

PREDICTIVE CONTROL PARSER: EXTENDED LINGOL

Hozumi TANAKA, Taisuke SATO and Fumio MOTOYOSHI
Electrotechnical Laboratory
2-6-1 Nagata-cho, Chiyoda-ku,
100 Tokyo, Japan

We have developed a flexible parser called Extended LINGOL which is an extended version of Pratt's LINGOL. Although original LINGOL has several advantages, the results of our experiments revealed that it has some deficiencies. One of them is that each augmented context-free rule does not enable us to examine their applicability in case of some situations. Another is that the morpheme analyzer of original LINGOL is too simple to analyze agglutinative languages such as Japanese. With respect to the former problem, we have introduced predictive control mechanism which has offered us flexible methods to implement a precise grammar with a small set of rules. Furthermore, as the mechanism avoids generating unnecessary partially parsed trees we have obtained high speed parsing. On the other hand, with respect to the latter problem, we have implemented a more powerful morpheme analyzer.

1. INTRODUCTION

The LINGOL, which was developed by V. Pratt, is a research tool for natural language processing. We implemented LINGOL on our T-5600 computer system about four years ago and used it for a while. The advantages of the LINGOL are its program size, speed, and its neat format of grammar and dictionary. It is interesting to note that the semantic interpretation method of the LINGOL is similar to that of Montague Grammar.

The result of our experiments, however, has revealed that Pratt's LINGOL has several deficiencies mainly with respect to syntactic processing. One of them is that each augmented context-free rule does not enable us to examine their applicability. Another is that the morpheme analyzer of original LINGOL is too simple to analyze agglutinative languages such as Japanese. In order to cope with these deficiencies, we have revised some portions of original LINGOL and added new programs to it.

2. PREDICTIVE CONTROL MECHANISM

2.1 <Advice> and <delayed-advice>

The Extended LINGOL incorporates a predictive control mechanism which enables us to examine the applicability of each rule in case of some situations. The string parser developed at New York University had implemented the same mechanism in their top-down parser [1].

However, as the LINGOL uses both top-down and bottom-up parsing algorithms, the predictive control mechanism differs from that of New York University. Descriptive format of our dictionary is,

```
[<word> <syntactic-category>  
  (<message-list> <cog>) <sem>]
```

Message-list is considered to be attached to the syntactic category of the <word> and is used later by predictive control mechanism as messages. (Functions of <cog> and <sem> are the same as described in [3] and [4]). The format of an augmented context-free rule in Extended LINGOL is,

```
[<left> <right> (<advice> <cog>) <sem>]
```

The <left>-<right> pair represents a context-free rule of $A \rightarrow B(C)$. The <advice>, which is introduced in our Extended LINGOL, is an arbitrary LISP function for controlling parsing process. The evaluator of predictive control mechanism works as follows:
For each <advice> or <delayed-advice> in rules,
if the evaluation value is true
then do not apply the rule
else apply it;

Suppose a Partially Parsed Tree with a root node E (PPT:E) is created at some intermediate stage of parsing. There are six possibilities to extend the tree. Four of them depend on the evaluation of <advice> in the following rules;

```
(R1)  $A \rightarrow E$  or  
(R2)  $A \rightarrow E F$ .
```

(See Fig.1).

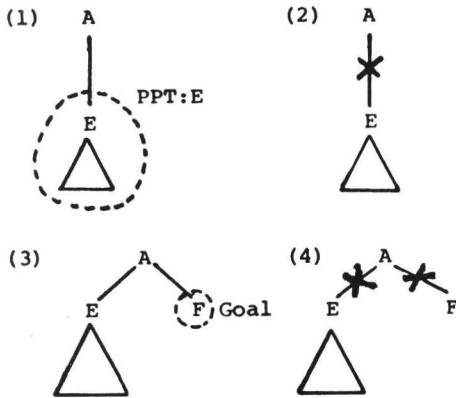


Fig.1 The result of the evaluation of advice

"X" in Fig.1 indicates that (R1) and (R2) fail to apply. Fig.1(1) indicates that (R1) applies to PPT:E. Fig.1(3) indicates that (R2) creates a new goal which predicts the next syntactic category F. In (4), a new goal is explicitly cancelled, while in (2) it is implicitly cancelled.

When a new goal is created as in Fig.1(3), we can record <delayed-advice> in the goal by using the Memo-Advice (MA) function in <advice> in the following way:

(MA <delayed-advice>)

The execution of <delayed-advice> is postponed until PPT:F will be created. (See Fig.2). With respect to <delayed-advice>, there are two more possibilities in growing PPTs depending on the evaluation value of <delayed-advice>. The <delayed-advice> acts as though it were a look-forward demon which would be activated when a goal is satisfied later.

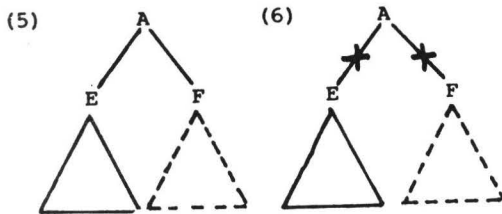


Fig.2 The result of the evaluation of <delayed-advice>

2.2 Control of Messages

In the field of advice and delayed-advice, we use messages to control parsing process. About 10 primitive functions are prepared.

A. Creating and Passing Messages: In <advice>, users can create and pass any kind of messages from node to node upward in accordance with the growth of PPT. There are six primitive functions as shown in Fig.3. The effects of the evaluation are also stated in Fig.3.

CF rules	Primitive Functions for creating and passing messages	Message "M" sent to Node A.
$A \rightarrow B$	(PM 'a)	(CONS 'a X)
$A \leftarrow M$ $B \leftarrow X$	(PML 'a)	(NCONS 'a)
$A \rightarrow B \ C$	(PML 'a)	(CONS 'a X)
$A \leftarrow M$ $X \rightarrow B$ $Y \rightarrow C$	(PMR 'a)	(CONS 'a Y)
	(PMRL 'a)	(CONS 'a (APPEND X Y))
	(PM2 'a)	(NCONS 'a)

Fig.3 Primitive functions for creating and passing messages.

B. Checking Messages: When applying a context-free rule to some PPTs and goals created so far, users can ask for messages at the root node of PPTs by using the function named QM. Recall the cases of Fig.1 where (R1) and (R2) have a very simple <advice> (QM 'm).

[A E ((QM 'm)...)] ; A $\rightarrow$ E
[A (E F) ((QM 'm)...)] ; A $\rightarrow$ E F

If a message m exist at the node E, (EVAL (QM 'm)) becomes true and two rules under consideration can not be applied ((2) and (4)), otherwise PPT:E grows in (1) or a new goal is created in (3).

C. Removing Messages: To remove a messages in the root node of PPTs, the function named RM is used.

3. EXAMPLES

In order to show how the predictive control mechanism works, we discuss two very simple examples, which inhibits the recursive application of rules of left and right recursive definitions.

Example (a)

Let us consider the following rule.

(R3) [NOUN (NLUN SUFFIX)
((OR (QM 'uncountable)
(QM 'suffix) (PMR 'suffix))...)]

and [estimation NOUN ((uncountable)...)]
[apple NOUN(...)]
[s SUFFIX ((suffix)...)]

Notice that the rule (R3) is left recursive, but it should not be applied recursively. The predictive control mechanism avoids the situation. The last three are from a dictionary.

The <advice> in (R3) works as follows:

If NOUN is not an uncountable noun
and Noun is not followed by a suffix
then create a goal by applying (R3)
and pass a message named suffix
upward if a goal SUFFIX is satisfied;
else do not apply (R3); (See Fig.4(a)).

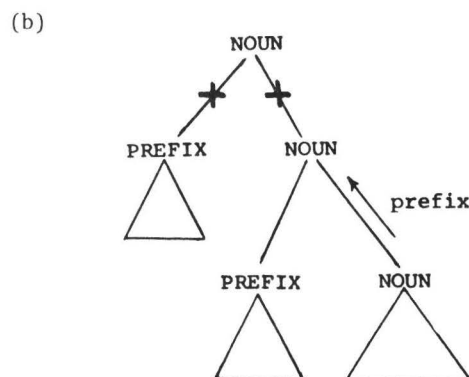
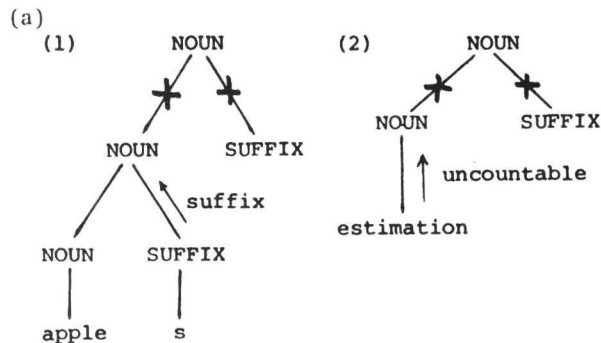


Fig.4. Example (a) and (b)

Example (b)

The following rule (R4) is right recursive.

(R4) [NOUN (PREFIX NOUN)
((MA (OR (QM 'prefix)
(PM 'prefix)))...)]

The <advice> creates a <delayed-advice> (OR (QM 'prefix) (PM 'prefix)). By using Fig.4(b), it is easy to follow how the <delayed-advice> prevents the right recursive application of (R4).

4. CONCLUSION

We have discussed a new parser called Extended LINGOL which has been used in our laboratory for about 3 years. Predictive control mechanism has offered us flexible methods to implement a grammar. As described in the preceding section, each context-free rule works as an active process that can pass messages to another process or create a new process. The process can also notify the parser the applicability of the rule attached. The newly created process works as if it were a look-forward demon. The <advice> attached to each (context-free) rule expresses an action of the rule (or process). Semantic information will become available during parsing process through the message control mechanism if necessary.

Automatic segmentation algorithms in our Extended LINGOL are reported in [6]. Various experiments have been carried out in our laboratory using Extended LINGOL. Some of them are reported in [5].

REFERENCES

- [1] Grishman, R.: "Implementation of the string parser of English" in "Natural Language Processing", Algorithmic Press, 1973, 89-109.
- [2] Heidorn, G. E.: "Augmented Phrase Structure Grammar", in Schank and Nash-Webber (Eds.): "Theoretical Issues in Natural Language Processing", Cambridge, Mass. 1975, 1-5.
- [3] Pratt, V. R.: "A Linguistic Oriented Programming Language", IJCAI3, 1973, 372-381.
- [4] Pratt, V. R.: "LINGOL - A Progress Report", IJCAI4, 1975, 422-428.
- [5] Tanaka, H., Sato, T. and Motoyoshi, F.: "EXPLUS--A Semantic Parsing System for Japanese Sentences", 3rd USA-JAPAN Computer Conference, 1977, 236-240.
- [6] "Manual of Extended LINGOL", Machine Inference Section, Electrotechnical Laboratory, Tokyo, 1978 (in Japanese version).