5-19-1988

A Survey of Computerized Writing Aid Systems and Implementation of a Homonym Checker Using Artificial Intelligence

Anna Marie Robbins

A Survey of Computerized Writing Aid Systems
and
Implementation of a Homonym Checker
Using Artificial Intelligence

by

Anna Marie Robbins

Rochester Institute of Technology

School of Computer Science and Technology

A Survey of Computerized Writing Aid Systems

and

Implementation of a Homonym Checker Using Artificial Intelligence

by Anna Marie Robbins

A thesis submitted to

The Faculty of the School of Computer Science and Technology,

in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science

Approved by:

Guy Johnson	3/28/88
Professor Guy Johnson	Date
John A. Biles	5/17/88
Professor John A. (Al) Biles	Date
Peter G. Anderson	5/19/88
Professor Peter G. Anderson	Date

Abstract

Computerized writing aid systems ideally determine whether word usage in written text is appropriate both syntactically and semantically. Currently developed computerized writing aid systems are reviewed with emphasis on their methods for and success at grammar checking. A limited homonym checker and correction advisor is devised and implemented using the Prolog programming language and artificial intelligence techniques. Insight is gained into the problem of developing a comprehensive computerized system for determining appropriate syntactic and semantic word usage.

Key Words and Phrases
(with Computing Reviews Categories)

Semantic Network (I.2.4.)
Natural Language Interfaces (I.2.1.)
Natural Language Parsing (I.2.7.): Text Analysis, Language Parsing and Usage, Pattern Matching
Information Systems Applications (H.4.): Style Analyzers, Word Processing (H.4.1.), Writing Aid Systems, Critique, Writer's Workbench, Grammar Checking
Homonym Checking
Appropriate Word Usage

Table of Contents

1. Introduction

The creation of written documents (such as letters, memoranda, papers, legal instruments, etc.) is an integral part of our daily lives. Today, most written documents are created at least in part using computers because of the computer's flexibility, performance, and economy.

Initially, only the most mechanical of writing tasks (such as formatting text, improving the appearance of the printed page, editing text, and checking spelling) could be effectively implemented using computers and so-called "word processing" software. As the state of computer technology has improved and people have become more accustomed to using computers, there has arisen a great desire to use computers and so-called "writing aid" software to automate writing tasks that are still being done manually, including the activities involved in prewriting organization and proofreading. Ideally, writing aid systems will determine whether word usage in written text is appropriate both syntactically and semantically.

Most of the writing aid systems currently developed have resulted from the investigation of what should constitute appropriate word usage. One system, developed by Lorinda Cherry and others at AT&T Bell Labs while conducting such

an investigation, is based upon a method for pattern matching the sentence structure in written text to a set of grammar rules, and then assigning parts of speech to the words as they are used in the sentence [FRAS83]. This system spurred AT&T researchers to create a set of related software modules for performing a variety of writing aid functions. These modules, which have become known collectively as Writer's Workbench, provide editorial comments on the appropriateness of punctuation, word use, spelling, text abstractness, grammatical parts of speech and text readability, and have formed the basis on which all other currently available writing aid systems have been designed.

IBM's investigations in the area of appropriate word usage have led to the development of another writing aid system, originally called EPISTLE and now referred to as Critique. Although not currently available, this system is reported to use a natural language parser to generate grammar and style analysis about written text. Writing aid information provided by Critique about a written document may be grouped into four critique categories: grammar, style, words and phrases, and summary analysis [RICH85].

Of the many aspects involved in determining appropriate syntactic word usage, the most difficult to accomplish is that of grammar checking. There are two major approaches

to grammar checking: pattern matching and parsing. There are many variations to the parsing approach including techniques commonly employed to "fix" incorrect parses (such as parse fitting, parse approximation and relaxation). As noted above, the two major writing aid systems, Writer's Workbench and Critique, use pattern matching and parsing, respectively, as the basis for their writing analysis.

It is generally believed that the optimum solution to determining if word usage is appropriate would be achieved only if computers could be "taught" to understand contextually the meanings of words as they are used in the sentence and as they are used within the complete written document. Although researchers in the field of Natural Language are studying the problem of how best to determine if word usage is appropriate semantically, no general solution has been found to date.

This thesis considers whether manipulating a subset of the English language can provide insight into the problem of determining appropriate word usage. In order for this task to be manageable in scope, consideration has been limited to the English Language word subset called homophones (i.e., words that sound alike but are spelled differently, which are customarily referred to as homonyms).

The unique relationships between sets of words that are homonyms make them ideal for examination. This subset is manipulated by implementing a homonym checker to detect both appropriate syntactic and semantic use of homonyms within a sentence. A determination is made in the first instance if the homonym is the correct part of speech for the sentence, and then, if so, if the homonym is related to other words in the sentence based upon a predefined network of word relationships.

2. Background

2.1 Previous Work - Writer's Workbench

Writer's Workbench is a comprehensive computerized writing aid system developed by AT&T Bell Laboratories and is the only major writing aid system presently commercially available. Writer's Workbench is designed to assist in the creation of documents by automating copy editing and proofreading tasks [FRAS83]. The purpose of Writer's Workbench is to improve the quality of computerized written documents and is designed to be used with any written text, regardless of the subject [RASK86].

Scientists at AT&T Bell Laboratories, including linguists, computer scientists and psychologists, were studying the

mechanics of language analysis when they developed the program PARTS for assigning word classes to text. They then created a variety of programs to support PARTS that now comprise the Writer's Workbench writing aid system [CHER78].

After a two year trial and study period, the designers of Writer's Workbench expanded the existing facilities, updated the current systems and released the system. Since many of the original pioneers have moved on to other research areas, there are no plans currently for any further updates of the system [Telephone Conversation 4].

The main features of the Writer's Workbench system include providing editorial comments on punctuation, word uses, spelling, text abstractness, analysis of grammatical part of speech, and calculation of text readability, all in a manner interactive with the user. The Writer's Workbench system includes about 40 programs, and requires 700 Kilobytes of memory and the UNIX operating system to execute, making the system accessible to many users [RASK86].

2.1.1 Design:

While developing Writer's Workbench the researchers at Bell Laboratories worked to incorporate six design principles into the program [FRAS83]. First, the system should be

rational, that is, based on research and expert consensus drawn from psychological research, linguistics research and writing and style experts. Second, the system should be diverse by providing many statistics about text and word characteristics, and readability analysis, and by providing a variety of options to the users by allowing the user to invoke the different modules separately. Third, the system should provide an evaluation of the text by comparing an individual document's statistics with "norms" set by expert researchers in language theory. Fourth, the system should be modifiable, allowing the user to add and delete words and phrases from the dictionary and change the output format and length. Fifth, the system should be specific in regard to sentence length. Last, the system should provide guidance and information about word use, punctuation and other features.

Most of programs comprising the Writer's Workbench system employ databases of stylistic and grammatical rules and use lookup tables to analyze the text by pattern matching.

These programs are divided into three areas: proofreading, stylistic analysis and reference information on English usage and other Writer's Workbench programs. All the programs in the system are based on the PARTS design [CHER81].

The PARTS program uses general rules of sentence structure and word order to assign word classes to English text. The PARTS system consists of three programs. The first program performs preprocessing functions (such as stripping headers and command lines) and dictionary lookups of 210 function words and 140 irregular verbs. The second program checks for suffixes based upon a specific database of suffixes defined in the system. The last program assigns word classes by reading the whole sentence with the partial word class assignments and calling a scan routine that looks up the rules of sentence structure from the database for each dependant and independent clause in the sentence. The scan routine first looks for verbs, then nouns, and finally all other word classes. Some special word classes are assigned based upon the proximity of the word to previously assigned verbs, nouns and matching English phrases.

The authors of PARTS claim a 95% accuracy rate in assigning words to their classes. The common errors PARTS makes involve confusion between nouns and adjectives, confusion between nouns and verbs, errors in PARTS' dictionary, and an inability to process idioms and imperatives (although imperatives can be processed if specially marked). PARTS assumes that the sentences are well written (i.e., follow common grammar structure), and, therefore, cannot accurately process non-sentences.

2.1.2 Components:

Writer's Workbench consists of four major program modules (PARTS, STYLE, DICTION and SUGGEST) that each have a different purpose, and several additional groupings of program modules for performing related functions [MACD83]. PARTS (as described above) is a program that assigns word classes to text using rules. STYLE is a program that summarizes information about the written document, including readability, sentence word length and structure, word usage, and sentence openers. DICTION is a program that identifies phrases that are frequently misused or indicate wordiness. SUGGEST is an interactive thesaurus program for phrases found by DICTION.

The proofreading program PROOFR invokes five separate program modules including: SPELLWWB, an interactive spell checking program; PUNCT, a program that searches for simple punctuation errors; DOUBLE, a program that detects consecutive occurrences of the same word; DICTION and SUGGEST, programs that are described above; and, SPLITINF, a program that finds split infinitives and suggests grammatical information about the errors.

A series of programs that may be generically referred to as style analyzers, give information about the document style. These programs include the STYLE program described

above, the PROSE program, which enhances the STYLE program by providing comparisons of the style values against established standards of English, and the FINDBE program, which deals with the various forms of "to be."

The following programs provide on-line reference information. WORDUSE provides a description of the correct use of over 300 English words and phrases. SPELLTELL deals with a list of 800 commonly misspelled words. The WWBINFO program lists all the programs and their functions. WWBHELP provides on-line access to a list of functions and program names.

Writer's Workbench also includes other associated programs. ORG provides an outline of the document by stripping the first and last sentences of each paragraph. SEXIST is a variation of DICTION containing 100 possible sexist words and phrases. ABST determines the conceptual abstractness of a document by counting the number of word occurrences that are on a list of words rated as abstract. TOPIC uses PARTS to locate frequent noun phrases from which it produces key words or indexes.

2.1.3 Conclusions:

The Writer's Workbench system can improve the user's knowledge of writing and style techniques. Users of the system

claim it saves proofreading time and provides immediate and objective criticism of the text [GING83].

However, implementing the advice received by the prose and diction programs are sometimes difficult. The Writer's Workbench system is limited by the text features it can recognize and does not use linguistic parsing algorithms. It cannot assess semantic word usage.

2.2 Previous Work - Critique

Critique, formerly know as EPISTLE, is a computerized writing aid system developed by IBM and not presently available outside IBM. As a consequence, the entire discussion that follows is derived from IBM's own reports and not independent evaluation.

Like Writer's Workbench, Critique is a system that aids in the computerized preparation of written text. The orientation of Critique is such that it provides diagnoses for problems in text that has already been written, rather than in the initial writing process. Although originally designed for reviewing business correspondence in an office environment, Critique is suitable for any application where written text is created and maintained.

The principle feature that distinguishes Critique from other writing aid systems is that it is based on the use of a broad coverage, natural language parser. The main functions of Critique are to provide useful information about usage of certain words and phrases, diagnose grammatical errors, and identify possible stylistic problems [RICH85].

2.2.1 Design:

Critique analyzes text using a Programming Language for Natural Language Processing (PLNLP) parser [RICH85]. The PLNLP parser uses Augmented Phrase Structure Grammar (APSG) rules to parse written text. The parser creates parse trees for each text-segment and, in those cases where no parse is produced, a special procedure is applied to generate a "fitted parse." Critique System Architecture includes two separate virtual machines identified as the user machine and the server machine. The server machine includes the PLNLP parser and a manager for controlling the flow of text segments and critique information between the machines.

Processing on the user machines includes a text labeler component that employes format clues to identify segments of text such as sentences, headings, paragraphs, etc. A lexical analyzer component process segment-like sentences to access the dictionary and analyze the style of the

text. The dictionary access procedure dynamically handles derivational and inflectional affixation as well as word compounding and spelling. The PLNLP parser performs parsing and analyses for grammar and style. Available style analysis output selections include calculation of readability and length and text annotation.

The manager portion of the server machine handles the flow of tasks between the user's machine and the server machine including the PLNLP parser, tracks system use and status information and collects comments from the users. The PLNLP parser receives (decodes), parses, and returns (encodes) critique information about text segments sent from the user's machine. Critique can be operated in either batch or interactive modes.

2.2.2 Components:

The Critique system is comprised of four component program modules: Grammar Critiques; Style Critiques; Word and Phrase Critiques; and Summary Analysis Critiques [HEID82].

Grammar Critiques detects approximately 25 grammar errors most of which involve various types of disagreement between, and incorrect forms of, verbs and pronouns. The PLNLP machine processes Grammar Critiques during the second parse attempt when constraints in the grammar rules are relaxed.

There are about 50 Style Critiques which are produced from rules which operate on the parse structures generated by the grammar processing. The Style Critiques module provides information on sentence length, number of modifiers and readability level.

Critiques generated by the grammar and style rules on the PLNLP machine address spelling errors; awkward, redundant and trite phrases; unacceptable structural context; and easily confused homonyms; and are among the over 40 errors detected by Word and Phrase Critiques.

Summary Analysis Critiques are mainly statistical in nature, and are generated by the user interface program using information produced by both the lexical analyzer and the PLNLP parser. These include sentence counts, sentence-type paradigms, and critiques of approximately 50 types of statistics.

2.2.3 Conclusions:

The use of a natural language parser makes it possible to identify grammar and style errors that would otherwise not be possible to diagnose. Since Critique is currently being used on an experimental basis, information on which to make an analysis of its performance is not available [RICH85].

2.3 Previous Work - General

In addition to the two major systems mentioned above, there are several other systems that provide a more narrow scope of assistance in writing. All of these other systems are designed in part based upon the Writer's Workbench writing aid system. Most contain large lexicons of grammatical and stylistic rules based on Strunk and White's renowned text "Elements of Style", and compare a given word processed document against this lexicon. These systems are word and phrase oriented and are blind to context. They simply cannot and do not attempt to understand individual style or information in context.

Although some of these writing aid systems claim to do grammar checking, they are in fact style analyzers and base their grammar analysis upon matching phase dictionaries included with their system software. The major function of these style analyzers is to flag simplistic but frequently encountered errors in word usage. Some of the specific features that may be incorporated into these style analyzers are listed in Table 2-1 below [RASK86].

Features	Style and Grammar Capabilities
Copy protection	
Spell checking	Flags split infinities
Display of error check on screen	Flags passive voice
	Flags awkward language
Suggestion of alternatives for errors	Flags archaic language
	Flags cliches and jargon
Marks up text	Flags redundancies
Marks text with comments	Flags confusibles
Allows customized rules and phrases	Flags gender-specific terms
	Flags vague words
Ignores blocks of text	Flags wordiness
Documentation	Flags wrong word
Word processing compatibility	Flags homonyms
Punctuation Capabilities	**Stylistic Indicators**
Flags run-on sentences	Readability
Flags unbalanced punctuation	Descriptiveness
Flags white space errors	Sentence structure
Flags capitalization errors	Word-frequency studies
Flags double words	

Table 2-1 Style Analyzers: Summary of Features

In general these style analyzers are totally inflexible to individual needs. (The User's Manual for RightWriter Version 2.1 expressly states on page ix that for writing requiring "creativity" its suggestions are "of little use".) For example, they assume that passive voice is better than active voice, they mark archaic language that may not be so, they act on the premise that good writing demands short sentences, and, in many cases, provide too much information, especially for individuals who produce a great deal of written documents. Since most of these

systems only flag about 25% of mistakes, the user must be aware of the need to continue to manually proofread their written text.

One advantage of these systems is that they force the users to think about their writing, helping to eliminate repetitive mistakes. Because of this advantage, they make good teaching tools in providing students with a computerized means of analyzing their writing. Also, since most of these smaller writing aid systems are relatively inexpensive, they are affordable for more users than the larger systems.

2.4 New System Ideas

Although much research is being done on determining appropriate word usage and improving writing aid systems, there are two recent developments of particular interest. The first development involves creation of a new writing aid system, while the second development may significantly improve the success of any writing aid system requiring assignment of parts of speech.

Hull et al. are working at the University of Pittsburgh under a grant from The Ford Foundation and Digital Equipment Corporation to develop a new system based upon pattern

matching theory to assist teaching good writing skills to low-performing college students [HULL87].

The Hull team wishes to implement a variety of modifications to the pattern matching scheme employed by Writer's Workbench. First, they would like to create an empirical database of error types and linguistic clues rather than basing the database on style guides and writing manuals. This is possible because there are vast numbers of student essays and other written texts stored on and available through the University of Pittsburgh's computer systems. They intend to search this collection of text to determine the most commonly found error categories and incorporate these in their database.

The Hull team also hopes to use a system of pattern matching based upon part of speech phrases rather than individual words, and introduce changes in the searching heuristics. A second pass checking system is to be included for those sentence constructions that are flagged as having possible errors. In the future it is intended that a parser will be added to check sentence grammars before being grammar checked by the pattern matching system. It is interesting to note that one of the 40 pattern errors to be included in the Hull taxonomy is a means of homophone error detection.

Another possible technique that may hold promise for improving the determination of appropriate word usage has been suggested in a dissertation by Gaven Duffy of the University of Texas at Austin. This technique, called Categorial Disambiguation, contemplates resolving grammar ambiguities caused when more than one category (i.e., part of speech assignment) is possible [DUFF86]. Resolution of the ambiguities is completed before sentence parsing is begun.

Resolution is achieved by creating a set of prioritized pattern action rules for each categorial ambiguity. (Forty such rules have been created by Gaven Duffy.) Each rule is passed a list of words and the categories that precede and succeed the categorial ambiguity. If the rule succeeds, it propagates solutions to other categorial ambiguities. If the rule fails, the next rule is checked. Usually, examining one preceding category and one succeeding category is sufficient to make the correct interpretation.

The implementation of this categorial ambiguity system still remains in the development stage, but this technique may become a useful tool in the development of grammar parsing systems.

2.5 Grammar Checking - Methods and Problems:

No writing aid system would be complete without an effective method of doing grammar checking. As the previous section explains, no truly effective system of this nature currently exists. Many linguists, psychologists and computer scientists are presently working on the solution to the grammar checking problem.

Grammar checking methods to date may be broadly classified as parsing or pattern matching. Parsing methods have been developed from the efforts of researchers in the field of Artificial Intelligence who are studying natural language processing solutions to the grammar checking problem. These solutions entail creating various systems for grammatical representation (such as by lexical-functional grammars) of language structures. These grammatical representations are then used as a basis for designing grammar parsing systems [KAPL82a].

One major problem that current grammar parsing systems suffer from is grammar structures that fail the parsing alternatives [JENS83]. These failures to parse may be caused by a variety of reasons, such as an input sentence's ingrammatical nature, or an ill-formed or ambiguous grammar. Some parsing systems simply give the user a

message that the sentence cannot be parsed, while others look for alternative solutions to the failed parsing problems. Alternative solutions include relaxing the parser by permitting a word which is slightly inappropriate to be replaced by a correct word in the sentence [KWAS81]; allowing for phrasal analysis in the absence of a total sentence parse; and, adding a pattern matching algorithm to recognize specifically identified grammar problems [DUFF86].

Some researchers are trying to alleviate ambiguity problems that exist in parsing systems by incorporating into their parsing algorithms some form of semantic analysis by assigning words to categories such as "subject", "object" and "tense" [WEIS80].

Other significant problems with grammar parsing systems that are currently being used are that they require substantial processing time and power, and therefore cannot be interactive, cannot be used on smaller computer systems, and are not accessible to the common user.

As will be appreciated from the New Systems Ideas section above, researchers are still trying to find a quality system for doing grammar checking using the theories of pattern matching [HULL87]. Grammatical error detection is made by searching a database of patterns. Some systems

match by using internal strings of characters (words) and others match on phrases or part of speech patterns.

Some of the difficulties in using the pattern matching approach arise because of the inability to specify surface feature patterns in the text, resulting in both correct and incorrect structures. Also, limitations must be made on the number of patterns that can possibly be contained in the database list.

The advantages that the pattern matching systems have over parsing systems include speed of processing time, the ability to interact with the user and provide immediate feedback, execution on smaller computer systems giving greater access to more users, and the ability to detect the most common (if not a large number) of grammar errors.

3. Project Description

3.1 Discussion of Homonyms

The unique characteristics of homonyms makes them ideal for implementation of this project. The principle characteristics making homonyms desirable are that homonyms by definition are already related (i.e., they sound alike and are often mistakenly interchanged in written text) and they are

limited in number. Therefore, investigating appropriate usage of homonyms is a much more manageable task than attempting to implement a generalized grammar checker and likely detects a greater number usage errors than for any other similarly sized class of words.

Restricting the implementation to homonyms also hopefully will have two other desirable immediate and practical implications: there will be one more (albeit limited) class of words for which computers may begin to check usage; and, valuable insight may be gained concerning how best to approach solutions to the significantly more difficult problem of determining appropriate word usage for the English language as a whole.

A true homonym is a word that is spelled the same as another word and has the same pronunciation but is different in meaning, as in bear (an animal) and bear (to carry). A homophone is a word that has the same pronunciation as another word but has a different spelling, as in bear (an animal) and bare (naked). A homograph is a word that has the same spelling as another word but is different in pronunciation and syllabication, as in lead (to show the way) and lead (the mineral). However, convention dictates that homographs, homophones and homonyms all be generically referred to as homonyms. Although this implementation

deals with homophones, the term homonyms is used throughout in accordance with convention [ELLY77],[NEWH78].

Because the vast majority of homonym sets are comprised of homonyms each of which have different parts of speech, it is feasible to determine appropriate word usage in the vast majority of instances using only parts of speech. This leaves only a small set of homonym words having the same part of speech for which the determination of appropriate word usage will have to rely solely on a network of semantic word relationships.

Statistically it is useful to understand that there are more than 3500 individual homonym words with more than 8000 meanings. Most, but not all, homonyms begin with the same letter and only 6 homonym sets begin with 3 different letters. Only 520 homonyms have more than 3 or more variant spellings; all others exist as pairs. The most homophonous sound is "air" with 14 variant spellings and 38, meanings and the next is "sol" with 6 variant spellings and 35 different meanings [NEWH78].

3.2 Functional Specification:

3.2.1 Functions Performed:

The homonym checker is designed to determine if a sentence

contains homonym words and if the homonyms are the appropriate homonyms for the sentence. Appropriateness is determined in the first instance by whether or not the homonym has the correct part of speech for the sentence, and in the second instance by whether or not the homonym is semantically related to the other words in the sentence. Additionally, the homonym checker searches the defined database of homonyms for and informs the user of related homonyms that have both the correct part of speech for the sentence and are semantically related to the other words in the sentence. Where two homonyms are found having the same correct part of speech as used in the sentence, the determination of the appropriate homonym is based upon the semantic word relationships.

3.2.2 Limitations and Restrictions:

a. The input data file may not contain any command characters commonly placed in such files by word processing systems, because such characters cause irrecoverable failure in certain of the programs and languages used in this implementation and there is no means of removing these characters prior to further processing.

b. The homonym checker only determines the erroneous use of those words that are defined in the database of homonyms. The homonym checker does not process homonyms that are contractions.

c. Semantic word relationships are found only if at least one of the other words in the sentence containing a homonym matches one of the words in a predefined list of related words stored in the database. Thus, appropriateness of a word may not be recognized simply because of the limited nature of the related words list contained in the database.
d. Since the homonym checker determines appropriate homonym usage in the first instance based on parts of speech, the homonym checker cannot be more accurate that the accuracy with which the parts of speech are determined. Because the homonym checker uses the PARTS module associated with the Writer's Workbench system, the assignment of part of speech usage to each word in the sentence is only as accurate as that achieved by PARTS. AT&T Bell Labs claims PARTS has a 95% accuracy rate (assuming the user gives special attention to sentences that begin with imperatives by marking them with the tilde symbol) [Writer's Workbench User's Guide, pp. 3-81].

3.2.3 User Input Forms:

The homonym checker can accept input from any file containing sentences so long as the files do not contain headers or special characters.

3.2.4 User Output Forms:

Output from the homonym checker identifies the sentence num-

ber, the homonym, the homonym's part of speech, all related homonyms (if any) and related sentence words (if any). Also, English language messages are output to the user regarding the appropriateness of homonym usage. No output is provided for sentences without homonyms found in the database of homonyms.

3.3 System Architectural Design:

3.3.1. System Organizational Charts:

There are three charts attached to the Appendix of this thesis: a Data Flow Diagram describing the flow of data between the various system files (Fig. 1); an overall System Organizational Chart (Fig. 2); and a Homonym Checker Structure Chart (Fig. 3).

3.3.2. Equipment Configuration:

The implemented homonym checker and associated program modules were created using the following software and programming languages running under the UNIX operating system:

a. PARTS, one of the many modules incorporated into the UNIX based AT&T Bell Labs Writer's Workbench writing aid system, was used to assign parts of speech to words as they are used in a sentence.

b. The Pascal programming language was used to write one

of the two data transformation programs, named "lower". This program converts all upper case letters to lower case.

c. The Awk Pattern Scanning and Processing Language was used to create the second data transformation program, named "awkprog". This Awk program changes the output data from PARTS into a database of facts that conforms to the syntax requirements of the Prolog programming language.

d. The Prolog programming language was used to create the homonym checking program. Prolog is an ideal language for this task because of the backtracking and database fact searching features of the language.

3.3.3. Implementation Tools:

a. Data Files and System programs:

Fig. 1 presents a Data Flow Diagram and an explanation of the data files and system programs that are part of or operate with the Homonym Checker. All programs that comprise the Homonym Checker are contained in the file "alfiles". The file "alfiles" consults the following files: "homfd", "amrtools" and "rnets". The file "homfd" contains all the code for the Homonym Checker. The file "amrtools" contains two separate Prolog modules for printing messages and for determining word membership in the homonym database "rnets". The file "rnets" is the homonym database, and is described below in Section 3.3.4.d.

b. User Instructions:

Since the PARTS module is located at RIT on the AT&T PC computer system, and Prolog and the other language compilers are located on the ISIS computer system, the various data files must be moved from one system to the other. (Of course, if everything was loaded on the same system, access between the various tools and programs would be simplified.) The PARTS module is located on RIT's AT&T PC computer system through "tinker". To run the input sentences datafile through PARTS the following command line is typed:

```
parts -w datafile > partsout
```

The partsout file is then moved to ISIS and the script that executes the data transformation programs is called. This script runs the partsout data file through the Pascal program "lower" and creates the file "convert", runs "convert" through the Awk program "awkprog" to create the a sentence facts database (called "sentdb") in a syntax compatible with Prolog, and removes "convert". Execution of the script to perform data transformation is invoked by typing:

```
amrrun
```

Then Prolog is invoked and the Homonym Checker program and Prolog compatible sentence facts database files are consulted by typing:

```
[alfiles,sentdb].
```

Finally, the Homonym Checker is run by typing the following request line:

```
homtest(Sentno,Hom,Part,Relatedhom,Relatedword).
```

The output is currently configured to be printed only to the screen. After each solution is output to the screen, entering a semicolon will bring up the next answer.

3.3.4. System Design: (See Fig. 2)

The main focus of the Homonym Checker is to determine if a homonym word is used appropriately within the structure of the sentence. Since the homonyms that comprise most homonym sets each have different parts of speech, in the first instance this determination is made based upon the homonym's part of speech resulting from its actual use in the sentence. In order to make this determination the part of speech for sentence words first must be somehow identified, such as by grammar parsing or pattern matching. Rather than creating a grammar parser (which no one has yet done successfully in doing and is a major project itself), the PARTS module portion of the Writer's Workbench computerized writing aid system is used on the sentences to make the parts assignments. Then a comparison between the homonyms' parts usage and a database containing the homonyms

and their correct part of speech is made.

Further consideration is given to determining the appropriate word usage by those homonyms in a given homonym set having the same part of speech. This aspect involves making some assessment of the semantic word relationships between the homonym and the other words in the sentence. The Prolog programming language is used because this is a natural language processing problem and the Prolog language is ideal for its solution.

Correct part of speech usage and word relationship determinations are made by comparing the sentence words and parts of speech to a database of facts relating to each homonym.

Since Prolog requires a specific syntax form for facts, the output from PARTS must be transformed into this acceptable form before the facts can be utilized.

The following sections describe in detail the major segments of the Homonym Checker system: PARTS; Data Transformation; The Homonym Checker; and, RNETS Creation - The Semantic network of word relationships.

a. PARTS

PARTS assigns parts of speech to words by using a dictionary of function words, irregular verb forms, and word endings to classify many of the words. Then it classifies the remaining words by looking for relations between them. PARTS assigns words to one of thirteen word classes: noun, verb, article, adjective, adverb, conjunctive, preposition, interjection, auxiliary verb, pronoun, subordinate conjunction, to be, and possessive. Since PARTS uses word endings to identify most verbs, it cannot recognize an imperative verb without specially marking the sentence [Writer's Workbench User's Guide p. 3-81]. Since the PARTS module can be used separately, the data file containing the sentences is run through PARTS and (by using the -w option) the output is formatted with only one word and its part of speech per line, separated by a tab.

b. Data Transformation

The output file from PARTS must be transformed into a Prolog sentence facts database. Two programs have been created to make this transformation. First, since Prolog treats all upper case letters as uninstantiated variables, all upper case letters must be changed to lower case. The Pascal program called "lower" makes this transformation.

Second, Prolog requires that all facts follow the form:

predicate (argument, argument).

The Awk program "awkprog" changes each word/part pair into the form:

sent (sentno,word,part).

and also creates sentence number facts for each sentence in the form:

sentno(no).

Both the "lower" program and "awkprog" are run using the "amrrun" script file. The output from the transformation is referred to as "sentdb".

c. Homonym Checker (See Fig. 3)

The Homonym Checker proceeds through the following process: First, the Homonym Checker determines if the sentence contains a homonym. If no homonym is found no action is taken.

Then, if there is a homonym the Homonym Checker compares the rnets database facts with the sentdb facts to see if the given homonym has the same part of speech as used in the sentence. If the Homonym does not have the correct part of speech, a message is sent to the user and the Homonym Checker then searches the rnets database to see if there is a related homonym that does have the correct part of speech. If a related homonym is also not found, a message is given to the user and the process stops. If a

related homonym is found then the Homonym Checker searches the list of words associated with the related homonym to see if any conform with the other words in the sentence. In either case, true or false, a message is given to the user and the process stops.

If, however, the homonym does have the correct part of speech, a message to this effect is given to the user and the Homonym Checker then tests to see if the given homonym has a word in its list of related words that conforms to other words in the sentence. In either of these cases a message to the user is given and the Homonym Checker then looks for related homonyms using the same process as discussed above.

d. Homonym Database

The homonym database (named "rnets") contains one fact for each homonym and for every part of speech that the homonym may have. Each fact contains the homonym, the part of speech for the homonym, a list of related homonyms, and a list of related words, and takes the form:

```
r_net(homonym,part,[related homonyms],[related words]).
```

The homonyms that are contained in this database were arbitrarily chosen. However, an attempt was made to choose the most commonly used homonyms. The list of words defined as semantically related to the homonyms were also arbitrarily

selected, although many of the word choices were taken from general text, books, newspapers and the dictionary. There remains many possibilities for the enhancement of this list, some of which will be discussed in the conclusion section of this thesis. (A copy of the "rnets" database may be found in the Appendix.)

4. Verification and Validation:

4.1 Test Plan and Procedures:

Three types of sentence files are used to determine how well the Homonym Checker performs. The first set of sentences were arbitrarily chosen to test each feature of the Homonym Checker and determine if correct messages are given to the user. The first set of sentences specifically tests whether:

a. The sentence contains a homonym;
b. The sentence does not contain a homonym;
c. The homonym has the correct part of speech;
d. The homonym does not have the correct part of speech;
e. The homonym is related to other words in the sentence;

f. The homonym is not related to other words in the sentence;

g. There is a related homonym that has the correct part of speech; and, if so, whether

h. The related homonym is related to other words in the sentence; and whether

i. The related homonym is not related to the other words in the sentence.

The second set of test sentences comprise a poem about homonyms. The poem was found in a dictionary of homonyms [NEW78]. The author purposely used some of the homonyms correctly and others incorrectly in the poem. The results from this test allow for a general analysis of the successes and failures of the Homonym Checker.

The third set of test sentences include excerpts from two newspaper articles and one letter to the editor in its entirety. Analysis of these results helps determine how well the Homonym Checker processes commonly found text.

Conclusions are drawn from the percentage of homonyms found, the ability of the Homonym Checker to identify appropriately used homonyms based on the correct part of speech, and the ability of the Homonym Checker to determine appropriate homonym usage based upon semantic word relation-

ships, especially where two homonyms have the same part of speech.

4.2 Test Results:

The first set of sentences includes 20 individual sentences and 49 occurrences of homonyms that are collected in a single file named SENTTEST. Of the 49 occurrences of homonyms in the file SENTTEST, the Homonym Checker identified 46. At the time this test was run, three occurrences of homonyms were words not in the homonym database. Two of these three occurrences involved the same homonym -- "in", and the other involved the word "your". All aspects of the Homonym Checker worked properly and all messages to the user were correct. There were no errors in this data set. Table 4-1 presents the results for this set of test sentences.

Total Number of:

Homonym Occurrences	Homonyms Identified	Related Words	Related Homonyms	Related Words to Related Homonyms
49	46	10	24	12

Reconciliation of Homonym Occurrences:

Homonyms Correctly Identified (with correct P.O.S.):

Homonyms & NO Related Homonyms :	23
Related Homonyms & NO Homonyms :	7*
Homonyms AND Related Homonyms :	15
Total Homonyms Correctly Identified	45*

Note: *Homonyms having multiple related homonyms with the correct part of speech are counted only once. In SENTTEST there were two homonym occurrences that each had two related homonyms with the correct part of speech, increasing from 45 to 47 the total number of homonyms correctly identified. The homonym "pare" (verb) was incorrectly used as a noun, resulting in the identification by the Homonym Checker of the two related homonyms "pair" and "pear" (both nouns). The homonym "road" (noun) was incorrectly used as a verb, resulting in the identification of the two related homonyms "rode" and "rowed" (both verbs).

Explanation of Incorrect Homonym Identification:

PARTS P.O.S. assignment errors:	0
Difference between PARTS P.O.S. assignment and P.O.S. listing in homonym database RNETS:	0
Stripped contractions:	0
No listing in homonym database for assigned P.O.S.:	1
Total Homonym Incorrect Identification	1

Total Number of Homonyms Not Identified:
(due to exclusion from homonym database RNETS): 3

Table 4-1 SENTTEST Test Results

The second set of test sentences comprised the poem noted above. The poem contained 60 occurrences of homonyms both correctly and incorrectly used in sentences. The Homonym Checker identified 46 of the 60 occurrences of homonyms.

Table 4-2 presents the results for this set of test sentences.

Total Number of:

Homonym Occurrences	Homonyms Identified	Related Words	Related Homonyms	Related Words to Related Homonyms
60	46	0	11	1

Reconciliation of Homonym Occurrences:

Homonyms Correctly Identified (with correct P.O.S.):

Homonyms & NO Related Homonyms	:	20
Related Homonyms & NO Homonyms	:	4
Homonyms AND Related Homonyms	:	5
Total Homonyms Correctly Identified		29

Explanation of Incorrect Homonym Identification:

PARTS P.O.S. assignment errors:	12
Difference between PARTS P.O.S. assignment and P.O.S. listing in homonym database RNETS:	2
Stripped contractions:	1
No listing in homonym database for assigned P.O.S.:	2
Total Homonym Incorrect Identification	17

Total Number of Homonyms Not Identified:
(due to exclusion from homonym database RNETS): 14

Table 4-2 SENTPOEM Test Results

The Homonym Checker correctly identified 20 occurrences where only the homonym had the correct part of speech and 4 occurrences (listed in Table 4-3) where only the related homonym had the correct part of speech.

Homonym	Assigned Part of Speech	Related Homonym
won	adjective	one
coarse	noun	course
lead	verb	heard
grate	adjective	great

Table 4-3 Occurrences in SENTPOEM of Related Homonyms & No Homonyms With Correct Part Of Speech

Table 4-4 lists the five instances where both the homonym and the related homonyms had the correct part of speech, and the one occurrence of these where a semantically related word was used to determine the appropriate homonym for the sentence.

Homonym	Assigned Part of Speech	Related Homonym	Related Words
lead	verb	led	--
principle	noun	principal	--
rode	verb	rowed	--
sail	noun	sale	--
knight	noun	night	day

Table 4-4 Occurrences in SENTPOEM of Homonyms & Related Homonyms With Correct Part Of Speech

There appears to be an unusually high number of homonym identification errors in this set of test sentences. These errors may be collected into four categories as set forth in the section of Table 4-2 entitled "Explanation of Incorrect Homonym Identification".

a. PARTS Part Of Speech Assignment Errors.

The vast majority of homonym identification errors in SENTPOEM resulted from the incorrect assignment by the PARTS module of part of speech to each homonym as it was used in the sentence. PARTS assumes that "good" grammar structures are used in all sentences. Since many of the sentences in SENTPOEM contained poor grammar structures due to the intentional misuse of homonyms by the author, PARTS made many erroneous word class assignments.

If PARTS is unsure of the part of speech assignment, it looks the word up in its dictionary and assigns to the word a part of speech regardless of its use in the sentence. Table 4-5 contains several examples of PARTS part of speech misassignments.

Actual Homonym	Assigned Part of Speech	Actual Part of Speech Usage	Correct Homonym
urn	noun	verb	earn
wood	noun	auxiliary verb	would
bee	noun	be	be

Table 4-5 Typical Part of Speech Assignment Errors by PARTS

b. Difference Between PARTS Part of Speech Assignment and Part of Speech listing in Homonym Database RNETS

One source of misassignments is that PARTS has a tendency to confuse noun and adjective usage, as described above in Section 2.1 Previous Work - Writer's Workbench. For

example, PARTS assigned the homonym "daze" the part of speech adjective when the correct part of speech was a noun and the correct homonym "days".

Another source of assignment errors results from PARTS assigning parts of speech to some words based not upon usage, but upon the relationship between the unassigned word to other words in the sentence already assigned parts of speech. Because of the large number of incorrectly used homonyms in SENTPOEM, homonyms that were used correctly were assigned incorrect parts of speech. This produced a difference between the assigned part of speech and the homonym's part of speech as it is properly contained in the homonym database RNETS. For example, PARTS assigned the homonym "more" the part of speech pronoun when its actual part of speech usage was adjective and the homonym "more" was the correct homonym for the sentence. Also, the correct homonym "see" was assigned the part "noun" when its actual part usage was as a verb.

Table 4-6 contains several additional examples of errors made by PARTS in assigning parts of speech to words in SENTPOEM.

Actual Homonym	Assigned Part of Speech	Actual Part of Speech Usage	Correct Homonym
hour	adjective	possessive	our
new	noun	verb	knew
eye	noun	pronoun	I
buy	noun	preposition	by (found bye)
oar	adjective	preposition	or

Table 4-6
Additional Part of Speech Assignment Errors by PARTS

c. Stripped Contractions

Incorrect part of speech assignment may also result from format modifications to the input file necessary to permit processing by the Prolog language Homonym Checker. One such format modification is the elimination of apostrophes. Since Prolog cannot process apostrophes, all apostrophes in words in the Prolog input file must be removed.

PARTS splits contractions between the main word and its contractive part. For example, PARTS splits the word "don't" into "aux do" and "adv n't". The awkprog program, in addition to its other functions, strips out all words with apostrophes. Therefore, in the example of "don't", the awkprog program removes out the "adv n't" line from the PARTS output file, leaving "do" with the wrong part of speech assignment.

d. No Listing In The Homonym Database RNETS For Assigned Part of Speech

Since so many nouns can be used as adjectives, there are some homonyms that do not have adjective listings in the homonym database RNETS. So, in some instances although a homonym may be found in the homonym database, it is not included in the database with the correct part of speech, producing an incorrect homonym identification. One example of this situation occurs with the phrase "ate homophones" in SENTPOEM. The Homonym Checker should have identified the homonym "eight", except that there is no adjective listing for "eight" in the database and no related homonym was found in the sentence containing this phrase. One possible solution to this problem is to simply increase the size of the database to include more adjective listings.

The results of processing the third set of test sentences through the Homonym Checker are presented below in Tables 4-7, 4-8 and 4-9 for the excerpts from two newspaper articles and one letter to the editor, respectively.

As can been appreciated from Tables 4-7 to 4-9, the Homonym Checker identified most of the homonyms and most were used with the correct part of speech. The majority of homonyms did not have related homonyms that possessed the correct part of speech, and only in the case of the homonym word

"sign" was there a related word found in the sentence (in this instance, "language").

Total Number of:

Homonym Occurrences	Homonyms Identified	Related Words	Related Homonyms	Related Words to Related Homonyms
14	11	0	1	0

Reconciliation of Homonym Occurrences:

Homonyms Correctly Identified (with correct P.O.S.):

Homonyms & NO Related Homonyms :	9
Related Homonyms & NO Homonyms :	0
Homonyms AND Related Homonyms :	1
Total Homonyms Correctly Identified	10

Explanation of Incorrect Homonym Identification:

PARTS P.O.S. assignment errors:	0
Difference between PARTS P.O.S. assignment and P.O.S. listing in homonym database RNETS:	0
Stripped contractions:	1
No listing in homonym database for assigned P.O.S.:	0
Total Homonym Incorrect Identification	1

Total Number of Homonyms Not Identified:
(due to exclusion from homonym database RNETS): 3
(Excluded words -"need" and two occurrences of "not")

Table 4-7 SENTNEW1 Test Results

Total Number of:

Homonym Occurrences	Homonyms Identified	Related Words	Related Homonyms	Related Words to Related Homonyms
8	8	2	1	0

Table 4-8 SENTNEW2 Test Results

Reconciliation of Homonym Occurrences:

Homonyms Correctly Identified (with correct P.O.S.):

Homonyms & NO Related Homonyms	:	6
Related Homonyms & NO Homonyms	:	0
Homonyms AND Related Homonyms	:	1
Total Homonyms Correctly Identified		7

Explanation of Incorrect Homonym Identification:

PARTS P.O.S. assignment errors:	0
Difference between PARTS P.O.S. assignment and P.O.S. listing in homonym database RNETS:	1
Stripped contractions:	0
No listing in homonym database for assigned P.O.S.:	0
Total Homonym Incorrect Identification	1

Total Number of Homonyms Not Identified:
(due to exclusion from homonym database RNETS): 0

Table 4-8 SENTNEW2 Test Results (Continued)

Total Number of:

Homonym Occurrences	Homonyms Identified	Related Words	Related Homonyms	Related Words to Related Homonyms
30	27	0	0	0

Reconciliation of Homonym Occurrences:

Homonyms Correctly Identified (with correct P.O.S.):

Homonyms & NO Related Homonyms	:	26
Related Homonyms & NO Homonyms	:	0
Homonyms AND Related Homonyms	:	0
Total Homonyms Correctly Identified		26

Explanation of Incorrect Homonym Identification:

PARTS P.O.S. assignment errors:	0
Difference between PARTS P.O.S. assignment and P.O.S. listing in homonym database RNETS:	0
Stripped contractions:	1
No listing in homonym database for assigned P.O.S.:	0
Total Homonym Incorrect Identification	1

Total Number of Homonyms Not Identified:
(due to exclusion from homonym database RNETS): 3
(Excluded Words - "soul", "time" and "way")

Table 4-9 SENTNEW3 Test Results

The test results for the SENTNEWs files underscores that the majority of homonym words used in common text are prepositions and other words whose parts of speech are not nouns, adjectives or action verb words. As can be appreciated from Table 4-10, the appropriateness of most homonyms used in common text may be determined by their part of speech since their related homonyms possess different parts of speech.

Total Homonyms Identified by correct P.O.S.:		
Prepositions	19	
Possessives	4	
Pronouns	4	
Non-Action Verbs	9	
All other parts (nouns, adjectives, auxiliaries, conjunctives, etc.)	7	
Total Homonyms Identified (with correct P.O.S.)		43
Total Homonyms Identified (NO correct P.O.S.)		3
Total Number of Homonyms Not in Homonym Database:		6
Total Homonym Words:		52

Table 4-10 Part Of Speech Characteristics in SENTNEW Files

5. Conclusions:

5.1 Problems Encountered and Solved:

Only two areas of difficulty were encountered in and had to be solved while implementing the Homonym Checker. These two areas can be broadly categorized and referred to as

data compatibilities and data separation. Peculiarities of the PARTS module and Prolog programming language require that data files be modified to be presented in compatible formats. For example, the PARTS input text file of sentences cannot contain headers or imbedded commands from word processing systems other than the UNIX based "nroff" system. All input to Prolog must use lower case letters since Prolog interprets upper case letters as uninstantiated variables, and must exclude special characters like apostrophes and hyphens. Data transformation programs were included in the Homonym Checker to meet these data compatibility requirements.

In order for the Homonym Checker to find related words in the sentence, a method was devised for keeping track of the sentence currently under analysis. The data transformation program awkprog addressed this need by adding a sentence number fact to the Prolog sentence database for every end of sentence delimiter.

5.2 Discrepancies and Shortcomings:

There are two major shortcomings of the Homonym Checker and a minor coding discrepancy. Also, it may be desirable to enhance the output of the Homonym Checker.

The first major shortcoming occurred because the Homonym Checker relies heavily on the overall effectiveness of PARTS to accurately assign parts of speech to the words in the input sentences. Therefore, the Homonym Checker fails to determine appropriate homonym usage every time PARTS fails to produce an acceptable part assignment. This can be clearly seen in the test results for SENTPOEM.

The second major shortcoming of the Homonym Checker involves the occasional ineffectiveness of the semantic word relationship network to establish appropriate homonym usage. In the few cases where both the homonym and related homonym had the same part of speech, determining the appropriate homonym was rarely made using the related sentence words.

The reasons for this difficulty with the semantic word relationship network can be traced to the creation and contents of the database. Since the words in the semantic network were arbitrarily chosen, creating a comprehensive database was very difficult. Also, there was a tendency to use synonyms as related words. Since synonyms are rarely used together in the same sentence, they rarely are able to resolve appropriate homonym usage and therefore are a poor choice for the semantic network. Finally, because the best words to use as semantically related words are nouns,

adjectives and action verbs, and the test sentences reveal instances in which the homonym was the only noun, adjective or action verb in the sentence, this leaves no other words in the sentence to support the determination of appropriateness of the homonym. In short, the sentence did not contain any related words from the semantic relationship network.

The minor coding discrepancy involves a homonym duplicity problem. The Homonym Checker finds all entries in the database for each homonym fact regardless of the part of speech usage. This is due to the nature of the coding and to the backtracking features of Prolog. This is a minor bug in the program and does not affect in any manner the outcome of the test results.

A useful minor enhancement to the Homonym Checker would be to print the complete input sentence with the output messages. Currently, the Homonym Checker only prints the sentence number.

5.3 Performance and Feasibility:

There are two basic premises underlying the Homonym Checker. The first premise is that appropriate homonym usage may be found in the first instance from parts of

speech (since the vast majority of related homonyms have different parts of speech). The second premise is that in those instances where related homonyms have the same part of speech, appropriate homonym usage may be found from a semantic relationship network. The performance and feasibility of the Homonym Checker can be seen to be directly related to the accuracy of these two premises.

After previewing the list of homonyms and assessing the test results, it is now known that most homonyms sets are of different parts of speech. Therefore, determining appropriate homonym usage by correct part of speech is feasible.

As to the set of homonyms with the same part of speech, although the test results are scarce, this aspect of Homonym Checking appears to be difficult to accomplish, but by no means impossible.

Certainly, the overall performance of the Homonym Checker is sufficiently good to justify further pursuing this approach.

5.4 Improvements & Suggested Future Extensions:

Several improvements that may be made to the Homonym Checker are evident. Perhaps the most apparent is to increase the size of the homonym database and the completeness of the semantic word relationship network listing. Another area of possible fruitful improvement would be to computerize the most difficult aspect of expanding the semantic relationship network -- that of identifying related words to each homonym. Finally, as the homonym database increases in size, it will become more desirable and imperative to implement better database searching techniques.

The Homonym Checker may be made more powerful and taken closer to that of a grammar checker by at least two extensions to the work done for this thesis. First, words that are not true homophones but have similar characteristics, which have been called in the literature "almost homonyms", very possibly could be successfully added to the Homonym Checker by inclusion in the database. "Almost homonyms" (which embrace such words as bed, bid, than, then, and them) are as troubling to most writers as the most confusing homonym sets.

Perhaps the most difficult enhancement, but one which might

yield the greatest step toward true grammar checking, would be to modify the Homonym Checker and semantic word relationship network to include probabilities of relationships between each related word and the homonym. Whether or not a probabilistic semantic word relationship network would be successful is unknown at this time and will require significant future research.

5.5 Generalized Application:

Applying the techniques of the Homonym Checker to a general determination of appropriate word usage is not believed to be possible because the entire English language word set does not possess the unique characteristics of homonyms. However, incorporating the Homonym Checker with other writing aid systems would be desirable. This would permit the accurate grammar checking of at least one additional set of words that are a frequently source of grammatical errors, homonyms. It also then may be possible to perform grammar checking for other word sets such as "almost homonyms". Further, the inclusion of the Homonym Checker in a general writing aid system may have a synergistic effect, helping users spot other grammar errors undetected by the writing aid system in use.

Bibliography:

[ATT85] AT&T UNIX System WRITER'S WORKBENCH Software User's Guide, 1985.

[AUGU87] Augustine, D. "Software Reviews: Writer's Helper." Computers and the Humanities, Vol. 21, pp. 119-121, 1987.

[BENT85] Bentley, J. "A Spelling Checker." Communications of the ACM, Vol. 28, No. 5, pp. 456-462, May 1985.

[BORL87] Borland International Turbo Prolog Toolbox User's Guide and Reference Manual, 1987.

[BORL86] Borland International Turbo Prolog Owner's Handbook, 1986.

[BRAT86] Bratko, I. Prolog Programming for Artificial Intelligence, Addison-Westley, 1986.

[BUCK87] Buckingham, W. J. "Software Review: Right-Writer." Computers and the Humanities, Vol. 21, pp. 3-19, 1987.

[BUMP87] Bump, J. "CAI In Writing At The University: Some Recommendations." Computer Education, Vol. 11, No. 2, pp. 121-133, 1987.

[CARL76] Carlson, R.; Granstrom, B. "A Text-To-Speech System Based Entirely On Rules." IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 686-688, 1976.

[CHER82] Cherry, L. L. "Writing Tools." IEEE Transaction on Communications, Vol. COM-30, No. 1, pp. 100-104, January 1982.

[CHER81] Cherry, L. L. "Writing Tools - The STYLE & Diction Programs." Computing Science Technical Report #91, Bell Laboratories, Murry Hill, New Jersey, February 1981.

[CHER78] Cherry, L. L. "PARTS - A System for Assigning Word Classes to English Text." Computing Science Technical Report #81, Bell Laboratories, Murry Hill, New Jersey, June 1978.

[CLOC81] Clocksin, W. F.; Mellish, C. S. Programming In Prolog, Springer-Verlag, New York, 1981.

[COLE80] Cole, R. A.; Jakimik, J. "A Model of Speech Perception." Perception and Production of Fluent Speech, Cole, R. A. ed., pp. 133-163, Lawerence Erlbaum Associates, 1980.

[DALT85] Dalton, R. "Specialized Writing Tools." Popular Computing, Vol. 5, No. 1, pp. 21-27, November 1985.

[DUFF86] Duffy, G. "Categorial Disambiguation." Proceedings of AAAI 86, 5th National Conference on Artificial Intelligence, Philadelphia, Pennsylvania, Vol. 2, pp. 1079-1082, 1986.

[ELLY81] Ellyson, L. A Dictionary of Homonyms, American House, New York, 1981.

[FERR86] Ferraro, S. "Computing Style." American Way, pp. 21-23, February 4, 1986.

[FRAS83] Frase, L. T. "The Unix™ Writer's Workbench Software: Philosophy." The Bell System Technical Journal, Vol. 62, No. 6, pp. 1883-1890, August 1983.

[GILL87] Gillis, P. D. "Using Computer Technology To Teach And Evaluate Prewriting." Computers and the Humanities, Vol. 21, pp. 3-19, 1987.

[GING83] Gingrich, P. S. "The Unix™ Writer's Workbench Software: Results of a Field Study." The Bell System Technical Journal, Vol. 62, No. 6, pp. 1909-1921, August 1983.

[GRAL87a] Gralla, P. "Examining the Value of Electronic Writing Aids." PC Week, Vol. 4, No. 21, pp. 121-122, May 26, 1987.

[GRAL87b] Gralla, P. "3 Style Checkers Are Put Through Their Paces." PC Week, Vol. 4, No. 21, pp. 126-130, May 26, 1987.

[GRAL87c] Gralla, P. "Packages Cater to All Phases of the Writing Process." PC Week, Vol. 4, No. 21, pp. 131, May 26, 1987.

[HAYE81] Hayes, P.; Mouradian, G. V. "Flexible Parsing." American Journal of Computational Linguistics, Vol. 7, No. 4, pp. 232-242, October-December 1981.

[HEID82] Heidorn, G. E.; Jensen, K.; Miller, L. A.; Byrd, R. J.; Chodorow, M. S. "The EPISTLE Text-Critiquing System." IBM Systems Journal, Vol. 21, No. 3, pp. 305-326, 1982.

[HULL87] Hull, G.; Ball, C.; Fox, J. L.; Levin, L.; McCutchen, D. "Computer Detection of Errors in Natural Language Texts: Some Research on Pattern-Matching." Computers and the Humanities, Vol. 21, pp. 103-118, 1987.

[JENS86] Jensen, K.; Heidorn, G. E.; Richardson, S. D.; Haas, N. "PLNLP, PEG, and CRITIQUE: Three Contributions to Computing in the Humanities." Conference on Computers and Humanities, University of Toronto, April 16-18, 1986.

[JENS84] Jensen, K.; Heidorn, G. E. "First Aid to Authors: The IBM EPISTLE Text-Critiquing System." '84 28th IEEE Computer Society International Conference, IEEE Computer Society Press, pp. 462-464, San Francisco, California, March 1984.

[JENS83] Jensen, K.; Heidorn, G. E.; Miller, L. A.; Ravin, Y. "Parse Fitting and Prose Fixing: Getting a Hold on Ill-Formedness." American Journal of Computational Linguistics, Vol. 9, No. 3-4, pp. 147-160, July-December 1983.

[KAPL82a] Kaplan, R. M. "Lexical-Functional Grammar: A Formal System for Grammatical Representation." The Mental Representation of Grammatical Relations, The MIT Press, 1982.

[KAPL82b] Kaplan, R. M. "Grammars as Mental Representations of Language." The Mental Representation Of Grammatical Relations, The MIT Press, 1982.

[KWAS81] Kwasny, S. C.; Sondheimer, N. K. "Relaxation Techniques for Parsing Ill-Formed Input." American Journal of Computational Linguistics, Vol. 7, No. 2, pp. 99-108, April-June 1981.

[LEMO85] Lemos, R. S. "Rating The Major Computing Periodicals On Readability." Communications of the ACM, Vol. 28, No. 2, pp. 152-157, February 1985.

[MACD82] MacDonald, N. H.; Frase L. T.; Gingrich P. S.; Keenan S. A. "The Writer's Workbench: Computer Aids for Text Analysis." IEEE Transaction on Communications, Vol. COM-30, No. 1, pp. 105-110, January 1982.

[MACD83] MacDonald, N. H. "The Unix™ Writer's Workbench Software: Rational and Design." The Bell System Technical Journal, Vol. 62, No. 6, pp. 1891-1908, August 1983.

[MCCL87] McClure, G. M. "Readability Formulas: Useful or Useless?" IEEE Transactions on Professional Communication, Vol. PC-30, No. 1, pp. 12-15, March 1987.

[MCMA78] McMahon, L. E.; Cherry, L.; Morris, R. "UNIX™ Time-sharing System: Statistical Text Processing." The Bell System Technical Journal, Vol. 57, No. 6, Pt. 2, pp. 2137-2154, July-Aug. 1978.

[MILL81] Miller, L. A.; Heidorn, G. E.; Jensen, K. "Text-Critiquing with the EPISTLE System: An Author's Aid to Better Syntax." AFIPS Conference Proceedings, Vol. 50, Arlington, Virginia, pp. 649-655, 1981.

[MORR78] Morris, W. ed. The American Heritage Dictionary, Houghton Mifflin Co., p. 631, 1978.

[NEWH78] Newhouse, D. Homonyms 'sound-alikes', Newhouse Press, CA, 1987.

[PETE86] Peterson, J. L. "A Note On Undetected Typing Errors." Communications of the ACM, Vol. 29, No. 7, pp. 633-637, July 1986.

[PIER69] Pierce, J. R. "Letters To The Editor: Whither Speech Recognition?" The Journal of the Acoustical Society of America, Vol. 46, No. 4 (Part 2), pp. 1049-1051, 1969.

[RASK86] Raskin, R. "The Quest For Style." IEEE Transactions on Professional Communications, Vol. PC 29, No. 3, pp. 10-18, September 1986.

[RICH85] Richardson, S. D. "Enhanced Text Critiquing Using a Natural Language Parser." Proceedings of the Seventh International Conference on Computers and the Humanities, Paradigm Press, June 26-28, 1985

[REDD80] Reddy, R. "Machine Models of Speech Perception." Perception and Production of Fluent Speech, Cole, R. A. ed., pp. 215-242, Lawerence Erlbaum Associates, 1980.

[SEYM86] Seymour, J. "Advise and Consent." PC Magazine, Vol. 5, No. 1, pp. 91-92, June 10, 1986.

[SHAW70] Shaw, H. Errors in English and Ways to Correct Them, Harper Row, New York, 1970.

[SLOC83] Slocum, J. "A Status Report on the LRC Machine Translation System." Proceedings Conference on Applied Natural Language Processing, Santa Monica, California, pp. 166-173, 1983.

[STRA49] Strathon, C. Guide to Correct English, McGraw-Hill Book Co., New York, 1949.

[STRU79] Strunk, W.; White, E. B. The Elements of Style, Macmillan Publishing Co., 1979.

[TOWN86] Townsend, C. Introduction to Turbo Prolog, Sybex, Alameda, California, 1986.

[VENO80] Venolia, J. Write Right, Periwinkle Press,

California, 1980.

[WATE86] Waterman, D. A. A Guide to Expert Systems, The Rand Corporation, 1986.

[WEIS80] Weischedel, R. M.; Black, J. E. "Responding Intelligently to Unparsable Inputs." American Journal of Computational Linguistics, Vol. 6, No. 2, pp. 97-109, April-June 1980.

[WHIT87] Whitney, M. A. "Combining Elegance and Readability: Walker Gibson's *Tough, Sweet, and Stuffy*" IEEE Transactions on Professional Communication, Vol. PC-30, No. 4, pp. 222-226, December 1987.

[WILK75] Wilks, Y. "An Intelligent Analyzer and Understander of English." Communications of the ACM, Vol. 18, No. 5, pp. 264-274, 1975.

[WINS84] Winston, P. H. Artificial Intelligence, 2nd ed., Addison Wesley, Massachusetts, 1984.

[ZUE85] Zue, V. W. "The Use of Speech Knowledge in Automatic Speech Recognition." Proceedings of the IEEE, Vol. 73, No. 11, pp. 1602-1615, November 1985.

Telephone Conversations

(unless otherwise noted)

1. Dr. Ronald Kaplan, Xerox Corporation, Palo Alto Research Center, Palo Alto, California, April 21, 1987.

2. Dr. Karen Jensen, IBM Corporation, Thomas J. Watson Research Center, Yorktown Heights, New York, April 21, 1987.

3. Dr. James Hillenbrand, RIT Research Corporation, Rochester, New York, personal conversation, May 20, 1987.

4. Ms. Lorinda L. Cherry, AT&T Bell Laboratories, Murry Hill, New Jersey, October 19, 1987.

5. Gannett Newspapers, April, 1987.

6. The Wall Street Journal, April, 1987.

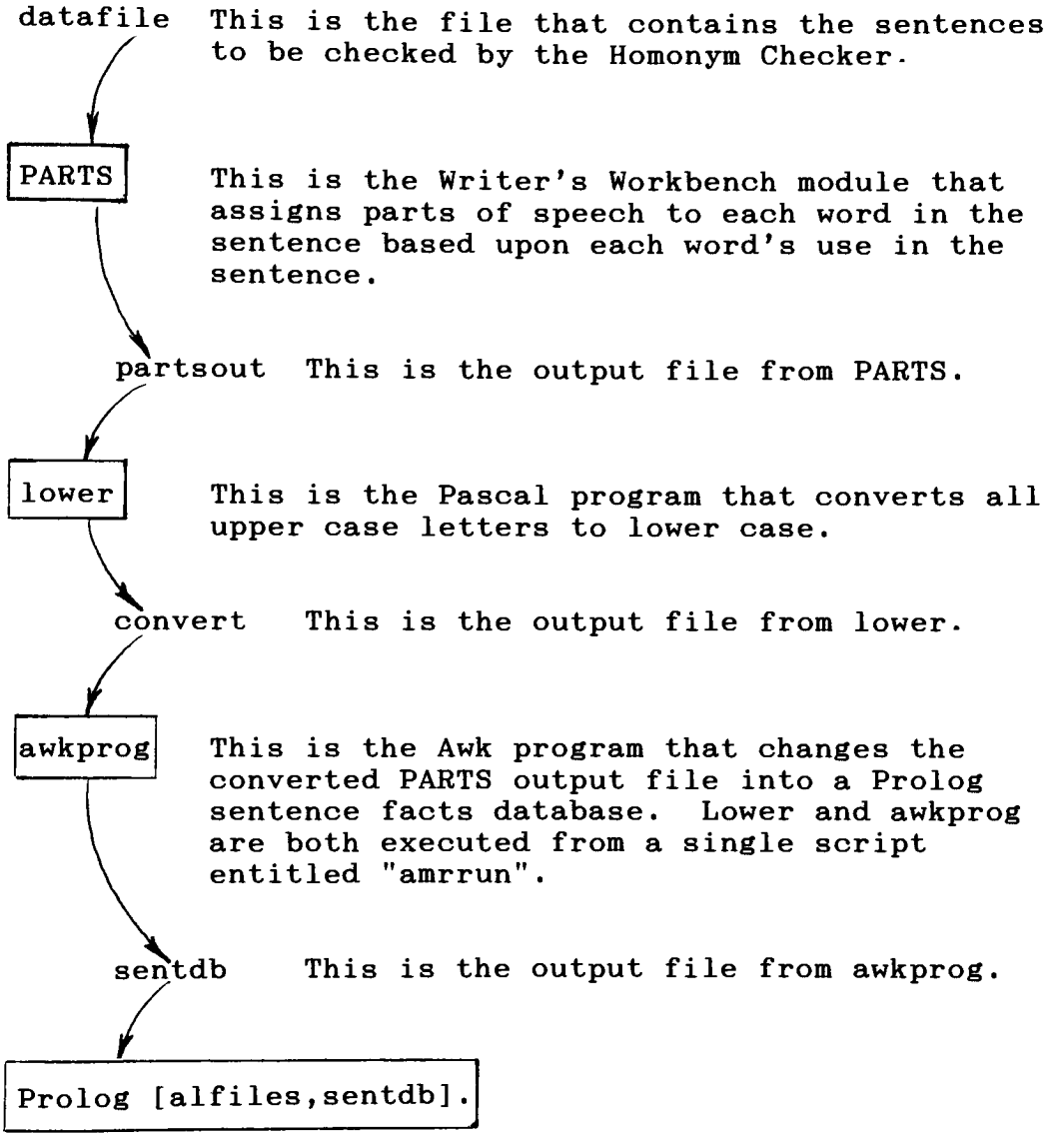
7. The New York Times, April, 1987.

APPENDIX

	Title	Description
Appendix A	FIGURE 1	Data Flow Diagram
	FIGURE 2	System Organization Chart
	FIGURE 3	Homonym Checker Structure Chart
Appendix B	RNETS	Homonym Database & Relational Network
Appendix C	SENTTEST	Test Sentences
	SENTPOEM	Poem
	SENTNEW1	General Text 1
	SENTNEW2	General Text 2
	SENTNEW3	Letter Text 3
Appendix D		Parts Output Samples
Appendix E		Prolog Input Samples
Appendix F		Homonym Checker Output Samples

FIG. 1

DATA FLOW DIAGRAM



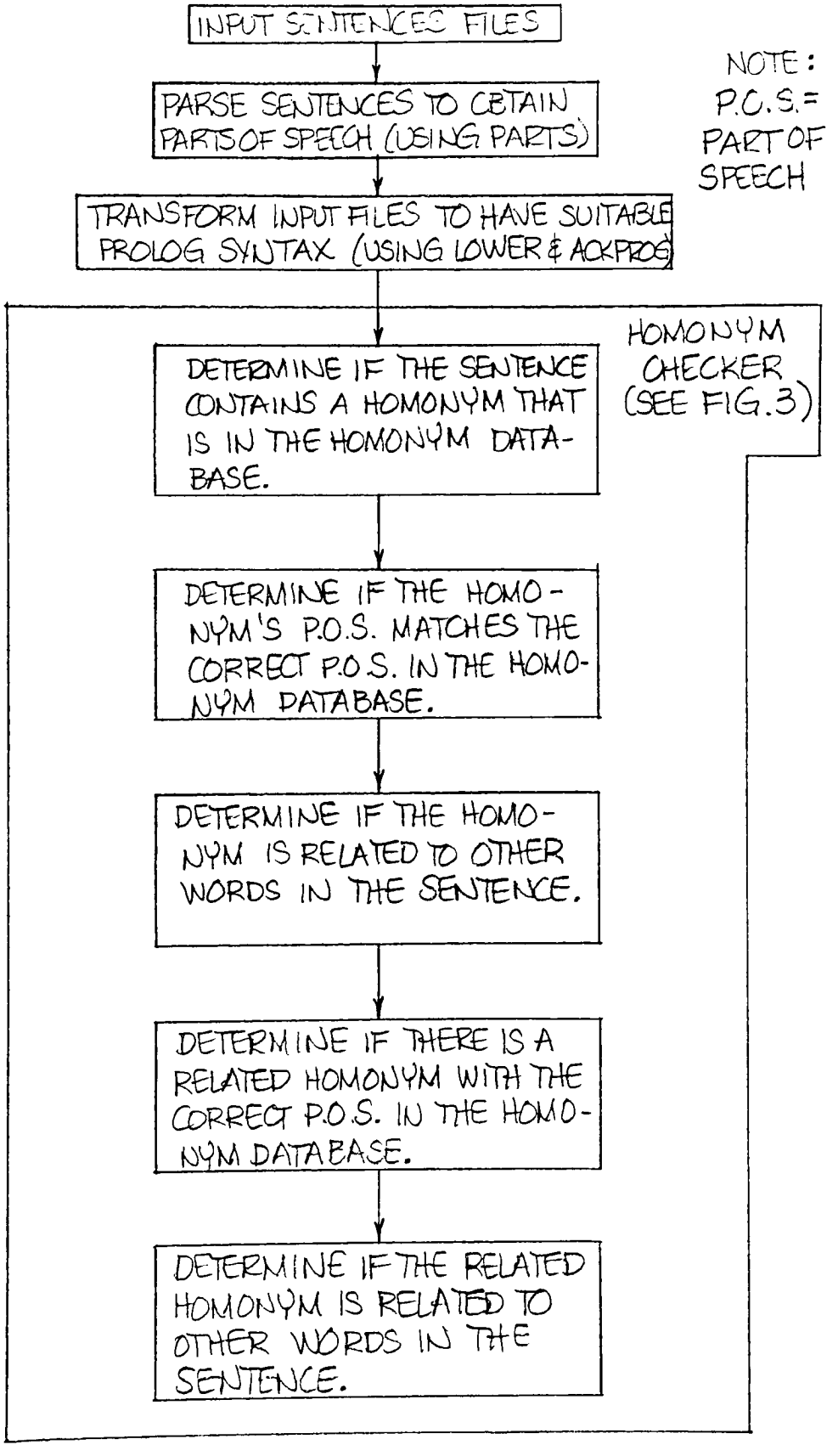
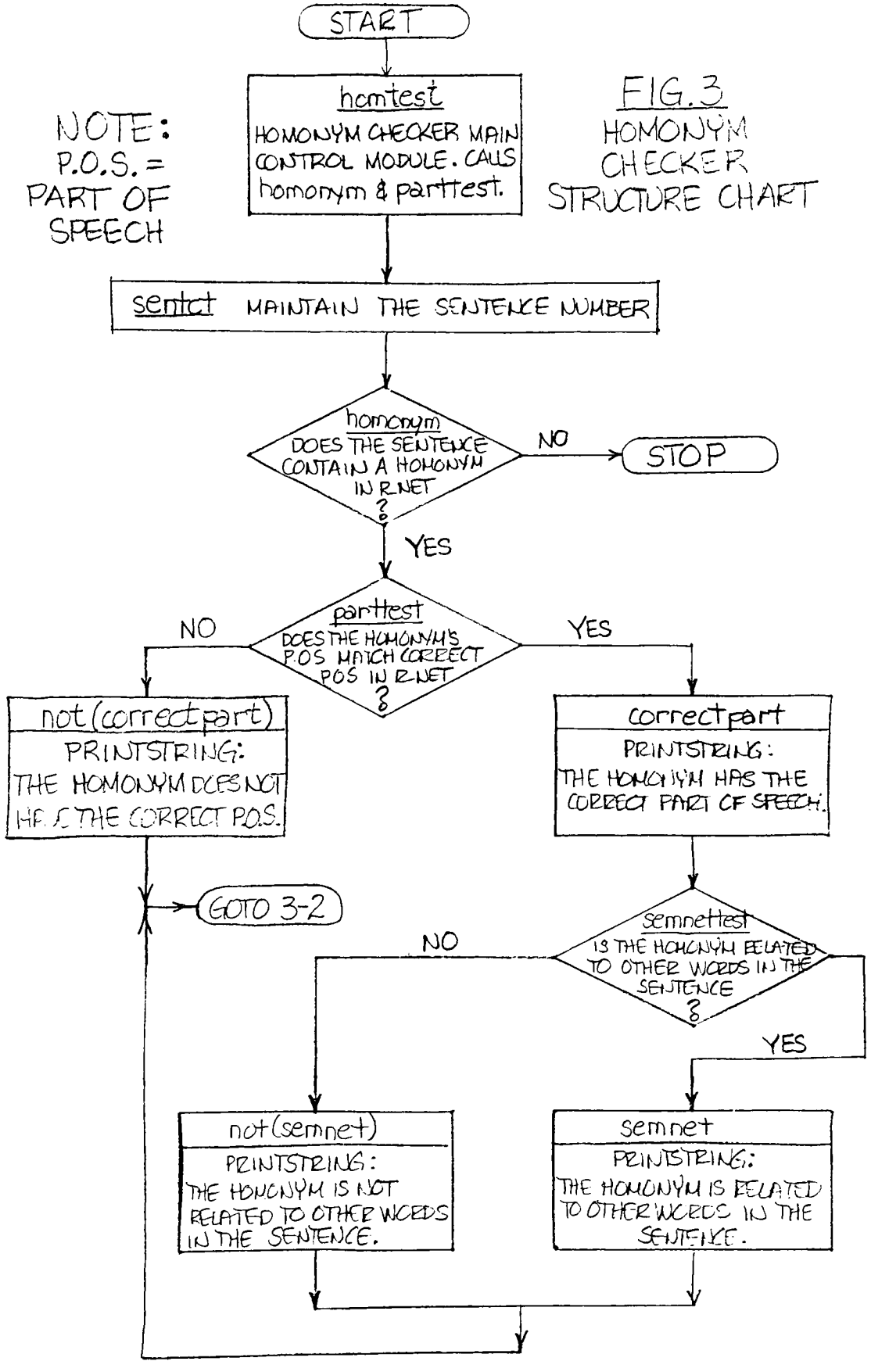


FIG. 2 SYSTEM ORGANIZATION CHART


START
homtest
HOMONYM CHECKER MAIN CONTROL MODULE. CALLS homonym & parttest.
NOTE: P.O.S. = PART OF SPEECH
FIG. 3
HOMONYM CHECKER STRUCTURE CHART
sentct MAINTAIN THE SENTENCE NUMBER
homonym
DOES THE SENTENCE CONTAIN A HOMONYM IN R-NET ?
NO
STOP
YES
parttest
DOES THE HOMONYM'S P.O.S MATCH CORRECT POS IN R-NET ?
NO
YES
not(correctpart)
PRINTSTRING:
THE HOMONYM DOES NOT HAVE THE CORRECT P.O.S.
correctpart
PRINTSTRING:
THE HOMONYM HAS THE CORRECT PART OF SPEECH.
GOTO 3-2
semnettest
IS THE HOMONYM RELATED TO OTHER WORDS IN THE SENTENCE ?
NO
YES
not(semnet)
PRINTSTRING:
THE HOMONYM IS NOT RELATED TO OTHER WORDS IN THE SENTENCE.
semnet
PRINTSTRING:
THE HOMONYM IS RELATED TO OTHER WORDS IN THE SENTENCE.

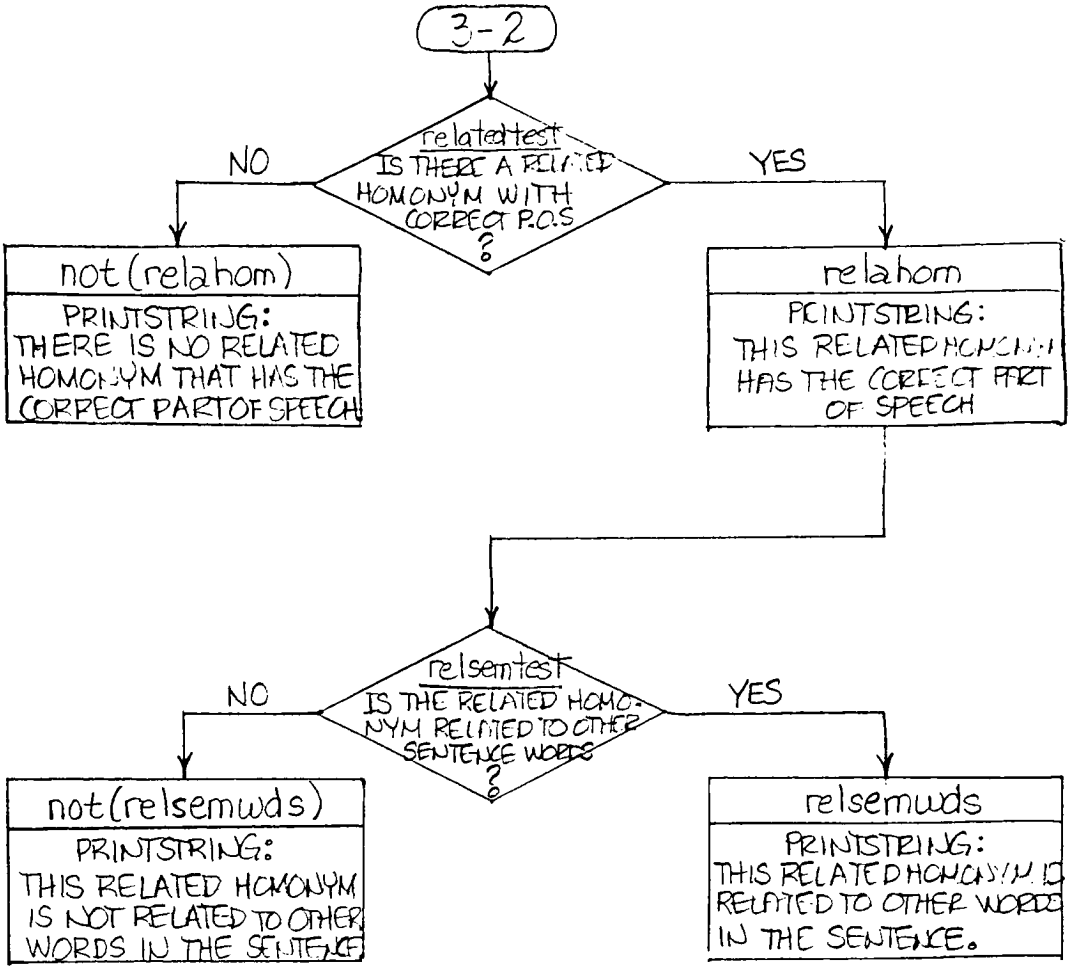


FIG. 3 (CONTINUED)
HOMONYM CHECKER
STRUCTURE CHART

```
r_net(beet,noun,[beat],[red,eat,vegetables]).
r_net(beat,verb,[beet],[drum,hit,noise]).
r_net(red,adj,[read],[]).
r_net(read,verb,[red],[book,see]).
r_net(read,adj,[red],[well,informed]).
r_net(would,aux,[wood],[]).
r_net(wood,noun,[would],[house]).
r_net(road,noun,[rode,rowed],[]).
r_net(rode,verb,[road,rowed],[bike,horse,train,wagon]).
r_net(rowed,verb,[road,rode],[boat]).
r_net(to,prep,[two,too],[]).
r_net(to,verb,[two,too],[]).
r_net(two,noun,[to,too],[]).
r_net(two,adj,[to,too],[]).
r_net(too,adv,[to,two],[]).
r_net(one,adj,[won],[]).
r_net(one,noun,[won],[]).
r_net(won,verb,[one],[]).
r_net(our,pos,[hour],[]).
r_net(hour,noun,[our],[finest]).
r_net(be,be,[bee],[]).
r_net(bee,noun,[be],[sting,bumble]).
r_net(i,pron,[eye],[]).
r_net(eye,noun,[i],[]).
r_net(steak,noun,[stake],[eat,ate,sauce]).
r_net(stake,noun,[steak],[wood,hammer]).
r_net(ate,verb,[eight],[steak,food,pear]).
r_net(eight,noun,[ate],[]).
r_net(see,verb,[sea],[eye]).
r_net(sea,adj,[see],[shore,shells]).
r_net(sea,noun,[see],[beach,sand,ocean]).
r_net(cents,noun,[sense],[penny, money,change]).
r_net(sense,noun,[cents],[common]).
r_net(find,verb,[fined],[]).
r_net(find,noun,[fined],[]).
r_net(fined,verb,[find],[ticket,money]).
r_net(new,adj,[knew],[old]).
r_net(knew,verb,[new],[]).
r_net(know,verb,[no],[]).
r_net(no,noun,[know],[]).
r_net(no,adj,[know],[]).
r_net(ad,noun,[add],[newspaper,magazine,television,tv]).
r_net(add,verb,[ad],[numbers,money,subtract]).
r_net(bare,adj,[bear],[]).
r_net(bear,noun,[bare],[animal]).
r_net(bear,verb,[bare],[]).
r_net(bite,verb,[byte],[]).
r_net(byte,noun,[bite],[computer,memory,kilo,k]).
r_net(board,noun,[bored],[wooden]).
r_net(board,verb,[bored],[]).
r_net(bored,adj,[board],[]).
r_net(brake,noun,[break],[step,engage,press]).
```

```
r_net(break,verb,[brake],[]).
r_net(break,noun,[brake],[]).
r_net(buy,verb,[bye,by],[]).
r_net(bye,noun,[buy,by],[say,good]).
r_net(by,prep,[buy,bye],[]).
r_net(capital,noun,[capitol],[city,depreciation]).
r_net(capital,adj,[capitol],[idea,punishment]).
r_net(capitol,noun,[capital],[building,statehouse]).
r_net(cent,noun,[scent,sent],[penny]).
r_net(scent,noun,[cent,sent],[odor,smell]).
r_net(sent,verb,[cent,scent],[]).
r_net(cereal,noun,[serial],[breakfast,eat,ate]).
r_net(serial,noun,[cereal],[show]).
r_net(coarse,adj,[course],[]).
r_net(course,noun,[coarse],[]).
r_net(days,noun,[daze],[]).
r_net(daze,noun,[days],[]).
r_net(dear,adj,[deer],[sir,mom,dad,madam,editor]).
r_net(deer,noun,[dear],[animal,doe,buck]).
r_net(dessert,noun,[desert],[]).
r_net(desert,noun,[dessert],[]).
r_net(desert,verb,[dessert],[]).
r_net(dew,noun,[due,do],[grass]).
r_net(due,adj,[dew,do],[owe,payable]).
r_net(do,verb,[dew,due],[]).
r_net(die,noun,[dye],[]).
r_net(die,verb,[dye],[]).
r_net(dye,verb,[die],[]).
r_net(dye,noun,[die],[]).
r_net(earn,verb,[urn],[living,money]).
r_net(urn,noun,[earn],[ashes]).
r_net(fir,noun,[fur],[tree,forest]).
r_net(fur,noun,[fir],[coat,jacket]).
r_net(for,prep,[four,fore],[]).
r_net(four,noun,[for,fore],[]).
r_net(fore,adj,[for,four],[]).
r_net(gate,noun,[gait],[wood,metal]).
r_net(gait,noun,[gate],[]).
r_net(grate,noun,[great],[]).
r_net(grate,verb,[great],[]).
r_net(great,adj,[grate],[]).
r_net(hair,noun,[hare],[cut]).
r_net(hare,noun,[hair],[animal]).
r_net(hear,verb,[here],[]).
r_net(here,adv,[hear],[]).
r_net(herd,noun,[heard],[]).
r_net(heard,verb,[herd],[]).
r_net(him,pron,[hymn],[]).
r_net(hymn,noun,[him],[prayer]).
r_net(night,noun,[knight],[day,moon,stars,black,dark]).
r_net(knight,noun,[night],[round,table,shining,armor]).
r_net(lead,verb,[led],[]).
```

```
r_net(lead,noun,[led],[]).
r_net(led,verb,[lead],[]).
r_net(lean,verb,[lien],[]).
r_net(lean,adj,[lien],[]).
r_net(lien,noun,[lean],[]).
r_net(loose,adj,[lose],[]).
r_net(loose,adv,[lose],[]).
r_net(lose,verb,[loose],[]).
r_net(mail,noun,[male],[man,letter]).
r_net(male,adj,[mail],[]).
r_net(more,adj,[moor],[]).
r_net(moor,noun,[more],[]).
r_net(moor,verb,[more],[]).
r_net(or,conj,[ore,oar],[]).
r_net(ore,noun,[or,oar],[mineral,iron]).
r_net(oar,noun,[or,ore],[boat,row]).
r_net(pain,noun,[pane],[back,neck]).
r_net(pane,noun,[pain],[window]).
r_net(pare,verb,[pair,pear],[vegatables]).
r_net(pair,noun,[pare,pear],[shoes,cards]).
r_net(pear,noun,[pare,pair],[eat,ate,green,spoiled]).
r_net(patience,noun,[patients],[]).
r_net(patients,noun,[patience],[]).
r_net(piece,noun,[peace],[meal]).
r_net(peace,noun,[piece],[war]).
r_net(peel,noun,[peal],[potatoes]).
r_net(peal,noun,[peel],[]).
r_net(pole,noun,[poll],[flag]).
r_net(poll,noun,[pole],[political]).
r_net(poll,verb,[pole],[]).
r_net(president,noun,[precedent],[]).
r_net(precedent,noun,[president],[]).
r_net(precedent,adj,[president],[]).
r_net(principle,noun,[principal],[]).
r_net(principal,noun,[principle],[school]).
r_net(principal,adj,[principle],[]).
r_net(rite,noun,[right,write,wright],[religious]).
r_net(right,adj,[rite,write,wright],[turn,left]).
r_net(write,verb,[rite,right,wright],[letter,page,thesis]).
r_net(wright,noun,[rite,write,right],[]).
r_net(sail,noun,[sale],[boat]).
r_net(sale,noun,[sail],[store,item]).
r_net(sow,verb,[sew,so],[wild,oats]).
r_net(sow,noun,[sew,so],[]).
r_net(sew,verb,[sow,so],[needle]).
r_net(so,adv,[sow,sew],[]).
r_net(so,noun,[sow,sew],[]).
r_net(sign,noun,[sine],[stop,street]).
r_net(sign,adj,[sine],[language]).
r_net(sign,verb,[sine],[]).
r_net(sine,noun,[sign],[]).
r_net(stair,noun,[stare],[]).
```

```
r_net(stare,verb,[stair],[]).
r_net(tacks,noun,[tax],[hammer]).
r_net(tacks,verb,[tax],[]).
r_net(tax,noun,[tacks],[income]).
r_net(tax,verb,[tacks],[]).
r_net(their,pos,[there],[]).
r_net(there,pron,[their],[]).
r_net(threw,verb,[through],[]).
r_net(through,prep,[threw],[]).
r_net(vain,adj,[vein,vane],[]).
r_net(vein,noun,[vain,vane],[]).
r_net(vane,noun,[vain,vein],[]).
r_net(ware,noun,[wear,where],[]).
r_net(wear,verb,[ware,where],[clothes,suit,jacket,coat]).
r_net(where,adv,[ware,wear],[]).
r_net(weak,adj,[week],[]).
r_net(week,noun,[weak],[month,day]).
r_net(weather,noun,[whether],[rainy,cold,snowy,sunny,hot]).
r_net(whether,conj,[weather],[]).
r_net(which,pron,[witch],[]).
r_net(which,adj,[witch],[]).
r_net(witch,noun,[which],[old,spell,evil,good]).
r_net(in,prep,[inn],[]).
r_net(inn,noun,[in],[sleep,stay]).
```

I ate the steak.
Eye eight the stake.
The bear is a brown animal.
I ate the green pair.
John ate the green pare.
Joe ate the green pear.
"This is a test to see What Parts does with Caps and,
punctuation."
This is a test file.
I rode the red bike.
I road the bike.
This is our finest hour.
She sells see shells by the sea shore.
Which witch cast the spell.
Make a right turn at the stop sign.
Write a letter to the principle.
Your income tax is due in April.
Press the tacks into the board.
She has four cents for you.
He is my knight in shining armour.
The bare was bear, he had no hare.

SENTPOEM

Wood you believe that I didn't no
About homophones until too daze ago?
That day in hour class in groups of for,
We had to come up with won or more.

Mary new six; enough to pass,
but my ate homophones lead the class.
Then a thought ran threw my head,
"Urn a living from homophones", it said.

I guess I just sat and staired into space.
My hole life seamed to fall into place.
Our school's principle happened to come buy,
and ask about the look in my eye.

"Sir", said I as bowled as could bee,
"My future rode I clearly see."
"Sun", said he, "move write ahead,
Set sail on your coarse. Don't be misled."

I herd that gnus with grate delight.
I will study homophones both day and knight.
For weaks and months, through thick oar thin,
I'll pursue my goal. Eye no aisle win.

Florence Ann Felton does not monkey around when there is a baby beast in need at her husband's zoo, she takes the critters home. Felton, whose husband, George, is the director of Greater Baton Rouge Zoo, can't bear it when the animals become orphaned or rejected by their natural mothers. "I love them," she said of her furry patients. "The babies, though, are exotic and wild. They do not make good pets." The Feltons live near the zoo in a home that has been modified for easy care of the baby animals.

SENTNEW2

Elisabeth Ann Zinser, who resigned as president of Gallaudet University amid protests from students who want a deaf leader, enraged students and faculty at the school for hearing impaired by suggesting she would quickly learn sign language. In truth, say professors, learning sign language, widely considered the nation's third most frequently used "foreign" language, is as difficult as mastering Russian or Chinese.

RPO should try service

For Christmas, our children thought they were giving us the ultimate in gifts with gift certificates to the Rochester Philharmonic Orchestra. We were thrilled. We love music and frequently spend our own money on concerts and other cultural events.

My husband called the Eastman box office in an attempt to reserve some seats for a recent performance. He was told that the only way to reserve seats by phone was to order them using a credit card.

If we wish to use our gift certificates, a personal appearance at the box office is required. (We live in Brockport.) To compound the irony, I received a phone call from a delightful young woman on behalf of the RPO soliciting my membership support. What do you think I told her?

We've been reading sad, soul searching articles about the failure of the Monroe County citizen to support his marvelous RPO. Somehow the blame is always placed on this uncultured, gauche group of clods who don't understand what the marvelous Phil is trying to accomplish.

Perhaps it's time for some self criticism in the area of service. We read how service is a growth industry. The meaning of the word has been lost.

When businesses (and the RPO) make a change, it's seldom for the convenience of the customer.

The RPO is dependent for much of its survival on the warmth, good feelings and generosity of many people. A lovely voice on the telephone isn't enough when people are being treated badly in other ways.

Jean Brooks Brockport

```
parts -m m -i g -w senttest
pron    I
verb    ate
art     the
noun    steak
.       .
noun    eye
verb    eight
art     the
noun    stake
.       .
art     the
noun    bear
be      is
art     a
adj     brown
noun    animal
.       .
pron    I
verb    ate
art     the
adj     green
noun    pair
.       .
noun    john
verb    ate
art     the
adj     green
noun    pare
.       .
noun    joe
verb    ate
art     the
adj     green
noun    pear
.       .
"       "
pron    this
be      is
art     a
noun    test
verb    to
verb    see
subcj   What
noun    Parts
verb    does
prep    with
noun    Caps
conj    and
,       ,
noun    punctuation
.       ."
```

```
parts -m m -i g -w sentpoem
noun    wood
pron    you
verb    believe
subcj   that
pron    I
verb    did
adv     n't
adj     no
prep    About
noun    homophones
adv     until
adv     too
adj     daze
adj     ago
.       ?
adj     that
noun    day
prep    in
adj     hour
noun    class
prep    in
noun    groups
adv     of
prep    for
,       ,
pron    We
verb    had
prep    to
adj     come
adv     up
prep    with
adj     won
conj    or
pron    more
.       .
adj     mary
noun    new
verb    six
;       ;
pron    enough
verb    to
verb    pass
,       ,
conj    but
pos     my
adj     ate
noun    homophones
verb    lead
art     the
noun    class
.       .
```

```
parts -m m -i g -w new1
adj     florence
adj     Ann
noun    Felton
aux     does
adv     not
verb    monkey
prep    around
subcj   when
pron    there
be      is
art     a
adj     baby
noun    beast
prep    in
noun    need
prep    at
pos     her
pos     husband's
noun    zoo
,       ,
pron    she
verb    takes
art     the
noun    critters
noun    home
.       .
noun    felton
,       ,
pron    whose
noun    husband
,       ,
noun    George
,       ,
be      is
art     the
noun    director
prep    of
adj     Greater
adj     Baton
adj     Rouge
noun    Zoo
,       ,
aux     can
adv     n't
verb    bear
pron    it
subcj   when
art     the
noun    animals
verb    become
adj     orphaned
```

```
parts -m m -i g -w new2
adj     elisabeth
adj     Ann
noun    Zinser
,       ,
pron    who
verb    resigned
prep    as
noun    president
prep    of
adj     Gallaudet
noun    University
prep    amid
noun    protests
prep    from
noun    students
pron    who
verb    want
art     a
adj     deaf
noun    leader
,       ,
adj     enraged
noun    students
conj    and
noun    faculty
prep    at
art     the
noun    school
subcj   for
noun    hearing
verb    impaired
prep    by
noun    suggesting
pron    she
aux     would
adv     quickly
verb    learn
adj     sign
noun    language
.       .
```

```
parts -m m -i g -w new3
noun    rpo
aux     should
verb    try
noun    service
prep    For
noun    Christmas
,       ,
pos     our
noun    children
adj     thought
pron    they
be      were
verb    giving
pron    us
art     the
noun    ultimate
prep    in
noun    gifts
prep    with
adj     gift
noun    certificates
prep    to
art     the
adj     Rochester
adj     Philharmonic
noun    Orchestra
.       .
pron    we
be      were
verb    thrilled
.       .
pron    we
verb    love
noun    music
conj    and
adv     frequently
adj     spend
pos     our
adj     own
noun    money
prep    on
noun    concerts
conj    and
adj     other
adj     cultural
noun    events
.       .
```

```
sentct(1).
sent(1,i,pron).
sent(1,ate,verb).
sent(1,the,art).
sent(1,steak,noun).
sentct(2).
sent(2,eye,noun).
sent(2,eight,verb).
sent(2,the,art).
sent(2,stake,noun).
sentct(3).
sent(3,the,art).
sent(3,bear,noun).
sent(3,is,be).
sent(3,a,art).
sent(3,brown,adj).
sent(3,animal,noun).
sentct(4).
sent(4,i,pron).
sent(4,ate,verb).
sent(4,the,art).
sent(4,green,adj).
sent(4,pair,noun).
sentct(5).
sent(5,john,noun).
sent(5,ate,verb).
sent(5,the,art).
sent(5,green,adj).
sent(5,pare,noun).
sentct(6).
sent(6,joe,noun).
sent(6,ate,verb).
sent(6,the,art).
sent(6,green,adj).
sent(6,pear,noun).
sentct(7).
sent(7,this,pron).
sent(7,is,be).
sent(7,a,art).
sent(7,test,noun).
sent(7,to,verb).
sent(7,see,verb).
sent(7,what,subcj).
sent(7,parts,noun).
sent(7,does,verb).
sent(7,with,prep).
sent(7,caps,noun).
sent(7,and,conj).
sent(7,punctuation,noun).
```

```
sentct(1).
sent(1,wood,noun).
sent(1,you,pron).
sent(1,believe,verb).
sent(1,that,subcj).
sent(1,i,pron).
sent(1,did,verb).

sent(1,no,adj).
sent(1,about,prep).
sent(1,homophones,noun).
sent(1,until,adv).
sent(1,too,adv).
sent(1,daze,adj).
sent(1,ago,adj).
sentct(2).
sent(2,that,adj).
sent(2,day,noun).
sent(2,in,prep).
sent(2,hour,adj).
sent(2,class,noun).
sent(2,in,prep).
sent(2,groups,noun).
sent(2,of,adv).
sent(2,for,prep).
sent(2,we,pron).
sent(2,had,verb).
sent(2,to,prep).
sent(2,come,adj).
sent(2,up,adv).
sent(2,with,prep).
sent(2,won,adj).
sent(2,or,conj).
sent(2,more,pron).
sentct(3).
sent(3,mary,adj).
sent(3,new,noun).
sent(3,six,verb).
sent(3,enough,pron).
sent(3,to,verb).
sent(3,pass,verb).
sent(3,but,conj).
sent(3,my,pos).
sent(3,ate,adj).
sent(3,homophones,noun).
sent(3,lead,verb).
sent(3,the,art).
sent(3,class,noun).
```

```
sentct(1).
sent(1,florence,adj).
sent(1,ann,adj).
sent(1,felton,noun).
sent(1,does,aux).
sent(1,not,adv).
sent(1,monkey,verb).
sent(1,around,prep).
sent(1,when,subcj).
sent(1,there,pron).
sent(1,is,be).
sent(1,a,art).
sent(1,baby,adj).
sent(1,beast,noun).
sent(1,in,prep).
sent(1,need,noun).
sent(1,at,prep).
sent(1,her,pos).

sent(1,zoo,noun).
sent(1,she,pron).
sent(1,takes,verb).
sent(1,the,art).
sent(1,critters,noun).
sent(1,home,noun).
sentct(2).
sent(2,felton,noun).
sent(2,whose,pron).
sent(2,husband,noun).
sent(2,george,noun).
sent(2,is,be).
sent(2,the,art).
sent(2,director,noun).
sent(2,of,prep).
sent(2,greater,adj).
sent(2,baton,adj).
sent(2,rouge,adj).
sent(2,zoo,noun).
sent(2,can,aux).

sent(2,bear,verb).
sent(2,it,pron).
sent(2,when,subcj).
sent(2,the,art).
sent(2,animals,noun).
sent(2,become,verb).
sent(2,orphaned,adj).
sent(2,or,conj).
sent(2,rejected,adj).
sent(2,by,prep).
sent(2,their,pos).
sent(2,natural,adj).
sent(2,mothers,noun).
```

```
sentct(1).
sent(1,elisabeth,adj).
sent(1,ann,adj).
sent(1,zinser,noun).
sent(1,who,pron).
sent(1,resigned,verb).
sent(1,as,prep).
sent(1,president,noun).
sent(1,of,prep).
sent(1,gallaudet,adj).
sent(1,university,noun).
sent(1,amid,prep).
sent(1,protests,noun).
sent(1,from,prep).
sent(1,students,noun).
sent(1,who,pron).
sent(1,want,verb).
sent(1,a,art).
sent(1,deaf,adj).
sent(1,leader,noun).
sent(1,enraged,adj).
sent(1,students,noun).
sent(1,and,conj).
sent(1,faculty,noun).
sent(1,at,prep).
sent(1,the,art).
sent(1,school,noun).
sent(1,for,subcj).
sent(1,hearing,noun).
sent(1,impaired,verb).
sent(1,by,prep).
sent(1,suggesting,noun).
sent(1,she,pron).
sent(1,would,aux).
sent(1,quickly,adv).
sent(1,learn,verb).
sent(1,sign,adj).
sent(1,language,noun).
```

```
sentct(1).
sent(1,rpo,noun).
sent(1,should,aux).
sent(1,try,verb).
sent(1,service,noun).
sent(1,for,prep).
sent(1,christmas,noun).
sent(1,our,pos).
sent(1,children,noun).
sent(1,thought,adj).
sent(1,they,pron).
sent(1,were,be).
sent(1,giving,verb).
sent(1,us,pron).
sent(1,the,art).
sent(1,ultimate,noun).
sent(1,in,prep).
sent(1,gifts,noun).
sent(1,with,prep).
sent(1,gift,adj).
sent(1,certificates,noun).
sent(1,to,prep).
sent(1,the,art).
sent(1,rochester,adj).
sent(1,philharmonic,adj).
sent(1,orchestra,noun).
sentct(2).
sent(2,we,pron).
sent(2,were,be).
sent(2,thrilled,verb).
sentct(3).
sent(3,we,pron).
sent(3,love,verb).
sent(3,music,noun).
sent(3,and,conj).
sent(3,frequently,adv).
sent(3,spend,adj).
sent(3,our,pos).
sent(3,own,adj).
sent(3,money,noun).
sent(3,on,prep).
sent(3,concerts,noun).
sent(3,and,conj).
sent(3,other,adj).
sent(3,cultural,adj).
sent(3,events,noun).
```

```
C-Prolog version 1.4
| ?- [alfiles,sttest].
homfd consulted 7716 bytes 0.6 sec.
amrtools consulted 212 bytes 0.0666667 sec.
rnets consulted 12304 bytes 2.91667 sec.
alfiles consulted 20232 bytes 3.61666 sec.
sttest consulted 5584 bytes 2 sec.

yes
| ?- homtest(Sentno,Hom,Part,Relhom,Relword).

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 1
Hom = i
Part = pron
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 1
Hom = ate
Part = verb
Relhom = _3
Relword = steak ;

The homonym has the correct part of speech.
The homonym is related to other words in the sentence.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 1
Hom = steak
Part = noun
Relhom = stake
Relword = ate ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 2
Hom = eye
Part = noun
Relhom = _3
Relword = _4 ;
```

* * * * * *

```
The homonym has the correct part of speech.
The homonym is related to other words in the sentence.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 6
Hom = pear
Part = noun
Relhom = pair
Relword = ate ;
The homonym is related to other words in the sentence.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 6
Hom = pear
Part = noun
Relhom = pair
Relword = green ;

The homonym does NOT have the correct part of speech.
There is NO related homonym that has the correct part of speech.
Sentno = 7
Hom = to
Part = _2
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 7
Hom = see
Part = verb
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 9
Hom = i
Part = pron
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is related to other words in the sentence.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 9
Hom = rode
Part = verb
Relhom = rowed
Relword = bike ;
```

```
The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 9
Hom = red
Part = adj
Relhom = read
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 10
Hom = i
Part = pron
Relhom = _3
Relword = _4 ;

The homonym does NOT have the correct part of speech.
This related homonym has the correct part of speech.
This related homonym is related to other words in the sentence.
Sentno = 10
Hom = road
Part = verb
Relhom = rode
Relword = bike ;
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 10
Hom = road
Part = verb
Relhom = rowed
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 11
Hom = our
Part = pos
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 11
Hom = hour
Part = noun
Relhom = _3
Relword = finest ;
```

* * * * * *

```
The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 13
Hom = which
Part = adj
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 13
Hom = which
Part = adj
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 13
Hom = witch
Part = noun
Relhom = _3
Relword = spell ;

The homonym does NOT have the correct part of speech.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 14
Hom = right
Part = noun
Relhom = rite
Relword = _4 ;
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 14
Hom = right
Part = noun
Relhom = wright
Relword = _4 ;
```

* * * * * *

```
The homonym does NOT have the correct part of speech.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 18
Hom = four
Part = adj
Relhom = fore
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 18
Hom = cents
Part = noun
Relhom = sense
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 18
Hom = for
Part = prep
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is related to other words in the sentence.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 19
Hom = knight
Part = noun
Relhom = night
Relword = shining ;

The homonym does NOT have the correct part of speech.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 20
Hom = bare
Part = noun
Relhom = bear
Relword = _4 ;
```

```
                    * * * * * *
The homonym does NOT have the correct part of speech.
There is NO related homonym that has the correct part of speech.
Sentno = 1
Hom = daze
Part = _2
Relhom = _3
Relword = _4 ;

The homonym does NOT have the correct part of speech.
There is NO related homonym that has the correct part of speech.
Sentno = 2
Hom = hour
Part = _2
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 2
Hom = for
Part = prep
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 2
Hom = to
Part = prep
Relhom = _3
Relword = _4 ;

The homonym does NOT have the correct part of speech.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 2
Hom = won
Part = adj
Relhom = one
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 2
Hom = or
Part = conj
Relhom = _3
Relword = _4 ;
                    * * * * * *
```

```
The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 8
Hom = i
Part = pron
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 8
Hom = i
Part = pron
Relhom = _3
Relword = _4 ;

The homonym does NOT have the correct part of speech.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 8
Hom = see
Part = noun
Relhom = sea
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 9
Hom = write
Part = verb
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 9
Hom = sail
Part = noun
Relhom = sale
Relword = _4 ;

The homonym does NOT have the correct part of speech.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 9
Hom = coarse
Part = noun
Relhom = course
Relword = _4 ;
```

* * * * * *

```
The homonym does NOT have the correct part of speech.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 11
Hom = grate
Part = adj
Relhom = great
Relword = _4 ;
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 11
Hom = grate
Part = adj
Relhom = great
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 12
Hom = i
Part = pron
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
This related homonym has the correct part of speech.
This related homonym is related to other words in the sentence.
Sentno = 12
Hom = knight
Part = noun
Relhom = night
Relword = day ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 13
Hom = for
Part = prep
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 13
Hom = through
Part = prep
Relhom = _3
Relword = _4 ;
```

```
C-Prolog version 1.4
| ?- [alfiles,stnew1].
homfd consulted 7716 bytes 0.6 sec.
amrtools consulted 212 bytes 0.1 sec.
rnets consulted 12496 bytes 2.96667 sec.
alfiles consulted 20424 bytes 3.73333 sec.
stnew1 consulted 3896 bytes 1.38333 sec.

yes
| ?- homtest(Sentno,Hom,Part,Relhom,Relword).

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 1
Hom = there
Part = pron
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 1
Hom = in
Part = prep
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 2
Hom = bear
Part = verb
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 2
Hom = bear
Part = verb
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 2
Hom = or
Part = conj
Relhom = _3
Relword = _4 ;
```

```
C-Prolog version 1.4
| ?- [alfiles,stnew2].
homfd consulted 7716 bytes 0.65 sec.
amrtools consulted 212 bytes 0.0833335 sec.
rnets consulted 12496 bytes 2.86667 sec.
alfiles consulted 20424 bytes 3.68333 sec.
stnew2 consulted 2568 bytes 0.916667 sec.

yes
| ?- [alfiles,          homtest(Sentno,Hom,Part,Relhom,Relword).

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
This related homonym has the correct part of speech.
This related homonym is NOT related to other words in the sentence.
Sentno = 1
Hom = president
Part = noun
Relhom = precedent
Relword = _4 ;

The homonym does NOT have the correct part of speech.
There is NO related homonym that has the correct part of speech.
Sentno = 1
Hom = for
Part = _2
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 1
Hom = by
Part = prep
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 1
Hom = would
Part = aux
Relhom = _3
Relword = _4 ;
```

```
C-Prolog version 1.4
| ?- [alfiles,stnew3].
homfd consulted 7716 bytes 0.633333 sec.
amrtools consulted 212 bytes 0.0833337 sec.
rnets consulted 12564 bytes 2.91667 sec.
alfiles consulted 20492 bytes 3.73333 sec.
stnew3 consulted 10968 bytes 3.86667 sec.

yes
| ?- homtest(Sentno,Hom,Part,Relhom,Relword).

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 1
Hom = for
Part = prep
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 1
Hom = our
Part = pos
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 1
Hom = in
Part = prep
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 1
Hom = to
Part = prep
Relhom = _3
Relword = _4 ;

The homonym has the correct part of speech.
The homonym is NOT related to other words in the sentence.
There is NO related homonym that has the correct part of speech.
Sentno = 1
Hom = to
Part = prep
Relhom = _3
Relword = _4 ;
```