INVITED PAPER

POMDP-Based Statistical Spoken Dialog Systems: A Review

This paper presents the theory and practice of belief tracking, policy optimization, parameter estimation, and fast learning.

By Steve Young, *Fellow IEEE*, Milica Gašić, *Member IEEE*, Blaise Thomson, *Member IEEE*, and Jason D. Williams, *Member IEEE*

ABSTRACT | Statistical dialog systems (SDSs) are motivated by the need for a data-driven framework that reduces the cost of laboriously handcrafting complex dialog managers and that provides robustness against the errors created by speech recognizers operating in noisy environments. By including an explicit Bayesian model of uncertainty and by optimizing the policy via a reward-driven process, partially observable Markov decision processes (POMDPs) provide such a framework. However, exact model representation and optimization is computationally intractable. Hence, the practical application of POMDP-based systems requires efficient algorithms and carefully constructed approximations. This review article provides an overview of the current state of the art in the development of POMDP-based spoken dialog systems.

KEYWORDS | Belief monitoring; policy optimization; partially observable Markov decision process (POMDP); reinforcement learning; spoken dialog systems (SDSs)

I. INTRODUCTION

Spoken dialog systems (SDSs) allow users to interact with a wide variety of information systems using speech as the primary, and often the only, communication medium [1]–[3]. Traditionally, SDSs have been mostly deployed in call center applications where the system can reduce the need for a human operator and thereby reduce costs. More recently, the use of speech interfaces in mobile phones has become common with developments such as Apple's "Siri" and Nuance's "Dragon Go!" demonstrating the value of integrating natural, conversational speech interactions into mobile products, applications, and services.

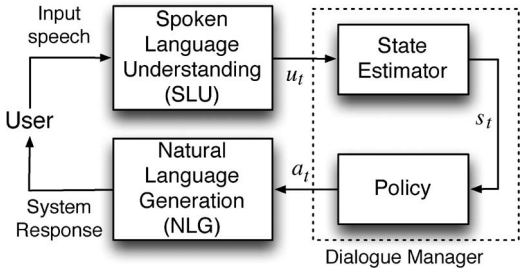


Fig. 1. ***Components of a finite-state-based spoken dialog system. At each turn the input speech is converted to an abstract representation of the user's intent u_t, the dialog state s_t is updated, and a deterministic decision rule called a policy maps the state into an action a_t in response.***

The principal elements of a conventional SDS are shown in Fig. 1.[1] At each turn t, a spoken language understanding (SLU) component converts each spoken input into an abstract semantic representation called a *user dialog act* u_t. The system updates its internal state s_t and

Manuscript received August 7, 2012; revised October 15, 2012; accepted October 15, 2012. Date of publication January 9, 2013; date of current version April 17, 2013.
S. Young, **M. Gašić**, and **B. Thomson** are with the Department of Engineering, Cambridge University, CB2 1PZ Cambridge, U.K. (e-mail: sjy@eng.cam.ac.uk; mg436@eng.cam.ac.uk; brmt2@eng.cam.ac.uk).
J. D. Williams is with Microsoft Research, Redmond, WA 98052 USA (e-mail: jawillia@microsoft.com).
Digital Object Identifier: 10.1109/JPROC.2012.2225812

[1]Multimodal dialog is beyond the scope of this paper, but it should be noted that the POMDP framework can be extended to handle multimodal input and output [4]. Depending on the application, both the input and the output may include a variety of modalities including gestures, visual displays, haptic feedback, etc. Of course, this could result in larger state spaces, and synchronization issues would need to be addressed.

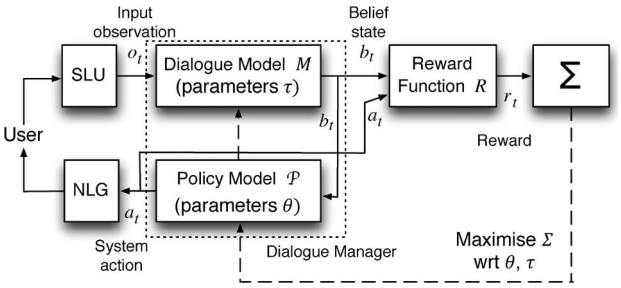


Fig. 2. ***Components of a POMDP-based spoken dialog system. In contrast to Fig. 1, the decoded input speech is now regarded as a noisy observation o_t of the underlying user intent u_t. Since u_t is hidden, the system maintains a distribution b_t over all possible dialog states and instead of trying to estimate the hidden dialog state, the system response is determined directly from b_t. In addition, the dialog model and policy are parameterized, and given an appropriate reward function, they can be optimized using reinforcement learning.***

determines the next system act via a decision rule $a_t = \pi(s_t)$, also known as a *policy*. The system act a_t is then converted back into speech via a natural language generation (NLG) component. The state s_t consists of the variables needed to track the progress of the dialog and the attribute values (often called slots) that determine the user's requirements. In conventional systems, the policy is usually defined by a flow chart with nodes representing states and actions, and arcs representing user inputs [5], [6].

Despite steady progress over the last few decades in speech recognition technology, the process of converting conversational speech into words still incurs word error rates in the range 15%–30% in many real-world operating environments such as in public spaces and in motor cars [7], [8]. Systems which interpret and respond to spoken commands must therefore implement dialog strategies that account for the unreliability of the input and provide error checking and recovery mechanisms. As a consequence, conventional deterministic flowchart-based systems are expensive to build and often fragile in operation.

During the last few years, a new approach to dialog management has emerged based on the mathematical framework of partially observable Markov decision processes (POMDPs[2]) [9]–[11]. This approach assumes that dialog evolves as a Markov process, i.e., starting in some initial state s_0, each subsequent state is modeled by a transition probability: $p(s_t|s_{t-1}, a_{t-1})$. The state s_t is not directly observable reflecting the uncertainty in the interpretation of user utterances; instead, at each turn, the system regards the output of the SLU as a noisy observation o_t of the user input with probability $p(o_t|s_t)$ (see Fig. 2). The transition and observation probability functions are represented by a suitable stochastic model, called here the *dialog model* $\mathcal{M}$. The decision as to which action to take at each turn is determined by a second stochastic model encoding the policy $\mathcal{P}$. As the dialog progresses, a reward is assigned at each step designed to mirror the desired characteristics of the dialog system. The dialog model $\mathcal{M}$ and the policy model $\mathcal{P}$ can then be optimized by maximizing the expected accumulated sum of these rewards either online through interaction with users or offline from a corpus of dialogs collected within a similar domain.

This POMDP-based model of dialog combines two key ideas: belief state tracking and reinforcement learning. These ideas are separable and have benefits on their own. However, combining them results in a complete and well-founded mathematical framework that offers opportunities for further synergistic gains. The potential advantages of this approach compared to conventional methods can be summarized as follows.

1) The belief state provides an explicit representation of uncertainty leading to systems that are much more robust to speech recognition errors [11]. The posterior probability of the belief state after each user input is updated via Bayesian inference in a process called *belief monitoring*. The design of the belief state allows user behavior to be captured via the model priors and the inference process is able to exploit the full distribution of recognition hypotheses such as confusion networks and N-best lists. Thus, evidence is integrated across turns such that a single error has significantly reduced impact, and in contrast to conventional systems, user persistence is rewarded. If the user repeats something often enough, the system's belief in what they said will increase in time as long as the correct hypothesis appears repeatedly in the N-best list.
2) By maintaining a belief distribution over all states, the system is effectively pursuing all possible dialog paths in parallel, choosing its next action not based on the most likely state but on the probability distribution across all states. When the user signals a problem, the probability of the current most likely state is reduced and the focus simply switches to another state. Thus, there is no requirement for backtracking or specific error correction dialogs. This allows powerful dialog policies to be embedded in a simple homogenous mapping from belief state to action.
3) The explicit representation of state and policy-derived action allows dialog design criteria to be incorporated by associating rewards with state–action pairs. The sum of these rewards constitutes an objective measure of dialog performance and enables reinforcement learning to be used to maximize performance both offline using dialog corpora and online through interaction with real users. This leads to optimal decision policies, avoids the cost of expensive manual tuning and

[2]Pronounced "pom dee pees."

refinement procedures, and enables more complex planning to be implemented than would be feasible using only manual handcrafted designs.

Converting these potential benefits of the POMDP approach into practice is, however, far from trivial, and there are many issues to resolve. The state–action space of a real-world SDS is extremely large, and its efficient representation and manipulation requires complex algorithms and software. Real-time Bayesian inference is equally challenging, and exact policy learning for POMDPs is intractable, hence efficient approximation techniques must be used. Finally, the most straightforward way to optimize a POMDP-based SDS is through direct interaction with users. However, real users willing to help train a system are often not available in sufficient numbers, hence, user simulators are required that can replicate user behavior with sufficient accuracy to enable model parameters to be optimized to an acceptable level of performance (a discussion of user simulation is given in Section V).

Despite these difficulties, considerable progress has been made over the last five years in solving these problems, and the purpose of this paper is to review that progress and provide a coherent up-to-date view of the state of the art in POMDP-based dialog systems.

The paper is organized as follows. First, Section II outlines the basic mathematics underlying POMDP-based dialog systems in order to provide the necessary background for the subsequent sections and to establish a consistent notation. Section III then explains the options available for approximating the belief state and presents algorithms for efficient belief monitoring. Section IV reviews policy representation and the use of reward functions and policy optimization via reinforcement learning. Section V completes the review of core technology with an overview of current approaches to user simulation. Having established the basic framework, Sections VI and VII review a number of recent developments in optimization of dialog model parameters and fast adaptation. To give some indication of the potential for real-world deployment, Section VIII briefly describes some existing prototype systems and applications that incorporate POMDP-based dialog management, with some example evaluations given in Section IX. Finally, for completeness, Section X provides a historical perspective and Section XI concludes.

II. PARTIALLY OBSERVABLE MARKOV DECISION PROCESSES

Formally, a POMDP is defined as a tuple $(\mathbb{S}, \mathbb{A}, \mathbb{T}, \mathbb{R}, \mathbb{O}, \mathbb{Z}, \gamma, b_0)$ where $\mathbb{S}$ is a set of states with $s \in \mathbb{S}$; $\mathbb{A}$ is a set of actions with $a \in \mathbb{A}$; $\mathbb{T}$ defines a transition probability $P(s_t|s_{t-1}, a_{t-1})$; $\mathbb{R}$ defines the expected (immediate, real-valued) reward $r(s_t, a_t) \in \Re$; $\mathbb{O}$ is a set of observations with $o \in \mathbb{O}$; $\mathbb{Z}$ defines an observation probability $P(o_t|s_t, a_{t-1})$; γ is a geometric discount factor $0 \leq \gamma \leq 1$; and b_0 is an initial belief state, defined below.

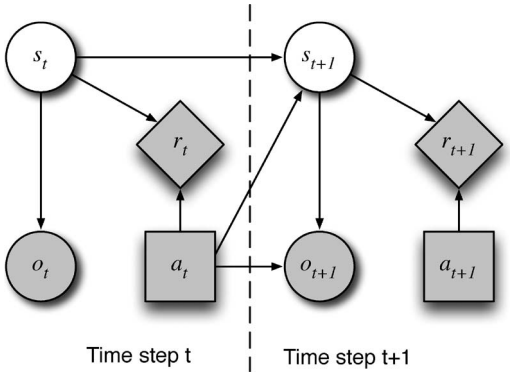


Fig. 3. *A POMDP, shown as an influence diagram. In this paper, clear circles are hidden random variables, shaded circles are observed random variables, squares are system actions, diamonds are real-valued rewards, and arrows show causality.*

The POMDP operates as follows. At each time step, the world is in some unobserved state s_t. Since s_t is not known exactly, a *distribution* over possible states called a *belief state* b_t is maintained where $b_t(s_t)$ indicates the probability of being in a particular state s_t. Based on b_t, the machine selects an action a_t, receives a reward r_t, and transitions to (unobserved) state s_{t+1}, where s_{t+1} depends only on s_t and a_t. The machine then receives an observation o_{t+1}, which is dependent on s_{t+1} and a_t. This process is represented graphically as an influence diagram in Fig. 3.

Given an existing belief state b_t, the last system action a_t, and a new observation o_{t+1}, the new updated belief state b_{t+1} is given by [12]

$$b_{t+1}(s_{t+1}) = \eta P(o_{t+1}|s_{t+1}, a_t) \sum_{s_t} P(s_{t+1}|s_t, a_t) b_t(s_t) \quad (1)$$

where $\eta = 1/P(o_{t+1}|b_t, a_t)$ is a normalization constant and where b_0 is the initial belief state distribution before the first system action has been taken.[3]

The system action is determined by a policy π, which can be represented in a variety of ways. It is most commonly either a deterministic mapping from belief states to actions $\pi(b) \in \mathbb{A}$ or stochastically via a distribution over actions $\pi(a|b) \in [0, 1]$ where $\pi(a|b)$ is the probability of taking action a in belief state b, and $\sum_a \pi(a|b) = 1 \; \forall \; b$. For convenience, both types of policy will use the same symbol π, with the presence of the action in the notation determining whether the policy is deterministic or stochastic. Note, however, that other definitions are possible such as finite state controllers [13], or mappings from

[3]The notation for belief states can be confusing: b_t represents a probability distribution over the hidden state space $\mathbb{S}$ at time t; $b_t(s)$ denotes the probability of a specific state s given belief state b_t; and $b_{t+1}(s)$ represents the probability of state s given a new updated belief state b_{t+1}, which, in general, will be different from $b_t(s)$.

finite-length sequences of observations to actions (cf. predictive state representations [14]).

The discounted sum of rewards expected by starting in belief state b_t and following policy π is given by the *value function* $V^\pi(b_t) = \mathbb{E}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots]$, which can be expressed recursively for a deterministic policy as

$$V^\pi(b_t) = r(b_t, \pi(b_t)) + \gamma \sum_{o_{t+1}} P(o_{t+1}|b_t, \pi(b_t)) V^\pi(b_{t+1}) \quad (2)$$

and by

$$V^\pi(b_t) = \sum_{a_t} \pi(a_t|b_t) \Big\{ r(b_t, a_t) + \gamma \sum_{o_{t+1}} P(o_{t+1}|b_t, a_t) V^\pi(b_{t+1}) \Big\} \quad (3)$$

for a stochastic policy. A related quantity is the Q-function $Q^\pi(b, a)$, which provides the expected discounted sum of rewards if a specific action a is taken given belief state b, and then policy π is followed. Clearly, for a deterministic policy, $V^\pi(b) = Q^\pi(b, \pi(b))$ and for a stochastic policy

$$V^\pi(b) = \sum_a \pi(a|b) Q^\pi(b, a). \quad (4)$$

An *optimal policy* π^* is one that maximizes V^π to yield V^*

$$V^*(b_t) = \max_{a_t} \Big[r(b_t, a_t) + \gamma \sum_{o_{t+1}} P(o_{t+1}|b_t, a_t) V^*(b_{t+1}) \Big] \quad (5)$$

which is the Bellman optimality equation for POMDPs [15]. In the POMDP literature, finding a policy π that satisfies (5) is often called "solving" or "optimizing" the POMDP. For simple tasks, both exact [12] and approximate [16]–[20] solution methods have been developed. However, standard POMDP methods do not scale to the complexity needed to represent a real-world dialog system. Even in a moderately sized system, the number of states, actions, and observations can each easily be more than 10^{10}. Even enumerating $P(s_{t+1}|s_t, a_t)$ is intractable, and as a result computing (1) directly and applying direct solution methods to (5) is very difficult. Instead, approximations have been developed that exploit domain-specific properties of the spoken dialog task in order to provide compact representations for both the model and the policy, and to allow tractable algorithms for performing belief monitoring and policy optimization. These are described in the following sections.

III. BELIEF STATE REPRESENTATION AND MONITORING

This section reviews the possible approaches to representing the dialog model $\mathcal{M}$ shown in Fig. 2. In a practical task-oriented SDS, the state must encode three distinct types of information: the user's goal g_t, the intent of the most recent user utterance u_t, and the dialog history h_t [11], [21]. The goal encompasses the information that must be gleaned from the user in order to fulfil the task, the most recent user utterance represents what was actually said in contrast to what was recognized, and the history tracks pertinent information relating to previous turns. This suggests that the state should be factored into three components

$$s_t = (g_t, u_t, h_t). \quad (6)$$

The resulting influence diagram is shown in Fig. 4 in which some reasonable independence assumptions have been introduced. Factoring the state in this way is helpful because it reduces the dimensions of the state transition matrix and it reduces the number of conditional dependencies.

Plugging the factorization in (6) into the belief update (1) and simplifying according to the independence

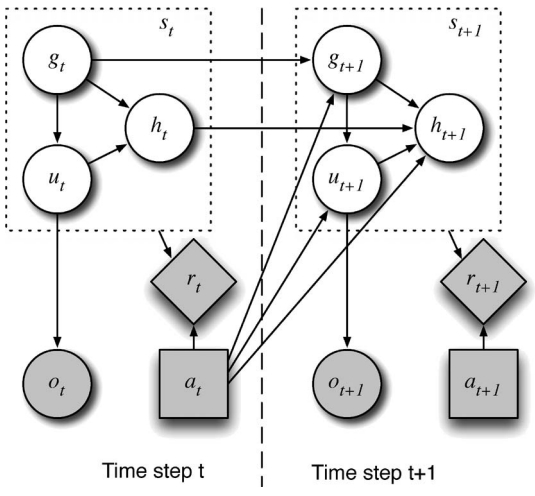


Fig. 4. *Influence diagram representation of a factored state SDS-POMDP. The hidden dialog state s_t is factored into a user goal g_t, the user's last input u_t, and key elements of the dialog history h_t. This allows conditional independence assumptions to be introduced both within the state and from one time slice to the next. No arc is required from actions to observations since the recognized speech o_t is conditionally independent given the user's actual utterance u_t.*

assumptions shown in Fig. 4 gives the basic belief update equation for a statistical SDS

$$\begin{aligned} b_{t+1}(g_{t+1}, u_{t+1}, h_{t+1}) = & \eta P(o_{t+1}|u_{t+1}) && \text{(a)} \\ & \cdot P(u_{t+1}|g_{t+1}, a_t) && \text{(b)} \\ & \cdot \sum_{g_t} P(g_{t+1}|g_t, a_t) && \text{(c)} \\ & \cdot \sum_{h_t} P(h_{t+1}|g_{t+1}, u_{t+1}, h_t, a_t) && \text{(d)} \\ & \cdot b_t(g_t, h_t). && (7) \end{aligned}$$

The terms on the right-hand side in (7) reflect each of the factors determining the belief state and the underlying models which are, therefore, needed to represent these factors in a practical system.

a) The *observation model* represents the probability of an observation o given the user's actual utterance u. This encapsulates the effects of speech understanding errors.
b) The *user model* represents the likelihood that the user would utter u given the previous system output and the new system state. This encapsulates user behavior.
c) The *goal transition model* represents the likelihood that the user goal has changed.
d) The *history model* represents the system's memory of the dialog to date.

There are, of course, possible variations to this factorization. For example, user affect may also be factorized out [22] but most current approaches fit broadly within this model.

While the factorization in (6) does significantly reduce the POMDP model complexity, it is still too complex to support tractable real-world systems. Further approximation is, therefore, necessary, for which two main approaches have emerged:

1) the N-best approach including pruning and recombination strategies [23]–[26];
2) the factored Bayesian network approach [22], [27], [28].

These two approaches are discussed below.

A. N-Best Approaches

In N-best approaches, the belief state is approximated by a list of the most likely states (or groups of states) with their probabilities. This means that dialog states corresponding to the most likely interpretations of the user's intent are well modeled, with other states given low probability mass. One example of this approach is the hidden information state (HIS) model [23], which groups similar user goals into equivalence classes called *partitions* on the assumption that all of the goals in the same partition are equally probable. The partitions are tree structured to take account of the dependencies defined in the domain ontology, and they are built using slot–value pairs from the N-best list of recognition hypotheses and the last system output. The combination of a partition, a user act from the N-best list, and the associated dialog history forms a *hypothesis*. A probability distribution over the most likely hypotheses is maintained during the dialog, and this constitutes the belief space. Belief monitoring then requires only the hypothesis beliefs to be updated and, since there are relatively few hypotheses, this can easily be done in real time. The update equation for the HIS model follows directly from (7) with the further simplification that the user goal is normally assumed to be constant [23]

$$\begin{aligned} b_{t+1}(p_{t+1}, u_{t+1}, h_{t+1}) = & \eta P(o_{t+1}|u_{t+1}) P(u_{t+1}|p_{t+1}, a_t) \\ & \cdot \sum_{h_t} P(h_{t+1}|p_{t+1}, u_{t+1}, h_t, a_t) \\ & \cdot P(p_{t+1}|p_t) b(h_t) \qquad (8) \end{aligned}$$

where p_t is a partition and the term $P(p_{t+1}|p_t)$ represents the probability that a partition p_t will be split into two subpartitions $p_t \rightarrow \{p_{t+1}, p_t - p_{t+1}\}$. A similar approach is taken in [24], except that both slot values and their complements are used to build a frame. This is particularly useful in negotiation-type dialogs, where the users can change their mind and ask for an alternative. It also enables a form of first-order logic to be expressed and understood by the system. However, the domain ontology is not used when building frames losing the benefits of modeling conditional dependence. In [26], it is shown that complements can also be incorporated into the HIS approach giving the advantage of modeling both conditional dependence and first-order logic.

The N-best approach can also be viewed as running N conventional dialog systems in parallel such that each parallel thread tracks a different interpretation of what the user said. In this case, a list of hypothesized dialog states is maintained; associated probabilities can be assigned using a mixture distribution [29], a discriminative classifier [30], or with a heuristic scoring function [31]. Although it lacks the compactness of the HIS representation, this approach provides a simple upgrade path for conventional deterministic dialog systems.

Both partition-based and frame-based approaches have the inherent problem that the number of partitions (frames) grows exponentially with dialog length. Hence, in practical implementations, some form of pruning is required. In [25], partitions are organized into a tree, and then low probability partitions are recombined with their parent, thus limiting the number of partitions. This has the problem, however, that high probability slots can be pruned if they are distributed over many partitions. A more effective solution is proposed in [26], where at every dialog

turn a marginal probability distribution is calculated for each slot. Then, low probability slot–value pairs are pruned by recombining all the partitions that have that slot–value pair with the partitions that have the complement of that slot–value pair. In this way, the method supports dialogs of arbitrary length.

B. Factored Approaches

The alternative to maintaining an N-best list of user goals is to factor the user goal into concepts that can be spoken about by the system. This is illustrated in Fig. 5, which shows an example from a tourist information system in which entities have a *type* such as *{hotel, restaurant, bar}*, the kind of *food* served and an *area* such as *{north, south, center, etc.}* [28]. The *food* and *area* slots are dependent only on the type, even though, in practice, there are many more restaurants in the center of town. This illustrates the differing tradeoffs between the N-best approach, which can model all dependencies but with an incomplete distribution, and the slot-level factoring approach, which can handle only a limited number of dependencies but can model the complete distribution.

Once the Bayesian network for the slot-level factoring has been compiled, standard belief propagation algorithms can be used to update the beliefs [32]. In the case of concepts that are conditionally independent, the belief propagation algorithm will give exact updates of the marginal distributions for each of the concepts, as used in [22]. Limited dependencies can also be modeled, although this requires approximate methods such as loopy belief propagation. The Bayesian update of dialog state (BUDS) model is one example that is based on this approach [27]; particle filters can also be used [33].

It is interesting to note that N-best approximations can also be used within a factored model. Instead of the factored model updating beliefs for all the possible values of a node in the network, only an N-best list of values is updated. Partitions of the states in a node are also possible [34]. This combines many of the benefits of the two approaches.

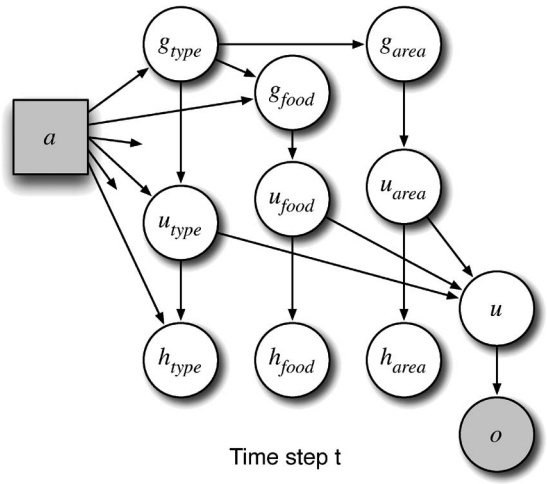


Fig. 5. *Influence diagram for a single time slice of a BUDS POMDP in which the state is further factored into concept or slot-level components. In this simplified example taken from the tourist information domain, three slot-level values are required: the type of venue, the kind of food served, and the area in which the venue is located. Note that food and area are conditionally independent given the type of venue and last action. (Although not shown explicitly to avoid cluttering the diagram, all slot-level nodes are dependent on the last system action.)*

IV. POLICY REPRESENTATION AND REINFORCEMENT LEARNING

This section presents the representation and estimation of the policy model $\mathcal{P}$, shown in Fig. 2, which provides a mapping between the belief state b and the appropriate system action a. The objective is to find an optimal policy π^* that maximizes the expected sum of discounted rewards at the end of the dialog.

The belief space of a POMDP spans an $(|\mathbb{S}|-1)$-dimensional simplex where $|\mathbb{S}|$ is the cardinality of the underlying hidden state space. Points in belief space that are close to each other will share the same action, and hence, a nonparametric policy must encode first a partitioning of belief space such that all points within any partition map to the same action, and second, it must encode the optimal action to take for each partition. While exact representations of a POMDP dialog policy are possible, for example, by compressing belief space [35] or dynamically reassigning states [36], exact representations are all intractable for real-world problems such as spoken dialog systems where both state and action spaces will typically be very large. Thus, a compact representation of the policy is essential.

Fortunately, there are some mitigating constraints that can be exploited. First, only a relatively small part of belief space will actually be visited during any normal dialog, and second, the range of plausible actions at any specific point in belief space will often be restricted. This introduces the notion of a compressed feature space called *summary space* in which both states and actions are simplified in order to allow tractable policy representation and optimization [37], [38]. Summary space can, therefore, be regarded as a subspace of the full master space whereby belief tracking is performed in master space, and decision taking and policy optimization take place in summary space. The runtime operation of a master summary-space POMDP is, therefore, as follows. After belief updating, the belief state b in master space is mapped to a vector of features $\hat{b}$ and a corresponding set of candidate actions $\{\hat{a}\}$. The policy is then used to select the best action to take $\hat{b} \rightarrow \hat{a}$ from the set of candidate actions and a second heuristic is used to map $\hat{a}$ back into a full action a in master space.

Summary-space mapping requires two components: a mechanism to select candidate actions in master space,

and functions to extract features from the belief state and candidate actions. The simplest method for selecting candidate actions is to include all types of *dialog acts* (e.g., *greet*, *ask*, *confirm*, *inform*, etc.) applicable to any *concept* or *slot* (e.g., *venue type*, *food type*, *star rating*, etc.), populating slot values by highest belief [28], [37]. This approach has the benefit of being entirely automatic; however, it still admits some spurious candidate actions, such as taking a greeting action in the middle of a dialog, or attempting to confirm a value before the system has asked about it. An alternative method for selecting candidate actions is to construct a handcrafted *partial program* [39], [40] or a Markov logic network [41]. These approaches have the benefit of allowing arbitrary human knowledge about dialog flow to be incorporated, and to explicitly set business rules, for example, requiring that certain actions be taken before others such as successfully collecting a password before allowing funds to be transferred. It has also been shown that constraining the set of candidate actions results in faster convergence to an optimal policy, since many spurious actions are pruned away [40]. However, rules require human effort to encode, and there is a risk that optimal candidate actions may be inadvertently excluded. Intermediate methods are possible as well, such as allowing every dialog act to be a candidate action, but constraining the slots that the act operates on using handcrafted heuristics [42].

The second component required for summary-space mapping are functions to extract features from the belief state and each candidate action. For actions, typically one binary feature is created for each dialog act, or for each valid dialog act/slot pair, such as *confirm(food)*. This generally results in a 20–30 dimensional vector of action features, where each dimension represents a unique (summary) action. State features are typically heterogeneous, consisting of real-valued quantities, binary values, categorical values, etc. Typical state features include: the belief in the top N user goals or partitions; the top marginal belief in each slot; properties of the top user goal or partition, such as the number of matching database entries; an indication of which system actions are available; properties of the dialog history, such as whether the top user goal/partition has been confirmed; the most likely previous user action; or combinations of these features [28], [37], [40], [42], [43]. A typical system has 5–25 features in total which are usually hand selected, although some work has been done to automate feature selection [43], [44]. State features need not be limited to information in the belief state: features may also draw on information being tracked outside the belief state, such as information in databases, information from past dialogs, usage context, etc.

Given a specific summary space, a policy may be represented as an explicit deterministic mapping: $\pi(\hat{b}) \rightarrow \hat{a}$ or as a conditional probability distribution $\pi(\hat{b}, \hat{a}) = p(\hat{a}|\hat{b})$ where the required action is selected by sampling the distribution. Note that the policy is now a function of the summary belief state and actions, instead of the original belief state and actions. One can think of this either as a new function that gives an approximation to the value in the original space, or as a policy over a new Markov decision process, where the states and actions are now the summary states and actions. The same is true of the Q-function, and both these functions will now be used in this context.

In the case of deterministic mappings, the dominant approach is to find an optimal Q-function (see Section II) Q^* that maximizes $\max_{\hat{a}}\{Q(\hat{b}, \hat{a})\}$ for all $\hat{b}$, then

$$\pi^*(\hat{b}) = \arg\max_{\hat{a}} \{Q^*(\hat{b}, \hat{a})\}. \tag{9}$$

The Q-function itself can either be parametric or nonparametric in which case the belief state is quantized into a codebook $\{\hat{b}_i\}$ and $Q^*(\hat{b}_i, \hat{a})$ can be computed for each discrete codebook entry $\hat{b}_i$.

To illustrate some of these options, five different methods of policy optimization are now presented: planning under uncertainty, value iteration, Monte Carlo optimization, least squares policy iteration (LSPI), and natural actor–critic. These methods have been selected because all have been applied to end-to-end working dialog systems. Some further optimization methods will be covered later in Section VII on fast adaptation. While representative, these five methods are not exhaustive, for example, Q-learning [45]–[48] and SARSA [49]–[51] are also popular techniques.

A. Planning Under Uncertainty

One approach to policy optimization is to view action selection in summary space as *planning under uncertainty*. In this approach, $\hat{b}$ is viewed not as a vector of arbitrary features, but rather as a distribution over a hidden state variable in summary space $\hat{s}$. This hidden summary state might express, for example, whether the top hypothesis in master space is correct. In addition, it is assumed observations o (in master space) can be mapped to a compact observation in summary space $\hat{o}$.

With these definitions in place, dialogs can be collected with a random policy, and three models of summary-space dynamics can be estimated: $P(\hat{s}'|\hat{s}, \hat{a})$, $P(\hat{o}'|\hat{s}, \hat{a})$, and $R(\hat{s}, \hat{a})$. These models define a classical POMDP *in summary space* and admit standard methods for planning under uncertainty, including different versions of point-based value iteration [16], [18], SARSOP [52], heuristic search value iteration [17], [19], and short look-aheads [22], [53]. These methods have been applied to dialog systems in a variety of domains, mainly in simulation [22], [37], [53] [54]–[60] and also in an end-to-end spoken dialog system [61].

Optimization by planning under uncertainty is attractive in that optimization methods are principled, and in some cases provide well-defined optimality bounds [17], [19]. Another strength is that a policy can be computed from any corpus that contains sufficient exploration of possible dialog paths. However, it suffers from several problems. First, the summary state $\hat{s}$ and observation $\hat{o}$ may be difficult to define, and incorporation of extrinsic features in $\hat{b}$ may require careful engineering of the models $P(\hat{s}'|\hat{s},\hat{a})$ and $P(\hat{o}'|\hat{s},\hat{a})$. Also, while great strides have been made in scaling optimization methods for planning under uncertainty, supporting large state or observation spaces in summary space can be problematic. For these reasons, techniques have been developed which avoid defining states and observations in summary space. These are described in the next four sections.

B. Value Iteration

Instead of defining states and observations in summary space, in *value iteration* belief states $\hat{b}$ are viewed as arbitrary feature vectors, and dynamics are estimated directly in this feature space. Starting again with dialogs collected with actions taken randomly, feature vectors are quantized into grid points $\{\hat{b}_i\}$, and then the transition function $P(\hat{b}_j|\hat{b}_i,\hat{a})$ and the reward function $R(\hat{b}_i,\hat{a})$ are estimated [40], [62]. Finally, standard value iteration can be applied [63]

$$Q(\hat{b}_i,\hat{a}) = r(\hat{b}_i,\hat{a}) + \gamma \sum_j P(\hat{b}_j|\hat{b}_i,\hat{a}) \max_{\hat{a}'} Q(\hat{b}_j,\hat{a}') \quad (10)$$

where (10) is applied repeatedly until convergence. The clusters $\hat{b}_i$ can either be distinct *hard* clusters, or *soft* clusters where observed $\hat{b}$ are shared by nearby $\hat{b}_i$. In this case, the contributions of $\hat{b}_i$ that are further away from $\hat{b}$ can be down-weighted [40], [62].

Value iteration's main attraction is that it is simple to apply, and it allows a policy to be estimated from any corpus that contains sufficient exploration of possible dialog paths. However, it suffers from several problems. First, it requires an estimate of transition dynamics for the entire space $P(\hat{b}_j|\hat{b}_i,\hat{a})$, when in practice it is only important to estimate the portions of space traversed by the currently optimal policy. Hence, value iteration is sample inefficient. Second, if the state features are not Markovian, i.e., if they do not accurately capture all relevant history for conditioning transitions, errors will be introduced into the policy. To address both of these issues, incremental *online* learning methods have also been explored.

C. Monte Carlo Optimization

In Monte Carlo optimization, the value function Q is estimated *online*, iteratively, while interacting with the user (or more likely interacting with a user simulator; see Section V). The current estimate of the policy guides future action selection, so less exploration time is expended on regions of the state space with low value. In addition, updates are performed at the end of each dialog; hence, whole-dialog rewards can be used to directly value the current policy, which mitigates against the effects of any non-Markovian dynamics in the state features.

To perform Monte Carlo optimization, the sequence of states, actions, and rewards observed in each dialog of length T are recorded as tuples $(\hat{b}_{i(t)},\hat{a}_t,r_t)$ for $t = 0, 1, \ldots, T$. The discounted return from each belief point $\hat{b}_i$ visited at time t is given by

$$R_{i(t)} = \sum_{t \le \tau \le T} \gamma^{(\tau - t)} r_\tau. \quad (11)$$

Two functions accumulate sums across all dialogs in the corpus. First, $q(i,\hat{a})$ is a sum of all returns R observed when action $\hat{a}$ was taken in grid point $\hat{b}_i$. Second, $n(i,\hat{a})$ is a count of the instances of action $\hat{a}$ being taken in gridpoint $\hat{b}_i$. Both $q(i,\hat{a})$ and $n(i,\hat{a})$ are initialized to zero for all i and $\hat{a}$. At the end of each dialog, these tallies are then updated

$$q(i(t),\hat{a}) = q(i(t),\hat{a}) + R_{i(t)} \quad (12)$$

$$n(i(t),\hat{a}) = n(i(t),\hat{a}) + 1 \quad (13)$$

for $t = 0, 1, \ldots, T$. The Q values can then be reestimated for all belief points i visited in the dialog and all $\hat{a}$ as

$$Q(\hat{b}_i,\hat{a}) = \frac{1}{n(i,\hat{a})} q(i,\hat{a}). \quad (14)$$

To converge to an optimal policy, the dialog system must take a mixture of currently optimal and exploratory actions

$$\hat{a} = \begin{cases} \text{RandomAction}, & \text{with probability } \epsilon \\ \arg\max_{\hat{a}} Q(\hat{b}_i,\hat{a}), & \text{with probability } 1-\epsilon \end{cases} \quad (15)$$

where ϵ controls the exploration rate. Typically, this is large initially but reduces as learning progresses. As in the value iteration case, the state discretization can be done by clustering neighboring points into distinct *hard* clusters [42], or *soft* clusters where the contribution of a belief point $\hat{b}$ to the tallies $q(i,\hat{a})$ and $n(i,\hat{a})$ is reduced in proportion to its distance from the template point $\hat{b}_i$ [64].

One important consideration for Monte Carlo optimization is the number of grid points to use. If too many grid points are used, poor performance will result due to overfitting. Conversely, having too few grid points leads to poor performance due to a lack of discrimination in decision making. For building practical systems with five summary-state components and ten distinct summary actions, 750–1000 grid points have been found to be sufficient when trained with around 100 000 simulated dialogs [42].

As compared to value iteration, online Monte Carlo optimization is more sample efficient and suffers less when the transitions are non-Markovian in the state features. However, since Monte Carlo optimization is an online method, it cannot be performed using a previously collected corpus. Like value iteration, Monte Carlo optimization requires quantizing the feature space into grid regions. As the number of features grows, this becomes a source of intractability, which motivates the use of functional approximations for the Q-function, presented next.

D. Least Squares Policy Iteration

Instead of using discrete grid points in summary space, the next two methods utilize linear models to represent a policy. The first is LSPI [65]. LSPI assumes that the Q-function can be represented as a weighted sum of the features ϕ

$$Q(b,\hat{a}) \approx \phi(b,\hat{a})^\top \theta \tag{16}$$

where θ is a (column) weight vector, $\phi(b,\hat{a})$ outputs features of b that are important for making decisions relevant to summary action $\hat{a}$, and θ and $\phi(b,\hat{a})$ are both of size K. Notice that the functional mapping from master belief space to summary belief space is now explicit in the function ϕ. The Q-function is thus approximated as a plane in feature space. This may result in suboptimal policies if the representation is not rich enough.

LSPI operates on a given, *fixed* dialog corpus with $N+1$ turns $\{b_n, \hat{a}_n, r_n, b'_n\}$ where b'_n is the belief state following b_n in the corpus. LSPI begins with some policy π, and optimizes it by iteratively estimating its value function Q, then using Q to improve π. To estimate Q, we start with the identity that defines the value function Q

$$Q(b,\hat{a}) = r + \gamma Q(b', \pi(\hat{a})). \tag{17}$$

Substituting (16) into (17) yields a system of N equations in K unknowns

$$\phi(b_n,\hat{a})^\top \theta = r_n + \gamma \phi\big(b'_n, \pi\big(b'_n\big)\big)^\top \theta \tag{18}$$

for $n = 0 \ldots N-1$. This system of equations can be solved in a least squares sense by setting $\theta = Z^{-1}y$ where [65]

$$Z = \sum_n \phi(b_n,\hat{a}_n)\big(\phi(b_n,\hat{a}_n) - \gamma\phi\big(b'_n, \pi\big(b'_n\big)\big)\big)^\top \tag{19}$$

$$y = \sum_n \phi(b_n,\hat{a}_n) r_n \tag{20}$$

where Z is a $K \times K$ matrix and y is a K-dimensional vector. $\pi(b)$ is then updated by

$$\pi(b) = \arg\max_{\hat{a}} \phi(b,\hat{a})^\top \theta \tag{21}$$

and reestimation of θ and π then continues until both have converged.

LSPI is attractive in that it avoids estimating grid points in belief space in favor of a linear model for Q. It also has no learning parameters to tune, and it can be used both online and offline, although in the case of the latter it can be difficult to find corpora with sufficient variability to learn policies that are significantly different from the policy used to collect the corpus.

One drawback of LSPI is that the least squares solution step is cubic in K; hence, for practical applications of LSPI, the feature set must be kept quite small. To assist this, methods have been developed that perform feature selection in conjunction with LSPI [43] whereby, at each iteration, a fast but approximate estimate of θ is computed on the full set of features; then only the features with the highest weights are used in the expensive but more accurate least squares solution. Experiments show this method produces policies of similar performance but with far fewer features [43].

E. Natural Actor–Critic Optimization

LSPI uses a linear model to estimate the value function Q. An alternative approach is to represent the policy directly as an explicit probability distribution over actions, parameterized by an adjustable weight vector θ

$$\pi(\hat{a}|b,\theta) = \frac{e^{\theta\cdot\phi_{\hat{a}}(b)}}{\sum_{\hat{a}'} e^{\theta\cdot\phi_{\hat{a}'}(b)}} \tag{22}$$

where $\phi_{\hat{a}}(b)$ determines the features of the belief state b that are important for making decisions relevant to summary action $\hat{a}$. As above, ϕ is specified by the system designer, and ϕ and θ both have dimensionality K. π is as before, with the θ simply indexing which policy parameters to use [i.e., policy $\pi_\theta(\hat{a}|b) = \pi(\hat{a}|b,\theta)$].

Representing the policy in this way has two advantages. First, it avoids the need to choose grid points and the problems that result. Second, since $\pi(\hat{a}|b,\theta)$ is differentiable with respect to θ, it allows gradient ascent to be applied for optimization. The expected cumulative reward for a single dialog as a function of θ is

$$R(\theta) = \mathbf{E}\left[\sum_{t=0}^{T-1} r(b_t, \hat{a}_t)|\pi_\theta\right]. \tag{23}$$

By rewriting the log likelihood ratio [66] and Monte Carlo sampling from a batch of N dialogs, the gradient can be estimated as

$$\nabla R(\theta) = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T_n-1} \nabla_\theta \log \pi\left(\hat{a}_t^n|b_t^n,\theta\right) Q\left(b_t^n,\hat{a}_t^n\right). \tag{24}$$

Although (24) provides an estimate for the plain gradient, it has been shown that the natural gradient $\tilde{\nabla} R(\theta) = F_\theta^{-1}\nabla R(\theta)$ is more effective for optimization of statistical models where F_θ is the Fisher information matrix [67]. Furthermore, this natural gradient w can be found without actually computing the Fisher matrix by using a least square method to solve the following system of equations

$$R^n = \left[\sum_{t=0}^{T_n-1} \nabla_\theta \log \pi\left(\hat{a}_t^n|b_t^n,\theta\right)^\top\right]\cdot w + C \tag{25}$$

for each dialog $n = 1 \ldots N$ where R^n is the actual cumulative reward earned for dialog n and C is a constant that can be interpreted as the expected cumulative reward of the dialog system starting from the initial state. The total expected reward can then be iteratively maximized through gradient ascent via the parameter update $\theta' \leftarrow \theta + \delta w$ where δ is the step size. This is called the natural actor–critic (NAC) algorithm [68], and has been applied to a number of spoken dialog systems [28], [69]–[72].

NAC is particularly well suited for online operation because the policy is represented as an explicit distribution over actions. Unlike the ϵ-greedy exploration used in Monte Carlo optimization (where random actions are chosen with a uniform distribution), the NAC optimization process can differentiate between less promising and more promising actions and, therefore, explores the solution space more efficiently.

As with LSPI, NAC requires the solution of a system of linear equations (25), which is again cubic in the number of features, so it is important to choose a compact set of representative features.

F. Section Summary

To conclude this section, five approaches to policy representation and optimization have been presented, which as well as being representative of current practice also illustrate some of the principal design choices. In particular, planning under uncertainty uses the belief state as a probability distribution directly, whereas value iteration and Monte Carlo optimization require belief space to be quantized, and LSPI and NAC use functional approximations based on weighted linear models of belief state features. Monte Carlo methods and NAC (and SARSA) must be run *on-policy*; planning under uncertainty, value iteration, and LSPI (and Q-learning) may be run off-policy, in *batch*. On-policy methods require that interactions be obtained following the policy under optimization, whereas off-policy methods can perform optimization with interactions obtained by following a different policy. A distinction can also be made between methods like planning under uncertainty and value iteration that perform *planning* on a given model of dynamics, and the other methods described (including Q-learning and SARSA) which concurrently *learn* about the environment and plan simultaneously. NAC also differs in that it utilizes a stochastic policy from which actual actions can be drawn by sampling the policy distribution. This allows a dialog system to explore alternative actions more effectively by avoiding searching regions of the belief action space that are very unlikely to ever be visited. Unfortunately, this gain is offset by the generally slow learning characteristics of gradient ascent and, in practice, the NAC algorithm requires around 10^5 dialogs to optimize a policy [28]. Since the availability of large diverse corpora and/or large numbers of real users willing to interact with a partially trained dialog system is normally limited, significant attention has been paid to developing user simulators for training statistical dialog systems, and this will be dealt with next.

V. USER SIMULATORS

Learning directly from corpora is problematic since the state space that prevailed during the collection of the data may differ from that used in policy optimization and also it precludes the use of online interactive learning algorithms. An alternative is to build a model of the user that can interact directly with a dialog system and which can itself be trained on corpora. Such a user simulator can then be used for a wide variety of development, training, and evaluation purposes [73].

User simulators normally operate at the abstract level of dialog acts. Given a sequence of user acts and system responses, the aim is to model the distribution of plausible user responses

$$p(u_t|a_t, u_{t-1}, a_{t-1}, u_{t-2}, \ldots) \tag{26}$$

from which an actual user response can be sampled. As indicated in Fig. 2, in a real system, the dialog manager only has access to a noisy observation $o_t = [\tilde{u}_t^1, \ldots, \tilde{u}_t^N]$ where $\tilde{u}_t^n$ is a *confused* version of u_t with confidence score $p(\tilde{u}_t^n|u_t)$. Thus, an error model is needed as well as a user model. The model for $p(u_t|\ldots)$ should match the statistics of user responses in available corpora and the error model should match the characteristics of the speech recognition and understanding system [74]–[77].

A. User Simulation Models

One of the earliest approaches to user simulation used N-grams to model (26) directly [78]–[80]. The problem with this approach is that large N is needed to capture context and ensure that the user conveys consistent behavior with the result that the model is inevitably undertrained. A response to this problem was to design simulators that were essentially deterministic and goal directed but which had trainable random variables wherever there is genuine user choice [81]–[87]. These systems can work well but, in practice, a large amount of handcrafting is required to achieve acceptable performance.

More recent approaches to user simulation have focused on the use of dynamic Bayesian networks and hidden Markov models paralleling the user model in the POMDP itself [88]–[90]. They can also incorporate explicit goals by using a Bayesian network to maintain the goal state of the user [91]. Bayesian network approaches have the advantage that they can model a rich set of conditional dependencies and can be trained on data, although once again data sparsity is a major problem. An alternative promising technique, which avoids the sparsity issues inherent in joint probability models uses conditional random fields [92]. These have the advantage that they can model very long sequences as features much more efficiently.

Ultimately, the most obvious approach to user simulation will be to train a POMDP-based dialog system to behave like a user. The simulator could then talk to the system, with the simulator and system each refining their individual policies to maximize reward. The principal barrier to doing this is the lack of an appropriately detailed reward function for the user side of the dialog. A solution to this might be to utilize inverse reinforcement learning to infer users' reward functions from real human–human dialogs [93], [94].

B. Error Simulation Models

User simulation was first introduced to train Markov decision process (MDP)-based dialog systems where only the single best output from the recognizer is used. In this case, the error model was required primarily to simulate automatic speech recognition errors [95]–[99]. In contrast, POMDP-based systems use the full distribution of recognized hypotheses at every turn. Hence, as noted at the start of this section, the error model for a POMDP system must generate not just a single confusion but a set of confusions, each with a probability consistent with the behavior of the actual recognizer. The information content of this distribution makes a significant difference to performance [100]. Hence, it is important to model it accurately.

To conclude this section, user simulation and error modeling are important practical tools for building complex statistical dialog systems for which training directly from real users would be impractical. Pragmatically, the approach works reasonably well but the results are highly sensitive to the simulator [101], [102]. POMDP policy optimization is extremely good at exploiting inconsistent behavior in a user simulator in order to increase its reward. Thus, it is very easy to obtain very high performance when training and testing on the same simulator, but then to find actual performance in field trials is poor. One way of mitigating against this is to train and test on different simulators [101], [103].

VI. DIALOG MODEL PARAMETER OPTIMIZATION

Referring back to Fig. 2, a complete POMDP-based dialog system is characterized by two sets of parameters: the dialog model $\mathcal{M}$ with parameters τ, which incorporates the user, observation, and transition probability distributions; and the policy model $\mathcal{P}$ with parameters θ. Most current POMDP development focuses on policy optimization and the θ parameters, whereas the dialog model and the τ parameters are frequently handcrafted. While handcrafting the dialog model may seem unsatisfactory, in many cases, the dialog designer will have a strong prior on where the parameters should be. For example, in many applications, it may be reasonable to assume that the user goal is constant throughout the dialog. In this case, the probability function $p(s_t|s_{t-1}, a_{t-1})$ becomes a delta function. Similarly, one might reasonably assume that the true user action cannot be a contradiction of the user goal, and the distribution over all actions that are not contradictions is uniform. Hence, a user action like "I want a Chinese restaurant" might have zero probability for goals where the food concept is not Chinese but uniform probability otherwise.

In some situations, it is also possible to annotate dialog corpora to provide complete knowledge of the states of the system. Simple frequency counts can then be used to provide maximum-likelihood estimates of key parameters [104]. In [105], maximum likelihood is used to estimate a bigram model of the type of user action while using a handcrafted model to give zero probability to user goals that contradict a given action. In this way, simple maximum likelihood can be used to tune relatively complex models.

However, user behavior and the noise distributions of the speech understanding components are complex and annotating dialog corpora of sufficient quantity and accuracy is

impractical. Hence, key state variables such as the user's goal must be treated as hidden, and model parameters must be estimated using inference techniques. Recent research is starting to address this problem of learning the dialog model parameters τ directly from dialog corpora using a variety of approaches.

A. Expectation Maximization

If the hidden state cannot be annotated, then one must turn to an approximate inference algorithm. Since the POMDP has an input–output hidden Markov model structure, the expectation–maximization (EM) algorithm is a sensible first choice. EM operates by using a fixed set of parameters to estimate marginal distributions of the hidden parameters in its first step. It then uses these marginals to reestimate the parameters of the model and repeats.

EM has been used to learn the user model for a small spoken dialog system with seven states [106]. However, this approach does not scale well when the number of states is significantly increased. The main problem is that marginals must be computed for every possible state, which becomes impractical for many applications. Syed and Williams [107] have shown how EM can be used to build models of the user behavior, although the method requires the user's goal in the dialog to remain constant. This is an unsuitable assumption for many real-world dialogs. Nevertheless, approaches based on EM have the major benefit that they do not require annotations of the dialog state.

B. Expectation Propagation

Another algorithm that has been applied to learning the parameters of a spoken dialog system is expectation propagation [108], [109]. This algorithm operates directly on a factored Bayesian network and simply extends the loopy belief propagation algorithm to handle continuous state spaces. This allows the parameters to be updated during the propagation step and means that all the conditional independence assumptions are used in simplifying the update. This approach has been used to learn the goal evolution parameters of a relatively large tourist information system [110]. The main advantage of this approach is that it requires no annotations of either the true semantics or the dialog state. This means that parameters can be improved with each dialog without any extra supervision.

C. Reinforcement Learning

All of the above approaches are motivated by the objective of designing a dialog model that captures the essential elements of the user's behavior. An alternative would be to view the dialog model as just another set of parameters that influence the total reward earned by the system. Hence, the task of the parameter optimization changes from an inference problem to a reinforcement learning task. Although this means that the parameters are no longer probabilities in the usual sense, the final aim of the system is to maximize its performance and this approach directly optimizes this metric.

One algorithm that has been proposed for this purpose is an extension of the NAC algorithm discussed in Section IV called by analogy natural belief critic (NBC) [111]. This algorithm replaces the parameters of the policy that would usually be learned, with the parameters of the user model, denoted τ. This is not entirely straightforward because the reward is not differentiable with respect to τ. However, if the τ parameters are fitted with Dirichlet priors, then the priors are differentiable. Hence, the priors can be optimized using natural gradient ascent, and the required τ parameters can then be obtained by sampling the optimized priors. The NBC algorithm can be further extended to optimize both policy and user model parameters at the same time. Experiments have shown that policies optimized by this joint natural actor and belief critic (NABC) algorithm outperform policies trained solely with the NAC algorithm [70].

VII. FAST TRAINING AND USER ADAPTATION

As noted in Section IV, policy optimization for a real-world spoken dialog system takes an order of $O(10^5)$ training dialogs. This is too large for the policy to be trained in direct interaction with human users and, hence, as discussed in Section V, policy optimization currently relies on interaction with user simulators. These simulators require a high level of sophistication and are not trivial to build. Moreover, the policies trained with simulated users are biased toward the particular behavior that these models incorporate. It is desirable, therefore, to speed up policy optimization so that policies can be trained with or adapted to real users.

A second issue with standard policy optimization techniques is that they do not provide a measure of certainty in the estimates. As dialog systems become more complex it will be important to convey to the user the system's confidence level. When the system is uncertain, then it needs to signal to the user that it is uncertain. More pragmatically, knowing the level of uncertainty in the various regions of belief action space will assist learning algorithms to explore parts of the space it is uncertain about instead of exploring randomly.

A final issue concerns the inherent approximation introduced by parameterizing the policy. As described in Section IV, typically a set of feature functions are chosen and then parameters are optimized with respect to the chosen features. To be effective, the feature functions must be carefully handcrafted, and even then the result will be suboptimal.

One method that has been recently applied to spoken dialog systems to address these issues is Gaussian-process-based reinforcement learning, which allows a policy model $\mathcal{P}$ to be defined and optimized in a nonparametric

manner.[4] A Gaussian process (GP) is a generative model of Bayesian inference that can be used for function regression [112]. A GP is specified by a mean and a kernel function, which defines prior function correlations and is crucial for obtaining good estimates with just a few observations.

As with the value iteration and Monte Carlo methods described in Section IV, the GP approach represents the policy via the Q-function [see (9)]. However, unlike those methods, it does not require discretization of belief space. Furthermore, given a suitable kernel, it can operate both in summary belief space and in the full master belief space.[5]

Current research on using GP for dialog systems is focused on the GP–SARSA algorithm [113], which models the Q-function as a zero mean Gaussian process

$$Q(b,a) \sim \mathcal{GP}(0, k((b,a),(b,a))) \tag{27}$$

where $k((b,a),(b,a))$ is a kernel representing the correlations in belief action space (b,a).

The posterior of the Gaussian process provides not only the mean estimate of the Q-function but also the variance, which gives an estimate of the uncertainty in the approximation. In online algorithms, this estimate of the uncertainly can be used for more efficient exploration, either via an active learning model or by defining a stochastic policy model, both of which have been shown to speed up the process of learning [114].

The GP–SARSA algorithm can be used to estimate the Q-function for real-world problems [115], [114]. It has been shown to optimize dialog policies faster than a standard reinforcement learning algorithm [116], and it has been used to successfully train a dialog policy in direct interaction with human users [117].

GP–SARSA is an online on-policy algorithm in that it learns by interaction, taking actions according to the same policy that it is optimizing. While on-policy methods guarantee that the overall reward acquired during the optimization process is maximized, off-policy methods guarantee that a particular policy behavior is followed during the process of optimization. Off-policy sample-efficient methods have been explored in the context of parametric optimization using the framework of Kalman temporal differences (KTD) [118]. Online off-policy learning typically uses a form of Q-learning that exploits the Bellman equation

$$Q(b,a) = E\left(r_{t+1} + \gamma \max_{a' \in \mathcal{A}} Q(b_{t+1}, a') | b_t = b, a_t = a \right) \tag{28}$$

whose maximization requires a nonlinear parametrization of the Q-function. The KTD-Q algorithm is a sample efficient algorithm that assumes a nonlinear parametrization of the Q-function and optimizes the parameters using Kalman filtering [119]. Similar to GP–SARSA, this algorithm also provides a measure of the uncertainty of the approximation that can be used for more efficient policy optimization [119]. This algorithm has been successfully applied to dialog policy optimization and shown to perform significantly faster than standard off-policy methods such as LSPI [44], [120].

VIII. SYSTEMS AND APPLICATIONS

The preceding sections of this review provide an outline of the core components of a statistical dialog system. It will be apparent that these systems are complex and the techniques are being continually refined. While commercial deployment is probably still some way off, several working systems have been built in a number of application areas. This section provides some specific examples of dialog systems and applications that have been implemented within the POMDP framework.

Most of the systems implemented to date have been information inquiry applications. These include: voice dialing [121], tourist information [28], [42], appointment scheduling [122], and car navigation [24]. Command-and-control applications have also been demonstrated, such as control of devices and services in the home via a multimodal interface [123]. Another class of application is the troubleshooting domain, which is an interesting example of planning for uncertainty where the observation space of the POMDP can be extended to include nonspeech variables such as the state of a modem, and the actions can include the invocation of test signals in addition to verbal responses [124].

POMDP systems have also been demonstrated in the "Let's Go" challenge run by Carnegie Mellon University (CMU, Pittsburgh, PA), which involves providing live out-of-hours bus information to residents in the Pittsburgh area [34], [125]. This application must deal with casual callers using a variety of mobile phones, often in noisy environments. When success rates were plotted as a function of word error rate, the POMDP-based systems consistently outperformed the conventional baseline system [8].

IX. EVALUATION AND PERFORMANCE

While methods for evaluation of most data-driven tasks in speech and natural language processing are well established [126], evaluating a spoken dialog system requires interaction with users and is, therefore, difficult [127]. Moreover, a spoken dialog system consists of distinct modules and, although there are well-defined evaluation metrics for most of them individually, joint evaluation of the complete system is challenging. Ultimately, the goal is

[4]Here nonparametric does not mean parameter free but rather that the choice of parameters does not restrict the solution.

[5]Hence, in this section, master and summary space are not distinguished.

usually to provide user satisfaction, but this is hard to measure and is anyway really only appropriate for real users with real needs. Testing a system by giving paid subjects artificial goals will inevitably create biases, and even then it is difficult to recruit sufficient subjects to achieve statistical significance for each contrast condition tested. Given these problems, evaluation of SDS typically falls into one of three testing regimes: testing with some form of simulated user, testing with paid subjects, and testing within a live deployed system.

There are no standardized measures for dialog manager evaluation. However, the paradigm for spoken dialog system evaluation (PARADISE) framework [128] suggests the general approach of predicting user satisfaction ratings using a corpus of dialogs annotated with user satisfaction ratings and a set of objective measures. It defines the dialog manager performance as a weighted function of the dialog success and dialog-based cost measures such as the dialog length or the number of times the system produced a confirmation. The weights can be inferred from the annotated corpus using regression, and this regression can then be used as the reward function in reinforcement learning [129], [130]. While this enables a large number of objective measures to be considered, in practice, the dialog success rate and the dialog length are typically the most important predictors. Hence, most statistical dialog systems are trained to maximize success while minimizing the length of the dialog.

The simplest and most efficient test regime is to use a simulated user. This enables wide coverage of the space of possible dialogs, with a variety of scenarios and the ability to vary the effective recognition error rates over a wide range [131]–[133]. An obvious disadvantage, of course, is the potential discrepancy between the behavior of a simulated user and the behavior of real users. Nevertheless, user simulation has been a very common way to evaluate different POMDP approaches [28], [124].

As an illustration, Fig. 6 reproduces the results from [28]. Three different dialog managers were evaluated with a simulated user at varying noise levels. These were a handcrafted deterministic dialog manager that takes only the most likely input from the simulated user (HDC), the BUDS dialog manager (Section III-B) with a hand-coded policy (BUDS–HDC), and the BUDS dialog manager with a policy trained using NAC (BUDS–TRA) (Section IV-E). The systems operate in a tourist information domain, where the users may ask about hotels, restaurants, bars, and amenities in a fictitious town. The reward function is defined as 20 for a successful dialog minus the dialog length.[6] The confusion rate is defined as the probability of a semantic concept from a hypothesis passed to the dialog manager being incorrect. A dialog is successful if a suitable

[6]This is equivalent to giving a reward of −1 in all nonterminal states and +20 in all terminal states which denote success.

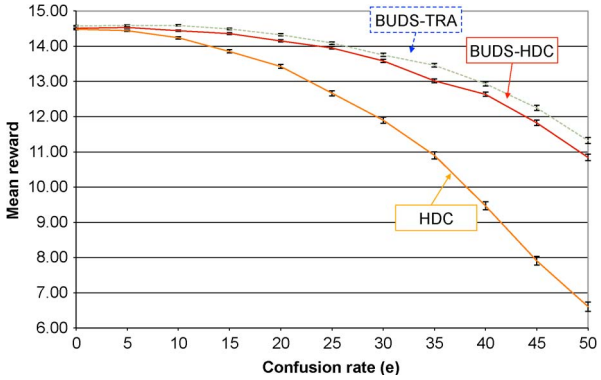


Fig. 6. ***Comparison between the BUDS POMDP-based dialog manager with a handcrafted policy (BUDS–HDC) and a trained policy (BUDS–TRA), and a conventional handcrafted deterministic dialog manager (HDC) as a function of simulated confusion rate. Each point gives the mean reward for 5000 simulated dialogs. Error bars show one standard error on each side of the mean. Taken from [28].***

venue was offered and all further pieces of information were given. The results in Fig. 6 highlight some important features of the different managers. First, at low error rates, all managers perform equally well, but at higher error rates, the BUDS systems are clearly more robust to noise. By using belief tracking, the BUDS dialog manager is better able to handle conflicting evidence from different dialog turns. Second, the BUDS manager with the trained policy gives further improvement compared to the BUDS system with a handcrafted policy, indicating that policy optimization via reinforcement learning is effective.

The second commonly used evaluation regime is to recruit human subjects to test the systems, with the aim of providing more realistic dialogs. Until recently, this has typically been done in a laboratory setting, where paid subjects are given predefined tasks, instructed to talk to the system in a particular way and then asked to rate the dialogs. For example, systems using the POMDP approach have been evaluated in this way, and improvements have been demonstrated relative to hand-coded or MDP baselines [28], [49], [134]. Setting aside the issue of how representative paid subjects are of real users, a major problem with this approach is the difficulty and expense of recruiting sufficient subjects. The recent emergence of crowdsourcing and the ability to transport audio with minimal latency over the Internet, has greatly mitigated this problem. For example, in [135], the Amazon Mechanical Turk service was used to recruit subjects and provide them with predefined tasks and basic instructions. Subjects then called the dialog system via a toll-free telephone number and provided feedback after each dialog. This now provides an effective way of collecting a large number of human–computer dialogs in order to achieve statistically significant results [136]. However, the use of crowdsourcing does not address the issue of how

representative paid subjects are of real users. Indeed, without the ability to monitor subjects in a laboratory setting, the problems of motivation and attention are exacerbated. Furthermore, there is evidence to suggest that crowdsourced subjects do not rate their dialog experience accurately [117].

The final evaluation regime is to test systems in the field with real users. This normally requires partnership with a commercial provider or the setting up of a service that the public are naturally motivated to use. Neither of these is easy to arrange and, as a response to this problem, CMU has set up an evaluation framework called the Spoken Dialog System Challenge. This utilizes a live service for providing bus information for Pittsburgh, PA, where users are able to call the system and ask for bus information. This service has been provided for some years using human agents during office hours, and by providing a fully automated out-of-hours service, a genuine user demand has been created which by default is serviced by a baseline spoken dialog system designed by CMU, but which can be substituted by other systems for testing. As an example, in the 2010 challenge, a number of systems were tested, first by a cohort of students to ensure that the systems were sufficiently robust to put before the public and then they were tested in the live service. Two of those tested were based on the POMDP framework.

The summary results are shown in Fig. 7, taken from [8], where the logistic regression of dialog success was computed against word error rate (WER) for each of the systems. It can be seen that the hand-coded baseline system "sys1" is slightly better at low error rates which was to be expected given the long period of development and refinement of that system. However despite the very short development times of the other two POMDP-based systems, both were more robust on high noise levels than the hand-coded baseline system.

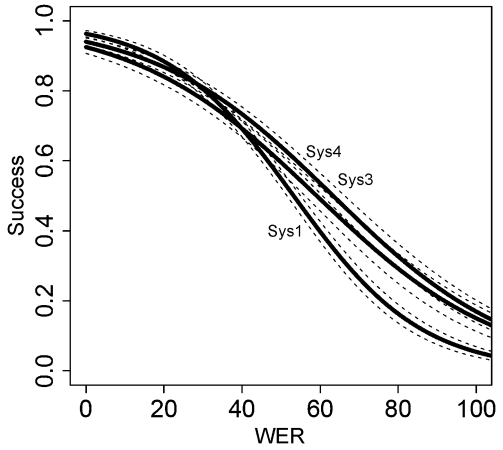


Fig. 7. *Logistic regression of dialog success versus word error rate (WER) for the three systems subjected to live testing in the 2010 Spoken Dialog Challenge. Sys3 and Sys4 were POMDP-based systems and Sys 1 was a hand-coded system. Taken from [8].*

X. HISTORICAL PERSPECTIVE

The first reference to the use of POMDPs in spoken dialog systems is thought to be Roy *et al.* in 2000 [9] and Zhang *et al.* in 2001 [137]. However, some of the key ideas were being explored before then. The idea of viewing dialog management as an observable Markov decision process (MDP) and optimizing a policy via reinforcement learning is due to Levin and Pieraccini [138], [139]. This work was quickly developed by others [45], [46], [48]–[51], [130], [140]–[151]. However, the use of MDP approaches lost momentum as it became apparent that the lack of any explicit model of uncertainty was limiting performance, and although the generalization to POMDPs was known, their use appeared to be intractable for real-world systems. The first reference to the need for an explicit representation of uncertainty in dialog is believed to be Pulman in 1996 who proposed modeling dialog as a *conversational game* [152]. Soon after, Heckerman and Horvitz [153], Horvitz and Paek [154], and Meng *et al.* [155] showed how Bayesian networks could be used to provide more robust conversational structures.

XI. CONCLUSION

The development of statistical dialog systems has been motivated by the need for a data-driven framework that reduces the cost of laboriously handcrafting complex dialog managers and which provides robustness against the errors created by speech recognizers operating in noisy environments. By providing an explicit Bayesian model of uncertainty and by providing a reward-driven process for policy optimization, POMDPs provide such a framework.

However, as will be clear from this review, POMDP-based dialog systems are complex and involve approximations and tradeoffs. Good progress has been made but there is still much to do. There are many challenges, most of which have been touched upon in this review such as finding ways to increase the complexity of the dialog model while maintaining tractable belief tracking; and reducing policy learning times so that systems can be trained directly on real users rather than using simulators. Down the road, there is also the task of packaging this technology to make it widely accessible to nonexperts in the industrial community.

There are other challenges as well. The POMDP framework depends critically on the notion of rewards. In principle, this is a key benefit of the approach since it provides an objective mechanism for specifying dialog design criteria. However, the problem in practice is that it is very difficult to extract reliable reward signals from users. Even the simple success/fail criterion is difficult to

compute since even if asked "Did the system give you all that you asked for?", many users will say "Yes" regardless just to be polite, or "No" because they had completely unrealistic expectations of what the system can do for them. Reward functions based on user satisfaction as predicted by a regression on objectively measurable features such as in the PARADISE framework [128] may mitigate this problem and need to be explored further [129], [130]. However, much of the experience to date suggests that online learning with real users will not be truly effective until robust biometric technology is available that enables the emotional state of the user to be accurately measured [117]. In the meantime, there is no shortage of further areas to develop and explore. ■

REFERENCES

[1] R. De Mori, *Spoken Dialogues With Computers*, New York: Academic, 1998.

[2] R. Smith and D. Hipp, *Spoken Natural Language Dialog Systems: A Practical Approach*. Oxford, U.K.: Oxford Univ. Press, 1994.

[3] V. Zue and J. Glass, "Conversational interfaces: Advances and challenges," *Proc. IEEE*, vol. 88, no. 8, pp. 1166–1180, Aug. 2000.

[4] S. Young, "Cognitive user interfaces," *IEEE Signal Process. Mag.*, vol. 27, no. 3, pp. 128–140, May 2010.

[5] M. Oshry, P. Baggia, K. Rehor, M. Young, R. Akolkar, X. Yang, J. Barnett, R. Hosn, R. Auburn, J. Carter, S. McGlashan, M. Bodell, and D. C. Burnett, *Voice extensible markup language (VoiceXML) 3.0*, W3C, W3C Working Draft, Dec. 2009. [Online]. Available: http://www.w3.org/TR/2009/WD-voicexml30-20091203/

[6] T. Paek and R. Pieraccini, "Automating spoken dialogue management design using machine learning: An industry perspective," *Speech Commun.*, vol. 50, no. 8–9, pp. 716–729, 2008.

[7] R. P. Lippmann, "Speech recognition by machines and humans," *Speech Commun.*, vol. 22, no. 1, pp. 1–15, 1997.

[8] A. Black, S. Burger, A. Conkie, H. Hastie, S. Keizer, O. Lemon, N. Merigaud, G. Parent, G. Schubiner, B. Thomson, J. Williams, K. Yu, S. Young, and M. Eskenazi, "Spoken dialog challenge 2010: Comparison of live and control test results," in *Proc. Annu. Meeting Special Interest Group Discourse Dialogue*, Portland, OR, 2011, pp. 2–7.

[9] N. Roy, J. Pineau, and S. Thrun, "Spoken dialogue management using probabilistic reasoning," in *Proc. Annu. Meeting Assoc. Comput. Linguist.*, Hong Kong, 2000, pp. 93–100.

[10] S. Young, "Talking to machines (statistically speaking)," in *International Conf Spoken Language Processing*, Denver, CO, 2002, pp. 9–15.

[11] J. Williams and S. Young, "Partially observable Markov decision processes for spoken dialog systems," *Comput. Speech Lang.*, vol. 21, no. 2, pp. 393–422, 2007.

[12] L. Kaelbling, M. Littman, and A. Cassandra, "Planning and acting in partially observable stochastic domains," *Artif. Intell.*, vol. 101, pp. 99–134, 1998.

[13] E. Hansen, "Solving POMDPs by searching in policy space," in *Proc. Conf. Uncertainty Artif. Intell.*, Madison, WI, 1998, pp. 211–219.

[14] M. L. Littman, R. S. Sutton, and S. Singh, "Predictive representations of state *Advances in Neural Information Processing Systems 14*. Cambridge, MA: MIT Press, 2002, pp. 1555–1561.

[15] R. E. Bellman, *Dynamic Programming*. Princeton, NJ: Princeton Univ. Press, 1957.

[16] J. Pineau, G. Gordon, and S. Thrun, "Point-based value iteration: An anytime algorithm for POMDPs," in *Proc. Int. Joint Conf. Artif. Intell.*, Acapulco, Mexico, 2003, pp. 1025–1032.

[17] T. Smith and R. G. Simmons, "Heuristic search value iteration for POMDPs," in *Proc. Conf. Uncertainty Artif. Intell.*, Banff, AB, Canada, 2004, pp. 520–527.

[18] M. Spaan and N. Vlassis, "Perseus: Randomized point-based value iteration for POMDPs," *J. Artif. Intell. Res.*, vol. 24, pp. 195–220, 2005.

[19] T. Smith and R. Simmons, "Point-based POMDP algorithms: Improved analysis and implementation," in *Proc. 21st Annu. Conf. Uncertainty Artif. Intell.*, Arlington, VA, 2005, pp. 542–549.

[20] S. Ji, R. Parr, H. Li, X. Liao, and L. Carin, "Point-based policy iteration," in *Proc. Nat. Conf. Artif. Intell.*, Vancouver, BC, Canada, 2007, pp. 1243–1249.

[21] J. Williams, P. Poupart, and S. Young, "Factored partially observable Markov decision processes for dialogue management," in *Proc. IJCAI Workshop Knowl. Reason. Practical Dialog Syst.*, Edinburgh, U.K., 2005, pp. 76–82.

[22] T. Bui, M. Poel, A. Nijholt, and J. Zwiers, "A tractable hybrid DDN-POMDP approach to affective dialogue modeling for probabilistic frame-based dialogue systems," *Natural Lang. Eng.*, vol. 15, no. 2, pp. 273–307, 2009.

[23] S. Young, J. Schatzmann, K. Weilhammer, and H. Ye, "The hidden information state approach to dialog management," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Honolulu, HI, 2007, pp. 149–152.

[24] K. Kim, C. Lee, S. Jung, and G. Lee, "A frame-based probabilistic framework for spoken dialog management using dialog examples," in *Proc. Special Interest Group Discourse Dialogue (SIGDIAL) Workshop Discourse Dialogue*, Columbus, OH, 2008, pp. 120–127.

[25] J. Williams, "Incremental partition recombination for efficient tracking of multiple dialogue states," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Dallas, TX, 2010, pp. 5382–5385.

[26] M. Gašić and S. Young, "Effective handling of dialogue state in the hidden information state POMDP dialogue manager," *ACM Trans. Speech Lang. Process.*, vol. 7, no. 3, 2011, Article 4.

[27] B. Thomson, J. Schatzmann, and S. Young, "Bayesian update of dialogue state for robust dialogue systems," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Las Vegas, NV, 2008, pp. 4937–4940.

[28] B. Thomson and S. Young, "Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems," *Comput. Speech Lang.*, vol. 24, no. 4, pp. 562–588, 2010.

[29] J. Henderson and O. Lemon, "Mixture model POMDPs for efficient handling of uncertainty in dialogue management," in *Proc. Annu. Meeting Assoc. Comput. Linguist.*, Columbus, OH, 2008, pp. 73–76.

[30] D. Bohus and A. Rudnicky, "A 'K hypotheses + other' belief updating model," in *Proc. AAAI Workshop Stat. Empirical Approaches Spoken Dialogue Syst.*, Boston, MA, 2006, pp. 13–18.

[31] R. Higashinaka, M. Nakano, and K. Aikawa, "Corpus-based discourse understanding in spoken dialogue systems," in *Proc. Annu. Meeting Assoc. Comput. Linguist.*, Sapporo, Japan, 2003, pp. 240–247.

[32] C. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer-Verlag, 2006.

[33] J. D. Williams, "Using particle filters to track dialogue state," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, Kyoto, Japan, 2007, pp. 502–507.

[34] B. Thomson, K. Yu, S. Keizer, M. Gasic, F. Jurcicek, F. Mairesse, and S. Young, "Bayesian dialogue system for the let's go spoken dialogue challenge," in *Proc. IEEE Spoken Lang. Technol. Workshop*, Berkeley, CA, 2010, pp. 460–465.

[35] P. A. Crook and O. Lemon, "Lossless value directed compression of complex user goal states for statistical spoken dialogue systems," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Florence, Italy, 2011, pp. 1029–1032.

[36] F. Doshi and N. Roy, "The permutable POMDP: Fast solutions to POMDPs for preference elicitation," in *Proc. Int. Joint Conf. Autonom. Agents Multiagent Syst.*, 2008, pp. 493–500.

[37] J. Williams and S. Young, "Scaling up POMDPs for dialog management: The 'summary POMDP' method," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, Cancun, Mexico, 2005, pp. 177–182.

[38] J. Williams and S. Young, "Scaling POMDPs for spoken dialog management," *IEEE Trans. Audio Speech Lang. Process.*, vol. 15, no. 7, pp. 2116–2129, Jul. 2007.

[39] D. Andre and S. Russell, "State abstraction for programmable reinforcement learning agents," in *Proc. 18th Nat. Conf. Artif. Intell.*, Edmonton, AB, Canada, 2002, pp. 119–125.

[40] J. D. Williams, "The best of both worlds: Unifying conventional dialog systems and POMDPs," in *Proc. Annu. Conf. Int.*

Speech Commun. Assoc., Brisbane, Australia, 2008, pp. 1173–1176.

[41] P. Lison, "Towards relational POMDPs for adaptive dialogue management," in *Proc. Annu. Meeting Assoc. Comput. Linguist.*, Uppsala, Sweden, 2010, pp. 7–12.

[42] S. Young, M. Gašić, S. Keizer, F. Mairesse, J. Schatzmann, B. Thomson, and K. Yu, "The hidden information state model: A practical framework for POMDP-based spoken dialogue management," *Comput. Speech Lang.*, vol. 24, no. 2, pp. 150–174, 2010.

[43] L. Li, J. D. Williams, and S. Balakrishnan, "Reinforcement learning for dialog management using least-squares policy iteration and fast feature selection," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Brighton, U.K., 2009, pp. 2475–2478.

[44] O. Pietquin, M. Geist, S. Chandramohan, and H. Frezza-Buet, "Sample-efficient batch reinforcement learning for dialogue management optimization," *ACM Trans. Speech Lang. Process.*, vol. 7, no. 3, pp. 1–21, 2011.

[45] K. Scheffler and S. Young, "Automatic learning of dialogue strategy using dialogue simulation and reinforcement learning," in *Proc. 2nd Int. Conf. Human Lang. Technol. Res.*, San Diego, CA, 2002, pp. 12–19.

[46] H. Cuayáhuitl, S. Renals, O. Lemon, and H. Shimodaira, "Reinforcement learning of dialogue strategies with hierarchical abstract machines," in *Proc. IEEE Spoken Lang. Technol. Workshop*, Aruba, 2006, pp. 182–185.

[47] H. Cuayáhuitl, S. Renals, and O. Lemon, "Learning multi-goal dialogue strategies using reinforcement learning with reduced state-action spaces," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Pittsburgh, PA, 2006, pp. 547–565.

[48] O. Pietquin and T. Dutoit, "A probabilistic framework for dialog simulation and optimal strategy learning," *IEEE Trans. Audio Speech Lang. Process.*, vol. 14, no. 2, pp. 589–599, Mar. 2006.

[49] J. Henderson, O. Lemon, and K. Georgila, "Hybrid reinforcement/supervised learning for dialogue policies from communicator data," in *Proc. IJCAI Workshop Knowl. Reason. Practical Dialog Syst.*, Edinburgh, U.K., 2005, pp. 68–75.

[50] M. Frampton and O. Lemon, "Learning more effective dialogue strategies using limited dialogue move features," in *Proc. Annu. Meeting Assoc. Comput. Linguist.*, Sydney, Australia, 2006, pp. 185–192.

[51] J. Henderson, O. Lemon, and K. Georgila, "Hybrid reinforcement/supervised learning of dialogue policies from fixed datasets," *Comput. Linguist.*, vol. 34, no. 4, pp. 487–511, 2008.

[52] H. Kurniawati, D. Hsu, and W. Lee, "SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces," in *Proc. Robot., Sci. Syst.*, 2008. [Online]. Available: http://www.roboticsproceedings.org/rss04/

[53] T. H. Bui, M. Poel, A. Nijholt, and J. Zwiers, "A tractable DDN-POMDP approach to affective dialogue modeling for general probabilistic frame-based dialogue systems," in *Proc. IJCAI Workshop Knowl. Reason. Practical Dialog Syst.*, Hyderabad, India, 2007, pp. 34–37.

[54] T. H. Bui, B. W. van Schooten, and D. H. W. Hofs, "Practical dialogue manager development using POMDPs," in *Proc. Annu. Meeting Special Interest Group Discourse Dialogue*, Antwerp, Belgium, 2007, pp. 215–218.

[55] F. Pinault, F. Lefevre, and R. D. Mori, "Feature-based summary space for stochastic dialogue modeling with hierarchical semantic frames," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Brighton, U.K., 2009, pp. 284–287.

[56] A. Boularias, H. R. Chinaei, and B. Chaib-draa, "Learning the reward model of dialogue POMDPs from data," in *Proc. NIPS Workshop Mach. Learn. Assistive Tech.*, 2010.

[57] F. Pinault and F. Lefevre, "Semantic graph clustering for POMDP-based spoken dialog systems," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Florence, Italy, 2011, pp. 1321–1324.

[58] F. Pinault and F. Lefevre, "Unsupervised clustering of probability distributions of semantic frame graphs for pomdp-based spoken dialogue systems with summary space," in *Proc. IJCAI Workshop Knowl. Reason. Practical Dialog Syst.*, Barcelona, Spain, 2011.

[59] H. R. Chinaei and B. Chaib-draa, "Learning dialogue POMDP models from data," in *Proc. Can. Conf. Artif. Intell.*, 2011, pp. 86–91.

[60] H. R. Chinaei, B. Chaib-draa, and L. Lamontagne, "Learning observation models for dialogue POMDPs," in *Proc. Can. Conf. Artif. Intell.*, 2012.

[61] S. Varges, G. Riccardi, S. Quarteroni, and A. Ivanov, "POMDP concept policies and task structures for hybrid dialog management," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Prague, Czech Republic, 2011, pp. 5592–5595.

[62] J. D. Williams, "Integrating expert knowledge into POMDP optimization for spoken dialog systems," in *Proc. AAAI Workshop Adv. POMDP Solvers*, Chicago, Illinois, 2008.

[63] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998, ser. Adaptive Computation and Machine Learning.

[64] F. Lefevre, M. Gasic, F. Jurcicek, S. Keizer, F. Mairesse, B. Thomson, K. Yu, and S. J. Young, "*k*-nearest neighbor Monte-Carlo control algorithm for POMDP-based dialogue systems," in *Proc. Annu. Meeting Special Interest Group Discourse Dialogue*, London, U.K., 2009.

[65] M. G. Lagoudakis and R. Parr, "Least-squares policy iteration," *J. Mach. Learn. Res.*, vol. 4, pp. 1107–1149, 2003.

[66] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, pp. 229–256, 1992.

[67] S. Amari, "Natural gradient works efficiently in learning," *Neural Comput.*, vol. 10, no. 2, pp. 251–276, 1998.

[68] J. Peters and S. Schaal, "Natural actor-critic," *Neurocomputing*, vol. 71, no. 7–9, pp. 1180–1190, 2008.

[69] T. Misu, "Dialogue strategy optimization to assist user's decision for spoken consulting dialogue systems," in *Proc. IEEE Spoken Lang. Technol. Workshop*, Berkeley, CA, 2010, pp. 354–359.

[70] F. Jurčíček, B. Thomson, and S. Young, "Natural actor and belief critic: Reinforcement algorithm for learning parameters of dialogue systems modelled as POMDPs," *ACM Trans. Speech Lang. Process.*, vol. 7, no. 3, 2011, Article 6.

[71] T. Misu, K. Sugiura, K. Ohtake, and C. Hori, "Modeling spoken decision making dialogue and optimization of its dialogue strategy," in *Proc. Annu. Meeting Special Interest Group Discourse Dialogue*, Portland, OR, 2011, pp. 221–224.

[72] T. Misu, K. Georgila, A. Leuski, and D. Traum, "Reinforcement learning of question-answering dialogue policies for virtual museum guides," in *Proc. Annu. Meeting Special Interest Group Discourse Dialogue*, Seoul, Korea, 2012, pp. 84–93.

[73] J. Schatzmann, K. Weilhammer, M. Stuttle, and S. Young, "A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies," *Knowl. Eng. Rev.*, vol. 21, no. 2, pp. 97–126, Jun. 2006.

[74] H. Hastie, "Metrics and evaluation of spoken dialogue systems," in *Data-Driven Methods for Adaptive Spoken Dialogue Systems: Computational Learning for Conversational Interfaces*. New York: Springer-Verlag, 2012.

[75] O. Pietquin and H. Hastie, "A survey on metrics for the evaluation of user simulations," *Knowl. Eng. Rev.*, 2012, DOI:http://dx.doi.org/10.1017/S0269888912000343

[76] J. Schatzmann, K. Georgila, and S. Young, "quantitative evaluation of user simulation techniques for spoken dialogue systems," in *Proc. Annu. Meeting Special Interest Group Discourse Dialogue*, Lisbon, Portugal, 2005, pp. 45–54.

[77] J. Williams, "Evaluating user simulations with the Cramervon Mises divergence," *Speech Commun.*, vol. 50, pp. 829–846, 2008.

[78] W. Eckert, E. Levin, and R. Pieraccini, "User modelling for spoken dialogue system evaluation," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, Santa Barbara, CA, 1997, pp. 80–87.

[79] K. Georgila, J. Henderson, and O. Lemon, "Learning user simulations for information state update dialogue systems," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Lisbon, Portugal, 2005, pp. 893–896.

[80] K. Georgila, J. Henderson, and O. Lemon, "User simulation for spoken dialogue systems: Learning and evaluation," in *Proc. Int. Conf. Spoken Lang. Process.*, Pittsburgh, PA, 2006, pp. 45–54.

[81] K. Scheffler and S. Young, "Probabilistic simulation of human-machine dialogues," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Istanbul, Turkey, 2000, pp. 1217–1220.

[82] V. Rieser and O. Lemon, "Cluster-based user simulations for learning dialogue strategies," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Pittsburgh, PA, 2006.

[83] O. Pietquin, "Consistent goal-directed user model for realistic man-machine task-oriented spoken dialogue simulation," in *Proc. IEEE Int. Conf. Multimedia Expo*, Toronto, ON, Canada, 2006, pp. 425–428.

[84] J. Schatzmann, B. Thomson, K. Weilhammer, H. Ye, and S. Young,

"Agenda-based user simulation for bootstrapping a POMDP dialogue system," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist. Human Lang. Technol.*, Rochester, NY, 2007, pp. 149–152.

[85] H. Ai and D. J. Litman, "Knowledge consistent user simulations for dialog systems," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Antwerp, Belgium, 2007, pp. 2697–2700.

[86] J. Schatzmann and S. Young, "The hidden agenda user simulation model," *IEEE Trans. Audio Speech Lang. Process.*, vol. 17, no. 4, pp. 733–747, May 2009.

[87] S. Keizer, M. Gasic, F. Jurcicek, F. Mairesse, B. Thomson, K. Yu, and S. Young, "Parameter estimation for agenda-based user simulation," in *Proc. Annu. Meeting Special Interest Group Discourse Dialogue*, Tokyo, Japan, 2010, pp. 116–123.

[88] O. Pietquin, *A Framework for Unsupervised Learning of Dialogue Strategies*, Louvain, Belgium: Presses Universitaires de Louvain, 2004.

[89] H. Cuayahuitl, S. Renals, O. Lemon, and H. Shimodaira, "Human-computer dialogue simulation using hidden Markov models," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, Puerto Rico, 2005, pp. 290–295.

[90] O. Pietquin and T. Dutoit, "A probabilistic framework for dialog simulation and optimal strategy learning," *IEEE Trans. Audio Speech Lang. Process.*, vol. 14, no. 2, pp. 589–599, Mar. 2006.

[91] S. Rossignol, M. Ianotto, and O. Pietquin, "Training a BN-based user model for dialogue simulation with missing data," in *Proc. Int. Joint Conf. Natural Lang. Process.*, Chiang Mai, Thailand, Nov. 2011, pp. 598–604.

[92] S. Jung, C. Lee, K. Kim, M. Jeong, and G. G. Lee, "Data-driven user simulation for automated evaluation of spoken dialog systems," *Comput. Speech Lang.*, vol. 23, no. 4, pp. 479–509, 2009.

[93] A. Ng and S. Russell, "Algorithms for inverse reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, Stanford, CA, 2000, pp. 663–670.

[94] S. Chandramohan, M. Geist, F. Lefevre, and O. Pietquin, "User simulation in dialogue systems using inverse reinforcement learning," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Florence, Italy, 2011, pp. 1025–1028.

[95] O. Pietquin and S. Renals, "ASR system modelling for automatic evaluation and optimisation of dialogue systems," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Orlando, FL, 2002, pp. 46–49.

[96] O. Pietquin and R. Beaufort, "Comparing ASR modeling methods for spoken dialogue simulation and optimal strategy learning," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Lisbon, Portugal, 2005, pp. 861–864.

[97] O. Pietquin and T. Dutoit, "Dynamic Bayesian networks for NLU simulation with application to dialog optimal strategy learning," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Toulouse, France, 2006, pp. 49–52.

[98] J. Schatzmann, B. Thomson, and S. Young, "Error simulation for training statistical dialogue systems," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, Kyoto, Japan, 2007, pp. 526–531.

[99] F. García, L. F. Hurtado, D. Griol, M. J. Castro, E. Segarra, and E. Sanchis, "Recognition and understanding simulation for a spoken dialog corpus acquisition," *Proc. Text Speech Dialogue*, pp. 574–581, 2007.

[100] B. Thomson, K. Yu, M. Gasic, S. Keizer, F. Mairesse, J. Schatzmann, and S. Young, "Evaluating semantic-level confidence scores with multiple hypotheses," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Brisbane, Australia, 2008, pp. 1153–1156.

[101] J. Schatzmann, K. Weilhammer, M. Stuttle, and S. Young, "Effects of the user model on simulation-based learning of dialogue strategies," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, Cancun, Mexico, 2005, pp. 220–225.

[102] D. L. Hua Ai and J. Tetreault, "Comparing user simulation models for dialog strategy learning," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.*, Rochester, NY, 2007, pp. 1–4.

[103] D. Kim, J. Kim, and K.-E. Kim, "Robust performance evaluation of POMDP-based dialogue systems," *IEEE Trans. Audio Speech Lang. Process.*, vol. 19, no. 4, pp. 1029–1040, May 2011.

[104] J. D. Williams, "Exploiting the ASR N-best by tracking multiple dialog state hypotheses," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Brisbane, Australia, 2008, pp. 191–194.

[105] S. Keizer, M. Gašić, F. Mairesse, B. Thomson, K. Yu, and S. Young, "Modelling user behaviour in the HIS-POMDP dialogue manager," in *Proc. IEEE Spoken Lang. Technol. Workshop*, Goa, India, 2008, pp. 121–124.

[106] F. Doshi and N. Roy, "Spoken language interaction with model uncertainty: An adaptive human-robot interaction system," *Connection Sci.*, vol. 20, no. 4, pp. 299–318, 2008.

[107] U. Syed and J. D. Williams, "Using automatically transcribed dialogs to learn user models in a spoken dialog system," *Proc. Annu. Meeting Assoc. Comput. Linguist.*, pp. 121–124, 2008.

[108] T. Minka, "Expectation propagation for approximate Bayesian inference," in *Proc. Conf. Uncertainty Artif. Intell.*, Seattle, WA, 2001, pp. 362–369.

[109] B. Thomson, F. Jurčíček, M. Gašić, S. Keizer, F. Mairesse, K. Yu, and S. Young, "Parameter learning for POMDP spoken dialogue models," in *Proc. IEEE Spoken Lang. Technol. Workshop*, Berkeley, CA, 2010, pp. 271–276.

[110] B. Thomson, "Statistical methods for spoken dialogue management," Ph.D. dissertation, Engineering Department, Cambridge Univ., Cambridge, U.K., 2009.

[111] F. Jurčíček, B. Thomson, S. Keizer, F. Mairesse, M. Gašić, K. Yu, and S. Young, "Natural belief-critic: A reinforcement algorithm for parameter estimation in statistical spoken dialogue systems," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Tokyo, Japan, 2010, pp. 90–93.

[112] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press, 2006.

[113] E. Engel, S. Mannor, and R. Meir, "Reinforcement learning with Gaussian processes," in *Proc. Int. Conf. Mach. Learn.*, Bonn, Germany, 2005, pp. 201–208.

[114] M. Gašić, "Statistical dialogue modelling," Ph.D. dissertation, Engineering Department, Univ. Cambridge, Cambridge, U.K., 2011.

[115] Y. Engel, "Algorithms and representations for reinforcement learning," Ph.D. dissertation, Engineering and Computer Science, Hebrew Univ., Jerusalem, Israel, 2005.

[116] M. Gašić, F. Jurčíček, S. Keizer, F. Mairesse, B. Thomson, K. Yu, and S. Young, "Gaussian processes for fast policy optimisation of a POMDP dialogue manager for a real-world task," in *Proc. Annu. Meeting Special Interest Group Discourse Dialogue*, Tokyo, Japan, 2010, pp. 201–204.

[117] M. Gašić, F. Jurčíček, B. Thomson, K. Yu, and S. Young, "On-line policy optimisation of spoken dialogue systems via live interaction with human subjects," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, 2011, pp. 312–317.

[118] M. Geist, O. Pietquin, and G. Fricout, "Kalman temporal differences: The deterministic case," in *Proc. IEEE Symp. Adaptive Dyn. Programm. Reinforcement Learn.*, 2009, pp. 185–192.

[119] M. Geist and O. Pietquin, "Managing uncertainty within the KTD framework," in *Proc. Workshop Active Learn. Exp. Design*, 2011, pp. 157–168.

[120] O. Pietquin, M. Geist, and S. Chandramohan, "Sample efficient on-line learning of optimal dialogue policies with Kalman temporal differences," in *Proc. Int. Joint Conf. Artif. Intell.*, Barcelona, Spain, 2011, pp. 1878–1883.

[121] J. Williams, "Demonstration of a POMDP voice dialer," presented at the Annu. Meeting Assoc. Comput. Linguist., Demonstration Session, Columbus, Ohio, 2008.

[122] S. Janarthanam, H. Hastie, O. Lemon, and X. Liu, "'The day after the day after tomorrow?' A machine learning approach to adaptive temporal expression generation: Training and evaluation with real users," in *Proc. Annu. Meeting Special Interest Group Discourse Dialogue*, Portland, OR, 2011, pp. 142–151.

[123] K. Kim and G. G. Lee, "Multimodal dialog system using hidden information state dialog manager," in *Proc. Int. Conf. Multimodal Interfaces Demonstration Session*, Nagoya, Japan, 2007.

[124] J. Williams, "Applying POMDPs to Dialog Systems in the Troubleshooting Domain," in *Proc. HLT/NAACL Workshop Bridging the Gap: Acad. Ind. Res. Dialog Technol.*, Rochester, NY, 2007, pp. 1–8.

[125] J. Williams, "An empirical evaluation of a statistical dialog system in public use," in *Proc. Annu. Meeting Special Interest Group Discourse Dialogue*, Portland, OR, 2011, pp. 130–141.

[126] L. Hirschman and H. S. Thompson, *Overview of Evaluation in Speech and Natural Language Processing*. New York: Cambridge Univ. Press, 1997, pp. 409–414.

[127] S. Gandhe and D. Traum, "An evaluation understudy for dialogue coherence models," in *Proc. Annu. Meeting Special Interest Group Discourse Dialogue*, Columbus, OH, 2008, pp. 172–181.

[128] M. Walker, D. Litman, C. Kamm, and A. Abella, "Paradise: A framework for evaluating spoken dialogue agents," in *Proc. Annu. Meeting Assoc. Comput. Linguist.*, Madrid, Spain, 1997, pp. 271–280.

[129] E. Levin, R. Pieraccini, and W. Eckert, "Learning dialogue strategies within the Markov decision process framework," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, Santa Barbara, CA, 1997, pp. 72–79.

[130] S. Singh, M. Kearns, D. Litman, and M. Walker, "Reinforcement learning for spoken dialogue systems," in *Proc. Neural Inf. Process. Syst.*, 1999.

[131] T. Watambe, M. Araki, and S. Doshita, "Evaluating dialogue strategies under communication errors using computer-to-computer simulation," *IEICE Trans. Inf. Syst.*, vol. E81-D, no. 9, pp. 1025–1033, 1998.

[132] H. Ai and F. Weng, "User simulation as testing for spoken dialog systems," in *Proc. Annu. Meeting Special Interest Group Discourse Dialogue*, Columbus, OH, 2008, pp. 164–171.

[133] J. Schatzmann, "Statistical user and error modelling for spoken dialogue systems," Ph.D. dissertation, Engineering Department, Univ. Cambridge, Cambridge, U.K., 2008.

[134] M. Gašić, F. Lefevre, F. Jurčíček, S. Keizer, F. Mairesse, B. Thomson, K. Yu, and S. Young, "Back-off action selection in summary space-based POMDP dialogue systems," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, Merano, Italy, 2009, pp. 456–461.

[135] F. Jurčíček, S. Keizer, M. Gašić, F. Mairesse, B. Thomson, K. Yu, and S. Young, "Real user evaluation of spoken dialogue systems using amazon mechanical turk," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Florence, Italy, 2011, pp. 3061–3064.

[136] I. McGraw, C. Lee, L. Hetherington, S. Seneff, and J. Glass, "Collecting voices from the cloud," in *Proc. Int. Conf. Lang. Resources Eval.*, Malta, 2010, pp. 1576–1583.

[137] B. Zhang, Q. Cai, J. Mao, E. Chang, and B. Guo, "Spoken dialogue management as planning and acting under uncertainty," in *Proc. EUROSPEECH*, Aalborg, Denmark, 2001, pp. 2169–2172.

[138] E. Levin and R. Pieraccini, "A stochastic model of computer-human interaction for learning dialogue strategies," in *Proc. EUROSPEECH*, Rhodes, Greece, 1997, pp. 1883–1886.

[139] E. Levin, R. Pieraccini, and W. Eckert, "Using Markov decision processes for learning dialogue strategies," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Seattle, WA, 1998, pp. 201–204.

[140] M. Walker, J. Fromer, and S. Narayanan, "Learning optimal dialogue strategies: A case study of a spoken dialogue agent for email," in *Proc. Annu. Meeting Assoc. Comput. Linguist./COLING*, Montreal, QC, Canada, 1998, pp. 1345–1351.

[141] M. Kearns, D. Litman, and M. Walker, "Automatic detection of poor speech recognition at the dialogue level," in *Proc. Annu. Meeting Assoc. Comput. Linguist.*, College Park, MD, 1999, pp. 309–316.

[142] D. Goddeau and J. Pineau, "Fast reinforcement learning of dialog strategies," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Istanbul, Turkey, 2000, pp. 1233–1236.

[143] S. Young, "Probabilistic methods in spoken dialogue systems," *Philosoph. Trans. Roy. Soc. A*, vol. 358, no. 1769, pp. 1389–1402, 2000.

[144] S. Singh, D. Litman, M. Kearns, and M. Walker, "Optimizing dialogue management with reinforcement learning: Experiments with the NJFun system," *J. Artif. Intell. Res.*, vol. 16, pp. 105–133, 2002.

[145] M. Denecke, K. Dohsaka, and M. Nakano, "Learning dialogue policies using state aggregation in reinforcement learning," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Jeju Island, Korea, 2004, pp. 325–328.

[146] M. Denecke, K. Dohsaka, and M. Nakano, "Fast reinforcement learning of dialogue policies using stable function approximation," in *Proc. Int. Joint Conf. Natural Lang. Process.*, 2005, DOI: 10.1007/978-3-540-30211-7_1.

[147] O. Pietquin, "A probabilistic description of man-machine spoken communication," in *Proc. IEEE Int. Conf. Multimedia Expo*, Amsterdam, The Netherlands, 2005, pp. 410–413.

[148] J. Tetreault and D. Litman, "Comparing the utility of state features in spoken dialogue using reinforcement learning," in *Proc. Conf. Human Lang. Technol. North Amer. Chapter Assoc. Comput. Linguist.*, New York, 2006, pp. 272–279.

[149] J. Tetreault and D. Litman, "Using reinforcement learning to build a better model of dialogue state," in *Proc. European Association Computational Linguistics*, Trento, Italy, 2006.

[150] P. Heeman, "Combining reinforcement learning with information-state update rules," in *Proc. Conf. Human Lang. Technol. North Amer. Chapter Assoc. Comput. Linguist.*, Rochester, NY, 2007, pp. 268–275.

[151] O. Pietquin, "Learning to ground in spoken dialogue systems," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Honolulu, HI, 2007, pp. 165–168.

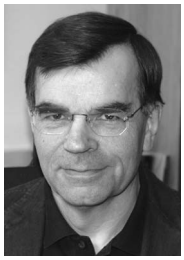
[152] S. Pulman, "Conversational games, belief revision and Bayesian networks," in *Proc. CLIN VII: 7th Comput. Linguist. Meeting*, The Netherlands, 1996.

[153] D. Heckerman and E. Horwitz, "Inferring informational goals from free-text queries: A Bayesian approach," in *Proc. Conf. Uncertainty Artif. Intell.*, Madison, WI, 1998, pp. 230–238.

[154] E. Horvitz and T. Paek, "A Computational architecture for conversation," in *Proc. 7th Int. Conf. User Model.*, Banff, AB, Canada, 1999, pp. 201–210.

[155] H. Meng, C. Wai, and R. Pieraccini, "The use of belief networks for Mixed-Initiative dialog modeling," *IEEE Trans. Speech Audio Process.*, vol. 11, no. 6, pp. 757–773, Nov. 2003.

ABOUT THE AUTHORS



Steve Young (Fellow, IEEE) received the B.A. degree in electrical sciences and the Ph.D. degree in speech processing from Cambridge University, Cambridge, U.K., in 1973 and 1978, respectively.

He was elected to the Chair of Information Engineering at Cambridge University in 1994. He was a co-founder and Technical Director of Entropic Ltd. from 1995 until 1999 when the company was taken over by Microsoft. After a short period as an Architect at Microsoft, he returned full-time to Cambridge University in January 2001 where he was Head of Information Engineering until 2009 and is now Senior Pro-Vice-Chancellor. His research interests include speech recognition, expressive synthesis, and spoken dialog systems. He has written and edited books on software engineering and speech processing, and he has published as author and coauthor, more than 200 papers in these areas.

Dr. Young is a Fellow of the U.K. Royal Academy of Engineering, the Institution of Engineering and Technology (IET), and Royal Society for the Encouragement of Arts, Manufactures and Commerce (RSA). He served as the Senior Editor of *Computer Speech and Language* from 1993 to 2004, and he was Chair of the IEEE Speech and Language Technical Committee from 2008 to 2010. He was the recipient of an IEEE Signal Processing Society Technical Achievement Award in 2004. He was elected a Fellow of the International Speech Communication Association (ISCA) in 2009 and he was the recipient of the ISCA Medal in 2010 for Scientific Achievement.

Milica Gašić (Member, IEEE) graduated in computer science and mathematics from the University of Belgrade, Belgrade, Serbia, in 2006 and the M.Phil. degree in computer speech, text, and Internet technology, in 2007 and the Ph.D. degree in statistical dialog modeling in 2011 from University of Cambridge, Cambridge, U.K.

She is a Research Associate in the Dialogue Systems Group, University of Cambridge. She has published around 20 peer-reviewed conference and journal papers in the area of dialog management.

Dr. Gašić served on the organizing committee for the Young Researcher's Roundtable on Spoken Dialogue Systems in 2009.

Blaise Thomson (Member, IEEE) received the B.S. degree in pure mathematics, computer science, statistics and actuarial science from the University of Cape Town, Cape Town, South Africa, and the M.Phil. degree in computer speech, text and internet processing and the Ph.D. degree in dialogue systems from the University of Cambridge, Cambridge, U.K., in 2006 and 2010, respectively.

He is a Research Fellow at St John's College, University of Cambridge, Cambridge, U.K. He has published around 30 peer-reviewed journal and conference papers, largely in the field of dialog management.

Dr. Thomson was awarded the IEEE Student Spoken Language Processing award for a paper at the International Conference on Acoustics, Speech, and Signal Processing (ICASSP) in 2008 and, in 2010, he coauthored best papers at a Spoken Language Technology Workshop (SLT) and an Annual Conference of the International Speech Communication Association (INTERSPEECH). He was Co-Chair of the 2009 Association for Computational Linguistics (ACL) Student Research Workshop and copresented a tutorial on POMDP dialog management at INTERSPEECH 2009.

Jason D. Williams (Member, IEEE) received the B.S.E. degree in electrical engineering from Princeton University, Princeton, NJ, in 1998 and the M.S. and Ph.D. degrees in speech and language processing from Cambridge University, Cambridge, U.K., in 1999 and 2006, respectively.

He is with Microsoft Research, Redmond, VA. His interests include spoken dialog systems, planning under uncertainty, spoken language understanding, and speech recognition. Prior to Microsoft, he was Principal Member of Technical Staff at AT&T Labs—Research from 2006 to 2012. He has also held several positions in industry building spoken dialog systems, including at Tellme Networks (now Microsoft) as Voice Application Development Manager.

Dr. Williams is on the Scientific Committee of the Special Interest Group on Dialog and Discourse (SigDial) and is on the Board of Directors of the Association for Voice Interaction Design (AVIxD). From 2009 to 2011, he served on the IEEE Speech and Language Technical Committee (SLTC) in the area of spoken dialog systems.