A Probabilistic Framework for Dialog Simulation and Optimal Strategy Learning

Olivier Pietquin, *Member*, *IEEE,* Thierry Dutoit, *Member*, *IEEE*

***Abstract*— The design of Spoken Dialog Systems cannot be considered as the simple combination of speech processing technologies. Indeed, speech-based interface design has been an expert job for a long time. It necessitates good skills in speech technologies and low-level programming. Moreover, rapid development and reusability of previously designed systems remains uneasy. This makes optimality and objective evaluation of design very difficult. The design process is therefore a cyclic process composed of prototype releases, user satisfaction surveys, bug reports and refinements. It is well known that human intervention for testing is time-consuming and above all very expensive. This is one of the reasons for the recent interest in dialog simulation for evaluation as well as for design automation and optimization.**

In this paper we expose a probabilistic framework for a realistic simulation of spoken dialogs in which the major components of a dialog system are modeled and parameterized thanks to independent data or expert knowledge. Especially, an Automatic Speech Recognition (ASR) system model and a User Model (UM) have been developed. The ASR model, based on articulatory similarities in language models, provides task-adaptive performance prediction and Confidence Level (CL) distribution estimation. The user model relies on the Bayesian Networks (BN) paradigm and is used both for user behavior modeling and Natural Language Understanding (NLU) modeling. The complete simulation framework has been used to train a reinforcement-learning agent on two different tasks. These experiments helped to point out several potentially problematic dialog scenarios.

***Index Terms*—Spoken dialog systems, speech processing, dialog simulation, dialog evaluation, dialog strategy learning.**

I. Introduction

DIALOG simulation became the subject of recent researches for lots of reasons. Initially, systems developed for simulating dialogs aimed at validating dialog or discourse models. They were essentially systems involving two artificial agents implementing a formal description of the model and communicating with each other. Their simulated exchanges were logged and compared with human-human dialogs [1].

During the Spoken Dialog System (SDS) design process, a series of prototypes is typically released and enhancements from one system to the next are made by collecting a corpus of dialogs between users and prototypes to evaluate their performance. One way to avoid prototype releases is to use Wizard-of-Oz (WOZ) techniques, but this still requires human users intervention. Thus, another application of dialog simulation by computer means is the evaluation of SDSs. A first way to do is to build handcrafted systems able to interact with the SDS by some means and to evaluate the quality of the resulting dialog sessions by using objective metrics [2]. While this technique avoids human involvement, it is driven by a set of fixed parameters and performs always the same in a given situation unless parameters are modified. So most powerful simulation systems for evaluation purpose are probabilistic. Data set expansion is also an application. Indeed, if user evaluations of one or more prototypes have already been realized, a data corpus is collected and is hopefully representative of possible interactions between users and prototypes. This corpus can be used to create a generative model of the SDS environment [3]. The resulting model is then theoretically able to produce new dialogs having the same statistical properties than those in the corpus and enables to evaluate unseen dialogs.

Finally, a last application is the simulation for optimal strategy learning purposes. Indeed, once a way to simulate many dialogs is available as well as a way to evaluate them, it is natural to think about exploiting this framework to learn optimal strategies from experience [4] [5]. The evaluation method is then used to build an optimality criterion. As the learning process requires lots of dialogs to converge toward an optimal strategy, computer-based simulation is really valuable. Indeed, real interactions between a learning agent and human users would be unfeasible since a large number of dialogs are needed; what is more, some of these dialogs can be long and sound unnatural, and the overall operation would be very expensive and time consuming.

In this paper we present a probabilistic framework for simulating dialogs based on parametric models of speech and language processing modules as well as on stochastic user modeling. This framework has been developed in the aim of evaluating dialog systems and learning optimal strategies but nothing prevents from using it for dialog model validation. It is based on a probabilistic description of a man-machine dialog, which

Manuscript received June 30, 2004 and revised December 1, 2004. This work was partly supported by the 'Direction Générale des Technologies, de la Recherche et de l'Énergie' (DGTRE) of the Walloon Region (Belgium, First Europe Convention n° 991/4351) as well as by the SIMILAR European Network of Excellence.

O. Pietquin is now with the Signal Processing Systems group at the Metz campus of the École Supérieure d'Électricité (Supélec), F-57070 Metz, France (e-mail: olivier.pietquin@ieee.org). This work was realized when he was with the Signal Processing department (TCTS Lab.) of the Faculty of Engineering, Mons, B-7000 Mons, Belgium

T. Dutoit is with the Signal Processing department (TCTS Lab) of the Faculty of Engineering, Mons, B-7000 Mons, Belgium (e-mail: thierry.dutoit@fpms.ac.be)

will be detailed in the two next sections.

II. Formal Description of Human-Machine Dialog

In the purpose of formalization, a Human-Machine Dialog (HMD) can be regarded as a sequential turn-taking process in which a human user and a Dialog Management (DM) system interact through speech and language processing subsystems (Figure 1) such as Automatic Speech Recognition (ASR), Natural Language Understanding (NLU), Natural Language Generation (NLG) and Text-To-Speech (TTS) synthesis subsystems. ASR and NLU modules are part of the speech input processing systems while NLG and TTS modules are part of the speech output generation systems. The DM usually has access to an external information source, the Knowledge Base (KB).

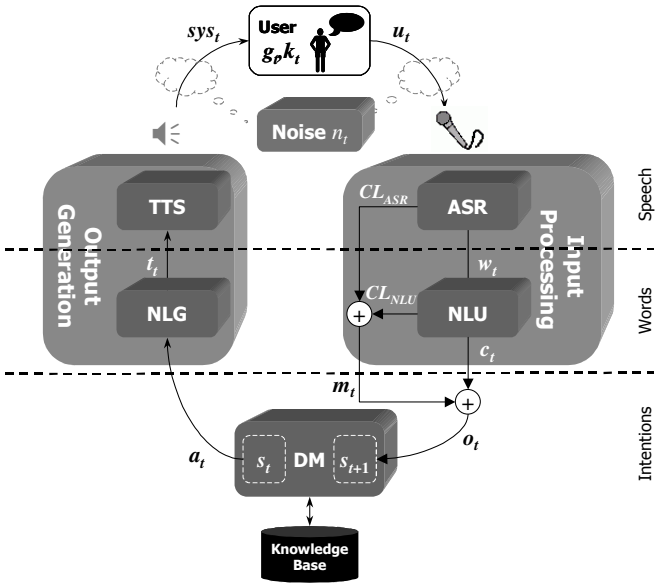


Figure 1: Communication during a Human-Machine Dialog

At each turn t, the DM generates a set of communicative acts $\boldsymbol{a_t}$ that have to be transmitted to the user. To do so the DM passes this set to the output generation subsystems (NLG + TTS) that translate it into an intermediate text $\boldsymbol{t_t}$, which is in turn processed to produce a synthesized spoken utterance $\boldsymbol{sys_t}$. The human user answers this by an utterance $\boldsymbol{u_t}$ built according to the communicative acts s/he could extract from sys_t, to his/her knowledge $\boldsymbol{k_t}$ at time t (possibly updated thanks to sys_t) and to the goal $\boldsymbol{g_t}$ s/he tries to achieve by interacting with the SDS. Both the user's and the system utterances are mixed with noise $\boldsymbol{n_t}$. The user's utterance is consequently processed by the speech input processing subsystems (ASR + NLU) to produce an observation $\boldsymbol{o_t}$ composed of a set of concepts $\boldsymbol{c_t}$ inferred from a recognized word sequence $\boldsymbol{w_t}$ and supposed to represent what the user meant (outputs of the NLU module) and a set of metrics $\boldsymbol{m_t}$ indicating the confidence of those systems in their processing. Here it will be assumed that the metrics are of two kinds, coming from both the ASR ($\boldsymbol{CL_{ASR}}$) and the NLU ($\boldsymbol{CL_{NLU}}$) modules. The observation can be considered as the result of the processing of the DM communicative acts by its environment. The DM finally uses o_t to update its internal state $\boldsymbol{s_{t+1}}$ thanks to a model of the task.

A. Knowledge and Goal

User knowledge in the framework of SDSs can be of different natures: it can be the knowledge about the history of the interaction, about the system, about the task, about the world etc. Its main role, in combination with the goal, is to insure user behavioral consistency across the interaction. For instance, if the system has just answered a particular question, the user is probably getting closer to his/her goal because his/her knowledge has increased; the goal can even be changed through a knowledge update. On another hand, if the user has just answered a system question, his/her knowledge about the history and his/her goal will prevent him from answering differently if the question is asked again or from answering again an already asked question.

From this, similarly to the SDS, which relies on a state update, the user's behavior relies on its knowledge update. Since, the system and the user model don't share the same state space, the presented HMD model allows grounding subdialogs to be modeled if some inconsistency between the system state and the user's knowledge was detected.

B. Markov Property

A SDS is said to be Markov or to meet the Markov Property if its DM choice about the action a_t to perform at time t and its state s_{t+1} at time t+1 are only dependent of the state at time t and not of previous states and actions:

$$P(a_t, s_{t+1} | s_t, a_{t-1}, s_{t-1}, \cdots, a_0, s_0) = P(a_t, s_{t+1} | s_t). \quad (1)$$

In the following, it will be assumed that the studied SDSs are Markov. This assumption can be met by a judicious choice of the state representation, which should embed the history of the interaction into the current state. Such a state representation is said *informational*.

III. Probabilistic Description of HMD

Probabilistically and from the point of view of the DM, one can describe the system functioning by the joint probability of signals $\{s_{t+1}, a_t, o_t\}$ given signals $\{s_t, n_t\}$. By omitting the time indices and denoting $\boldsymbol{s'}$ the state at time t+1 ($s_{t+1} = s'$), one can write:

$$P(s', o, a | s, n) = \underbrace{P(s' | o, a, s, n)}_{\text{Task Model}} \cdot \underbrace{P(o | a, s, n)}_{\text{Simulation Env.}} \cdot \underbrace{P(a | s, n)}_{\text{DM}}. \quad (2)$$

Making the reasonable assumptions that the task model (which interprets the observation in order to build a new internal state for the DM) is independent from the action and the noise and that the DM does not take the noise into account when deciding which communicative acts to produce next (if it does, it is through the observations and thus, according to the state representation extracted from it), (2) can be rewritten as:

$$P(s', o, a | s, n) = \underbrace{P(s' | o, s)}_{\text{Task Model}} \cdot \underbrace{P(o | a, s, n)}_{\text{Simulation Env.}} \cdot \underbrace{P(a | s)}_{DM}. \quad (3)$$

The problem of dialog simulation is therefore to evaluate

the second term of (3), which represents the simulation of all what is surrounding the DM in a SDS (its *environment*), including the user. While the formalism was different, several proposals for dialog simulation, and thus for estimating this term, have been proposed in the literature during the last decade. One of them, described in [5], consists in the extraction of transition probabilities from a dialog corpus obtained by direct utilization of an SDS prototype by users. Before this, the authors of [4] proposed to model the environment by creating a memory-less user model. More task-dependent simulation environments have been also proposed recently in [6] and [7]. Here, it is proposed to build models of each subsystem of a SDS separately to generate realistic unseen dialogs like firstly described in [8]. To do so, the functioning of the system will be described thanks to the joint probability of all signals appearing during one interaction given the current state s and the noise n. This joint probability can be factored as follows:

$$\begin{aligned} P(s',o,a,u,sys,g,k \mid s,n) &= \underbrace{P(sys \mid a,s,n)}_{\text{Output Generation}} \cdot \underbrace{P(o \mid u,g,sys,a,s,n)}_{\text{Input Processing}} \\ &\cdot \underbrace{P(k \mid sys,a,s,n)}_{\text{Knowledge Update}} \cdot \underbrace{P(g \mid k,sys,a,s,n)}_{\text{Goal Modification}} \cdot \underbrace{P(u \mid g,k,sys,a,s,n)}_{\text{User Utterance}} \\ &\cdot \underbrace{P(s' \mid o,u,g,sys,a,s,n)}_{\text{Task Model}} \cdot \underbrace{P(a \mid s,n)}_{\text{DM}} . \end{aligned} \quad (4)$$

In addition to previous assumptions, the following can also been made:

- The output generation is independent from the noise. The text generated by the NLG module is synthesized without taking noise into account although one could think about output signal level adaptation. This doesn't mean that the noise doesn't affect the perception of the system utterance by the user, but the production process is not affected by the noise.
- The user's utterance is independent from the actual DM acts a_t and state s_t as well as from the noise. The acts are transmitted to the user by sys_t only and the history is contained in the user's knowledge.
- If there is a goal modification it is only because the user's knowledge has been updated by the last system utterance.
- The knowledge update is not conditioned by the DM's acts a_t which are only communicated by the last system's utterance.
- The input processing results are independent from the user's goal and the system utterance. Moreover, if the actual DM action can be responsible for some tuning, the spoken realization of the concepts embedded in the action is not responsible for any tuning.
- The task model is independent from the user's utterance, knowledge and goal as well as from the system utterance and noise.

With these assumptions, (4) becomes:

$$\begin{aligned} P(s',o,a,u,sys,g,k \mid s,n) &= \underbrace{P(s' \mid o,s)}_{\text{Task Model}} \cdot \underbrace{P(a \mid s,n)}_{\text{DM}} \\ &\cdot \underbrace{P(sys \mid a,s)}_{\text{Output Gen.}} \cdot \overbrace{\underbrace{P(k \mid sys,s,n)}_{\text{Knowledge Update}} \cdot \underbrace{P(g \mid k)}_{\text{Goal Mod.}} \cdot \underbrace{P(u \mid g,k,sys)}_{\text{User Utterance}}}^{\text{User Model}} \cdot \underbrace{P(o \mid u,a,s,n)}_{\text{Input Processing}} . \end{aligned} \quad (5)$$

In this last equation, some terms can still be factored as will be shown in the following subsections.

A. Output Generation

The output generation subsystems are the NLG and the TTS modules, thus the output generation term of (5) can be factored:

$$\begin{aligned} \underbrace{P(sys \mid a,s)}_{\text{Output Gen.}} &= \sum_t P(sys \mid t,a,s) \cdot P(t \mid a,s) \\ &= \sum_t \underbrace{P(sys \mid t)}_{\text{TTS}} \cdot \underbrace{P(t \mid a,s)}_{\text{NLG}} \end{aligned} . \quad (6)$$

The TTS process is not conditioned by the actual DM acts or state but only by the text generated by the NLG module. This assumption provides the last equality of (6). Generally, one text t corresponds to only one synthesized utterances sys so the summation over all possible texts in (6) is not really necessary.

B. Knowledge Update

The knowledge update is an incremental process and it can also be expressed by (7):

$$\begin{aligned} P(k \mid sys,s,n) &= \sum_{k^-} P(k \mid k^-,sys,s,n) \cdot P(k^- \mid sys,s,n) \\ &= \sum_{k^-} P(k \mid k^-,sys,n) \cdot P(k^- \mid s), \end{aligned} \quad (7)$$

where $\boldsymbol{k}^-$ stands for k_{t-1}. The underlying assumption made in (7) is that the current system state, containing the history of the dialog seen from the system side, is related to the user's knowledge at time t-1 since the user's knowledge is updated before the state during one dialog turn. The dependence of the k variable on the state s is then related to the previous possible knowledge states while the actual update is done according the previous knowledge state and the current system utterance (possibly misunderstood because of noise).

C. Input Processing

The input processing subsystems are the ASR and the NLU modules. The ASR module aims at transcribing spoken utterances like u into a sequence of written words w while the NLU module aims at extracting a sequence of concepts c supposed to represent the meaning of the spoken utterance (see Figure 1). Each module usually also provides a confidence measure about the results. In this paper, the confidence measure is considered to be defined on a concept (and its acoustical realization) and not on a word or a sentence base. This confidence level is generally a real number ranging between 0 and 1 or an integer ranging between 0 and 1000 and expressing the confidence of the system in its result [9]. The observation is then a tuple $o = \{c, CL_{ASR}, CL_{NLU}\}$ where CL_{ASR} is the confidence level of the ASR system in the speech recognition result and CL_{NLU} is the confidence of the NLU module in the understanding process. The complete speech input processing can then be described by:

$$
\begin{aligned}
P(o \mid u,a,s,n) &= P(c, CL_{ASR}, CL_{NLU} \mid u,a,s,n) \\
&= \sum_w P(c, CL_{ASR}, CL_{NLU} \mid w,u,a,s,n) \cdot P(w \mid u,a,s,n) \\
&= \sum_w P(c, CL_{NLU} \mid w, CL_{ASR}, u,a,s,n) \cdot P(w, CL_{ASR} \mid u,a,s,n) \\
&= \sum_w P(c, CL_{NLU} \mid w, CL_{ASR}, a,s) \cdot P(w, CL_{ASR} \mid u,a,s,n) \\
&\approx \max_w \underbrace{P(c, CL_{NLU} \mid w, CL_{ASR}, a,s)}_{\text{NLU}} \cdot \underbrace{P(w, CL_{ASR} \mid u,a,s,n)}_{\text{ASR}}.
\end{aligned} \tag{8}
$$

The last equality of (8) is obtained by making the assumption that the understanding process doesn't rely on the acoustics anymore but only on the ASR results, which result themselves from a maximization process.

IV. Level of Communication

Now that a probabilistic framework for dialog description has been defined, it is proposed to use this framework in the purpose of dialog simulation. To do so, we suggest to use stochastic models based on simpler variables for estimating probabilities expressed in (5), (6), (7) and (8). Yet, it is necessary to define the nature of the messages that will be transmitted between models since each of them has to simulate the processing of those messages.

In the field of dialog simulation and dialog formal description, it is often argued that *intention-based* communication is sufficient to model dialogs since what is to be investigated is the behavior of the dialog management system, which is a very high-level component working with concepts etc. Nevertheless, some of the components modeled in this study are also working at the signal level (such as the ASR module for example). This is why we opted for a message representation based on Attribute-Value (AV) pairs, which allows very high-level communication (attributes are considered as concepts) while values (particular values for the concepts) allows in a certain extend to come back to lower levels of communication. This message description is based on an Attribute-Value-Matrix (AVM) representation of the task [10].

With this representation, each communicative act can be seen as the transmission of a set of AV pairs. The set of all the possible attributes (concepts) included in the task will be denoted $\mathcal{A}$, and the set of all possible values is denoted $\mathcal{V}$. The system utterances *sys* are then modeled as sets of AV pairs in which the attribute set will be denoted $\boldsymbol{S}=\{s^\sigma\} \subset \mathcal{A}$ and the set of possible values for each attribute s^σ will be denoted $\boldsymbol{V^\sigma} = \{ v_i^\sigma \} \subset \mathcal{V}$. The system utterance contains a special attribute $\boldsymbol{A_S}$ defining the type of the embedded act. Allowed types can be constraining questions, relaxing prompts, greeting prompt, assertions, confirmations queries, etc. The user model outputs a set of AV pairs u transmitted to the ASR model in which attributes belong to $\boldsymbol{U} = \{u^\upsilon\} \subset \mathcal{A}$ and the set of possible values for u^υ is $\boldsymbol{V^\upsilon} = \{ v_i^\upsilon \} \subset \mathcal{V}$. Results of the ASR model processing is a possibly modified set of AV pairs which is in turn processed and possibly modified by the NLU model to provide a part of the observation o as a new set of AV pairs c. The user's goal and knowledge are also AV pair sets.

V. User Model

User modeling for the purpose of dialog simulation is not so common, while it is more often used for system adaptation to users preferences or goal inference [11]. Yet, in [4] for example, the authors developed a simple memoryless User Model (UM) in which the user's behavior is assumed to be independent of the history of the interaction and the user's goal. In [8], a parametric goal-directed UM with non-empty memory is presented. The models exposed in [4] and [8] are based on a set of conditional probabilities describing the behavior of the UM in particular situations like answering constraining or open-ended questions, - confirmation or relaxation prompts etc. In [8], these probabilities are conditioned by the knowledge of the UM and by its goal. The goal representation and consequently the probability set is a function of the task structure. In [4], it has also been assumed that some intuitive rules can be used to assess *a priori* values for the probabilities while they could also be learned from a corpus of data obtained by releasing the SDS to a group of testing users.

After having enumerated all those features, it seems natural to implement a UM using the Bayesian Networks (BN) framework [12]. Indeed, a BN graphically encodes conditional probabilities according to prior knowledge of the task and allows for parameters and dependencies learning starting from prior estimates. Here, the purpose is to use the BN framework for generation and not for goal inferring or knowledge representation like proposed in [13] or in [14].

A. Network Structure

The design process of a BN starts with the assessment of the topology and particularly by the enumeration of stochastic variables that will be represented by nodes in the network. Here, the UM should model the user-related terms of (5), which are:

$$
\underbrace{P(k \mid sys, s, n)}_{\text{Knowledge Update}} \cdot \underbrace{P(g \mid k)}_{\text{Goal Mod.}} \cdot \underbrace{P(u \mid g, k, sys)}_{\text{User Utterance}}. \tag{9}
$$

This defines stochastic variables that can be grouped in three sets (remember the AV-pair representation of messages):

- UM output variables:
  - The set of attributes transmitted by the UM utterance u. This set will be denoted $U = \{u^\upsilon\} \subset \mathcal{A}$.
  - The set of actual values associated to transmitted attributes. This set will be denoted $V \subset V^\upsilon$.
  - We add a Boolean variable indicating whether the UM decides to close the dialog session in order to make the UM able to stop the interaction before its natural end. It will be denoted '***close***' $\in \{true, false\}$.
- UM internal variables:
  - The set of knowledge variables $\boldsymbol{K}$. In the simplest case, we can have $K = \{k^\kappa\}$ in which each k^κ is a counter indicating how many times the κ^{th} attribute has been referred to by the dialog system.
  - The set of goal variables including AV pairs contained in the UM goal: $\boldsymbol{G} = \{[g^\gamma, gv_i^\gamma]\}$ (goal attributes g^γ and values

gv^{γ} are here in the same set)

- UM input variables (also dialog system outputs):
  - The type of system act A_S.
  - The attributes concerned by the system utterance: $S = \{s^{\beta}\} \subset \mathcal{A}$. For example, a constraining question is concerned by one particular attribute and a relaxation prompt as well.

According to conditional probabilities of (9), the BN structure is built and shown on Figure 2.

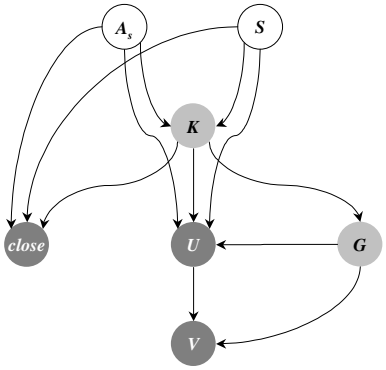


Figure 2: Bayesian-network-based user model for spoken dialog simulation. Dark gray circles stand for output variables, light gray circles stand for internal variables and empty circles stand for input variables.

On this figure we voluntarily omitted the noise but it can of course perturb the user's understanding of the system utterance (and thus the knowledge update). We also omitted the system state *s* since, as said before, the knowledge update is an incremental process and the state doesn't affect directly the knowledge at time *t* but is embedded in knowledge at time *t*-1, which is the base for update. We could actually use Dynamic BNs (DBN) to model this time dependency of the knowledge update.

B. Network parameters

Networks parameters are the conditional probabilities relating variables among them. They can either be hand-tuned by experts or learned from a corpus (or both). Of course, in order to learn knowledge update probabilities the corpus should be built with user's intervention (which is not easy) or it should be inferred from user's behavior.

C. Use of the BN-based User Model

To use this BN-based UM, the general algorithms for probabilistic inference in BNs can be used to compute the probability distribution of the variables '*close*', *U* and *V* (output variables) according to the given state of variables *G*, *S*, *K* and A_S, which is considered as the *evidence*.

VI. ASR Model

The ASR process aims at transcribing a spoken word sequence $\boldsymbol{w_i}$ uttered by a user into a written word sequence $\boldsymbol{w_h}$. At least, it is supposed to extract data usable by subsequent subsystems from an acoustic signal in which speech and noise are often mixed up. ASR errors occur when $w_h \neq w_i$. According to (8) we can write:

$$P(w, CL_{ASR} \mid u, a, s, n) = P(CL_{ASR} \mid w, u, a, s, n) \cdot P(w \mid u, a, s, n) \,, \tag{10}$$

and the recognition result w_h (the *hypothesis*) is:

$$w_h = \arg\max_w P(w \mid u, a, s, n) = \arg\max_w \frac{\overbrace{P(u \mid w, a, s, n)}^{\text{Acoustic Model}} \cdot \overbrace{P(w \mid a, s)}^{\text{LM}}}{P(u \mid a, s, n)}. \tag{11}$$

The last equality of (11) is the general equation of ASR for SDSs and one can see that the language model (LM) is dependent of the DM state-action pair. Indeed, it is a particular feature of SDSs to have a possibly context-dependent language model that allows for constraining valid speech entries according to the dialog context. Equation (11) also demonstrates that recognition confusion between two acoustically similar word sequences can only happen when these word sequences are both allowed by the language model. Recognition performances should therefore be estimated according to the used language model at time *t*.

A. Acoustic Distance

In the aim or deriving dynamically realistic estimates of ASR performances from a given language model, it is proposed to firstly compute an acoustic distance between sequence of words. To do so, it will be first assumed that one or several phonetic transcriptions can be obtained for each word sequence w_j. The set of phonetic transcriptions of w_j will be

$$\Phi(w_j) = \left\{ \varphi^{\alpha}(w_j) \right\}_{\alpha=1}^{N_j}, \tag{12}$$

where each $\varphi^{\alpha}(w_j)$ is a possible phonetic transcription itself composed of a sequence of phonemes belonging to the alphabet $\Phi = \{\varphi_i\}$ of valid phonemes for the studied language:

$$\varphi^{\alpha}(w_j) = \left\{ \varphi_k^{\alpha}(w_j) \right\}_{k=1}^{M_{\alpha}}. \tag{13}$$

The problem of finding an acoustic distance can then be considered as an alignment problem between two sequences of symbols. The most popular method for measuring the difficulty of aligning two sequences is the *edit distance* [15] which can be efficiently computed by a Dynamic Programming (DP) algorithm. In this framework, to each edit operation is associated a cost, for deletions (insertions) there exists a single mapping between the symbol and the cost c_i^d (c_i^i) of deleting (inserting) the symbol φ_i while for substitutions, a $|\Phi| \times |\Phi|$ substitution cost matrix $[c_{ij}^s]$ has to be built, where c_{ij}^s is the cost of substituting symbol φ_i to φ_j. One could think about defining the cost of each edit operation as its log-likelihood but this implies to measure probabilities on real data. Thus, it is proposed to use articulatory features of phonemes to establish these costs. If f_i (f_j) denotes the set of articulatory features of phoneme φ_i (φ_j), a distance between phonemes can be derived from a modified version of the Tverky's *ratio model*:

$$d(\varphi_i,\varphi_j)=\frac{F(f_i\backslash f_j)}{F(f_i\cap f_j)+F(f_i\backslash f_j)+F(f_j\backslash f_i)}. \quad (14)$$

In (14), $F(.)$ is a function applied on a set that assigns a weight to each element. One can notice that this distance is asymmetric. The $F(.)$ function can be defined by heuristics learned from linguistics such as:

- Confusions between vowels and consonants are unlikely.
- Confusions between consonants that only differ by the articulatory point are more likely.
- Confusions between vowels that only differ by the aperture degree are more likely and even more when augmenting the aperture (thus, the distance is asymmetric).
- A voiced consonant can become an unvoiced consonant when it is just before or after an unvoiced consonant.
- A vowel can be articulated with a different aperture degree when it is close to another vowel differing only by this feature ('close to' means in the surrounding syllables). It is even more likely when a tonic accent is on the other vowel.

Linguistics also bring some clues about insertion and deletion costs:

- Deletions or insertions of liquid consonants are more likely than for other consonants.
- Deletions or insertions of the phoneme @ (*schwa*) are more likely than other vowels.
- Insertions or deletions of consonants are more likely immediately before or after another consonant.
- Deletions are more likely at the beginning and the end of a word and should then deserve a smaller cost.

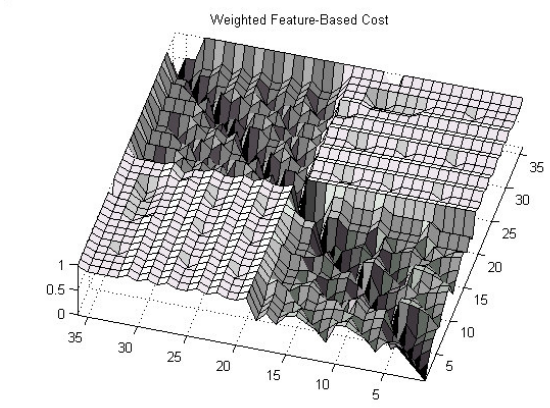


Figure 3: Cost matrix for French phonemes. Phonemes are numbered from 0 to 35 starting with the 15 vowels, while the empty phoneme has been added in 36th position to include insertion and deletions. The @ phoneme is in 12th position and has a lower cost for insertion or deletion.

According to those heuristics and to (14), a cost matrix can be defined and an example is shown on Figure 3. This cost matrix allows the computation of a distance between phoneme sequences thanks to a DP algorithm. If there are several phonetic transcriptions for two given word sequences, the distance between both will be defined as the mean distance between transcriptions.

B. Inter-Word distance vs. WER

Since we modeled the messages exchanged between modules within the dialog simulation system as AV pairs, it will be considered that an ASR error is simulated by introducing errors in the values while NLU errors are simulated by errors in the attribute-value associations. The associated language model is then considered as the lexicon of all words that can represent a value. It is then interesting to compute a distance between words of a particular vocabulary and to derive performance estimates of an ASR system on this vocabulary. An experiment has been realized on the French BDSONS database containing a vocabulary of 538 different isolated words spoken by male and female users in more than 8000 one-word utterances [17]. This experiment consisted in clustering the lexicon around each word thanks to the inter-word distance and to compare the size of each cluster with the WER (frequency of bad recognitions) obtained on the central word. Results are shown in Table 1.

TABLE 1: ASR PERFORMANCE AND WORD CLUSTERING

Word	Cluster Size	WER	Words in the cluster
Barre	23	95%	Bal, Balle, Dard, Gare, Par, Berre, Car, Jars, Tard, Parle, Dalle, Gale, Pal, Beurre, Bord, Bore, Gère, Guerre, Père, Char, Phare, Sar
Feinte	16	75%	Faite, Sainte, Fente, Teinte, Peinte, Quinte, Tinte, Pinte, Geinte, Fonte, Fête, Sente, Vente, Chante, Faute
Express	1	0%	

According to these results (and taking into account that the performance was really bad), it is possible to think that there is a good correlation between the cluster size and the WER meaning that the larger is the number of acoustically close words according to the predefined distance, the worst will be the ASR performance. And an estimate of the WER on the whole vocabulary V would be given by:

$$\mathrm{WER}(V)=\alpha\cdot\frac{\sum_{w\in V}\#(cluster(w))}{|V|}. \quad (15)$$

This allows estimating the ASR result probability of (10):

$$P(w_h\mid u,a,s)=\begin{cases}1-WER/100 & \text{for } w_h=w_i.\\ \dfrac{WER/100}{\#(cluster(w_i))-1} & \text{for } w_h\in \mathrm{cluster}(w_i)\setminus\{w_i\}.\end{cases} \quad (16)$$

C. Acoustic Perplexity

For more complex language models than lexicons, the use of the *acoustic perplexity* like defined in [18] is proposed. Unlike the lexical perplexity [19], which often poorly correlates with the WER, the acoustical perplexity takes acoustic similarities into account and is defined on a joint corpus (C,S) by:

$$APP_{\mathrm{LM}}^{C,S}=P_{\mathrm{LM}}(C\mid S)^{-1/|C|}, \quad (17)$$

where C is a text corpus containing written word sequences, S is the acoustical realisation of C, that is the spoken version of the written text contained in C and P_{LM} is the likelihood of the

current language model on the joint corpus. The likelihood of the model on the joint corpus can be decomposed as:

$$\begin{aligned} P_{LM}(C \mid S) &= P_{LM}\left(w_1, w_2, ..., w_{|C|} \middle| s\left(w_1, w_2, ..., w_{|C|}\right)\right) \\ &= \prod_{w_h \in C} P_{LM}\left(w_h \middle| w_1, ..., w_{h-1}, s\left(w_1, w_2, ..., w_{|C|}\right)\right), \\ &= \prod_{w_h \in C} P_{LM}\left(w_h \middle| \ell_h, s(\ell_h, w_h, r_h)\right) \end{aligned} \quad (18)$$

where ℓ_h is the left context of word w_h (its history), r_h is its right context (its future) and $s(w_1,\ldots, w_{|C|})$ is the spoken version of the word sequence. Using Bayes rule, each term of this product can be expressed as:

$$\begin{aligned} P_{LM}\left(w_h \middle| \ell_h, s(\ell_h, w_h, r_h)\right) &= \frac{P\left(s(\ell_h, w_h, r_h) \middle| w_h, \ell_h\right) \cdot P_{LM}\left(w_h \middle| \ell_h\right)}{P\left(s(\ell_h, w_h, r_h) \middle| \ell_h\right)} \\ &= \frac{P\left(s(\ell_h, w_h, r_h) \middle| w_h, \ell_h\right) \cdot P_{LM}\left(w_h \middle| \ell_h\right)}{\sum_{w_i \in V} P\left(s(\ell_h, w_h, r_h) \middle| w_i, \ell_h\right) \cdot P_{LM}\left(w_i \middle| \ell_h\right)} \end{aligned}, \quad (19)$$

where V is the vocabulary of all possible distinct words in the corpus. Since there isn't any joint corpus large enough to estimate the probability of words being uttered in any context, we have to assume that the spoken version of a word is independent of the previously uttered words, thereby neglecting the co-articulation phenomenon:

$$P\left(s(\ell_h, w_h, r_h) \middle| w_i, \ell_h\right) \approx P\left(s(w_h) \middle| w_i\right). \quad (20)$$

According to (12) and (13), we can write:

$$P\left(s(w_h) \middle| w_i\right) = \sum_{\alpha=1}^{N_h} \sum_{\beta=1}^{N_i} P\left(s\left(\varphi^{\alpha}(w_h)\right) \middle| \varphi^{\beta}(w_i)\right) \cdot P\left(\varphi^{\beta}(w_i) \middle| w_i\right). \quad (21)$$

If we consider that $\varphi^{\beta}(w_h) \in \Phi(w_j)$ for any $j \neq h \rightarrow P_{LM}(w_j \mid \ell_h) = 0$, i.e. that there is no homophonous word in the set of the possible words at a given grammar state (in other words, the context can always help to distinguish homophonous words), we also have:

$$\begin{aligned} P_{LM}\left(\varphi^{\beta}(w_h) \middle| \ell_h\right) &= \sum_{w_j \in V} P\left(\varphi^{\beta}(w_h) \middle| w_j\right) \cdot P_{LM}\left(w_j \middle| \ell_h\right) \\ &\approx P\left(\varphi^{\beta}(w_h) \middle| w_h\right) \cdot P_{LM}\left(w_h \middle| \ell_h\right) \end{aligned}. \quad (22)$$

Putting (17), (18), (19), (20), (21) and (22) together, we finally obtain the following approximation for the acoustic perplexity:

$$\hat{APP}_{LM}^{C,S} \approx \left(\prod_{w_h \in C} \frac{\sum_{\alpha=1}^{N_h} \sum_{\beta=1}^{N_h} P\left(s\left(\varphi^{\alpha}(w_h)\right) \middle| \varphi^{\beta}(w_h)\right) \cdot P_{LM}\left(\varphi^{\beta}(w_h) \middle| \ell_h\right)}{\sum_{w_i \in C} \sum_{\alpha=1}^{N_h} \sum_{\beta=1}^{N_i} P\left(s\left(\varphi^{\alpha}(w_h)\right) \middle| \varphi^{\beta}(w_i)\right) \cdot P_{LM}\left(\varphi^{\beta}(w_i) \middle| \ell_h\right)} \right)^{\frac{1}{|C|}}. \quad (23)$$

Here, the confusability between word sequences is embedded in $P(s(\varphi^{\alpha}(w_h))|\varphi^{\beta}(w_i))$, which can be interpreted as the probability of decoding the word w_h while the intended word w_i was uttered (because the spoken utterance $s(\varphi^{\alpha})$ sounds like the acoustic realisation of the phoneme sequence $\varphi^{\alpha}(w_h)$). It is proposed to estimate this probability using the acoustic distance described previously by stating:

$$P\left(s\left(\varphi^{\alpha}(w_h)\right) \middle| \varphi^{\beta}(w_i)\right) \propto e^{-\lambda \cdot d\left(\varphi^{\alpha}(w_h), \varphi^{\beta}(w_i)\right)} \quad (24)$$

Using this definition, the acoustic perplexity of several vocabularies belonging to the BDSONS database were computed and compared with the WER of an ASR system on these different vocabularies. Results are shown on Figure 4 where one can see that acoustic perplexity explains quite well the WER. Acoustic perplexity therefore allows predicting this WER given a language model and can be used to determine the probability $P(w_h \mid u,a,s,n)$ similarly to (16).

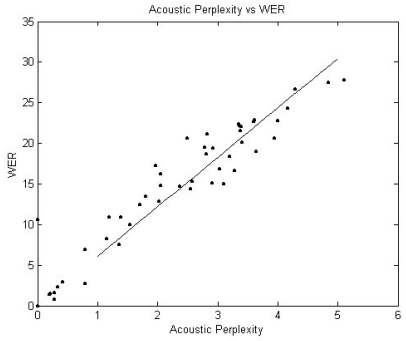
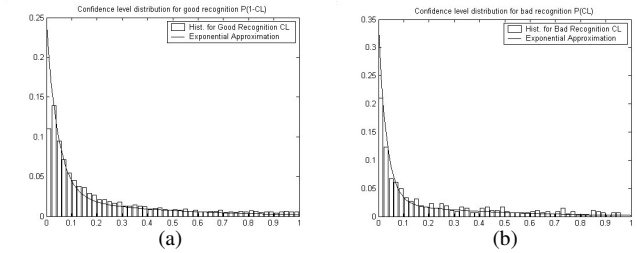


Figure 4: Acoustic perplexity vs. WER.

D. *Confidence Level Distribution*

Figure 5 shows the histogram distributions of the confidence level measured with an ASR system performing on the BDSONS database.



(a) (b)

Figure 5: Confidence level histograms for good (a) and bad (b) recognition results. Distribution of 1-CL is shown for good recognitions. The approximation of the histograms by the sum of 2 exponential distributions is also shown.

It is generally assumed that these distributions can be approximated by a sum of exponentials. In this case, we approximated them by the sum of two exponential distributions:

$$\begin{aligned} &P\left(CL_{ASR} \mid w,u,a,s\right) = \lambda_{low} \cdot e^{\lambda_{low} \cdot CL} + \lambda_{high} \cdot e^{\lambda_{high} \cdot CL} \\ &\text{with } \lambda_{low} = f\left(w,u,a,s\right) \text{ and } \lambda_{high} = f\left(w,u,a,s\right) \end{aligned}. \quad (25)$$

When the mean distance between words in the lexicon decreases, the curves flatten and both λ_{low} and λ_{high} decrease and especially for bad recognition results. A similar effect is measured at some extents for good recognition results. It might be because smaller mean distance means that words in the vocabulary induce possible problems of pronunciations (since the distance can also takes this problem into account). A pronunciation problem doesn't always result in a bad recognition result but the acoustic priors implied in the CL computation might be modified. We then have a method for simulating the generation of an ASR confidence level.

VII. NLU Model

From (8) we know that the NLU term is:

$$P(c, CL_{NLU} \mid w, CL_{ASR}, a, s) = P(CL_{NLU} \mid c, w, CL_{ASR}, a, s) \cdot P(c \mid w, CL_{ASR}, a, s) \quad (26)$$

With the AV-pair representation of messages, concept extraction consists in associating attributes to values given *w*. In section V, we have defined a BN-based user model which can be considered as a generative model. It is then possible to use again this model to find out the probabilities of each attribute to have generated one of the values embedded in *w*. Those values are therefore used as evidences for the inference engine of the UM as well as *S* and inference provides probabilities for each pair [20]. The one with the higher score comes up and its probability is kept as a confidence level (CL_{NLU}). Both probabilities of (26) are thus computed by the inference engine. If there are several values in *w*, the product of probabilities provides the CL_{NLU} value.

Nbests can also be simulated within this framework and the score of each AV pair would then be the product of both CL_{ASR} and CL_{NLU}. The AV pairs with the highest scores would be chosen to build the observation.

VIII. NLG Model

The job of the NLG module is to translate the DM intended actions *a* in a sequence of words *t* (Figure 1). Several methods are generally used, ranging from using recorded spoken utterances or handwritten prompts to automatically generated sentences.

Although most systems use recorded prompts or more generally human authored prompts, the possibility of generating more natural sentences thanks to a NLG system is a novel research area in the framework of SDSs [21]. Nevertheless, it can be considered that recorded prompts or human authored prompts are not subject to error in the transmission of the concepts while the NLG method might be error prone.

Unlike for ASR, no commonly admitted standard really exists in NLG, and it is therefore difficult to simulate thanks to a prior knowledge of the implementation. Yet, whatever the NLG system, errors in the transmission of the concepts embedded in *a* generally arise because of bad references in the generation of pronouns or anaphora. Thus they can only arise when talking about an attribute that has already been refered to in the conversation. This is why it is proposed to add an ambiguity parameter $\xi \in [0,1]$ modifying the meaning of a system utterance *sys* when a system action is modified by an already referenced attribute. If ξ is non zero, action *a* might be mismatched with another action *a´*:

$$P(t = f(a) \mid s) = 1 - \xi \text{ and } P(t = f(a') \mid s) = \xi \quad (27)$$

This mismatch is generally the result of a user misunderstanding but it is because of an ambiguity in the generated sentence, this is why ξ is included in the NLG model and not in the user model.

IX. Dialog Simulation for Optimal Strategy Learning

It this section, it is suggested to use the simulation techniques described above to train a reinforcement learning (RL) agent so as to find optimal strategies as proposed in [22]. Indeed, the simulation is suitable for using the Markov Decision Process (MDP) framework in terms of actions and states but we have to define a reward function $\boldsymbol{r_t}$ providing information about the quality of each DM decision of performing an action *a* when in state *s*. In general, the contribution of each action to the user's satisfaction is considered as the best reward function for dialog systems [5]. Several studies showed that the dialog time duration, the ASR performances and the task completion were the major contributors to user's satisfaction [10]. For this reason, we chose a reward function of the form:

$$r_t = w_{TC} \cdot TC + w_{CL} \cdot CL - w_N \cdot N, \quad (28)$$

where *TC* is a task completion measure, *CL* is a confidence measure (correlated with ASR and NLU performance) and *N* is equal to 0 if the final state is reached and to 1 otherwise (measure of the dialog length) and w_x are positive weights. This framework has been tested on two different artificial tasks in order to show the influence of each simulation module in the learned strategy. A Watkin's Q(λ) RL algorithm has been used with a softmax action selection strategy [23].

A. Database Access

The task consists in querying a database containing 350 computer configurations split into 2 tables (for notebooks and desktops) containing 6 fields each: pc_mac (pc or mac), processor_type, processor_speed, ram_size, hdd_size and brand which are attributes of the AVM representation of the task. Possible actions are 'greet' (greeting), 'constQ' (constraining question), 'openQ' (open-ended question), 'expC' (explicit confirmation), 'allC' (confirm all,), 'rel' (relaxation prompt), 'dbQ' (database query) and 'close'. Since there are as many constQ, expC and rel actions as there are attributes, there are 25 different actions. Each state is composed of 7 Boolean values indicating whether the corresponding attribute is known, a binary value indicating whether the CL_{ASR} is *high* or *low* for each attribute value, and a binary value indicating whether the last DB query resulted in a *high* or *low* amount of retrieved data. Before each simulation, a user goal is defined as a DB record and *TC* is defined as the mean number of common values between the retrieved records and the user's goal at the end of the dialog. The experiment is done with two different simulation environments, the first (Sim_1) is composed of a BN user model and a simple ASR model with fixed WER and CLs' distributions and the second (Sim_2) with the ASR model described in section VI. Results obtained by averaging values on 10,000 simulated dialogs following the learned strategy are shown in Table 2 and Table 3. Results show that in both cases, the learned strategy consists in greeting the user and then asking for some constraining questions (resulting in a better CL) or open-ended question to gather enough information to perform a meaningful DB query. Sometimes there is an explicit confirmation (because the CL is *low*) or a relaxation (because

there are no retrieved records when querying the DB). Eventually, there is a confirmation of all values and the dialog is closed. Yet, only a small set of values are needed in order to obtain a *low* number of records and the system only asks for 3 or 4 of them.

TABLE 2: RESULTS OF STRATEGY LEARNING

	N	*TC*
Sim_1	7.99	6.1
Sim_2	7.65	6.3

	greet	constQ	OpenQ	expC	AllC	rel	dbQ	close
Sim_1	1.00	1.57	1.24	0.33	1.32	0.31	1.22	1.00
Sim_2	1.00	1.55	1.22	0.20	1.33	0.24	1.22	1.00

The upper table shows mean values of *N* and *TC* and the second table shows the mean frequency of action types.

TABLE 3: PROBABILITY OF CONSTQ ACCORDING TO THE LEARNED STRATEGY

	note_desk	pc_mac	p_type	p_speed	ram	hdd	brand
Sim_1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Sim_2	0.28	0.23	0.01	0.01	0.11	0.5	0.01

This table shows the probability of constraining questions in the initial state according to the learned strategy (the 'greet' action having the highest probability).

The difference between both experiments lies in the order of constraining questions (Table 3). Indeed, with sim_1, all attributes can be asked for with the same probability, while with Sim_2 attributes providing better ASR results are asked first. This results in slightly better *N* and *TC* values because of an automatic task adaptation of the ASR model.

B. Form Filling

In this case, users are invited to provide information about a departure city (dep) and time (dep_t) as well as the destination city (dest) and time (dest_t), simulating a train ticket booking system. The desired class is also a possible option. There are 50 possible values for the cities. Possible actions are 'greet', 'constQ', 'openQ', 'expC', 'close' and it will be considered that the 'openQ' action can concern at least 2 and at most 3 attributes. Two state spaces are defined similarly as in the previous example without the DB query result variable but the CL variable can either be the CL_{ASR} (S_1) or the CL_{ASR}*CL_{NLU} (S_2). This distinction is also done in the reward function. At the beginning of each dialog, a user goal is built by filling a complete form and the *TC* variable is considered as the number of common values between retrieved values and the initial user goal. The experiment is carried out using two simulation environments. The first one (Sim_1) is composed of a BN user model and the ASR model described in section VI. In the second, the NLU model of section VII is added. Results are shown in Table 4 and Table 5.

TABLE 4: RESULTS OF STRATEGY LEARNING

	N	*TC*
Sim_1, S_1	5.39	0.81
Sim_2, S_1	7.03	0.74
Sim_2, S_2	5.82	0.79

	greet	constQ	openQ	expC	close
Sim_1, S_1	1.0	0.85	1.23	1.31	1.0
Sim_2, S_1	1.0	1.25	1.18	2.60	1.0
Sim_2, S_2	1.0	1.05	1.18	1.58	1.0

The upper table shows mean values of *N* and *TC* and the second table shows the mean frequency of action types.

TABLE 5: PROBABILITY OF PARTICULAR OPENQ ACTIONS

(dest,dep)	(t_dest,dep)	(dest,t_dep)	(dest,class)
0.005	0.11	0.11	0.12

This table shows the probability of openQ actions concerned by the couples of attributes of the first row in the initial state.

The first experiment, the baseline, shows that since the modeled user satisfaction relies as much on the duration time as on the task completion, dialogs are short, but task completion is not optimal since one attribute is often missing in the presented data (one of the cities in general). There are more open-ended questions than constraining questions. No NLU error is introduced and ASR errors are handled by the introduction of CL_{ASR} in the state space and the reward function. In the second experiment, NLU errors are introduced because the NLU model is in Sim_2 but are not handled by the learning process since no clue can be found in the state space S_1 or the reward function. This results in systematic confirmations, in an increase of the mean dialog length and in a lower *TC* value. By adding the CL_{NLU} measure in S_2 and the reward function, one can see that the performance is getting better and almost equal to the baseline in terms of task completion. This is due to the fact that some open-ended questions are avoided because they result in a *low* CL_{NLU}. It is shown in Table 5 that open-ended questions concerning both the departure and the arrival city have a lower probability to occur since they introduce possible confusions. Indeed, those two attributes share the same values.

X. CONCLUSIONS AND PERSPECTIVES

In this paper we first proposed a probabilistic description of man-machine dialogs. Based on this description, a framework for dialog simulation has been developed in the aim of evaluating dialogs and learning optimal strategies. This framework relies on independent models of modules composing a spoken dialog system and on user modeling using Bayesian Networks. Each model is based on the task structure or the definition of language models and can be trained independently. Experiments have also shown that introducing realistic ASR error modeling and confidence measure prediction results in modified learned strategies, according to the lexicons used. The learning process is thus really data driven and dependent on the task. In the same way, it has been shown that introducing a model of NLU errors and confidence measures results in highlighting problematic understanding tasks.

It is not argued that fully automated design of spoken dialogue strategies can be reached using the proposed framework. Yet, a first acceptable baseline strategy can be easily obtained. Moreover modifying the simulation setup can help pointing out problematic scenarios. Considering practical issues, we think that the presented framework could be integrated in a SDS design interface (some steps have already been taken in this direction [24]). Once again, *aided* development in the framework of an intelligent graphical interface is considered here, as opposed to *fully automated* design.

Some interesting features of the framework are still to be investigated. For instance, as said in section II.A, grounding subdialogues could be added to the set of SDS possible actions.

The need for grounding actions could be detected through inconsistency between the current SDS state and an inferred estimate of the user's knowledge about the history of the ongoing dialogue. Inference in the BN framework could also provide information about the user's goal. New state spaces and reward functions could then be built thanks to such information. On the other hand, online learning of the user model can be investigated. A Bayesian approach to user modeling is suitable for online learning. Going a step further, learning accurate user model parameters can be considered as a sub-goal of the SDS. Thus, incorporating this sub-goal in the reinforcement-learning algorithm could lead to other strategies. The system would then perform sub-optimal actions in term of short-term dialogue performance to discover user model parameters and improve subsequent dialogues. The likelihood of the model given the observed dialogues could be used as an objective function.

XI. Acknowledgement

Many thanks go to Dr. Roberto Pieraccini for his inspiring work and for his helpful remarks during the writing of this paper

XII. References

[1] R. Power, 'The Organization of Purposeful Dialogues.' *Linguistics 17*, 1979, pp. 107-152.

[2] B.-S. Lin, L.-S. Lee, 'Computer-Aided Analysis and Design for Spoken Dialogue Systems Based on Quantitative Simulations.' *IEEE Trans. Speech and Audio Processing*, vol.9, No.5, July 2001, pp. 534-548.

[3] W. Eckert, E. Levin, R. Pieraccini, 'Automatic Evaluation of Spoken Dialogue Systems.' *Tech. Report TR98.9.1*, AT&T Labs Research, 1998.

[4] E. Levin, R. Pieraccini, 'A Stochastic Model of Computer-Human Interaction for Learning Dialogue Strategies,' *Proc. Eurospeech'97*, Rhodes, Greece, 1997, pp. 1883-1886.

[5] S. Singh, M. Kearns, D. Litman, M. Walker, 'Reinforcement Learning for Spoken Dialogue Systems.' *Proc. NIPS'99*, Denver, USA, 1999.

[6] K. Scheffler, S. Young, 'Simulation of Human-Machine Dialogues.' *Technical Report CUED/F-INFENG/TR 355, Cambridge University Engineering Dept*, 1999.

[7] R. López-Cózar, A. de la Torre, J. Segura, A. Rubio 'Assesment of Dialogue Systems by Means of a New Simulation Technique.' *Speech Communication,* vol. 40, 2003, pp. 387-407.

[8] O. Pietquin, S. Renals, 'ASR System Modeling for Automatic Evaluation and Optimization of Dialogue Systems.' *Proc. ICASSP'02*, vol. 1, Orlando, May 2002, pp. 45-48.

[9] G. Williams, S. Renals, 'Confidence Measures for Hybrid HMM/ANN Speech Recognition.' *Proc. Eurospeech'97*, Rhodes, Greece, 1997, pp. 1955-1958.

[10] M. Walker, D. Litman, C. Kamm, A. Abella, 'PARADISE: A Framework for Evaluating Spoken Dialogue Agents.' *Proc. 35th Annual Meeting of the Association for Computational Linguistics*, Madrid, Spain, 1997, pp. 271-280.

[11] I. Zukerman, D. Albrecht, 'Predictive Statistical Models for User Modeling.' *User Modeling and User-Adapted Interaction,* vol. 11(1-2), 2001, pp. 5-18.

[12] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers, Inc. San Francisco, California, 1988.

[13] E. Horvitz, J. Breese, D. Heckerman, D. Hovel, K. Rommelse, 'The Lumiere Project: Bayesian User Modeling for Inferring the Goals and Needs of Software Users.' *Proc. 14th Conference on Uncertainty in Artificial Intelligence*, July 1998.

[14] H. Meng, C. Wai, R. Pierracinni, 'The Use of Belief Networks for Mixed-Initiative Dialog Modeling.' *Proc. ICSLP'00*, Beijing, China, October 2000.

[15] V. Levenshtein, 'Binary Codes Capable of Correcting Deletions, Insertions and Reversals.' *Soviet Physics Daklady,* vol. 10, 1966, pp. 707-710.

[16] A. Tversky, 'Features of Similarity.' *Psychological Review,* vol. 84, no. 4, 1977, pp. 327-352.

[17] R. Carre, R. Descout, M. Eskenazi, J. Mariani, M. Rossi, 'The French Language Database: Defining, Planning, and Recording a Large Database.' *Proc. ICASSP'84*, 1984, pp. 42.10.1-42.10.4.

[18] H. Printz, P. Olsen, 'Theory and Practice of Acoustic Confusability.' *Proc. ISCA ITRW ASR2000 Workshop*, Paris, France, 2000, pp. 77-84.

[19] F. Jelinek, R. Mercer, L. Bahl, J. Baker, 'Perplexity - A Measure of Difficulty of Speech Recognition Tasks.' *94th Meeting of the Acoustic Society of America*, Miami Beach, Florida, December 1977.

[20] O. Pietquin, *A Framework for Unsupervised Learning of Dialogue Strategies*, Presses Universitaires de Louvain, SIMILAR Collection, ISBN 2-930344-63-6, 2004.

[21] M. Walker, O. Rambow, M. Rogati, 'Training a Sentence Planner for Spoken Dialogue Using Boosting.' *Computer Speech and Language Special Issue on Spoken Language Generation* , July 2002.

[22] E. Levin, R. Pieraccini, W. Eckert, 'Learning Dialogue Strategies within the Markov Decision Process Framework.' *Proc. ASRU'97*, Santa Barbara, California, 1997.

[23] R. Sutton, A. Barto, *Reinforcement Learning: An Introduction.* Cambridge, MA: MIT Press, 1998.

[24] O. Pietquin, T.Dutoit, 'Aided Design of Finite-State Dialogue Management Systems', *Proc. ICME 2003*, vol. 3, Baltimore, July 2003, pp 545-548.

Olivier Pietquin was born in Namur (Belgium) in 1976. He obtained an Electrical Engineering degree from the Faculty of Engineering, Mons (FPMs, Belgium) in June 1999 and a PhD degree in April 2004. He joined the FPMs Signal Processing department (TCTS Lab.) in September 1999. In 2001, he has been a visiting researcher at the Speech and Hearing lab of the University of Sheffield (UK). Between 2004 and 2005, he was a Marie-Curie Fellow at the Philips Research lab in Aachen (Germany). Now he is an Assistant Professor at the Metz campus of the École Supérieure d'Électricité (Supélec, France), within the Signal Processing Systems group.

His research interests include spoken dialog systems evaluation, simulation and automatic optimisation, image processing, gesture recognition and multimedia signal processing.

Thierry Dutoit holds an Electrical Engineering degree from the Faculty of Engineering, Mons (FPMs, Belgium, 1988), and PhD degree (FPMs, 1993). He teaches Circuit Theory, Signal Processing, and Speech Processing at FPMs since 1993. Between 1996 and 1998, he spent 16 months at AT&T-Bell Labs, in Murray Hill (NJ) and Florham Park (NJ). He is the author of a reference book on speech synthesis (in English), and co-author of a book on speech processing (in French). He wrote or co-wrote about 60 papers on speech processing and software engineering.

He is an Associate Editor of the IEEE Transactions on Speech and Audio Processing and a member of the IEEE Speech Technical Committee. He is also involved in industrial activities for Acapela Group, S.A., and Multitel ASBL.