GENERALIZED MEMORY MANIPULATING ACTIONS FOR PARSING NATURAL LANGUAGE

Irina Prodanof

Istituto di Linguistica Computazionale
CNR-Pisa

Giacomo Ferrari

Department of Linguistics
University of Pisa

Abstract

Current (computational) linguistic theories have developed specific formalisms for representing linguistic phenomena such as unbounded dependencies, relatives, etc. In this contribution we present a model of linguistic structures storing and accessing, which accounts for the same phenomena in a procedural way. Such a model has been implemented in the frame of an ATN parser.

1. Introduction

Literature on parsing natural language has recently been concerned with topics which are unfamiliar to traditional Computational Linguistics such as functional similarities of different syntactic constructions (1), reanalysis of 'indelible' structures (9), manipulation of 'indelible' trees (7).

Most of the linguistic problems involved are connected with unbounded dependencies and antecedent-referent binding. Grammatical formalisms have been developed, which represent certain linguistic phenomena in terms of discontinuous constituents (10), or generate unboundedly dependent symbols on the basis of specific conventions (4,5) However, the corresponding parsing algorithms are simple modifications of traditional parsers, where the extentions of formalisms are not adequately accounted for on the procedural side

Our approach to the problem of parsing Natural Language is to identify a set of processing strategies, which may not only process, but also represent the alluded phenomena within the frame of a psychologically motivated CF parsing algorithm.

2. Generalized memory manipulating actions

EXAM-LC (**EX**tendend **A**ccess and **M**anipulation of the **L**eft **C**ontext) is a model of linguistic structures storing and accessing designed according to such a psychological procedural approach. It relies upon the following assumptions:

- the left context, i.e. the structure corresponding to the currently analyzed part of the input string, is often affected in some way by newly incoming information and may contain information which affect further analysis. Therefore, it must be accessible beyond possible limits imposed by the structure of the parser. In fact, in many cases the data within the scope of the current constituent or rule are more freely accessible than the others. On the contrary, the scope of accessibility to the left context should be specified according to how far it is affected or affects the current analysis;
- a set of general actions can be defined corresponding to mental operations actually accomplished during the process of comprehension. Any surface construction is described both by its (deep) representation and by the operations which perform the mapping. These actions can be assumed as linguistic procedural universals;
- it is possible that syntactic phenomena, that have different structural explications, are handled by a common process or sequence of operations

A common space of memory is assumed to contain the current hypothesis about the analysis of the parsed segment of the input from the beginning We will refer to such a structured space as Current Global Hypothesis (CGH) The following set of abstract operations on the space of memory has been defined till now

a) an *opening* and a *closing* action will respectively start and end the storing of the information related to a phrase/clause in a current subspace subsequently merged with the global space The way of storing depends upon the representation of the output and the corresponding actions are desgined in accordance to it

b) a *retrieving* action involving two participants, a symbol that triggers the action (*trigger*) and the information to be retrieved (the target of the action) will retrieve entire constituents which appear to be possible antecedents fragments of structure, or even simple lexical features The trigger may be a gap, a pronoun, an ellipsis and any other phenomenon which requires the search for an antecedent (*target*), or the need for an agreement

After the identification of such a *trigger* the action is decomposed in three steps:

i) extraction of constraints which must guide the search for the target.

ii) scanning of the CGH under the specified conditions, and

iii) retrieving of the required information On this functional ground, this description fits to more or less all the cases mentioned in (1)

The action of searching back may be constrained by several types of restrictions, including

i) morphological features i.e the gender and number of a pronoun or those required for agreement by the syntactic environment (e.g the verb),

ii) syntactic idiosyncrasies of a lexical item, this is the case of STRANS verbs (i.e those verbs requiring a complement clause) that determine which of their arguments is to be the subject of the complement, as shown in

(1) *a* A credette di udire un rumore
A thought to hear a noise
b A persuase B a fuggire
A persuaded B to run away
c A ordino' a B di sedere
A imposed to B to sit down

iii) semantic features or cognitive descriptions that may be introduced in the process, and

iv) syntactic determination of the scope of the search, such as, for instance, *Subjacency*

Retrieving of an antecedent may actually correspond to two different operations depending upon whether the antecedent to be bound linearly preceeds or follows the

symbol it is to be bound to. In fact, in many common sentences the antecedent linearly follows its dependent, as in

(2) *a* Quando ___ si arrabbia, Giovanni diventa rosso
When (he) gets angry, John becomes red
b Se lo vedi, saluta Giovanni da parte mia
If (you) see him, say hallo to John on my behalf

In this case, the binding should take place in two steps, the flagging of the need for a forward binding and the moving of the pointer from the antecedent, once detected, to the flag.

c) Many of the retrieving actions are to end up with a binding or, more generally, with a *moving* action. This can be realized at least in two ways, by actually copying the retrieved constituent into the trigger's subspace or simply moving a pointer from one to the other. Also in this case the choice is a matter of representation.

d) A *riconfiguration* action is necessary in order to modify an already (partially) built structure, as new incoming information indicate the need for such a change. The types of modifications to be performed again depend upon the representation of the output, but the one required by the relative pronoun is likely to be a general one.

Relative pronouns need to be bound to an antecedent and, besides, are the surface signal of an embedding. No special processing difficulty is proposed by the sentence

(3) il ragazzo che corre
the boy who runs

where the relative pronoun occurs exactly where the embedding begins. In this case a scope restriction can limit the search for an antecedent to the immediately preceeding NP. But in the case of *pied-piping* as in

(4) il cane della fedelta' del quale nessuno dubita
the dog about the fidelity of which nobody doubts

the relative clause boundery is set three (in english four) words before the relative pronoun. In the framework we have been discussing an action which structurally modifies the left context can be proposed, which should embed the component(s) being processed in a relative clause as the relative pronoun is met.

e) A final type of access to the left context is the *relabelling* of a processed component, already used for the passive transformation.

3. An experimental implementation

An experimental realization of the above discussed ideas has been implemented in the frame of the ATN parsing algorithm (13) running a functional grammar *a'la* M. Kay (6).

ATN has been preferred because it meets the requirements of being a psychologically motivated CF parser. Moreover ATN can be considered a very well stabilized parsing algorithm. The data structure is a list which is mainly accessed with a typical LIFO stack policy. It represents a unique memory space non splitted into registers. It contains at any point of the process the CGH, i.e. the entire left context literally represented in terms of attribute-value pairs, as required by functional grammar.

We give hereafter a list of the functions which access the CGH in Backus notation.

1. *Actions*
   a. ‹storing actions›::=
      ADD pair location |
      ASSIGN label path
      ‹location›::= NIL|‹form›
      ‹label›::= any label
      ‹pair›::= label value
      ‹value›::= *|‹form›
   b. ‹list manipulation›::=
      PUSH |
      POP |
      INSERT data item
      ‹data›::= any data
      ‹item›::= ‹form›

2. *Forms*
   FIND path test level dtype |
   FINDVAL path test level dtype |
   LOCATE path test level dtype
   ‹path›::= ‹label+›
   ‹test›::= T|any test
   ‹level›::= T|CL
   ‹dtype›::= T|ND|L

The basic storing action is ADD which is used to store any incoming piece of structure.

Extraction of information is done by the forms FIND, which returns a pair, and FINDVAL, which returns only the value of a pair. LOCATE works exactly in the same way, but returns a pointer to a given radix. All the three functions can work in different modes. They can search either only the current level (CL) or through the entire list (T). In this latter case the current level is excluded and, if no further options are specified, the lower (the nearest to the top) occurrence is returned. Another option (dtype) returns all the occurrences either appended in a list (L) or one by one, non-deterministically (ND). A third option evaluates conditions in order to select the component identified by the specified path.

The three last actions, PUSH, POP, and INSERT, manipulate the items in the list. PUSH adds a new (empty) item in front of the list. The elements of the component being analysed (phrases or sentences) are ADDed in this top item, which has been therefore referred to as current level. POP removes the current top-item and embeds it into the new top-item, possibly assigning a label to the corresponding component. Finally INSERT inserts an item, corresponding to a new level, somewhere back between 'item' and the front part of the list, and fills it with 'data'. It is the only instance of reconfiguration action designed till now while others can be introduced according to the studied phenomena and the adopted representation.

List manipulation takes place independently from the starting or the ending of the process expressed in a subnet. Thus a component can be POPed after the end of its recognition procedure, when also its function is clarified. This list of actions is open-ended and is supposed to be updated as new general operations are identified.

The retrieving action scans the entire left context according to specified searching constraints and is able to return any required fragment, regardless of its structure. In particular, information associated to the lexical items already inserted in a structure can be extracted for further processing at any point, without having previously been saved in a register or raised to higher nodes, as often lexical features are.

This turns out to be particularly usefull for the treatment of lexical idiosyncrasies. We have already mentioned that an STRANS verb determines which of its arguments is to be the subject of the complement clause. Many other words have special linguistic behaviours. The possibility of treating such syntactic peculiarities of lexical items by associating particular fragments of grammar to lexical entries and processing them when needed has already been experimented (3). A wider and easier use of such a device contributes to our parser certain advantages of word expert parsing still sticking to a syntactic model of analysis.

The details of this implementation and some examples of processing have been discussed in (11).

4. An intuitive psychological argument

It has been shown (7, 11) that a reconfiguration action, which modifies the structure of the already analysed part of the input string limits backtrack, since a certain class of parsing errors can be corrected without actually restarting the processing of the input string. Under a psychological viewpoint the hypothesis is that during the comprehension of a sentence guesses (CGH's) are progressively enriched and stored in a space of memory. In this process errors may occur. For some of them, such as relative clauses (11) and adjective attachment (7), it is enough to modify the previous guess while for others a real backtrack and reanalysis is necessary. A possible explanation is that in the activity of sentence comprehension a phase of structuring is distinguished from a phase of perception. Errors occuring in the former are remedied by modifying a guess, while those occuring in the latter need backtrack and the choice of another strategy.

5. Parsing efficiency

A relevant claim is that the data structure and the set of actions of EXAM-LC-ATN improve the computational efficiency of an ATN parser. This depends upon two peculiarities of the parser, the reduction in register setting and storage accesses and the limitation of backtrack. In ATN, register setting is done by searching through a register table; if the called register is already present it is set to the new declared value, while if it is not, it is initialized and filled. In EXAM-LC-ATN ordinary register setting (SETR) is substituted by an operation which stores information in an already initialized memory subspace. Thus, for an equivalent grammar, the number of storage accesses in EXAM-LC-ATN is exactly the same of a traditional ATN, but no searching or initialization is performed. There is a similarity between a register initialization operation and the opening of a new memory subspace (PUSH). However, this is done only for the higher level constituents, such as S, NP, PP, etc., while in an ATN new registers are activated even for the storage of terminals or features.

Extra register setting, commonly used in an ATN to laterally pass parameters to different levels of computation from the current one (SENDR, LIFTR), is unnecessary in EXAM-LC-ATN. In fact, the required parameters are retrieved if and when they are needed from the level where they are needed. This guarantees efficiency in two ways. On one hand, the SENDR/LIFTR - GETR pairs are substituted by a single search through the left context. Within a memory subspace the search proceeds linearly as for pattern-matching. On the other hand, the overhead due to the sending or lifting of parameters which may turn out to be useless at the next upper or lower level is eliminated. In conclusion, the average of storage spaces activation operations and storage accesses is in favour of EXAM-LC-ATN.

Backtrack is the typical mechanism by which non-deterministic depth first parsers like ATN manage the choice of alternatives. It consists in going back to the nearest choice point and trying the following alternative. It involves also a restoring of the input pointer to the element in the input string where it was before the current alternative was chosen, and reprocessing of the input from that point in the new alternative. Generally this mechanism is activated when some incoming piece of information signals that the chosen alternative is the wrong one. In some cases, the information give meaningfull indications about the correct alternative analysis to be chosen. In these cases backtrack can be substituted by special techniques for modifying the output of the analysis without reprocessing the involved segment of the input string. INSERT is a case of such a modification action and avoids backtrack. The full access to the left context allows for the introduction of other ways of limiting backtrack.

REFERENCES

1. **Berwick R.C., Weinberg A.S.**. Syntactic Constraints and Efficient Parsability. in Proceedings of the 21th Annual Meeting of ACL, Cambridge MA. 15-17 June 1983. pp.119-122.
2. **Berwick R.C.**, A Deterministic Parser with Broad Coverage. in Proceedings of the Eighth International Joint Conference on Artificial Intelligence . Karlsruhe 8-12 August 1983. pp.710-712.
3. **Cappelli A., Ferrari G., Moretti L., Prodanof I., Stock O.**, Parsing an Italian Text with an ATN Parser. NT-ILC-CNR, Pisa. 1978.
4. **Gazdar G., Klein E., Pullum G., Sag I.**. Generalized Phrase Structure Grammar. Blackwell. 1985.
5. **Kaplan R., Bresnan J.**. Lexical Functional Grammar. A Formal System for Grammatical Representation. in Bresnan J. (ed.). The Mental representation of Grammatical Relations, MIT Press 1982. pp.173-281.
6. **Kay M.**, Functional Grammar. in Proceedings of the 5th Meeting of the Berkeley Linguistic Society. Berkeley 1979. pp.142-158.
7. **Lesmo L., Torasso P.**, A Flexible Natural Language Parser Based on a Two-Level Representation of Syntax. in Proceedings of the First Conference of the European Chapter of the ACL, 1983. pp.114-121.
8. **Marcus M.**. A Theory of Syntactic Recognition for Natural Language. MIT Press. Cambridge MA. 1980.
9. **Marcus M., Hindle D., Fleck M.**. D-Theory Talking about Talking about Trees. in Proceedings of the 21th Annual Meeting of ACL. Cambridge MA. 15-17 June 1983. pp.129-138.
10. **Pereira F.**, Extraposition Grammars. in AJCL. 7. 4. 1981. pp.243-256.
11. **Prodanof I., Ferrari G.**, Extended Access to the Left Context in an ATN Parser. in Proceedings of the First Conference of the European Chapter of the ACL. 1983. pp.58-65.
12. **Shipman W.D., Marcus M.**, Towards Minimal Data Structure for Deterministic Parsing, in Proceedings of the 6th International Joint Conference on Artificial Intelligence, Tokyo, August 20-23 1979. pp.815-817.
13. **Stock O.**, ATNSYS: Un Sistema per l'Analisi Grammaticale Automatica delle Lingue Naturali. NI B76-29. IEI. CNR.