83-15

UIUCDCS-F-83-908

ISG 83-8

ON THE REPRESENTATION OF RULES

by

Jorge L. Orejel-Opisso

January 1983

Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

This research was supported, in part, by the National Science Foundation under Grant No. MCS-82-05166.

ON THE REPRESENTATION OF RULES

ABSTRACT

This paper reports an efficient rule-representation scheme to be used in a computer program for splitting words. The work is a continuation of earlier efforts to mechanize word splitting in natural language, and is a first step towards the development of a computer program capable of learning such a skill.

The scheme presented here can be seen to solve some problems inherent in the location of applicable rules and in the modification of rules. As a side-effect, it is suitable to the learning process, at least in the particular problem domain considered.

Additionally, a simple rule-description language is presented along with an informal description of its semantics.

Key words: Expert systems, machine learning, rule-representation, natural language processing, word-splitting, pattern-directed invocation, declarative knowledge.

ACKNOWLEDGMENT

The author wishes to thank the suggestions received from Prof. R.S. Michalski with regard to the first draft of this paper.

INTRODUCTION

Statement of the Problem

When writing a text, it is common to face a situation in which a word cannot be entirely written down at the end of a line --we say that the word must be split between two lines. The process of word-splitting entails determining a word's constituent syllables to determine the one up to which the word can be written down, yet without exceeding the line length.

At first glance, syllabication appears to convey no problems at all; perhaps because it has becomed an unconscious process. Yet people, more often than not and even on purpose, do incorrect syllabication. Remarkably, people also split unfamiliar words.

How can syllabication be explained? People are either taught a suitable set of rules or, alternatively, learn by heart the syllables of each word in their active vocabulary. In any case, a syllable is essentially an "atomic" phonetic entity, so familiar, that people are mostly unaware of it.

English has four basic syllabication rules, barely of help to the process. These rules are {Legget 1982}:

a) Never divide words of one syllable.
b) Never divide a word so that a single letter stands alone in a line.
c) When dividing a compound word that already contains a hyphen, make the break where the hyphen occurs.
d) When in doubt about the syllabication of a word, consult a good dictionary.

The rules are too broad to be used practically --in fact, the last one seems to cover most of the cases and represents a rather boring approach. Therefore, it is fair to say that there are no explicit rules to resort to in the process of word splitting, the reason being that it is so natural that people never think of the existence

of rules governing it. Much the same can be said about Spanish although there are some rules dealing with indivisible pairs of letters such as "ll", "rr" and "ch".

Word-splitting by Computer

Even though syllabication is not a simple task, it seems to be a mechanical process (when performed by educated people) which, by definition, is governed by a set of well-established rules. If this is the case, a computer can be programmed to split words in much the same way as people do. The major hypothesis of this research is that people actually have, and unconsciously use, a set of syllabication rules synthesized and refined through experience.

The problem suggested two possible lines of work. In the one hand, an expert system was built which incorporated some syllabication rules for Spanish {Orejel 1982}. As discussed in the following sections, this paper presents a more efficient way to represent these rules in the current system. On the other hand, the skill of word splitting offers a concrete problem domain which is small enough to undertake the design of a learning program, that is, a program able to acquire by experience such a skill. Descriptions of two programs that show a limited amount of learning from their mistakes can be found in {Sussman 1975} and {Winston 1970}. A side-effect of the rule-representation scheme to be discussed is that it is suitable to the learning process.

REPRESENTATION OF RULES

Two problems that must be solved when designing a rule-based expert system are: (a) Deciding how to locate applicable rules and (b) What to do if, in a given situation, several rules are satisfied simultaneously. To this end some authors {Lenat et al 1982} have suggested the introduction of "strategic meta-knowledge" into the expert system. By constraining the search for a solution, strategic meta-knowledge is a much better alternative to the otherwise blind approach of testing the conditions of each rule given to the system.

Strategic meta-knowledge may take the form of simple meta-rules such as "if several rules are applicable at the same time, arbitrarily choose one." In some problem domains, however, domain-specific knowledge guides the selection.

With regard to the domain of word-splittting, a quite primitive kind of meta-knowledge was present in the first version of the program. It took the form of a distinction between consonants rules and vowels rules to reduce the search time, and an implicit ordering of rules according to their scope to determine their applicability {Orejel 1982}. In spite of that the system retained much inefficiency.

Other domains may not require the use of strategic meta-knowledge. Chess represents an instance in which computer-generated decision trees have been used to classify legal starting positions in a restricted endgame {Shapiro and Niblett 1981}. Every synthesized decision tree is in fact a single rule that classifies correctly all the positions of such an endgame. Clearly, a single rule eliminates the search problem. As discussed next, the same situation applies to word-splitting. At least in this domain, the new rule-representation acheme will solve the problem of rule location as well as the possible conflicts among rules.

Following the conventions established in the first version of the program (Orejel 1982), a word to be split will be represented by a pattern of V's, standing for the vowels, and K's, standing for the consonants. There is a slight change in the strategy for handling words: the remaining spaces in the line will determine how many leading letters (constituting a prefix) of the word are to be considered to find syllables. For instance, assuming that the word 'progression' is under consideration for splitting and that there are six spaces available then the prefix taken would be 'progre'. This prefix would, in turn, be represented by the pattern 'KKVKKV'.

In the new rule-representation scheme, the condition parts of syllabication rules form a decision tree. Branches are labeled with a symbol taken form the set

$$\{\Lambda, V, K\}$$

where 'Λ' stands for the empty string, 'V' and 'K' being interpreted as before. The function of those labels will become clear later. Every internal node in the tree may have at most three branches, and every terminal node will be thought of as a pointer to the action part of some rule. Fig.1 shows the decision tree corresponding to the syllabication rules of Spanish. To avoid repeating the action parts of rules and their extensions, rules are identified with the same numbers given in (Orejel 1982, Table 2).

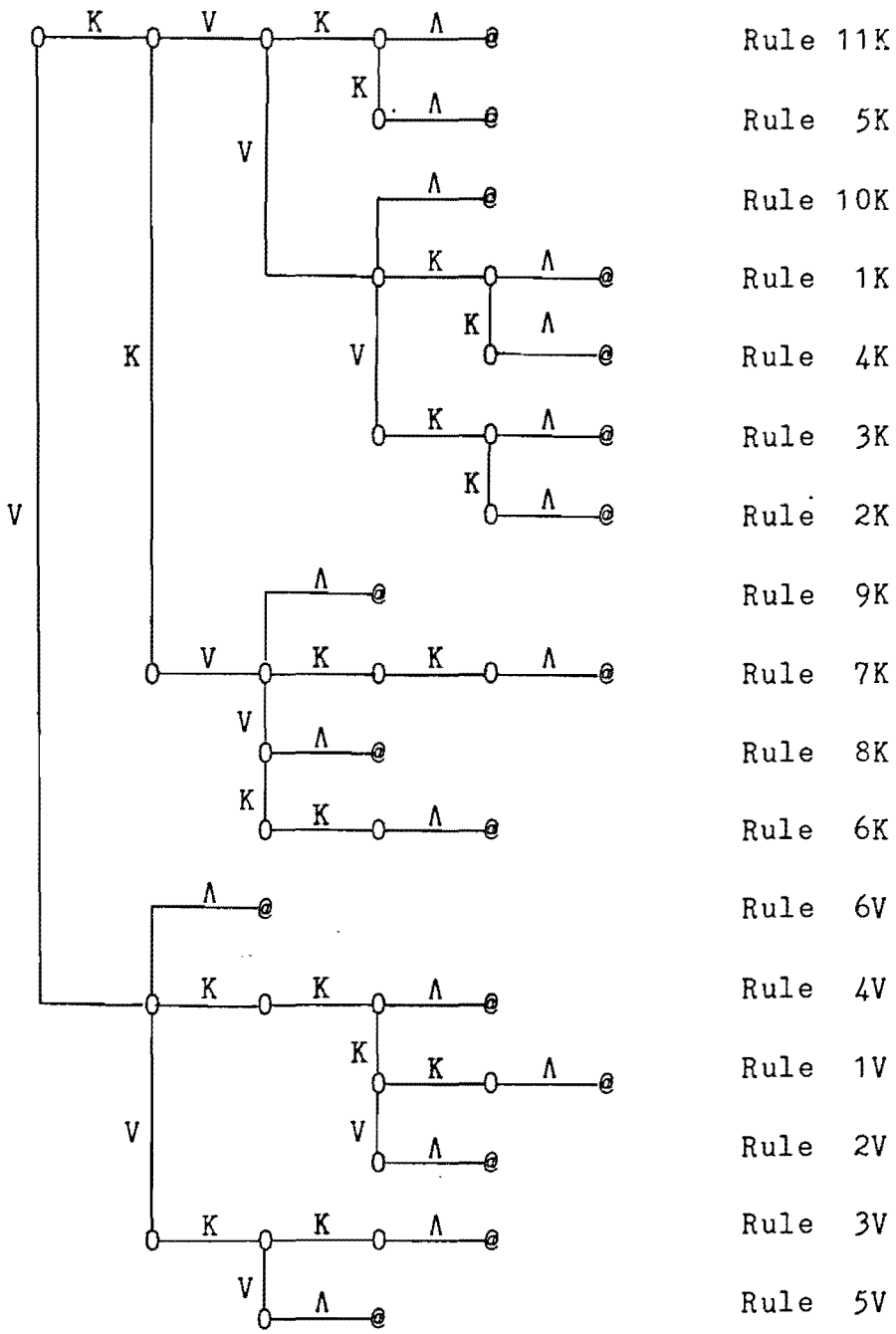


Figure 1. Decision-tree representation of the condition parts of syllabication rules for Spanish. 0 denotes an internal node, and @ denotes a terminal node.

Now it is an easy matter to find the rules when looking for syllables in a word to be split. Using the pattern representing the prefix, a syllable is determined by traversing the decision tree until a terminal node, hence a rule, is reached. This is yet another example of pattern-directed rule invocation (Sussman and McDermott 1972; Stallman and Sussman 1976).

Execution of the action part of a rule entails shortening the prefix. Then the process is repeated on the remaining portion of the prefix. There are two conditions for termination:

a) The prefix is exhausted normally: the process has produced the empty string and the entire original prefix may be written down on the line.

b) A 'dead end' is reached: it is impossible to follow a branch because the prefix does not contain the appropiate label. The system must back up the traversal until either finding an internal node having a branch labeled 'Λ' or reaching the root node. In the first case a rule is found and the overall process may continue whereas in the second case it stops.

Table I depicts the steps followed to split two example words. While seeing these examples, note that branches labeled 'Λ' in the decision tree serve two purposes: they tell what to do when the prefix is the empty string, or they can be used as the only alternative to reach a rule even though the prefix is not the empty string.

Table I. Two examples of word splitting. A rule number followed by an asterisk indicates that an exception to the rule was satisfied.

(a) Split the word 'desaparicion' having six spaces available in the line. Prefix = 'desapa' , pattern = 'KVKVKV'

(b) Split the word 'condado' having four spaces available in the line. Prefix = 'cond' , pattern = 'KVKK'.

Case	Step	Remaining prefix	Rule found	Syllable determined
a	0	KVKVKV	11K*	'des'
	1	VKV	6V	'a'
	2	KV	none	none
b	0	KVKK	5K	'con'
	1	K	none	none

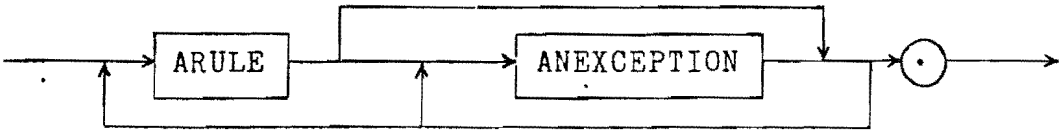
Results: (a) desa- paricion
(b) con- dado

A SIMPLE RULE-DESCRIPTION LANGUAGE

A useful way to let an expert system acquire knowledge about the problem domain is through a rule-description language. This feature is particularly helpful when the program is to have some advice from the programmer. Usually the description of rules constitutes the declarative knowledge of the system, and the advantages of this declarative approach have been shown in a number of applications {Stallman and Sussman 1976; Sussman and Steele 1980; deKleer and Sussman 1980; Steele 1980}. A rule-description language, on the other hand, eases the interaction with the system since rules may be described in a sort of high-level language. Hopefully the ultimate goal would be communicate in the natural language of the user, but this is yet to come.

Next we present the syntax diagrams of a language for describing syllabication rules. When necessary, an informal discussion of its semantics follows a syntax diagram. For the interested reader, an appendix lists the syllabication rules for Spanish written in terms of such a language.

RULES

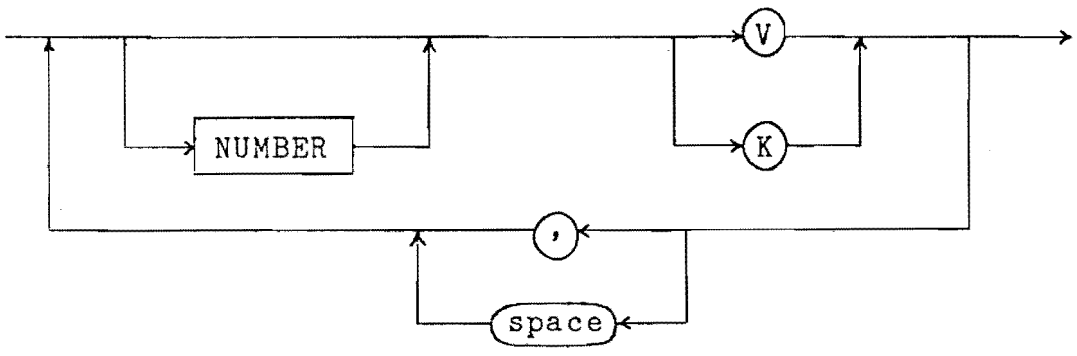


{ The description consists of one or more rules, each followed by an arbitrary number of exceptions. Once processed, a rule is assigned a unique identification number. A period signals the end of the description. }

ARULE

{ A rule states a sequence of characters that signal presence of a syllable, followed by the number of leading characters in the sequence that actually comprise the syllable }

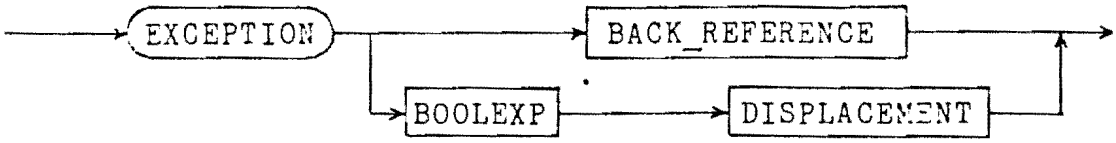
SEQUENCE



{ In describing sequences of characters a convention is observed, namely 'V' stands for any vowel and 'K' stands for any consonant }

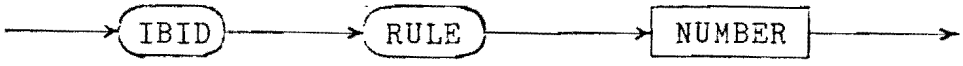
DISPLACEMENT

ANEXCEPTION

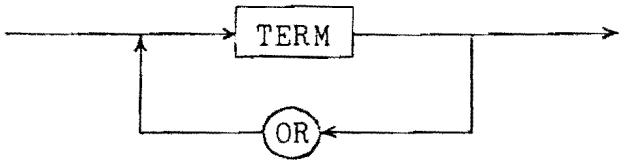


{ Exceptions to a rule may be the same as that (those) of a previous rule, or may be a Boolean expression followed by a new indication of the number of characters in the syllable }

BACK_REFERENCE

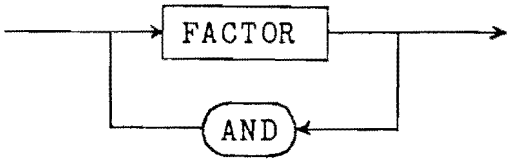


BOOLEXP



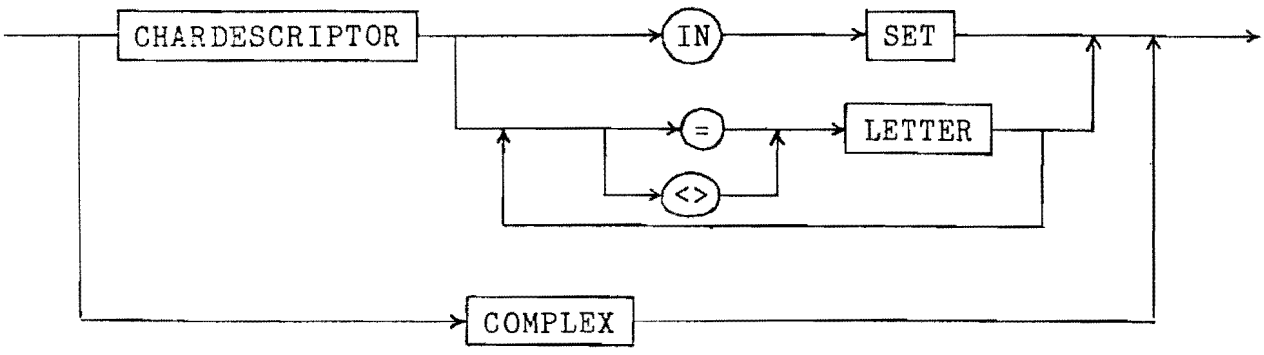
{ A Boolean expression may be the disjunction of an arbitrary number of terms. The operator 'OR' has the weakest precedence and is left-associative }

TERM



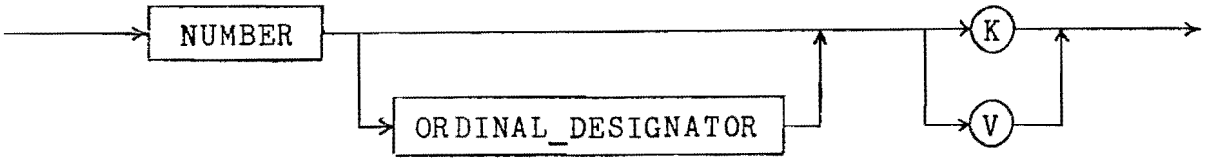
{ A term may be the conjunction of an arbitrary number of factors. The operator 'AND' has higher precedence than 'OR' and is also left-associative }

FACTOR



{ A factor establishes some conditions that must be met by characters appearing in certain positions of the word. Alternatively it may be an specification of the words prefix }

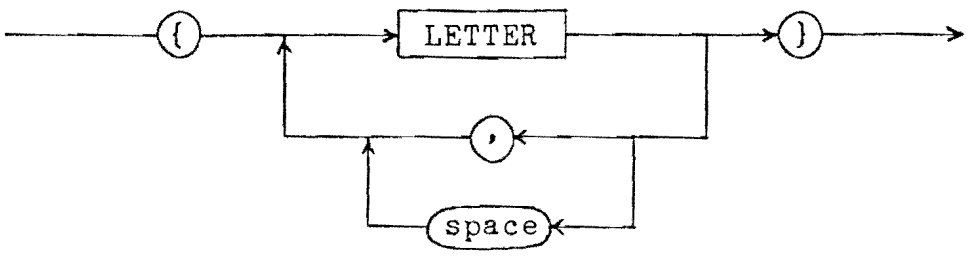
CHARDESCRIPTOR



{ A character within a word is specified by its relative

position with respect to the beginning of the word and by its class (vowel or consonant.) An optional 'ordinal-designator', which is ignored by the system, adds clarity to the position indicator }

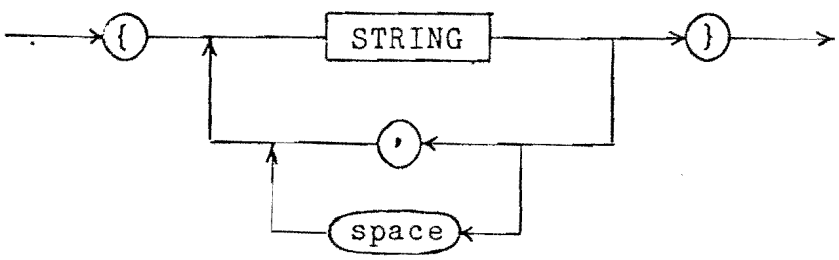
SET



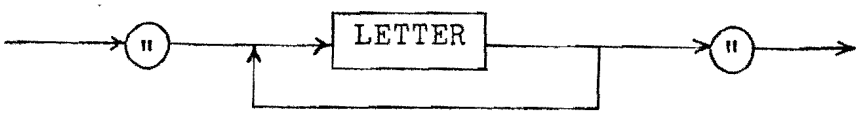
COMPLEX

{ This construct allows a rule to require the prefix under consideration to have a meaning. At present, only one syllabication rule for Spanish has made use of it }

STRING_SET



STRING



CONCLUSIONS AND FUTURE WORK

A scheme for representing syllabication rules in terms of a decision tree has been discussed along with an elementary rule-description language. The scheme not only improves the search for rules, but also is easily modifiable at the implementation level.

In the light of the future development of a program that learns the skill of word splitting, the proposed organization is particularly useful to the learning process where speed in rule location and ease of rule modification are paramount. However, since rules may have attached exceptions of arbitrary complexity, it remains to propose an ordering on the exceptions to quickly find the relevant one.

An additional advantage concerns the issue of "knowledge compression" {Michie 1982}. The use of decision trees to encode knowledge is an excellent alternative to the simple but expensive approach (in terms of storage use) of data-base lookup, the data base being in this case a dictionary containing the constituent syllables of each word.

The next step in this research involves the design of the basic learning process (the learning "primitives"), including a mechanism enabling the program to monitore (and possibly modify) itself. This problem has remained unsolved for quite a while although recently some researchers {Lenat et al 1982; Doyle 1980} have set forth long-term research directions about it.

APPENDIX

Syllabication rules for Spanish, in terms of the rule-description language. As written here, they are essentially a restatement of Table 2 given in {Orejel 1982}. Comments are enclosed in parentheses.

```
( 1V)  RULE  V 4K   TAKE 3
( 2V)  RULE  V 3K  V   TAKE 3
           EXCEPTION 4th K = h  OR
                     4th K = l  AND  3rd K <> r  OR
                     4th K = r  AND  3rd K IN {t c p b}   TAKE 2
( 3V)  RULE 2V 2K   TAKE 3
( 4V)  RULE  V 2K   TAKE 2
           EXCEPTION 2nd K IN {b c p t}  AND  3rd  K IN {h l r}
                     OR  2nd K = 3rd K = l   TAKE 1
( 5V)  RULE 2V  K  V   TAKE 2
( 6V)  RULE  V   TAKE 1

( 1K)  RULE  K 2V  K   TAKE 0
( 2K)  RULE  K 3V 2K   TAKE 5
( 3K)  RULE  K 3V  K   TAKE 4
( 4K)  RULE  K 2V 2K   TAKE 4
           EXCEPTION 5th K IN {r l}   TAKE 3
( 5K)  RULE  K  V 2K   TAKE 3
           EXCEPTION IBID RULE 2  (rule 2V)
( 6K)  RULE 2K 2V 2K   TAKE 5
           EXCEPTION 5th K = 6th K = l   TAKE 4
( 7K)  RULE 2K  V 2K   TAKE 4
           EXCEPTION IBID RULE 10  (rule 4K)
( 8K)  RULE 2K 2V   TAKE 4
( 9K)  RULE 2K  V   TAKE 3
(10K)  RULE  K 2V   TAKE 3
(11K)  RULE  K  V  K   TAKE 2
           EXCEPTION PREFIX IN {"Des" "des"}   TAKE 3 .
```

REFERENCES

deKleer, J. and G.J. Sussman (1980). Propagation of Constraints Applied to Circuit Synthesis, MIT AI Lab. Memo 485.

Doyle, J. (1980). A Model for Deliberation, Action and Introspection, MIT AI Lab. TR-581.

Legget, G. et al (1982). Handbook for Writers, Prentice Hall, 8th ed., pp. 114-115.

Lenat, D. et al (1982). Meta Cognition: Reasoning About Knowledge. In Hayes-Roth, Waterman and Lenat, eds. Building Expert Systems, Addison Wesley.

Michie, D. (1982). Class Note No. 12. Course CS-397, Knowledge-based Programming. University of Illinois.

Orejel, J.L. (1982). Word Hyphenation by Computer in Spanish-written texts. (Unpublished paper).

Shapiro, A. and T. Niblett (1981). Automatic Induction of Classification Rules for a Chess Endgame, Memo MIP-R-129, Machine Intelligence Research Unit, Univ. of Edinburgh.

Stallman, R.M. and G.J. Sussman (1976). Forward Reasoning and Dependency-Directed Backtracking In a System for Computer-Aided Circuit Analysis, MIT AI Lab. Memo 380.

Steele, G.L. (1980). The Definition and Implementation of a Computer Programming Language Based on CONSTRAINTS, MIT AI Lab. TR-595.

Sussman, G.J. and D.V. McDermott (1972). Why Conniving is Better than Planning, MIT AI Lab. Memo 255A.

Sussman, G.J. (1975). A Computer Model of Skill Acquisition, Elsevier.

Sussman, G.J. and G.L. Steele (1980). CONSTRAINTS - A Language for Expressing Almost-Hierarchical Descriptions, Artificial Intelligence 14, pp. 1-39.

Winston, P.H. (1970). Learning Structural Descriptions from Examples. In Winston, P.H. ed. The Psychology of Computer Vision, McGraw-Hill, ch. 5 (1975).

BIBLIOGRAPHIC DATA SHEET	1. Report No. UIUCDCS-F-83-908	2.	3. Recipient's Accession No.
4. Title and Subtitle On the Representation of Rules			5. Report Date January 1983 6.
7. Author(s) Jorge L. Orejel-Opisso			8. Performing Organization Rept. No.
9. Performing Organization Name and Address Department of Computer Science University of Illinois Urbana, IL			10. Project/Task/Work Unit No. 11. Contract/Grant No. MCS 82-05166
12. Sponsoring Organization Name and Address National Science Foundation Washington, DC			13. Type of Report & Period Covered 14.

15. Supplementary Notes

16. Abstracts

This paper reports an efficient rule-representation scheme to be used in a computer program for splitting words. The work is a continuation of earlier efforts to mechanize word splitting in natural language, and is a first step towards the development of a computer program capable of learning such a skill.

The scheme presented here can be seen to solve some problems inherent in the location of applicable rules and in the modification of rules. As a side-effect, it is suitable to the learning process, at least in the particular problem domain considered.

Additionally, a simple rule-description language is presented along with an informal description of its semantics.

17. Key Words and Document Analysis. 17a. Descriptors

Expert systems, machine learning, rule-representation, natural language processing, word-splitting, pattern-directed invocation, declarative knowledge.

17b. Identifiers/Open-Ended Terms

17c. COSATI Field/Group

18. Availability Statement	19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 18
	20. Security Class (This Page UNCLASSIFIED	22. Price

INSTRUCTIONS FOR COMPLETING FORM NTIS-35 (10-70) (Bibliographic Data Sheet based on COSATI Guidelines to Format Standards for Scientific and Technical Reports Prepared by or for the Federal Government, PB-180 600).

1. **Report Number.** Each report shall carry a unique alphanumeric designation. Select one of the following types: (a) alphanumeric designation provided by the sponsoring agency, e.g., FAA-RD-68-09; or, if none has been assigned, (b) alphanumeric designation established by the performing organization e.g., FASEB-NS-87; or, if none has been established, (c) alphanumeric designation derived from contract or grant number, e.g., PH-43-64-932-4.

2. Leave blank.

3. **Recipient's Accession Number.** Reserved for use by each report recipient.

4. **Title and Subtitle.** Title should indicate clearly and briefly the subject coverage of the report, and be displayed prominently. Set subtitle, if used, in smaller type or otherwise subordinate it to main title. When a report is prepared in more than one volume, repeat the primary title, add volume number and include subtitle for the specific volume.

5. **Report Date.** Each report shall carry a date indicating at least month and year. Indicate the basis on which it was selected (e.g., date of issue, date of approval, date or preparation.

6. **Performing Organization Code.** Leave blank.

7. **Author(s).** Give name(s) in conventional order (e.g., John R. Doe, or J.Robert Doe). List author's affiliation if it differs from the performing organization.

8. **Performing Organization Report Number.** Insert if performing organization wishes to assign this number.

9. **Performing Organization Name and Address.** Give name, street, city, state, and zip code. List no more than two levels of an organizational hierarchy. Display the name of the organization exactly as it should appear in Government indexes such as USGRDR-I.

10. **Project/Task/Work Unit Number.** Use the project, task and work unit numbers under which the report was prepared.

11. **Contract/Grant Number.** Insert contract or grant number under which report was prepared.

12. **Sponsoring Agency Name and Address.** Include zip code.

13. **Type of Report and Period Covered.** Indicate interim, final, etc., and, if applicable, dates covered.

14. **Sponsoring Agency Code.** Leave blank.

15. **Supplementary Notes.** Enter information not included elsewhere but useful, such as: Prepared in cooperation with . . . Translation of . . . Presented at conference of . . . To be published in . . . Supersedes . . . Supplements . . .

16. **Abstract.** Include a brief (200 words or less) factual summary of the most significant information contained in the report. If the report contains a significant bibliography or literature survey, mention it here.

17. **Key Words and Document Analysis. (a). Descriptors.** Select from the Thesaurus of Engineering and Scientific Terms the proper authorized terms that identify the major concept of the research and are sufficiently specific and precise to be used as index entries for cataloging.
(b). Identifiers and Open-Ended Terms. Use identifiers for project names, code names, equipment designators, etc. Use open-ended terms written in descriptor form for those subjects for which no descriptor exists.
(c). COSATI Field/Group. Field and Group assignments are to be taken from the 1965 COSATI Subject Category List. Since the majority of documents are multidisciplinary in nature, the primary Field/Group assignment(s) will be the specific discipline, area of human endeavor, or type of physical object. The application(s) will be cross-referenced with secondary Field/Group assignments that will follow the primary posting(s).

18. **Distribution Statement.** Denote releasability to the public or limitation for reasons other than security for example "Release unlimited". Cite any availability to the public, with address and price.

19 & 20. **Security Classification.** Do not submit classified reports to the National Technical Information Service.

21. **Number of Pages.** Insert the total number of pages, including this one and unnumbered pages, but excluding distribution list, if any.

22. **Price.** Insert the price set by the National Technical Information Service or the Government Printing Office, if known.