USING NATURALLY OCCURRING TEXTS AS A KNOWLEDGE ACQUISITION RESOURCE FOR KNOWLEDGE BASE DESIGN: DEVELOPING A KNOWLEDGE BASE TAXONOMY ON MICROPROCESSORS

by

Michael F. Emero

Project and Report submitted to the Faculty of the

Virginia Polytechnic Institute and State University

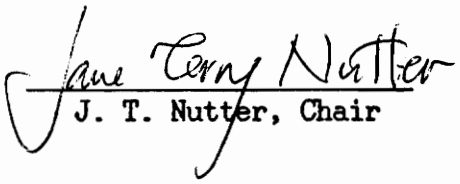
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

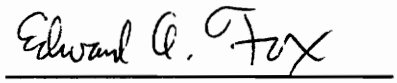
in

Computer Science

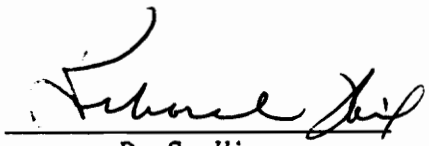
APPROVED:



J. T. Nutter, Chair



E. A. Fox



D. S. Hix

August, 1992

Blacksburg, Virginia

USING NATURALLY OCCURRING TEXTS AS A KNOWLEDGE ACQUISITION RESOURCE FOR KNOWLEDGE BASE DESIGN: DEVELOPING A KNOWLEDGE BASE TAXONOMY ON MICROPROCESSORS

by

Michael F. Emero

Committee Chair: J. Terry Nutter
Computer Science

(ABSTRACT)

Many artificial intelligence applications suffer severely from a bottleneck in acquiring domain information necessary to go beyond toy hand-built demonstrations to realistic applications. This project examines one approach to reducing that bottleneck by using automated and semi-automated techniques to analyze published domain-relevant material.

A taxonomy of terms related to computers with an emphasis on microprocessors is developed and presented. The methods used are experimental and not yet fully validated, but are potentially of great use for extracting useful domain information from published material. Preliminary validation by comparison with a published taxonomy shows that these methods have produced a taxonomy which is better suited for the immediate use of this taxonomy.

Acknowledgments

I would like to thank my advisor and mentor Dr. Terry Nutter for her guidance and patience. I thank my friend and source of inspiration Ben Cline and wish him success with his natural language generation system. I am grateful to my wife and daughters for their support and sacrifices over the past two years. Thanks go to Dr. Deborah Hix and Dr. Ed Fox for their focused attention when asked for it. I thank Captain Kari Everett, Professional Development Officer for the United States Army Signal Corps, for her invaluable moral support and administrative assistance.

Table of Contents

Table of Contents (continued)

List of Figures

List of Figures (continued)

Chapter 1 Introduction

1.1 Problem Statement

Recent studies have shown that people tend to relate to real world ideas through natural taxonomies [Berlin 73] [Rosch et al. 76] [Rosch 78] [Jolicoeur 84] [Reiter 91]. Unlike taxonomies found in most traditional AI systems, natural taxonomies are not uniform. Instead, they have a distinguished basic level with several important features. Their terms are most frequently used by adult speakers and seem to have the largest number of associated attributes [Rosch et al 76]. Using natural taxonomies as opposed to uniform hierarchies provides a number of advantages for natural language systems. Some such benefits have been shown for understanding systems [Peters & Shapiro 87] [Peters, Shapiro & Rappaport 88]. Text generation systems that store and use knowledge structured in such natural taxonomies may benefit from similar advantages [Cline & Nutter 90] [Cline 91].

For large scale applications, however, it is not clear how to acquire the knowledge that would be reflected by such a natural taxonomy. For toy domains, constructing an *ad hoc* taxonomy might suffice for initial results. However, to examine seriously these claims of advantages, we need substantial knowledge bases that accurately reflect some real phenomena. This inevitably raises the problem of knowledge acquisition.

1.2 Solution Approach

The purpose of this project is to build a knowledge base under the assumption that the following hypotheses are true.

o Natural hierarchies influence the structure of existing texts. If researchers are right about the importance of natural hierarchies for computer understanding and generation, we should be able to see the effects also in published discourse. An analysis of a couple of textbooks, which for purposes of this project are assumed to be novice level texts. This analysis has been performed to determine word frequency and selectively analyze schema use, following patterns in [McKeown 85].

o Within a taxonomy, the hierarchy level identified as basic differs between those texts intended for novices and those intended for experts. Texts geared towards expert readers tend to include hierarchically lower level basic level terms than those used in novice texts.

o The structural effects referred to in the first two hypotheses can be used to extract hierarchy information in complex domains from existing texts, including distinguishing between the expert and novice basic levels. If published discourse materials are structured in ways that reflect the structure of natural taxonomies, we can use them to extract such taxonomies by an analysis approach as outlined in the first two hypotheses. This closes the knowledge acquisition gap.

This project has resulted in construction of a domain-specific semantic network knowledge base compatible with the currently developing

KALOS natural language generation system [Cline 91], under the assumption of the hypotheses noted above. The taxonomy has sufficient detail and scope to be of functional use for the KALOS project. The data for the taxonomy result from manual and automatic analysis of two computer organization textbooks, assumed by this author to be novice level due to the nature of textbooks, and two microprocessor technical manuals, assumed by this author to be expert level due to the nature of the intended audience.

The analysis involves two basic steps. First, I have determined patterns of word frequency that signal use of basic level terms. Second, I selectively analyzed their positions relative to understood discourse schemata. Comparing results of this process on different texts, I have looked for variations in basic level terms between the two types of texts to mark basic level terms which appear in the knowledge base as expert or novice. The complete sequence of steps involved in this project are listed in Figure 1.1.

This methodology is experimental. Preliminary validation by comparison with a published taxonomy on computer science shows that the taxonomy resulting from this project is much more microprocessor specific than the published one. More complete validation of results will occur when the KALOS natural language generation system [Cline 91] is fully developed and text is actually generated.

1.3 Report Organization

Chapter two contains background on natural taxonomies, the

1. Select 2 textbooks and 2 microprocessor manuals as sources.

2. Obtain copyright permission to optically scan and store sources.

3. Scan sources on an OCR and store text in ASCII data files.

4. Write/run a computer program to extract and count the frequency of occurrence of unique words in each source.

5. Manually filter stop words (verbs, conjunctions).

6. Manually identify noun phrases in the sources.

7. Write/run a computer program to extract and count the frequency of occurrence of unique nouns and noun phrases (as candidates for inclusion into a taxonomy) in each source.

8. Develop a taxonomy by manually finding semantic relationships between nouns and noun phrases in the sources which link nouns and noun phrases hierarchically. Compile a list of attributes for things as they are noticed during the search for hierarchical relationships.

9. Separate synonyms (including acronyms and abbreviations), parts of things, and members of sets from the list of nouns and noun phrases.

10. Write/run a computer program which computes the sample mean and sample standard deviation of the taxonomy terms in each source.

11. Identify basic level terms as those terms in each source category which occur more often than the sum of the sample mean and sample standard deviation.

12. Write/run a computer program which computes the coefficient of rank correlation between the frequency of occurrence of taxonomy terms in each source. This number indicates the likelihood of the frequency of use of taxonomy terms in one source to indicate the frequency of use of the same terms in another source.

13. Compare developed taxonomy with a published taxonomy on computer science.

14. Write Semantic Network Processing System User Language (SNePSUL) commands to build the domain knowledge base for the KALOS system [Cline 91].

Figure 1.1: Steps in Project Methodology

knowledge base, and our representation. The third chapter discusses materials and methods used for this project. Chapter four examines the taxonomy of terms related to microprocessors which was derived in this project. The fifth chapter examines additional lists if information which were deemed necessary to supplement information in the taxonomy. Chapter six discusses the effect of tying several lists of information together in the semantic network. The last chapter contains a self-critique, a look at the usefulness of the results obtained in this project, and proposed further work of interest.

Chapter 2 Background

2.1 Taxonomies

The world is comprised of many classes or categories of entities which relate in some cognitively meaningful manner to an individual. A category is a named group of similar objects. A hierarchy of categories related by class inclusion is called a taxonomy.

Anyone who has taken a high school biology course has studied at least a subset of the taxonomy of living things. Human beings are mammals, therefore 'human being' is a subclass of 'mammal'. Dogs and cats are also mammals, and they are likewise subclasses of 'mammal'. The taxonomy entity relationship provides for inheritance of attributes from a superclass to a subclass. At the subclass level, however, additional attributes may occur, as may exceptions to one or more attributes of a higher class. These additions and exceptions are what distinguish members of one category from those of another.

The animal kingdom includes animals that are not mammals, such as reptiles. Thus a snake, which is a reptile, can be compared with mammals such as human beings by their common superclass 'animal.' The world of living things includes weeds, which are plants. It would normally be inappropriate and fruitless to attempt to compare a human being to a snake by discussing weeds, since the category 'weeds' is outside of the smallest hierarchical subtree connecting human beings and snakes. Figure 2.1 illustrates a partial taxonomy of living things using the biological classes discussed here.

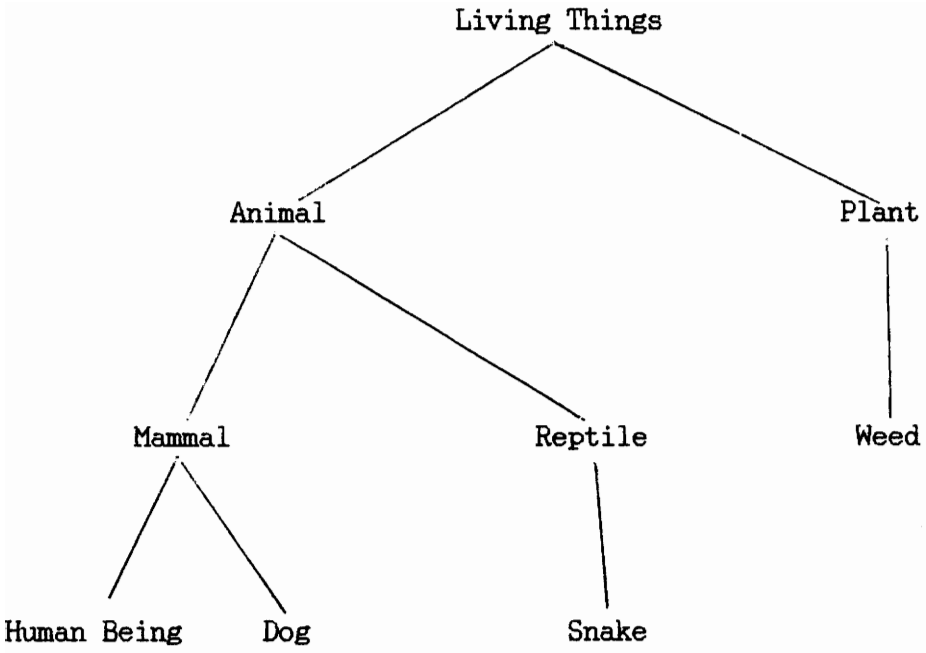


Figure 2.1: Partial Taxonomy of Living Things

A taxonomy of terms relating to computers with an emphasis on microprocessors has been compiled in this project. The terms within the hierarchy are understood by people to stand in meaningful relationships to each other. This taxonomy can be used by the deep generation component of the KALOS text generation system [Cline 91] as a source of real world information.

2.2 Basic Level Categories

Research in cognitive psychology has shown that people tend to associate related classes of objects at distinctive places within a taxonomy [Berlin 73] [Rosch et al. 76] [Rosch 78] [Jolicoeur 84] [Reiter 91]. These distinctive disjoint classes are called *basic level categories*. Entities within a basic level category will tend to have significantly more non-inherited attributes in common than members of a superclass. An example is the class ´toasters´, which is immediately recognizable by people. The superclass ´small kitchen appliances´ does not bring a clearly recognizable picture of a typical small kitchen appliance to the mind of most people. Entities within a class subordinate to the basic level will tend to have few additional attributes to the basic level attributes. A typical four-slice toaster does not differ greatly from a typical two-slice toaster. Research has also suggested that the word frequency of basic level terms in natural language is much higher than the word frequency of higher or lower category terms.

Empirical studies have shown that people will generally first

categorize an entity at the basic level category, although that entity can usually accurately be described using either a higher class term or a lower one. The primary exception to this rule occurs with atypical members of a class subordinate to a basic level category. In this case, most people will first identify the atypical entity with a class subordinate to the basic level [Rosch et al 76] [Jolicoeur 84]. An example of an atypical member of the class of birds is a penguin, which does not fly.

Research has also shown that especially in technological domains, domain experts use different basic level terms than those used by domain novices. For domain experts, the basic level occurs lower in the taxonomy due to their having the ability to differentiate classes with greater detail than most people. Domain experts´ basic level terms are hence lower in abstraction than novice basic level terms. It has been surmised that a person´s expertise is normally confined to localized parts of a taxonomy rather than an entire taxonomy [Rosch et al 76]. An example is an airplane mechanic whose basic level categories are lower classes in the airplane hierarchy than the single basic level category ´airplane´ which most people recognize [Rosch et al 76, 430].

2.3 Knowledge Bases

Varying forms of knowledge representation have been used by text generation systems. A tedious but simple form of representation is enumerated language phrases or sentences which may be found is restricted question and answer systems. This form of representation

attaches no meaning to the sentences and so is feasible only in extremely limited applications.

A slightly more flexible form of knowledge representation is with templates combined with enumerated possibilities for slot contents. This form of representation still attaches little meaning for the generation system to use in making decisions, and is tedious to compile. An example of such a system is MYCIN [Shortliffe 76] as described in [McKeown & Swartout 87].

Some generation systems have used underlying relational databases (see, for example, McKeown's TEXT system, [McKeown 85]). However, relational databases are known to have profound representational limitations.

Recent developments in database technology have provided for more sophisticated knowledge representation. The entity-relationship model can be a powerful mechanism for relating different entities to each other. This model is not always appropriate for certain types of knowledge and, more applicably, it does not interrelate with the lexical and linguistic knowledge of a text generation system [Cline & Nutter 90].

This project has been an attempt to construct a knowledge base of sufficient breadth to be usable by a text generation system for the non-trivial domain of microprocessors with some general information about computers included as well. The same knowledge base could also be used for language understanding or other problem solving tasks. To achieve generality, we have chosen a standard AI representational scheme, SNePS,

which is discussed in the next section.

2.4 Semantic Networks

A semantic network is a means of knowledge representation which connects nodes by directed arcs. A specific node can be related to another specific node by one or more arcs which directly connect the two nodes. An example using animal categories is a relationship between the category node ´mammal´ and the category node ´human being´. In a semantic network these two nodes could be directly connected with the directed arcs ´is a´ and ´includes´. The category ´mammal´ ´includes´ the category ´human being´ and the category ´human being´ ´is a´ category ´mammal´. Nodes can also be indirectly connected with directed arcs via one or more intermediate nodes. Using the animal taxonomy again, the category ´human being´ can be related to the category ´dog´. A ´human being´ ´is a´ ´mammal´ and a ´dog´ ´is a´ ´mammal´ as well. Figure 2.2 illustrates a network including these animal categories.

We have chosen to represent our taxonomy using the Semantic Network Processing System (SNePS) paradigm [Shapiro 92] for a number of reasons. First, it is rich enough to support all the forms of representation KALOS needs [Cline 91]. Second, it is a generally available software package that obviates the need to construct a new semantic network processor. However, the information extracted in this project is independent of the representation chosen. The same techniques could be applied with any sufficiently powerful target representation for the hierarchy.

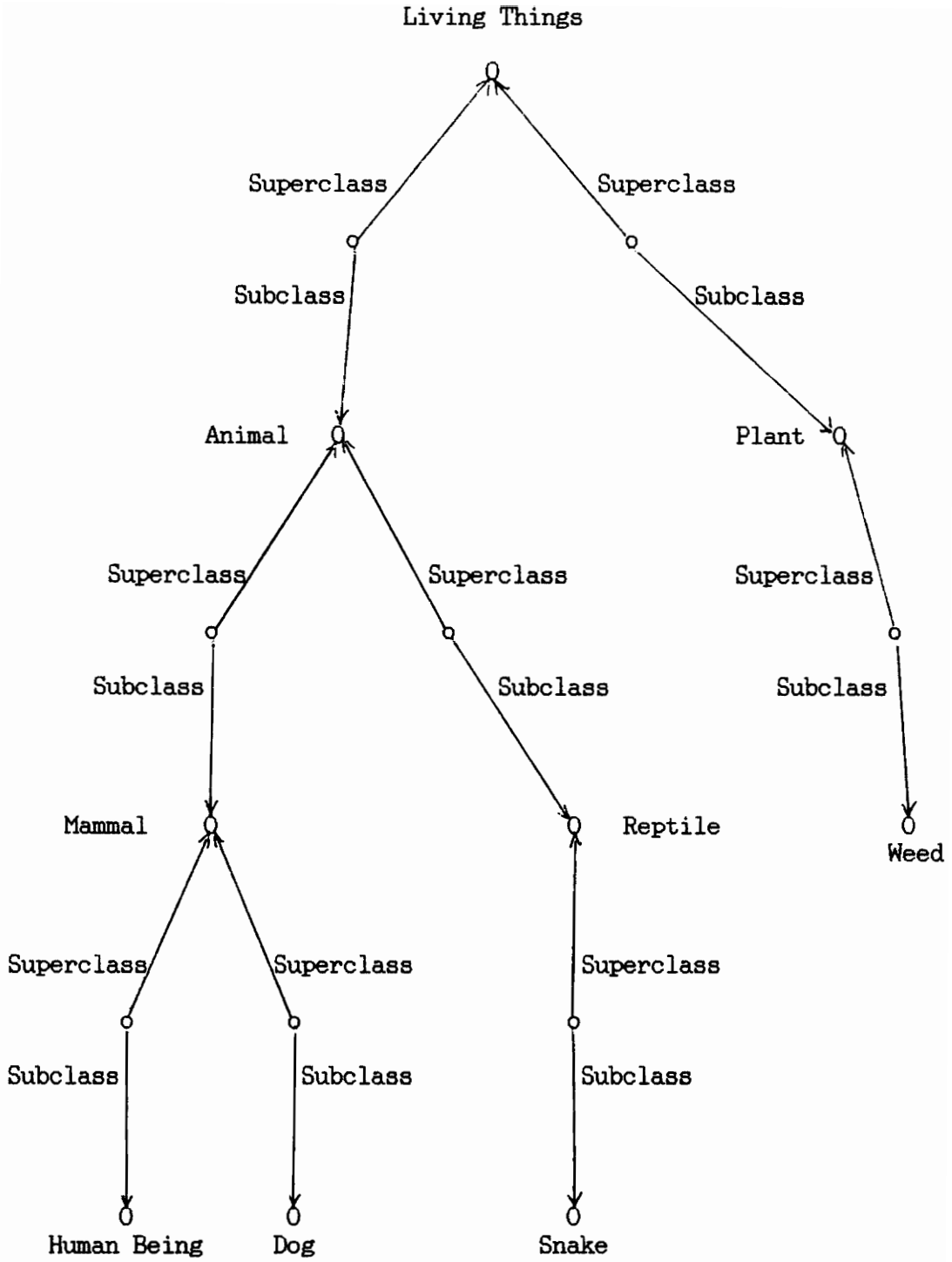


Figure 2.2: Sample Network for Partial Taxonomy of Living Things

Chapter 3 Materials and Methods

3.1 Source Material

Four published works about computer organization and microprocessors were used as source material for the knowledge base. Two of the sources were textbooks (hence referred to as textbook source 1 and textbook source 2), and the others were manuals specific to a microprocessor (hence referred to as manual source 1 and manual source 2). Textbook source 1 is *Structured Computer Organization* by Andrew S. Tanenbaum. Copyright permission to electronically scan and store this text was denied by Prentice-Hall, Inc. Textbook source 2 is *Computer Organization: Hardware/Software* by George W. Gorsline. Copyright permission was granted by Prentice-Hall, Inc. Manual source 1 is *MC68010 16-Bit Virtual Memory Microprocessor*. Copyright permission was granted by Motorola Semiconductors, Inc. Manual source 2 is *68000 Processor Handbook* by Gerry Kane. Copyright permission was granted by Osborne/McGraw Hill. A table of source material characteristics is shown in Figure 3.1.

The materials within each category were analyzed to determine the relative similarity of frequencies of occurrence of terms between the sources by their coefficient of rank correlation. This correlation is obtained when two sets of data (in this case, frequencies of term occurrence) are ranked in some arbitrary order such that the I_{th} item in each list represents a data value for an equivalent event (in this case, the same term). The coefficient of rank correlation coefficient

Source	Number of Pages	Number of Words	Author
Manual 1	96	19,000	Motorola, Inc.
Manual 2	115	16,000	Gerry Kane
Textbook 1	424	129,000	Andrew Tanenbaum
Textbook 2	605	154,000	George Gorsline

Figure 3.1: Source Material Characteristics

analysis of the sources is discussed in this section to demonstrate that the sources are legitimately categorized in two categories.

The results from application of Spearman´s formula for rank correlation, which is presented in section 3.3, show whether or not the values contained in one set of data tends to indicate similar magnitudes in the other set. The selected materials within the textbook category were found to have a rank of correlation coefficient of 0.337 and the selections within the microprocessor manual category had a rank of correlation coefficient of 0.883. The positive coefficients (which cannot be greater than 1.0) indicate the close correlation of the frequency of occurrence of terms used within each category, and that the sources within each category genuinely share the same category of expertise. The higher coefficient for the manuals is attributable to the narrow focus domain of a microprocessor compared to the broad area of computer organization.

The individual sources within each source category were compared with each of the sources of the other category. The resulting coefficients of rank correlation were far less than zero. The large negative coefficients indicate that the frequency of occurrence of the terms in one category significantly fail to correlate with the other category, and that the sources within one category genuinely do not share the same expertise level of those sources in the other category. Figure 3.2 lists all of the rank of correlation coefficients.

Source A	Source B	Rank of Correlation Coefficient
Manual-1	Manual-2	0.883
Manual-1	Textbook-1	-20.562
Manual-1	Textbook-2	-55.036
Manual-2	Textbook-1	-33.513
Manual-2	Textbook-2	-81.228
Textbook-1	Textbook-2	0.337

Figure 3.2: Rank of Correlation Coefficients

3.2 Optical Scanning

With the publishers' written permission, both microprocessor manuals and one textbook were optically scanned and stored in ASCII data files. The files of scanned material allowed the development of experimental computer programs to perform selective analysis on the files. The scanner used for this project, a Xerox/Kurzweil Personal Reader, tends to produce erroneous data wherever the source materials changes type font size or boldness. The scanner also has occasional difficulty discerning the letter 'S' and the number '5', the letter 'I' and the number '1', as well as the letter 'O' and the number '0'. Because the microprocessor manuals are relatively short, making the scanning errors relatively more significant, a greater effort was made to edit those files than was made for the sources within the textbook category. The editing to reduce this noise was performed by this author by correcting obvious scanner errors using a word processor.

3.3 Taxonomy Construction Algorithm

The method used in this project to elicit taxonomy terms and structural relationships from the source references are very simple and straightforward. Extracting a set of domain-related nouns and noun phrases from the source materials is a simple process. This set is initially modeled as a set of disjoint points in space, from which a directed graph will be built.

A semantic relationship between two arbitrary terms is established only when a corresponding relationship exists within the body of text of

a source reference. Key semantic links are ´is a´ and ´a type of X is Y´. Noun phrases indicate, by default, a hierarchical relationship with the root noun and sometimes with other noun phrases. Noun phrase relationships may be altered by specific class - subclass relationship descriptions found within the source body of text. As semantic relationships between terms (points in the model) are established, corresponding directed arcs are added to the model to represent each relationship.

Extracting information from multiple source references can lead to some taxonomy structure conflicts. For example, one source may show that an ´X´ is a ´Y´ and another source may show that ´X´ is a ´Z´. Two solutions to this type of conflict are readily apparent. One way to resolve the conflict is to determine if a relationship between ´Y´ and ´Z´ exists which would merge the relationships. Another way to resolve this type of conflict is to establish a source material preference rating scheme. If source ´A´ is deemed preferable to source ´B´, then all conflicts between the two sources should be resolved in favor of the relationships established by source ´A´. In a situation where a number of sources are used, the relationships used by a majority of them can be used to resolve a conflict.

The set of domain-related nouns and noun phrases will normally have a number of terms which are synonyms, acronyms, or abbreviations for other terms. These terms must be merged to be represented by only one point in the points in space model. Other nouns and noun phrases may be used only as attributes or parts of objects represented by other terms.

These terms normally should not appear in the resulting taxonomy. An algorithm outline for taxonomy construction is shown in Figure 3.3.

3.4 Statistical Methods

This project involved the use of a few very basic statistical procedures. To provide data, word concordances were determined. The counting of word frequency proceeded by simple automated word extraction, one word at a time, from the text files. These words were placed in an array upon their first appearance and a counter for each word was incremented upon subsequent appearance of each word.

Statistical procedures used in this project involved taking the sample means, sample standard deviations, and sample coefficients of rank correlation of the source materials. The sample means were computed by the familiar formula

$$\text{sample mean} = \frac{\sum_{i=1}^{n} \text{frequency(word } i)}{n}$$

Likewise the sample variances were derived by summing the squares of the differences of the frequency of each word and the sample mean; standard deviation is the square root of the variance.

$$\text{sample variance} = \frac{\sum_{i=1}^{n} (\text{frequency(word } i) - \text{sample mean})^2}{n}$$

1. Extract domain-relevant nouns from source references by filtering terms.

2. Extract noun phrases from source references. This can be done semi-automatically by automatically listing all phrases containing root nouns found in step 1 and manual filtering.

3. Consider the set of nouns and noun phrases as disjoint points in space.

4. In the points in space model, connect noun phrases to root nouns with directed arcs by default.

5. Analyze the source references for semantic relationships (such as 'X is a Y') between members of the set of nouns and noun phrases. Each semantic relationship found will connect the corresponding points in the model with a directed arc.

6. Systematically resolve any conflicts arising from steps 4 and 5.

7. Merge synonyms, eliminate terms used only as attributes, and eliminate terms used only as parts of objects from the points in space model.

8. The resulting directed graph is the taxonomy.

Figure 3.3: Taxonomy Construction Algorithm

$$\text{sample standard deviation} = \sqrt{\text{sample variance}}$$

The coefficients of rank correlation between source texts were computed using the *Spearman's formula for rank correlation* listed below. The purpose for which this formula was chosen was detailed in section 3.1. Individual word frequencies of the second data set in the formula (called source b) were adjusted by the ratio of the summed word frequencies of both files (represented by the expression size(a) over size(b) within the formula) to compensate for sample size differences.

$$\text{correlation coefficient(source a and source b)} = 1 - \frac{6 * \sum_{i=1}^{n} \left[\text{frequency(word i(a))} - \frac{\text{size(a)}}{\text{size(b)}} * \text{frequency(word i(b))}\right]^2}{n(n-1)^2}$$

3.5 Discourse Schemata

Rhetorical predicates are the means which a speaker has for describing information. They characterize the different types of predicating acts the speaker may use and they delineate the structural relations between propositions in a text. Some examples are analogy (the making of an analogy), constituency (description of sub-parts or sub-types), and attributive (providing detail about an entity or event).

Linguistic discussion of such predicates indicates that some combinations are preferable to others [McKeown 85, p. 20].

A discourse schema is a pattern of rhetorical predicates. A schema is used in natural language to perform a specific communications goal. There are four basic schemata which McKeown found in her analysis of samples of expository text paragraphs [McKeown 85]. They are the attributive schema, the identification schema, the constituency schema, and the compare and contrast schema. McKeown´s work was chosen to be used as a guide to schema form because her work has also been used during the development of Cline´s KALOS system [Cline 91]. The structure of each schema is shown in Figure 3.4.

The most direct hierarchical link between a superclass and a subclass is found in the identification predicate of the identification schema. Another use of basic level terms in the source materials used in this project occurred in the depth identification and particular illustration predicates of the constituency schema.

The basic level terms common to both categories of sources were manually examined for the type of discourse schemata in which they occurred in the source materials. The only use of the identification schema in the manuals was to identify the particular product as a microprocessor. The textbook sources did use the identification schema more than the manuals, but not very frequently. Basic level terms were found much more frequently in attribute predicates in the constituency schema.

Attributive Schema

```
Attributive
{Amplification; restriction}
Particular illustration*
{Representative}
{Question; problem, Answer} ¦ {Comparison; contrast, Adversative}
Amplification ¦ Explanation ¦ Inferences ¦ Comparison
```

Identification Schema

```
Identification (class & attribute ¦ function)
{Analogy ¦ Constituency ¦ Attributive ¦ Renaming ¦ Amplification}*
Particular illustration ¦ Evidence+
{Amplification ¦ Analogy ¦ Attributive}
{Particular illustration ¦ Evidence}
```

Constituency Schema

```
Constituency
Cause-effect* ¦ Attribute* ¦
    {  Depth identification ¦ Depth attributive
       {Particular illustration ¦ Evidence}
       {Comparison ¦ Analogy}
    }+
{Amplification ¦ Explanation ¦ Attributive ¦ Analogy}
```

Compare and Contrast Schema

```
Positing ¦ Attributive (not A)
{Attributive ¦ Evidence ¦ Amplification ¦ Inference ¦ Explanation (A)
}+
{Comparison ¦ Explanation ¦ Generalization ¦ Inference (A and not A)}+
```

Figure 3.4: Discourse Schemata Constituency of Predicates
[McKeown 85, pp. 27-31]

3.6 SNePS User Language

The semantic network used for this project is SNePS 2.1 [Shapiro 92]. The only SNePS User Language commands used in the construction of the knowledge base are the assert and define commands. The syntactic appearance of the commands is similar to LISP in that each command is enclosed in parenthesis. The syntax of SNePS requires that each node entry have the format:

(assert {arc-name node}*)

The arc-name determines what semantic relationship is established for the node. The arc-name used must be independently established by using the define command. The define command has the format: (define {arc-name}*). As a collection of SNePSUL commands constructs a network, any nodes which are named more than once acquire additional arcs. A sample collection of SNePSUL commands which would build the animal network discussed earlier, and was illustrated in Figure 2.2, is shown in Figure 3.5.

```
(define super sub)

(assert super living-thing sub animal)

(assert super living-thing sub plant)

(assert super animal sub mammal)

(assert super animal sub reptile)

(assert super mammal sub human-being)

(assert super mammal sub dog)

(assert super reptile sub snake)

(assert super plant sub weed)
```

Figure 3.5: Sample SNePSUL Commands
(To Build Partial Taxonomy of Living Things)

Chapter 4 Creating the Taxonomy

4.1 Choosing Terms

As is discussed in section 2.3, the structural representation of information in a knowledge base affects the manipulative power of a text generation system. The basic forms of knowledge base structure are listed in Figure 4.1. Although a taxonomy may be useful to incorporate within either a relational database or a semantic network, the latter has been chosen to use with this project to integrate with the other knowledge bases of the KALOS system [Cline 91].

A pool of possible candidates for inclusion in a taxonomy was collected by automatically counting the frequency of words in the microprocessor manuals and one textbook. The output of this automatic process was filtered to include only nouns because they were the most likely candidates for inclusion in a taxonomy. All four sources were then manually scanned to identify multiple word combination noun phrases in which the nouns output from the automatic process appeared. Noun phrases of up to six words in length were detected. Further concordances were then performed automatically to locate and compute frequencies for these phrases. The entire collection of nouns and noun phrases were manually cross referenced with the semantic meaning of selected passages in which they were found.

The hierarchical relationships between individual classes of the pool of taxonomy candidates were established with the set of noun phrases and each noun phrase's root noun. Examples of this type of

- Phrase/Sentence Enumeration
- Templates
- Relational Databases
- Semantic Networks

Figure 4.1: Knowledge Base Structures

relationship are the terms 'processor', 'I/O processor', 'central processor', and 'peripheral processor'.

More hierarchical relationships were established through semantic relationships found by manual analysis of selected passages where the taxonomy candidate terms appeared. An example is below.

"The effective operand address is the address in program memory of the desired datum." [Gorsline 86, p. 67]

Occasionally, semantic analysis of selected passages brought to light hierarchical relationships that countered the relationships implied by noun phrases. In these cases, the semantic relationship was used in place of the purely noun phrase relationship. In the following sample passage, the noun phrase peripheral processor, along with channels and DMA's, is shown to be a subclass of I/O processors.

"Almost all of the maxi- and super-computers possess separate I/O instructions that execute in parallel on separate I/O processors (channels, direct memory access (DMA), peripheral processors, etc.)." [Gorsline 86, p. 167]

Synonyms were manually compiled from obvious abbreviations and acronyms used in the sources, specifically labeled equivalent terms, and word changes for a subject in a sentence or sequential sentences within a paragraph. In the case of synonyms, the more common term (determined by overall frequency of occurrence) was selected to represent all terms. The resulting taxonomy contains all terms within the (ideal complete target) taxonomy that actually occur in at least one of the sources used, and which are not manufacturer specific.

4.2 Arranging the Terms

Based upon the combination of automatic and manual processes outlined in the previous section, the taxonomy was constructed as a collection of sixteen smaller hierarchies which are detailed in Figures 4.2, 4.3, 4.4a, and 4.4b. The hierarchies grew from the set of independent nouns and noun phrases which were hierarchically linked through noun phrases with the same root noun and by semantic relationships found in the source materials as outlined in the previous section.

The terms used in the taxonomy are those that are common to many computer and microprocessor architectures. Many terms used in the source materials which are not included in the taxonomy were deemed to be applicable only to a specific computer, microprocessor, or manufacturer. Terms of such limited application would be attached to the taxonomy in the semantic network knowledge base via a separate domain knowledge base which supplements the general domain knowledge base. This additional component of the overall domain knowledge base would include specific members of classes and unique subclass structures which apply to the specific microprocessor which will be the topic of the text generated by the KALOS system [Cline 91].

The hierarchies within the taxonomy are linked together by creation of additional lists for the knowledge base. These lists are discussed in the next chapter.

Figure 4.2 details the two-level hierarchies which were constructed

Superclass	Subclasses
digit	bit, decimal digit, hexadecimal digit, octal digit
data type	array, character, complex number, constant, integer, pointer, queue, real, signed integer, stack, string, structure, variable
field	destination address, opcode, operand, source address
logic size	binary digit, block, byte, double word, half word, gigabyte, kilobyte, megabyte, nibble, page, segment, word
machine level	assembly language, digital logic, machine code, operating system
software	applications program, assembler, compiler, interpreter, operating system
table	page table, segment table, symbol table, truth table, vector table
transmission mode	asynchronous, half-duplex, full-duplex, simplex, synchronous
addressing mode	direct addressing, indexed addressing, indirect addressing, relative addressing, virtual addressing

Figure 4.2: Two-Level Hierarchies

based upon analysis of the source materials. The three-level hierarchies are shown in Figure 4.3. One six-level hierarchy was constructed. It is detailed in Figures 4.4a and 4.4b. Again, the depth of each hierarchy is determined by chaining the class inclusions asserted through semantic relationships found within the source materials. Figure 4.5 illustrates the hierarchical relationship of some of the terms within the taxonomy.

4.3 Comparison with a Published Taxonomy

The taxonomy developed in this project was manually compared with the taxonomy of computer science in [Ashenhurst et al. 80]. There are a total of 249 terms which appear in this project's taxonomy. Of this total, 52 of the terms also appear in the published taxonomy in essentially the same lexical root form. The ratio of the number of common terms to the total size of our taxonomy is 20.8%. This low percentage of intersecting terms indicates that the two taxonomies are substantially dissimilar.

These two taxonomies have very different purposes, and therefore should not be very similar. The taxonomy developed in this project is focused on the microprocessor, whereas the published taxonomy is a broad look at the entire realm of computer science. This project's taxonomy has low level detail, therefore has many terms which are not in the published taxonomy. The 52:249 ratio of overlapping terms to total terms tells us that the 197 non-intersecting terms would not have been included in the knowledge base had the published taxonomy been used to

Superclass	Subclasses
state	
active state	active high, active low
process state	asleep, ready, running
processor state	supervisor state, user state
information	
address	effective operand address, logical address, physical address
data	
result	normal result, abnormal result
signal	
exception	
interrupt	hardware interrupt, software interrupt, external interrupt
machine	
computer	microcomputer, minicomputer, midicomputer, mainframe computer, supercomputer
network	
distributed network	
local-area network	ring network, star network
instruction	
arithmetic instruction	add instruction, divide instruction, multiply instruction
branch instruction	call instruction, return instruction, skip instruction, trap instruction
compare instruction	
i/o instruction	read instruction, write instruction
logical instruction	and instruction, exclusive-or instruction, or instruction
loop control instr.	
misc. instruction	
mode control instr.	
move instruction	fetch instruction, store instruction
privileged instr.	
rotate instruction	
shift instruction	
stack instruction	pop instruction, push instruction

Figure 4.3: Three-level Hierarchies

```
Superclass
      Subclasses


hardware
    battery, box, circuit board, connector, controller, power supply,
    switch
    cable
        wire, coaxial cable
    chip
        adder, comparator, decoder, demultiplexer, eeprom, encoder,
        eprom, lsi chip, msi chip, multiplexer, programmed logic array,
        prom, ram, rom, shifter, ssi chip, uart, usart, vlsi chip
        processor
            co-processor, floating-point processor
            central processor
                microprocessor
            i/o processor
                channel processor, peripheral processor, DMA processor
            parallel processor
                array processor
                pipeline processor
                    vector processor
    data path
        bus
            address bus, control bus, data bus
        channel
    electronic component
        capacitor, diode, flip-flop, fuse, resistor, transistor
        latch
            clocked D latch, clocked JK latch, clocked SR latch
    line
        control line, data line
    memory
        primary memory
            buffer, cache, core, eprom, eeprom, prom, rom, scratchpad,
            wrom
            ram
                static ram, dynamic ram
            register
                accumulator, address register, base address register,
                control register, device register, general purpose
                register, i/o register, index register, instruction
                register, interrupt register, memory address register,
                memory data register, page control register, program
                counter, program status word, stack pointer, valid area
                register
```

Figure 4.4a: Six-level Hierarchy

```
Superclass
    Subclasses


(hardware)
    (memory)

        secondary memory
            card, magnetic drum, magnetic tape, paper tape, photographic
            film, virtual memory
            disk
                disk pack, floppy disk, video disk
    peripheral
        card punch, card reader, keyboard, modem, monitor, paper tape
        punch, paper tape reader, plotter, printer, teletype
        secondary storage device
            magnetic drum, tape drive, video disk
            disk drive
                floppy disk drive
```

Figure 4.4b: Six-level Hierarchy (continued)

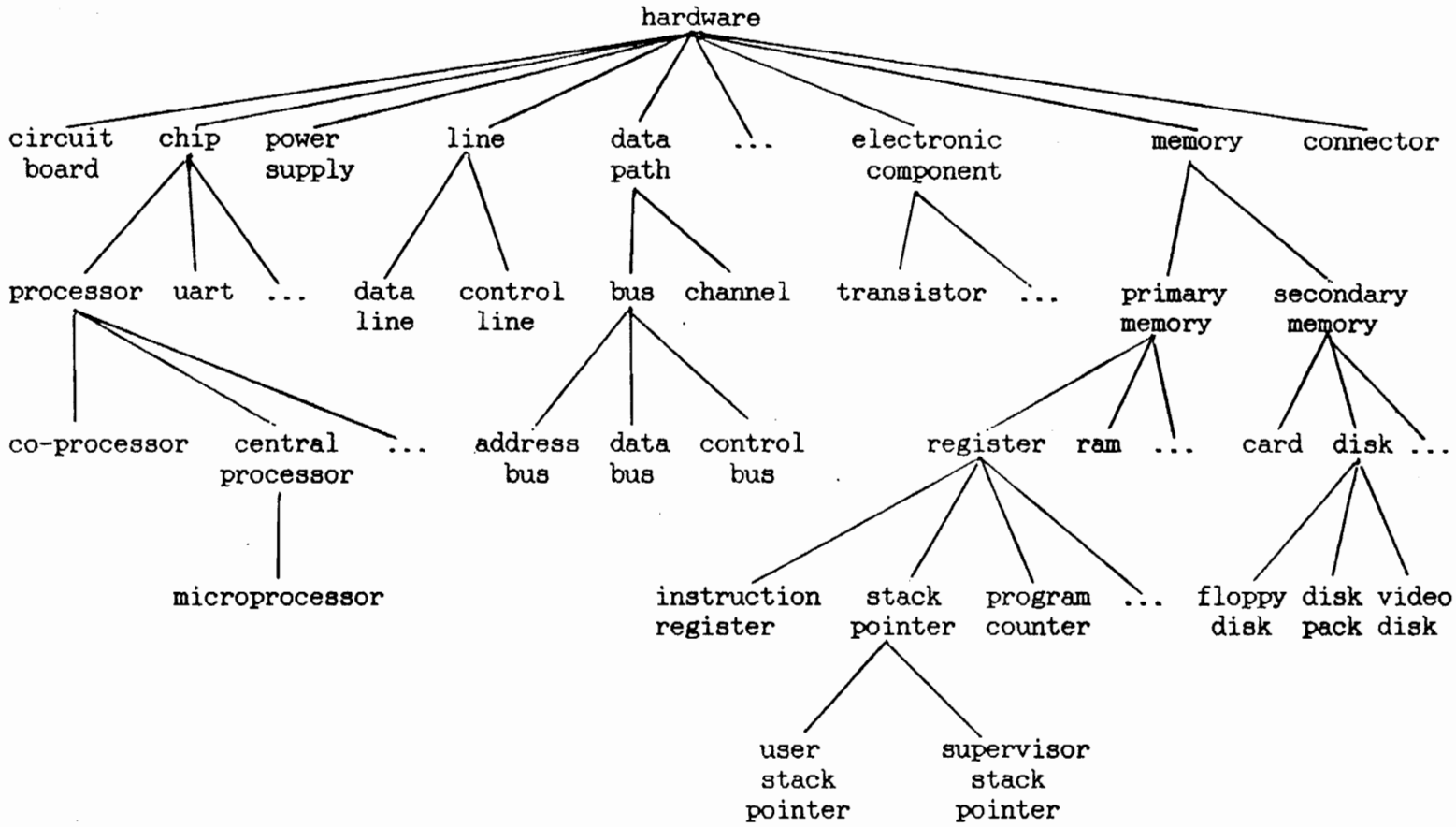


Figure 4.5: Sample Taxonomy Hierarchical Relationships

establish this knowledge.

An interesting characteristic of the 197 terms unique to our taxonomy is that over 81% of them are deemed by this author to be of importance to microprocessors. Terms such as those which relate to networks and peripherals are classified by this author as not being important to a microprocessor. Of the terms in the published taxonomy which are in relevant sections but do not overlap with this project's taxonomy, less than 30% of them are deemed by this author to possibly be of importance to a microprocessor. Our taxonomy is richer and more appropriate for use with the KALOS generation system than the published taxonomy.

Chapter 5 Additional Lists for the Knowledge Base

5.1 Basic Level Categories

Identification of basic level categories within a taxonomy allows a natural language generation system to select a level of abstraction to use to define, compare and contrast, or to illustrate a point about a new concept when first discussing this new concept [Cline & Nutter 90, p. 2]. To provide the deep text generation module of the KALOS system [Cline 91] with the salience of some categories over others, an effort was made to identify basic level terms. Since research in cognitive psychology ([Rosch et al. 76] [Rosch 78] [Jolicoeur 84] [Reiter 91]) has suggested that people tend to use basic level categories more often than other categories, the relative frequency of occurrence of category names was analyzed by computer program. The criteria used for differentiating the basic level terms from other terms used in each source was choosing the words whose frequency was at least one standard deviation over the sample mean. More sophisticated criteria have not been explored in this project.

A list of terms found in the hierarchies was established, and for each source used the frequency of occurrence of each term was noted. By independently analyzing the frequency of occurrence of terms within each source, the basic level terms for each source were identified. Lists of the words in each microprocessor manual category source that were above the numerical cutoff level (the sample mean plus one standard deviation) are shown in Figures 5.1 and 5.2. The high frequency words in each

Word	Frequency
instruction	341
interrupt	215
bus	169
processor	152
data	143
address	126
word	119
bit	113
register	101
exception	100
signal	89
operand	60
fetch instruction	50

Sample mean = 9.847
Sample standard deviation = 34.526
Sample mean + sample standard deviation = 44.374

Figure 5.1: High Frequency Taxonomy Terms in Manual Source 1

Word	Frequency
instruction	216
bit	182
register	144
signal	142
data	115
address	114
bus	111
exception	89
word	88
interrupt	79
byte	78
memory	70
state	58
address register	41
microprocessor	36
operand	36

Sample mean = 8.200
Sample standard deviation = 27.375
Sample mean + sample standard deviation = 35.576

Figure 5.2: High Frequency Taxonomy Terms in Manual Source 2

textbook category source are shown in Figures 5.3 and 5.4. Graphs of the number of taxonomy words which have word frequencies up to two hundred and fifty in each source are shown in Figures 5.5, 5.6, 5.7, and 5.8.

The basic level terms for each category of source material (textbooks and microprocessor manuals) were compiled by using the union of the list of basic level terms for each source within a category. The list of expert basic level terms was obtained from the microprocessor manual category of source material. The novice basic level terms were obtained from the textbook category. The list of expert basic level terms is shown in Figure 5.9, and the list of novice basic level terms is shown in Figure 5.10. Figure 5.11 lists the correlation by percent ratio of overlapping words of the basic level terms from each source to the total number of basic level terms identified for each source.

The authors of the source materials evidently assume that the terms identified as basic level during this project are previously understood by the reader. The first sentence of one of the textbook category sources, quoted below, contains three basic level terms which are underlined for identification.

"A stored digital computer is a device for manipulating information according to rules provided as an ordered set of instructions known as a program." [Gorsline 86, p. 1]

Here, only the basic level term 'computer' is given a definition

Word	Frequency
bit	1080
instruction	828
memory	658
address	623
machine	524
register	406
word	351
data	343
computer	312
line	212
microprogram	204
store instruction	200
segment	187
page	186
character	169

Sample mean = 48.136
Sample standard deviation = 119.241
Sample mean + sample standard deviation = 167.377

Figure 5.3: High Frequency Taxonomy Terms in Textbook Source 1

Word	Frequency
data	1066
memory	1059
instruction	1046
bit	960
computer	805
processor	541
register	485
information	440
address	431
operand	284
machine	239
result	225
data path	210

Sample mean = 56.055
Sample standard deviation = 153.031
Sample mean + sample standard deviation = 209.0868

Figure 5.4: High Frequency Taxonomy Terms in Textbook Source 2

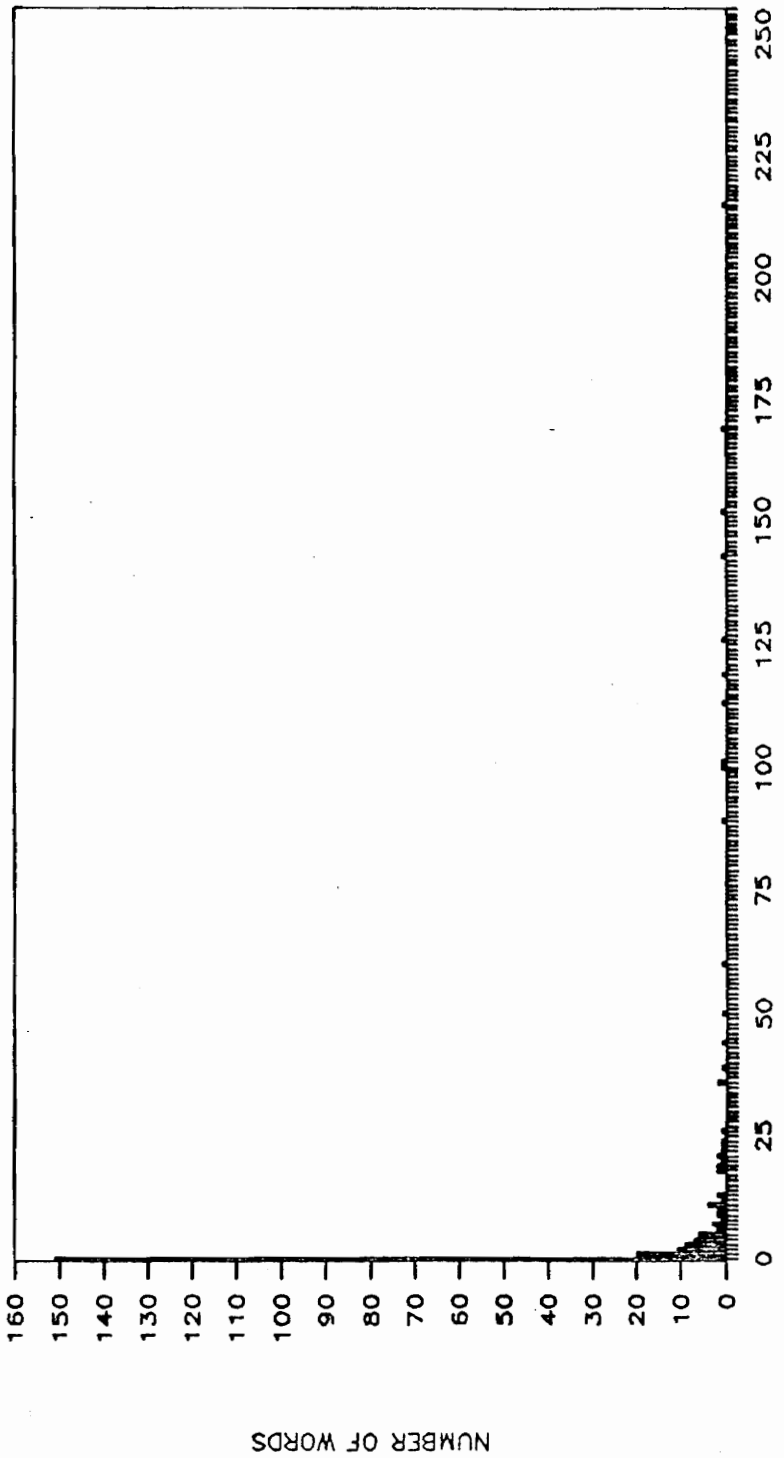


Figure 5.5 Graph of Number of Words With Word Frequency: Manual Source 1

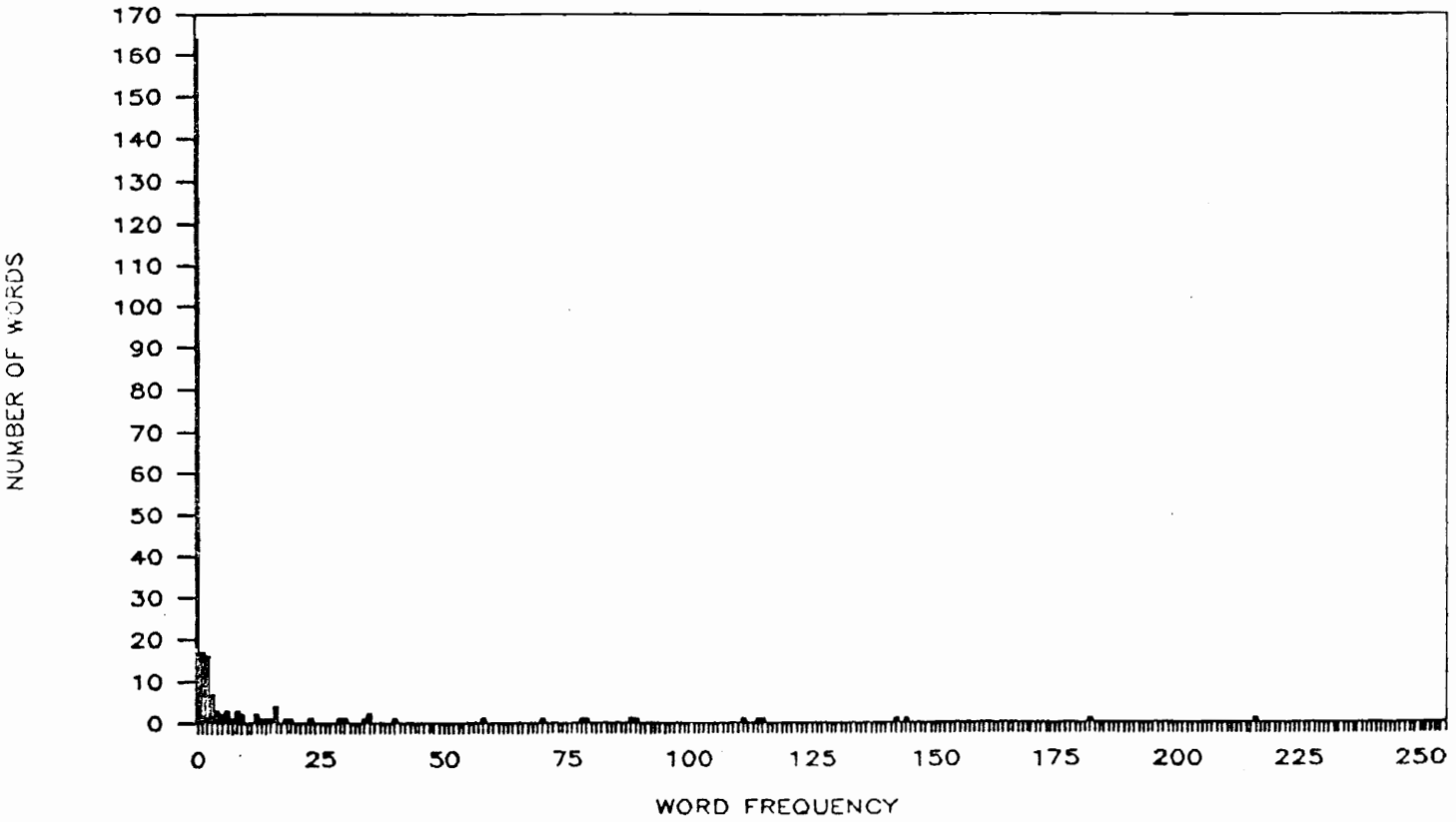


Figure 5.6 Graph of Number of Words With Word Frequency: Manual Source 2

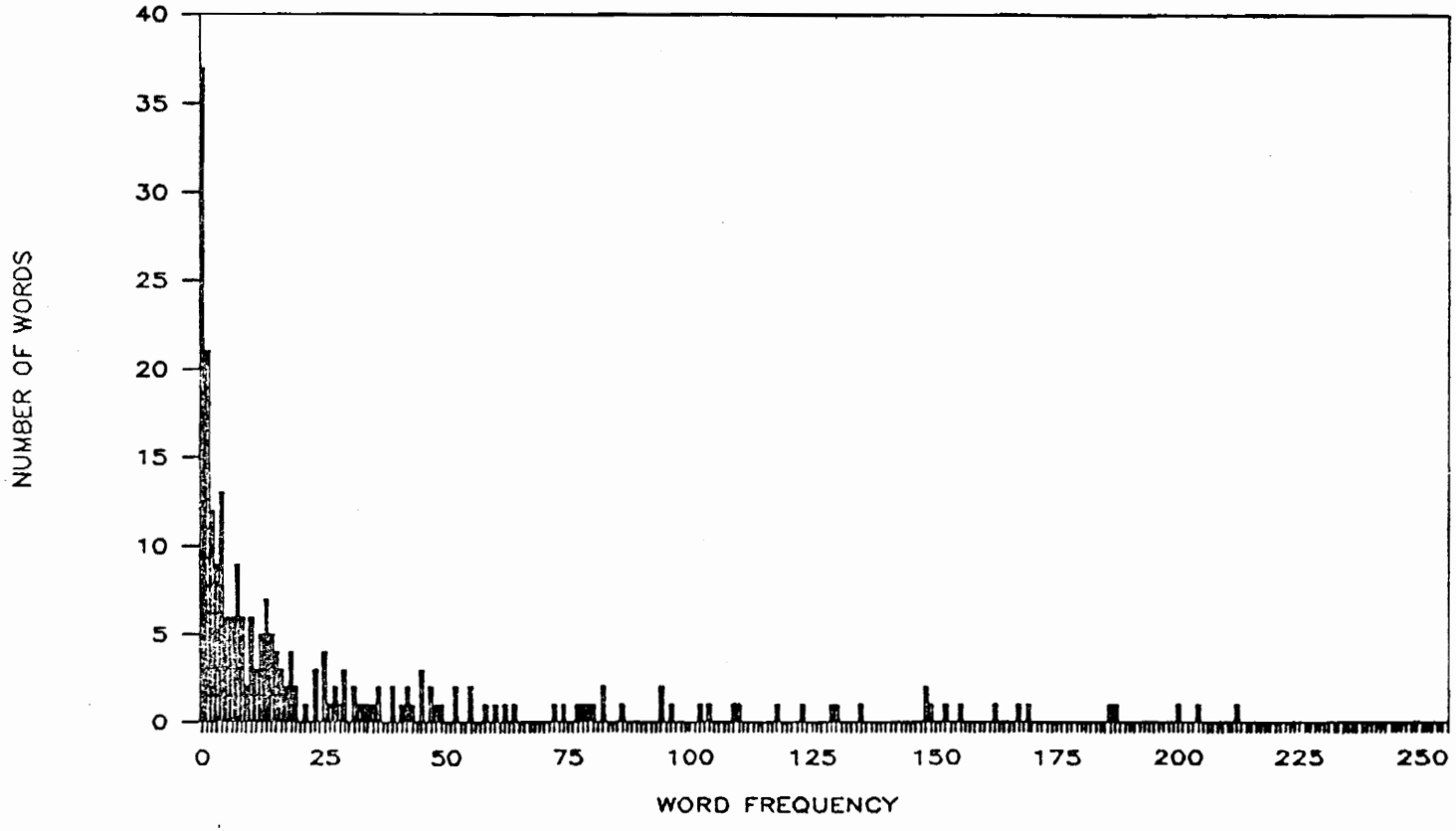


Figure 5.7 Graph of Number of Words With Word Frequency: Textbook Source 1

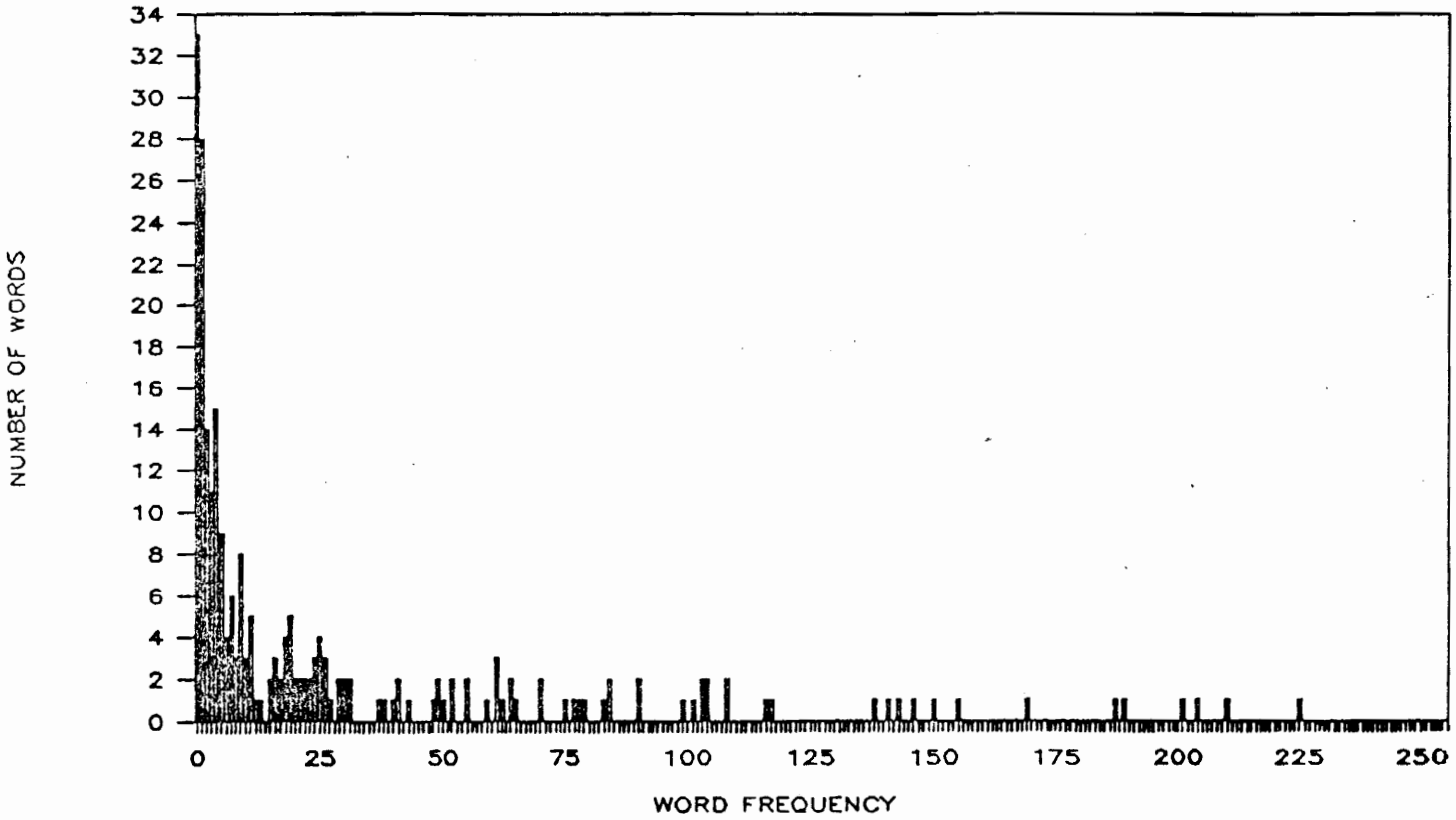


Figure 5.8 Graph of Number of Words With Word Frequency: Textbook Source 2

* address	interrupt
address register	* memory
* bit	microprocessor
bus	* operand
byte	* processor
* data	* register
exception	signal
fetch instruction	state
* instruction	* word

* indicates overlapping term

Figure 5.9: Expert Basic Level Terms

* address	* memory
* bit	microprogram
character	* operand
computer	page
* data	* processor
data path	segment
information	* register
* instruction	result
line	store instruction
machine	* word

* indicates overlapping term

Figure 5.10: Novice Basic Level Terms

Source A	Percent Overlapping Basic Level Terms in Source	Source B	Percent Overlapping Basic Level Terms in Source
Manual-1	84.6%	Manual-2	68.7%
Manual-1	46.1%	Textbook-1	40.0%
Manual-1	53.8%	Textbook-2	53.8%
Manual-2	43.7%	Textbook-1	46.6%
Manual-2	50.0%	Textbook-2	61.5%
Textbook-1	53.3%	Textbook-2	61.5%

Note: Ratio% = number of overlapping terms : total basic level terms

Figure 5.11: Percent Overlapping of Basic Level Terms

while two basic level terms, namely ´information´ and ´instruction´, are assumed by the author to be previously understood by the reader. The microprocessor manuals naturally assume that the reader already knows what a microprocessor is. The second sentence of one of the microprocessor manuals, quoted below, illustrates that the material targeted toward an expert audience assumes that more detailed knowledge is possessed by the reader.

"Utilizing VLSI technology, the MC68010 is a fully-implemented 16-bit microprocessor with 32-bit registers, a rich basic instruction set, and versatile addressing modes." [Motorola 82, p. 1]

This passage is clearly intended to be understood by a reader who knows what the basic level terms ´register´ and ´bit´ mean, but also that the size of microprocessors and registers may be characterized by their number of bits.

The selected word frequency criteria for determining which terms are basic level, and which are not basic level, resulted in several instances where a basic level class is a subclass of another basic level class. The first of these vertical conflicts is between the terms ´machine´ and ´computer´ in both textbook sources. Examination of the sources shows that this is result of the word ´machine´ being used as a synonym for the word ´computer´ in many instances.

Basic level terms also conflict in textbook source 1 between the terms ´instruction´ and ´store instruction´, and in manual source 1

between the terms ´instruction´ and ´fetch instruction´. These conflicts appear to occur as a result of lengthy discussion of store instructions as compared to other types of instructions in the first case, and of fetch instructions in the second case. The term ´signal´ and its subclasses ´exception´ and ´interrupt´ conflict in both manual sources. In addition, the ´exception´ subclass ´bus error´ occurs as frequently as basic level terms in manual source 1. This seems to occur due to lengthy description, at several levels of abstraction, of signals used within the microprocessor of discussion.

In the hardware hierarchy, several more vertically related basic level terms appear manual source 2 and both textbook sources. The terms ´memory´ and ´register´ conflict in all three sources, and the ´register´ subclass ´address register´ occurs with the frequency of a basic level term in manual source 2. These conflicts seem to be due to the lengthy discussion of memory and registers at several levels of abstraction in these sources. In the expert basic level list, the terms ´processor´ and ´microprocessor´ both appear because the manual sources used different levels of abstraction.

All of these instances where a basic level category is a subclass of another basic level category have been left untouched in the knowledge base. Future text generation by the KALOS system [Cline 91] will use the hierarchically closest basic level category to use in a particular discussion. With this intended approach for KALOS to use for term selection, results will show whether or not instances of vertically related hierarchical basic level classes are desirable or not.

5.2 Attributes

Creating a list of attributes for taxonomy classes was not a goal of this project. While manually examining the source materials for semantic relationships, attributes were incidentally noticed by the author. An example of a passage which establishes a class - attribute relationship between the class 'interrupt' and the attribute 'priority' is below.

> "In fact, one interrupt may imply a higher priority than usual for another interrupt." [Gorsline 86, p. 232]

These attributes will help provide the KALOS text generation system [Cline 91] with a set of terms to use when describing things. A list of those attributes which were compiled is shown in Figures 5.12a, 5.12b, and 5.12c. It is emphasized that a specific search for attributes was not conducted in this project, and that this list is by no means claimed to be complete.

5.3 Parts

An effort was made to manually compile a list containing the parts of whole entities represented by categories within the hierarchy. The list is based solely upon semantic relationships found in the source material. An example of a passage which establishes a class - part relationship between the class 'memory' and 'cell' follows.

> "The memory unit consists of 1000 cells or locations, numbered 000 through 999." [Gorsline 86, p. 3]

Item	Attributes
alu	logic size
bit	position, set or reset
bus	bus cycle, bus state, communications mode, direction, number of lines, parallel lines
bus cycle	number of clock cycles, signals, timing
character	character code
chip	integration level, operating temperature range, passive, power consumption, supply voltage
clock	frequency, speed
clock cycle	speed
communications	communications mode, data path, protocol
complement instruction	monadic
computer	architecture, character set, digital logic, machine levels, operating system, runs programs, system state
control bus	control signals
control unit	executes instructions, fetches instructions
data	data type, encoding
data path	capacity
data type	data format, logic size
destination	address
digit	width
digital circuit	two logic levels
direct addressing	direct address
disk drive	direct access
distributed system	distance between nodes, number of nodes
dyadic instruction	two operands
dynamic memory	refresh needed
electronic component	semiconductor
eprom	read only access
eeprom	read only access
exception state	externally generated
file	path
flip flop	bistable, edge triggered

Figure 5.12a: Attributes

Item	Attributes
handshaking	asynchronous
i/o cycle	state
indexed addressing	indexed address
indirect addressing	indirect address
instruction	address mode, dyadic, instruction format, length, number of operands, result
instruction format	number of addresses
interrupt	priority level, vector table entry, asynchronous
latch	level triggered
logical instruction	truth table
macro	linking vector
magnetic tape	bpi, length
memory	capacity, stores information, subsystem of computer
memory cell	address, contents, size
microprocessor	active, addressing mode, address range, internal state, logic size, maximum address capability, operating mode, speed, subsystem of computer
microprogram	stored in rom
mode control instruction	monadic
monadic instruction	one operand
monitor	resolution
multicomputer	coupling
multiprocessor	common primary memory, master operating system
network	number of computers
not instruction	monadic
octal	logic size
operand	address, size
os	privileged instruction access
parallel processor	interprocess communications
peripheral	serial access, subsystem of computer
pop instruction	decrements stack pointer

Figure 5.12b: Attributes (continued)

Item	Attributes
primary memory	random access
printer	resolution, speed
process	priority, process state
processor	executes instructions, interfaces with peripherals, internal state
program	instruction sequence, size
program counter	address of next instruction
prom	read only access
push instruction	increments stack pointer
queue	head, tail
ram	read/write access
read cycle	state
register	number
reset	active level
rom	read only access
rotate instruction	monadic
secondary memory	serial access
set	active level
shift instruction	monadic
signal	active state
signal	inactive state
source	address
stack	stack pointer, top
stack instruction	monadic
stack addressing	stack address
static memory	no refresh needed
trap instruction	synchronous
uart	asynchronous
usart	synchronous
virtual addressing	virtual address
virtual machine	language
voltage	value
write cycle	state

Figure 5.12c: Attributes (continued)

The list of parts is shown in Figures 5.13a and 5.13b.

5.4 Synonyms

All source materials contained various synonyms for repeatedly discussed objects. As discussed in section 4.1, synonyms were identified by manual analysis of the sources. When compiling the frequency of occurrence for each class in the taxonomy, the frequency of occurrence of all synonyms for each class were summed to represent the number of times that each class was discussed. This was done because synonyms for a given term only reflect different surface text representations for an unique category. A list of the synonyms found is shown in Figures 5.14a and 5.14b. This list should be helpful to the surface text generation and revision modules of the KALOS system [Cline 91].

5.5 Members of Sets

Some of the terms used in the source materials which were not part of the taxonomy or other lists compiled, but were obviously members of sets, were deemed to be of potential use to a natural language generation system. An example of a passage that assigns ´FORTRAN´ to the set of ´languages´ is below.

> "An interesting assembly language programming assignment involves providing pseudorecursive subprogram calls for a language such as FORTRAN." [Gorsline 86, p. 51]

Item	Parts
alu	accumulator, flag
applications program	data, declaration, function, instruction, procedure, subroutine
back end	code generator, code optimizer
bus	line
bus cycle	machine cycle
chip	logic gate, pin, plastic casing, silicon wafer
clock cycle	falling edge, reset phase, rising edge, set phase
compiler	back end, front end, symbol table
computer	firmware, hardware, input/output, memory, processor, software
computer system	computer
condition codes	status bit
control register	condition codes, interrupt mask
control unit	address register, control register, instruction register, program counter
cylinder	track
device register	control register, data register, status register
disk	cylinder, root directory, sector, track
disk drive	disk
distributed system	node
dynamic memory	capacitor
file	data, header
front end	lexical analyzer, semantic analyzer, syntactic analyzer
i/o cycle	machine cycle
instruction	field
instruction cycle	execute cycle, fetch cycle, status check
job	task
kernal	vector table
library	macro
machine cycle	clock cycle

Figure 5.13a: Parts of Items

Item	Parts
magnetic tape	track
memory cell	bit
memory chip	address decoder, chip select pin
memory cycle	read cycle, write cycle
microprocessor	alu, bus, clock, control unit, instruction set, register
network	node
number	base, exponent, magnitude
opcode	operand, operator
operating system	communications manager, kernal, process control block, resource manager, scheduler, shell
peripheral	controller
primary memory	memory cell
program status word	condition code, interrupt code, mask bit, priority, program counter
protocol	baud, carrier, channel, checksum bit, clocking, data stream, encoding, handshaking, parity bit, stop bit
program	process
read cycle	machine cycle
register	flip-flop
resource manager	paging system
root directory	file, subdirectory
secondary memory	file
secondary storage device	read/write head
signed integer	sign
silicon wafer	electronic component, logic gate
static memory	latch
status register	condition codes, interrupt mask
subdirectory	file, subdirectory
tape drive	magnetic tape
track	sector
vector table	interrupt vector
write cycle	machine cycle

Figure 5.13b: Parts of Items (continued)

Term	Synonyms
abnormal result	erroneous result, incorrect result, invalid result
active state	active
address	location
address register	ar
addressing mode	address mode
applications program	user program
arithmetic logic unit	alu
array processor	simd processor
ascii	american standard code for information interchange
asleep	blocked
asserted	logical 1, logical one, set, true
bit	binary digit
bpi	bits per inch
branch instruction	jump instruction, skip instruction
central processor	central processing unit, cpu
chip	ic, integrated circuit
circuit board	printed circuit board
clock generator	clock
clock cycle	clock period
coaxial cable	coax
condition code register	ccr
control register	cr
data path	communications link, communications path, link
destination address	destination
device register	interface register, port
dma	direct memory access processor, dma processor
eeprom	earom, electrically alterable read-only memory, electrically erasable programmable read-only memory
emulator	interpreter
eprom	erasable programmable read-only memory
external state	system state
floppy disk	diskette, flexible disk, floppy, floppy diskette
floppy disk drive	floppy drive
general purpose register	general register
hard disk	disk pack, platter
hard drive	hard-disk drive, Winchester drive
i/o instruction	input/output instruction
i/o port	input/output port
i/o processor	input/output processor
i/o register	input/output register
inactive state	inactive

Figure 5.14a: Synonyms

Term	Synonyms
information	contents
instruction	operation
instruction register	ir
logical instruction	boolean instruction
lsi	large-scale integration
machine level	virtual machine
macroprogram	macro
memory address register	mar
memory data register	mdr
memory location	memory cell
microprogram	machine code, machine language, microcode
monitor	console, crt, terminal, terminal screen, video display terminal
msi	medium-scale integration
negated	false, logical 0, logical zero, reset
normal result	correct result, valid result
opcode	mnemonic
os	operating system
parallel processor	multiprocessor
peripheral	external device, i/o device, input/output device
program counter	pc
programmed logic array	pla
processor state	internal state, processor mode
program status word	process status word, psw
prom	programmable read-only memory
ram	random access memory
real	floating-point number
rom	read-only memory
secondary memory	mass storage
secondary storage device	mass storage device, secondary memory device
source	source address
ssi	small-scale integration
ssp	supervisor stack pointer
state	operating mode
supervisor state	supervisor mode
supply voltage	vcc, voltage
user stack pointer	usp
user state	user mode
valid area register	boundary register, memory protection register
vector processor	misd processor
vector table	interrupt table
virtual addressing	logical addressing
vlsi	very-large-scale integration
wrom	writable read-only memory

Figure 5.14b: Synonyms (continued)

The lists of members of the sets ´language´ and ´operating system´ were manually supplemented with additional members that were known by the author but not mentioned in the source materials. Of the set of characters, only a few were actually used in the sources. The members of sets are shown in Figure 5.15.

Set Category	Members
encoding	ascii, bcd, ebcdic, baudot
language	ada, algol, apl, assembly language, basic, c, c++, clu, cobol, fortran, lisp, machine language, modula 2, pascal, prolog, rpg, simula 67, smalltalk
operating system	a/ux, cp/m, jcl, macintosh, ms-dos, mvs, os/2, vms, unix
character	the set of alphanumeric keyboard characters

Figure 5.15: Members of Sets

Chapter 6 The Complete Knowledge Base

The taxonomy of terms related to microprocessors and some general knowledge about computers is significantly enhanced with the information contained in the other lists compiled in this project. To illustrate this, the term 'bus' will be examined.

With merely the taxonomy as its only source of domain knowledge, a text generation system would know that a bus is a type of data path. It would also know that there are three types of busses, namely address busses, control busses, and data busses. This information tells the text generation system that these things exist, what they are called, and that there is a class relationship between them.

The basic level terms provide a natural language generation system with knowledge of what terms the target audience should be familiar with. The term bus is in the expert basic level terms but not in the novice basic level terms. If the target audience is expert, the text generation system should choose to explain a particular bus for a particular microprocessor as the bus's unique characteristics relate to the characteristics of a typical bus. If on the other hand the target audience is novice, the text generation system should first tell the audience that a bus is a type of data path before explaining the details of the specific bus to be discussed.

The bus class has some attributes. A natural language generation system would know from the attribute list that a bus typically has parallel lines, cycles, speed, capacity and communication mode. The

particular bus to be discussed by a natural language generation system should be discussed in terms of either its instantiated size and speed values or its exceptions to typical bus attributes.

The list of parts of things tells a natural language generation system that a bus is part of a microprocessor and that a bus has lines. A generation system could tell the audience that the particular bus under discussion is part of a particular type of microprocessor. It can also discuss the quantity and type of lines contained within the bus.

A pictorial representation of the semantic network nodes and arcs that exist for the terms discussed earlier in this section is illustrated in Figure 6.1.

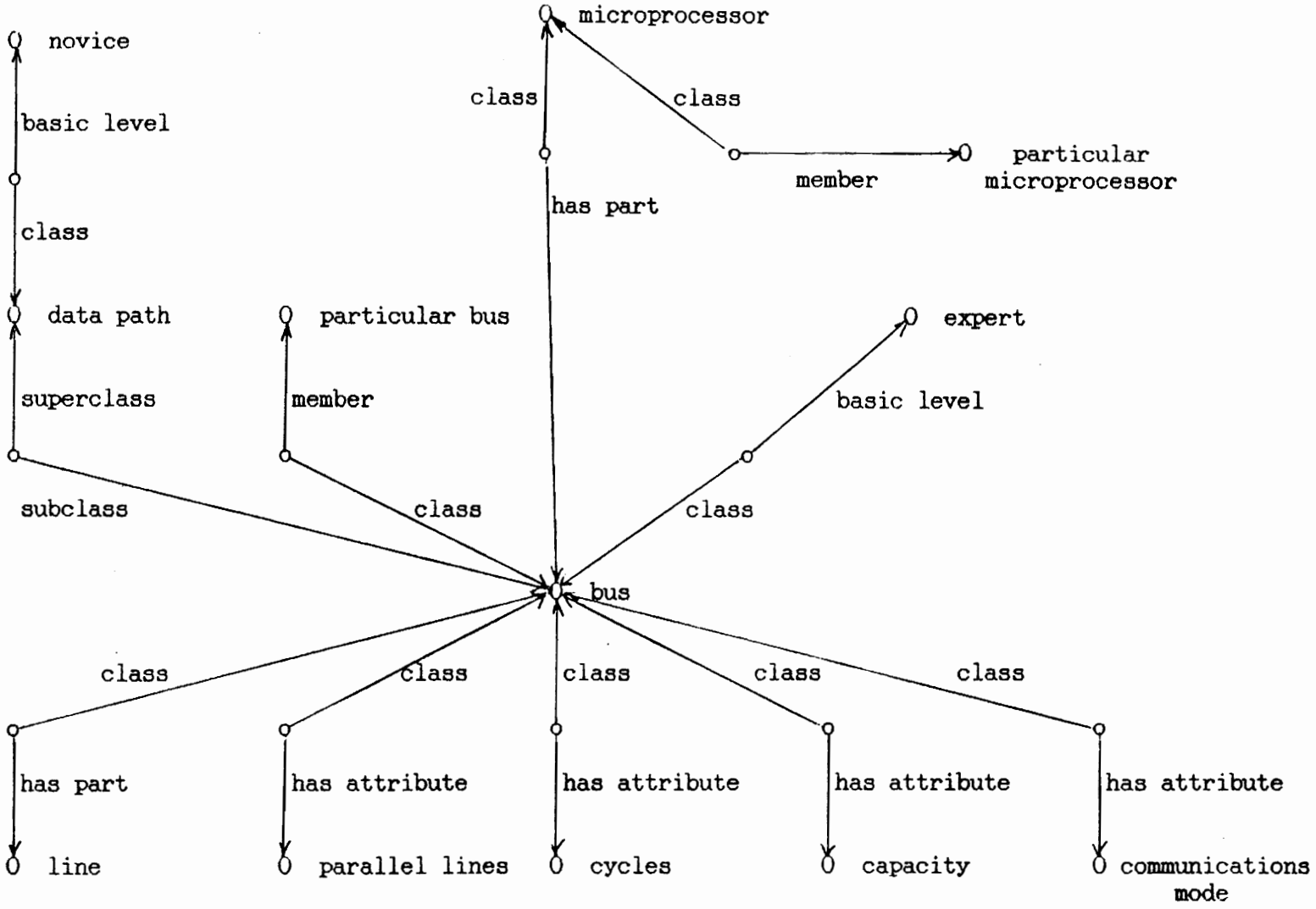


Figure 6.1: Partial Semantic Network

Chapter 7 Conclusions

7.1 Critique

This report has explored a means of developing a domain knowledge base. A brief summary of what has been done in this project is presented in Figure 7.1.

An obvious source of information for developing a knowledge base is published material within the domain of interest. Although only a small sample of published work was used for this project, the number of sources was sufficient to demonstrate that analysis of these materials can be of tremendous value in constructing a taxonomy and in compiling lists of non-taxonomic information which enriches a knowledge base.

The selection of textbooks as novice level material to find novice basic level terms is difficult. The purpose of the textbooks used in this project is to transform an undergraduate computer science novice into more of an expert. More sophisticated means of analysis to determine novice basic level terms from this type of source may be important to use as a supplement to the methods described in this report. These other techniques could include separate examination of term usage within different sections within a source book. The terms of the taxonomy which were labeled as basic level terms in this project and which are subclasses of other categories which are labeled basic level terms may have been incorrectly identified as such. If so, this is likely due to skewing of the frequency of occurrence by extended discussion in localized parts of the source material.

- Developed a taxonomy on microprocessors and some computer-related terms for use in the domain knowledge base of the KALOS natural language generation system [Cline 91].

- Explored a new method of knowledge base development which entailed a semi-automatic process to extract information from existing published domain discourse.

- Experimented with high frequency of occurrence of terms as criteria for identification of basic level categories.

- Demonstrated that a straightforward and simple process may provide information necessary to construct knowledge bases for complex expert systems.

Figure 7.1: Project Summary

Other sources for novice level material besides textbooks should be explored as well, and objective criteria for textbook level differentiation should be compiled. For this project, novice basic level terms are not of great concern because the KALOS system [Cline 91], as currently envisioned, will primarily be used to generate text about microprocessors which is targeted to an expert audience.

None of the basic level terms determined by this project have been validated through empirical testing of human subjects. Such testing, as described in [Rosch et al. 76] and [Jolicoeur 84] would show whether or not the terms identified as basic level in this project are in fact basic level terms in most people. Tests could be organized in a manner to find basic level terms which may not have been found by analysis of published discourse in this project. The empirical test results could also show whether or not the word frequency cutoff of the sample mean plus the sample standard deviation is optimal.

7.2 Usefulness

The acquisition methods explored in this project are potentially of great use in developing non-trivial knowledge bases. The lists contained within this report should be helpful to the KALOS natural language generation system and may be of use in other applications requiring domain knowledge of microprocessors.

7.3 Further Work

The results obtained in this project should be further validated

through empirical testing and actual text generation. More work is needed to refine all methods used in this project, especially in compiling lists of attributes for categories.

Examining the frequency of term use along with choice of term use versus schema slot location should demonstrate that term use is consistent with predictions. That is, where predictions say basic level terms should appear, there should be a consistent word choice. Complete discourse schemata analysis to determine slot contents will be a tremendous undertaking.

References

Allen 87
Allen, J. *Natural Language Understanding.* Benjamin/Cummings, Menlo Park, CA (1987).

Ashenhurst et al. 80
Ashenhurst, R.L. et al. *Taxonomy of Computer Science & Engineering.* American Federation of Information Procesing Societies, Inc., Arlington, VA (1980).

Berlin, Breedlove & Raven 73
Berlin, B., Breedlove, D.E., & Raven, P.H. "General Principles of Classification and Nomenclature in Folk Biology." *American Anthropologist* V 75 (1973). pp. 214-242.

Beyer 91
Beyer, W.H. *CRC Standard Mathematical Tables and Formulae.* 29th ed. CRC Press, Inc., Boca BRaton, FL (1991).

Cline 91
Cline, B.E. "Project Proposal." Unpublished (1991).

Gorsline 86
Gorsline, G.W. *Computer Organization: Hardware/Software.* 2d ed. Prentice-Hall, Inc., Englewood Cliffs, NJ (1986).

Hirschberg 85
Hirschberg, J.B. *A Theory of Scalar Implicature* (1985) Diss. U. of Pennsylvania (Dep't. of Comp. Sci.).

Jolicoeur, Gluck & Kosslyn 84
Jolicoeur, P., Gluck, M.A. & Kosslyn, S.M. "Pictures and Names: Making the Connection." *Cognitive Psychology* V16 (1984). pp. 243-275.

Kane 81
Kane, G. *68000 Processor Handbook.* OSBORNE/McGraw-Hill, Berkeley, CA (1981).

McKeown 85
McKeown, K.R. *Text Generation.* Cambridge University Press, Cambridge, GBR (1985).

McKeown & Swartout 87
McKeown, K.R. & Swartout, W.R. "Language Generation and Explanation." *Annual Rev. Comp. Sci.* V2 (1987), pp. 401-449.

Miller 78
Miller, G.A. "Practical and Lexical Knowledge." *Cognition and Categorization* (ed: E. Rosch and B. Lloyd), Lawrence Erlbaum Associates, Hillsdale, NJ (1978), pp. 305-319.

Motorola 82
Motorola Semiconductors, Inc. *MC68010 16-Bit Virtual Memory Microprocessor.* Austin, TX (1982).

Nutter & Cline 91
Nutter, J.T. & Cline, B.E. "Benefits of Natural Taxonomies for Natural Language Generation." Unpublished (1991).

Palmer 78
Palmer, S. E. "Fundamental Aspects of Cognitive Representation." *Cognition and Categorization* (ed: E. Rosch and B. Lloyd), Lawrence Erlbaum Associates, Hillsdale, NJ (1978), pp. 259-303.

Peters & Shapiro 87
Peters, S.L. & Shapiro, S.C. "A Representation for Natural Category Systems." *Proc. 10th IJCAI, Milan* (Los Altos, CA: Morgan Kaufman) (1987) pp. 140-146.

Peters, Shapiro & Rapaport 88
Peters, S.L., Shapiro, S.C. & Rapaport, W.J. "Flexible Natural Language Processing and Roschian Category Theory." *Proc. 10th Annual Conf. Cog. Sci. Soc.* (1988), pp. 125-131.

Reiter 90
Reiter, E. "Generating Descriptions that Exploit a User's Domain Knowledge." *Current Research in Natural Language Generation* (ed: R. Dale, C. Mellish and M. Zock) London: Academic Press (1990), pp. 257-285.

Rosch et al. 76
Rosch, E. et al. "Basic Objects in Natural Categories." *Cognitive Psychology* V8 (1976), pp. 382-439.

Rosch 78
Rosch, E. "Principles of Categorization." *Cognition and Categorization* (ed: E. Rosch and B. Lloyd), Lawrence Erlbaum Associates, Hillsdale, NJ (1978), pp. 27-48.

Shapiro 92
Shapiro, S.C. "SNePS-2.1 User's Manual." Department of Computer Science, State University of New York, Buffalo, NY (1992).

Shortcliffe 76
Shortcliffe, E.H. *Computer-Based Medical Consultations.* Elsevier, New York (1976).

Spiegel 75
Spiegel, M.R. *Schaums Outline of Theory and Problems of Probability and Statistics.* McGraw-Hill, New York (1975).

Stark & Bowyer 91
Stark, L. & Bowyer, K. "Achieving Generalized Object Recognition through Reasoning about Association of Function to Structure." *IEEE Trans. on Pattern Analysis and Machine Intelligence* V13 N10 (Oct 91), pp. 1097-1104.

Tanenbaum 84
Tanenbaum, A.S. *Structured Computer Organization.* 2d ed. Prentice-Hall, Inc. Englewood Cliffs, NJ (1984).

Vita

The author is a Captain in the United States Army Signal Corps. Significant positions held in the U.S. Army have been: Electronic Warfare/Voice Intercept Equipment Repair Supervisor, Division Artillery Communications Platoon Leader, Forward Area Signal Center Platoon Leader and Signal Battalion Assistant Operations Officer. Educational background includes an Associate of Arts degree in general studies from the University of the State of New York, a Bachelor of Science degree with majors in computer science and mathematics from Pembroke State University, Pembroke, North Carolina. This project is in partial fulfillment of the requirements for a Master of Science degree in computer science for Virginia Polytechnic Institute and State University located in Blacksburg, Virginia. Born in Gloucester, Massachusetts, the author, along with his wife and two children, have lived in several states and Berlin, Germany. Future plans include continuation of a satisfying career with the United States Army.

Michael F. Emero