Providing users with adapted services: Dynamic building of dialogues to make heterogeneous agents cooperate

Romaric Charton, Anne Boyer, François Charpillet

Providing users with adapted services: Dynamic building of dialogues to make heterogeneous agents cooperate

Romaric Charton
LORIA - Dialoca

Anne Boyer
LORIA - Nancy 2 University
LORIA - BP 239 - 54506
Vandoeuvre-lès-Nancy - France

François Charpillet
LORIA - INRIA

{charton,boyer,charp}@loria.fr

CATEGORY

Coordination and Cooperation

KEYWORDS

Heterogeneous Agent Cooperation, Dialogue Systems, Uncertainty.

ABSTRACT

This paper deals with a prospective way to allow the communication between heterogeneous agents in the framework of multimedia services. The focus is set on the interaction between the user of a service and his/her virtual assistant. The interaction takes the form of a dialogue where the assistant tries to identify the tasks that the current user wants to perform. After a short description of dialogue systems, we propose a way to adapt them for discovering the user's goals. Our approach to control the dialogue relies on Markov Decision Processes, which are particularly suitable to handle uncertainty that occurs at many points of the communication. We moreover collect knowledge about the users in order to build profiles that will be reused for future sessions.

1. CONTEXT OF THE STUDY

The study takes place in a communication service framework whose concepts and architecture are presented in [1]. This framework aims at enhancing the supply of adaptive services over an heterogeneous agents set for the DIALOCA UNIMEDIA© platform. The range of services is relatively wide: remote meeting support, electronic commerce, message management or urban tourism assistance. The communication itself can use various media: web, telephone, short messages (SMS), e-mail, etc. This variety is progressively hidden by several abstraction layers that allow a uniform access to the available resources. Agents within the system are assumed to be social (they have knowledge of others) and rational (they have goals and act in order to reach them). They may be human beings as well as software "smart" agents and they are involved in roles like users, assistants, experts or information providers. These roles can be viewed as specific behaviors exhibited by the agents.

2. OBJECTIVES

In order to cooperate, agents must use a common language and common references (like an ontology). Communication between homogeneous agents is an active research area for many years and studies led to the definition of languages like KQML [2] or ACL [3]. Breaking the homogeneity of the agent set introduces gaps in the communication language. Furthermore, considering human beings as agents of the system raises its complexity. The communication must use a more natural language and it is heavily more difficult to have common references. These considerations led us to set the focus on the interaction between two particular roles: the user and his/her assistant, which helps the user in his/her interaction with other agents of the system. The user is supposed to have a goal to reach but he/she does not have all the required knowledge to achieve it. That is why the assistant is required to play as an intermediary because it knows how to access the relevant services and how to use them efficiently.

The nature of the interaction is cooperative: in order to provide services to the user, the assistant needs to discover what are his/her goals and intentions. Once these goals are discovered, the assistant will assume them to find relevant services it will adapt to perform the user's task. The assistant helps the user to formulate a precise request and we bring this request formulation problem to the management of a dialogue with the user.

The assistant has to take into account that the success of any action it performs may be uncertain: the action can simply fail or the result can be erroneous (like the transmission of a message during the interaction). For example, the user may misunderstand questions and give nonsensical answers or he/she changes his/her mind during the process. Moreover, available recognition systems are still not perfect and when they are used for the dialogue, the final answer is not necessarily the one that has been given. We propose to determine the context of the interaction and to take past interactions into account to overcome these problems of uncertainty and to allow an easier determination of user goals and a better adaptation of the services for the user. The knowledge gathered in the user profile will for example permit to take short cuts in dialogue sequences.

3. WASHMACHINE ASSISTANT

The following futuristic scenario tries to illustrate the concept of the adaptive user-assistant interaction.

Paul is a 6 years old boy. He decides to help his mum by washing the linen. As he does not know which program of the machine he must use, he connects on his computer and asks the domestic assistant to help him. Paul does not wish to damage the various articles and knows that the assistant is able to indicate to him how to avoid catastrophes. The assistant dialogues with him to help him to formulate his request precisely. For that, it asks Paul to describe clothing, which he wants to wash, and seeks to determine their color, their matter, etc. to select the adequate program. Obviously, Paul cannot answer certain questions, and in particular, the matter of his mum's new shirt. Paul has difficulties because he does not know the entire vocabulary required to describe this matter. Another additive difficulty happens on the voice recognition level: the radio is close to him and disturbs a little the system, which cannot say with certainty that Paul pronounced a given word. Fortunately, the assistant will use its

expert knowledge in the domain to ask precise questions, allowing to better identify the type of garment. It will also seek to use simple terms, because it knows that its user is very young and his pronunciation is far from being perfect for complicated words. Then, the assistant will ask him questions, which require simple answers like "yes" or "no". It will be able for example to ask whether clothing can crumple or not, whether it is fine or thick... The dialogue must however remain of reasonable duration to avoid loosing the attention of the child. At the end of the dialogue, the agent indicates to Paul how he should program the machine. Paul follows these advices and makes a pleasant surprise to his mum.

This scenario shows several functionalities that could be expected from an assistant: helping it's users to formulate precise requests through adaptive dialogues and sub-dialogues, respecting given constraints (like the dialogue length) and so on. It also shows that knowledge of the domain is required and that information about the user must be collected in a profile. This profile would contain various features (age, expertise level, history of past sessions...) that would allow the assistant to choose the better action to perform in function of available resources (which recognizer for example) and the environment characteristics (like noise).

4. DIALOGUE SYSTEMS

Dialogue systems manage cycles of questions and answers to identify an application goal while optimizing given criteria: dialogue length, resource usage... This goal can be, for example, the filling of a form provided by another agent, the formulation of an adequate request for an object described by features or the determination of the good destination for a phone call switching. Dialogue systems are usually speech oriented, based on speech recognition and synthesis but can be easily generalized to different media. Dialogue systems are useful in various domains and typically those addressed by DIALOCA© applications: information retrieval, electronic commerce, help systems, personal assistance... The schema [fig. 1] describes the structure of a dialogue system. It uses at least two resources: one to transmit messages to the user with a signal synthesis, another one to recognize signals and interpret answers. The next part will focus on the dialogue manager, which controls the loop for the assistant part.

5. BUILDING DIALOGUE SYSTEMS

A classical solution is to lead the dialogue with a decision tree that determines the class of the user goal/problem. Decision trees (DT) are models that learn the most useful question to ask: the one that best discriminates the set of possible goals of the training set. DTs have been used in a lot of systems and a way to build them is described in [4]. However, DTs have drawbacks that limit their use for adaptive dialogue systems: first, they require that the user knows the answer for each question he/she is asked, secondly the order of questions is determined in advance.

A more flexible method is to allow the user to give answers like the typical "I don't know". Therefore, the assistant has to know what it can do in the case of an invalid, incomplete or uncertain answer. Markov Decision Processes (MDP) are models that allow taking uncertainty into account, like the accuracy of a recognizer or the noise level. The approaches presented in ([5] and [6]) have shown that it is possible to use an MDP to build a

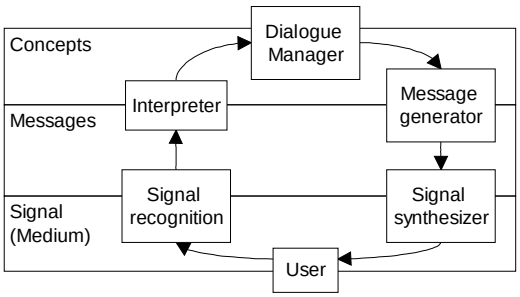

Figure 1 - Architecture of a dialog system

dialogue system. Recently, [7] have shown some limits of the MDP approach to handle the uncertainty about the dialogue state and after having studied the potential of Markov Decision Processes (POMDP), they proposed an enhancement of MDP with belief factorization. Various projects have been designed around stochastic models:

- At AT&T Research Labs Levin, Pieraccini and Eckert [6] have designed an "Air Travel Information Service" that learns dialogue strategies.

- At the Canergie Mellon Robotics Institute, Roy, Pineau and Thrun [7] have built a mobile robot called Florence Nightingale that used a dialogue to assist elderly people.

6. GOAL REPRESENTATION

We assume that the dialogue occurs in a multidimensional space where objects are described by a set of attributes $Attrib = \{a_1, ..., a_m\}$. An attribute is a tuple $<L, D, Q>$ where L is the label of the attribute, where the domain D is the finite set of possible values $\{v_1, ..., v_k\}$ of the attribute and where the set of queries Q contains the requests transmitted to the user. To access the fields of the tuple, we define the functions *label*, *domain*, and *queries*.

Let $Goal = \{g_1, \ldots, g_n\}$ be a set of possible goals for the service, tasks or items that are known by the assistant. By the introduction of the *Index* function, each goal $g_i \in Goal$, can be described with a vector of values where, $v_{i,j} \in domain(a_j)$, $\forall j \in [1;m]$ corresponds to the value taken for the attribute a_j.

$$Index(g_i) = [v_{i,1}, \ldots, v_{i,m}]$$

Another possible notation for the valuation of a goal is:

$$g_i = \{ label(a_1) = v_{i,1} \ldots label(a_m) = v_{i,m} \}$$

One can represent goals as points in the attribute referential and envision that the assistant must reach the good one through a dialogue with the user. In fact the user will have an approximate idea of where the goal is and consequently, the dialogue will aim at determining its precise position or in bad cases, it would have to show out that the goal couldn't be identified.

7. BUILDING DIALOGUES WITH STOCHASTIC MODELS

The formulation of the problem in terms of stochastic models requires the definition of the state space, actions and transitions. In order to quantify the efficiency and the quality of the assistant's

behavior and then to compare different policies, we also define rewards that are given to the assistant.

- The state of the decision process will represent the knowledge the assistant has about the goal that is partially instantiated.

- The actions of the decision process may be questions attached to a given attribute under the form of requests sent to the user. They also may be confirmation of answers, internal calculus or an action directed to another agent of the system (for example, an information request to an expert).

- A transition will occur when the system asks a question to the user and perceives his/her answer (correct or not, depending on the performances of the speech recognizer if any, on the quality of the transmission...).

- The rewards may be positive or negative values according to the final satisfaction of the user, the length of the interaction, the cost of resource use… For example, to measure the user satisfaction, various indicators and clues can be exploited. The most simple is to get an explicit marking at the end of the interaction but it can also be the result of a calculus on the final state (number of hits of a request).

After a question has been asked to the user, the current state of the process will evolve depending on the answer and the value of the attribute associated to the query will be modified. For each dimension j of the attribute referential $\{a_1, ..., a_m\}$ a couple (st_j , val_j) can be used. The state variable st_j will take a value in the set *{open, affected, closed}* and the second variable val_j will store the corresponding value if it exists (when st_j is *affected*). The following table explains the possible cases.

State	Value
Open	The question has not already been asked to the user and the dimension is not constrained. This state will be noted by a question mark "?" which means the value val_j is not yet determined.
Affected	The question has been asked and received an answer val_j.
Closed	The question has been asked to the user but did not receive any usable answer (for example, he/she said:"I don't know!"). This state will be noted by a dash "-" which means the value val_j will remain unknown.

Consequently, the current state will be expressed as a vector built on these couples (st_i , val_j). The whole space of states forms a lattice structure over the attribute referential. The system is supposed to start from a point where the whole goal is open: this state will be noted $s_0 = <?, \dots ,? > = \varnothing$ and will be at the top of the lattice structure. Solving the problem means to reach the good node of the lattice: a point where user goal is clearly recognized and confirmed.

Markov Decision Process solution is obtained as a policy π that indicates which action to perform in which state. For dialogue systems, policies can be interpreted as dialogue strategies that dictates which is the best question to ask the user to identify the goal with a minimal cost. The following table illustrates the rewards that could be given during and after the dialogue.

Accumulated Rewards	**Cost**
For each error detected / or confirmation	- to ---
For a resource access	- to ----
For each dialogue interaction	-
Final Rewards (End of dialogue)	**Cost**
Reach user the goal	+++++
Reach a failure state	-----

MDP Policies can be learnt by dynamic programming, using algorithms like Value Iteration or Policy Iteration algorithms. Another possibility is to compare and enhance policies by learning on traces, using Monte-Carlo or Temporal Differences approaches… The advantage of methods that learn on traces is that they did not require a complete knowledge of the state space. The problem is that learning a dialogue policy requires a huge amount of experiences to obtain a good policy. A possible shortcut is to build a finite state probabilistic automaton that would reproduce the behavior of real users observed during Wizard of Oz experiences… Putting the automaton and the learning dialogue system in a closed loop to help the dialogue system build a policy without requiring real users was the idea of Young when he designed his system. More details and a precise argumentation can be found in [5]. Details on MDP and machine learning can be found in [8].

Roy, Pineau and Thrun in [7], who also worked on dialogue systems identified limits to the use of MDP. They considered the user and not the system as a Markov process, saying that the user is moreover partially observable. They define dialogue systems in terms of POMDP where states are only perceived through uncertain observations. The use of POMDP led however Roy, Pineau and Thrun to intractable solutions due to the complexity of the model. They proposed to abstract the distribution over possible states, called belief state, into a couple containing for one part, the most probable estimate state and for another part, the entropy value of the distribution. This allowed them to formulate and solve the problem back in a MDP sight.

In our model, the right answer can also be known by the user and perceived by the system with a given degree of certainty. Thus, instead of having a single valuation val_j per attribute a_j, we need a probability distribution $\{p(v_1) \dots p(v_k)\}$ over the possible values $\{v_1 ,\dots, v_k\}$ of this attribute to represent the confidence in the answer. This distribution is intrinsically managed by the belief state of the POMDP. Here, the addition of knowledge about the user collected in a user model informs the assistant about the user characteristics. In the Washing machine example, the assistant would be aware that the user is a young boy and it would act adequately by tuning the recognizer for child voice and selecting a simple vocabulary. Our method tries to take advantage of past interactions for the building of the user models and will learn not only for the current user but also from those with similar profiles.

8. BACK TO THE WASHMACHINE

For a given cloth, several parameters have to be determined such as the quantity of wash powder and of fabric softener, the temperature, and the maximum rotation speed for the spin-drying. These parameters can depend on the kind of cloth, its color or its matter… To drastically restrict the problem size, let assume that clothes are described with the following attributes:

- Color: *{white, blue, red, yellow, black}*
- Dirt: *{low, medium, high}*
- Matter: *{wool, cotton}*

For a yellow cotton article heavily soiled, the full description will be expressed by the value of each attribute:

g={ Color = yellow; Matter = cotton; Dirt = high}

The dialogue would start from the state *<?, ?, ?>* and ask, for example, "what is the color of the cloth?". If Paul answers "yellow", then the dialogue state transits to *<yellow, ? , ?>*. If speech recognition is able to quantify the probability of the answer, and for example, if "black" is recognized with a probability of 0.62 and "blue" with 0.38, then the belief state would contain *< black, ?, ?>* with a probability of 0.62 and *< blue, ?, ?>* with a probability of 0.38. At this point, the knowledge about the user can be used by the dialogue system to short cut a given part of the dialogue. For example, it may not be necessary to ask if a given cloth is made of silk if the user washed only jeans and tee shirts in the past sessions.In order to validate this approach, a prototype has been designed and the experimentation for a toy application is in progress.

9. IMPROVEMENTS TO THE MODEL

The current model can be improved in many ways. For example, adding confirmation questions would permit to avoid any misunderstanding. Besides, dialogues are often organized in sub-dialogues. The approach presented in [9] shows that Hierarchical POMDP have a good potential to take sub-domains into account and to cut in the complexity. Using a hierarchical approach would allow to agglomerate parts of the dialogue but also to reuse sub-dialogues that have already been planned.

10. TOPTRADE© SERVICE

DIALOCA© has designed a service that accesses the TOPTRADE© website and provides to its subscribers a way to manage their share portfolio by telephone. This service is typically in the application field for this approach since the goal of the user can be to place an order for a given action or to ask for information. This service has the same characteristics as the wash machine problem since it requires an adaptive dialogue between the portfolio assistant and the subscriber. A corpus is currently in acquisition from real service interactions for the validation of the approach. Attributes of goals would contain the order (buy, sell, ask for value...) but also the name of the share that may be a vast domain on which the user has a partial knowledge. The use of the telephone may also increase the risk of misrecognition. The risk of asking a question will depend also on the context, because in the case of a dialogue by phone in a noisy environment, it is more judicious to ask questions that lead to simple answers (like "yes" or "no") than, for example, to ask for the name of the share (there is potentially an infinity of answers which, moreover, may be complex). Another part of the context relates to the vocal characteristics of the person: for a given speaker, words are more or less easy to recognize and it is better to avoid asking questions that risk to lead to an unrecognizable answer. The system will be able to anticipate the risks and to take the required measures: the adding of a noise filter, the use of an ad-hoc recognition engine, or the degradation of interface (like the use of keyboard entry).

11. CONCLUSION

Making agents cooperate is far from being easy, especially when some of the agents are human beings. The aim of the study is to focus on the top-level and especially on the user-mediator interaction. A prospective way to determine and achieve user goals has been proposed under the form of a dialogue system managed also by Markov Decision Processes. This way shows a good potential to efficiently manage uncertainty in dialog systems and the use for DIALOCA© services looks promising but requires still strong refinements and experimentations. In order to deal with the complexity of the dialogue, hierarchical models will also be investigated.

12. REFERENCES

[1] Charton R., Boyer A. and Charpillet F., Towards bringing heterogeneous agents to cooperation: An architecture for multimedia services. In *AAMAS 2002*, Bologna, Italy, July, 2002.

[2] Chalupsky H., Finin T., Fritzson R., McKay D., Shapiro S. and Wiederhold G.: "An overview of KQML: A knowledge query and manipulation language". *Technical Report, KQML Advisory Group*, April 1992.

[3] Agent Communication Language, *FIPA 97 Specification, Version 2.0.*, ref. 0C00003, Geneva, Switzerland, October, 1998.

[4] Quinlan J. R., Induction of decision Trees. In *Machine Learning*, Kluwer Academic Publishers, Boston, Vol. 1, pp 81-106, 1986.

[5] Young S., Probabilistic Methods in Spoken Dialog Systems. In *Royal Society*, London, September 1999.

[6] Levin E, Pieraccini R. and Eckert W. Using Markov Decision Process for Learning Dialogue Strategies. In *IEEE ICASSP'98*, Seattle, USA, 1998.

[7] Roy N., Pineau J. and Thrun S., Spoken Dialogue Management Using probabilistic Reasoning. In *ACL 2000*, Hong Kong, 2000

[8] Sutton R. S. and Barto A. G., *Reinforcement Learning: An introduction*. MIT Press, Cambridge, MA, 1998.

[9] Pineau J. and Thrun S., Hierarchical POMDP Decomposition for A Conversational Robot. http://www-2.cs.cmu.edu/~thrun/papers/pineau.hier-pomdp.html, Jan 2000.