.6 msec for the MDP and 719 msec for the Augmented MDP.

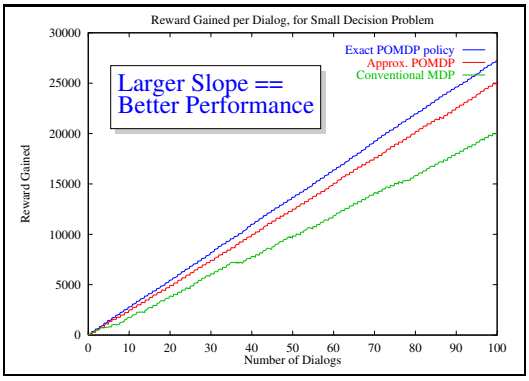

Figure 3: A comparison of the reward gained over time for the exact POMDP, POMDP approximated by the Augmented MDP, and the conventional MDP for the 7 state problem. In this case, the time is measured in dialogues, or iterations of satisfying user requests.

4.2 The Full State Space Problem

Figure 4 demonstrates the algorithms on the full dialogue model as given in Figure 2. Because of the number of states, no exact POMDP solution could be computed for this problem; the POMDP policy is restricted to the approximate solution.

The POMDP solution 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 dialogue.

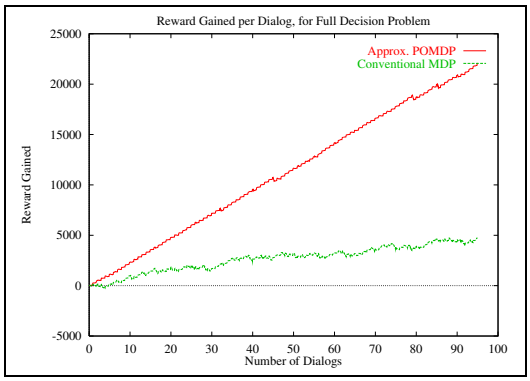

Figure 4: A comparison of the reward gained over time for the approximate POMDP vs. the conventional MDP for the 13 state problem. Again, the time is measured in number of actions.

Table 3 breaks down the numbers in more detail. The average reward for the POMDP is 18.6 per action, which is the maximum reward for most actions, suggesting that the POMDP is taking the right action about 95% of the time. Furthermore, the average reward per dialogue for the POMDP 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 dialogue.

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

4.3 Verification of Models on Users

We verified the utility of the POMDP approach by testing the approximating model on human users. The user testing of the robot is still preliminary, and therefore the experiment presented here cannot be considered a rigorous demonstration. However, Table 4 shows some promising results. Again, the POMDP policy is the one provided by the approximating Augmented MDP.

The experiment consisted of having users interact with the mobile robot under a variety of conditions. The users tested both the POMDP and an implementation of a conventional MDP dialogue manager. Both planners used exactly the same model. The users were presented first with one manager, and then the other, although they were not told which manager was first and the order varied from user to user randomly. The user labelled each action from the system as "Correct" (+100 reward), "OK" (-1 reward) or "Wrong" (-100 reward). The "OK" label was used for responses by the robot that were questions (i.e., did not satisfy the user request) but were relevant to the request, e.g., a confirmation of TV channel when a TV channel was requested.

The system performed differently for the three test subjects, compensating for the speech recognition accuracy which varied significantly between them. In user #2's case, the POMDP manager took longer to satisfy the requests, but in general gained more reward per action. This is because the speech recognition system generally had lower word-accuracy for this user, either because the user had unusual speech patterns, or because the acoustic signal was corrupted by background noise.

By comparison, user #3's results show that in the limit of good sensing, the POMDP policy approaches the MDP policy. This user had a much higher recognition rate from the speech recogniser, and consequently both the POMDP and conventional MDP acquire rewards at equivalent rates, and satisfied requests at similar rates.

5 Conclusion

This paper discusses a novel way to view the dialogue management problem. The domain is represented as the partially observable state of the user, where the observations are speech utterances from the user. The POMDP representation inverts the traditional notion of state in dialogue management, treating the state as unknown, but inferrable from the sequences of observations from the user. Our approach allows us to model observations from the user probabilistically, and in particular we can compensate appropriately for more or less reliable observations from the speech recognition system. In the limit of perfect recognition, we achieve the same performance as a conventional MDP dialogue policy. However, as recognition degrades, we can model the effects of actively gathering information from the user to offset the loss of information in the utterance stream.

In the past, POMDPs have not been used for dialogue management because of the computational complexity involved in solving anything but trivial problems. We avoid this problem by using an

POMDP		Conventional MDP	
Average Reward Per Action	18.6 +/- 57.1	Average Reward Per Action	3.8 +/- 67.2
Average Dialogue Reward	230.7 +/- 77.4	Average Dialogue Reward	49.7 +/- 193.7

Table 3: A comparison of the rewards accumulated for the two algorithms (approximate POMDP and conventional MDP) using the full model.

		POMDP	Conventional MDP
User 1	Reward Per Action	52.2	24.8
	Errors per request	0.1 +/- 0.09	0.55 +/- 0.44
	Time to fill request	1.9 +/- 0.47	2.0 +/- 1.51
User 2	Reward Per Action	36.95	6.19
	Errors per request	0.1 +/- 0.09	0.825 +/- 1.56
	Time to fill request	2.5 +/- 1.22	1.86 +/- 1.47
User 3	Reward Per Action	49.72	44.95
	Errors per request	0.18 +/- 0.15	0.36 +/- 0.37
	Time to fill request	1.63 +/- 1.15	1.42 +/- 0.63

Table 4: A comparison of the rewards accumulated for the two algorithms using the full model on real users, with results given as mean +/- std. dev.

augmented MDP state representation for approximating the optimal policy, which allows us to find a solution that quantitatively outperforms the conventional MDP, while dramatically reducing the time to solution compared to an exact POMDP algorithm (linear vs. exponential in the number of states).

We have shown experimentally both in simulation and in preliminary user testing that the POMDP solution consistently outperforms the conventional MDP dialogue manager, as a function of erroneous actions during the dialogue. We are able to show with actual users that as the speech recognition performance varies, the dialogue manager is able to compensate appropriately.

While the results of the POMDP approach to the dialogue system are promising, a number of improvements are needed. The POMDP is overly cautious, refusing to commit to a particular course of action until it is completely certain that it is appropriate. This is reflected in its liberal use of verification questions. This could be avoided by having some non-static reward structure, where information gathering becomes increasingly costly as it progresses.

The policy is extremely sensitive to the parameters of the model, which are currently 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. For example, the optimality of a policy is strongly dependent on the design of the reward structure. It follows that incorporating a learning component that adapts the reward structure to reflect actual user satisfaction would likely improve performance.

6 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

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

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

Anthony Cassandra, Michael L. Littman, and Nevin L. Zhang. 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)*, pages 54–61, San Francisco, CA.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134.

Esther Levin, Roberto Pieraccini, and Wieland Eckert. 1998. Using Markov decision process for learning dialogue strategies. In *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

George E. Monahan. 1982. A survey of partially observable Markov decision processes. *Management Science*, 28(1):1–16.

Yasuhisa Niimi and Yutaka Kobayashi. 1996. Dialog control strategy based on the reliability of speech recognition. In *Proc. International Conference on Spoken Language Processing (ICSLP)*.

Ronald Parr and Stuart Russell. 1995. Approximating optimal policies for partially observable stochastic domains. In *Proceedings of the 14th International Joint Conferences on Artificial Intelligence*.

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

Nicholas Roy and Sebastian Thrun. 1999. Coastal navigation with mobile robots. In *Advances in Neural Processing Systems*, volume 12.

Satinder Singh, Michael Kearns, Diane Litman, and Marilyn Walker. 1999. Reinforcement learning for spoken dialog systems. In *Advances in Neural Processing Systems*, volume 12.

E. Sondik. 1971. *The Optimal Control of Partially Observable Markov Decision Processes*. Ph.D. thesis, Stanford University, Stanford, California.

Sebastian Thrun. 1999. Monte Carlo POMDPs. In S. A. Solla, T. K. Leen, and K. R. Müller, editors, *Advances in Neural Processing Systems*, volume 12.

Marilyn A. Walker, Jeanne C. Fromer, and Shrikanth Narayanan. 1998. Learning optimal dialogue strategies: a case study of a spoken dialogue agent for email. In *Proc. ACL/COLING'98*.

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