Spoken Dialog Systems based on Online Generated Stochastic Finite-State Transducers

Lluís-F. Hurtado, Joaquin Planells, Encarna Segarra, Emilio Sanchis

Departament de Sistemes Informàtics i Computació
Universitat Politècnica de València
E-46022 València, Spain

Abstract

In this paper, we present an approach for the development of spoken dialog systems based on the statistical modelization of the dialog manager. This work focuses on three points: the modelization of the dialog manager using Stochastic Finite-State Transducers, an unsupervised way to generate training corpora, and a mechanism to address the problem of coverage that is based on the online generation of synthetic dialogs. Our proposal has been developed and applied to a sport facilities booking task at the university. We present experimentation evaluating the system behavior on a set of dialogs that was acquired using the Wizard of Oz technique as well as experimentation with real users. The experimentation shows that the method proposed to increase the coverage of the Dialog System was useful to find new valid paths in the model to achieve the user goals, providing good results with real users.

Keywords:
Spoken Dialog Systems; Dialog Management; Statistical Models; Stochastic Finite-State Transducers; User Simulation; Coverage Problems

Email address: {lhurtado, xplanells, esegarra, esanchis}@dsic.upv.es (*Lluís-F. Hurtado, Joaquin Planells, Encarna Segarra, Emilio Sanchis*)

1. Introduction

1.1. Background literature

A dialog system can be viewed as a human-machine interface that recognizes and understands speech input and generates a spoken answer in successive turns in order to achieve a goal, such as obtaining information or carrying out an action. The development of spoken dialog systems is one of the main objectives of spoken language technology research. Voice-driven applications such as in-car navigation systems or telephone information services are common examples of spoken dialog systems. Most dialog systems are oriented to restricted domain tasks, mixed initiative, and telephone access; however, several new applications have appeared in portable devices such as mobile phones or tablets.

Different modules are necessary to be able to carry out the final goal of a spoken dialog system: a Speech Recognition module converts the audio signal into words; an Understanding module converts these words into a semantic representation; a Dialog Manager decides the next system action in order to fulfill the user's needs; an Answer Generator converts the action of the system into one or more sentences; and a Text-to-Speech Synthesizer converts the system sentences into audio. Each module has its own characteristics and the selection of the most convenient model for it varies depending on certain factors: the goal of each module, the possibility of manually defining the behavior of the module, or the capability of automatically obtaining models from training samples. The use of statistical techniques for the development of the different modules that compose the dialog system has been of growing interest over the last years. These methodologies have traditionally been applied within the fields of Automatic Speech Recognition and Natural Language Understanding (Levin and Pieraccini, 1995), (Minker et al., 1999), (Segarra et al., 2002), (Esteve et al., 2003), (He and Young, 2003), (Raymond and Riccardi, 2007), (Hahn et al., 2010), (Tür and Mori, 2011).

The Dialog Manager is in charge of selecting the action that the dialog system must perform at each turn. This is usually done by taking into account the last user turn and the history of the dialog. Thus, a Dialog Manager can be seen as a function that maps the user turn to an action. Even though there are models for dialog management in the literature that are manually designed using hand-written rules, over the last years, approaches that use statistical models to represent the behavior of the Dialog Manager

have been providing compelling results. Statistical models can be trained from real dialogs, modeling the variability of user behaviors.

In the literature, statistical models have been successfully used to select the system action for each user turn. These include Multi-layer Perceptrons (Griol et al., 2008), (Hurtado et al., 2006), Maximum a Posteriori classifiers (Hurtado et al., 2005), Example-Based modelization (Lee et al., 2007), Bayesian networks (Martinez et al., 2009), (Meng et al., 2003), (Paek and Horvitz, 2000), Finite State Transducers (Hori et al., 2009), (Hurtado et al., 2010), and Partially Observable Markov Decision Prossess (Williams and Young, 2007). These approaches are usually based on modeling the different processes probabilistically and learning the parameters of the different statistical models from a dialog corpus. In (Griol et al., 2014), a technique for automatic acquiring dialog corpora in which the simulated dialogs are automatically generated, is applied to develop a Dialog Manager based on Multi-layer Perceptron classifiers for four different tasks.

In a previous work, we presented an approach to dialog management based on the use of Multi-layer Perceptrons (Griol et al., 2008). That approach have in common with the one presented in the current work the use of a data structure -dialog register- that contains all the information provided by the user throughout the dialog without considering the order in which the information is provided. The main difference between the two approaches is that in the current proposal the next action to be taken by the system at a given point of the dialog is determined not only by the previous history of the dialog but also by a look-ahead mechanism that estimates the quality of the possible finalizations of the dialog from this point.

One of the most commonly used approaches for dialog modeling is based on the use of Partially Observable Markov Decision Processes (POMDPs) (Jurčíček et al., 2012), (Williams and Young, 2007). The algorithms of parameter estimation used in POMDPs are mainly based on Reinforcement Learning techniques (Sutton and Barto, 1998).

A first approach that uses Reinforcement Learning techniques for the estimation of the Dialog Manager consists of modeling human-computer interaction as an optimization problem using Markov Decision Processes (MDPs) (Levin and Pieraccini, 1997), (Levin et al., 2000), (Singh et al., 1999).

Partially Observable MDPs (POMDPs), which are an extension of the MDPs, outperform MDP-based dialog strategies. In POMDPs, the dialog state is not known with certainty (as opposed to MDPs); therefore, the model needs to have a representation for the distribution of the dialog states (belief

states). The main drawback of these approaches is the large state space required by practical spoken dialog systems, whose representation is intractable if represented directly (Young et al., 2007). Therefore, those approaches are limited to small-scale problems. In the last few years, many studies have been conducted to implement practical spoken dialog systems based on POMDPs. An approach that scales the POMDP framework by the definition of two state spaces is presented in (Young et al., 2010). Another approach based on the use of hierarchical optimization is presented in (Cuayáhuitl et al., 2007). Also, an approach that uses a Bayesian update of the dialog states has been presented in (Thomson and Young, 2010). Besides the computational complexity, the POMDP models need a large number of dialogs to learn good policies. This has been addressed in part by using Gaussian Processes and directly learning from interactions with users (Gašić et al., 2011).

The approaches based on POMDPs maintain active in the belief state all possible states with their attributes values. Differently from them, in the current proposal we modelize the uncertainty by means of a confidence score for each attribute-value pair -belief at attribute level-. Thus, we have active only one state at each point of the dialog allowing our approach to deal with realistic state space sizes.

Statistical models have in common their need to have enough training samples to estimate parameters. Since it is very difficult and time consuming to obtain labeled dialog corpora, even with the Wizard of Oz technique, many works have been developed to automatically generate training dialogs (Georgila et al., 2005), (Schatzmann et al., 2006), (Hurtado et al., 2007), (Keizer et al., 2010), (Ai and Litman, 2011). This is usually done by developing a user simulation module that interacts with a preliminary system that can be manually defined or obtained by a bootstrapping process.

An interesting initiative to evaluate this kind of systems was the Dialog State Tracking Challenge (Williams et al., 2013). In it, differently from previous Spoken Dialog Evaluation challenges (Black et al., 2010), the interaction progress is measured in terms of finding the correct dialog state that describes the result of the interaction until a certain time t. The dialog state tracker takes as input all of the observable elements up to time t in a dialog, including all of the results from the automatic speech recognition and spoken language understanding components, and external knowledge sources such as databases and models of past dialogs. It also takes as input a set of possible dialog state hypotheses, where a hypothesis is an assignment of values to slots in the system. The tracker outputs a probability distribution

over the set of hypotheses. This is adequate to evaluate heterogeneous Dialog Managers, in particular, those based on POMDPs that works considering multiple state hypotheses.

The dialog model proposed in this article is based on the transduction concept and on the use of Stochastic Finite-State Transducers ($SFST$) (Casacuberta and Vidal, 2004). This approach is based on the assumption that the entire dialog history can be condensed into a finite representation (a dialog state). Based on this state and on the user utterance, the system outputs an answer and moves to another state. Preliminary versions of this work have been presented (Hurtado et al., 2010), (Planells et al., 2012).

1.2. Our approach to Dialog Management

In this approach, given a state of the model and a user turn, a system action is selected and a transition to a new state is performed. Therefore, dialog management is based on the modelization of the sequences of system action and user turn pairs. Then, a dialog describes a path in the transducer model from its initial state to a final one.

Since the space of all combinations of possible sequences of system action and user turn pairs is very large, we establish a partition in the space of sequences of pairs. We define a data structure, called Dialog Register (DR), that contains a summary of the information (concepts and attribute values) that is provided by the user throughout the previous history of the dialog. Since the same information can be provided in different order, different sequences of pairs can lead to the same DR. We represent the history of the dialog throughout the corresponding DR. This representation makes the estimation of a statistical model from the training data manageable. This reduction in the space of all the histories of the dialogs had previously been introduced in another dialog system proposed in our laboratory (Griol et al., 2008). In recent years, within the framework of the POMDPs for dialog management, there have been some approaches to state compression to reduce the search space and make the dialog management problem tractable. Among these works we can highlight (Crook and Lemon, 2011), (Cuayhuitl et al., 2011), where the state compression is done after the state space definition. In our approach the definition of the DR implicitly includes a reduction of state space.

In many dialog applications that interact with an Information System, especially where the system is limited to provide the information required by the user, the Dialog Manager has sufficient information supplied by the

DR to decide the next system action. However, in the case of tasks where the system can update the data of the Information System as a result of interaction with the user (as is the case of EDECAN-SPORTS task (Hurtado et al., 2012)), the dialog manager needs to have additional information to take decisions about its next action. These kinds of more complex dialog systems include an Application Manager, which is a module that communicates the Dialog Manager with the Information System. For example, a sentence like *I want to book a basketball court on Monday afternoon*, can induce a booking (update) in the Information System or an error message when there is no court available.

In order to take into account this information, we define a data structure called Dialog State (DS), which includes the following: all the information provided by the user throughout the dialog that is stored in the DR (codified in terms of not-supplied, low-confidence, or high-confidence), the last request to the Application Manager, a summary representation of the last database query result, and the last system answer. Each DS corresponds to a state in the $SFST$ model.

The success of statistical approaches depends on the quality of the data used to develop the dialog model. Considerable effort is necessary to acquire and label a corpus with the data necessary to train a good model. In order to estimate the $SFST$ parameters, a large number of labeled dialogs is required. Due to the high cost of acquiring dialogs with real users a dialog simulator is used. We develop a process in which dialogs are automatically simulated. Unlike other similar proposals in which user simulation takes deterministic and non-deterministic decisions (Keizer et al., 2010), our proposal is based on generating completely random sequences of user and system dialog act pairs.

As in other approaches (Williams and Young, 2007), we consider a dialog act to be a detailed structure rather than considering it just a label that represents the general intention of the turn, such as in DAMSL (Core and Allen, 1997). In our system a dialog act is a frame containing concepts and attribute-value pairs. Once a dialog has been generated, a set of correctness criteria is applied to classify the dialog as valid or not valid. All the valid simulated dialogs are used to learn the $SFST$ parameters.

In general, statistical methodologies for dialog management have a reasonable performance in a laboratory environment, but they can have some problems when they are applied to more realistic environments. They have to deal with the lack of robustness when there are unexpected user utter-

ances or with relevant recognition or understanding errors. Even though a large number of labeled training samples of dialogs is automatically supplied for the estimation of the Dialog Manager parameters, coverage problems can arise. In a dialog with a real user, the dialog manager can get to a situation (a state of the transducer and a user turn) that was never seen in the training samples and therefore has no information on what action should be performed next.

Some proposals have been done to handle uncertainty and thereby to provide the Dialog Manager with mechanisms to improve its robustness against recognition or understanding errors in the framework of POMDPs. (Gašić et al., 2008) show that POMDPs system can learn noise robust policies and that n-best outputs from the speech understanding component can be used to improve robustness. In (Henderson et al., 2008), dialog systems based on Information State Update are provided with a hybrid model that combines reinforcement learning with supervised learning; they also use linear function approximation to deal with states that were not in the training data. Differently from these approaches that learn how to deal with uncertainty during the training process our approach is based on the use of dialog generation during the test process.

This work makes three main contributions. First, we have developed and evaluated a Dialog Manager based on SFSTs whose structure and parameters are automatically estimated from dialog corpora. Second, we have developed a procedure to automatically generate simulated dialogs in terms of system and user dialog act pairs. We have also proposed a procedure to select the correct dialogs from the simulated dialog set. As a result, we have obtained a very large corpus of dialogs to estimate the Dialog Manager model. Third, we propose the use of an online dialog generator to solve coverage problems. Every time an unseen situation occurs (i.e., there is no answer in the model for the user turn in the current state), a simulator is used to obtain a set of valid dialogs that share the same first turns with the current dialog. These dialogs are used to estimate the most likely system answer to the unseen situation.

Our evaluation was carried out in the context of a spoken dialog system for a booking task in a sport facilities in Spanish. These experiments confirm that this approach has a reasonable behavior with real users and can be used in closed-domains, that is, in specific task domains where user and system actions are defined a priori.

The rest of the paper is organized as follows: Section 2 presents the archi-

tecture and the different modules of our dialog system; Section 3 describes the application task; Section 4 describes the Dialog Manager model based on SFSTs; Section 5 introduces the procedure to automatically generate simulated dialogs; Section 6 proposes the use of an online dialog generator to solve coverage problems, and, finally, the experimental evaluation and some conclusions are presented.

2. The EDECAN-SPORTS dialog system

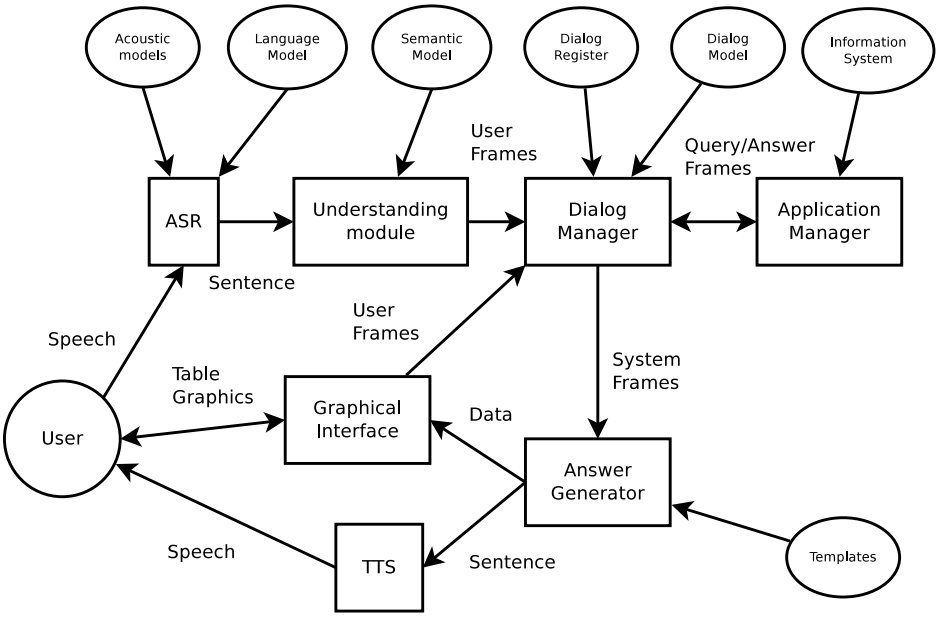

Figure 1: *Architecture of the dialog system.*

A task called EDECAN-SPORTS that consists of mixed-initiative dialogs for booking sport facilities using spontaneous speech was designed (Hurtado et al., 2012). The definition of the semantics of the EDECAN-SPORTS task was carried out considering the different functionalities required for the booking system and the information required to complete them.

Figure 1 presents a scheme of the dialog system that was developed. The system was implemented to allow the integration, substitution and collaboration of the modules even if they are located in different computers. The system contains the different modules of a dialog system, (i.e., the Automatic Speech Recognizer, the Understanding module, the Dialog Manager,

the Answer Generator, and the Text-to-Speech module). In addition, it contains specific modules that are referred to as the Application Manager and the Graphical Interface. The Application Manager controls the access to the database, not only to provide information but also to modify it when a booking or a cancellation must be made. The Graphical Interface is used by the system to give some information to the user in terms of tables and text on the screen. It must be noted that the Automatic Speech Recognizer, the Understanding module, and the Dialog Manager are based on statistical models that are learned from training samples, as described in the following sections.

2.1. Automatic Speech Recognition and Understanding

The design of the dialog system allows the integration of independent Automatic Speech Recognition (ASR) modules to the architecture.

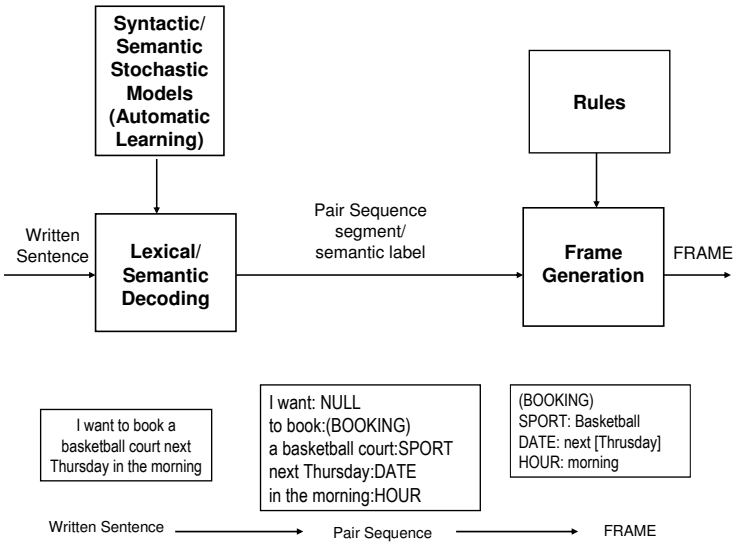

Figure 2: *Understanding module diagram.*

The main feature of our approach to speech understanding (García et al., 2011), (Ortega et al., 2010), (Segarra et al., 2002) is the integration of syntactic and semantic constraints in a single stochastic finite-state automaton.

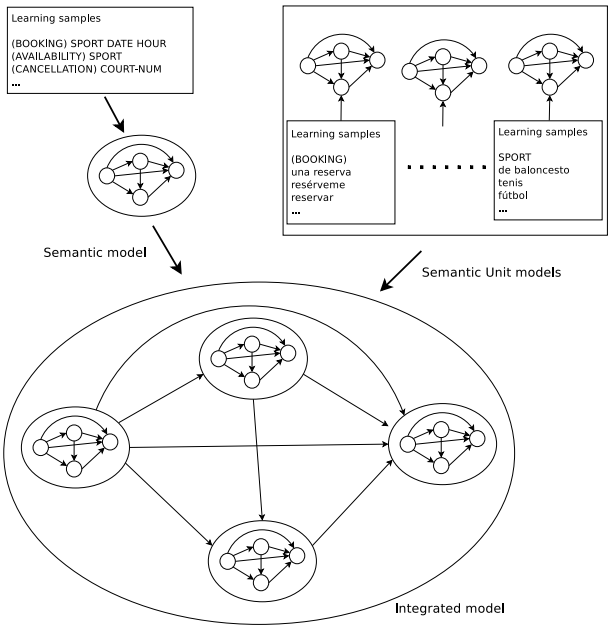

Figure 3: *Understanding model.*

The chosen semantic representation is the frame, which consists of a set of concepts and their associated attribute-value pairs. This representation is especially useful when the application runs in an information service that requires access to a database, so the understanding module must complete a set of fields to accomplish a requirement (slot-filling). The understanding module works in two phases as Figure 2 shows.

The first phase consists of a transduction of the input sentence in terms of an intermediate semantic language. In the example in Figure 2, the sentence *I want to book a basketball court next Thursday in the morning* is translated as follows:

Intermediate Semantic Representation:
I want: NULL
to book: (BOOKING)
a basketball court: SPORT
next Thursday: DATE
in the morning: HOUR

In the second phase, a set of rules translates this intermediate representation in terms of frames. As the intermediate language is close to the frame representation,

this phase only requires a small set of rules to construct the frame. This second phase consists of the following: the deletion of irrelevant segments of the input sentence; the reordering of the relevant concepts and attributes that appeared in the user sentence following an order which has been defined a priori; the automatic instantiation of certain task-dependent values, etc. This last action consists of the conversion of dates and hours into their canonical values. For example, "on September the 14th" into "[Sunday-2014-14-09]".

In this phase, the sentence *I want to book a basketball court on September the 14th in the morning* is translated as follows:

Semantic Representation (frame):
(*BOOKING*)
SPORT: basketball
DATE: [Sunday-2014-14-09]
HOUR: morning

The goal of the first phase is to find the best sequence of semantic units given the input sentence; a two-level statistical semantic modelization is used in this phase (Segarra et al., 2002). Figure 3 shows a schema of the two-level statistical semantic model. The meaning of each sentence is represented as a sequence of semantic units, and a segmentation of the sentence in terms of the corresponding semantic units is associated to it. From an annotated training corpus, we learn two kind of models: one of them (the Semantic Model) represents the concatenations of semantic units, and the other (Semantic Unit Model) represents the lexical realization of each semantic unit (i.e., the model of sequences of words associated to each semantic unit). In both cases, the models used for this task are bigrams, (i.e., bigrams of semantic units and bigrams of words into each semantic unit). The decoding process consists of a Viterbi search over the integrated network that supplies not only the best sequence of semantic units but also the segmentation of the input sentence associated to that best sequence. This segmentation is used in the second phase of the semantic module to associate the values to the attributes after a normalization process.

Let W be the vocabulary of the task, and let V be the set of intermediate semantic units. Given the input sentence $w = w_1 w_2 \cdots w_n \in W^*$, the understanding process consists of finding the sequence of semantic units $\hat{v} = v_1 v_2 \cdots v_m \in V^*$ that maximizes the probability:

$$\hat{v} = \underset{v \in V^*}{\operatorname{argmax}} P(w|v)P(v)$$

The term $P(w|v)$ is the probability of the sequence of words w given the sequence of semantic units v. We estimate this probability (following the Viterbi

algorithm) as the maximum for all segmentations of w in m segments.

$$P(w|v) = \max_{\forall l_1, l_2, \ldots, l_{m-1}} \Big\{ P(w_1, \ldots, w_{l_1}|v_1) \cdot P(w_{l_1+1}, \cdots, w_{l_2}|v_2) \cdot \cdots \\ \cdots P(w_{l_{k-1}+1}, \ldots, w_{l_k}|v_k) \cdot \cdots \cdot P(w_{l_{m-1}+1}, \ldots, w_n|v_m) \Big\}$$

where, $1 < l_1 < l_2 < \ldots < l_k < \ldots < l_{m-1} < n$.

If bigram models are used, the probability of a segment of words $w_{l_{k-1}+1}, \cdots, w_{l_k}$ given the associated semantic unit v_k, can be expressed as:

$$P(w_{l_{k-1}+1}, \ldots, w_{l_k}|v_k) = \prod_{i=l_{k-1}+1}^{l_k} P(w_i|w_{i-1}, v_k)$$

And, the term $P(v)$ is the bigram probability of the sequence v:

$$P(v) = \prod_{j=1}^{m} P(v_j|v_{j-1})$$

To learn the understanding models, a corpus of segmented and labeled sentences is needed. Each sentence of the corpus must be segmented and each of its segments must have a semantic label assigned to it. The label assigned to each segment represents a semantic interpretation of that segment. From a segmented and labeled corpus two types of finite-state models are estimated: the Semantic Model (an automaton) is estimated from sequences of semantic labels; and a Semantic Unit Model (an automaton) for each semantic label is estimated from all segments of words associated to that semantic label.

In order to perform the understanding process, an integrated model (automaton) is generated by replacing each state in the semantic model with the automaton that corresponds to the semantic label represented by that state.

2.2. Dialog Manager

In a dialog system, the Dialog Manager (DM) is the module that is responsible for choosing the best system answer at each dialog turn. The dialog model proposed in this paper is based on the transduction concept and on the use of Stochastic Finite-State Transducers. Given a state of the model and a user turn, the DM generates an answer and a transition to a new state is done. Dialog management is therefore based on modeling sequences of pairs: user turns and system turns. A dialog can be viewed as a path in the transducer model from its initial state to a final state in which the user has reached his goal.

The *DM* model covers the most common user behavior. However, during a dialog with real users, it needs to deal with unexpected situations. To address this problem, the *DM* is able to dynamically increase the coverage of the dialog model to cover those situations that are not seen in training. The dialog model and the strategy for increasing the coverage of this model are two of the main contributions of this work and are detailed in following sections.

The system answers are represented in terms of system dialog acts. An example of a system dialog act is shown below:

System dialog act:
(*CONFIRMATION-ACTION-BOOKING*)
SPORT: basketball
DATE: tomorrow
HOUR: 10:00
COURT-NUMBER: 3

From this system dialog act, the Answer Generator would generate the sentence *If you want to play basketball tomorrow, court number 3 is available at 10:00 . Do you want to book this court?*.

2.3. Application Manager

In some dialog systems, the dialog manager makes its decisions based only on the information provided by the user in the previous turns and its own model. The main difference between some of these slot-filling tasks and the EDECAN-SPORTS task is that, in this last task, the dialog manager not only provides information but also modifies the application data. Therefore, the dialog manager generates the following system answer taking into account both the information provided by the user and the information generated by the module that controls the sport facilities booking application, which we call the Application Manager (*AM*).

The *AM* performs the queries to the information system and updates it when necessary. For example, the information system for the EDECAN-SPORTS task must be updated when `BOOKING` or `CANCELLATION` actions have been performed.

In dialog tasks of this kind, the result of the queries to the AM has to be considered in order to generate the system answer. For instance, in order to book the facilities, if there is no court available, the system can suggest a change in the user restrictions (i.e., the *AM* verifies whether or not it is possible to perform the booking by changing the hour). If only one court is available, the system confirms whether or not everything is correct before making the booking (as shown in the previous example). Finally, if there is more than one court available, the system asks which court should be booked.

2.4. Multimodal interface

Once the DM has decided the system answer, the answer is sent to the Answer Generator, which is responsible for generating two different types of answers: oral answers and graphical answers. For the generation of an oral answer, a set of predesigned templates is used; with these templates, the Answer Generator generates a sentence that is sent to the Text-to-Speech (TTS) module. The generation of the graphic answer is a bit more complex. First, the information provided by the DM is processed to generate a set of graphical elements. Second, semantic information is added to some graphical elements, mainly buttons; this semantic information is sent back to the DM if the user presses the button. Thus, the semantic interpretation of the user actions on the graphical interface is fairly easy. Finally, the semantically labeled graphical information is sent to the Graphical Interface module which will be responsible for displaying it on the screen. Figure 4 shows a view of the screen at a moment of the interaction.

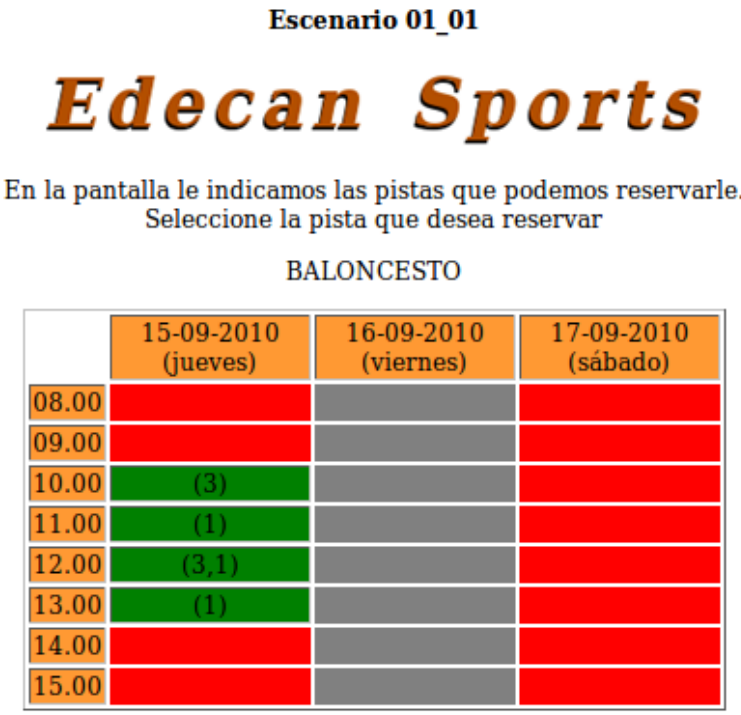

Figure 4: *Screenshot of the visual information for the user.*

3. The EDECAN-SPORTS task

A sport facilities booking task was defined within the framework of the EDECAN project (Lleida et al., 2006). This task is called EDECAN-SPORTS. It consists of mixed-initiative dialogs where users can book a court, cancel an existing booking, search for available facilities, or view their own bookings.

The semantic representation of user turns is a frame structure that includes the different functionalities required for the task: a set of 4 task-dependent concepts representing user intentions (`BOOKING`, `CANCELLATION`, `AVAILABILITY`, `BOOKED`), and a set of 3 task-independent concepts (`ACCEPTANCE`, `REJECTION`, `NOT-UNDERSTOOD`). Up to 6 attributes can be attached to each concept (`SPORT`, `HOUR`, `DATE`, `COURT-TYPE`, `COURT-NUMBER`, `ORDER-NUMBER`).

An example of the semantic interpretation of an input sentence is shown below:

User Turn:
I want to book a basketball court for tomorrow morning.

Semantic Representation:
(`BOOKING`)
`SPORT`: basketball
`DATE`: tomorrow
`HOUR`: morning

Dialog Manager answers are represented using a set of 21 actions. There are actions for opening and closing the dialog, confirming user supplied attributes, asking for more information, or showing information to the user. Each action can have some attributes with the same names as the user frame attributes.

An example of the labeling of a dialog manager turn is shown below:

Dialog Manager Turn:
If you want to play basketball tomorrow, court number 3 is available at 10:00 . Do you want to book this court?

Semantic Representation:
(`CONFIRMATION-ACTION-BOOKING`)
`SPORT`: basketball
`DATE`: tomorrow
`HOUR`: 10:00
`COURT-NUMBER`: 3

The EDECAN-SPORTS corpus (Hurtado et al., 2012) consists of a set of 165 dialogs for the EDECAN-SPORTS task that was acquired using the Wizard of Oz (WOz) technique and a specific software platform (Segarra et al., 2010). During the corpus acquisition process, a specific WOz was used to play the role of the natural language understanding module and a second WOz was used to control the dialog manager. As a result of this acquisition process, we not only obtained the dialog corpus, we also obtained the dialog acts corresponding to the labeling of the user and system turns.

An initial set of 165 dialogs by 16 different speakers from different origins was acquired for this task. The languages involved in the acquisition were Spanish and Catalan. A set of 15 types of scenarios was defined in order to cover all the expected use cases of the task. Table 1 shows the main characteristics of the EDECAN-SPORTS corpus: number of speakers, number of dialogs, number of user turns, average user turns per dialog, number of words, vocabulary size, and average words per user turn. Some characteristics of the semantic representation of the EDECAN-SPORTS corpus are also shown: the total number of user semantic labels (concepts and attributes), the total number of system semantic labels (concepts and attributes), the size of the user semantic vocabulary (frame units), and the size of the system semantic vocabulary (frame units).

Table 1: Main characteristics of the EDECAN-SPORTS corpus

	Spanish	Catalan
Number of speakers	15	3
Number of dialogs	137	28
Number of user turns	687	144
Average user turns per dialog	5.01	5.14
Number of words	4,740	995
Vocabulary size	335	179
Average words per user turn	6.90	6.91
Number of user concepts & attributes	1,283	285
Number of system concepts & attributes	954	212
User semantic vocabulary size	13	13
System semantic vocabulary size	21	19

In this work, the EDECAN-SPORTS corpus has only been used for testing the dialog models that were learned from synthetic dialogs generated using automatic dialog generation.

4. Statistical dialog modeling and Stochastic Finite-State Transducers

We have developed a DM based on the statistical modelization of the sequences of dialog acts (user and system dialog act pairs). Let (u_i, a_{i+1}), $i = 1...n$ be a pair where u_i is the user utterance at turn i and a_{i+1} is the system answer to this turn. We consider that a dialog is a sequence of pairs as follows:

$$a_0, (u_0, a_1), (u_1, a_2), (u_2, a_3), \ldots, (u_i, a_{i+1}), \ldots, (u_n, a_{n+1})$$

For each time i, the best system answer $\hat{a}_{i+1}$ can be selected with a local process which takes into account the sequence of dialog pairs preceding time i. This sequence is the last user turn u_i and all the information provided in previous turns either by the system or the user. This selection is made by maximizing:

$$\hat{a}_{i+1} = \underset{a \in A}{\operatorname{argmax}} P(a | a_0, u_0, a_1, \ldots, a_i, u_i) \tag{1}$$

where A contains all the possible system answers. The system answer $\hat{a}_{i+1}$ is selected taking into account both u_i and all the information provided by the user and the system throughout the entire dialog sequence.

The estimation of these probabilities requires an exponential number of dialogs. Since the number of all possible sequences of pairs is very large, we establish a partition in the space of the history of the dialog that precedes time i. To do this, the concepts of Dialog Register (DR) and Dialog State (DS) are introduced.

The DR is defined as a data structure that contains the information about concept and attribute values that are provided by the user throughout the previous history of the dialog.

All the information captured by the DR at a given time i (DR_i) is a summary of the information provided by the sequence:

$$a_0, (u_0, a_1), (u_1, a_2), (u_2, a_3), \ldots, (u_{i-1}, a_i)$$

Taking into account the concept of the DR, we establish a partition in the space of sequences of states such that, at each time i, two different sequences are considered to be equivalent if they lead to the same DR_i. We obtain a great reduction in the number of different histories in the dialogs at the expense of a loss in the chronological information. We consider this to be a minor loss, since, although the way in which a user provides the information may be useful in determining the behavior of the system, once a dialog state is reached, it is much more important *what* information was supplied than *in what order* this information was supplied. In addition, in our approach the previous user turn is explicitly taken into account, as we will be seen below.

Furthermore, we assume that the specific value of an attribute in the DR is not needed to calculate the next system answer. Thus for each attribute, we are only interested in knowing whether or not it has been provided during the dialog and its confidence value. To further reduce the variability in the value space of DR, we quantify the confidence value of each attribute according to a confidence threshold, and consider only three values: a) the attribute is not provided `-not-supplied-`, b) the confidence of the attribute is lower than the threshold `-low-confidence-`, and c) the confidence of the attribute is higher than the threshold `-high-confidence-`. Therefore, the information we use from the DR is a codification of each of its fields in terms of these three values. The value of the confidence threshold must be experimentally determined, taking into account the performance of the ASR and understanding module.

User turns supply the system with information about the task; that is, the user asks for a specific concept and/or provides specific values for certain attributes. However, a user turn could also provide other kinds of information, such as task-independent information. This is the case of turns corresponding to `AFFIRMATION`, `NEGATION`, and `NOT-UNDERSTOOD` dialog acts. This kind of information implies decisions that are different from simply updating the DR. In order to take into account all the information required by the Dialog Manager to select the next system answer, the concept of Dialog State (DS) is introduced. At a given time i, the DS_i is defined as a data structure that includes: a) the Dialog Register at time i, DR_i; b) the last request to the application manager; c) the cardinality of the last Application Manager result (with four possible values: `no-row`, `one-row`, `two-rows`, `three or more rows`); and d) the last system answer a_i.

Therefore, using the concept of DS, the maximization in Equation 1 can be approximate as:

$$\hat{a}_{i+1} = \underset{a \in A}{\operatorname{argmax}} P(a|DS_i, u_i) \tag{2}$$

4.1. The Dialog Manager

The dialog model presented in this work is based on the transduction concept and on the use of Stochastic Finite-State Transducers ($SFST$) (Casacuberta and Vidal, 2004). In our proposal, given a state of the dialog and a user turn, a system turn is generated and a transition to a new state is done. Therefore, dialog management is based on the modelization of the sequences of system and user dialog turn pairs. Thus, a dialog describes a path in the transducer model from its initial state to a final one.

A SFST can be defined by a 6-tuple $(Q, \Sigma, \Delta, q_0, p, f)$, where Q is a set of states, Σ is the input alphabet, Δ is the output alphabet, q_0 is the initial state,

$p : Q \times \Sigma \times \Delta \times Q \rightarrow [0, 1]$ is the transition probability distribution, and $f : Q \rightarrow [0, 1]$ is the final-state probability distribution.

The adaptation of a $SFST$ model for dialog management assumes that:

- Q contains all the possible dialog states represented as stated above (DS).
- The input alphabet Σ contains all the allowed user utterances (i.e., all the allowed combinations of user dialog acts).
- The output alphabet Δ is the set of system dialog acts that represents all the possible system answers (the set A in the previous maximization).
- q_0 is the "welcome to the system" state. All dialogs start at this state.
- Since the destination state (q') of a transition is univocally defined with the origin state (q) and the user input (u), the transition probability is independent of the destination state, and $p(q, u, a, q') = p(q, u, a)$. Therefore, p can be estimated from a labeled dialog corpus as:

$$p(q, u, a) = P(a|q, u) = \frac{C(q, u, a)}{C(q, u)}$$

  where $C(q, u, a)$ is the number of times that, being at dialog state q and observing the user utterance u, the system answer was a; and $C(q, u)$ is the number of times that, being at dialog state q, the user utterance u was observed.

- $f(q) = 1$ for all states that can be reached with a system answer that involves the end of the dialog. For those states in which the dialog remains active, $f(q) = 0$.

The selection of the best system answer at time i (a_i) is made by means of the following local maximization:

$$\hat{a}_{i+1} = \underset{a \in A}{\operatorname{argmax}} P(a|DS_i, u_i) = \underset{a \in \Delta}{\operatorname{argmax}} P(a|q_i, u_i)$$

This approach considers a dialog (a sequence of user and system dialog acts) to be a path in a $SFST$ from the initial state to one of the final states. A sub-dialog can be viewed as a path from a dialog state (not necessarily the initial state q_0) to another dialog state (not necessarily a final state).

5. Automatic Dialog Generation

Due to the high cost of acquiring dialogs with real users, an Offline Dialog Simulator was developed in order to obtain a large number of labeled dialogs. The Offline Dialog Simulator has no prior information about user behavior. An arbitrary goal from the task domain must be chosen at the beginning of each dialog simulation.

In the EDECAN-SPORTS task, four general goals were defined:

- To see the available courts,
- To book a court,
- To see the courts booked by the user,
- To cancel a booking.

These goals correspond to the concepts defined for the frame representation of the user turns: `AVAILABILITY`, `BOOKING`, `BOOKED`, and `CANCELLATION`. To establish the goal, a random choice among these concepts and a random choice among the attributes associated to each concept are made. For example, the attributes associated to the concept `BOOKING` are `SPORT`, `DATE`, `HOUR`, `COURT-TYPE`, `COURT-NUMBER`. Since it is not necessary to generate the exact values of the attributes (it is enough to state whether or not the attributes exist), we assign the label `CORRECT` as value to each chosen attribute to indicate that this value is supplied. For example, for the concept `BOOKING`, the following goal could be chosen:

`BOOKING`
  `SPORT`:*CORRECT*
  `DATE`:*CORRECT*

It is possible to generate a multiple goal scenario by combining several concepts, as sometimes occurs in real dialogs. This scenario can be viewed as different goals that must be sequentially accomplished in the same dialog. An example of a multiple goal scenario is shown below. It corresponds to the `CANCELLATION` of a court and the posterior `BOOKING` of another one.

`CANCELLATION`
  `SPORT`:*CORRECT*
`BOOKING`
  `SPORT`:*CORRECT*
  `DATE`:*CORRECT*

Once the goal of the dialog is chosen, it starts an iterative process which involves the following components: a user simulator, a communication channel simulator, and a dialog manager simulator.

- At each turn during the simulation, the user simulator randomly generates a user dialog act. This is done by randomly selecting some concepts and a subset of attributes for each one of the selected concepts. Random confidence values in $[0, 1]$ are attached to each concept and attribute. It should be noted that it is not necessary to choose real values for the attributes; a generic label *CORRECT* is associated to each attribute.

  An example of a user dialog act generated by the user simulator is:

  `BOOKING` 0.75

  `SPORT`:*CORRECT* 0.67

  `DATE`:*CORRECT* 0.32

  `HOUR`:*CORRECT* 0.46

- The communication channel simulator: In order to simulate errors that can appear in real dialogs, (i.e., speech recognition errors or misunderstandings), each user turn is modified using the communication channel simulator. The decision to introduce an error is taken individually for each concept and attribute. The mechanism for introducing errors consists of generating a new random value in the range $[0, 1]$ for each concept and attribute in the turn. If this new value is greater than the confidence value assigned to the same concept or attribute by the user simulator, an error is introduced in the concept or attribute.

  When the error concerns a concept, the concept is randomly changed for another concept. When the error concerns an attribute, the label *CORRECT* is replaced by the label *ERROR*. Although an error is introduced, the confidence value is not modified. That is, the value assigned by the user simulator is kept. This ensures that low confidence elements are more likely to be confused than higher confidence ones.

  An example of a simulated user dialog act after being processed by the communication channel simulator is:

  `BOOKING` 0.75

  `SPORT`:*CORRECT* 0.67

  `DATE`:*ERROR* 0.32

HOUR:*CORRECT* 0.46

In this example, the value of the attribute DATE has been changed because its confidence value was lower than the random value generated by the communication channel simulator.

The inclusion in the training set of user turns with errors (as shown in the previous example) generates dialog samples where errors have to be corrected. Therefore, the Dialog Manager model can learn strategies to tackle misunderstandings.

- The dialog manager simulator chooses an action at random from the set of actions defined for the task. If the selected action requires a database query, a random value in {0, 1, 2+} is used to represent the answer cardinality of the query result. No channel simulation is used for system turns.

This approach to automatic dialog simulation is easy to implement because it only requires the semantic definition of the task (i.e., the structure of user and system dialog acts), and a validation function to determine the correctness of the generated dialog.

The correctness of each simulated dialog must be evaluated. A validation function that considers a set of correctness criteria checks whether or not the dialog is coherent and the user goal is met. This function performs some tests on the turn sequence and the user goal and rejects every dialog that fails at least one test. We have defined a set of simple criteria in order to automatically determine the correctness of a dialog. A dialog is considered to be unsuccessful if one of the following conditions takes place:

- The dialog exceeds the maximum number of turns. The maximum number of turns allowed for the simulated dialogs is defined to be a slightly higher than the average number of turns of the dialogs acquired with real users.
- The dialog manager simulator has modified the information system with information marked as *Error* or with attributes not present in the user goal.
- The dialog manager simulator has chosen an action that needs information not provided by the user. For example: asking for a confirmation of an attribute that the user has not said.
- The dialog manager simulator performs an update of the application information system not related to the user goal.

Otherwise, the dialog is considered correct and is added to the dialog corpus.

Even though thousands of random dialogs are needed to get a few valid dialogs, the procedure is fast enough to obtain a large corpus in a reasonable amount of time and the procedure is easy to parallelize. Only valid dialogs are considered for estimating the model.

6. Online Generated Transducer

Given enough dialogs, every state should have been visited enough times to achieve a reliable parameter estimation and, in the limit, the estimated model could manage all possible dialog situations. However, this is an unrealistic scenario because there is a huge dialog state space, and, therefore, the number of training samples to get full coverage of the model is impossible to generate.

The coverage problem arises in dialog management because, when interacting with a real user, the $SFST$ can lead to a state in which the current user turn has never been seen in the training samples and, therefore, the model has no information on what action should be performed next. The possibility of this kind of situations arising is increased by the fact that, even though the Offline Dialog Simulator generates some training samples that have errors, new ASR or understanding errors could appear during an interaction with real users.

In a previous implementation, an ad-hoc heuristic smoothing approach was proposed (Hurtado et al., 2010). It was based on the definition of a distance measure between states. Thus, when the $SFST$ arrived to an unseen situation, the closest state in the model was searched, and a transition to this state was performed and the dialog continued from this new state. This approach solved the coverage problem in some cases, but it was restricted to working with the existing dialog states in the model, and no additional states or transition could be added dynamically based on real interactions. This approach also had problems when more than one unseen state was found consecutively because that means that there was an important difference between the belief of the user and the state of the system.

The new approach proposed in this work is oriented to dynamically adapting the model when interacting with real users; it is based on the use of the Offline Dialog Simulator. Given a user turn, when the current dialog achieves a state where the model has no information on what to do next, the dialog simulator is used. At this point, the simulator is used to generate a small corpus of dialogs with which new states and transitions can be created.

The dialog simulator procedure allows us to generate a set of labeled samples given some fixed turns or a sub-dialog. In other words, the simulator can be used to obtain a corpus of dialogs that have the same sequence of turns ending at a

chosen position. If the first n turns of the dialog are fixed, a set of correct endings for that situation can be generated. Using this idea, we have added the Offline Dialog Simulator to the $SFST$ model. From now on, we refer to this model as the Online Generated Transducer (OGT).

Let $a_0, u_0, a_1, u_1, \ldots, u_i$ be the initial sequence of user turns and system turns of a dialog that reaches a state q_i for which u_i is an unseen situation. These prefix turns are used to simulate a corpus of N synthetic sub-dialogs that begin from state q_i and whose first user turn is u_i. As we discussed in the previous section, in order to determine the correctness of a simulated dialog it is necessary to know the user goal. However, in real dialogs the user goal is unknown by the system. Instead of using the real user goal, the dialog simulator uses an approximation to this goal based on the dialog history.

From this corpus, the dialog manager obtains a set of possible actions for q_i: $A_{i+1} = \{a_{i+1_1}, a_{i+1_2}, \ldots, a_{i+1_N}\}$. Then, the transducer output for (q_i, u_i) is the most likely action.

$$\hat{a}_{i+1} = \underset{a \in A_{i+1}}{\operatorname{argmax}} P(a|q_i, u_i)$$

In the implementation, the next action is selected using the simulated dialogs; however, the model parameters are not updated at that moment because the system cannot determine whether or not the selected action is correct. The information is saved and a coherence test is used at the end of the dialog to determine whether or not the action was correct. A requirement for this approach is that the Online Generated Transducer needs to be fast enough to accomplish a fluent interaction with the user. It is not realistic to expect to simulate one thousand dialogs each time the model gets to an unseen state because a typical user is not going to wait more than a few seconds for a system answer. Therefore a small N needs to be chosen.

7. Evaluation

In order to evaluate the performance of the Dialog Manager based on $SFST$ and the Automatic Dialog Simulation techniques (both the online and the offline techniques) presented in this paper, we carried out three different experiments. Throughout all the experimentation, we used the Offline Dialog Simulator (described in section 5) to generate the training corpora.

7.1. Evaluation of the Dialog Manager based on SFST

The first experiment consisted of a study of the coverage achieved by a dialog manager based on $SFST$. In this experiment, no technique for dealing with unseen

situations was used. The coverage of the DM was studied for a set of models that were learned using a growing number of dialogs that were automatically generated using the Offline Dialog Simulator technique presented in Section 5. The 137 Spanish dialogs (687 user turns) of the EDECAN-SPORTS corpus, in Section 3, were used as the test set for the experimentation.

Table 2 shows the increase in coverage when more dialogs are used for the $SFST$ model estimation. Each row of the Table represents the statistics of a different DM model. For each model, we show the number of dialogs that were used to learn it, the total number of states of the $SFST$, and the unseen situation rate for the test set, (i.e., the percentage of states from the test corpus that had not been founded in the model). The unseen ratio initially decreases when the number of dialogs used to learn the model is increased. However, over 100,000 dialogs no significant decrease in the unseen ratio is observed.

An additional measure was used to evaluate the correctness of the model. For every system turn in the dialogs of the test corpus, we also checked if the answer of the estimated DM was the same answer as the one selected by the WOz during the acquisition. This measure is presented in Table 2 as the Exact Turn Rate. Note that Exact Turn Rate is a lower bound of the correctness of the model because, at a given state, more than one action may be correct. For example, if there is more than one attribute with low confidence at a given time, confirming any of them would be correct. However, to calculate this measure, we only consider correct the action selected by the WOz.

Table 2: Ratio of out-of-model states when the number of correct dialogs used to estimate the SFST is increased.

Dialogs	Total states	Unseen rate	Exact Turn Rate
1,000	3,121	45.6	34.3
5,000	9,543	24.1	57.3
10,000	14,375	17.6	64.0
20,000	20,894	11.6	64.0
50,000	32,332	9.6	70.1
100,000	43,163	6.9	75.3
120,000	46,249	6.9	73.4
150,000	50,211	6.9	74.9
170,000	52,538	6.9	74.9
200,000	55,645	6.7	75.0

For the coverage, no significant improvement in the Exact Turn Rate measure was achieved using models learned with more than $100,000$ correct dialogs.

This experiment outlines the difficulty of covering every possible situation during training.

7.2. Comparison between an offline and an online generated transducer

Next, we tried to assess the convenience of the Online Generated Transducer for solving the problem of coverage, see Table 3. We generated 1,000 dialogs offline and created a SFST from them. This model covered only 67% of the situations of the test. We used this poorly-estimated model as a starting point for the Online Generator Transducer. We compared this model against the best model of the previous experimentation, that is the model learned with 200,000 dialogs shown in the last row of Table 2. This last model was generated completely offline. Even though it covered more than 93% of the situation that appeared in the test, it had no strategy for dealing with the 6.7% of unseen situations. If an unseen situation arose the system simply aborted the dialog.

To evaluate the performances of the two managers, we used the following measures:

- Out of model turn rate: the percentage of dialog turns in the test set for which the DM has no answer.
- Exact turn rate: the percentage of turns in the test set for which the answer of the transducer is equal to the WOz answer during the acquisition. It is the same measure used for the previous experimentation.
- Exact dialog rate: the percentage of dialogs in the test set for which the answer of the transducer is equal to the answer produced by the WOz during the real acquisition in all the dialog states.
- Completed dialog rate: the percentage of dialogs in the test set accepted by the transducer (i.e., the percentage of dialogs in the test set for which there is a path between the initial state and a final state in the transducer).

As in the previous experiment, the Spanish part of the EDECAN-SPORTS corpus was used as the test set.

Table 3 compares the SFST learned with 200,000 dialogs generated offline and without treatment of the unseen situations (offline column) against the SFST learned with only 1,000 dialogs but with the Online Generation technique (OGT column). Due to the lower size of the training set, the *OGT* system presents a higher *Out-of-model turn rate*. However, as Table 3 shows, the online model improves every measure. Both the *Exact turn rate* and the *Exact dialog rate* are considerably improved. Moreover, the new model was able to deal with every

Table 3: Evaluation with the EDECAN-SPORTS corpus

	Offline	OGT
Training dialogs	200,000	1,000
States	55,645	3,378
Out-of-model turn rate	6.7	32.9
Exact turn rate	75.0	87.1
Exact dialog rate	26.6	61.0
Completed dialog rate	73.4	100.0

possible situation, so it managed to finalize all the dialogs in the test set even though not every dialog met the user goal.

For this experiment, the OGT simulated 10 dialogs for each *Out-of-model turn* ($N = 10$).

7.3. Evaluation of the OGT with real users

It is well known that the performance of a dialog system decreases when it is tested in real conditions as it is shown in (Ai et al., 2007) (Young et al., 2014). Some problems, such as misunderstanding errors produced by a noisy acoustic environment, or unexpected dialog situations due to the interaction with untrained real users, can lead the dialog manager to generate incorrect answers. This occurs because during the training process it is impossible to generate all the dialog situations, even if a simulation of recognition errors is done in order to increase the robustness of the model.

In order to test the performance of our system in close to real conditions, we carried out an evaluation of the OGT with real users using our Spoken Dialog System prototype (Segarra et al., 2010). A corpus of 120 dialogs was acquired from 12 users of the university staff. All these users were different from those participants in the acquisition of the EDECAN-SPORTS corpus (Section 3); these users were untrained but the experimentation was done in laboratory conditions.

To acquire each dialog, a scenario was selected and explained to the user. The user's mission was to achieve a goal by interacting with the dialog system. Examples of these goals are: to book a court on a day of the week or to check the user bookings. Some dialogs had complex goals with combinations of `BOOKING`, `CANCELLATION`, `AVAILABILITY`, and `BOOKED` that the user had to perform in a predefined order in order to achieve the global goal of the dialog.

Table 4 shows the results of the evaluation with real users. From 945 system turns, 233 were *Out-of-model turns* (28.9%) and triggered the Online Dialog Simulator. Every action resulting from this online simulation was locally correct or coherent, that is, it did not lead the dialog to an unrecoverable incorrect state like

booking a wrong court. Even with almost two unseen situations per dialog, the goal of the user was achieved in 94.2% of the dialogs.

Although the WER is lower than 6%, most of the times the recognition errors affect attribute values. When these kind of errors occurs it may be difficult to the Dialog Manager overcome the situation because it lacks information necessary to fulfill the goal of the dialog.

Table 4: Evaluation with real users

Number of users	12
Number of dialogs	120
ASR Word Error Rate	5.97
User turns	804
System turns	945
Out-of-model turns	233 (28.9%)
Successful dialogs	113 (94.2%)

The evaluation shows that the model is robust enough for real-life interaction and that possible recognition and understanding errors can be corrected during the dialog by using confirmations.

As stated above, two of the main problems when dialog systems are used in real conditions are: recognition errors affecting attribute values and unseen situations due to the interaction with naive untrained users. In our proposal, we have addressed the problem of recognition errors by means of confidence scores. Each attribute/value pair provided by the user has associated a confidence score. Often, low confidence in some attribute forces the DM to ask for explicit confirmations. Regarding the unseen dialog situations, they can be sometimes solved by the online generation process that selects an system action that allows the DM for carrying on with the dialog.

The following example, Figure 5, illustrates how the online dialog generation system works when unseen situations, due to recognition errors, appear. The example is a sequence of user and system turns. In the user turns it can be seen: the correct transcription of the user utterance (CT), the output of the Automatic Speech Recognizer (ASR) and the semantic representation of the utterance (SLU). In the system turns it can be seen: the query to the Application Manager (AM), the action selected by the Dialogue Manager (DM) and its translation into natural language (TTS).

It should be noted that in user turn 1 the recognizer fails in the value of an attribute (it recognizes "qual" instead of "squash"). The model triggers the OGT process and a question about the sport is selected.

Figure 5: *Example of a dialog using the online dialog generation system.*

DM0:	`(OPENING)`
TTS0:	**Bienvenido al servicio de consulta, reserva y anulación de pistas deportivas de la UPV. ¿En qué puedo ayudarle?** (*Welcome to the service of query, booking and cancellation of the sports facilities of the UPV. Can I help you?*)
CT1:	**quiero reservar una pista de squash** (*I want to book a squash court*)
ASR1:	**quiero reservar una pista de *cual*** (*I want to book a court which*)
SLU1:	`(BOOKING)` / 0.630433
DM1:	`(SPORT)` # selected by OGT
TTS1:	**¿A qué deporte quiere jugar?** (*What sport do you want to play?*)
CT2:	**jugar a squash** (*play squash*)
ASR2:	**jugar a squash** (*play squash*)
SLU2:	`()` `SPORT: squash` / 0.755294
AM2:	`(AVAILABILITY-QUERY)`
DM2:	`(SHOW-AVAILABILITY)`
TTS2:	**En la pantalla puede ver las pistas que tenemos disponibles de Squash** (*On the screen you can see the Squash courts we have available*)
CT3:	**quiero reservar una pista el sábado** (*I want to book a court on Saturday*)
ASR3:	**quiero reservar una pista el sábado** (*I want to book a court on Saturday*)
SLU3:	`(BOOKING)` / 0.717141 `DATE: saturday` / 0.836906
AM3:	`(AVAILABILITY-QUERY)`
DM3:	`(SHOW-AVAILABILITY)`
TTS3:	**En la pantalla puede ver las pistas que tenemos disponibles para el sábado** (*On the screen you can see the courts we have available for Saturday*)
CT4:	**quiero reservar a las trece horas** (*I want to book at thirteen hours*)
ASR4:	**quiero reservar a las trece *hora*** (*I want to book at thirteen hour*)
SLU4:	`(BOOKING)` / 0.407799 `HOUR: 01:00:PM` / 0.765674
AM4:	`(AVAILABILITY-QUERY)`
DM4:	`(CONFIRMATION-ACTION-BOOKING)`
TTS4:	**¿Desea reservar la pista de Squash seleccionada?** (*Do you want to book the selected squash court?*)
CT5:	sí (*yes*)
ASR5:	sí (*yes*)
SLU5:	`(ACCEPTANCE)` / 1.000000
AM5:	`(BOOKING-QUERY)`
DM5:	`(BOOKING-OK)`
TTS5:	**Ha reservado la pista seleccionada. ¿Desea algo más?** (*You have booked the selected court. Anything else?*)
CT6:	**no nada más** (*No nothing more*)
ASR6:	**no nada más** (*No nothing more*)
SLU6:	`(REJECTION)` / 0.669649
DM6:	`(CLOSING)`
TTS6:	**Gracias por utilizar este servicio** (*Thank you for using this service*)

8. Conclusions

In this paper, a proposal for the development of the statistical dialog manager component of a spoken dialog system has been presented. This proposal is based

on finite-state models, particularly SFSTs. Our approach tries to overcome the problem of the large state space in real tasks by means of condensing in a state the whole information provided by the user throughout the previous history of the dialog and codifying the attribute values in terms of 3 labels: not-supplied, low-confidence, or high-confidence.

One of the main characteristics of this approach is that the learning process can be performed in an unsupervised way. Not only is the dialog manager model automatically estimated from a dialog corpus, but the labeled corpus is also artificially generated by a user/system simulation process. This way, since a manually labeled corpus acquired with real users is not required, the design of statistical dialog systems for new applications becomes faster and cheaper. In our approach, for the estimation of the dialog manager model it is only necessary to define the semantics of the task and the system/user dialog acts.

We have also addressed the problem of the lack of coverage of the model by means of an online mechanism for sub-dialog generation. This mechanism was useful to find new valid paths in the transducer to achieve the user goals. This allows, all the situations in the dialogs to be answered by the dialog manager. The experimental results show the good behavior of the proposals when real users interact with the system and when real unseen situations arise.

In order to compare our proposal for the development of the Dialog Manager component with other approaches (e.g., the most commonly used approach based on POMDPs models and reinforcement learning as the estimation paradigm), a standard framework of dialog competitions would be necessary. Unfortunately, the definition of an homogeneous framework for the experiments, which includes the user behavior, the language understanding module, the ASR, etc.., is a very complex task; however, it could be the subject of future works.

Some issues regarding the differences between our proposal and POMDPs can be studied in the future. One of the differences is that while POMDPs keep all possible states with their attributes values active in the belief state, in the current proposal, we model the uncertainty by means of a confidence score for each attribute-value pair. Thus, only one state is active at each point of the dialog, allowing our approach to deal with realistic tasks easily. In addition, the proposed approach could be enriched by introducing a reward element as in the POMDPs as a direction for future work.

Acknowledgements

This work is partially supported by the project ASLP-MULAN: Audio, Speech and Language Processing for Multimedia Analytics (MINECO TIN2014-54288-C4-3-R).

Ai, H., Litman, D. J., 2011. Comparing user simulations for dialogue strategy learning. TSLP 7 (3), 9.

Ai, H., Raux, A., Bohus, D., Eskenazi, M., Litman, D. J., 2007. Comparing Spoken Dialog Corpora Collected with Recruited Subjects versus Real Users. In: SIGdial Workshop on Discourse and Dialogue. pp. 124–131.

Black, A., Burger, S., Langner, B., Parent, G., Eskenazi, M., 2010. Spoken Dialog Challenge 2010. In: Proc. of the SLT 2010. Berkeley, USA, pp. 448–453.

Casacuberta, F., Vidal, E., 2004. Machine translation with inferred stochastic finite-state transducers. Computational Linguistics 30 (2), 205–225.

Core, M. G., Allen, J., 1997. Coding dialogs with the damsl annotation scheme. In: AAAI fall symposium on communicative action in humans and machines. Boston, MA, pp. 28–35.

Crook, P. A., Lemon, O., 2011. Lossless Value Directed Compression of Complex User Goal States for Statistical Spoken Dialogue Systems. In: Proceedings of Interspeech 2011. pp. 1029–1032.

Cuayáhuitl, H., Renals, S., Lemon, O., Shimodaira, H., 2007. Hierarchical dialogue optimization using semi-markov decision processes. In: Proc. of the 8th Annual Conference of the International Speech Communication Association, INTERSPEECH'07. ISCA, Antwerp, Belgium, pp. 2693–2696.

Cuayhuitl, H., Renals, S., Lemon, O., Shimodaira, H., 2011. Learning Multi-Goal Dialogue Strategies Using Reinforcement Learning With Reduced State-Action Spaces. In: Proceedings of Interspeech 2006. pp. 547–565.

Esteve, Y., Raymond, C., Bechet, F., Mori, R. D., 2003. Conceptual Decoding for Spoken Dialog systems. In: Proc. of European Conference on Speech Communications and Technology (Eurospeech'03). Vol. 1. Geneva (Switzerland), pp. 617–620.

García, F., Hurtado, L.-F., Sanchis, E., Segarra, E., 2011. An active learning approach for statistical spoken language understanding. In: Proc. of 16th Iberoamerican Congress on Pattern Recognition, CIARP'11. Pucón (Chile), pp. 565–572.

Gašić, M., Jurcicek, F., Thomson, B., Yu, K., Young, S., 2011. On-line policy optimisation of spoken dialogue systems via live interaction with human subjects. In: Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on. IEEE, pp. 312–317.

Gašić, M., Keizer, S., Mairesse, F., Schatzmann, J., Thomson, B., Yu, K., Young, S., 2008. Training and Evaluation of the HIS POMDP Dialogue System in Noise. In: Proceedings of the 9th SIGdial Workshop on Discourse and Dialogue. Association for Computational Linguistics, Association for Computational Linguistics, pp. 112–119.

Georgila, K., Henderson, J., Lemon, O., 2005. Learning User Simulations for Information State Update Dialogue Systems. In: Proc. of the 9th European Conference on Speech Communication and Technology (Eurospeech'05). Lisbon (Portugal), pp. 893–896.

Griol, D., Callejas, Z., López-Cózar, R., Riccardi, G., 2014. A domain-independent statistical methodology for dialog management in spoken dialog systems. Computer Speech and Language 3 (28), 743–768.

Griol, D., Hurtado, L. F., Segarra, E., Sanchis, E., 2008. A statistical approach to spoken dialog systems design and evaluation. Speech Communication 50 (7-9), 666–682.

Hahn, S., Dinarelli, M., Raymond, C., Lefèvre, F., Lehnen, P., De Mori, R., Moschitti, A., Ney, H., Riccardi, G., 2010. Comparing stochastic approaches to spoken language understanding in multiple languages. Audio, Speech, and Language Processing, IEEE Transactions on 6 (99), 1569–1583.

He, Y., Young, S., 2003. A data-driven spoken language understanding system. In: Proc. of IEEE Automatic Speech Recognition and Understanding Workshop (ASRU'03). St. Thomas (U.S. Virgin Islands), pp. 583–588.

Henderson, J., Lemon, O., Georgila, K., 2008. Hybrid reinforcement/supervised learning of dialogue policies from fixed data sets. Computational Linguistics 34 (4), 487–511.

Hori, C., Ohtake, K., Misu, T., Kashioka, H., Nakamura, S., 2009. Recent Advances in WFST-based Dialog System. In: Procs. of InterSpeech'09. Brighton, UK, pp. 268–271.

Hurtado, L.-F., García, F., Sanchis, E., Segarra, E., 2012. The acquisition and dialog act labelling of the edecan-sports corpus. In: Proc. of LREC'12. Istambul, Turkey, pp. 1416–1420.

Hurtado, L. F., Griol, D., Sanchis, E., Segarra, E., 2005. A stochastic approach to dialog management. In: Automatic Speech Recognition and Understanding, 2005 IEEE Workshop on. IEEE, pp. 226–231.

Hurtado, L. F., Griol, D., Segarra, E., Sanchis, E., 2006. A Stochastic Approach for Dialog Management based on Neural Networks. In: Proc. of the 9th International Conference on Spoken Language Processing (Interspeech/ICSLP). Pittsburgh (USA), pp. 49–52.

Hurtado, L. F., Griol, D., Segarra, E., Sanchis, E., 2007. A Statistical User Simulation Technique for the Improvement of a Spoken Dialog System. In: Proc. of 12th Iberoamerican Congress on Pattern Recognition, CIARP'07. Viña del Mar/Valparaiso (Chile), pp. 743–752.

Hurtado, L.-F., Planells, J., Segarra, E., Sanchis, E., Griol, D., 2010. A stochastic finite-state transducer approach to spoken dialog management. In: Proc. of InterSpeech'10. Makuhari, Japan, pp. 3002–3005.

Jurčíček, F., Thomson, B., Young, S., Jun. 2012. Reinforcement learning for parameter estimation in statistical spoken dialogue systems. Computer Speech and Language 26 (3), 168–192.

Keizer, S., Gašić, M., Jurcicek, F., Mairesse, F., Thomson, B., Yu, K., Young, S., September 2010. Parameter estimation for agenda-based user simulation. In: Proceedings of the SIGDIAL 2010 Conference. Association for Computational Linguistics, Tokyo, Japan, pp. 116–123.

Lee, C., Jung, S., Lee, D., Lee, G., 2007. Example-based error recovery strategy for spoken dialog system. In: Automatic Speech Recognition & Understanding, 2007. ASRU. IEEE Workshop on. IEEE, pp. 538–543.

Levin, E., Pieraccini, R., 1995. Concept-Based Spontaneous Speech Understanding System. In: Proc. of European Conference on Speech Communications and Technology (Eurospeech'95). Madrid (Spain), pp. 555–558.

Levin, E., Pieraccini, R., 1997. A stochastic model of human-machine interaction for learning dialog strategies. In: Proc. of European Conference on Speech Communications and Technology (Eurospeech'97). Rhodes (Greece), pp. 1883–1896.

Levin, E., Pieraccini, R., Eckert, W., 2000. A stochastic model of human-machine interaction for learning dialog strategies. In: IEEE Transactions on Speech and Audio Processing. Vol. 8(1). pp. 11–23.

Lleida, E., Segarra, E., Torres, M. I., Macías-Guarasa, J., 2006. EDECÁN: sistEma de Diálogo multidominio con adaptación al contExto aCústico y de AplicaciónN. In: IV Jornadas en Tecnología del Habla. Zaragoza (Spain), pp. 291–296.

Martinez, F. F., Ferreiros, J., Cordoba, R., Montero, J. M., San-Segundo, R., Pardo, J. M., 2009. A bayesian networks approach for dialog modeling: The fusion bn. In: Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP '09. IEEE Computer Society, Washington, DC, USA, pp. 4789–4792.
URL http://dx.doi.org/10.1109/ICASSP.2009.4960702

Meng, H. H., Wai, C., Pieraccini, R., 2003. The Use of Belief Networks for Mixed-Initiative Dialog Modeling. In: IEEE Transactions on Speech and Audio Processing. Vol. 11(6). pp. 757–773.

Minker, W., Waibel, A., Mariani, J., 1999. Stochastically-Based Semantic Analysis. Kluwer Academic Publishers, Dordrecht (Holland).

Ortega, L., Galiano, I., Hurtado, L.-F., Sanchis, E., Segarra, E., 2010. A statistical segment-based approach for spoken language understanding. In: Proc. of InterSpeech'10. Makuhari, Japan, pp. 1836–1839.

Paek, T., Horvitz, E., 2000. Conversation as action under uncertainty. In: Proc. of the 16th Conference on Uncertainty in Artificial Intelligence. San Francisco (USA), pp. 455–464.

Planells, J., Hurtado, L.-F., Sanchis, E., Segarra, E., 2012. An online generated transducer to increase dialog manager coverage. In: Proc. of InterSpeech'12. Portland , USA, pp. 1–4.

Raymond, C., Riccardi, G., 2007. Generative and discriminative algorithms for spoken language understanding. Proceedings of Interspeech 2007, 1605–1608.

Schatzmann, J., Weilhammer, K., Stuttle, M., Young, S., 2006. A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. The Knowledge Engineering Review 21 (2), 97–126.

Segarra, E., Hurtado, L., Gømez, J., García, F., Planells, J., Pastor, J., Ortega, L., Calvo, M., Sanchis, E., 2010. A prototype of a spoken dialog system based on statistical models. In: Proc. of FALA 2010. Vigo, Spain, pp. 243–246.

Segarra, E., et al., 2002. Extracting Semantic Information Through Automatic Learning Techniques. International Journal on Pattern Recognition and Artificial Intelligence 16 (3), 301–307.

Singh, S., Kearns, M., Litman, D., Walker, M., 1999. Reinforcement learning for spoken dialogue systems. In: Proc. of Neural Information Processing Systems (NIPS'99). Denver (USA), pp. 956–962.

Sutton, R. S., Barto, A. G., 1998. Reinforcement learning: An introduction. Vol. 1. Cambridge Univ Press.

Thomson, B., Young, S., 2010. Bayesian update of dialogue state: A pomdp framework for spoken dialogue systems. Computer Speech & Language 24 (4), 562–588.

Tür, G., Mori, R. D., 2011. Spoken Language Understanding: Systems for Extracting Semantic Information from Speech, 1st Edition. Wiley.

Williams, J., Raux, A., Ramachadran, D., Black, A., 2013. The dialog state tracking challenge. In: Proc. of the SIGDIAL 2013 Conference. Metz, France, pp. 404–413.

Williams, J., Young, S., 2007. Partially Observable Markov Decision Processes for Spoken Dialog Systems. In: Computer Speech and Language. Vol. 21(2). pp. 393–422.

Young, S., Breslin, C., Gai, M., Henderson, M., Kim, D., Szummer, M., Thomson, B., Tsiakoulis, P., Hancock, E. T., 2014. Evaluation of Statistical POMDP-based Dialogue Systems in Noisy Environment. In: Proc. of International Workshop Series on Spoken Dialogue Systems Technology (IWSDS'2014). pp. 50–61.

Young, S., Gašić, M., Keizer, S., Mairesse, F., Schatzmann, J., Thomson, B., Yu, K., Apr. 2010. The hidden information state model: A practical framework for POMDP-based spoken dialogue management. Computer Speech and Language 24 (2), 150–174.

Young, S., Schatzmann, J., Weilhammer, K., Ye, H., 2007. The Hidden Information State Approach to Dialogue Management. In: Proc. of 32nd IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP). Vol. 4. Honolulu, Haway (USA), pp. 149–152.