PARTIAL SYNTHESIS OF SENTENCES BY COROUTINING CONSTRAINTS ON DIFFERENT LEVELS OF WELL-FORMEDNESS

GERARD MILHAUD ROBERT PASERO PAUL SABATIER

Groupe Intelligence Artificielle
CNRS UA 816
Faculté des Sciences de Luminy
163 Avenue de Luminy Case 901
13288 Marseille Cedex 9
France

ABSTRACT

We show how the two main characteristics of the ILLICO natural language interface — guided composition mode by partial synthesis, and the modularity of the encoding of linguistic knowledge specifying the lexical, syntactic, semantic and conceptual levels of well-formedness — have lead us to develop an approach and a system in which all the constraints on the different levels of well-formedness are coroutined when the system analyzes a given sentence or synthesizes a partial one. We describe the principle of the general coroutining process and the associated Prolog program.

KEYWORDS

Natural Language Interface, Analysis, Synthesis, Guided Composition, Partial Synthesis, Coroutine, Prolog.

1. INTRODUCTION

The objective of the ILLICO project, is the development of a generator of natural language interfaces for the consultation of different kinds of knowledge bases in French. The main external characteristic of the ILLICO interface lies in the fact that it can guide, if necessary, the user while he/she composes sentences. Guided composition is done according to the principle of partial synthesis of a sentence. The main internal characteristic of an ILLICO interface is the modularity of its linguistic knowledge specifying the lexical, syntactic, semantic and conceptual levels of well-formedness. In order to take the consequences of these two main characteristics into account, we have developped an approach and a system in which all the constraints on the different levels of well-formedness are coroutined when the system analyzes a given sentence or synthesizes a partial one. In this paper, we describe the external and internal characteristics of the ILLICO interface, and their consequences on sentence processing. Then we describe the principle of coroutining constraints on different levels of well-formedness and the associated Prolog program.

2. PARTIAL SYNTHESIS FOR GUIDED COMPOSITION

Using the ILLICO interface, one can compose sentences in a "free" mode or by means of various kind linguistic and conceptual information dynamically synthesized by the interface. This last situation, called the "guided composition" mode, occurs when the user directly asks the interface for help, or as soon as the interface has detected an unexpected expression typed by the user. Guided composition is done by partial synthesis of sentences, a principle discussed in [Sabatier 1989], [Rincel and Sabatier 1990]. The same system is used both for analyzing a given sentence and for providing partial synthesis of a sentence.

According to a general point of view, we assume that a sentence is correct if it is well-formed at different levels : the lexical, syntactic, conceptual and contextual levels, in

particular. A sentence is lexically well-formed if all the words and expressions it contains belong to the lexicon. It is syntactically well-formed if its structure is described by the grammar. A sentence is conceptually well-formed if the relations and the objects it describes are compatible. It is contextually well-formed if the conceptual situation it describes agrees with the discourse context. In practice, a given sentence may be analyzed in different manners. One way is to process it via successive stages. Each analysis produces a result with respect to which the next analysis takes place. The ordered stages and analyses are the following : the lexical, syntactic, conceptual and contextual analyses. A process with successive stages is not efficient if one wants the system to stop the analysis of a (complete or uncomplete) sentence as soon as a lexical, a syntactical or a conceptual error is detected. In order to halt the process as soon as possible, the process must be done in one pass and must take all the levels into account at the same time. It is the same problem in the framework of a partial synthesis process. A realistic manner to ensure that a partially synthesized sentence is well-formed is to produce it in one pass by taking all the levels of well-formedness into account simultaneously.

3. MODULAR LINGUISTIC KNOWLEDGE

One could take all the levels of well-formedness into account simultaneously by merging them into one formalism as is done for instance in "semantic grammars". We do not follow such an approach because it leads to the development of interfaces not easily portable to different application domains. When we have to port an interface to a different application domain, we must modify at least lexical and conceptual knowledge. One could develop a grammar with symbols reflecting linguistic properties, and associate to particular symbols general conditions for processing, for example, conceptual constraints, as described in [Rincel and Sabatier 1990]. Our approach is different. Within an ILLICO interface, knowledge that comes under different levels is clearly separated into distinct modules. Four modules are available :

- A lexicon (lexical level) containing expected words and expressions.

- A grammar (syntactic level) specifying expected structures of sentences and grammatical agreement. Syntactic rules are expressed in the Metamorphosis Grammar formalism.

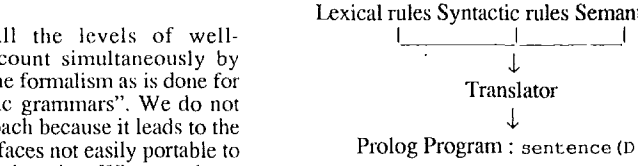
- A set of compositional semantic rules expressed by lambda-expressions for producing semantic representations from the syntactic rules of the grammar.

- A conceptual model (conceptual level) specifying, in terms of domains and relations, the lexical presuppositions associated with expected word and expressions [Milhaud 1990], [Godbert 1991].

Formalisms in which this linguistic knowledge is expressed are independant of any programming language.

4. COROUTINING CONSTRAINTS : THE CORE PROLOG PROGRAM

The reversibility of certain Prolog programs is well known. So, in order to facilitate the implementation of a system running both in analysis and in synthesis, the run time of the ILLICO interface is a Prolog program. Lexical rules (the lexicon), syntactic rules (the grammar) and compositional semantic rules are translated into a Prolog program according to the following schema :



The top call of the Prolog program produced by the translator is `sentence(D,R)`. From the axiom `sentence` of the grammar, it generates all the couples `(D,R)` where `D` is a derivation tree and `R` its associated semantic representation. Derivation trees are generated according to a top-down, depth-first, left-to-right and non-deterministic strategy. According to this method, a sentence `S` is lexically and syntactically well-formed if `S` is the list of leaves of a derivation tree `D` generated by `sentence(D,R)`. We have the following Prolog program :

```
well_formed(S) :-
  list_of_leaves(D,S),
  sentence(D,R).
```

This program is used both for analyzing and synthesizing sentences. In the analysis mode

(i.e. when S is bound), for reasons of efficiency, as soon as a leaf of a derivation tree is generated, one must verify that this leaf is identical to the current word (or expression) to be analyzed in the sentence. If they are identical, the generation of the current derivation tree goes on ; else backtracking automatically occurs in the process of the derivation tree generation. In order to do that, the call to the predicate `list_of_leaves` is placed before the call to `sentence`. As defined in [Giannesini et al. 1985], it is coroutined by using the built-in predicate `freeze`.

```
list_of_leaves(D,S) :-
 freeze(D,leaf(D,[],S)).

leaf(Leaf,L,[Word|L]) :-
  atom(Leaf),
  Leaf = Word.

leaf(Root(Tree),L1,L2):-
 freeze(Tree,leaf(Tree,L1,L2)).

leaf(Root(Tree1,Tree2),L1,L2) :-
  freeze(Tree1,leaf(Tree1,L3,L2)),
  freeze(Tree2,leaf(Tree2,L1,L3)).

leaf(Root(Tree1,Tree2,Tree3),L1,l2)
... etc.
```

The first clause `leaf` tests if unification is possible between the current leaf of the derivation tree (`Leaf`) and the current word of the sentence (`Word`). If unification succeeds, the process goes on, else backtracking automatically occurs in the generation of the derivation tree. This process runs both for analysis (`Word` is bound) and for synthesis (`Word` is free). In order to do partial synthesis in one pass, one must record the generated set of leaves as possible current words for the sentence. Current leaves must be recorded as soon as a current word is unexpected (analysis) or absent (synthesis) in the sentence. In order to do that, we must modify the first clause of `leaf`. We now describe the algorithm.

The process associates with each word an integer corresponding to its position in the sentence. The algorithm needs two counters `Rightmost` and `Current`. The value of `Rightmost` is the integer associated to the rightmost expected word in the sentence. `Rightmost` increases according to the words accepted, but never decreases : backtracking in the generation of the derivation tree has no effect on it. The value of `Current` is the integer associated to the current word of the sentence. It increases and decreases according to the evolution of the derivation tree generation. The first clause `leaf` is now as follows :

```
leaf(Leaf,L,[Word|L]) :-
  atom(Leaf),
  test(Leaf,Word).
```

The `test` procedure is based on the following algorithm :

```
• Current < Rightmost
    if Leaf = Word
    then continue
    else  backtracking


• Current = Rightmost

    if Word is free
    then record Leaf
         backtracking

    if Word is bound
    then if Leaf = Word
         then
            erase recorded leaves
            Rightmost:=Rightmost+1
            continue
    else record Leaf
         backtracking
```

In order to verify (in analysis mode) or to specify (in synthesis mode) that a sentence `S` is conceptually well-formed, one coroutines an initial constraint `conceptually_well_formed` on the semantic representation `R` associated to `S`. This condition is a call to the rules specifying the conceptual model related to the application domain of the system. The correctness of a semantic representation `R` (associated with a sentence `S`) is verified during its composition.

Finally, The core Prolog program ensuring that a sentence is well-formed is the following :

```
well_formed(S) :-
  conceptually_well_formed(R),
  list_of_leaves(D,S),
  sentence(D,R).
```

Here, the two last calls `list_of_leaves` and `sentence` express the constraint lexically and syntactically well-formed defined above.

5. CONCLUSION

Mastering the control of partial synthesis in order to avoid dead ends is an interesting challenge in natural language processing. The approach consisting in coroutining constraints on different levels of well-formedness (lexical, syntactical and conceptual ones) is a technical

solution, as the first stage of our ILLICO project illustrates. The next stage will consist in integrating constraints on contextual well-formedness.

6. ACKNOWLEDGMENTS

The ILLICO project is funded by the Commission of the European Communities (TIDE Project : Nr. 158 : KOMBE), the French Ministère de le Recherche et de la Technologie and the French Conseil Régional Provence-Alpes-Côte d'Azur.

7. REFERENCES

Giannesini F., Kanoui H., Pasero R., Van Caneghem M., *Prolog*, InterEditions 1985, Addison-Wesley 1986.

Godbert E., *Les contraintes de domaines dans un modèle conceptuel associé à une interface en langage naturel*, Rapport Interne, Groupe Intelligence Artificielle, 1991.

Milhaud G., *Définition et intégration de contraintes conceptuelles dans un système d'analyse de phrases*, Mémoire de Diplome d'Etudes Approfondies, Faculté des Sciences de Luminy, Groupe Intelligence Artificielle 1990.

Rincel Ph., Sabatier P., *Using the Same System for Analyzing and Synthesizing Sentences*, Proceedings of the 13th International Conference on Computational Linguistics, COLING-90, Helsinki, Finland, August 1990, pp. 440-442.

Sabatier P., *Interfaces en langage naturel : du traitement du non-attendu à la composition de phrases assistée*, Annales des Télécommunications, CNET, 1989.