INFORMATION TO USERS

This was produced from a copy of a document sent to us for microfilming. While the most advanced technological means to photograph and reproduce this document have been used, the quality is heavily dependent upon the quality of the material submitted.

The following explanation of techniques is provided to help you understand markings or notations which may appear on this reproduction.

1. The sign or “target” for pages apparently lacking from the document photographed is “Missing Page(s)”. If it was possible to obtain the missing page(s) or section, they are spliced into the film along with adjacent pages. This may have necessitated cutting through an image and duplicating adjacent pages to assure you of complete continuity.

2. When an image on the film is obliterated with a round black mark it is an indication that the film inspector noticed either blurred copy because of movement during exposure, or duplicate copy. Unless we meant to delete copyrighted materials that should not have been filmed, you will find a good image of the page in the adjacent frame.

3. When a map, drawing or chart, etc., is part of the material being photographed the photographer has followed a definite method in “sectioning” the material. It is customary to begin filming at the upper left hand corner of a large sheet and to continue from left to right in equal sections with small overlaps. If necessary, sectioning is continued again—beginning below the first row and continuing on until complete.

4. For any illustrations that cannot be reproduced satisfactorily by xerography, photographic prints can be purchased at additional cost and tipped into your xerographic copy. Requests can be made to our Dissertations Customer Services Department.

5. Some pages in any document may have indistinct print. In all cases we have filmed the best available copy.

University Microfilms International
300 N ZEEB ROAD, ANN ARBOR, MI 48106
18 BEDFORD ROW, LONDON WC1R 4EJ, ENGLAND
TREATMENT OF UNGRAMMATICAL AND EXTRA-GRAMMATICAL PHENOMENA
IN NATURAL LANGUAGE UNDERSTANDING SYSTEMS

DISSERTATION

Presented in Partial Fulfillment of the Requirements for
the Degree Doctor of Philosophy in the Graduate
School of The Ohio State University

By

Stan C. Kwasny, B.S., M.A.T., M.S.

* * * * *

The Ohio State University
1980

Reading Committee:

Dr. Norman K. Sondheimer
Dr. H. William Buttelmann
Dr. B. Chandrasekaran
Dr. Marshall C. Yovits

Approved By

Adviser
Department of Computer and Information Science
to Carol and Jennifer
ACKNOWLEDGMENTS

This work represents more than just an individual effort on the part of the author. Without constant encouragement from my adviser and friend, Norm Sondheimer, the ideas in this work would have remained just ideas.

Bill Buttelmann has helped to develop in me that sixth sense exhibited by researchers in judging what constitutes good research. I value this as perhaps the single most important ability one obtains while in pursuit of an advanced degree.

Professor Chandrasekaran has influenced my academic life in a variety of ways. As my teacher and mentor in Artificial Intelligence, he has been especially influential in shaping my view of that world.

Professor Yovits' influence on my academic life has also been extensive. As teacher, professor, chairman of my department, and member of my reading committee, he has helped to guide me with his wisdom.

My first exposure to the field of Artificial Intelligence, to the field of Computational Linguistics, and to Augmented Transition Networks can be traced to Stu Shapiro. I will always remember his intellectually stimulating and intriguing discussions.

I am grateful to Professor N. John Castellan through whose influence I first became seriously interested in studying Computer Science.

Research sponsored in part by The Air Force Office of Scientific Research, Air Force Systems Command, USAF, under Grant No. AFOSR-75-2811. The United States government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon.
VITA

January 5, 1947 . . . . Born - Hammond, Indiana

1969. . . . . . . . . . . . . . . . . . B.S., Indiana University,
Bloomington, Indiana

1973. . . . . . . . . . . . . . . . . . M.A.T., Indiana University,
Bloomington, Indiana

1974. . . . . . . . . . . . . . . . . . M.S., Indiana University,
Bloomington, Indiana

1977. . . . . . . . . . . . . . . . . . M.S., The Ohio State University,
Columbus, Ohio

1977-79 . . . . . . . . . . . . . . . . Researcher, Battelle Columbus
Laboratories, Columbus, Ohio

PUBLICATIONS

with Stuart C. Shapiro, "Interactive Consulting via Natural

with D. C. Brown, B. Chandrasekaran, and N. K. Sondheimer,
"An Experimental Graphics System with Natural Language

with Robert T. Niehoff, "The Role of Automated Subject
Switching in a Distributed Information Network," ONLINE

with Robert T. Niehoff and Michael Wessels, "Overcoming the
Data-Base Vocabulary Barrier--A Solution," ONLINE MAGAZINE,
October, 1979.

FIELDS OF STUDY

Major Field: Computer & Information Science


Studies in Programming Languages. Professor Anthony P. Lucido

Studies in Systems Programming. Professor David K. Hsiao
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledgments</td>
<td>iii</td>
</tr>
<tr>
<td>Vita</td>
<td>iv</td>
</tr>
<tr>
<td>List of Figures</td>
<td>viii</td>
</tr>
<tr>
<td><strong>Chapter</strong></td>
<td></td>
</tr>
<tr>
<td>I. Introduction</td>
<td>1</td>
</tr>
<tr>
<td>II. Language Phenomena</td>
<td>11</td>
</tr>
<tr>
<td>Cooccurrence Restrictions</td>
<td>12</td>
</tr>
<tr>
<td>Ellipsis</td>
<td>15</td>
</tr>
<tr>
<td>Conjunction</td>
<td>21</td>
</tr>
<tr>
<td>III. Relaxation of Cooccurrence Restrictions</td>
<td>26</td>
</tr>
<tr>
<td>Relaxing Tests</td>
<td>27</td>
</tr>
<tr>
<td>Relaxing Categories</td>
<td>32</td>
</tr>
<tr>
<td>IV. Patterns of Language</td>
<td>38</td>
</tr>
<tr>
<td>Patterns and the Pattern Arc</td>
<td>39</td>
</tr>
<tr>
<td>Telegraphic Language</td>
<td>46</td>
</tr>
<tr>
<td>Extraneous Words and Punctuation</td>
<td>48</td>
</tr>
<tr>
<td>V. Contextual Patterns in Language</td>
<td>50</td>
</tr>
<tr>
<td>Automatic Production of Patterns</td>
<td>51</td>
</tr>
<tr>
<td>Expectations</td>
<td>52</td>
</tr>
<tr>
<td>Contextual Ellipsis</td>
<td>57</td>
</tr>
<tr>
<td>Other Work on Ellipsis</td>
<td>59</td>
</tr>
</tbody>
</table>

**vi**
### VI. CONJUNCTION

- Macro Arcs
- Forms of Conjunction
- Other Work on Conjunction

### VII. EXAMPLES FROM AN IMPLEMENTATION

- Sample Dialogue
- Example of Cooccurrence Relaxation
- Example of Correlative Conjunction

### VIII. SUMMARY

- Accomplishments
- Problematic Cases
- Future Work

### APPENDIXES

- A. Syntax of ATN Grammar Specification
- B. Overview of ATNs
- C. The NLG Grammar
- D. A Category Hierarchy
- E. Relaxable Tests
- F. Arc and Arcset Specifications

### BIBLIOGRAPHY
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Sample Sentences from Natural Language Graphics System</td>
<td>3</td>
</tr>
<tr>
<td>2. A Syntactic Category Hierarchy for Nouns</td>
<td>35</td>
</tr>
<tr>
<td>3. Arc Partitioning</td>
<td>55</td>
</tr>
<tr>
<td>4. NLG Sentence-Level Network</td>
<td>87</td>
</tr>
<tr>
<td>5. NLG Noun Phrase Network</td>
<td>89</td>
</tr>
<tr>
<td>6. Finite State Network with Regular Grammar</td>
<td>116</td>
</tr>
<tr>
<td>7. Recursive Transition Network with Context-Free Grammar</td>
<td>117</td>
</tr>
<tr>
<td>8. Augmented Transition Network Fragment</td>
<td>119</td>
</tr>
<tr>
<td>9. NLG Sentence-Level Network</td>
<td>123</td>
</tr>
<tr>
<td>10. NLG Noun Phrase Network</td>
<td>124</td>
</tr>
<tr>
<td>11. NLG Question Network</td>
<td>125</td>
</tr>
<tr>
<td>12. NLG Prepositional Phrase Network</td>
<td>126</td>
</tr>
<tr>
<td>13. NLG Touch Network</td>
<td>127</td>
</tr>
</tbody>
</table>
Chapter I

INTRODUCTION

Natural Language Understanding (NLU) systems usually contain a number of components, one being a grammar which specifies much of the linguistic structure for acceptable inputs. Such a grammar also controls the acceptability of potential input forms and is used for that purpose during parsing. With few exceptions (to be noted later), most NLU systems make little attempt to permit inputs rejected by grammatical processing (i.e., "ungrammatical inputs"), neither by attempting to understand them nor by providing helpful feedback to the user in the form of error messages.

The need to devise ways of handling such inputs is clear both because people regularly communicate through ungrammatical sentences and because current grammatical models are limited in what can be readily included in them. (For example, conjunction is often beyond their range.) We shall characterize sentences which are not handled because
of this latter reason as "extra-grammatical". Heretofore, researchers have paid little attention to this area of language research, yet Wilks in his survey of NLU systems emphasizes that "...understanding requires, at the very least, ...some attempt to interpret, rather than merely reject, what seem to be ill-formed utterances." [WIL76]

Our study has investigated several types of ungrammaticality. A cross-section of the phenomena can be found in Figure 1 where sentences from a dialogue with a Natural Language Graphics system are shown. These will be discussed in greater detail in Chapter VII.

In this study, rather than investigating a given linguistic theory in some suitable context, we have chosen to extend an existing computational formalism to make it more amenable to the specification of new types of language processing. Our motivation stems from a desire to improve the robustness of the language processing task as well as to increase the habitability [WAT68] of such a system.

We have chosen to focus on three broad categories of language phenomena: the manifestation of cooccurrence relations in language, ellipsis, and conjunction. In the next chapter, we will define each of these and present examples illustrating the extent to which we address each problem.
A SMALL CIRCLE AT <T>, PLEASE

MAKE A LINE WITH FROM <T> TO <T>

DRAW A SMALL CIRCLES AT <T>

DRAW A CIRCLE TO RADIUS FOUR UNITS HERE <T>

LINE <T> <T>

DRAW A LINE FROM <T> TO <T>

CALL THIS POINT P1 <T> AND THIS ONE P2 <T>

NOW CONNECT P1 AND P2 AND CALL THE LINE LP1P2

CONNECT <T> AND <T> AND <T> AND <T>

DRAW A CIRCLE WITH A 2 INCH DIAMETER AT P2

AND A 1 INCH DIAMETER AT P1

DRAW BOTH A SMALL VERTICAL LINE AND A SMALL HORIZONTAL LINE THROUGH (11 . 11)

MAKE LINES FROM P1 TO <T>, <T>, <T>, AND <T>

Figure 1

Sample Sentences
from Natural Language Graphics System
"<T>" indicates a touch to the screen to show position
Some of these phenomena have been studied previously in particular NLU systems. For example, cooccurrence restriction violations have been explored in a system by Weischedel [WEI78]. Ellipsis-handling mechanisms have been added to the LIFER system [HEN77] and to the PLANES system [WAL78]. The handling of conjunction has been investigated by Petrick, Postal, and Rosenbaum within a transformational approach to parsing [PET69], by Woods in the LUNAR system [WO072], by Winograd [WIN72], and by Biermann and Ballard in the Natural Language Computation project [BIE78]. This and other work will be discussed in more detail and compared with the work presented here in appropriate sections of this thesis.

Several approaches could be followed in handling such ungrammatical or extra-grammatical inputs. One approach is simply to revise the grammar to handle these extra cases as they are discovered. This may be a valid procedure in restricted domains (as Woods points out in [WO078]), but this approach has only limited applicability*. Besides

* It is interesting to follow Woods' own experiences with the LUNAR system in examining why some inputs are rejected by the grammar [WO073b]. It will always be impossible to anticipate every conceivable contingency.
missing several important linguistic generalizations concerning the formation of conjunctions, ellipsis, and ungrammatical inputs, the grammar would experience an unwarranted increase in complexity.

Other approaches which might seem viable have been explored recently in the context of speech understanding (see, for example, [MIL74], [PAX74], and [ROB74]). Many of these parsers are based on finding "islands of reliability" or something similar in the signal, working right to left as well as left to right. Other approaches try to improve the selection mechanism's performance in processing strictly left to right. We rejected trying to adapt these techniques since they only experienced limited success and they were aimed at overcoming noise in the signal and limitations of the analysis techniques, as opposed to user idiosyncracies (which we contend are more predictable).

In this thesis, we will not attempt to survey the entire literature on various approaches to understanding language and the models they entail, except as they relate to the goals of our research. Instead, we will concentrate on one model and the changes necessary to facilitate the inclusion of these linguistic phenomena. For several excellent surveys of the field, see [GRI75], [HEI76], [KEN77], and [PET76].
In our approach, we assume the existence of a "normative" grammar which specifies the structure of well-formed inputs to a particular NLU system and show how to build the machinery necessary for allowing the processing of many types of ungrammatical or extra-grammatical forms. Our approach assumes that just as the normative grammar can be precisely specified, so, too, the manner in which a given input can deviate from the standard form can be precisely specified as well. Specifying deviance is done succinctly and both the structural characteristics of the original grammar as well as its inherent efficiency in processing grammatical inputs is preserved.

With this approach, the grammar takes on the appearance of allowing a much wider range of acceptable sentences. In reality, the size of the language acceptable to the grammar has not changed, but rather those unacceptable sentences that are significantly close to acceptable ones are marked as to their deviance and accepted as such. Thus, the language processor is made more sophisticated to permit the specification of deviation, either through viewing the failure of certain tests required in the grammar as simply "negative evidence" and not grounds for rejection or through using the grammar as a complicated and imperfect pattern which is matched against the input in a non-precise manner.
The techniques and mechanisms are developed within the framework of Augmented Transition Network (ATN) grammars, proposed in their present form by Woods in [W0070] and elaborated in [W0073a]. An ATN is usually presented as a graph structure which indicates the possible paths that may be traversed during parsing. Recursion is an important feature of the network, in that any subnetwork may be recursively traversed as a requirement to traversing an arc at a higher level. The process of following a network is nondeterministic in the sense that backtracking may be required before finding the correct path to follow. While traversing, registers may be set to hold the result of computations on the input or portions of the input itself. For readers unfamiliar with ATNs, a brief introduction and history of the formalism is presented in Appendix B, as well as a discussion of an implementation of the model and our extensions.

The processing efficiency of the original normative grammar for grammatical inputs is not affected by the mechanisms we have added. ATN processing paths which exist in the original grammar are always attempted first in preference to any alternatives suggested by the mechanisms for treating non-grammatical inputs. Therefore, for a fully grammatical input, the same number of ATN arcs are attempted as would be tried with the original grammar. Only in cases
where the input is ungrammatical will additional alternatives be attempted. An improved backtracking mechanism has been implemented which distinguishes between grammatical and ungrammatical alternatives to accomplish this task.

The processing characteristics of the ATN grammar are preserved through the monitoring of ATN registers during ungrammatical processing. For example, whenever a structure is being built and an undefined register value is requested, a partial structure is built and the missing component is marked in the structure as undefined along with the name of the undefined register. The testing of undefined registers on arcs pursued during ungrammatical processing works in a similar way. Tests on these registers are defaulted so that parsing never becomes blocked merely because a register value was never set. Structures produced as a result of following ungrammatical alternatives are marked as deviant through the association of deviance notes with the parse from which other system components can note how the sentence deviated during parsing.

Our main goal in this research is to identify several types of ungrammaticality, to achieve an understanding of the language phenomena involved, and to develop methodology directed at extending existing technology to treat these
forms in accordance with our understanding of how they work. In many instances, producing a representation for a sentence which highlights those elements which are found to be deviant is sufficient to invoke appropriate further processing by other system components. In this research, we are addressing the problems of word misuse, inflectional deviation, telegraphic and contextual ellipsis, conjunction, and miscellaneous other types of ungrammaticality. Various mechanisms and techniques are proposed for the treatment of such phenomena. Specifically, the contributions of this work are as follows:

1. The exploration of "relaxation" techniques both for tests which specify cooccurrence restrictions in the grammar and for categories which distinguish usage for a particular word sense;

2. A mechanism which permits several cases of conjunction to be handled;

3. Ellipsis-handling techniques that allow the use of context as well as processing resulting from the grammar-writer's own analysis of the forms of ellipsis that are likely to occur;

4. A macro facility for specifying meta-level knowledge for use in parsing;

5. Integration of patterns with traditional parsing to allow a form of "bottom-up driven/top-down verified" mechanism;

6. Several pattern matching algorithms including ones that allow types of inexact matching;

7. Dynamic and automatic patterns which permit ATN grammars to be self-extending;
8. Expectations in the context of traditional parsing which both facilitate a higher degree of context-sensitivity and provide an alternate approach to ellipsis processing from those previously attempted; and

9. Orchestration of all of the techniques in one coherent system.

The next chapter discusses the language phenomena addressed in this work and motivates and introduces our proposed solutions. Chapter III explores cooccurrence restrictions and ways of relaxing them in accepting otherwise unacceptable inputs. Chapter IV investigates how the identification of patterns in analyzing language forms can be useful in developing parsing strategies for abbreviated forms of input. Chapter V discusses how language patterns can arise automatically during parsing and be dynamically used to guide the parsing of some commonly expected language forms. Chapter VI deals with conjunction forms under investigation and describes various approaches to that problem. Chapter VII presents some illustrative examples of how the mechanisms interact to handle specific language problems. Finally, Chapter VIII summarizes the work, outlining questions for further study.
In this chapter, the specific language phenomena that we are investigating are introduced and defined. Motivation for the need to include such phenomena is provided through examples which illustrate the range of sentences considered.

These examples were carefully selected so as to illustrate the type of deviation without seeming unplausible; however, the boundaries of language are not well defined. This is clearly evidenced through studies like those of Ross [ROS79] and Spencer [SPE73]. Ross shows that it is quite easy to find examples of sentences which elicit conflicting judgements as to their grammaticality. Spencer found that such judgements vary significantly between linguists and non-linguists.

Clearly, ungrammatical sentences do occur frequently. Malhotra's study [MAL75] showed that deviant forms were a significant portion of the data collected in simulating a Natural Language management system. Such studies demonstrate the need to investigate these boundary areas of
language, some of which are discussed in this thesis.

Cooccurrence Restrictions

A common form of deviance is word misuse resulting from the violation of cooccurrence restrictions. Problems in this realm include various kinds of disagreement among constituents including number, case, person, mood, voice, or other inflectional differences. For example, in the sentence (* indicates an ill-formed or ungrammatical sentence):

*The old inmate knows that he feel angry
the subject of the embedded sentence (he) disagrees with the verb (feel) in number. A similar disagreement can be noted in the sentence

*Please put the two red block in the box
in which the adjective "two" and its noun "block" again disagree in number. In the sentence

*Socrates am mortal
a disagreement is noted in person between subject and verb.

Many sentences which would be classified as ungrammatical fall into the category of being deviant because of similar disagreements. These sentences, however, are sufficiently close to being grammatical that they are, indeed, understandable. Such sentences, when encountered as
inputs to NLU systems should be treated as less-preferred variations of acceptable sentences.

Chomsky ([CH065], Chapter 4) makes a similar point in claiming that degrees of grammaticality exist. He argues that selectional rules involving lexical features may be violated and an ungrammatical sentence will result. The extent to which the sentence is ungrammatical depends on the level of the feature. The violation of rules involving high-level features will result in sentences that are more difficult to interpret than those resulting from the violation of rules involving features at a lower level.

Cooccurrence violations do occur naturally in dialectic variations as well as in individual variations of language. For example, Shore in [SH077] analyzes fifty-six freshman English papers written by Black college students and reveals patterns of nonstandard usage ranging from uninflected plurals, possessives, and third person singulars to overinflection (use of inappropriate endings.)

Consider the issue of number agreement. Attempting to give a precise statement of such a linguistic rule is nearly impossible, even though it is taken as a rather fundamental idea. Again, there are interesting cases in which matters are less than clear. To illustrate, each of the following examples is correct according to an elementary grammar text,
where the underlined verb is a possible source of disagreement:

One of the children goes to the store

Mathematics is my worst subject

Broadway at 48th Street and the surrounding area is known as Tin Pan Alley

She is one of those girls who are too shy to be popular

The captain, as well as the coaches, was disappointed in the team

Each of these sentences could easily be misstated by even a native speaker of English. Therefore, an investigation of these violations addresses a useful area of language.

Besides the violation of features, cooccurrence restrictions can be violated in another way, namely through the violation of subcategorizational restrictions. Words are categorized according to their usage, but they can be placed in categories which are more specific or less specific. For example, suppose the syntactic word class "ADJ" is divided into the subcategories "ABSOLUTE", "COMPARATIVE", and "SUPERLATIVE". Then the sentence

*Find a best banana in the bunch

is deviant because a SUPERLATIVE would not be allowed following an indefinite determiner.
The reality of these types of errors was noted by Nooteboom in his study of slips of the tongue in which he said, "a mistakenly selected word always or nearly always belongs to the same word class as the intended word". [NOO73] Here, by "word class", he is referring to a very general classification of words, like NOUN, VERB, etc.

These examples, again, point out very clearly Chomsky's notion of grammaticalness and that one may not view the notion of grammaticality in an absolute way. Corroborating evidence also comes from psycholinguistic studies, such as that done by Chapman [CHA74], which have shown that levels of grammaticality are indeed perceived. Linguistic studies also continue to investigate these ideas as documented in [LAK70] and [HAU71].

**Ellipsis**

More extreme forms of ungrammaticality can occur with radical structural differences between the sentence and any expected form. Part of a constituent may be omitted which results in ellipsis. There are several types of ellipsis which are fruitful to investigate.

In considering ellipsis, the most relevant distinction to make, as far as computational linguistics is concerned, is one first made by Gunter [GUN63] between contextual
ellipsis and telegraphic ellipsis.

Contextual ellipsis occurs when a form only makes proper sense in the context of other sentences with the elided element being found in preceding sentences. For example, the form

*President Carter has

seems ungrammatical without the preceding question form

Q: Who has a daughter named Amy?

A: President Carter has.

This type of ellipsis is quite common in response to questions and in compound sentences. In another example, the question

Q: Where is President Carter?

might elicit elliptical answers like

A: at the White House

A: in Vienna

These are expected answers to this type of question. Another example is seen in the sentence

Q: Where is President Carter?

A: in Vienna

Q: and Mrs. Carter?

where the last question is elliptical, but not unexpected.
Quirk, et al. discusses this type of ellipsis as it occurs in dialogue. Three conditions are seen as predicking such occurrences ([QUI72,p.708]):

1. "REPETITION: the second speaker repeats what is said by the first."
2. "EXPANSION: the second speaker adds to what is said by the first."
3. "REPLACEMENT: the second speaker replaces what is said by the first with new material."

These conditions can also occur in various combinations.

For example, if the system were to generate the question

Is Ohio State going to a bowl?

a variety of responses might be possible and, in fact, expected:

Yes, the Rose Bowl (Expansion)
the Rose Bowl (Expansion)
to the Rose Bowl (Expansion)
Yes, they are going to a bowl (Repetition)
Yes, they are going to Pasadena for the Rose Bowl (Expansion & Repetition)

Examples of Replacement are even more common. Any "Wh" question or, for that matter, any question that demands an answer where there exist expectations about the answer may involve Replacement. For example, in the system question:

Who won the Florida primary
any of a variety of responses could be expected, as in Carter
Reagan and Carter
Carter won

The Florida primary was won by Carter for the Democrats and Ford for the Republicans.

These are all cases of Replacement. In every one of these examples, the context provides clues about the form to be expected.

Returning to the case of Black English, Shore notes that a copula may be absent in certain constructs, as in

*If it night...
*Joe sick

This is clearly not contextual and is, therefore, in direct opposition to contextual ellipsis which we have been discussing. Another type of ellipsis, "telegraphic ellipsis", is now discussed.

A large number of elliptical sentences are telegraphic in the sense that what has been left out is commonly shared knowledge or omitted because the form follows a conventional jargon. Instances of this occur when a form only makes proper sense in a particular situation. For example, the forms

3 chairs no waiting (sign in barber shop)
Yanks split (headline in sports section)
average annual income for single-parent families in L.A., please (query submitted to a hypothetical NLU system)

are cases of telegraphic ellipsis with the situation noted in parentheses. In the first example, little sense can be made of it without knowing that the chairs are barber chairs and that there is no waiting required to get a haircut. In the second case, there is a high degree of ambiguity. Only by knowing that this was printed at the top of the sports section of a newspaper can it be resolved. The third form could be an answer to a question, but knowing that it was directed at a system designed to accept queries helps in finding a meaning for the sentence. Such real-world knowledge is required in order to address these potential problems.

In the Malhotra study, cited above, many of the examples analyzed lacked elements which could be considered necessary for grammaticality. In that domain, a typical user might enter

Total sales for March

when, in fact, he wished to see a listing of sales activity for the month of March. Often in his data, some articles and function words were omitted, as in

List profits each item

All of these make sense in the proper context. Some of Malhotra's data will be further discussed in Chapter IV in
light of our solutions to these problems.

Forms of language which are similar to telegraphic forms of language occur in situations where two speakers of different languages must communicate. A pidgin form of language which borrows features from each of the languages develops. Characteristically, it has a limited vocabulary and lacks several grammatical devices (like number and gender, for example) and exhibits a reduced number of redundant features. This phenomenon, we believe, can similarly be observed in some styles of man-machine dialogue. Once the user achieves some success in conversing with the machine, whether the conversation is being conducted in natural language or not, there is a tendency to continue to use those forms and words which were previously handled correctly. The result is a type of pidginization between the machine dialect and the user dialect which exhibits a limited vocabulary, a limited use of some grammatical devices, and other pidgin-like characteristics. Certainly in the Malhotra study, this type of phenomenon was observed. It is therefore reasonable to investigate techniques aimed at dealing with these forms of language. Woods points out that the use of Natural Language

"...does not preclude the introduction of abbreviations and telegraphic shorthands for complex or high frequency concepts -- the ability of natural English to accommodate such abbreviations is one of its strengths."

[W0077, p.18]
Finally, Quirk, et al. [QUI72] examine the phenomenon of ellipsis in detail. They note that which constituent may be elided does indeed often depend on context and that ellipsis can occur in many forms. The following set of examples illustrate their point:

John will play the guitar at the party if Tom will (play the guitar at the party)

I'm happy if you are (happy)

Tom was at Oxford when his brother was (at Oxford)

Bill drinks (sparingly) and (Bill) smokes, sparingly

He walked up (the hill) and (he) ran down, the hill

But, many of these forms of ellipsis involve conjunction and should be treated as such because the missing elements in each case occur totally within one sentence. Also, there is a high degree of parallelism in their structure. Some of these cases will, in fact, be investigated as instances of conjunction, discussed next.

Conjunction

Conjunction is an extremely common and yet poorly understood linguistic phenomenon. Sentences may be conjoined in so many different ways that it seems doubtful whether one could ever specify all possible cases within a single grammar. Moreover, constituents being joined are
often not fully specified, as was seen above, and therefore any serious attempt to address the problem of conjunction must necessarily investigate the related problem of ellipsis. This suggests a different view of the phenomenon than has been taken previously. In this work, conjunction is considered to be an extra-grammatical phenomenon, wherein at least one of the constituents involved obeys the basic rules expressed within the grammar while other constituents are variants of those rules.

With this in mind, consider the various types of conjunction that can be investigated fruitfully from this perspective. Simple forms of conjunction occur most frequently, as in

John loves Mary and Mary loves John.
John loves Mary and hates Sue.
John loves Mary and Sue.

The primary difference in these examples is the level at which the conjunction is active. In the first case, similar sentences are being joined. Verb phrases are joined in the second example and noun phrases are joined in the third. The level becomes important in determining a structural analysis for each of these examples.
Besides simple forms of conjunction in which the joined constituents are well-formed, limited forms of ellipsis can occur along with conjunction. Simple forms of this are best illustrated through sentences which have experienced a phenomenon known as "gapping". For example,

John loves Mary and Mary John  
Birds eat worms and monkeys bananas  
The manager looked shocked, the clerk amazed, and the customer amused

In the first and second examples, the verb is not required in the last half of the sentence; however, there are obvious structural similarities in the two halves which suggest an interpretation. In the last example, several things are illustrated. The elided elements in this example of gapping are copulative verbs which take a predicate nominative in each case. Only the adjectivals, "amazed" and "amused", are given in the latter two conjoined elements. In another example,

The table is made of cak, the large chairs maple, and the small chairs rosewood

the entire verb form "is made of" is not repeated. This shows that, in fact, what is omitted in examples of gapping is not limited to simple forms. Thus, a general pattern for gapping in which elements contained in the initial portion of a conjoined sentence are allowed to be left out of subsequent portions has emerged.
The last two examples above also illustrates a "list" form of conjunction. In this form, several constituents are joined using commas or semi-colons. Other examples of this type of conjunction are sentences like:

John loves Mary, Sue, Nancy, and Bill.

How many did you see; what are they wearing; and which way were they heading?

In cases like these, it is useful to consider the punctuation as a type of conjunction, just as for words. The fact that such a list can contain a large number of elements to be conjoined indicates that the requirements for an adequate theory must be to include some form of repeatable operation.

Another type of conjunction, correlative conjunction, occurs in sentences to coordinate the joining of constituents, as in

John loves both Mary and Bill.
John both loves and hates Sue.
Both Mary and Bill love John.
Neither does John love Mary nor does Mary love John.

It can readily be seen from these examples that such forms of conjunction are again operative on various levels and for various constituent types.
Certainly conjunction is not an unusual or uncommon form. It is a critical element in cohesive text, as Halliday and Hasan [HAL76] argue. We contend that forms of this type can indeed be permitted without a great number of alterations to existing approaches. Several additional types of conjunction are discussed in the final chapter as limitations to the work.

Presented in this chapter were examples of language phenomena often not considered in contemporary NLU systems. We have argued that they can be included since they occur with enough frequency to warrant their inclusion. To omit them would constitute a major impediment to natural communication. Furthermore, as we will illustrate in the ensuing chapters, the inclusion of these forms of language can be achieved without much additional effort on the part of the grammar-writer.
Chapter III

RELAXATION OF COOCCURRENCE RESTRICTIONS

Techniques developed in this chapter allow ungrammatical sentences which violate some cooccurrence relations to be successfully parsed through the systematic relaxation of some test and category restrictions. Test relaxation permits specified tests to be violated during parsing while category relaxation permits parsing to proceed even when a word is encountered which is in a more general category than the grammar specifies. These methods allow the successful traversal of ATN arcs that would not normally be traversed. Arcs forced to succeed in this fashion account for the recognition of a misused word.

The technique applies whenever an arc cannot be traversed during parsing. The "failed" arc is checked to see if one of the relaxation schemes applies. If relaxation can occur, then a backtrack configuration (see Appendix B) is created for the relaxed version of the arc and kept on a
special list of deviant arc alternatives called DEVIENT-ALTS. Only upon complete failure of all alternatives in the normative grammar, including unsuccessful attempts of all possible paths generated through backtracking, will the relaxed arc alternatives be investigated. In this way, the integrity of the original grammar is maintained. Notice that successive relaxation of arcs is possible under this scheme in the case where an arc selected from the DEVIENT-ALTS list cannot be successfully traversed, but is itself relaxable. When this happens, another relaxation alternative will be generated for it and the process repeated. Each level of relaxation causes a deviance note to be generated which marks the final structure of the parse as deviant. These deviance notes are maintained as a record of the acceptability of the sentence in an appropriate form within a structure which accompanies the final parse structure. Two types of relaxation are discussed in this chapter.

Relaxing Tests

Grammars necessarily must distinguish between forms acceptable to the language and those that are not. This is often done through grammatical specifications of form augmented by tests that rule out those forms which the
grammar fails to distinguish. Where simple tests exist within a grammar to filter out unacceptable forms, these tests may be relaxed to allow the acceptance of these forms. Since these tests usually represent linguistic rules, the violation of a test signifies the violation of such a rule. In this manner, those sentences which adhere to the form of the language specified by the grammar without the tests enforced, but not to the language with the tests enforced are marked as deviant. These can only be accepted after some type of relaxation has been permitted.

**Test relaxation** occurs when the test portion of an arc (indicated by `<test>` in Appendix A) contains a predicate function call which has been declared to be relaxable by the grammar writer. A list of the relaxable tests is maintained independent of the grammar. Relaxable tests may be designated as **absolutely violable** in which case the predicate's opposite value (determined by the LISP function **NOT** applied to the predicate) is substituted for the predicate during relaxation. Another designation is that of **conditionally violable**, in which case a substitute predicate is provided. For example, assume that `NUMBER-AGREE` and `PERSON-AGREE` are two absolutely violable predicates representing tests for agreement of subject and verb in number and person respectively. Further assume that the predicate `INTRANS` is conditionally violable with a
substitute predicate of TRANS. This might represent the relaxing of an intransitive verb test to accept a transitive verb. If the test portion of a failed arc contains

\[
(\text{AND}
  (\text{NUMBER-AGREE SUBJ V})
  (\text{PERSON-AGREE SUBJ V})
  (\text{INTRANS V}))
\]

indicating that the arc can only be taken if there is agreement in person and number between subject and verb and the verb is intransitive, then relaxation would involve attempting a substitution for each relaxable predicate until the test succeeds. At the time relaxation is attempted, the modified versions of the test are all tried as candidates for making the arc succeed. Those that do succeed are included in the list of relaxable alternatives. If no manner of substitution will cause the test to succeed, then the test is not relaxable. In the example above, if NUMBER-AGREE and TRANS succeed but PERSON-AGREE fails, then relaxation involves calculating the value of

\[
(\text{NOT (PERSON-AGREE SUBJ V}}))
\]

which gives the LISP value \( \text{T} \) (true) in this case. Subsequent substitution in the test would give

\[
(\text{AND}
  (\text{NUMBER-AGREE SUBJ V})
  \text{T}
  (\text{INTRANS V}))
\]

In this case, one level of relaxation was sufficient to allow the test to be relaxed. The deviance note generated
from this example would consist of the form

(RELAXED-TEST
 (FROM
  (AND
   (NUMBER-AGREE SUBJ V)
   (PERSON-AGREE SUBJ V)
   (INTRANS V)))
(TO
  (AND
   (NUMBER-AGREE SUBJ V)
   T
   (INTRANS V)))))

to indicate that test relaxation occurred with an absolutely violable test and that the second component within an AND was relaxed. Of course the actual form of the deviance note is somewhat arbitrary and could be adjusted to a form better suited to the specific way in which it was going to be utilized by other system components.

When a single level of relaxation is not sufficient to cause the test to succeed and more than one relaxable predicate is contained in the test, then a backtrack CFG is created for each single-level relaxed arc, even though the test portion of the arc is known to fail. Subsequent relaxation will permit multiple substitution of relaxable predicates. This method assures that all possible alternatives are investigated and that the proper deviance notes are generated.
Returning to our example, Suppose both PERSON-AGREE and INTRANS had failed. Then the test relaxation mechanism would determine that a single substitution by itself could not cause the test to succeed. Three backtrack CFGs are generated, each with a different deviance note and each with a slightly different version of the arc. The three test portions of the arcs included for further relaxation would be:

1. (AND
   T
   (PERSON-AGREE SUBJ V)
   (INTRANS V))

2. (AND
   (NUMBER-AGREE SUBJ V)
   T
   (INTRANS V))

3. (AND
   (NUMBER-AGREE SUBJ V)
   (PERSON-AGREE SUBJ V)
   (TRANS V))

A second level of relaxation would permit the substitution of T for PERSON-AGREE in the first alternative, thereby allowing for that test to succeed. Appendix E contains a sample set of relaxable tests specified in a manner acceptable to our implementation.

The idea of relaxing tests during parsing has been explored by Weischedel [WEI77] in his language instruction system. Utilizing an ATN grammar, his system accepts German
sentences which are normally considered grammatical as well as some that are not. The system will instruct the user in how to form good German sentences when poor ones are encountered. The approach taken is to extend the grammar to handle those ungrammatical cases that occur often and to permit the relaxation of grammatical tests like subject-verb agreement and word ordering of adverbs. His approach differs slightly from ours in that he permits tests on individual arcs to be relaxed. In this work, the notion has been modified and incorporated into a general scheme of relaxation.

**Relaxing Categories**

In the last chapter, we discussed Chomsky's concept of degrees of grammaticalness. His ideas have been reconsidered in the context of language processing, resulting in a second type of relaxation. Category relaxation can occur for arcs of type PUSH, CAT, WRD, and MEM. A hierarchy of categories is given independent of the grammar which specifies how the subnetworks, categories, and words in the system relate. The category relaxation mechanism utilizes this hierarchy in determining what categories can be substituted when relaxing one of these arcs. Relaxing a PUSH arc involves PUSHing to a more
general network instead of a specific one*. CAT arcs are relaxed to PUSH arcs which allow the traversal of an appropriate subnetwork or they are relaxed to a more general CAT arc containing categories substituted from the hierarchy. The category relaxation of a WRD or MEM arc involves checking the lexicon for the category or categories of the word or words given in the arc. A relaxed WRD or MEM arc becomes a CAT arc having categories substituted according to the hierarchical specification of categories. Consider, for example, the syntactic category hierarchy for nouns shown in Figure 2. For this example, the category relaxation mechanism would allow, for instance, the relaxation of PERSONAL pronouns to include the categories PRONOUN, REFLEXIVE, DEMONSTRATIVE, and PERSONAL, that is, the entire PRONOUN sub-hierarchy. As in test relaxation, a second level of relaxation for PERSONAL pronouns is permitted which, in this case, would allow any of the categories in the hierarchy of Figure 2. Appendix D contains a sample category hierarchy specified as it would be for use by our implementation of the ATN.

* Note that our implementation, as discussed in Appendix B, does not as of yet allow relaxation to PUSH arcs.
Category relaxation applied at the word level works similarly. For example, the word "yourself" occurring in a WRD or MEM arc would relax to a CAT arc permitting any word in the category REFLEXIVE since the lexical entry for "yourself" places it in that category. Further relaxation could then apply to that CAT arc as described above.

In the example,

Draw me a circle

the normative grammar may be written so that a WRD arc is used to specify the processing of the word "me". Subsequent relaxation of that arc, following Figure 2, would permit any personal pronoun. This would specify a method of processing sentences like (where ? indicates a sentence for which this author's judgement of grammaticality does not yield a clear decision):

?Draw you a circle
?Draw us a circle

A second level of relaxation would then permit forms like:

?Draw yourself a circle
?Draw that a circle

In this case, we can imagine that further levels of relaxation would also be useful to consider in which possibly in the most extreme case, a PUSH arc would be generated which processed an entire noun phrase.
Figure 2

A Syntactic Category Hierarchy for Nouns
Each level of relaxation again generates a deviance note, indicating which categories or words are being relaxed at each level and what the form of the new arc is. The note itself is similar to that shown for test relaxation with the tag RELAXED-CTGY given at the beginning of the note. The length of our deviance notes list roughly corresponds to Chomsky's metric for grammaticalness in the sense that the more deviance mechanisms brought to bear in processing the sentence, the further from grammaticality.

The technique of category relaxation is certainly not limited merely to syntactic categories. Burton's [BUR76] semantic grammar approach to language processing makes exclusive use of categories which distinguish semantic entities rather than the traditional syntactic ones. A category hierarchy can easily be constructed which describes how these categories relate in the system without extending the mechanism as described here. Relaxation will apply similar to the syntactic example. In applying these techniques to semantic grammars, relaxation from PUSH arc to PUSH arc will prove especially useful, since typically there is a myriad of ways to say essentially the same thing.

In summary, this chapter has discussed two types of relaxation that can be performed in the context of ATN grammars to permit input forms not explicitly allowed by the
normative grammar to be associated with an expected form. Relaxation was considered on the word level, on the category level, and on the level of coordinated elements of the grammar in the form of test relaxation. Limitations to the work in this chapter and subsequent chapters will be presented in the summary chapter, Chapter VIII.
Chapter IV

PATTERNS OF LANGUAGE

Language contains regularities. Grammars for major subsets of language are possible only by capturing these regularities. The ATN grammar-writer, for example, takes advantage of this by organizing his grammar into networks and subnetworks which are linked together through the mechanism of recursion. In this chapter, we make the assumption that language deviations occur with enough consistency that the way these deviations are manifested can also be described succinctly. While relaxation applied to words was discussed in the last chapter, this chapter discusses relaxation techniques applied to the grammar itself. In this study, we chose to do this using patterns and pattern-matching algorithms.
Patterns and the Pattern Arc

In our formulation, a **pattern** is simply a linear sequence of ATN arcs which is matched against the input sentence according to one of several matching algorithms. A **pattern arc** (PAT) has been added to the ATN formalism to permit the application of a pattern to the input and the specification of actions to be executed when such a match is successful. The syntax of the pattern arc is simply

(PAT <pat spec> <test> <act>* <term>)

with the pattern specification portion (<pat spec>) defined as

(<patt> <pat parm>)

The pattern (<patt>) is either the name of a pattern or a list of ATN arcs each of which may be preceded by the symbol ">" to indicate that the arc is optional, while the pattern parameters <pat parm> can be any of the keywords, "UNANCHOR", "OPTIONAL", or "SKIP". These are discussed below. See Appendix A for the syntax of patterns, pattern arcs, and related details.

Since patterns are not part of the usual ATN model, it is assumed that whenever patterns are required in the processing of an input, that input is being judged as ungrammatical relative to the norm. A deviance note is therefore generated with every pattern application. The
note itself is again simple, containing the essential information to permit other system components to determine easily the nature of the deviance. The pattern is included along with the words from the input that were matched.

Consider the following simple example of a pattern arc. Suppose the sentence

Mary drove the car

was processed by the ATN path of three arcs, namely PERSON, TRANSACT, and OBJECT which processed Mary, drove, and the car respectively. Then the pattern of the sentence could be specified as

(PERSON TRANS-ACT OBJECT)

This, then, could be included in a pattern arc as

(PAT (PERSON TRANS-ACT OBJECT) T . . . )

It could further be modified to permit optional constituents, as in

(PAT ((PERSON TRANS-ACT OBJECT) OPTIONAL) T . . . )

The manipulation of registers within these arcs poses some interesting problems. With the optionality feature, not every arc will be selected in a pattern. To complicate matters, registers previously set may be used within a test or they may hold elements of the final structure. When an undefined register is involved in a test, the test is defaulted so that undefined registers will not impede
potential parse paths. When a register intended to hold structural elements is undefined, that portion of the structure is marked as undefined and the name of the register which was not set is indicated. A potential problem arises, however, in cases where a sequence of arcs is constantly setting and resetting register values. We argue that in cases like this, the register will contain exactly the correct value, namely the one which results from skipping intermediate arcs.

The inclusion of this new arc type can be motivated in several ways. First, this new arc provides the capability of adding entire constructions or even whole sentence types to the grammar with minimal effort. This permits some existing arc paths to be specified alternately in a more relaxed form. It also provides a clean way to embed a formal language within the grammar to permit abbreviated forms.

Second, patterns can be derived dynamically based on previous (linear) processing paths. Where parallel sentence structure exists, dynamic patterns can capture the similarities and restrain the less fruitful alternatives.

Third, patterns can be computed and used to specify or constrain future processing paths through a mechanism of "expectations." These may arise within the processing of a
sentence, from the previous sentence, or from some other component of the system. More discussion of this mechanism is contained in Chapter V.

The patterns referenced in PAT arcs can be contained in a pattern dictionary and referenced by pattern name. When this dictionary is built, a user-selected pattern name is associated with each pattern. PAT arcs can either contain this name or the pattern itself within the arc, as illustrated above. Examples of pattern arcs and the pattern dictionary are contained in Appendix F.

Pattern matching proceeds by operating the ATN interpreter in a manner which constrains it to the path of arcs given in the pattern. It is further affected by the chosen "mode" of matching. Three parameters can be associated with the pattern being applied in a PAT arc to control the extent to which approximate matching is allowed. These are discussed below.

As mentioned above, an arc in a pattern may be designated as optional by preceding it with the character ">". To activate this feature in the pattern matcher, the parameter OPTIONAL must be given. This allows patterns to be written which specify that certain constituents normally included in such a pattern need not be present for a successful instance of that pattern. Of course the matching
algorithm always attempts to successfully match optional arcs in preference to not matching them.

A pattern unanchoring capability is included in the matcher and is activated by specifying the parameter UNANCHOR. When not in this mode, successful matching must occur beginning with the first word of the string to which the pattern is being applied. Unanchored patterns may successfully match by ignoring words to the left of the string of words matched by the pattern, but even in unanchored mode, preference is given to matching as if it were anchored, if this is possible.

Finally, a SKIP parameter has been included. This allows the ignoring of words between elements of the pattern when an arc within the pattern has failed. This is a more general form of the UNANCHOR parameter and, therefore, specification of both is not necessary. In fact when SKIP is specified, UNANCHOR is assumed to be on.

The notion of a pattern is certainly not unique to this work, although our formulation and integration of patterns with ATNs is unique to the best of our knowledge. Other systems rely on patterns to varying degrees.
Weizenbaum's ELIZA [WEI66] uses patterns triggered by keywords exclusively to break the input string into components used by the system's rewrite rules. Our mechanisms could easily support this type of pattern. Keyword-based patterns could be formulated in our system by constructing patterns out of primarily WRD arcs and by selecting the SKIP mode for matching. This does not, however, prove to be a very effective way to utilize our patterns.

A program called PARRY [COL71] in its most recent form [PAR77] mixes patterns with a noun phrase parser to handle most of the parsing duties. The system is designed to simulate paranoid behavior. In this constrained environment, a series of nine passes over the input is required to transform it into a usable internal form. One of these passes is based on syntactically analyzing noun phrases while most of the remaining passes are pattern-based. Certainly the patterns found in PARRY could be formulated in terms of ATN arc patterns as we have proposed. The advantage would be that parsing could be performed strictly left-to-right without the need for multiple passes. In this way, the patterns would support, through PUSH arcs, a form of "bottom-up driven - top-down verified" processing.
Wilks has developed a theory of "preference semantics" [WIL73] [WIL78] which is implemented largely through the matching of semantic patterns called templates and para-templates. Templates are matched against pre-processed input and successful matching causes the building of preliminary structures. Para-templates are then matched against these structures to transform them into final structures. Our pattern mechanism is certainly general enough to support the specification of templates. Para-templates could be allowed if we first assume that subnetworks of the ATN are constructed to permit the application of templates in the form of patterns. These subnetworks are constructed to build substructures corresponding to those of Wilks' templates. Next, patterns are formed which contain either PUSH arcs to cause the traversal of these subnetworks or patterns of pattern arcs to thereby build the proper structures.

In many ways, pattern-based methods tend to exhibit more robustness in their language-processing capabilities than non-pattern methods. This can be easily seen in contrasting the pattern-based ELIZA program which rarely fails to process any input string and the ATN-based LUNAR system which contains a large, but limited, grammar of a subset of English. ELIZA responds to every input whether of not keywords are present, but achieves no "understanding" of
the input in any sense of the word. The LUNAR grammar, on the other hand, facilitates the building of structures which realize a significant degree of adequacy in its representation of the meaning of the input within the context of lunar geology. One of our goals in including patterns within the ATN formalism is to bridge this gap between robust but shallow language processors and exacting, non-forgiving, representationally more adequate ones.

**Telegraphic Language**

Once patterns have been introduced into the ATN formalism, several language phenomena become more amenable to processing. In particular, when forms are abbreviated either through the elimination of sentence constructs or through a process resembling pidginization, these can often be anticipated and permitted in the grammar via patterns.

A pattern can be seen as capturing a single path through the network. The matcher gives some freedom in how that path relates to a string by supporting arc optionality and skipping.

Generally, for telegraphic sentences, there are elements which have been explicitly omitted. For example, in a sentence from Malhotra's study [MAL75]

?Profit margins for each product
a missing verb must be inferred telegraphically with no contextual cues. Simple cases, such as this one, may be handled straightforwardly by including patterns which permit, for example, optional verb forms. Malhotra's sentence may then be related to a path in the grammar that expects the verb to be present. If the sentence

List profit margins for each product

was permitted in a normative grammar for this system and the path of arcs taken formed the pattern

( IMP-ACT IMP-OBJ )

then making the verb optional would require the pattern

( > IMP-ACT IMP-OBJ )

The actions within the pattern arc which is applying this pattern could determine the proper verb based on objects specified. Obviously, this will prove to be difficult to do in general, but cases that can be anticipated in this manner can be included.

Similarly, some language forms that arise as pidgin-like dialects during an interaction can be included through the use of these techniques. However, it is questionable whether our solution poses a general approach to these issues. A great deal of interaction is required from other components in the system to support the kinds of processing necessary to achieve some success in handling such forms. DeCamp points out that the process of
pidginization "has been often called simplification, but it is now considered debatable whether the less-redundant pidgin is simpler or more complex than the standard language." [DEC71] Our methods offer a solution to simplistic pidgin forms which violate certain agreement relationships, or skip determiners, but which form cohesive structures.

Extraneous Words and Punctuation

If certain patterns are semantically cohesive in themselves, they may, in fact, be matched without regard to certain extraneous elements in the input string. Words of the input may be skipped to achieve this result. In particular, punctuation marks which are not necessary to the interpretation of the sentence are redundant and may, therefore, be skipped during processing without adverse effects.

In an example given by Malhotra, wordiness seems to be the primary problem:

*List prices of single unit prices for 72 and 73

Semantically determined patterns, which are cohesive in the sense that minimally required elements are processed, can be constructed and are useful in achieving a correct processing method for examples of this type.
In this chapter, we have examined several types of linguistic problems centered around the notion of ellipsis and introduced the notion of a pattern to handle problems of this nature. We will continue our discussion of patterns and their usefulness in the next chapter.
Language can only be properly treated in context. Forms considered in isolation will often be judged as ungrammatical, while in the proper context, they become perfectly acceptable. In Halliday and Hasan [HAL76] they promote the consideration of language as text rather than in isolation in discussing the notion of cohesion. The most striking examples of this can be found in question-answer dialogues in which contextual forms of ellipsis are prevalent. For example, the ungrammatical form

*Mary should

becomes completely acceptable in the context of an appropriate question form such as in

Who should be able to chair this meeting?

Mary should.

Such elliptical answers to questions are obvious and predictable, given the context of the question. Ellipsis as a language phenomenon, however, is certainly not limited only to question-answer situations. Consider the sentence
If Bill would not be able to chair the meeting, then Mary should.

in which the same elliptical form appears within one sentence.

This chapter addresses some of the problems relating to context in language. Specifically, in the ATN formalism, grammars are written to statically describe some of the dynamics of the processing required for language. Below, we elaborate on some techniques directed at improving the dynamism of the grammar and enabling the successful use of processing context.

Automatic Production of Patterns

Patterns may be computed automatically. Automatic patterns are derived from previous processing in the network by tracing the execution path taken. This back-tracing is done at the time the PAT arc containing the pattern is invoked and only those arcs found at the same recursive level are included in the pattern. The reason for maintaining this level distinction is to allow these patterns to match only those constituents on the same level. The inherent assumption in this, of course, is that ATN subnetworks are organizational units which may be conjoined, elided, or otherwise used for such pattern derivation.
There are two types of automatic pattern currently implemented. In place of a pattern or pattern name in the first type, the special symbol "\textgreater\textgreater" is used. This indicates to the pattern retrieval mechanism that this pattern is to be automatically constructed based on the state of current processing and when the pattern is built, every arc is to be preceded by the optionality indicator "\textgreater\textgreater". This allows a particular PAT arc to apply the automatic pattern with the OPTIONAL parameter either on or off. The second type of automatic pattern is constructed similar to the first, except the special symbol "\textless\textless" is used and the pattern contains no optionality indicators. This form of automatic pattern permits further user-specified processing to manipulate the pattern in its basic form so that, for example, some arcs can be marked as optional or partial path patterns can be constructed.

**Expectations**

Riesbeck and Schank [RIE76] as well as others have demonstrated the utility of using expectations as a basis for language processing. Experiments with humans discussed in Norman and Rumelhart [NOR75] have verified that various types of predictions are indeed being made during language understanding. Based on this fact and the fact that our
pattern matching facility permits entire paths to be remembered and predicted, we have added an expectation mechanism to the ATN formalism. Using this mechanism, arcs may be designated as viable for future processing by the ATN. Expectations are performed as actions within arcs and therefore no additional arc types are required. An expectation may be generated in one of two ways.

One form of an expectation is

(EXPECT <creat act> <st>)

In this case, the arc is bound to a designated state. Thus, it is known as a bounded expectation. When later processing leads to that state, the expected arc will be attempted as one alternative at that state. If the arc is already part of the grammar at that state, it is placed in the front of the list of arcs so that it will be attempted first. These arcs which have been re-ordered to the front of the arc list are known as preferred arcs for that state. The set of remaining arcs at that state in the grammar are collectively referred to as grammatical arcs. Those expected arcs that are not among the grammatical arcs at a state are known as added arcs.

Another type of expectation has the form

(EXPECT <creat act>)

Here, no state is specified. This is known as an unbounded
expectation and results in placing the arc on a list of arcs to be attempted for every state visited in the remainder of the processing of that sentence. These arcs are considered to be added arcs and are treated similar to these.

ARC PARTITIONING

Through expectations, arcs are partitioned into three arc sets at each state. By including the set of conjunction arcs discussed earlier and a set of alternative arcs which includes any arc added by hand, arcs at every state can be partitioned into five arc sets. In Figure 3, we illustrate the treatment of the various arc sets including preferred, grammatical, added, conjunction, and alternative arcs.

When selecting an arc to traverse from a state, the ATN first selects arcs in the preferred set. If all these fail, it next selects from the grammatical set. All arcs in the original normative grammar are initially placed in the grammatical set of arcs. If no arc can be successfully followed in these two partitions, then ungrammatical possibilities must be investigated. The cross-hatched line in Figure 3 indicates the boundary between grammatical alternatives and ungrammatical ones. Ordinarily, backtracking occurs when no grammatical arcs can be followed
Figure 3

Arc Partitioning
from a state and the remaining alternatives are placed on the DEVIANALT list of possible ungrammatical alternatives, as was done in the relaxation techniques. The ATN interpreter can be told to treat added, conjunction, and alternative arcs as if they were grammatical also and avoid excessive backtracking. This is done by effectively specifying where to draw the cross-hatched line. The arcs in the added, conjunction, and alternative sets are attempted in that order when they are processed.

INTER-SENTENTIAL EXPECTATIONS

The range of an expectation is ordinarily limited to within a single sentence. Expectations are generated within arcs traversed in processing a sentence and unsatisfied expectations disappear afterwards. However, special processing occurs when an expectation is encountered for the start state, S*. This is interpreted to mean that the expectation is intended for the next sentence to be processed. These expectations are held through the end of the current process and transmitted to the procedures that will begin the analysis of the next sentence. This type of expectation may signal that a particular type of answer to a question is expected or that an elliptical input is probable. This type of expectation can only be made effectively to the start state of the ATN grammar used to
process the next sentence. The arc to be expected at that state, however, can contain expectations which then take effect during that processing. In this way, expectations can be constructed which have a ripple effect through the processing of several consecutive sentences.

**Contextual Ellipsis**

The mechanisms outlined in this chapter along with those discussed earlier can be applied to the phenomena of ellipsis. Specifically, patterns may be constructed which describe the processing required to permit elliptical forms. Since the patterns may permit optional constituents, these may be omitted from the sentence and the pattern will still successfully match. Deviance notes are automatically generated for the application of a pattern and it is left to the appropriate system component to further process the resultant structure and construct an interpretation. In addition, appropriate transformations may be permitted on the expected forms so that more natural forms of contextual ellipsis may be permitted.

Returning to the analysis of Quirk, et al. discussed in Chapter II, examples of Repetition, Expansion, and Replacement are seen as following directly from the pattern mechanism applied through expectations. When the system
generates a question such as

Did John Barrymore play Captain Hook in the play?

Several answers could be given without seeming unusual. For example,

John Barrymore played Captain Hook in the play. (Repetition)

John Barrymore played and perfected Captain Hook in the play. (Expansion)

John Barrymore played Peter Pan. (Replacement)

For each of these examples, the system should be receptive to forms of this type. This points out the need for a general mechanism which is flexible enough to permit this range of variation.

This level of generality is supported in the pattern mechanism since expectations are not limited strictly to patterns developed to be similar to the form of the question itself. One possibility is that the answer may be merely a transformation on the original question with the yes or no appended to the front, as in the examples of Repetition above. An optional WRD arc should be included at the beginning of patterns intended for any questions which may be answered in a yes or no fashion.

Expansion forms are permitted quite easily since ATN patterns are commonly organized around PUSH arcs which find constituents by traversing subnetworks. These subnetworks
generally contain sufficient flexibility in the forms that they permit to allow added material to be accepted.

Replacement forms are treated through patterns also with reliance on the Expectation mechanism described above. For the most part, the particular type of answer to a question of this type can be predicted and such predictions can be encapsulated in the appropriate pattern within an expectation.

**Other Work on Ellipsis**

Several investigators have addressed the problem of ellipsis. Each of these takes a slightly different view of how it should be handled. We present a brief description of each work along with a comparison to ours below.

The LIFER system developed at Stanford Research Institute by Gary Hendrix [HEN77], is a facility in which to develop Natural Language interfaces. A user is permitted to access the system as an interface designer and specify the format for inputs expected as well as their semantics. LIFER features an "automatic" ellipsis facility for parsing sentence fragments when they are similar to the previous input. This is done by forming analogy patterns derived from the parse tree of the previous sentence. In being able to offer such a generalized ellipsis capability, the
assumption is made that LIFER grammars will always be semantic in nature insofar as the syntactic categories will always incorporate a great deal of semantic knowledge. Our mechanisms for handling ellipsis are more general than those offered in LIFER. Elliptical forms in LIFER must match some constituent of the previous parse, while our mechanism permits the application of either a pattern computed based on the previous input or some other pattern based on possibly transformations or variations of the previous input. Applications developed in this system include a system called LADDER which allows access to data in a distributed database via Natural Language [HEN78].

Barbara Grosz, also working at S.R.I., has suggested a method of handling contextual issues in dialogues by providing a way of focusing on relevant facts [GRO77]. She also suggests a simple extension to this idea wherein focus is applied at a very local level to deduce the elided element in an elliptical sentence. The approach taken is to recognize that an elliptical sentence has been entered, then try to discover what has been left out. Her approach differs from ours in that we require that a prediction be made where ellipsis is possibly expected. We make the argument that the retrospective approach fails to account for the relative ease with which most examples of ellipsis, in context, can be discerned.
Burton's "semantic grammar" approach to NLU [BUR76] has also been used to investigate ellipsis-handling mechanisms. In his work, semantic categories must be satisfied by the elided elements. Again, a retrospective approach to ellipsis is assumed.

A system called PLANES, which is similar in some respects to LIFER, has been developed by Waltz [WAL78] at the University of Illinois to access a relational database of aircraft flight and maintenance data. It too attempts limited forms of ellipsis. The system is based on the utilization of ATN subnetworks and case frames for language analysis along with special context registers. These registers are utilized to fill in missing components in elided sentences and for pronoun resolution. This system represents an attempt to loosen certain constraints of the grammar by emphasizing subnetworks and their ability to successfully identify components of the parse.

In this chapter, we have shown how patterns may be utilized to constrain future processing within the grammar and also to add processing paths dynamically based on context. The approach is shown to have advantages over that of others.
Chapter VI

CONJUNCTION

This chapter discusses some additional mechanisms designed to handle forms of conjunction. First, we discuss the workings of the mechanisms and then we demonstrate how they are used to deal with various forms of conjunction.

Conjunction poses some difficult problems for a language understanding system. Traditionally, systems were built around the central idea of processing a single sentence, understanding it completely, and then processing another. This process was repeated indefinitely until the text was exhausted or the dialogue was terminated. Only recently have researchers begun incorporating more extra-sentential and extra-linguistic information into the sentence analysis procedure. These studies have provided results which enable a more serious attack on the problem of conjunction.
We view the processing of conjunction as a problem in dynamically adjusting and extending a grammar specification for simple sentences to meet the processing requirements imposed by the more complex input sentence. That is, if a sentence containing conjoined constituents is to be successfully processed, the simple grammar must be capable of repeating the processing necessary for each conjoined constituent. Attempts to incorporate conjoined structures in transformational grammars (for example, see the "gapping" transformation discussed by Ross in [ROS70]), have led to the inclusion of what Bach [BAC74] calls "rule schemata". These are transformations in which the structure index allows for an infinite number of substructures.

In our ATN model, we are proposing a "macro" arc mechanism which permits a static ATN grammar to extend itself dynamically by adding arcs as appropriate. In the case of conjunction, these arcs are usually dynamically computed pattern arcs, based on earlier constituent processing. In addition, such arcs can allow for elided constituents through the pattern arc's optionality feature. Besides being quite common, the occurrence of ellipsis with conjunction, we believe, is the source of many of the problems previous efforts have experienced. Since our solution contains mechanisms aimed at solving problems in both linguistic domains, we can address this problem more
directly. For a thorough analysis of the interaction of these phenomena, the reader is referred to the study of cohesion by Halliday and Hasan [HAL76] as well as the monolithic grammar of contemporary English by Quirk, et al. [QUI72].

Macro Arcs

Several additions have been made to some existing arc types in the ATN formalism. These additions involve only those arcs that normally absorb or parse a word of the input. Included in this list are the CAT, MEM, TST, and WRD arcs. As a result, four new "macro" arcs are now offered, namely, CAT*, MEM*, TST*, and WRD*. These arcs are similar in every way to their counterparts, except that as a final action, instead of indicating the state to which the traversal leads, an arc is computed which is followed in place of the macro arc. This closely resembles the notion of a LISP macro function in which the macro computes a function call for evaluation. The difference in the form of macro arcs can be seen by comparing the following pair where <creat act> is used to define the computed arc:

(CAT  <cat>  <test>  <act>*  <term     >)
(CAT* <cat>  <test>  <act>*  <creat act>)

The complete syntax of arc types is contained in Appendix A.
Arcs computed by macro arcs can be of any type permitted by the ATN, but one of the most useful arcs to compute in this manner is the PAT arc discussed previously. Macro arcs allow pattern arcs to be dynamically selected. Also, PAT arcs allow their patterns to be computed automatically which permits them to use information only available during parsing, like the path of particular arcs followed up to that point or the pairing of correlative conjunctions (e.g., both...and, neither...nor, etc.) Since macro arcs can make extensive use of patterns, part of the discussion that follows will concern itself with special ways in which pattern arcs are used when constructed within macro arcs.

DYNAMIC ARCS

By selecting the name of an arc to be tried or by constructing an arc in its entirety based on computations performed within a macro arc, a dynamic choice of the arc to be used can be made. Consider the following hypothetical macro arc:

```
(WRD* AND T
 (BUILDQ
   (PAT (P23 OPTIONAL) T
     (SETR S *)
     (TO ST*)
   )
 )
)
```
This WRD* macro arc attempts to find the word "AND" in the input. If successful, it computes a PAT arc that applies the pattern P23 with the OPTIONAL switch on. If pattern P23 successfully matches the input string following the word AND, then the S register is set, using SETR, to the list of string components matched by the pattern (indicated by the special "*" register). Next, the state is changed to the state ST* and processing continues. The ATN function BUILDQ is used to construct the arc. This permits register values to be easily substituted into the arc so that parts of the resultant arc may be based on computations performed earlier.

The set of conjunction arcs of this type are globally defined for a given grammar and not associated with any particular state of the grammar. This is a departure from the usual way arcs are included as part of a grammar. During parsing, when every arc in the original grammar has been attempted at a given state, the set of conjunction arcs become candidates for further attempts. The reasons for approaching the problem in this way are:

1. The nature of conjunction is such that it can occur almost anywhere in a given sentence;

2. We believe that it is far easier to specify under what conditions conjunction cannot occur than those under which it can;
3. This approach does not interfere with those cases in which the original grammar handled conjunction; (For example, the construct "between...and..." is often included in grammars that have a reason to expect it; see [BRO79] [BUR76])

4. Most of the globally-defined conjunction arcs are of the word (WRD*), member (MEM*), or category (CAT*) macro arc variety. These all involve comparatively simple tests of the input to determine their applicability and even though several conjunction arcs may be required, selection of the correct arc can be made quickly.

Forms of Conjunction

SIMPLE FORMS OF CONJUNCTION

The first demonstration of the applicability of the mechanisms described above is with some very simple, straightforward cases. For the first example, assume that the sentence

John loves Mary

is correctly processed by the original normative grammar. In so processing, let us assume that three arcs are traversed at the top level, one for each word, and that the arcs traversed are named

SUBJ-NP MAIN-V OBJ-NP

Now suppose the sentence

John loves Mary and Mary loves John

is to be processed by the ATN. The grammar would still
begin processing by traversing the same three arcs as before; however, instead of permitting a POP arc to accept the sentence, more input would need to be processed. At this point, processing would be blocked at a particular state (call it state X) with a path of three arcs followed at that level and with the sentence fragment "and Mary loves John" yet to be processed. The set of conjunction arcs would now be examined to find an arc to follow which is looking for the word AND in this context. In this case, a particular macro arc is selected which creates a pattern arc to process the remainder of the sentence. The pattern portion of that pattern arc is computed based on the path of arcs already followed. In this case, the pattern that will work is simply

( SUBJ-NP MAIN-V OBJ-NP )

This is formulated as a PAT arc containing the appropriate register-appending actions and no state changing action. The result of executing this arc is to end up back at state X having processed the conjoined constituent. At this point in the processing, the final POP arc can be successfully taken and the resultant structure built. In this case, the structure would represent the combining of the two sentences.
We have thus illustrated how some simple cases of conjunction could be handled in a somewhat straightforward manner. Note that conjunction at the topmost, or sentence, level of the grammar can always occur between two rather disparate forms. For example, the sentences

John visited the Statue of Liberty and Mary was mugged in Central Park

Mary was examined by a specialist and the prognosis is poor

show that the strategy outlined above will not work for all cases of conjoining at the top level. In fact, there is little reason to expect such conjoined structures to agree in form at all, except possibly for cohesive factors as outlined in [HAL76]. For this reason, the set of conjunction arcs should always include the possibility of an entire legal sentence following a conjunct at the top level. This can easily be done, of course, through a macro arc which computes a PUSH arc leading to the start state.

Besides considering conjunction as existing only at the sentential level, certainly there are examples at the level of other constituents. For example,

Max relayed the good and the bad news.

Here, the conjunction arc is activated within the processing of a noun phrase when it is only partially completed. The macro arc will construct its pattern based on the partial path at that level.
The very simple types of conjunction are, therefore, handled straightforwardly by specifying conjunction arcs which utilize the pattern-computing facilities within macro arcs. Although the discussion above centered on the treatment of the conjunction "and", similar techniques apply to other forms of conjunction such as "or", "but", "as well as", etc., with the main difference lying in their treatment within the NLU system.

GAPPING

The next set of examples involve the occurrence of conjunction along with ellipsis. In fact, many forms of conjunction can be usefully viewed as cases of ellipsis as well. The interaction of the mechanisms being proposed for each produces a reasonable approach to these problems. One of the simplest examples, to illustrate this approach, is the phenomena known as "gapping".

In gapping, the verb complex of all but one clause in a conjoined sentence is elided. Consider the examples:

John loves Mary and Mary (loves) John
John ordered meat and Bill (ordered) fish
Mary went to New York, Harry (went) to Washington, and Sally (went) to Las Vegas

where the elements in parentheses may be elided. In these examples, the verb complex is only required in the first of
the conjoined constituents.

Considering the first example above, a pattern arc will be produced by a macro arc in a manner similar to that of the analysis in the previous section; however, the second component of the pattern is marked as optional, giving:

( SUBJ-NP > MAIN-V OBJ-NP )

In building the resultant structure, the pattern arc must check if the second arc was in fact traversed. If not, the structure built by the arc must reflect the defaulted verb. In these examples, as in the ones above, the resultant structure would represent the combining of the complete sentences involved.

LISTS

A more extreme form of conjunction occurs when several constituents of a similar type are joined to form a list or series. In most cases, the conjunction itself only occurs between the penultimate and the last constituent, with commas or semi-colons substituting for the omitted conjunctions. Preceding the conjunction is usually a comma or semi-colon, but often this too will be omitted. Consider the following examples:

John loves Mary, Jane, and Bill

I promise to obey the rules; execute the duties of my office; and never take a bribe.
Every man, woman, and child must participate in the cleanup activities

Once again, a case can be made for viewing these sentences as instances of ellipsis wherein the complete sentences are combined where they share common parts.

Once again we examine the first example above and assume an analysis similar to before. Consider the following examples together:

John loves Mary, Jane, and Bill

John loves Mary, likes Jane, and hates Juanita

John loves Mary, Tom loves Jane, and Bill loves Juanita

John loves Mary, Tom Jane, and Bill Juanita

By considering the comma as a form of conjunction, a macro arc can easily pick up the first comma and compute a pattern as before. The most appropriate pattern general enough to handle all three cases above would be:

( > SUBJ-NP > MAIN-V OBJ-NP )

Here, the first and second arcs are marked as optional. When this pattern is applied as part of a pattern arc, it will succeed in all three cases above with matching occurring up to the second comma. At this point, the second comma is picked up as a conjunction and a pattern built, this time based on what matched in the previous pattern. If the third comma is followed by a conjunction, then this is handled by lookahead, so that the comma-and combination is
always treated as a single conjunction.

The structures built as a result of processing each of the above sentences vary depending on what arcs are traversed in each pattern. In some cases, several entire sentences are to be represented almost independently. In other cases, conjunction occurs among constituents serving as sub-components in the final structure. The general answer to this, provided by the technique, lies in the actual arcs traversed within each pattern. This information is simple to ascertain based on the path of arcs traversed which is maintained by the ATN interpreter.

CORRELATIVES

Correlative conjunction can be handled quite easily by again bringing in the notion of expectation. Consider the following examples:

John loves both Mary and Jane
John both loves Mary and hates Juanita
Both John and Mary run to the store

In all of these examples, the initial correlative (both) signals the beginning of a conjoining between two elements and the later correlative (and) delimits the first element. Processing of the first of the pair is performed by a conjunction arc which need not be a macro arc. It can, more
simply, be a CAT, WRD, or MEM arc which contains as one of its actions an expectation of a pattern arc which contains the pattern computed based on the path followed back to the initial correlative. Within this pattern, the number of variations appear to be slight and usually a strictly applied pattern suffices. A more detailed trace of the actual processing is contained in Chapter 7.

MULTIPLE CONJUNCTIONS

In this section, we examine several examples in which more than one conjunction occurs. It is reasonably common to have several conjunctions in the same sentence. For example, in the sentences

John and Bill love Mary and Juanita

John will go to the store and buy food either this morning or this afternoon

John and Mary or Bill and Juanita will win the contest

John and Mary, Tom and Jane, and Bill and Juanita live together

various combinations of conjunctions occur. In the current implementation, many of these types of examples can be handled without any problems; however, some are not handled because of implementation limitations. Primarily, these center around the manner in which patterns in pattern arcs are matched. Once a pattern succeeds, none of the
alternative ways of matching are ever attempted. That is, the backtracking mechanism does not allow backtracking to within the middle of a pattern. If this limitation were removed, then the various alternative parsings become possible. Specifically, the patterns are formed around constituent boundaries and as long as the multiple forms of conjunction are consistent with those boundaries, processing progresses smoothly.

Other Work on Conjunction

Other researchers have investigated the problem of conjunction within an NLU system. The attempt to handle conjunction which is closest to ours is the SYSCONJ facility in the LUNAR system (see [WO073a] for a description and the appendixes of [WO072] for more details). This facility is an attempt to automatically permit the occurrence of conjunction in a grammar that otherwise doesn't include it. In this approach, when a conjunction occurs in a sentence, normal processing is suspended and the entire processing path is examined for reasonable places in previous processing at which to attempt an analogous type of processing for what follows the conjunction. This is performed non-deterministically. When the processing succeeds in finding a similar constituent, the appropriate constituents are automatically conjoined and the suspended
processing is allowed to continue.

Our approach compares closely to SYSCONJ, but with a few important differences. First, there is no provision for elided constituents in SYSCONJ as there is in our approach. If the conjoined constituents don't precisely parallel each other in the processing they evoke in the grammar, then SYSCONJ fails. Secondly, our approach doesn't cause the complete disruption of processing that occurs in SYSCONJ. In SYSCONJ, the current processing stops and the parser is called to operate under the constraints imposed by the restart configuration. The entire parsing path is utilized in constraining the processing. In the macro arc approach, pattern arcs are usually constructed to achieve a similar effect, but without this additional, very specialized mechanism. Finally, SYSCONJ automatically builds the conjoined structure from the constituents found, while our approach permits the grammar-writer to specify the building of such structures. The SYSCONJ approach permits forms of cataphoric ellipsis as in the example from Woods [W0073a]

John drove the car through and broke the plate glass window.

Our approach, although limited in some respects, is more flexible and more easily transferred to other ATN grammars.
Conjunction is also treated in Winograd's SHRDLU system [WIN72]. His approach is to define special programs for each conjunction which are invoked whenever one is encountered. These programs can, in turn, execute PROGRAMMAR programs to perform further parsing. A program for "AND", for example, looks to process something similar to what just had been processed. Conjoined constituents are later processed into pieces of PLANNER programs as appropriate.

In comparing our approach to Winograd's, all of the differences mentioned above in connection with SYSCONJ also apply. In addition, our treatment of conjunction is uniform for all conjunctions and doesn't require completely separate programs for each. Our approach attempts to capitalize on the similarity in the mechanisms at work in conjunction.

Additional work on conjunction has been performed by others. Petrick, Postal, and Rosenbaum [PET69] have examined coordination reduction within a transformational grammar framework. Some ideas on how conjunction could be included in a natural language program based on a transformational approach are explored. Biermann and Ballard [BIE78] have built on Winograd's concepts of conjunction handling to include a facility (called "mix") in a natural language computation capability.
In summary, the most powerful feature of the macro arc and pattern arc approach to conjunction is the ability to allow elliptical processing in treating conjunctions. Other approaches have been limited by not including such a capability. Our approach smoothly integrates this feature into a reasonably general conjunction processing mechanism.
Chapter VII

EXAMPLES FROM AN IMPLEMENTATION

In this chapter, we present some of the details of the implementation and coordination of the ideas developed earlier. First, we give a sample dialogue from our prototype implementation and then we trace two examples in some detail. The reader is referred to Appendix B for an overview of ATNs and to [KWA74] for a more detailed description of our implementation.

Sample Dialogue

To illustrate the potential of the techniques proposed, a grammar for a short sample dialogue has been implemented using a prototype Natural Language Graphics (NLG) system. (See [BR077] and [BR079] for further details.) The techniques, we argue, will apply equally well in any context. The NLG system was merely chosen for its availability.
The prototype graphics system was implemented as an experimental facility for drawing simple pictures on a computer graphics terminal using Natural Language input. It was developed to draw lines, points, and circles and to answer simple questions posed by the user. The input may be supplemented with touches to the screen to provide positional information.

In the dialogue below, user inputs are preceded by a question mark (?) and shown in capital letters. System responses are also shown in capital letters, while the commentary is not. Actual drawings and erasures are not shown. The symbols "<T>" indicate a touch to the screen.

?A SMALL CIRCLE AT <T>, PLEASE.

Most NLU systems accept only complete sentences, often using the verb to determine the basic framework of the structure to be built. Here, we illustrate that meaningful utterances need not contain a verb. Nonetheless, within our restricted domain of Natural Language Graphics, a drawing action is inferred. After drawing, the system responds

OK

?MAKE A LINE WITH FROM <T> TO <T>.
In this input, a deviant syntactic form is noted. Most systems would balk in attempting to parse it. Although unusual in form, a reasonable interpretation can be made by allowing a pattern which approximates the path to be applied with the SKIP option selected. In such cases of minor deviance where a reasonable interpretation can be found through semantics, we believe that the system should permit slight variations from the normative grammar. Patterns for this purpose can easily be constructed based on an analysis of the paths in the original grammar. In this case, the sentence is recorded as deviant, allowing user-initiated corrections to be made as necessary to clear up any misunderstandings. The system, of course, should be receptive to any changes requested of deviant inputs.

OK

DRAW A SMALL CIRCLES AT <T>.

Here, the plural form of the noun CIRCLE is given along with the singular determiner "A". This is a case of test relaxation in which the agreement of noun and modifier is violated. A singular interpretation is given to the sentence, the deviance is noted, and a single small circle is drawn.

OK
DRAW A CIRCLE TO RADIUS FOUR UNITS HERE <T>.

The word "TO" used in this context is somewhat unusual. The original grammar would only correctly process this sentence if the word "TO" were changed to "OF". The categorization of prepositions in this system divided them into the subcategories PREPC (for prepositions which often take a compound form, like "between"), PREPF (for prepositions whose phrases closely modify noun phrases, like "of"), and PREP (for all other prepositions). The original grammar requires a PREPF category for correct understanding. Category relaxation applied to that arc removes the block, allowing a processing path to be found in the grammar. This example is discussed in more detail later in this chapter.

OK

?LINE <T> <T>.

This form again illustrates a verbless form. In this case, only a minimum specification is given. We can compare this to forms found in formal command languages or "pidgin" forms that occur naturally from languages in contact. This is an example of a "pidgin-like" form and was discussed in Chapter IV.

OK
DRAW A LINE FROM <T>.

The drawing of a line requires the specification of two endpoints or an endpoint, length, and angle, etc. A partial structure can be built for this sentence with an indication of deviance. The system can then elect to handle the sentence in a number of different ways. A default endpoint may be chosen or the user may be asked to supply one, as below:

TO WHERE?

?TO <T>.

A sentence fragment such as this is really quite natural since the question asked by the system is answered most directly in this manner. The behavior of the language processor in an NLU system can often be influenced by factors outside of the processing itself. In this case, we see where an answer to a given type of question is expected. So-called "Wh" questions (i.e., who, what, when, where, etc.) frequently lend themselves to such elliptical answers as the discussion of Chapter II suggested. Expectations arise in the language generation component which affect the processing of the next input. Our expectation mechanism is general enough to permit such interaction.

OK
CALL THIS POINT P1 <T> AND THIS ONE P2 <T>.

Here is a case of conjunction wherein two naming actions are joined. A complication is introduced through the use of the pronoun ONE. Our original NLG system did not attempt to handle conjunction or pronouns. The conjunction is handled through the dynamic conjunction facility discussed in Chapter VI, while the pronoun referent is found through category relaxation discussed in Chapter III.

OK

?NOW CONNECT P1 AND P2 AND CALL THE LINE LP1P2.

Conjunction occurs here also, but there are two conjunctions in the sentence. This causes no problem though, since the verb CONNECT allows the first AND as part of a legal syntactic construct. Our techniques for handling conjunction only become active when unexpected forms are encountered. In this case, that occurs when the second AND is parsed.

OK

?CONNECT <T> AND <T> AND <T> AND <T>.

As in the previous sentence, multiple conjunction can be handled even when some of them are expected by the grammar. Here, two lines are drawn. The sentence can be
construed to have other meanings in a more complex type of environment, but in our simplistic system, we have used our techniques to construct the most logical meaning.

OK

?DRAW A CIRCLE WITH A 2 INCH DIAMETER AT P2.

OK


Some forms of ellipsis only make sense in context. Here, a sentence beginning with a conjunction makes perfect sense following the previous sentence. This is handled directly through the same mechanism that handles other forms of conjunction. Since the conjunction occurs first, a macro arc is activated immediately. An automatic pattern is constructed from the previous input and applied to the remainder of the sentence.

OK

?DRAW BOTH A SMALL VERTICAL LINE AND A SMALL HORIZONTAL LINE THROUGH (11, 11).

Conjunction can occur within a sentence wherein the conjoined pieces indicate two actions with shared constituents. The use of correlative conjunction causes no problem since the conjunction mechanism can detect the word BOTH and establish an expectation for the subsequent word
AND to form parallel structures. In this case, a macro WRD* arc is constructed dynamically which looks for the word "AND" in order to coordinate with the word "BOTH". Here, two lines are to be drawn which criss-cross at the point (11,11). This sentence is examined in detail later.

OK

?MAKE LINES FROM P1 TO <T>, <T>, <T>, AND <T>.

Another common form of conjunction is the "list" form in which several items are conjoined using commas. These forms are treated straightforwardly by treating the comma as a just another form of conjunction.

OK

?PLEASE STOP

LEAVING NLG - THANK YOU.

Example of Cooccurrence Relaxation

In this section, we closely examine an example which requires the use of the relaxation mechanisms. The sentence

DRAW A CIRCLE TO RADIUS FOUR UNITS HERE <T>

is discussed in relation to its processing by some of the networks in the NLG grammar. The sentence-level network
Figure 4

NLG Sentence-Level Network
from the NLG grammar is given in Figure 4, while the noun phrase network is presented in Figure 5. The entire grammar is given in a more complete form in Appendix C.

The processing of this sentence in the network begins at the state 5 by convention. The first word of the sentence is "DRAW" which matches none of the arcs expected at the initial state. This causes the JUMP arc to be taken to state S IMP where a CAT V arc may be traversed, thus bringing the parse to state VP HEAD. Here, several arcs are attempted, but they all fail except the PUSH NP arc. At this point in the parse, the word DRAW has been processed and we have yet to process the remainder of the sentence.

The state NP is entered as a result of the PUSH s that traversal of that subnetwork is required for a noun phrase to be found. Since a determiner (category DET) is the next word in the sentence, the word "A" is parsed, leaving the computation in state NP DET. The parse proceeds on to states NP ORD, NP NUM, and NP CLASSF. Once in the latter state, the head noun of the phrase, CIRCLE, is finally parsed. This activity now leaves the parse in state NP NP. At this point, one of the alternatives being considered is to cycle back to state NP ORD if a word in the category PREPF is discovered. This is precisely where the category relaxation enters the process. The next word in the input
Figure 5

NLG Noun Phrase Network
is actually "TO" which is in the general category PREP. According to the category hierarchy given in Appendix D, the category PREPF may be relaxed to include the categories PREP, PREPC, and PREPF. A CAT arc expecting this list of categories is generated from the original CAT arc and included as a deviant alternative to be attempted after all valid parse paths have been exhausted.

Attempts to extend the parse beyond the state NP NP fail and eventually all untried grammatical paths have been attempted with the same result.

The deviant alternatives are now investigated further. One of the alternatives is selected to attempt to extend further in the grammar. Eventually, as each of these alternatives are tried and discarded one by one, the selection criteria selects the alternative produced by relaxing the prepositional categories. This alternative leads back to state NP ORD from which a JUMP is performed to NP NUM. Another JUMP takes the processing to the state CLASSF which parses the word "RADIUS" and moves the processing forward to once again be in state NP NP. The arc TST NUM is now taken to permit the parsing of the word "FOUR". The word "UNITS" is parsed next after JUMPing to state NP NP. Following a JUMP to state NP NP1, a POP is performed taking the parse back to complete the PUSH begun.
from state VP HEAD.

The PUSH arc is completed and moves the parser forward to the state VP. From here, a simple PUSH to the TOUCH subnetwork is all that is required to complete the parse.

Thus, allowing one simple violation of the agreement between the category of the word encountered while parsing and the category of the word expected permitted the interpretation of the deviant sentence.

Example of Correlative Conjunction

In this section, a trace of the complete processing of another sentence is discussed. For this example, an actual trace of the processing is shown which has been annotated to provide a more comprehensible narrative. In order to follow this discussion, the reader may wish to refer to the NLG grammar given in diagram form in Appendix C. The sentence DRAW BOTH A SMALL VERTICAL LINE AND A SMALL HORIZONTAL LINE THROUGH (11 . 11).

from the dialogue given earlier is traced in its entirety. This sentence was chosen since it exercises most of the more complicated features of the proposed mechanisms. The sentence is an example of the use of correlative conjunction within a noun phrase. The conjunction mechanisms are utilized to parse the initial correlative. This leads to an
expectation in the form of a WRD* macro arc which looks for the matching conjunction. When that expectation is met, a PAT arc containing an automatic pattern is generated to parse what follows the conjunction.

The trace begins after the user has entered the sentence and it has been formed into a LISP list structure which is passed on to the ATN parser. Some portions of the trace which are unimportant to the discussion or redundant have been edited out. The message STRING= in the trace contains that portion of the input that has yet to be parsed. The ATN processing begins from state S* in the grammar.

STRING= --> (DRAW BOTH A SMALL Vertical LINE AND A SMALL HORIZONTAL LINE THROUGH (11 . 11) .) <--
JUMPing to --> S*IMP <-- on JUMP arc
Traverse with --> DRAW <-- on CAT arc

STRING= --> (BOTH A SMALL Vertical LINE AND A SMALL HORIZONTAL LINE THROUGH (11 . 11) .) <--
About to PUSH to --> NP* <-- from state VP*HEAD
JUMPing to --> NP*DET <-- on JUMP arc
JUMPing to --> NP*ORD <-- on JUMP arc
JUMPing to --> NP*NUM <-- on JUMP arc
JUMPing to --> NP*CLASSF <-- on JUMP arc
Blocked at --> NP*CLASSF <--

Parsing is blocked at state NP*CLASSF since there are no arcs that can be successfully traversed from that state. Backtracking is attempted.

Trying alternative at --> VP*HEAD <--
Cannot POP from --> VP*HEAD <--
Blocked at --> VP*HEAD <--

Backtracking leads back to state VP*HEAD which still
contains an untried POP arc. This arc is attempted and fails. Backtracking to state S* is attempted next.

Trying alternative at --> S* <--
About to PUSH to --> TOUCH <-- from state S*
JUMPing to --> TOUCH1 <-- on JUMP arc
Blocked at --> TOUCH1 <--

The parser is again blocked, but this time no conventional backtrack alternatives exist. Standard ATN parsing would quit at this point indicating failure. In the new ATN, conjunction arcs are yet to be attempted. With this run, four conjunction arcs are present and each must be attempted for each state encountered by the grammar thus far. The selection of which state to begin attempting conjunction is determined by a heuristic selection function. Currently, selection is based on the depth of penetration into the grammar (i.e., the path length). For most cases, this choice works well, but for this sentence, a great deal of backtracking is performed within the NP* network since in the original grammar the deepest point achieved in the grammar resulted when an unsuccessful PUSH was performed to that network. Several JUMP arcs were followed then, looking for a valid noun phrase and several states were investigated. Each of these states could legally find a conjunction. The first state tried is the state NP*CLASSF which was visited earlier and whose path length in the grammar is 7.

DEVIANT arcs --> ((WRD*-1-2 AND T) (WRD*-2-2 AND T) (CA T-1-0) (MEM*-1-0)) <--
An EXPECTation is generated when this CAT arc is followed. Correlative conjunctions require that there be a matching conjunction later in the sentence. Here, that matching word is stored as part of the lexical entry for the word BOTH. Notice also that a macro arc was not required to handle this form of conjunction (as with the typical instance of a conjunction) since the initial correlative only signals later processing. This later processing does, however, require the power of the macro arc, as will be seen shortly.

Part of the definition of the word BOTH indicates that it may be paired with the word AND to form a correlative conjunction (i.e., the result of (GETF CONJ) is AND). Subsequently, the WRD* macro arc, that is constructed dynamically, looks for this word. No state was specified, so the dummy state *ANY-STATE* is defaulted and every state from this point on will allow the added arc to be attempted.

At this point, all conjunction arcs have been attempted, but
the EXPECTed arc must still be tried. The attempt to follow this arc fails and more backtracking is required.

DEVIAN'T arcs \[\rightarrow ((\text{WRD}^* \text{ AND T} \ (\text{BUILDQ} \ (\text{PAT-2-1 T})))) \leftarrow\]

with weight \[\rightarrow 8 \leftarrow\]
Blocked at \[\rightarrow \text{NP}^*\text{CLASSF} \leftarrow\]

Several similar attempts to follow conjunction arcs are now made from states within the NP\(^*\) network, including NP\(^*\text{NUM},\) NP\(^*\text{ORD},\) and NP\(^*\text{DET}.\) All of these are unsuccessful. These attempts have been omitted from the trace since they all experience essentially the same fate as the backtrack attempt above. We next see the trace of conjunction arcs being attempted from state NP\(^*\) which then leads to successful parsing of the noun phrases. This state was on a path resulting from traversing three arcs.

DEVIAN'T arcs \[\rightarrow ((\text{WRD}^*-1-2 \text{ AND T}) \ (\text{WRD}^*-2-2 \text{ AND T}) \ (\text{CAT T-1-0}) \ (\text{MEM}^*-1-0)) \leftarrow\]

with weight \[\rightarrow 3 \leftarrow\]
Calling arc \[\rightarrow \text{WRD}^*-1-2 \leftarrow\]
Calling arc \[\rightarrow \text{WRD}^*-2-2 \leftarrow\]
Calling arc \[\rightarrow \text{CAT}-1-0 \leftarrow\]
Traverse with \[\rightarrow \text{BOTH} \leftarrow\] on CAT CORRCONJ arc
EXPECTing \[\rightarrow (\text{BUILDQ} \ (\text{WRD}^*-1-2 \ # \ T) \ (\text{GETF CONJ})) \leftarrow\]
Adding arc \[\rightarrow (\text{WRD}^*-1-2 \text{ AND T}) \leftarrow\]
at state \[\rightarrow *\text{ANY-STATE} \leftarrow\]
STRING= \[\rightarrow (\text{A SMALL VERTICAL LINE AND A SMALL HORIZONTAL LINE THROUGH (11 \ . \ 11) .}) \leftarrow\]
Traverse with \[\rightarrow \text{A} \leftarrow\] on CAT DET arc
STRING= \[\rightarrow (\text{SMALL VERTICAL LINE AND A SMALL HORIZONTAL LINE THROUGH (11 \ . \ 11) .}) \leftarrow\]
JUMPing to \[\rightarrow \text{NP}^*\text{ORD} \leftarrow\] on JUMP arc
JUMPing to \[\rightarrow \text{NP}^*\text{NUM} \leftarrow\] on JUMP arc
Traverse with \[\rightarrow 64 \leftarrow\] on CAT ADJ arc

The system default for the word SMALL is 64 screen units.

This is determined to be the root form of the word from
consulting the lexicon and used when the trace messages are printed. Similarly, the word VERTICAL defaults to the root form 90 degrees as seen below.

\[
\text{STRING} = \rightarrow (\text{VERTICAL LINE AND A SMALL HORIZONTAL LINE THROUGH (11 . 11) .}) \leftarrow
\]
Traverse with \( \rightarrow 90 \leftarrow \) on CAT ADJ arc

\[
\text{STRING} = \rightarrow (\text{LINE AND A SMALL HORIZONTAL LINE THROUGH (11 . 11) .}) \leftarrow
\]
Traverse with \( \rightarrow \) ST-LINE \( \leftarrow \) on CAT (N NPR) arc

The root form of LINE is the system entity ST-LINE. This is the head noun of the phrase. The processing of the noun phrase is complete with the parsing of this word.

\[
\text{STRING} = \rightarrow (\text{AND A SMALL HORIZONTAL LINE THROUGH (11 . 11) .}) \leftarrow
\]
JUMPing to \( \rightarrow \) NP*NPl \( \leftarrow \) on JUMP arc

POPPing from \( \rightarrow \) NP*NPl \( \leftarrow \)
on arc \( \rightarrow \) (POP (GETR NP) T) \( \leftarrow \)
POP completed to state \( \rightarrow \) VP*HEAD \( \leftarrow \)
* \( \rightarrow \) ((DET A) (ADJ 64 90) (N ST-LINE)) \( \leftarrow \)
Resuming execution at level \( \rightarrow 1 \) \( \leftarrow \)

At this point in the parse, a POP has been executed from the NP* network which answers the PUSH attempted at the very beginning of the trace. Note that the EXPECTed arc is still active, although there are several "grammatical" arc possibilities to be attempted first. (Note that we use the term grammatical in quotes, since the parse has already experienced an ungrammaticality.) These are eliminated one by one and we again pick up the trace with the EXPECTed arc being executed from state VP*. Although this arc is marked as deviant in the trace, it is really an ADDED arc in our terminology.
DEVIANT arcs —> ((WRD* AND T (BUILDQ (PAT-2-1 T)))) <-

with weight —> 12 <-
Taking MEM*ber or WRD*: —> (AND) <-
Calling arc —> PAT-2-1 <-
Attempting PAT arc —> (PAT (> ANCHOR OPTIONAL) T) <-

The EXPECTed arc was a macro arc for processing the conjunction that corresponded to the correlative found earlier. Here we see that macro WRD* arc executed and expanded into a PATtern arc. This arc in turn requires the automatic derivation of its pattern based on previous processing. We now follow the matching of this pattern.

About to PUSH to —> NP* <-- from state VP*
Traverse with —> A <-- on CAT DET arc
JUMPing to —> NP*ORD <-- on JUMP arc
JUMPing to —> NP*NUM <-- on JUMP arc
Traverse with —> 64 <-- on CAT ADJ arc
Traverse with —> 0 <-- on CAT ADJ arc
Traverse with —> ST-LINE <-- on CAT (N NPR) arc
JUMPing to —> NP*NPl <-- on JUMP arc
POPping from —> NP*NPl <--
on arc —> (POP (GETR NP) T) <-
POP completed to state —> VP* <-
* —> ((DET A) (ADJ 64 0) (N ST-LINE)) <-
Resuming execution at level —> 1 <-
Successful PATtern match with * —> (A 64 0 ST-LINE) <-

At this point, the pattern has successfully matched. The trace shows what portion of the input was matched by exhibiting a list of successive "*" register values. Next in the trace is the processing of the final prepositional phrase showing the position of the drawing.

STRING= —> (THROUGH (11 . 11) .) <-
About to PUSH to —> PP* <-- from state VP*
Traverse with —> THROUGH <-- on CAT PREP arc
STRING= —> ((11 . 11) .) <-
About to PUSH to —> NP* <-- from state PP*P
JUMPing to —> NP*DET <-- on JUMP arc
The parse is finally completed. For the NLG system, a case structure is built which reflects the command entered by the user. Note that two orientations are given (the case %Orient) and two sizes are also specified (the case %Size). A single object is given (in the case %Object) and the position case is filled with a LOCATOR (i.e., a center point) designated at the point (11,11).

In summary, this chapter has discussed the details of blending the various mechanisms together into one unified, workable system. A sample dialogue with comments about the
nature of the processing required for each sentence was presented. Next, a detailed discussion of two examples, taken from the dialogue, was presented to illustrate the complexity of the coordination problem and to provide the reader with a deeper understanding of the applicability of the techniques to the problems described earlier.
Chapter VIII

SUMMARY

To conclude this work, we present a summary of the accomplishments and contributions of the work. We then identify and discuss some of the problems encountered which aren't solved by the mechanisms presented. Finally, we outline that work which is yet to be done and provide some speculation, based on this work, as to how those future problems can be realistically addressed.

Accomplishments

The original goals of this work, as outlined in the first chapter and reiterated throughout the work, were centered around enhancing the ability of a given Natural Language Understanding component so that, in particular, ill-formed sentences could be accepted by relating them in regular ways to well-formed ones. In so doing, the ungrammatical sentences would be marked as such. Several
techniques were proposed herein.

Techniques for relaxation within the grammar were explored. These techniques involved allowing certain tests which were dictated by the grammar to be overridden. That is, at the discretion of the grammar-writer, a collection of tests required by the grammar to adequately restrict and interpret the input string are permitted to fail during the processing of the input. In some cases, the tests were permitted to fail absolutely and the processing allowed to continue, while in other cases when the test failed an alternate test was performed in its place.

A special case of this latter type of test failure was discussed under a separate section on category relaxation. In this case, the test being relaxed is one performed internally in the ATN to check the validity of the category of the input word. When this type of test fails, a category hierarchy, given by the grammar-writer, is consulted so that a list of categories, usually more general than the one required by the grammar, is used in its place.

Relaxation techniques were applied to problems of cooccurrence violation. These techniques permit the extension of a given grammar to include forms which would otherwise be rejected because of cooccurrence restrictions. It was found that many of these problems could be handled
uniformly through these techniques.

Another part of this investigation centered around the processing of various types of conjunction. Again, the basic approach was to assume that all well-formed inputs were specified by a grammar and the task was to relate the conjoined input string to one or more well-formed strings. Our proposal for how to do that centers around the extensive use of macro arcs in the ATN. These arcs are permitted to compute the arc or arcs appropriate for a given situation. They have available to them a specially constructed mechanism for forming a pattern arc dynamically based on previous processing. A special set of macro arcs is assembled by the grammar-writer for use at any state. The idea behind this approach is that we believe that one universally-defined assemblage of arcs, if formulated in a robust manner, is sufficient to handle all cases of conjunction we have discussed.

Patterns and the pattern mechanism, in their own right, are shown to be invaluable in allowing the specification of arc paths. Used in this way, they aid in permitting certain restrictive portions of the normative grammar to be less restrictive without jeopardizing the structural characteristics of the grammar.
We have shown how this can be applied conceptually to many situations in which elliptical sentences have been encountered or anticipated, as well as for sentences which follow a telegraphic or pidgin-like form relative to the normative grammar.

Problematic Cases

Although the mechanisms and techniques proposed in this work are contributions toward achieving the goal of robust language understanding, some problems have been identified which illustrate shortcomings and serve to verify that, indeed, there is additional work to be accomplished.

Within the ATN interpretation mechanism, the grammar is followed from state to state with alternatives suitable for backtracking being remembered for later processing, if required. In the same way, additional alternatives are generated by the ungrammatical processing mechanisms discussed in previous chapters. With the standard backtrack alternatives, it is clear that a depth-first traversal of the network is required, under the assumption that successful arc traversals will generally lead to more successful arc traversals. From this, the order in which to attempt alternatives is clear.
For ungrammatical processing alternatives, the situation is not as clear. In the first place, these are attempted only after all other alternatives have been exhausted. In the second place, how can it be decided whether one alternative should be favored over another? There are clear cases of number agreement, for example, where the subject is in error in one case and where, in an analogous case, the verb is in error. For example,

*John love Mary.

*Ant love sugar.

In the first case, the ungrammaticality is due to an incorrect verb form, while in the latter case, the problem is with the subject.

Developing heuristics which help to decide which alternative to pursue in cases like these is a major problem. In our initial experimentation with these mechanisms, we have relied on the heuristic originally used by Weischedel [WEI77] which was to always choose the alternative with the longest path length (i.e., the one with the deepest penetration in the network). As can be seen from the examples above, this will not always work. It may often lead to time consuming extraneous processing and, at worst, it may lead to an incorrectly built structure. The answer lies in being able to examine the semantics of each partial structure and decide which one is most plausible.
Our mechanisms occasionally involve extreme amounts of time in that all grammatical paths must be investigated before ungrammatical choices are permitted. This organization may be tempered by adjusting operational parameters in our implementation so that at the moment a block occurs, some or all of the ungrammatical alternatives may be investigated. This is suggested, in particular, for conjunction arcs since they generally involve simple tests which may be examined quickly and which lead to plausible paths when successful.

Another difficulty which may be felt in working with these mechanisms is the task of interfacing the techniques suggested with the original normative grammar. It is often difficult, without some external experimental data, to decide which paths are best suited to permit some form of relaxation. In addition, it is equally difficult to decide which functions and categories are best relaxed without impacting the system adversely. These are problems which fall into the realm of experience. It is felt that as the techniques are applied to unique situations, they will be utilized with ever increasing skill.

Limitations exist to the test relaxation methods described in Chapter III. In permitting relaxation of this type, we are assuming that the relaxable tests can be
adequately specified by merely naming the function which is to be relaxed. It is often the case that the same function is being used differently depending on context. For absolutely violable tests, the assumption being made is that these tests are being used essentially as predicates which are expected to either succeed or fail. A common situation in writing an ATN grammar is for the function GETR to be used in retrieving the value of a flag. In such cases, GETR acts like a predicate taking on the value of either T or NIL, but in the more general case it does not. A more general scheme (which we have not chosen to implement, but which should be a straightforward extension) is to permit the specification of some or all of the arguments to the relaxable function when describing the form of relaxable function calls. This could be accomplished rather easily and would overcome these weaknesses.

Lest the reader be left with the impression that all conceivable forms of conjunction can be handled by these techniques, some limitations to the methods explored in Chapter VI are now discussed.

All of the methods presented depend strongly on the organization of the original grammar for parallel structures. Only where constituent boundaries have been delineated in the grammar can conjunction be properly
treated in most cases. This problem can be partially circumvented if the grammar-writer is willing to make adjustments to the original grammar to accommodate the conjunction mechanisms. If this is done, then conjunction arcs may be developed at approximately the same time as the rest of the grammar. This would seem to be a sensible approach, even though the conjunction arcs are strictly not considered part of the grammar.

A related type of limitation can be seen in attempting to handle certain types of what Quirk [QUI72] calls "cataphoric" ellipsis. Considering some of his examples (pp.570-1):

John can (pass the examination), and Bob certainly will, pass the examination

Tom is (playing for the school), Peter will be (playing for the school), and Harold might be, playing for the school

we can see that the elements realized in the sentence occur last in the series, rather than first as with most types of ellipsis. The reason these forms pose problems for our mechanisms relates to the first limitation discussed above. In each of these examples, the verb complex is split so that part of it is elided and part is not. Most grammars would consider the verb complex as a unit to be processed at one level of the grammar. Utilizing our techniques on the examples above, it would necessitate extracting the proper
pattern from the entire path of traversed arcs. Although this may be possible to do in a reasonably efficient and direct way, it is beyond the current scope of this work (however, see the discussion of the SYSCONJ facility of LUNAR in Chapter VI).

Another assumption made above is that the conjunction will always occur in the initial position in the constituent to be conjoined. This may not be the case for all types of conjunction. For example, from the sentence

John loves Mary and Bill loves Juanita

we cannot have

*John loves Mary;  Bill, and, loves Juanita

but we can have

John loves Mary;  Bill, however, loves Juanita

Fortunately, these forms are less common than the others discussed. One approach to handling such forms, however, might be to treat the additional punctuation as if they were conjunctions and proceed to devise a family of techniques that work for those cases.

Future Work

This research has led to the formulation of additional areas for research. Some of these areas extend the methods presented here, while others investigate slightly different
Several linguistic phenomena have been overlooked in this study. Idiomatic and dialectic use of language is common and many of the examples which can be classified as ungrammatical, by our definition, fit into this realm. Studies like [MAK72] and [FRA70] give encouragement to those who would like to think these can be dealt with in some consistent manner. Recent work on the PARRY system [PAR77] has shown that with a moderate sized dictionary of idioms, some remarkable effects can be achieved. Our approach with its intensive use of patterns could easily be extended, we believe, to encompass such a capability.

Also, with some additional extensions, our mechanisms could support the type of organization outlined by Becker in [BEC75]. In his paper, a phrasal lexicon is proposed which allows language forms to be specified as phrasal patterns which are then applied to the input. The applicability of our techniques to situations of this type should be obvious and our future plans include moving in that direction.

In the development of our mechanisms, an emphasis was placed on using structural similarities in the sentences being processed. The techniques, therefore, apply both when comparing the forms of sentences and when comparing their meanings. Of course, the most obvious applications for the
techniques developed in this work are in the realm of syntactic processing, rather than semantic. However, we have not artificially restricted their application. For example, a semantic grammar could be used as the normative grammar (see Burton's semantic grammar in [BUR76]). In such a case, category relaxation would simply allow relaxation from one semantic category to another, more general one.

The problem of selecting which ungrammatical alternative to pursue during parsing can be addressed through the use of better heuristics which measure the potential of each alternative. Selection might be based on the "interestingness" of the words being processed by a given alternative. Schank, et al. has more to say about just how this might be measured in [SCH78b].

Our techniques are also well-suited to implementation in a multiprocessor environment. Each viable ungrammatical alternative may be scheduled for processing independently. It is often the case that in attempting to pursue one of these alternatives, it very quickly becomes blocked, say within one or two arc traversals. Therefore, even though the number of alternatives suggested during the course of parsing may be large when extensively using our techniques, multiprocessing is feasible to overcome this problem.
Another area to be investigated in the future of this study is how errors interact. Obviously, the presence of one error may often influence the remaining processing of the sentence. An approach to this problem might involve studying specific types of error combinations. The mechanisms, as implemented, do not conflict with each other in operation. That is, one mechanism may apply at the beginning of the processing of a sentence and another different mechanism may apply later.

The future of Natural Language Understanding rests in part, we believe, with the increased ability to address problems related to handling unusual and unexpected forms. The most striking property of human understanding capabilities when compared to machines in this regard is their overwhelming tolerance for error. When machine models of language understanding exhibit this capability, then true Natural Language Understanding will be closer at hand.
Appendix A: Syntax of ATN Grammar Specification

Below is a description of how to specify grammars for the Augmented Transition Network (ATN). It is specified in a modified BNF with non-terminals given in pointed brackets. An asterisk which occurs after a non-terminal should be taken to mean zero or more occurrences of that non-terminal.

```
<atn net> ::= (<arc set>*)
<arc set> ::= (<st> <arc>*)
<arc> ::= (CAT <cat> <test> <act>* <term>)
     (CAT* <cat> <test> <act>* <creat act>)
     (JUMP <st> <test> <act>*)
     (MEM <words> <test> <act>* <term>)
     (MEM* <words> <test> <act>* <creat act>)
     (PAT <pat spec> <test> <act>* <term>)
     (POP <form> <test>)
     (PUSH <st> <test> <pre>* <act>* <term>)
     (TST <lab> <test> <act>* <term>)
     (TST* <lab> <test> <act>* <creat act>)
     (VIR <lab> <test> <act>* <term>)
     (WRD <words> <test> <act>* <term>)
     (WRD* <words> <test> <act>* <creat act>)
<test> ::= <act>
<act> ::= <stor act>
     <retr act>
     <expec act>
     <creat act>
     <form>
<stor act> ::= (SETR <reg> <form>)
     (SETRQ <reg> <form>)
     (LIFTR <reg> <form>)
     (LIFTRQ <reg> <form>)
     (HOLD <lab> <form>)
```
<retr act> ::= (GETR <reg>)
(GETF <feat>)
(RFEAT <feat> <form>)
(CTGY <cat>)
(NULLR <reg>)
(NEXTWORD)
FEATURES

<expec act> ::= (EXPECT <creat act> <st>)
(EXPECT <creat act>)

<creat act> ::= (BUILDQ <frag> <reg>*)
(LIST <form>*)
(APPEND <form> <form>)
(QUOTE <expr>)

<pre> ::= (SENDR <reg> <form>)
(SENDRQ <reg> <expr>)
(! <form>)

<term> ::= (TO <st>)

<st> ::= any ATN state name

<cat> ::= ( <ctgy> )
<ctgy>

<ctgy> ::= any lexical category

<words> ::= ( <word> )
<word>

<word> ::= any lexical word

<pat spec> ::= ( <patt> <pat parm>* )

<patt> ::= ( <p arc> )
<patt name>

<p arc> ::= <arc>

<pat name> ::= user-assigned pattern name
<lab> ::= a LISP atom
<reg> ::= any ATN register name
<form> ::= a LISP expression to be EVALed
<expr> ::= any LISP expression
<feat> ::= any lexical feature
<brag> ::= LISP skeletal expression using *, +, and #
Appendix B: Overview of ATNs

As a formal model, the Augmented Transition Network (ATN) exhibits all of the power of a programming language. The ATN formalism considered herein has been carefully constructed to optimize its utility in addressing the problems surrounding the processing of Natural Language. Various extensions to the ATN model which deal with classes of linguistic phenomena are presented in this paper. Readers who wish more discussion should refer to either the tutorial article by Bates [BAT78] or one of the articles by Woods ([WO070] or [WO073a]). Below we introduce some terminology and summarize the capabilities included in our implementation.

The ATN as a Formal Model of Grammar

A finite state network is a directed graph containing nodes and arcs in which the nodes represent "states" of the computation while the arcs represent "transitions" between states. Acceptance of a string by a finite state machine
involves making an appropriate transition for each character of the string and ending up in a specially designated "acceptance" or "final" state. Grammars represented by diagrams of this type are known as regular grammars and languages accepted by these grammars are known as regular languages (or type 3 languages in the Chomsky hierarchy [CH059]) (See an example of a finite state network along with its corresponding regular grammar in Figure 6).

\[
\begin{align*}
S & \rightarrow aX \\
S & \rightarrow a \\
X & \rightarrow aA \\
X & \rightarrow bB \\
A & \rightarrow aA \\
A & \rightarrow c \\
A & \rightarrow cB \\
B & \rightarrow bB \\
B & \rightarrow cA \\
B & \rightarrow b
\end{align*}
\]

**Figure 6**

Finite State Network with Regular Grammar

Context-free (or type 2) languages can be recognized by grammars represented by finite state networks to which recursion has been added. Such a network is called a Recursive Transition Network (RTN). In an RTN, traversal of a single arc which contains the name of a subnetwork
involves recursively traversing a subnetwork. Figure 7 is an example of an RTN with its corresponding context-free grammar.

![Recursive Transition Network with Context-Free Grammar](image)

In an Augmented Transition Network (ATN) the RTN formalism has been augmented by the addition of the capability to set "registers" during arc traversal and also to test registers as a condition to traversing an arc. Such registers can contain flags, words from the input string, partial parse trees, case structures, nodes of a semantic network, or any expression permitted by the host language. Further augmentation of the model, which involves allowing
arbitrary computations to be performed during arc traversal, raises the power of the model to that of a Turing Machine, an acceptor of type 0 languages.

There are eight arc types permitted in the ATN. Four of these parse one word of the input string (WRD, MEM, CAT, and TST), three manipulate the network through recursion and state changes (PUSH, POP, and JUMP), and one re-processes a previously parsed component (VIR). Figure 8 shows a fragment of an ATN intended to parse noun phrases. In this Figure, we have not shown all of the tests and actions, but have only identified the type of transition for each arc along with some of the more important tests.

An ATN Implementation

The ATN was first formalized by W.A. Woods [W0070], but its roots can be traced to early work on compilers in [CON63] and later work on natural language processing in [THO68] and [BOB69]. The implementation discussed in this paper is documented in [KWA74] and was based on the LUNAR [W0072] system implementation described in [W0073a]. This work has since been modified and extended to accommodate the mechanisms of this research. These changes are discussed next.
Figure 8

Augmented Transition Network Fragment
The ATN permits the specification of a non-deterministic grammar. In this implementation, the non-determinism is circumvented through the ordering of arcs at each state and the investigation of alternatives using backtracking. Parsing is performed by interpreting the network in a depth-first manner, with the unexplored alternatives at each state being packaged into a configuration (CFG) which contains sufficient information to continue exploring these alternatives. When no arc traversal can be made from a state, parsing is said to be "blocked" at that state. Backtracking then occurs. In our implementation, backtracking in the network is achieved by recovering a CFG generated earlier and continuing to explore the remaining alternatives.

In our revised ATN model presented here, four new arc types known as "macro" arcs have been added. These are called WRD*, MEM*, CAT*, and TST*. They are identical to their corresponding arc types in the original model, except that instead of providing a destination state, the macro arc provides a new arc computationally, which is then traversed.

The behavior of the parser is determined by the grammar and lexicon provided. The syntax for ATN arc specification is given in Appendix A, while the lexicon consists of a list of feature-value pairs for each word sense where each
feature represents an associated property of the word and the value is that property's corresponding value.

In our implementation, arcs may be stored in an arc dictionary in the form of arc templates and user-defined arc names assigned. Arcs may then be referenced by name. In addition, portions of the arcs can be substituted when they are referenced. The symbol "^" is used in an arc template to indicate that a substitution is required for that element of the arc. For example, suppose the arc template named PUSH-NP-3 has the form

\[(\text{PUSH NP}^* \ ^\text{ (SETR }^*\text{ ) }^\text{)}\]

Then the arc reference

\[(\text{PUSH-NP-3 T SUBJ (TO S*VP))}\]

expands into

\[(\text{PUSH NP}^* \ T \ (\text{SETR } \text{SUBJ }^*) \ (\text{TO S*VP}))\]

and the arc reference

\[(\text{PUSH-NP-3 (TRANS (GETR V)) OBJ (TO S*S))}\]

expands into

\[(\text{PUSH NP}^* \ (\text{TRANS (GETR V)}) \ (\text{SETR OBJ }^*) \ (\text{TO S*S}))\]

There are several advantages to the arc dictionary. Allowing arcs to be named permits the easy manipulation of arcs for the purpose of constructing patterns or computing arcs dynamically. Arc references via the arc dictionary may be made absolutely anywhere that an arc may occur, since arc
references are realized in our implementation through the use of LISP macro functions. Each named arc template has associated with it a macro function by the same name which expands the arc template when referenced. This means that both traditional ATN grammar specifications and arc references may exist within the same system with no problems. Thus, very little extra code or overhead is required in the system to offer this capability. Furthermore, similar arcs may be stored compactly in the arc dictionary utilizing similar arc templates, as can be seen in the above example. Of course, a slight increase in execution time must be tolerated for arc references to be expanded, but these mechanisms could easily be incorporated into an ATN compiler such as the one described in [BUR76] wherein macro expansion would take place at compile time for most references.

An example of a category hierarchy as it would be specified for use by the ATN is given in Appendix D, while a sample set of relaxable tests is given in Appendix E. Sample arcs, arc templates and arcsets are listed in Appendix F.
Appendix C: The NLG Grammar

Figure 9
NLG Sentence-Level Network

123
Figure 10
NLG Noun Phrase Network
Figure 11
NLG Question Network
Figure 12
NLG Prepositional Phrase Network
Figure 13
NLG Touch Network
Appendix D: A Category Hierarchy

The category hierarchy below is shown as it would be actually specified for use by the relaxation mechanism for ATNs. Levels of parenthesis nesting correspond to levels in the hierarchy. For example, the category CORRCONJ would relax to the category CONJ. CTGY-RELAX is a LISP function that defines the hierarchy for the system.

(CTGY-RELAX
  (CONJ
    CORRCONJ
  )
  (ADJ
    N
  )
  (N
    NPR
    (PRON
      DEM
    )
  )
  (V
    COMPL
    COPULA
    AUX
  )
  (Q
    QWORD
    QADV
    QDET
  )
  (#
    ORD
    NUM
  )
  (PREP
    PREPC
    PREPF
  )
)
Appendix E: Relaxable Tests

The list of tests to be relaxed is shown as it would actually be specified for use by the relaxation mechanism for ATNs. TEST-RELAX is a LISP function that defines the relaxable tests. For example, NULLR is an absolutely violable test while INTRANS is a conditionally violable test which relaxes to TRANS.

( TEST-RELAX
    NULL
    INTERSECT
    NULLR
    GETF
    ( INTRANS TRANS )
    RFEAT
  )
Appendix F: Sample Arc and Arcset Specifications

Below is shown a sample arc dictionary similar to the one used in the sample dialogue discussed in detail in Chapter VII. See Appendix B for a discussion of the arc dictionary and arc templates.

(ARC-DATABASE

WRD*-1-2  (WRD* ^ ^ (BUILDQ (PAT-2-1 T)))

WRD*-2-2  (WRD* ^ ^ (BUILDQ
           (PUSH-2-1
           (AND
           (NULL STACK)
           (NOT (EQ STATE @S*))))))

MEM*-1-0  (MEM* (/ , /;) T
            (BUILDQ (WRD*-1-2 AND T)))

MEM-PUNCT-1  (MEM (? 1 / .) (NULL STACK) ^)

PUSH-PP*P-3  (PUSH PP*P ^
              (SENDR PROTOTYPE (GETR PROTOTYPE))
              (SENDRQ TAG ^)
              (SENDRQ C (%POSIT))
              ^)

PUSH-2-1  (PUSH S* ^
            (SETR PROTOTYPE
            (APPEND
            (LIST (PBUILD))
            (LIST *)))))
PUSH-NP-2

(PUSH NP* ^
(PROTOTYPE @OBJECT (GETR N))
(COND
((GETR NAME)
 (PROTOTYPE @NAME
 (GETR NAME)))
(T NIL))
(COND
((GETR CVP)
 (SETR PROTOTYPE
 (UNIONP (GETR CVP)
 (GETR PROTOTYPE))))
(T NIL))
(COND
((GETR NODE)
 (PROTOTYPE @POSIT
 (APPEND (GETP @POSIT)
 (LIST
 (LIST @TOUCH
 (GETR NODE)))))))
(T NIL))
^)

PAT-2-1

(PAT (> ANCHOR OPTIONAL) ^)

PAT-1-3

(PAT (PLINE ANCHOR OPTIONAL) ^
 (SETR ^ *)
 ^)

CAT-V-2

(CAT V ^
 (SETR V *)
 (SETRQ TNS UNTENSED)
 (PROTOTYPE @TYPE (GETR TYPE))
 (PROTOTYPE @ACTION (GETR V))
 ^)

CAT-1-0

(CAT CORRCONJ T
 (EXPECT
 (BUILDQ
 (WRD*-1-2 # T)
 (GETF CONJ))))

POP-S

(POP (PBUILD) (NULL STACK))

POP-1-2

(POP (BUILDQ (S /+) ^) ^)
Below is shown the specification of a sample pattern, PLINE, which is to be included in the pattern dictionary. In this case, the pattern can match the specification of two endpoints with optional specification of action, object, and punctuation.

(PATTERN-DATABASE

PLINE

  ( > (CAT-V-2 (EQ * @DRAW) NIL)
    > (PUSH-NP-2 T NIL)
    (PUSH-PP*P-3 T ENDPOINT NIL)
    (PUSH-PP*P-3 T ENDPOINT
     (COND
      ((NULL (GETP @%ACTION))
       (PROTOTYPE @%ACTION @DRAW)))))

    > (MEM-PUNCT-1 NIL)
    (POP-S))

)

Below is shown a sample specification of the set of conjunction arcs. Each entry refers to an arc template contained in the arc dictionary.

(CONJUNCTION-DATABASE

(WRD*-1-2 AND T)

(WRD*-2-2 AND T)

(CAT-1-0)

(MEM*-1-0)

)
Below is shown a sample specification of a set of alternative arcs. This specification is similar to the specification of the original network insofar as the state names are explicit.

(ALTNETWORK

(S*
   (PAT-1-3 T S (TO VP*))
)

(S*NP
   (PUSH-NP-2 T (TO S*VP))
   (PUSH-NP-2 (GETF TRANS) (TO S*TRANS))
)
)
BIBLIOGRAPHY


Chomsky, Noam, "Degrees of Grammaticalness," in [FOD64], 384-389.


Fraser, Bruce, "Idioms within a Transformational Grammar," Foundations of Language, 6, 22-42, 1970.


<table>
<thead>
<tr>
<th>Reference</th>
<th>Title and Authors</th>
</tr>
</thead>
</table>


[NOO73] Nooteboom, S.G., "The Tongue Slips into Patterns," in [FRO73].


Weischedel, R.M. and J. Black, "Responding to Potentially Unparseable Sentences," manuscript, Department of Computer and Information Sciences, University of Delaware, Newark, Delaware, 1979.


