A REFERENTIAL APPROACH TO REDUCE PERPLEXITY IN THE VOCAL COMMAND SYSTEM COMPPA

F.-A. Mathieu, B. Gaiffe and J.-M. Pierrel
CRIN-CNRS & INRIA-Lorraine
B.P. 239, 54506 Vandoeuvre les Nancy
TEL. 33 3 83 59 20 00, FAX: 33 3 83 41 30 79, E-mail: {arnmat, gaiffe, jmp}@loria.fr

ABSTRACT

The reliability of automatic speech recognition systems depends mainly on the local perplexity of the language to recognise. In the framework of vocal command dialogue systems, we propose an approach based on pragmatic, mainly through a precise treatment of referential expressions, which we use in order to reduce dynamically the local perplexity that the recognition process is confronted with. Therefore, we take into account not only the left context of the current hypothesis but also the state of the application. The article justifies the architecture we propose, describes the treatments and shows the resulting reduction of perplexity when using contextual information as compared to that obtained when using only semantic ones.

Keywords: vocal command system - natural language - pragmatics - language perplexity - reference calculus

1. INTRODUCTION

An important possible application domain for speech based systems is the command of an application, in where a vocal entry is added to a designation means (typically a mouse) and to a graphical representation of the current state of the application in order to constitute a so-called multimodal interface. Usually, the corresponding command language (related to the vocal input) is an artificial one.

An important property of such artificial languages is that they may be very predictive (if correctly designed) and therefore very appropriate for speech recognition.
However, the problem with artificial languages is that the user has to learn them and this problem increases with their size, which is directly related to the complexity of the underlying application.
In such a context, we have to face two contradictory constraints:

- providing the user with as a natural command language as possible,
- keeping as efficient and reliable as possible the speech recognition process.

When the language which is being recognized becomes more complex, we have to reduce the local perplexity even more, and so make use of pragmatic constraints.

2. DEFINING A COMMAND LANGUAGE

2.1. The size of the language

Although we aim at naturalness for the design of command languages, we suppose in this paper that the users of the system will be experts of the application which is being piloted. Moreover, these users are supposed to use the system in order to perform a task that the application supports. The definition of the command language is then restricted by the semantics of the application.
Even though the semantics strongly restricts the language (in particular the lexicon gets strongly restricted by the application domain), the local perplexity remains too important for a reliable speech recognition. In our sonar test application [1] for instance the total number of possible utterances is over a billion and the local perplexity may reach 200 possibilities.

2.2. Taking the context into account

In order to reduce the number of hypothesis, we may suppose that the user is not only an expert but that he is also attentive: he knows the context of the application. Typically, in the case when there is no "red cube" at some time in the application, he will not ask to "move the red cube" (see [2] for another approach using such constraints). The admitted language in a given context is then the set of utterances that directly yields a sequence of function calls in the application. As interpreting an utterance then consists in identifying the object(s) to modify and the modifications to perform, an incorrect utterance is an utterance for which either the concerned object(s) cannot be identified or one of the modifications cannot be performed. The main problem therefore is to identify the objects an utterance refers to.

3. CONTEXT AND SEMANTICS

We briefly mentioned semantics saying that it could be limited to the semantics of the application. We also argued for reducing even more the set of possible utterances to the contextually correct ones. An important point is that, by definition, a contextually correct hypothesis is a semantically correct one. To be more precise, an utterance for which actions can be performed necessarily mentions objects and properties compatible

with each other as well as possible actions on these objects. The only difference between a semantically correct hypothesis and a contextually correct one is that the latter enables to perform actions in the current context whereas, for the former, there could exist a context in which it would enable to perform the required actions.
According to us, this justifies to take semantics and context into account in the same modules, namely one in charge of the treatment of referring expressions and one in charge of identifying the actions.

4. SYNTAX AND CONTEXT

In the approach we propose, the syntactic module is not in charge of instanciating words from the lexicon. If it would, the reference modules would mostly (and strongly) filter those instanciations. The syntactic module therefore only provides the reference modules with uninstanciated trees which leaves are such categories as Noun, Verb, Adjective, etc. As a matter of fact, most of the semantic calculus consists in generating coherent lexical instanciations.
Almost all the command dialogue systems we know make use of so called syntactic-semantic grammars [3]. Such grammars are usually designed in a Top/Down process: the designer writes prototypical utterances forms associated to each possible command in the application.
A typical example of such a grammar could be:

Utterance ::= Move-Order | Paint-Order .....
Move-Order ::= Move-Verb + Mov-Object + Place
Mov-Object ::= the + Mov-Adjectives + Mov-Noun
....

As we will see now, such grammars may lack of language regularity and yield problems because they consider isolated utterances and not the dialogue context.

4.1. Language regularity

The first problem with syntactico-semantic grammars is that they do not guaranty any regularity of the designed language. Such a property however ease the user's learning of the accepted language. For instance, one could expect that if such utterances as: *move the red cube* and *create a cube* are authorised, then *create a red cube* is also possible. If however the function *CreateCube* in the application takes no parameter, the designer of the grammar may have forgotten this possibility [4].

4.2. Semantics and dialogue

The other problem with syntactico-semantic grammars is that they only ensure the semantic correctness of isolated utterances. Suppose for instance the following sequence:

(USER): paint the red circle in blue.
(SYSTEM): and now, what should I do ?
(USER): draw its diagonals.

This sequence is of course inappropriate in any application context (which is our definition of semantics). The last utterance however would be correct if the first one had referred to a square instead of a circle.
Another problem arises when only semantics is used to guide the speech recognition process. Consider for instance the following sequence:

(USER): move the red circle (the systems understands : *move the green square*).
(SYSTEM): there is no green square, should I move the blue one ?

According to us, such a sequence is quite troublesome for the user. That is the reason why most systems take pragmatic correctness into account in order to re-evaluate *a posteriori* the recognition scores of the semantically correct hypotheses. Our proposition consists in taking that pragmatic correctness into account earlier. We thus reduce the local perplexity intead of ruling out complete sentences.

5. LOCAL TREATMENTS

So far, we advocated:

- a general syntax,
- an integration of semantics and pragmatics. Which means modules for treating reference to objects and reference to actions.

Each of these modules yield partial hypotheses: we thus address somehow a generation problem.

5.1. Syntax

The syntactic module is not in charge of any lexical instanciation, therefore, we chose to integrate agreement treatments to those modules that know about the lexicon. A simple context free grammar is thus sufficient for representing our command utterances, and recursive transition networks provide an efficient mean to analyse such grammars. Our grammar integrates arguments coordinations and counts a hundred rules.

5.2. Reference to objects and actions

Provided by the syntactic modules with NPs or pronouns, the reference to objects module instanciates only those which actually refer to objects according to the dialogue and task contexts [5]. To illustrate this point *Nouns* are instanciated only when an object of the corresponding type is present (at least, for definite and demonstratives NPs), *Adjectives* if the corresponding property is present on an object corresponding to the type of the *Noun*, etc...
In what concerns actions, a semantic part verifies the case structure and a referential one verifies the applicability of the modifications. To get a more precise description of these two modules, see [6].

6. HYPOTHESIS STRUCTURE AND MANAGEMENT

The whole recognition process consists in successive phases of generation/completion of partial hypotheses followed by a filtering of these hypotheses by the recognition system.
Our purpose is to develop first the pragmatically correct hypotheses. This induces in fact to compute, during the recognition process itself, the pragmatic validity of the partial hypotheses (as long as it is impossible to generate all these hypotheses before starting the recognition process, given the number of possible utterances).

6.1. Structure of the utterances hypotheses

The generation phase is mainly held by the modules defined above, that is the syntactic module, the object reference module and the action reference module. The utterance hypotheses are then quadruples of sub-structures: a syntactic sub-structure, an object sub-structure. an action sub-structure; and, also, a surface sub-structure. The surface sub-structure is of course the relevant information for the recognition system, and it has to be maintained apart since the syntactic structure does not have the knowledge of the lexical entries, that are held part by the object structure and part by the action structure; but none of these two structures have knowledge about the linear order in which to present them.
Several utterances can share sub-structures. For example "select the red cube" and "erase the red cube" share their syntactic structure and their object reference one. This means that any local computation is done only once for all hypotheses that share the concerned sub-structure. In particular, this reduces the computational cost required to evaluate the accuracy of a NP in the current state of the application and in the current state of the dialogue. This is an important issue as these pragmatic calculus have to be performed on each utterance hypothesis.

6.2. Global filtering on shared hypotheses

Although our sharing of structures looks optimal, not all the semantic and pragmatic filtering can be done on the sub-structures:

- the verification that the action can actually be performed on the objects implies to deal with couple of local structures, that is, the object reference and the action references structures that are hold by at least one given utterance hypothesis;
- in the same way, the verification of the casual constraints between the verb and the type of the direct complement and the indirect complement implies to compute together object and action substructures.

The verification of these global constraints is performed by the use of "constraint functions". These functions are called when and only when all the needed information have been computed on the local structures. For example, the two utterances hypothesis "paint the little cube in green" and "select the little cube" share the same object structure (the structure associated to "the little cube"). In order to verify the pragmatic validity of these two utterances, one have to verify that the little cube is not already green, and is not already selected, respectively. This computing is in charge of a given constraint function, and this function is called only once on each of the utterance: as soon as the object associated to "the little cube" has been computed for the first utterance and, in what concerns the second utterance, after the final colour is known. This process is more precisely described in [1].

6. STRATEGY

This section describes the whole process of hypothesis generation/hypothesis filtering. In our system, there are two ways to filter the utterance hypotheses: either with the speech recognition module, either using the pragmatic constraints.

6.1. Management of the pragmatic constraints

Our system (COMPPA) works in two stages. In the first one, pragmatics is taken into account as we described. In case no pragmatically correct hypothesis is accepted at the recognition level, the system enters the second stage in which the sole semantic constraints are taken into account.
Although the pragmatic constraints are relaxed in the second stage, the "dialogue semantics" keeps being taken into account. For example, in this second step, an hypothesis such as "erase the red cube" is generated even when there is no red cube in the current state of the application. However such an hypothesis as "erase it" is not generated if the previous utterance has been "draw two red circles". We suppose in fact that even though the user may happen to be less attentive to the state of the application, he speaks a correct French (although we took dummy examples about cubes in this article, our system is dedicated to the command of a sonar console in French).
The implemented strategy (pragmatics first) is well suited for sonar operators that are actually attentive. Other strategies may be implemented without major revision of the system: for example applying the pragmatic filtering only when the recognition module is overloaded, and therefore unable to filter the hypotheses in real-time with a reasonable accuracy.

6.2. Recognition module

The recognition system we use has been developed by Thomson/Marconi/Sonar [7]. It is an analytical, speaker-independent system, handling continuous speech. It

replies two types of requests from the comprehension modules:

- "comparisons", that take co-articulation into account and therefore need to be left anchored either to the beginning of the speech signal or to another anchored comparison request.
- "localisations", that do not take co-articulation into account. These requests allow pre-filtering of word hypotheses. This pre-fitering is faster, but less accurate than the filtering by means of the comparison requests. Moreover, words that are exposed at strong co-articulation phenomenas (as the articles) can not be accuratly filtered at that level. In fact, this type of requests are provided by the recognition system in order to test quickly hypotheses of long enough words without taking into account articles or other small words that can preceed them.

6.3. Generation/filtering process

The system is constructed around six modules: the syntactic module, the object reference module, the action reference module, the module in charge of inter-structure verification, and the procedures provided by the recognition system.

Taking into account that the request to the recognition module hold at the word level and in a left to right process, the generation process consists in :

- Calling the syntactic module in order to determine the concurrent syntactic categories that may follow (immediately right) the not finished syntactic hypotheses,
- Calling either the object module or the action module to lexically instantiate the predicted syntactic categories. Only the lexical instantiations that preserve the possibility of a pragmatically correct utterance are achieved. For example, the instantiation of the category "noun" with "cube" is not proposed by the object module in the context "the red *Noun*" if there is no red cube in the current context. These modules can also ask for the calling of a constraint function, if the proposed instantiation may be incompatible with the instantiation provided by the other structure.
- Calling the inter-structure constraint functions on every couple of structure that are candidate for that.
- Calling the localisation on the lexical instantiation that correspond to long enough words.
- Calling the comparison procedure if the localisation has been validated.

This process goes on until either an hypothesis is accepted by the recognition or no hypothesis remains.

In the latter case the system enters into its semantic only mode.

7. TESTS AND CONCLUSION

Reducing the perplexity by means of pragmatics is particularly effective for proper names (in our application the reduction is in average of 400%). In fact, the pragmatic constraints at this level are often taken into account in vocal command systems that deals with proper names, but at the cost of a special treatment.

In what concerns adjectives, the reductions is about 50% in our sonar application (most objects have the same colours, etc...). The decrease is less important at the level of nouns (about 10%) in this application that doesn't manipulate a lot of different types of objects, but it could of course be different in other application.

Globally, the number of utterances (the "size" of the language) containing at least six words is less than 50% smaller when the pragmatics constraints are taken into account, even if the pragmatics constraints are always kept on the proper names.

As a conclusion, this architecture provides an effective way to reduce the perplexity of a vocal command language, and thus could authorise the use of a more natural language without decreasing the recognition accuracy.

REFERENCES

[1] F.-A. Mathieu, B. Gaiffe et J.-M. Pierrel *"A Vocal Command System using contextual Information to Constrain the Recognition Process"*, in proceedings of SPECOM'96, St. Petersburg (Russia), October 96

[2] B. Lindberg and al. *"An Integrated Dialogue Design and Continuous Speech Recognition System Environment"*, in proceedings of ICSLP 1992, p.1653-1656

[3] Burton R *"Semantic Grammar, An Engineering Technique for Constructing Natural Language Understanding System"*, BBN, Report n° 3353, Cambridge, Mass., USA.

[4] F. Duermael and B. Gaiffe *"Referring to Actions In Man-Machine Command Dialogues"*, in proceedings of EUROSPEECH-93, Berlin (Germany), September 93

[5] B. Gaiffe, L. Romary, J.M. Pierrel *"Reference in a Multimodal Dialogue: towards a Unified Processing"*, in proceedings of EUROSPEECH-91, Geneva, Italy, Sept. 91

[6] F.-A. Mathieu *"Prise en compte de contraintes pragmatiques pour guider un systeme de reconnaissance de la parole : le systeme COMPPA"*, PhD Thesis, University of Nancy 1, to appear march 1997

[7] P. Alinat and J.M. Pierrel *"ROARS : Robust Analytical Speech Recognition System"* ,in Advanced Speech Applications, K.C. Varghese, S. Pfleger J.P. Lefèvre editors, Springer Verlag, p. 197-214, 1994