Decision Making Strategies for Finite State Bi-Automaton in Dialog Management

Fabrizio Ghigi[1] and M. Inés Torres[1]

Abstract Stochastic regular *bi-languages* has been recently proposed to model the joint probability distributions appearing in some statistical approaches of Spoken Dialog Systems. To this end a deterministic and probabilistic finite state *bi-automaton* was defined to model the distribution probabilities for the dialog model. In this work we propose and evaluate decision strategies over the defined probabilistic finite state *bi-automaton* to select the best system action at each step of the interaction. To this end the paper proposes some heuristic decision functions that consider both action probabilities learn from a corpus and number of known attributes at running time. We compare either heuristics based on a single next turn or based on entire paths over the automaton. Experimental evaluation was carried out to test the model and the strategies over the Let's Go Bus Information system. The results obtained show good system performances. They also show that local decisions can lead to better system performances than best path-based decisions due to the unpredictability of the user behaviors.

Key words: statistical dialog management, decision strategies

1 Introduction

Spoken Dialog Systems (SDS) enable human-machine interaction using natural spoken language [8, 9]. The process of interaction between the machine and a real user pass through several steps. One of the crucial steps in this process is the election of a next system action, a task performed by the Dialog Manager (DM). The DM is the module responsible of pursue the dialog goal by choosing a coherent action in response to a user input [2]. Due to its complexity the design of DM has been traditionally based on hand-crafted rules [1, 7]. However, over the last few years, approaches

[1]Dpto. Electricidad y Electrónica. Universidad del País Vasco, Spain
e-mail: fabrizio.ghigi@ehu.es, e-mail: manes.torres@ehu.es

that use statistical frameworks to deal with decision strategies and task models have been providing compelling results on modeling interaction. These include Bayesian networks [10], Stochastic Finite-State models [5, 11] and the state-of-the-art Partially Observable Markov Decision Process [6, 14]. The interactive pattern recognition framework [13] has also be proposed to represent SDS [12]. This formulation needs to estimate the joint probability distribution over the semantic language provided by the speech understanding system and the language of actions provided by the DM. In a previous work [11] we have proposed to model this joint probability distribution by stochastic regular *bi-languages*. To this end a deterministic and Probabilistic Finite State Bi-Automata (PFSBA) was defined in that work. Our goal now in this paper is to propose and evaluate DM strategies over this PFSBA-based dialog model. We are aimed at providing the DM with the best decision at each system turn. This decision will be selected according with some heuristic search on the model graph at running time. In Section 2 we summarize the deterministic PFSBA defined in [11]. In Section 3 we propose four decision strategies to be implemented by the DM at each system turn. The experiments and the results obtained are described in Section 4. Then Section 5 reports some final remarks and the future work planned.

2 Model definition

Let us consider an SDS as an interactive pattern recognition system [12, 13]. Let now h be an hypothesis or output that the dialog manager of a SDS proposes. Then the user provides some feedback signals, f, which iteratively help the dialog manager to refine or to improve its hypothesis until it is finally accepted by the user. A basic simplification is to ignore the user feedback except for the last interaction an hypothesis h'. Assuming the classical *minimum-error criterion* the Bayes' decision rule is simplified to maximize the posterior $Pr(h|h',f)$, and a best hypothesis $\hat{h}$ is obtained as follows:

$$\hat{h} = \arg\max_{h \in \mathscr{H}} P(h|h',f) \tag{1}$$

This maximization procedure defines the way the dialog manager of a SDS choose the best hypothesis, i.e. the best action at each interaction step, given the previous hypothesis h' and the user feedback f. However, alternative criteria could also be considered as shown in Section 3. In a SDS, the interpretation of the user feedback f can not be considered a deterministic process. In fact the space of decoded feedback $\mathscr{D}$ is the output of an ASR system. Thus a best hypothesis can be obtained as follows [4, 12, 13]:

$$\hat{h} = \arg\max_{h \in \mathscr{H}} \sum_{d \in \mathscr{D}} P(h,d|h',f) \tag{2}$$

where f is the user turn, d is the decoding of the user turn, h is the hypothesis or output produced by the system and h' is the *history of the dialog*.

The user feedback f depends on its previous feedback f' according to some unknown distribution $P(f|f',h)$, which represents the user response to the history of system hypotheses and user feedbacks. This distribution considers the user behavior and stands for a user model $\mathscr{M}_{\mathscr{U}}$. However, feedback f' produced by the user in the previous interaction is not corrupted by any noisy channel, such as an ASR system, before arriving to the user again. Thus, a deterministic decoding $d : \mathscr{F} \to \mathscr{D}$ maps each user turn signal into its corresponding unique decoding $d' = d(f')$ before arriving to the user. Consequently the *best* user feedback $\hat{f}$ is the one that maximizes the posterior $P_{\mathscr{M}_{\mathscr{U}}}(f|d',h)$

$$\hat{f} = \arg\max_{f \in \mathscr{F}} P(f|d',h) \approx \arg\max_{f \in \mathscr{F}} P_{\mathscr{M}_{\mathscr{U}}}(f|d',h) \tag{3}$$

where $\hat{f}$ is estimated using only the hypothesis produced by the system and the feedback produced by the user in the previous interaction step according to its *user model*. Figure 1 shows some user-manager interaction steps.

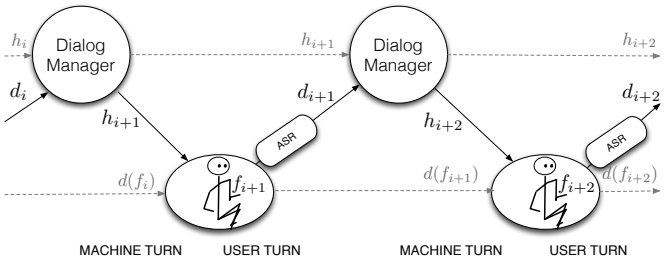

Fig. 1 User-Manager interaction steps. h is the hypothesis produced by the system that depends on the previous hypothesis h' and the decoded user feedback d. f is the user turn that depends h and on the previous user feedback f'.

We are now summarizing the probabilistic Dialog Model defined in [11] to deal with both the dialog manager hypothesis probability distribution $P(h|d,h')$ and the user feedback probability distribution $P(f|h,d')$. Let Σ be the finite alphabet of semantic symbols provided by some speech understanding system. Thus, $\tilde{d}_i = d_1 \dots d_{|\tilde{d}_i|} \in \Sigma^{\leq m}$ represents the decoding of a user feedback f. Let now Δ be the finite alphabet of dialog acts that compose each of the hypotheses $\tilde{h}_i = h_1 \dots h_{|\tilde{h}_i|} \in \Delta^{\leq n}$ provided by the dialog manager. Let $\mathbf{z}$ be a *bi-string* over the extended alphabet $\Gamma \subseteq \Sigma^{\leq m} \times \Delta^{\leq n}$ such as $\mathbf{z} : \mathbf{z} = z_1 \dots z_{|\mathbf{z}|}$, $z_i = (\tilde{d}_i : \tilde{h}_i)$ where $\tilde{d}_i = d_1 \dots d_{|\tilde{d}_i|} \in \Sigma^{\leq m}$ and $\tilde{h}_i = h_1 \dots h_{|\tilde{h}_i|} \in \Delta^{\leq n}$. A Dialog Model $\mathscr{DM}$ is defined as a deterministic and probabilistic finite-state *bi-automaton* $\mathscr{DM} = (\Sigma, \Delta, \Gamma, Q, \delta, q_0, P_f, P)$ where

- Σ and Δ are two finite alphabets representing semantic symbols provided by the user and dialog acts provided by the dialog manager respectively, Γ is an extended alphabet such that $\Gamma \subseteq (\Sigma^{\leq m} \times \Delta^{\leq n})$, $m,n \geq 0$. ε represents the empty

symbol for both alphabets, i.e., $\varepsilon \in \Sigma$, $\varepsilon \in \Delta$ and $(\tilde{\varepsilon} : \tilde{\varepsilon}) \in \Gamma$. To simplify let $\tilde{\varepsilon}$ be ε.

- $Q = Q_{\mathscr{M}} \bigcup Q_{\mathscr{U}}$ is a finite set of states labelled by *bi-strings* $(\tilde{d}_i : \tilde{h}_i) \in \Gamma$. The set $Q_{\mathscr{M}}$ includes machine states before a machine turn providing an hypothesis and the set $Q_{\mathscr{U}}$ includes user states before providing a feedback.
- $\delta \subseteq Q \times \Gamma \times Q$ is the union of two sets of transitions $\delta = \delta_{\mathscr{M}} \bigcup \delta_{\mathscr{U}}$ as follows:
  - $\delta_{\mathscr{M}} \subseteq Q_{\mathscr{M}} \times \Gamma \times Q_{\mathscr{U}}$ is a set of transitions of the form $(q, (\varepsilon : \tilde{h}_i), q')$ where $q \in Q_{\mathscr{M}}$, $q' \in Q_{\mathscr{U}}$ and $(\varepsilon : \tilde{h}_i) \in \Gamma$
  - $\delta_{\mathscr{U}} \subseteq Q_{\mathscr{U}} \times \Gamma \times Q_{\mathscr{M}}$ is a set of transitions of the form $(q, (\tilde{d}_i : \varepsilon), q')$ where $q \in Q_{\mathscr{U}}$, $q' \in Q_{\mathscr{M}}$ and $(\tilde{d}_i : \varepsilon) \in \Gamma$
- $q_0 \in Q_{\mathscr{M}}$ is the unique initial state and it is labelled as $(\varepsilon : \varepsilon)$.
- $P_f : Q \to [0,1]$ is the final-state probability distribution
- $P : \delta \to [0,1]$ defines transition probability distributions ($P(q,b,q') \equiv Pr(q',b|q)$ for $b \in \Gamma$ and $q, q' \in Q$) such that:

$$P_f(q) + \sum_{b \in \Gamma, q' \in Q} P(q,b,q') = 1 \ \forall q \in Q \tag{4}$$

where a transition (q,b,q') is completely defined by q and b. Thus, $\forall q \in Q$, $\forall b \in \Gamma$ $|\{q' : (q,b,q')\}| \leq 1$

Let $\mathbf{z}$ be a *bi-string* over the extended alphabet $\Gamma \subseteq \Sigma^{\leq m} \times \Delta^{\leq n}$ such as $\mathbf{z} : \mathbf{z} = z_1 \dots z_{|\mathbf{z}|}$, $z_i = (\tilde{d}_i : \tilde{h}_i)$. Let now $\theta = (q_0, z_1, q'_1, z_2, q_2, \dots, q'_{|\mathbf{z}|-1}, z_{|\mathbf{z}|}, q_{|\mathbf{z}|})$, $q_i \in Q_{\mathscr{M}}$, $q'_i \in Q_{\mathscr{U}}$, be a path for $\mathbf{z}$ in $\mathscr{DM}$. The probability of generating θ is:

$$Pr_{\mathscr{DM}}(\theta) = \left(\prod_{j=1}^{|\mathbf{z}|} P(q_{j-1}, z_j, q'_j) \right) \cdot P_f(q_{|\mathbf{z}|}) \tag{5}$$

$\mathscr{DM}$ is unambiguous. Then, a given *bi-string* $\mathbf{z}$ can only be generated by $\mathscr{DM}$ through a unique valid path $\theta(\mathbf{z})$. Thus, the probability of generating $\mathbf{z}$ with $\mathscr{DM}$ is $Pr_{\mathscr{DM}}(\mathbf{z}) = Pr_{\mathscr{DM}}(\theta(\mathbf{z}))$. Additionally, each machine and/or user state need to be labelled with the values of all relevant internal variables, which can be updated after each user turn. Thus, an additional alphabet appears to represent valued attributes of these internal variables, thus leading to an *attributed* model [11]. These internal variables are a subset of the semantic decoding set, i.e. the subset of Σ set that consists of task dependent symbols. These internal variables can lead to simple *known*, *unknown* attributes that can just be represented by the presence or absence of the attribute at each state. Thus, the new alphabet represents just the knowledge of the value. Alternatively confidence measures can also been considered. The model $\mathscr{DM}$ was then extended to add another finite alphabet Ω. Each state $q \in Q$ is now labelled by *bi-strings* $[(\tilde{d}_i : \tilde{h}_i), \tilde{w}_i] \in \Gamma \times \Omega$ where the valued attributes are also considered. The knowledge of the attributes leads to different strategies for the dialog manager since the transition function $\delta \subseteq Q \times \Gamma \times Q$ and the transition probability distribution $P : \delta \to [0,1]$ have a strong dependency of internal attributed attached to the states.

Example. Let us take a dialog from Let's Go [8] task that was used in this work for experiments.

U: I'm leaving from CMU
PlaceInformation[DeparturePlace]
S: Leaving from <query.departureplace CMU>. Is this correct?
Explicit confirm
U: Yes.
Generic[Yes]
S: Right. What is your destination?
Inform:confirm_okay Request:query_arrival

$\Sigma = \{PlaceInformation.DeparturePlace, Generic.Yes\}$ is the set of user symbols, $\Delta = \{Explicit.confirm, Inform.confirm\_okay, Request.query\_arrival\}$ is the alphabet of system dialog acts and $\Omega = \{\text{query.departureplace}\}$ is the alphabet of the task attributes. Figure 2 shows a $\mathscr{DM}$ where bold lines define path θ matching some *bi-string*.

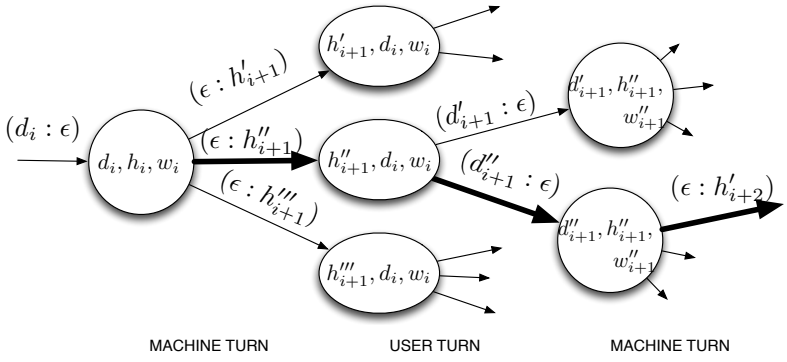

Fig. 2 Bold lines show a path θ matching some *bi-string* in a probabilistic *bi-automaton* $\mathscr{DM}$

3 Heuristic functions to achieve user goals

In this section we define four heuristic functions that represent different strategies to deal with user goals while minimizing the involved cost. Thus the next action to be selected by the DM at each system turn is the one that maximizes the corresponding heuristic function. The first two strategies (MP and MPA) deal with local decisions, i.e., they only evaluate next turn nodes and edges. Two more proposals (BP and BPA) evaluate entire paths from the actual state to a closing state in the graph. On the other hand strategies MP and BP only take into account the transitions probabilities

in the model whereas strategies MPA and BPA also base the decision in the amount of attributes potentially be filled.

Maximum Probability (MP) strategy

This strategy just deals with Equation 1. The best DM hypothesis to be selected by the DM is the one that maximizes the posterior $Pr(h|h',f)$ according with Equation 2. A suboptimal approach can be considered through a two step decoding: find first an optimal user feedback $\hat{d}$ and then, use $\hat{d}$ to decode system hypothesis $\hat{h}$ as follows:

$$\hat{d} = \arg\max_{d \in \mathscr{D}} P(f|d)P(d|h') \tag{6}$$

$$\hat{h} \approx \arg\max_{h \in \mathscr{H}} P(h|\hat{d},h') \tag{7}$$

The dialog manager hypothesis probability distribution $P(h|d,h')$ and the user feedback probability distribution $P(f|h,d')$ have been modeled in this work by the PFSBA presented in previous section. As a consequence the search for the most likely hypothesis $\hat{h}$ in Equation 7 is equivalent to choose the edge of highest transition probability at each system turn. i.e.

$$\hat{h} = \arg\max_{h_{ij} \in \mathscr{H}(q_i)} P(q_i,(\varepsilon\colon h_{ij}),q'_j) \tag{8}$$

where $q_i \in Q_{\mathscr{M}}$ is a system state labelled as $((\tilde{d}_i : \tilde{h}_i) : \tilde{w}_i)$ and $q_j \in Q_{\mathscr{U}}$ is a user turn labelled as $((\tilde{d}_j : \tilde{h}_j) : \tilde{w}_j)$ such that $P(q_i,(\varepsilon\colon h_{ij}),q_j) > 0$, being $h_{ij} \in \mathscr{H}(q_i)$ the associated system hypotheses. This strategy has been evaluated in [4] in a dialog generation task showing good task completion rates and good model behaviors.

Maximum Probability strategy with Attributes (MPA)

We want to know the number of attributes filled as a consequence of DM decisions. Let us consider now two states of the model q_i and q_j labelled as $((\tilde{d}_i : \tilde{h}_i) : \tilde{w}_i)$ and $((\tilde{d}_j : \tilde{h}_j) : \tilde{w}_j)$. According to the model definition in Section 2 $\tilde{w}_i$ and $\tilde{w}_j$ are two sequences of symbols $w \in \Omega$ representing filled attributes in the model states i and j. We want now to define a transformation distance between $\tilde{w}_i$ and $\tilde{w}_j$ aimed at representing the number of new attributes filled in state q_j relative to the state q_i. To this end let now consider the number $n_{del}(\tilde{w}_{ij})$ of single-symbol deletions and the number $n_{ins}(\tilde{w}_{ij})$ required to transform sequence $\tilde{w}_i$ to $\tilde{w}_j$. Let now $d_w(q_i,q_j)$ be the attribute distance between nodes q_i and q_j defined as follows

$$d_w(q_i,q_j) = n_{ins}(\tilde{w}_{ij}) - n_{del}(\tilde{w}_{ij}) \tag{9}$$

Notice that we are now focussing on the number of filled attributes regardless of their associated value. Thus symbol substitutions are not considered here.

The DM has to take into account that the DM fruitful actions, like the ones aimed at consulting a database to provide the user information requirements, need the related task attributes to be previously filled. Thus this strategy is aimed at selecting an hypothesis with a high probability according with the training corpus but also at filling a high number of task attributes according with the history of the current running dialog. Notice that only user actions can change the number of filled attributes. Let $q_i \in Q_{\mathscr{M}}$ be a system state and $q_j \in Q_{\mathscr{U}}$ be a destination state $q_j \in Q_{\mathscr{U}}$ such that $P(q_i,(\varepsilon\colon h_{ij}),q_j) > 0$ being $h_{ij} \in \mathscr{H}(q_i)$ the associated system hypothesis. The user being at state q_j can provide $|\mathscr{F}(q_j)|$ feedbacks $f_{jk} \in \mathscr{F}(q_j)$ leading to $|\mathscr{F}(q_j)|$ systems states q_{jk}. Thus the attribute distance $d_w(q_i,q_{jk}) \quad k = 1,\ldots,|\mathscr{F}(q_j)|$ between node q_i and each of node q_{jk} has to be computed. Then we define a heuristic function $F(q_i,(\varepsilon\colon h_{ij}),q_j)$ associated to each potential transition as follows:

$$F(q_i,h_{ij},q_j) = \log P(q_i,(\varepsilon\colon h_{ij}),q'_j) + \max_{k\in|\mathscr{F}(q_j)|} d_w(q_i,q_{jk}) \tag{10}$$

Then the action to be taken by the system is the one that maximizes the heuristic function as follows

$$\hat{h} = \arg\max_{h_{ij}\in\mathscr{H}(q_i)} F(q_i,h_{ij},q_j) \tag{11}$$

Best Path Probability (BP) strategy

This strategy is aimed at exploring all the paths in the graph that begin in the current dialog state q_0. However only paths that lead to a closing state in the model are considered.

Let $\mathbf{z}$ be a *bi-string* over the extended alphabet $\Gamma \subseteq \Sigma^{\leq m} \times \Delta^{\leq n}$ such as $\mathbf{z} : \mathbf{z} = z_1 \ldots z_{|\mathbf{z}|}, z_i = (\tilde{d}_i : \tilde{h}_i)$ such that the associate path $\theta_{\mathbf{z}} = (q_0, z_1, q'_1, z_2, q_2, \ldots, q'_{|\mathbf{z}|-1}, z_{|\mathbf{z}|}, q_{|\mathbf{z}|})$, $q_i \in Q_{\mathscr{M}}$, $q'_i \in Q_{\mathscr{U}}$ begins in the current state of the system $q_0 \in Q_{\mathscr{M}}$. The probability $Pr_{\mathscr{D},\mathscr{M}}(\theta_{\mathbf{z}})$ of generating $\mathbf{z}$ is calculated according to Equation 5. Let now $\Theta_f(q_0)$ be the set of paths $\theta_{\mathbf{z}} \in \Theta_f(q_0)$ beginning at state q_0 and ending in a final state, i.e. $P_f(q_{|\mathbf{z}|}) = 1$. The best path $\hat{\theta}_{\mathbf{z}} \in \Theta_f(q_0)$ is the one that maximizes the normalized probability, i.e.

$$\hat{\theta}_{\mathbf{z}} = \arg\max_{\theta_{\mathbf{z}}\in\Theta_f(q_0)} \frac{1}{|\mathbf{z}|} Pr_{\mathscr{D},\mathscr{M}}(\theta_{\mathbf{z}}) \tag{12}$$

Thus the DM selects the first hypothesis $\tilde{h}_1$ defining the first element z_1 of the *bi-string* $\mathbf{z}$ associated to $\hat{\theta}_{\mathbf{z}}$, such that $z_1 = (\varepsilon : \tilde{h}_1)$

Best Path Probability strategy with Attributes (BPA)

In the same way as in MPA we want now to include in the score the number of attributes filled as a consequence of DM decisions. To this end we define a heuristic function $F(\theta_{\mathbf{z}})$ associated to each $\theta_{\mathbf{z}} \in \Theta_f(q_0)$ beginning at state q_0 and ending in a final state as follows:

$$F(\theta_{\mathbf{z}}) = \sum_{i=1}^{|\mathbf{z}|} \log P(q_{i-1}, z_i, q_i) + \log P_f q_{|\mathbf{z}|} + \sum_{i=1}^{|\mathbf{z}|} (\max_{k \in |\mathscr{F}(q_j)|} d_w(q_i, q_{jk})) \tag{13}$$

The best path $\hat{\theta}_{\mathbf{z}} \in \Theta_f(q_0)$ is now the one that maximizes the normalized heuristic function $F(\theta_{\mathbf{z}})$, i.e.

$$\hat{\theta}_{\mathbf{z}} = \arg\max_{\theta_{\mathbf{z}} \in \Theta_f(q_0)} \frac{1}{|\mathbf{z}|} F(\theta_{\mathbf{z}}) \tag{14}$$

This strategy is similar to BP, but it also takes into account the number of attributes filled at each step. Thus the DM also selects now the first hypothesis $\tilde{h}_1$ defining the first element z_1 of the *bi-string* $\mathbf{z}$ associated to $\hat{\theta}_{\mathbf{z}}$, such that $z_1 = (\varepsilon : \tilde{h}_1)$

4 Experiments

The four strategies described in Section 3 were evaluated over a dialog generation task from a corpus of transcribed dialogs between real users and an automatic information system.

Learning and using models to generate dialogs

For these experiments a DM model and a user model were estimated from Let's Go corpus [8]. Let's Go is a set of spoken dialogues in English in the bus information domain. Let's Go system has been developed by Carnegie Mellon University over the Olympus-Ravenclaw framework [1]. It provides schedules and route information about the city of Pittsburgh's bus service to the general public. In this work we use a set of dialogues collected by the Let's Go Bus Information system in about two months, from March 2005 to April 2005. This set consists in 1840 dialogs between Ravenclaw DM and 1840 real users that include 28141 system turns and 28071 user turns. In order to have the Let's Go corpus labelled in terms of Dialogue Acts, we have collected the information associated to each system turn from the log files of the Ravenclaw DM, whereas the information associated to the user turn was collected from the output of the Phoenix semantic decoder. Thus we got a Σ alphabet

consisting of 138 semantic symbols provided by the use and a Δ alphabet consisting of 49 dialog acts provided by the dialog manager. Additionally the attribute alphabet Ω consists of 14 attributes. A dialog example including Σ, Δ and Ω symbols can be found at the end of Section 2.

Let split the corpus into two subsets to train two models, one acts as a dialog manager and provides hypotheses according to the strategies defined in Section 3. The other one acts as simulated user that proposes a random user feedback in order to generate a wider variety of dialogs [4, 12]. Both models have to deal with unseen events, i.e. unknown situations at training corpus. The dialog manager can provide an hypothesis $\tilde{h}_i$ that does not lead to any of the existing states created when trained from the dialog corpus. In the same way the simulated user can provide a user feedback $\tilde{f}_i$ not appearing in the training corpus, so not in the model. The generalization issue is tackled by adopting the back-off smoothing strategy proposed in [4] for unseen events. Then a set of new dialogs were obtained from the interaction between the DM and the simulated user, as showed in Figure 1. For those experiments an error model simulated the ASR recognition errors. This model was trained from the dialog corpus where both the transcription of the user utterance and the output of the ASR can be found.

Metrics

In order to evaluate the system we have decided to use 3 different metrics, Task Completion (TC), Appropriate Utterance (AU) and Average Dialog Length (ADL). The metrics used to evaluate the system are:

Task Completion (TC). Measures the success of the system in providing the user with the information requested [3]. This is an automated metric and we compute it by checking if in the dialog we arrive to the point of making a query to the backend, to retrieve the information about a schedule asked by the user.

Appropriate Utterance (AU). An utterance is considered appropriate when it provides the user the required information, when it asks for additional information which is essential to respond to the user's request or when it is dealing with a repair strategy. AU evaluates whether the DM provides a coherent response at each turn according to its input (output of the ASR). We measured this metric manually, thus for each turn we check if the system answer to an user turn was appropriate.

Average Dialogue Length (ADL). The average number of turns in a dialog. A dialog that achieve the goal but has a really long length could be indication of repeated ASR errors, so the user and the system collaborate with recovering techniques in orders to recover the error and the dialog gets longer.

Task Completion will give us a feedback on the global success of a single dialog. Appropriate Utterance can give us a look into the specific answer of the system. Average Dialog Length can give us a feedback about the quality of a dialog.

Experimental results

We carried out four sets of experiments to evaluate strategies defined in Section 3, MP, MPA, BP and BPA, to get the best hypothesis to be provided by the DM at each interaction step. A set of 100 dialogs was generated for each strategy. TC , AU and ADL were computed for each of the set of generated dialogs. For comparisons purposes these metrics were also computed for a random set of 100 dialogs extracted from the corpus, which was conducted by Ravenclaw DM. Table 1 shows the results of this evaluation. This Table shows higher TC values for dialogs generated by all the proposed models than the one computed over the reference set that was managed by Ravenclaw. However the ADL value is higher in all the sets generated by the proposed models than in the reference set. Thus, the proposed formulation seems to achieve well the user goals measured in terms of TC but needs a higher number of turns to finish a dialog.

		Local Decisions		**Path-based Decisions**	
	Ravenclaw	**MP**	**MPA**	**BP**	**BPA**
TC(%)	51.96%	76.09%	79.59%	54.54%	68.00%
AU(%)	95.53%	97.41%	96.23%	94.58%	87.93%
ADL	20.45	26.57	34.77	30.60	31.88

Table 1 Task Completion (TC), Average Dialog Length (ADL) and Appropriate Utterance (AU) for the experiments carried out. The four strategies defined in Section 3 (MP, MPA, BP and BPA) for the DM based on Probabilistic *bi automaton* were evaluated and then compared to the Ravenclaw DM

Then we compared the results obtained for each strategy shown in Table 1. Experimental results Table 1 show that the strategies using a local decision process (MP, MPA) present significantly higher performance in terms of Task Completion. This is likely due to the unpredictable user behavior, modeled in the simulated user by choosing randomly the next user action. Path-based decision strategies BP and BPA select the next action as the first one in the best complete path from the current system state up to a final node. User unpredictable behavior often causes the user model to change the path in the *bi-automaton*, making worthless the selection of a best path. Furthermore we notice an increase in ADL for strategies including the attributes in the heuristic decision. This kind of heuristic tends to make the system ask for the whole set of possible attributes, also if some of them, like the bus line number, are not required. As a consequence the number of turns in generated dialogs increases. AU values are higher when local decision strategies were considered. This is due to the fact that the best path may not include the first action with the highest probability or the one with the maximum heuristic function. Table 2 and Table 3 show the total Number of Turns (NT), the Number of Turns generated through a Smoothed edge (NTS) and the Smoothing Rate (SR) representing the percentage of turns obtained through smoothing techniques of both DM and user models, when

local and path-based decision strategies were considered. These tables reveal a high use of smoothed edges that underlines the importance of considering an appropriate generalization strategy. These tables also show that strategies performing local decisions seem to have a slighter lower smoothing rate percentage.

	Local Decision Strategies					
	MP			**MPA**		
	Total	**User**	**System**	**Total**	**User**	**System**
NT	1891	944	947	2418	1207	1211
NTS	664	304	300	927	509	418
SR	35.11	32.20	38.01	38.33	42.17	34.51

Table 2 Total Number of Turns (NT), Number of Turns generated through a Smoothed edge (NTS) and Smoothing Rate (SR) representing the percentage of turns obtained through smoothing techniques for system and user turns when strategies based on local decisions were considered.

	Path-based Decision Strategies					
	BP			**BPA**		
	Total	**User**	**System**	**Total**	**User**	**System**
NT	2028	994	1034	2203	1101	1102
NTS	889	493	396	849	458	391
SR	43.84	49.59	38.29	38.54	41.59	35.48

Table 3 Number of Turns generated through a Smoothed edge (NTS) and Smoothing Rate (SR) representing the percentage of turns obtained through smoothing techniques for system and user turns when strategies based on exploring sets of paths were considered.

5 Conclusions & Future Work

In conclusion we have defined several decision making strategies over deterministic and probabilistic finite state *bi-automaton* for dialog management. Our goal was to provide the dialog manager with the best decision at each system turn. The best system hypothesis was selected at running time according with some heuristic search aimed at achieving the user goals. Two strategies dealt with local decisions, i.e., they only evaluated the next turn nodes and edges, and obtained the best system performance on task completion. Two more proposals evaluated entire paths from the current system state to a closing state. Experimental results showed that path-based strategies that implement decisions based on possible future user actions achieved lower system performances due to unpredictability of the user behavior. Furthermore we observed a small increase in Task Completion when the heuristic function

also consider the number of attributes potentially be filled by the user as a consequence of the dialog manager decisions. Ongoing work will focuses on deploying a complete spoken dialog system demo and testing these strategies with real users.

Acknowledgements This work has been partially supported by the Spanish Ministry of Science under grants BES-2009-028965 and TIN2011-28169-C05-04, and by the Basque Government under grants IT685-13 and S-PE12UN061.

References

1. Dan Bohus and Alexander I. Rudnicky. The ravenclaw dialog management framework: Architecture and systems. *Computer Speech and Language*, 23:332–361, 2009.
2. Gavin E Churcher, Eric Stevan Atwell, and Clive Souter. Dialogue management systems: a survey and overview. *RESEARCH REPORT SERIES-UNIVERSITY OF LEEDS SCHOOL OF COMPUTER STUDIES LU SCS RR*, 1997.
3. Morena Danieli and Elisabetta Gerbino. Metrics for evaluating dialogue strategies in a spoken language system. In *Proceedings of the 1995 AAAI Spring Symposium on Empirical Methods in Discourse Interpretation and Generation*, volume 16, pages 34–39, 1995.
4. Fabrizio Ghigi, M. Inés Torres, Raquel Justo, and José-Miguel Benedí. Evaluating spoken dialogue models under the interactive pattern recognition framework. In *INTERSPEECH*, pages 480–484, 2013.
5. Lluís F Hurtado, Joaquin Planells, Encarna Segarra, Emilio Sanchis, and David Griol. A stochastic finite-state transducer approach to spoken dialog management. In *INTERSPEECH*, pages 3002–3005, 2010.
6. Filip Jurčíček, Blaise Thomson, and Steve Young. Reinforcement learning for parameter estimation in statistical spoken dialogue systems. *Computer Speech & Language*, 26(3):168–192, 2012.
7. Cheongjae Lee, Sangkeun Jung, Jihyun Eun, Minwoo Jeong, and Gary Geunbae Lee. A situation-based dialogue management using dialogue examples. In *Proceedings of ICASSP*, pages 69–72, 2006.
8. Antoine Raux, Brian Langner, Dan Bohus, Alan W Black, and Maxine Eskenazi. Let's go public! taking a spoken dialog system to the real world. In *Proc. of Interspeech*, 2005.
9. Stephanie Seneff and Joseph Polifroni. Dialogue management in the mercury flight reservation system. In *Proceedings of the 2000 ANLP/NAACL Workshop on Conversational systems - Volume 3*, pages 11–16, Stroudsburg, PA, USA, 2000.
10. 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 *Spoken Language Technology Workshop (SLT), 2010 IEEE*, pages 460–465. IEEE, 2010.
11. M Inés Torres. Stochastic bi-languages to model dialogs. *Finite State Methods and Natural Language Processing*, page 9, 2013.
12. M. Inés Torres, Jose Miguel Benedí, Raquel Justo, and Fabrizio Ghigi. Modeling spoken dialog systems under the interactive pattern recognition framework. In *SSPR&SPR. Lecture Notes on Computer Science*, pages 519–528, 2012.
13. Alejandro H. Toselli, Enrique Vidal, and Francisco Casacuberta, editors. *Multimodal Interactive Pattern Recognition and Applications*. Springer-Verlag, 2011.
14. Jason D. Williams and Steve Young. Partially observable markov decision processes for spoken dialog systems. *Computer Speech and Language*, 21:393–422, 2007.