

TOWARDS MINIMAL DATA STRUCTURES
FOR DETERMINISTIC PARSING

David W. Shipman
Bell Laboratories
Room 2D-443
Murray Hill, New Jersey 07974

Mitchell P. Marcus
Artificial Intelligence Laboratory
Massachusetts Institute of Technology
545 Technology Square
Cambridge, Massachusetts 02143

The determinism hypothesis suggests that natural language may be parsed in a single pass without resort to backtracking techniques. The PARSIFAL system, developed by Marcus, incorporates this philosophy in an English language parser. Here, we show that the data structures used by this parser may be considerably simplified resulting in more elegant grammatical specifications.

Keywords: artificial intelligence, natural language processing, deterministic parsing

1. THE DETERMINISM HYPOTHESIS

Marcus [1][2] proposes that natural language may be parsed in a fully deterministic manner. Under this determinism hypothesis portions of the parse-tree, once constructed, necessarily become part of the final parse-tree output. This is in contrast to the traditional backtracking paradigm in which "guesses" are made about the syntactic structure of the input sentence; and, when the guess proves later to be incorrect, any parse-tree structures resulting from that guess must be erased prior to testing alternate hypotheses.

An important discovery was that English could be parsed deterministically by using an, at most, three-constituent look-ahead strategy. This differs from the k-token look-ahead strategy used by an LR(k) parser in that the bounded look-ahead is with reference to parsed constituents rather than to individual words of the sentence. For example, the sequence <auxverb> <noun phrase> appearing at the beginning of the sentence would indicate that a yes-no question was being parsed. Note that in order for this sequence to appear in the look-ahead buffer, it would have been necessary to first parse the noun phrase in the second position. This is accomplished by means of an attention shifting rule in the grammar. In the example above, the attention shifting rule would detect the beginning of a noun phrase starting in the second position of the look-ahead buffer. It would then shift the look-ahead window one position to the right, just

past the <auxverb>, and activate grammar rules suitable for parsing noun phrases. The noun phrase parse then operates only with respect to words in the input sentence to the right of the <auxverb>. When the noun phrase parse is complete, attention shifts back to the original look-ahead window position with the <auxverb> in first position and the new <noun phrase> in the second position.

By using such an approach it is possible to parse natural language in a time on the order of the length of the input sentence. A more detailed discussion, including arguments for the psychological reality of the determinism hypothesis can be found in [1][2].

This deterministic mechanism has been incorporated as part of the PARSIFAL system developed by Marcus. In this paper we propose a simplification of the data structures employed by PARSIFAL. The change significantly lessens the amount of detail which must be explicitly specified in the rules of the grammar. This results from the fact that this detail can now be implicitly embedded in the control structure of the parser.

The new approach will be presented after a cursory overview of the current PARSIFAL mechanisms.

"Is the boy John saw yesterday going to the party?"

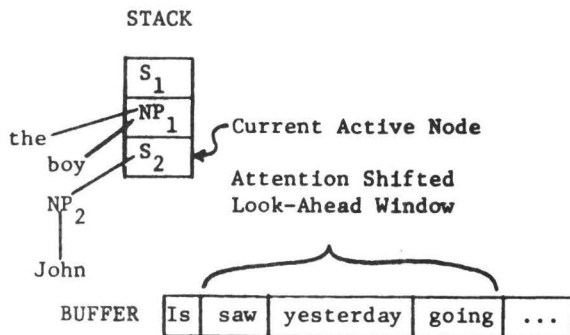


Figure 1. PARSIFAL's STACK and BUFFER

2. THE PARSIFAL APPROACH

2.1 Data Structures

PARSIFAL uses two data structures to record parsing progress (Fig. 1). These are the Stack and the Buffer. The Buffer is the look-ahead buffer discussed in the previous section. The Stack is used to hold partial structures which have already been parsed by the system. The Buffer contains only terminals and complete sub-trees, while the Stack contains sub-trees under construction. (Note that the Stack differs from the pushdown stack used by backtracking parsers in that structures once parsed and placed on the Stack are always used later as subordinates of larger structures.) Instructions to move nodes between the Buffer and the Stack must be explicitly stated in the system's grammar rules. Grammar rules build the structures on the stack by removing elements from the Buffer and attaching them to the most recent node on the Stack, this may be a newly created non-terminal node or a previously created non-terminal which already has some subordinate structures attached to it. When a complete sub-tree has been built on the stack, the parent node is dropped from the stack back into the buffer to be used there in the construction of higher-level constituent structures.

2.2 Attention Shifting Rules

Attention shifting rules are responsible for adjusting the look-ahead window of the Buffer to the proper position for parsing a new sub-constituent. They must be treated as a special case since they need to be able to scan the buffer to locate the beginning of the sub-constituent. Once this has happened, the new node is created, pushed on the stack and

the look-ahead window is explicitly advanced. Special attention must also be given to the problem of restoring the look-ahead window after the sub-constituent parse is complete.

2.3 Node Reactivation Rules

An additional type of grammar rule is the node reactivation rule. Its function can perhaps best be explained through an example. Consider the case of a noun phrase which includes an attached relative clause. During the processing of the relative clause it is sometimes necessary to access the attributes of parent nodes of the noun phrase under construction. However, since the noun phrase sub-tree has not yet been completed at the time the relative clause is being parsed, the noun phrase is not attached to a parent. Thus, access to parent attributes from the relative clause would be impossible. To remedy this situation, the noun phrase parse tree is prematurely taken to be complete just prior to parsing the relative clause. The noun phrase can then be attached by its parent node. A node reactivation rule is then called into play. It "reactivates" the noun phrase processing in order to parse the relative clause. Now, the grammar rules for processing the relative clause are able to reference the parents of the noun phrase.

Thus, in figure 1, if it were desirable to attach the NP_1 to S_1 as its subject, NP_1 would have to be prematurely marked complete and later reactivated to continue parsing the phrase.

Node reactivation rules were intended to be only a temporary solution to certain parsing problems. The data structuring approach described in the next section eliminates the need for such a reactivation mechanism.

3. THE UNIFIED APPROACH

3.1 A Unified Data Structure

Under the proposed strategy, the former Stack and Buffer are combined into a single Buffer.

"Is the boy John saw yesterday going to the party?"

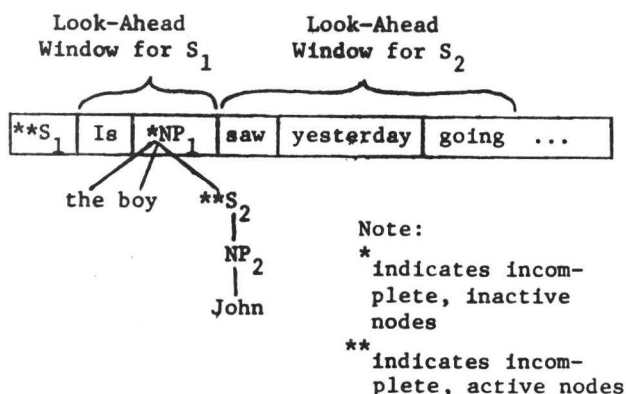


Figure 2. Unified BUFFER Organization

This buffer holds both complete and incomplete sub-trees (see Fig. 2). For each incomplete sub-tree, the most subordinate incomplete node is considered to be the active node for that subtree. A parsing process is associated with each active node. This parsing process operates over a look-ahead window consisting of the three Buffer cells following that Buffer cell which contains the active node.

Because the new Buffer structure now contains all the information previously contained in the Stack as well, the explicit instructions which transferred data between the two structures are no longer necessary.

"Is the boy John saw yesterday going to the party?"

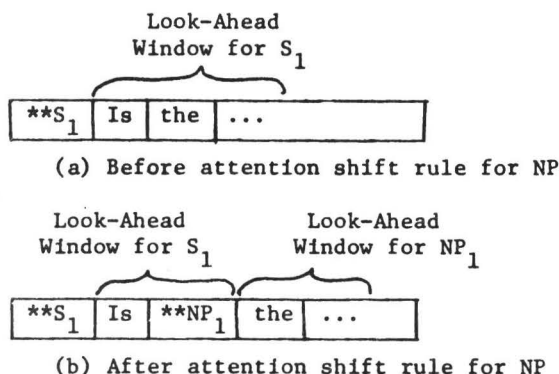


Figure 3. Automatic Attention Shifting

3.2 Automatic Attention Shifting

Under this scheme, the attention shifting rules simply insert a newly created node into the Buffer (see Fig. 3). Since a newly created node is necessarily an incomplete node, it is automatically active. The look-ahead window for the new node, as for any active node, is defined to be the next three cells in the Buffer. No explicit mechanism for shifting attention is needed.

3.3 Automatic Node Reactivation

All active nodes are considered to be parsing their look-ahead windows in parallel. Note however that the look-ahead may not extend beyond an incomplete node. This is because this portion of the buffer is actively being considered for attachment by the incomplete node.

Thus, in figure 3b, it would be possible for S_1 to attach NP_1 as its subject without a deactivation and subsequent reactivation of NP_1 .

The mechanism can be thought of as a kind of linear cellular automaton where each cell has access to the contents of up to three cells to its right. Because multiple cells can be active at the same time, there is no need for an explicit reactivation mechanism.

REFERENCES

- [1] Marcus, M. P. A Theory of Syntactic Recognition for Natural Language Ph.D Thesis, Department of Electrical Engineering, Massachusetts Institute of Technology, Cambridge, Massachusetts 1978
- [2] Marcus, M. P. "A Computational Account of the Syntactic Recognition of Natural Language" Proceedings of the Workshop on Computational Aspects of Linguistic Structure and Discourse Setting Cambridge University Press [to appear]