Phrasal Analysis of Long Noun Sequences

Yigal Arens, John J. Granacki, and Alice C. Parker

University of Southern California
Los Angeles, CA 90089-0782

ABSTRACT

Noun phrases consisting of a sequence of nouns (sometimes referred to as *nominal compounds*) pose considerable difficulty for language analyzers but are common in many technical domains. The problems are compounded when some of the nouns in the sequence are ambiguously also verbs. The phrasal approach to language analysis, as implemented in PHRAN (PHRasal ANalyzer), has been extended to handle the recognition and partial analysis of such constructions. The phrasal analysis of a noun sequence is performed to an extent sufficient for continued analysis of the sentence in which it appears. PHRAN is currently being used as part of the SPAN (SPecification ANalysis) natural language interface to the USC Advanced Design AutoMation system (ADAM) (Granacki *et al*, 1985). PHRAN-SPAN is an interface for entering and interpreting digital system specifications, in which long noun sequences occur often. The extensions to PHRAN's knowledge base to recognize these constructs are described, along with the algorithm used to detect and resolve ambiguities which arise in the noun sequences.

1. Introduction

In everyday language we routinely encounter noun phrases consisting of an article and a head noun, possibly modified by one or more adjectives. Noun-noun pairs, e. g., *park bench, atom bomb,* and *computer programmer,* are also common. It is rare, however, to encounter noun phrases consisting of three or more nouns in sequence. Consequently, research in natural language analysis has not concentrated on parsing such constructions.

The situation in many technical fields is quite different. For example, when describing the specifications of electronic systems, designers commonly use expressions such as:

bus request cycle
transfer block size
segment trap request
interrupt vector transfer phase
arithmetic register transfer instruction.

During design specification such phrases are often constructed by the specifier in order to reference a particular entity: a piece of hardware, an activity, or a range of time. In most cases the nouns preceding the last one are used as modifiers, and idiomatic expressions are very rare. In almost all cases the meaning of noun sequences can therefore be inferred largely based on the last noun in the sequence*. (But see Finin (1980) for in-depth treatment of the meaning of such constructions).

The process of recognizing the presence of these expressions is, however, complicated by the fact that many of the words used are syntactically ambiguous. Almost every single word used in the examples above belongs to both the syntactic categories of noun and verb. As a result,

bus request cycle

may conceivably be understood either as a com-

* When a sequence has length three or more the order of modification may vary. Consider:
[engine damage] report
January [aircraft repairs]
[boron epoxy] [[rocket motor] chambers]
1970 [[balloon flight]
[[solar-cell standardization] program]].
But the last noun is still the modified one. These examples are from (Rhyne, 1976) and (Marcus, 1979).

mand (to bus the request cycle) or as a noun phrase.

Considerable knowledge of the semantics of the domain is necessary to decide the correct interpretation of a nominal compound and the natural language analyzer must ultimately have access to it. But before complete semantic interpretation of such a noun phrase can even be attempted the analyzer must have a method of recognizing its presence in a sentence and determining its boundaries.

1.1. The Rest of this Paper

The rest of this paper is structured as follows: In the next section, Section 2., we describe the phrasal analysis approach used by our system to process input sentences. In Section 3. we discuss the problems involved in the recognition of long noun sequences, and in Section 4. we present our proposed solution and describe its implementation. Sections 5. and 6. are devoted to related work and to our conclusions, respectively.

2. The PHRAN-SPAN System

PHRAN, a PHRasal ANalysis program, (Arens, 1986) (Wilensky and Arens, 1980), is an implementation of a knowledge-based approach to natural language understanding. The knowledge PHRAN has of the language is stored in the form of pattern-concept pairs (PCPs). The linguistic component of a pattern-concept pair is called a *phrasal pattern* and describes an utterance at one of various different levels of abstraction. It may be a single word, or a literal string like

Digital Equipment Corporation,

or it may be a general phrase such as

(1) <component> <send> <data> to <component>

which allows any object belonging to the semantic category *component* to appear as the first and last constituents, anything in the semantic category *data* as the third constituent, any form of the verb *send* as the second, while the lexical item *to* must appear as the fourth constituent.

Associated with each phrasal pattern is a *conceptual template,* which describes the meaning of the phrasal pattern, usually with references to the constituents of the associated phrase. Each PCP encodes a single piece of knowledge about the language the database is describing.

For the purpose of describing design specifications and requirements a declarative representation language was devised, called SRL (Specification and Requirements Language). In SRL the conceptual template associated with phrasal pattern (1) above is a form of *unidirectional value transfer.* In this specific case it denotes the transfer of the data described by the third constituent of the pattern by the controlling agent described by the first constituent to the component described by the fifth. For further details of the representation language used see (Granacki *et al,* 1987).

PHRAN analyzes input by searching for phrasal patterns that match fragments of it and replacing such fragments with the conceptual template associated with the pattern. The result of matching a pattern may in turn be present as a constituent in a larger pattern. Finally, the conceptual template associated with a pattern that accounts for all the input is used to generate a structure denoting the meaning of the complete utterance.

A slightly more involved version of the PCP discussed above is used by PHRAN-SPAN to analyze the sentence:

> *The cpu transfers the code word from the controller to the peripheral device.*

3. The Problem with Long Noun Sequences

Long noun sequences pose considerable difficulty to a natural language analyzer. The problems will be described and treated in this section in terms of phrasal analysis, but they are not artifacts of this approach. A comparison with other approaches to such constructs, mentioned later in this paper, also makes this clear.

The main difficulties with multiple noun sequences are:

- Determination of their length. One must make sure that the first few nouns are not taken to constitute the first noun phrase, ignoring the words that follow. For example, upon reading *bus request cycle* we do not

want the analyzer to conclude that the first noun phrase is simply *bus*, or *bus request*.

- Interpretation of ambiguous noun/verbs. A large portion of the vocabulary used in digital system specification consists of words which are both nouns and verbs. Consequently the phrase *interrupt vector transfer phase*, for example, might be interpreted as a command to interrupt the vector transfer phase, or (unless we are careful about number agreement) as the claim that phase is transferred by interrupt vectors.

  In spoken language stress is sometimes used to "adjective-ize" nouns used as modifiers. For example, the spoken form would be "arith*me*tic register transfer" rather than "a*rith*metic register transfer". Obviously, such a device is not available in our case, where specifications are typed.

- Determination of enough about their meaning to permit further analysis of the input. Full understanding of such expressions requires more domain knowledge than one would wish to employ at this point in the analysis process (Cf. Finin (1980)). However, at least a minimal understanding of the semantics of the noun phrase is necessary for testing selectional restrictions of higher level phrasal patterns. This is required, in turn, in order to provide a correct representation of the meaning of the complete input.

The phrasal approach utilizes the phrasal pattern as the primary means of recognizing expressions, and in particular noun sequences. In effect, a phrasal pattern is a sequence of restrictions that constituents must satisfy in order to match the pattern. The most common restrictions on a constituent in a PHRAN phrasal pattern, and the ones relevant in our case, are of the following three types:

1. The constituent must be a particular word;
2. It must belong to a particular semantic category; or,
3. It must belong to a particular syntactic category.

In addition, simple lookahead restrictions may be attached to any constituent of the pattern. In the original version of PHRAN such restrictions were limited to demanding that the following word be of a certain syntactic category.

Simple phrasal patterns are clearly not capable of solving the problem of recognizing multiple noun sequences. It is not possible to anticipate all such sequences and specify them literally, word for word, since they are often generated on the fly by the system specifier.

For a similar reason phrasal patterns describing the sequence of semantic categories that the nouns belong to are, as a rule, inadequate.

Finally, from the syntactic point of view all these constructions are just sequences of nouns. A pattern simply specifying such a sequence provides little of the information needed to decide which expression is present and what it might refer to.

4. A Heuristic Solution

PHRAN's inherent priority scheme was used to solve part of the problem. If a word can be used either as a noun or a verb, it is recognized first as a noun, all other things being equal. This simple approach was modified to be subject to the following rules:

1. If the current word is a noun, and the next word may be either a noun or a verb, test it for number agreement (as a verb). If the test is unsuccessful do not end the noun phrase.
2. If the current word is a noun, and the next word may be either a noun or a verb, test if the current word* is a possible active agent with respect to the next (as a verb). If not, do not end the noun phrase.
3. If the current word is a noun, and the next word may be either a noun or a verb, check the word after the next one. If it is (unambiguously) a verb, end the noun phrase with the next word. If it is (unambiguously) a noun, do not end the noun phrase. If the second word away may be either a noun or a verb, treat the utterance as potentially ambiguous, with a noun phrase ending either at the current word or with the next word.

Once a complete noun phrase is detected a new token is created to represent its referent.

* The current word may be the last in a sequence of nouns; we are again assuming that its meaning can be used to approximate the meaning of the noun sequence.

While all nouns used in its construction are noted, it inherits the semantics of the last noun in the sequence. This information may be used in later stages of the analysis. Other programs which receive the analyzer's output will inspect the representation of the noun phrase again later to determine its meaning more precisely.

The heuristic described above has been found to be sufficient to deal with all inputs our system has received up until now. It detects as ambiguous a sentence such as the following:

The cpu signal interrupts transfer activity.

When looking at the word *cpu* PHRAN-SPAN finds that Rule 1. can be used. Since number agreement is absent between *cpu* and *signal* (used as a verb), the noun phrase cannot be considered complete yet. When the word *signal* is processed, the system notes that *interrupts* may be either a (plural) noun or a verb. Number agreement is found, and it is also the case that a signal may act as an agent in an action of interruption, so rules 1. and 2. provide no information. Using Rule 3. we find that the following word, *transfer* is an ambiguous noun/verb. Thus the result of the analysis to this point is indicated as ambiguous, possibly

a. [the cpu signal] [interrupts] [transfer activity], or

b. [the cpu signal interrupts] [transfer] [activity].

The type of ambiguity detected by Rule 3. can often be eliminated by instructing the users of the specification system to use modals when possible. In case of the example above, to force one of the two readings for the sentence, a user might type *the cpu signal will interrupt transfer activity,* or *the cpu signal interrupts will transfer activity,* as appropriate.

4.1. Requesting User Assistance

When Rule 3. detects an ambiguity, the system presents both alternatives to the user and asks for an indication of the intended one.

PCPs encode in their phrasal pattern descriptions, among other things, selectional restrictions that at times allow the system to rule out some of the ambiguities detected by Rule 3. For example, it is conceivable that interrupts might not be acceptable as agents in a transfer. PHRAN-SPAN would thus be capable of eventually ruling out analysis b. above on its own.

However, more often than not it is the case that both interpretations provided by Rule 3. are sensible. We decided that the risk of a wrong specification being produced required that in cases of potential ambiguity the system request immediate aid from the user. Therefore, when sentences like the one in the example above are typed and processed, PHRAN-SPAN will present both possible readings to the user and request that the intended one be pointed out before analysis proceeds.

4.2. Rule Implementation

The rules described above are implemented in several pattern-concept pairs and are incorporated into the standard PHRAN knowledge base of PCPs. For example, one of the PCPs used to detect the situation described in Rule 1. while taking into consideration Rule 3. is (in simplified form):

Pattern:
{<article> <sing-noun & next N/V &
next non-sing &
after-next verb>}

Concept
{part of speech: noun phrase
semantics: inherit from (second noun)
modifiers: (first noun)}

4.3. Current Status

The system currently processes specifications associated with all primitive concepts of the specification language, which are sufficient to describe behavior in the domain of digital systems. Pattern-concept pairs have been written for 25 basic verbs common in specifications and for over 100 nouns. This is in addition to several hundred PCPs supplied with the original PHRAN system.

The system is coded in Franz LISP and runs on SUN/2 under UNIX 4.2 BSD. In interpreted mode a typical specification sentence will take 20 cpu seconds to process. No attempt has been made to optimize the code, compile it, or port it to a LISP processor. Any of these should result in an

interface which could operate in near real-time.

5. Related Work

The problem of noun sequences of the kind common in technical fields like digital system specification has received only limited treatment in the literature. Winograd (Winograd, 1972) presents a more general discussion of Noun Groups, but the type of utterances his system expects does not include extended sequences of nouns as are common in our domain. Winograd therefore does not address the specific ambiguity problems raised here.

Gershman's Noun Group Parser (NGP) (Gershman, 1979) dealt, among other things, with multiple noun sequences. While our algorithm is consistent with his, our approach differs from NGP in major respects. NGP contains what amount to several different programs for various types of noun groups, while we treat the information needed to analyze these structures as data. PHRAN embodies a general approach to language analysis that does not require components specialized to different types of utterances. A clear separation of processing strategies from knowledge about the language has numerous advantages that have been listed elsewhere (Arens, 1986). In addition, our treatment of noun groups as a whole is integrated into PHRAN and not a separate module, as NGP is.

In evaluating the two systems, however, one must keep in mind that the choice of domain greatly influences the areas of emphasis and interest in language analysis. NGP is capable of handling several forms of noun groups that we have not attempted to deal with.

Marcus (1979) describes a parsing algorithm* for long noun sequences of the type discussed in this paper. It is interesting to note that the limited lookahead added to the original PHRAN for the purpose of noun sequence recognition is consistent with Marcus' three-place constituent buffer. The major difference between Marcus' algorithm and ours is that the former requires a semantic component that can judge the relative "goodness" of two possible noun-noun modifier pairs. For example, given the expression *transfer block size*, this component would be responsible for determining whether *block size* is semantically superior to *transfer block*.

Such a powerful component is not necessary for achieving our present objective – recognizing the presence and boundaries of a noun sequence. Our heuristic does not require it.

A complementary but largely orthogonal effort is the complete semantic interpretation of long noun sequences. There have been several attempts to deal with the problem of producing a meaning representation for a given string of nouns. See (Finin, 1980) and (Reimold, 1976) for extensive work in this area, and also (Brachman, 1978) and (Borgida, 1975). Such work by and large assumes that the noun sequence has already been recognized as such. I. e., it requires the existence of a component much like the one described in this paper from which to receive a noun sequence for processing.

6. Conclusions

We have presented a heuristic approach to the understanding of long noun sequences. The heuristics have been incorporated into the PHRasal ANalyzer by adding to its declarative knowledge base of pattern-concept pairs. These additions provide the PHRAN-SPAN system with the capability to translate digital system specifications input in English into correct representations for use by other programs.

7. Acknowledgements

We wish to thank the anonymous reviewers of this paper for several helpful comments.

This research was supported in part by the National Science Foundation under computer engineering grant #DMC-8310744. John Granacki was partially supported by the Hughes Aircraft Co.

8. Bibliography

Arens, Y. *CLUSTER: An approach to Contextual Language Understanding.* Ph.D. thesis, University of California at Berkeley, 1986.

* Discovered by Finin (1980) to be erroneous in some cases.

Borgida, A. T. *Topics in the Understanding of English Sentences by Computer.* Ph.D. thesis, Department of Computer Science, University of Toronto, 1975.

Brachman, R. J. *Theoretical Studies in Natural Language Understanding.* Report No. 3833, Bolt Beranek and Newman, May 1978.

Finin, T. W. *The Semantic Interpretation of Compound Nominals.* Ph.D. thesis, University of Illinois at Urbana-Champaign, 1980.

Gershman, A. V. *Knowledge-Based Parsing.* Ph.D. thesis, Yale University, April 1979.

Granacki, J., D. Knapp, and A. Parker. The ADAM Design Automation System: Overview, Planner and Natural Language Interface. In *Proceedings of the 22nd ACM/IEEE Design Automation Conference,* pp. 727-730. ACM/IEEE, June, 1985.

Granacki, J., A. Parker, and Y. Arens. Understanding System Specifications Written in Natural Language. In *Proceedings of IJCAI-87, the Tenth International Joint Conference on Artificial Intelligence.* Milan, Italy. July 1987.

Marcus, M. P. *A Theory of Syntactic Recognition for Natural Language.* The MIT Press, Cambridge, Mass. and London, England, 1979.

Reimold, P. M. *An Integrated System of Perceptual Strategies: Syntactic and Semantic Interpretation of English Sentences.* Ph.D. thesis, Columbia University, 1976.

Rhyne, J. R. A Lexical Process Model of Nominal Compounding in English. *American Journal of Computational Linguistics,* microfiche 33. 1976.

Wilensky, R., and Y. Arens. PHRAN: A Knowledge-Based Natural Language Understander. In *Proceedings of the 18th Annual Meeting of the Association for Computational Linguistics.* Philadelphia, PA. June 1980.

Winograd, T. *Understanding Natural Language.* Academic Press, 1972.