1978

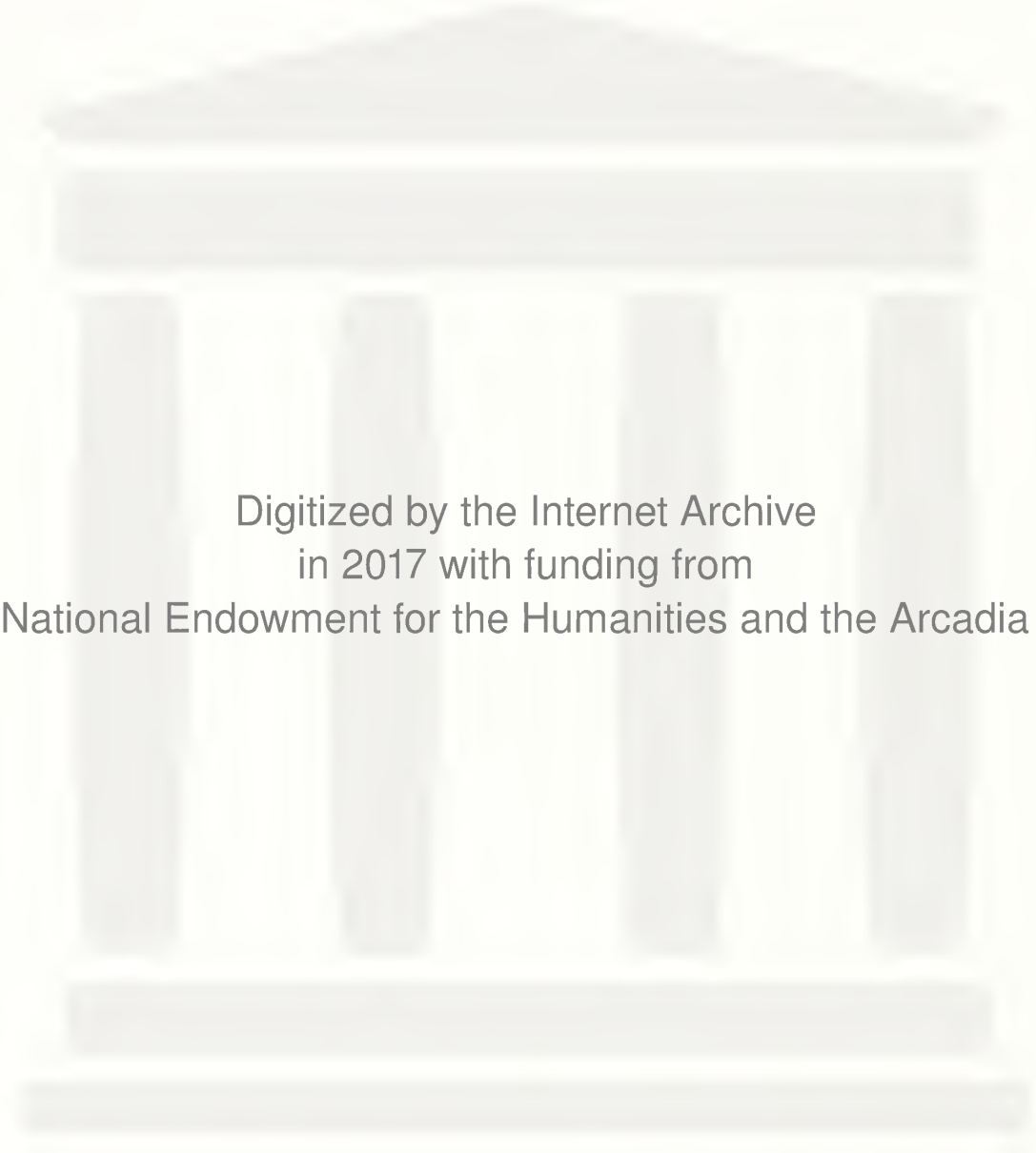
Automatic coding of medical problem lists

Seth M. Powsner
Yale University

AUTOMATIC CODING OF MEDICAL PROBLEM LISTS

SETH M. POWSNER

1978



Automatic Coding of Medical Problem Lists

by Seth M Powsner

BSEE Massachusetts Institute of Technology 1974

A Thesis Submitted to the

Yale University School of Medicine

in Partial Fulfillment of

the Requirements for the Degree of

Doctor of Medicine

March 1, 1978

To Dr and Mrs Thomas R Forbes

Acknowledgements

A number of people deserve thanks for their help in preparing this thesis. Tom Chappel, Mariann Ogilvie, and Ed Rupp all helped program even while pursuing their undergraduate studies at the Georgia Institute of Technology (GIT). The faculty and staff of the GIT Computer Science Department generated a friendly atmosphere and many stimulating lunch-hour discussions. Martha Hansert, Marshal Mandelkern, and Neal Ryan kindly donated their time to edit earlier drafts of the thesis itself. A very special thanks is due Dr Katherine Finseth who assisted in the ever-so tedious testing of the final program. She and Dr Brunjes are responsible for setting-up the clinical event coding system used in the thesis.

To close I would like to thank my advisors Dr Brunjes and Dr Seligson, not only for supervising this work, but for taking the time and interest four years ago to convince me that Yale Medical School was the place to pursue my interests in medical computing.

This work funded in part by a research grant from the U.S. Public Health Service and a training grant from the National Library of Medicine.

Table of Contents

I -- Introduction

There is valuable data in medical records which presently requires human processing for use. For individual patients, important aspects of their medical histories may literally be burried in their "old chart". For the hospital and the community at large medical trends may be observed by tabulation of the illnesses recorded. For the researcher the clinical course, family history, and environmental factors may already be noted for a disease in question if the approriate charts can be retrieved. All of this, the traditional medical record, is written in longhand in a free form style. Aside from the difficulties presented by physicians' handwriting, mechanical processing of this information is very complex because it is english prose.

Attempts are being made to improve the quality and increase the availability of information in medical records. The problem-oriented record introduces more structure to clarify the status of data noted in hopes of clarifying the purpose and approach of the treatment undertaken. Medical centers hoping to obtain some of the benefits of automated processing and retrieval have introduced multiple-choice check-off forms for the review of systems and physicals. Perhaps the greatest benefit is the elimination of a few of the scrawled pages. Even the most complete of these multiple-choice forms for medical records still include prose histories, problem lists, and comments.

However stylized, medical charts are collections of ordinary language. Natural language processing techniques allow computers to handle typed prose if it is limited in form and content. Understanding

natural language, "english as she is spoke", remains a major research area in computer science. When reading even simple prose, human beings bring to bear knowledge derived from years of living in the world and an innate ability for language comprehension. For practical application of natural language processing techniques the subject area must be precisely defined. If it is not, only statistical observations on word usage may be possible. As only the simpler grammars may be rapidly parsed, the form of the text must also be limited.

Automated systems for processing medical prose have been constructed. This has generally been done in an ad-hoc fashion based on semantics. These systems have been used to index records based on diagnositic phrases. In part, their ad-hoc nature may reflect an emphasis on producing a practical system. Linguistically, it reflects a limited understanding of the construction of "ungrammatical" sentence fragments so common in daily discourse. Are these fragments, often noun phrases, truly examples of incorrect grammar? A semantic approach can avoid this issue. It is a semantically based technique that is used in this thesis.

A procedure was developed in this thesis to encode problem lists from a general clinic. While the goal of this thesis was to develop a functioning program to solve an existing problem, the program is discussed in relation to general approaches to natural language processing and medical coding.

II -- Problem Statement

A -- Overview

The problem addressed is the development of a computer program capable of encoding problem list entries keypunched from general medical clinic charts with the constraint that the encoding used be compatible with the numeric codes used to store the rest of the charts. Clinic visits were recorded on forms which allowed most information to be recorded with check-marks. The forms were designed for keypunching, and over a number of years a computer data-base was constructed from them. Not all information was in check-off form. Areas of the form contained blank lines for the physician to write comments; and, on the front of the form, the problem list was written out. The problem list and any comments were keypunched as best the typist could read them and stored without editing in the data-base.

Previous work tabulating the various types of medical problems seen in the clinic led to the compilation of a dictionary of terms used in the problem lists. A great deal of manual effort was expended to create this dictionary and a list of spelling corrections to facilitate its use. However, the content of the dictionary was never checked as the initial tabulation was completed without it.

The numeric code used to store most of the information in the records was developed along the lines of the Systematized Nomenclature of Pathology. The major revisions have been to allow appropriate coding of signs and symptoms making up a large portion of the data in a general clinic, but almost never part of anatomic pathology reports.

To limit its scope, this thesis does not delve into the problems of consistency, utility, and privacy of automated medical record keeping. Computer based storage of data makes incosistencies in records glaringly apparent. Practically, there may be no significant difference between "strep throat" written as a temporary problem list entry and "sore throat" checked-off as a common pediatric complaint. The critical datum will be the bacteriology results which are not available when the initial chart entry is written. However,"strep throat" and "sore throat" will have to be encoded and stored in some form when they are first noted. A consensus of the record system users will have to be reached around this coding problem and a watch maintained to insure the consistent encoding of all data entered. It is assumed that overall, an automated medical record system would be cheaper than the present manual methods. That certainly is the experience of any large organization in keeping records of its business activities. However, the high salaries of the people whose actions will have to accommodate any automation and the nature of the data to be recorded require serious consideration of the economics. Further complication is caused by the development costs that will be incurred by any organization attempting medical record automation at this time. The problem of privacy is equally important. While sufficient safeguards can be implemented, these would increase the already expensive development of an automated medical record system. Finally, attention to privacy and security would complicate access to medical records for physicians and nurses who are accustomed to simply picking up any chart they need.

II -- Problem Statement (cont)

B -- Specific Issues

1 -- Clinical Event Concept for Medical Coding

The numeric coding system for clinical records is based on a fundamental hypothesis: medical records are listings of discrete datum describing clinical events and a clinical event may be completely characterised by listing its values along a small number of dimensions [BRU71, PRA73]. For automated records it is a great simplification if the record can be divided into smaller pieces rather than being forced to deal with it as an amorphous whole. Making each of these pieces a small collection of numbers affords easy storage and retrieval. The question, which is never fully answered, is whether something significant is "lost in the translation".

The major dimensions of a clinical event are organ system affected, specific topography, function or dysfunction, and etiology. Taken with the fundamental hypothesis, this says that a medical record is a collection of statements about the functioning of organ systems and the suspected etiology of any abnormality. Some descriptive power is added by the minor dimensions. Nevertheless, statements can not be formulated about interrelationships between clinical events. Empirically the majority of information in medical charts does not require desciption of clinical interrelationships. Systemitized Nomenclature of Pathology (SNOP) [CAP65] codes pathology reports in a similar style very sucessfully. A SNOP coding does not specify a system dimension, only topography. In addition, it does allow morphology to be specified which

is important to pathology reports.

To clarify clinical event coding some examples of information from a medical chart is presented followed by its encoding. The use of letters as part of a "numeric" code is not meant to confuse the reader. It may be imagined that these codes are stored in base 36.

i) Patient complains of pain in his lower abdomen.

System: SE120 No system specified. Function Only (Sign or Symptom)

Topography: TY424 Hypogastric Region

Function: F320F Pain, not otherwise specified

Etiology: E0000 Not specified

Source: Patient

ii) WBC 12,000

System: S9700 Leukocyte Line

Function: FCBA0 Lab Test

Quantity: 12000

iii) Appendicitis

System: S6400 Lower Gastrointestinal

Typography: T6600 Appendix, not otherwise specified

Function: F420F Inflamation, not otherwise specified

Etiology: E0000 Not specified

Within a computer only the codes themselves would be stored. Example iii would simple be stored as S6400 T6600 F420F E0000.

The values of the major dimensions are numbers denoting a place in hierarchy of possible codes. A portion of the code list appears below.

```
System Codes

0000 Psychological
1000 Nervous System
2000 Eye
3000 Ear
4000 Respiratory
      ...
     4200 Breathing
     4400 Upper Respiratory
     4460 Lower Respiratory
      ...
5000 Cardiovascular
 ...

Function Codes

1000 Basic Functions
      ...
     1600 Cessation of Function
          1620 Paralysis
          1640 Absent Function
               1642 Function Cessation Due to Aging
      ...
2000 Dysfunctions
      ...
     2C00 Unusual Movements
      ...
3000 Pain & Itching
      ...
     3200 Pain
      ...
4000 Inflamation & Infection
      ...
     4400 Infection
          ...
          4440 Bacterial Infection
           ...
      ...
 ...
```

A hierarchical coding system allows specification of either broad or narrow ranges of interest for information retrieval. A search for Function=4440 would imply interest in only bacterial infections. A search for Function=44-- would imply interest in any type of infection. This approach has proved quite successful in SNOP.

The minor dimensions are time, information source, subject if other than the patient, and modifiers of dimenstions. A chart might include "parent reports younger brother had 3+ sugar in urine" which would be coded:

System: S7200 Urological

Topography: T7X10 Urine

Function: FCA40 Lab Test for Sugar

Quantity: +++

Source: Parent

Subject: Younger Brother

A complete list of acceptable values for the minor dimensions has not been formulated, but the principle should be clear.

II -- Problem Statement (cont)

B -- Specific Issues (cont)

2 -- Problem Lists

The problem list serves as the index for a medical record and thus as an important synopsis of the record. Weed set out the purpose of a problem list in his scheme for problem oriented medical records [WEE69]. At any point in time a physician should be able to identify the patient's current medical difficulties as well as any significant previous difficulties from the problem list. The problem list is placed at the front of the chart and numbered for easy reference to serve as the focus of a clear image of the patient's medical status. Dates indicating when a problem was first noted and when it resolved allow a time course to be seen. This is particularly important in effective outpatient care where long intervals between visits make continuity difficult to maintain. In the Community Health Care Center Plan "Encounter Form" for recording clinic visits [CHC74] space for the problem list is reserved on the face sheet.

Community Health Care Center Plan, Inc. (CHCP) is a prepaid health maintenance organization located in New Haven, Conneticut. It first opened in October 1971 [LYN75]. CHCP was one of a number of such health maintenance organizations initially sponsored by the goverment to test the feasiblity and effectiveness of preventive medicine, general practice oriented medical centers. From the outset, a concerted effort was made to maintain a computer data-base of the medical records. It was hoped that this would simultaneously increase efficiency in record

handling and in general operation by providing accurate tabulation of the quantity and types of service most needed. While the computer data-base never replaced the regular chart for daily use by clinicians, it did serve it's purpose in providing accurate information about the medical service being provided.

From previous work on the problem lists [LYN75], the number of entries was set at about 130,000. The average length of an entry in a problem list is two words. The most common was "general care" which is to be expected in a health maintenance clinic.

An initial dictionary and spelling correction list of 2000 defined words and 4000 misspellings had been manually compiled. Each word was defined in terms of the system and/or function code it implied. The accuracy of the dictionary was never tested until this thesis work was begun. More importantly the dictionary was not designed to function with any specific encoding procedure. Any dictionary must be keyed to the person or procedure which will be referencing its entries. In spite of these shortcomings it was planned to use both the definitions and the misspellings as the basis for the dictionary needed for this work.

The CHCP data-base is derived mainly from clinic visits over the period October 1971 to October 1974. These were recorded on the standard form and then keypunched. About 140,000 visits by 16,000 patients are stored. CHCP serves employees of participating businesses in New Haven and surrounding communities, an environ of about half a million residents. The participants in the health care plan are mainly white middle class families. The twelve most common problems were general

care, pain, upper respiratory problems, visual problems, trauma, fever, malaise, hypertension, rash, otalgia, and obesity [LYN75].

III -- Review of the Literature

Before considering work done to automatically code medical phrases it is useful to review work in natural language procesing and medical coding which form the foundation for automatic medical coding. In spite of the ad-hoc nature of much of the automatic coding, linguistics can provide a conceptual framework. A medical coding system provides a form for output and may also provide a structure for medical semantics and medical knowledge within an automatic coding program.

III.A -- Natural Language Processing

Natural language processing covers a variety of computer techniques that all take prose directly as input [DAM76]. This includes programs that "understand" english and the work that continues to improve their performance [RAP76]. It also includes a wide range of work from concordance compilation to automatic literature indexing. Verbal speech processing might also be included although this is usually taken separately from techniques that assume typed input. Handwriting recoginition is also often not included.

Historically the impetus for language processing programs was interest in automatic translation. Partially spurred by the cold war, the initial hope was for automatic translation of russian texts. Although a lot of time and money was spent in the late 1950's and early 1960's, quality translations were never achieved. At least one of the programs developed is still in use providing rough translations of scientific literature [JOR77]. Perhaps the best known results of this

work are the jokes about a paper concerning "hydraulic rams" which translated into a paper on "water goats".

The attempt at automatic translation should be credited with stimulating a number of advances in linguistics and computer science. Mathematical linguistics dates from this period. The complexity of the programming involved led to the development of a new programming langage, COMIT [YNG72].

Chomsky's mathematical formalization of language provided linguists with a more powerful tool for dealing with grammar and sentence formation [KIM73, CHO57]. Further, it was rapidly introduced into computer science as a means to describe computer languages. Both natural linguistics and computer language developement benefitted from the studies which followed on automatic parsing based directly on the formal description. Briefly, the formal description of a language's grammar consists of four parts: a starting symbol, a list of production rules, a set of intermediate forms, and the vocabulary or set of symbols which will finally form the sentence. For english the intermediate forms could be entities such as <noun clause>, <predicate>, and <prepositional clause>. The production rules indicate how to proceed from a sentence vacuously comprised of the start symbol to various combinations of intermediate forms and finally to a well formed sentence. An example follows:

```
Start Symbol: <START>
  Vocabulary: a after and broke came crown down fell fetch hill
              his Jack Jill of pail ran the to tumbling water up
Intermediates: <DETERMINANT> <INTRANSITIVE VERB> <TRANSITIVE VERB>
              <NOUN> <OBJECT CLAUSE> <PREPOSITIONAL PHRASE>
              <PREDICATE> <SUBJECT>
```

```
Productions: <START> -> <SUBJECT> <PREDICATE>
             <START> -> <SUBJECT> <PREDICATE> and <START>
             <SUBJECT> -> <NOUN CLAUSE>
             <SUBJECT> -> <NOUN CLAUSE> and <SUBJECT>
             <NOUN CLAUSE> -> <NOUN>
             <NOUN CLAUSE> -> <DETERMINANT> <NOUN>
             <NOUN CLAUSE> -> <DETERMINANT> <NOUN>
                       <PREPOSITIONAL CLAUSE>
             <NOUN CLAUSE> -> <NOUN CLAUSE> and <NOUN CLAUSE>
             <PREDICATE> -> <INTRANSITIVE VERB> <ADVERBS>
             <PREDICATE> -> <INTRANSITIVE VERB>
                       <PREPOSITIONAL CLAUSE>
             <PREDICATE> -> <TRANSITIVE VERB> <OBJECT CLAUSE>
             <PREDICATE> -> <PREDICATE> and <PREDICATE>
             <PREPOSITIONAL CLAUSE> -> <PREPOSITION>
                       <NOUN CLAUSE>
             <PREPOSITIONAL CLAUSE> -> <PREPOSITION> <PREDICATE>
             <PREPOSITIONAL CLAUSE> -> <PREPOSITIONAL CLAUSE>
                       <PREPROSITIONAL CLAUSE>
             <ADVERBS> -> <ADVERB>
             <ADVERBS> -> <ADVERB> <ADVERBS>
             <DETERMINANT> -> a his the
             <NOUN> -> crown hill Jack Jill pail
             <INTRANSITIVE VERB> -> came fell ran went
             <TRANSITIVE VERB> -> broke fetch
             <PREPOSITION> -> of to up
             <ADVERB> -> after down tumbling
```

This grammar is capable of generating the well known nursery rhyme:

Jack and Jill went up the hill

to fetch a pail of water.

Jack fell down and broke his crown

and Jill came tumbling after.

This particular grammar is quite inadequate as it also generates:

A crown and the hill went to Jill.

Water broke tumbling after down.

The importance of Chomsky's formalization was not that it adequately described english or any other natural language. Rather, it provided a framework for classifying languages in terms of complexity

and the types of mechanisms that would be required for parsing. Grammars could now be classified into four major types. Basically, type 0 grammars are unrestricted and thus the most difficult to parse. Type 3 grammars are the most restricted and easily parsed. Programming techniques have developed to the point that certain type 2 (context-free) grammars may be easily parsed. Natural languages are type 0 and no general programs for parsing them are available.

Transformational grammars based on type 2 base grammars plus a collection of transforms were developed in an attempt to generate english without resorting to a type 0 grammar. As this approach has not proven fruitful in automated language parsing, although it is linguistically interesting, it will not be discussed.

Yngve devloped COMIT, the forerunner of SNOBOL and other string processing and pattern matching computer techniques for linguistic work [SAM69, YNG72]. Previous computer languages had only allowed operations on characters and simple groups of characters. COMIT was the first language to allow easy manipulation of strings and substrings as required for linguistics. The fundamental COMIT statement was based on the production rule of formal grammars. An intermediate form could be selected and replaced with its expansion. If this was done repeatedly, sentences would be formed. However, it was also possible to work in the reverse direction: portions of sentences could be matched and replaced with a symbol indicating the type of clause. If this was successfully repeated a parsing would be obtained.

A major improvement in natural language processing occurred with a

balanced approach utilizing both semantics and formal syntax [MCC68]. Reffering back to the example presented earlier, consider the sentence:

A crown and the hill went to Jill.

One possibility is that this is ungrammatical. A more adequate grammar would classify "went" as verb requiring an animate subject and if an object is present, one that is a place. This leads to a proliferation of word classes. Another possibility is to consider the sentence grammatical but obviously false: it describes a situation which could only happen in nursery rhymes (Hey diddle diddle, the cat and the fiddle, the cow jumped over the moon ...). Some early programs which took natural language input operated almost solely on semantic clues [GRE63, LIN63]. This is not to imply that semantic approaches should have primacy over syntactic ones or vice versa. For certain types of input one may be much simpler than the other.

Winograd's SHRDLU program is the classic example of a program which is sophisticated about accepting english input [WIN72]. SHRDLU "understands" commands and questions about a collection of children's blocks. It operates in this very restricted "play" environment so that it can semantically disambiguate and verify the meaning of its input. If told to

"pick up the red block on top of the green one"

it can check the coordinates of all the objects it has noted as red blocks and see which if any are located above an object noted as a green block. In every day conversation the sentence

Harry ran to the ball

is understood to mean "Harry ran to the dance" or "Harry ran deep into left field to get to the baseball" depending on what the listener remembers of the previous conversation.

The SHRDLU program incorporated grammatical rules in the form of a program specifically for grammatical parsing. While the parsing proceeded primarily on syntactic clues, references were made to the semantic model of the blocks to test the appropriateness of alternative parses. Since then the favored approach to specifying semantically guided syntax rules has become the use of augmented transition networks (ATN). The sketch of an ATN looks like a finite state machine diagram. Unlike a finite state machine, an ATN may recursively invoke other ATNs, and even itself, to parse a sentence [WOO70]. Futher, when making state transitions an ATN may change the contents of storage registers which will affect later states. This facility can be used to test aspects of the semantic model and modify the model. The major limitation is that the semantics of all but carefully chosen subject matter has proven intractable.

A notable program dealing with everday discourse was put together by Schank. It exploited a list of primitive actions and relationships which had to be assembled into a consistent stucture before the program would claim to have understood a sentence [SCH73]. The sentence

'John gave Mary an apple to eat

generates a structure using the primitives give-physically-transport twice and ingest once. The first give-physically-transport describes the apple's motion from wherever it was to Mary through John's action. The

second describes the apple's motion to Mary's mouth which is a pre-condition for the ingestion primitive. There is a different "give" primitive to describe

John gave Mary a headache.

This approach to semantics is very appealing in that it allows a natural formulation for requirements such as ingestion presuming that a food type object has been transported to the subject's mouth. Defaults can be listed for objects of the primitive actions. Ingestion normally applies to edibles. The normal defaults could be changed to suit the context of the conversation, be it MacDonald's or the Waldorf-Astoria. It has been suggested that plans of actions covering standard situations are stored as "frames" retrieved in appropriate contexts [MIN75].

Over the same period of time as the work directly aimed at "understanding" prose, techniques were developed to deal with prose on a more limited basis [BOR68, THO75]. Classical scholars were interested in studying concordances. With automation, frequencies of word usage and other statistical parameters could be measured which were previously much too time consuming. Programs to perform these statistical measurements were much simpler than those attempting to determine the "meaning" of their input. Usually no syntactic parsing was performed so that the two sentences

Smoking is not good for you and is expensive.

Smoking is good for you and is not expensive.

would appear identical in terms of word counts. This may or may not be a problem depending on the subject matter and the reseacher's particular

interest.

The tremendous volume of published scientific literature spurred developement of automated indexing techniques. Statistical procedures were developed to find key words and phrases. These programs also operated without any "understanding" of the text. The basic approach was to identify those words or phrases which occured often enough to represent something integral to the subject matter, but not so often as the common words of the language or fundamental terms in the field.

The Linguistic String Project attempted to improve the capabilities of automatic indexing by using a linguistic approach. A type 2 grammar was developed which would be common to any english scientific text. Restriction rules were formulated to guide parsing much as the semantic guides described above. Small modifications to the grammar and to the restriction rules could easily be made if required for peculiar word usage or jargon. Making use of the syntactic clues for parsing, the indexing program could make note of all unrecognized vocabulary and then try to determine its gramatical class by statistical techniques [SAG72]. The Linguistic String Parser is quite sophisticated. Its major shortcoming as a general approach to english language understanding is that it forms no semantic model. Instead it attempts to group words in sufficently well refined catagories to avoid reading

John eats today

as "John ingested Monday instead of something more substantial" [SAG75].

Readers are probably aware that of all the approaches tried, author supplied keywords and manual indexing by trained readers are the main

indexing systems in use at present. The citation index is also very popular. It is also based on direct use of author supplied information rather than statistically culling through the text.

Content analysis was another area which applied natural language processing. The General Inquirer was designed to provide consistent measures of content in written or transcribed language specimens [STO66]. A dictionary giving the general import of words to be checked has to be supplied along with the text to be scanned. With a dictionary that rated "murder", "mugging", "rape", and "arson" as having high violence content it was possible to scan news articles and rate their relative violence content.

Text processing systems are perhaps the simplest examples of natural language processing. Their purpose is to reduce the human effort required to prepare printed documents. So-called "word" processing systems are becoming popular in business offices. These typically allow a document to be rough typed, edited, and then a final copy made without overall retyping. After changes have been marked on the rough copy they can be incorporated in a machine copy, usually on magnetic tape cassettes or card. Only the changes and new text need be typed before making the final copy. More sophisticated systems automatically adjust the right margin, hyphenate when necessary, paginate, and make font changes. This results in a significant reduction in labor and an increase in accuracy of the final printed text when used for papers or books.

While no existing system "understands" discourse over as large a

domain as most people compass in the course of a day, there are an increasing number of systems which accept english input within a well delineated subject area. This greatly simplifies their use. Primarily this has been done for data-base systems to make their information available to relatively untrained personnel. This trend will undoubtedly continue.

III -- Review of the Literature (cont)

B -- Medical Coding

Originally, medical coding systems allowed uniform tabulation of international mortality statistics. Presently they are increasingly oriented towards indexing clinical data for research [WHO77, PRA73]. To some extent this echoes the increasingly microscopic and molecular focus in medicine.

III.B.1 -- Case Retrieval without Coding

The problem of retrieving pertinant clinical records can be handled the same way as retrieving pertinant articles in scientific literature. If key-words are automatically culled from the medical text no classification system is needed at all [LAM66]. A thesaurus turns out to be very helpful for formulating retrieval requests for this sort of system. Indeed, as the thesaurus becomes more sophisticated, listing close and distant synonyms, subsuming and subcategories, it becomes a form of coding system. The system referred to, [LAM66], was used for a large pathology report collection with good results. A notable shortcoming was the inability to recognize negation. This is not suprising as no grammatical parsing was performed.

III.B.2 -- ICD: International Classification of Diseases

The International Classification of Diseases (ICD) and its adapted forms are the most widely used coding systems. Presently in its ninth revision, ICD is maintained by the World Health Organization. In

summarizing the goals of this classification scheme the authors quote work on British mortality figures from the 19th century:

> "The aims of a statistical classification of disease cannot be better summarized than in the following paragraphs written by William Farr a century ago:
>
> [Registrar General of England and Wales, Sixteenth Annual Report, 1856, Appendix, 75-76]
>
> 'The causes of death were tabulated in the early Bills of Mortality (Tables mortuaires) alphabetically, and this course has the advantage of not raising any of those nice questions in which it is vain to expect physicians and statisticians to agree unanimously. But statistics is eminently a science of classification; and it is evident, on glancing at the subject cursorily, that any classification that brings together in groups diseases that have considerable affinity, or that are liable to be confounded with each other, is likely to facilitate the deduction of general principles.
>
> 'Classification is a method of generalization. Several classifications may, therefore, be used with advantage: and the physician, the pathologist, or the jurist, each from his own point of view, may legitimately classify the diseases and the causes of death in the way that he thinks best adapted to facilitate his inquiries, and to yield general results.
>
> 'The medical practioner may found his main divisions of diseases on their treatment as medical or surgical; the pathologist, on the nature of the morbid action or product; the anatomist or the physiologist on the tissues and organs involved; the medical jurist on the suddenness or the slowness of the death; and all these points well deserve attention in a statistical classification.
>
> 'In the eyes of national statists the most important elements are, however, brought into account in the ancient subdivision of diseases into plagues, or epidemics and endemics, into diseases of common occurrence (sporadic diseases), which may be conveniently divided into three classes, and into injuries, the immediate results of violence or of external causes.' " [WHO77]

ICD may legitimately trace it origins back to the Bills of Mortality and after nine revisions still serves the same purpose.

In this country ICDA (International Classification of Diseases -- Adapted) maintained by the U. S. Public Health Service and ICDA-H

(Hospital Adaptation of ICD) maintained by the Commission on Professional and Hospital Activities are the most popular coding systems [CPH73]. Salient characteristics of all the ICD based codes are:

they are linear lists

they are oriented to statistical tabulation, not individual case retrieval

they are inconsistent about sign and symptom code placement

Classifications based on linear lists are limited in their ability to group "like" entities. It is not possible to arrange the list so that pneumonia is at the same time close to lung cancer and also to cholera. As Farr noted, it not possible to satisfy both the anatomists and the microbiologists.

A classification scheme oriented to statistically significant divisions is likely to gloss over the individual findings in a medical case in favor of the final diagnosis. The problem is one of purpose. Mortality tables are used at a national level and so emphasize diseases of fatal or very morbid outcome. A reseacher attempting to establish the validity of a particular diagnostic finding requires access to all cases regardless of morbidity or mortality.

The problem with ICD's coding of signs and symptoms is also related to it original purpose and orientation. It has only been of late that there has been increasing interest in the common-place findings which cause little gross morbidity but occupy a fair portion of physicians' time. Having been added only of late these codes often are chosen from the end of the list. The reader should examine the coding examples given

in the following table:

COUGH	COLD	SORE THROAT	STREP THROAT	
779.3	460	777.6	034.0	ICDA-H
F7131	E9314	M4100 T2410	E1681 T2410	SNOP
S4244 F2C20	S4000 F4420	S4600 F3260	S4600 F4440	Clinical Event Coding

-- Coding Examples --

III -- Review of the Literature (cont)

B -- Medical Coding (cont)

3 -- SNOP: Systematized Nomenclature of Pathology

The Systematized Nomenclature of Pathology is a well established classification for pathology findings which was designed specifically to facilitate automated case retrieval. SNOP does not use a simple list of codes as does ICD. It is multi-dimensional in that it classifies along more than one axis. From the introduction to SNOP [CAP65]

> "Diseases may be defined in terms of four areas of information: 1) the part of the body affected (Topography); 2) the structural changes produced (Morphology); 3) the etiologic agent (Etiology); and 4) the functional manifestations (Function). This code is divided into four separate, interdependent fields comparable to these areas: Topography, Morphology, Etiology, and Function. Within a field, terms are assigned a four-digit number. The first (left hand) digit indicates the section of the field. The other numbers indicate progressively finer subdivisions. These grouping relect, as far as possible, natural relations. This structure and organization are given in the Table of Contents and numeric portions of the code. An alphabetic listing of terms is included to permit coding. "

The major benefit of multi-dimensional code is that it allows pneumonia to be assigned a code close to that of lung cancer along one dimension while being assigned a code that is close to that of cholera along another dimension. This is especially well suited to the anatomic and surgical pathologist who may wish at one time to compare a specimen to other specimens from the same site in the body and then at another time wish to compare a specimen with other specimens of similar morphology. This would be of little help in preparing mortality tables. Statistical tabulations could be made from SNOP indexed data by

preparing a list of code groups to be summed together. Since SNOP codes make finer divisions of the data it would in principle be possible to prepare ICD type lists from SNOP coded cases.

The other major feature of SNOP is its use of hierarchical code assignments. This allows retrievals of not just specific entities but also subsuming or subcategories. This is true to some extent of ICD as well. Lampson's work [LAM66] achieved this capability only through the use of a relatively complicated thesaurus. Appropriately assigned numeric codes require only that the retrieval request indicate the number of digits that are desired for a match. A match of only the left most digit retrieves any grossly related case. A match of all four digits retrieves only identically coded cases.

The main shortcoming of SNOP is that it does not code signs and symptoms. This is an unfair criticism in that SNOP was never meant to serve as a coding system for general medicine. Refering back to the table of coding examples (end of section III.B.2) the reader will note the wide variety of codes assigned to common clinic upper respiratory complaints. It may be additionally noted that M4100 is a morphologic code for inflammation which is not quite the same as "sore" in "sore throat".

III.B.4 -- Clinical Event Coding

Clinical event coding was designed to classify all of the medical information collected in a general clinic in a form facilitating atuomatic storage and retrieval [BRU71]. This approach has already been

described in section II.B.1 but a short discussion is included here.

Like SNOP, clinical event coding is multi-dimensional and uses a hierarchical assignment of code values. The major differences are that clinical event coding allows more dimensions to cope with the greater diversity of data in a general medical clinic and a special effort has been made to cogently assign values to common signs and symptoms.

Clinical event coding uses four major dimensions and seven minor dimensions. These are listed below with the major dimensions first:

System -- personality, respiratory, digestive ...

Topography -- head, neck, abdomen, ...

Function -- pain, inflammation, fracture, ...

Etiology -- streptococcus, lye, ...

Quantity -- 104 degrees F, 10 pounds, +++ ...

Flag -- chief complaint, problem number, ...

Time -- 3 days ago, 4/6/54, ...

Modifiers of Signs and Symptoms -- aggravated by ___, relieved by ___, ...

Function Modifiers -- increasing, stable, improving, ...

Source -- patient, family, witnesses at accident, ...

Who -- patient, siblings, relatives, ...

Other -- related to event ___, see dictation, ...

Of the major dimensions, three are very similar to SNOP. Topography and Etiology are taken directly from SNOP. Function is reorganized and shows an orientation to a general clinic. System, however, has no SNOP counterpart and Morphology is not used as a clinical event dimension.

The separate System dimension allows accurate coding of data where the Topography is known, for example a patient complaining of lower abdomenal pain, but whether the digestive or urogenital organs are involved is not known. Similarly, complaints about changes in personality may be properly assigned a System without having yet determined if any organic lesion is present.

The minor dimensions are not all well-formulated although the basic information to be recorded is indicated. The Source and Who dimension solve problems of differentiating nursing notes from family reports on information that may pertain to the family or the patient and yet allow simple retrieval of all data.

The Other dimension if used for reffering to other recorded events could allow coding of relationships like "secondary to". This becomes important in a coding scheme which strives to code entire medical records.

An additional minor dimension RLBUML has been suggested as a adjunction to the Topography dimension. The name comes from the first letters of "right, left, bilateral, upper, middle, lower".

III.B.5 -- Other Coding Systems

A number of codes are in use: COMIT, local Blue Cross classifications, SNO-MED, and others. None of these present a significantly different approach from the major codes presented. Medical coding systems are not static. New disease syndromes are described. New, hopefully more fundamental,disease relationships are discovered, and different

applications require greater or lesser detail in different areas.

III -- Review of the Literature (cont)

C -- Automated Medical Coding

1 -- General Observations

While automated translation or encoding of medical phrases falls within the realm of natural language processing, much of the work has been done in a quite ad-hoc manner. There are a number of reasons for this. Diagnostic phrases are usually very short requiring little or no syntax analysis. The semantics of medical statements have not been well formalized. Medical coding is usually undertaken without the computational tools and expertise brought to bear on computer linguistic projects.

In spite of the short-comings in this work, the results have been of practical utility. Medical coding is an onerous task. Any scheme that automatically processes a fair percentage of its input is helpful. Once the medical records have been indexed, chart studies can be done that would otherwise be impossible.

The major approaches for automated encoding will be described in historical order. These demonstrate increasing complexity and sophistication but suprisingly decreasing comprehensiveness.

III.C.2 -- The "Fruit Machine"

The so-called "fruit machine" method [HOW68, GRE72] was one of the earlier automatic encoders. It was very clever. Its approach can not really be described as either semantically or syntactically based. Perhaps it is best described as a phrase retrieval system. It dealt with

phrases as irreducible entities.

The procedure is aptly descibed by the authors:

> "In the conventional fruit machine [one-armed bandit], the 'jackpot' is obtained when the lemons appear in line. Similarly, in this method of diagnosis coding, the 'jackpot' (the correct code number) is obtained when a code number appears which is common to all words in the diagnsosis (Fig. 1) [below]. In the main fruit machine dictionary each significant word of a diagnosis is stored with all the code numbers with which it has been associated.

```
  Acute 727-      Appendicitis 552-      Perforation 603-
  Acute 600.0     Appendicitis 551-*     Perforation 578-
--Acute 550.1-----Appendicitis 550.1-----Perforation 550.1-->
  Acute 550.0     Appendicitis 550.0
```

> "Fig. 1. -- Computer coding of the diagnosis 'Acute Appendicitis with Perforation' " [HOW68]

An asterix marks the appropriate code for single word phrases. In this sample of the dictionary 551- is the prefered code for the phrase "Appendicitis".

The procedure is notable for its speed and simplicity. However, it becomes apparent that this is just an algorithm for recognizing permutations of catalogued phrases!

As noted before, this procedure uses neither semantics nor syntax. It incorporates no knowledge of medicine except that it only recognizes the phrases listed in its dictionary.

The "fruit machine" method makes use of three word lists or dictionaries. The primary dictionary consists of entries listing a word followed by all the codes assigned to phrases in which the word has appeared. A secondary dictionary is required to resolve cases where the appropriate code is not apparent from the matching process alone.

"An example of a cross-over which occurs in practice is shown in Fig. 2 [below]. As 'myocardial insufficiency' is coded as 422.2 and 'myocardial infarction' as 420.1, both code numbers are stored in the dictionary with 'myocardial'. Similaryly, 420.1 and 422.2 for 'insufficiency' are derived from coronary insufficiency and, of course, from myocardial insufficiency.

```
  Myocardial 431-     Insufficiency 578-
--Myocardial 422.2----Insufficiency 422.2-->
--Myocardial 420.1----Insufficiency 420.1-->
  Myocardial 197.1
```

Fig. 2.--Coding of 'Myocardial Insufficiency' by the fruit machine dictionary." [HOW68]

The third dictionary is a list of words to ignore such as "with" and common synonyms such as "malignant neoplasm", "ca", and "carcinomatous" all of which are treated as "carcinoma".

The only preprocessing which is performed on the phrases is to remove parenthetical comments. This allows information for other purposes to be keypunched on the same card.

The "fruit machine" makes no provision for spelling correction. Common misspellings could be entered as synonyms. A number of approaches to spelling correction have been tried in other applications [MOR70, ALB67, DAM64]. The two basic techniques are to develop some measure of "closeness" to allow the selection of a word from the dictionary that might be the one intended or to extract the "important" features of the misspelled word, its consonants for example, and match those to a dictionary entry. To some extent it is suprising that spelling correction algorithms are not a common part of computer language compilers. Compilers process large amounts of human typed text.

There are two major drawbacks to the "fruit machine" approach.

There is no "leverage". Each entry in the dictionary allows only one new phrase to be encoded. Indeed if cross-over occurs a second entry may be necessitated to generate the approriate encoding. The second shortcoming is that the technique can not cope with phrases containing two diagnosises.

Some improvements could easily be added. Simple spelling correction could be included. The simplest would be to allow a word not found in the dictionary to match the first word found differing by only one letter. Preprocessing to remove common suffixes would reduce the number of dictionary entries required.

To conclude, the notable features of the fruit machine approach are listed:

simple to program

fast execution

dictionaries may be built-up slowly as needed

proven effectiveness and accuracy of about 95%

III -- Review of the Literature (cont)

C -- Automated Medical Coding (cont)

3 -- The NIH Pathology Diagnosis Encoder

This program is used at the National Institues of Health to classify pathology reports according to SNOP catigories [DUN77, PRA73]. The coded pathology reports are then entered into an automated system for case retrieval. The NIH Encoder is based on an unconventional approach of matching the input phrases to the english text used to describe the SNOP categories.

The encoding procedure involves a number of steps. First punctuation, prepositions, and phrases like "due to" are marked. Then words are looked-up in the exception dictionary. The exception dictionary includes the following sorts of entries:

feet -> foot, plural noun

renal -> kidney, noun

Keep in mind that the program must translate words into the standard vocabulary found in the SNOP manual. Words still remaining are categorized by their endings to determine the root form, part of speech, and associated forms [PRA69, DAM76, KLE63]. As a simple example

cortical -> cortex, adjective

After all the words have been examined a right to left scan is performed. The program ignores preposition and other phrase delimiters at this point and simply seeks to find a word which matches the first word of one of the standard SNOP phrases. The right to left scan saves time because the key-word of a noun often is the last word. A version of

this program for processing french [WHI77] scans from left to right since in french adjectives usually follow the noun.

Once a word has been found which matches the beginning of a standard phrase the programs scans in both directions around the word found to match other words to the standard phrase. Three words on either side are checked, but the program will stop sooner if a word or punctuation marked as a phrase delimiter is encountered.

Words from the input do not have to precisely match standard phrase words. Having been reduced to their root forms, it sufices that the root or the root with a standard suffix matches the words found in the standard phrases.

Scanning continues until all the words have been matched into one or more phrases. At this point the codes associated with the matched phrases are merged together to form SNOP quadruples (Topography, Morphology, Etiology, and Function). After the basic quadruples have been formed syntactic clues are used to allow negation and phrases like "metastatic to" to operate on the formed quadruples.

The NIH Pathology Diagnosis Encoder operates with a simple semantic model. Medical information contained in a pathology report is presumed to consist of a collecton of quadruples specifying values along the four SNOP dimensions. If a value has not been supplied at least for the Topography and Morphology or Function dimension then either there has been a missunderstanding or the input is not well formulated. Alternatively if more than one value has been specified for a dimension then presumebly more than one complete pathology finding is being

described.

A number of dictionaries are used in the first phases of encoding. They are briefly described below:

Full Word List -- contains entries giving the root form and part of speech for words like "feet"

Word Endings List -- a list of common suffixes, how to derive the root form, part of speech, and alternate suffixes that may be used during matching

Sepcial Terms -- lists punctuation, prepositions, and words like "probably"; phrase delimiters are included

Non-Key Word List -- contains "tissue", "space", and other words which can never be key-words in a standard phrase

Ignorable SNOP Words -- Words used to describe SNOP classifications that would never be used in actual pathology reports

No preprocessing is necessary. Pathology reports are directly submitted to the program; neither is any attempt made at spelling correction.

The most significant shortcoming of this approach is that input vocabulary is limited by the vocabulary used to describe SNOP codes. Some training is required of the pathologists. For usage in a general medical setting the limitations of SNOP would also be a problem.

The only simple enhancement that could be made of this system would be to add spelling correction. Some extension of its input vocabulary could be achieved by adding entries to the full word dictionary.

In conclusion the notable features of the NIH Pathology Diagnosis Encoder are :

works with a well tested coding and retrieval system

has the capacity to reject some forms of nonsense

has the capacity to parse multi-part phrases

recognizes some syntactic forms including negation

performs very complete suffix analysis

III.C.4 -- X-ray Report Coding: The Linguistic String Parser

This work utilizes a highly developed syntactic parser, giving it the ability to analyze full sentences as well as sentence fragments [HIR76, SAG75, GRI73, SAG72]. Conceivably, an extension of this work could encode daily notes in a medical chart. The Linguistic String Parser was discussed earlier in section III.A.

To parse sentences from x-ray reports, the Linguistic String Parser is primed with a context-free grammar and a list of restriction rules. The context-free grammar is only a slight adaptation of one used for parsing scientific english. Some modification is necessary since radiologists commonly write

Chest x-ray -- negative.

Other sentence fragments and "ungrammatical" forms are also common. This particular form is handled by treating "--" as a verb. The fragment then becomes

Chest x-ray is negative.

The restriction rules have to be specific for the subject matter.

"Being" can not act as a noun in the sentence

No report of x-rays being taken.

In science fiction writing one might encounter

No report of inter-galactic being taken on the evening shift.

The parser then accepts x-ray reports and parses them according to the grammar and restriction rules. The context free grammar by itself would generate many unacceptable parses. The actual output is created by performing a series of formating transformations on the final parse tree.

The output format was formulated by examining the sentences in the input sample and noting what kinds of information are recorded. The final form contained about 30 different headings. It is much too detailed to present here, but the overall effect is much the same as breaking the statement of a medical finding into the dimensions of a clinical event. For a more detailed description of the encoding process the reader is reffered to [HIR76]. Sager's work, [SAG75, SAG72] should be consulted for more details about the Linguistic String Parser.

As mentioned before, the Linguistic String Parser does not operate with a semantic model. Semantic rules are incorporated into the restriction rules for the subject area. This is an advantage in that it does not require programs to be written to manipulate and test interelationships in the semantic model to verify the approriateness of a particular parsing.

The authors do not explicitly discuss any dictionaries used. However, the dictionary is probably a list of words with an indication

of their part of speech.

No preprocessing of the x-ray reports was required, and no spelling correction was performed. This may be less of a problem with dictated reports like x-ray reports.

The major shortcoming of this work was the limited scope of the reports processed. The input sample consisted of less than 200 x-ray reports on patients followed after treatment for breast cancer. Further, almost half the reports were a simple phrase indicating no findings. Another problem is the very non-standard output format.

Simple improvements that could be made would be some provision for spelling correction. Output of standard code should be possible with appropriate revision of the output transformations.

In conclusion notable features of this work are:

it builds on a general system for parsing english and can benefit from improvements in those techniques

its strong syntactic underpinning easily handles problems of negation and conjunction, and makes possible parsing of forms like "... leading to ..."

it may be possible to process present medical records verbatim

IV -- General Approach

A -- Clinical Event Coding as the Semantic Model

An ad-hoc procedure based on the semantics of the clinical event model was chosen for the encoder developed in this thesis. Examination of the problems lists showed little syntactic structure. Normally two word noun phrases appeared. Use of syntactic rules would undoubtably be useful with longer phrases, but seemed an unnecessary complication for problem list coding.

Both [HIR76] and [PRA73] indicate that the meaning of simple medical phrases can be adequately recorded in a form similar to that of the coded clinical event. As in the NIH Pathology Diagnosis Encoder it would be possible to recognize well formed encodings when they were obtained. Phrases composed of two separate clinical events could be detected and encoded as was done in [DUN77] and [WHI77]. Beyond this it would be possible to recognize obviously false encodings such as "hemorrhage of the personality system". Presumebly no physician would write such a phrase so the encoding program could assume it was pursueing the wrong parse and try another.

The encoder could first reduce phrases from problem lists to coded clinical events and then process negation and conjunction words if present. "And" would imply that two clinical events share some dimensions in common as in "back and leg pain" which should encode to "back pain" and "leg pain". "Not" or "no" could be taken to mean that either an entire clinical event was not reported or be taken to modify one dimension. Consider

no back pain

hypertension, no change

Using the coded clinical event as the semantic model would also provide another test of its adequacy for representing medical information. Succesful construction of the encoder would at least imply that some useful amount of information was represented and could be manipulated as coded clinical events. It should be noted that it would not mean the same thing to simply output encodings as coded clinical events. That would just indicate that some transform could be programmed which converted the encoder's internal structures into clinical event format as was done in [HIR76].

IV.B -- Words in Context

A major difficulty in processing natural language is that it is based on a context sensitive grammer. The simplest example of context sensitivity appears as idiomatic word usage. This is quite common in medical problem lists. Although it does not appear in the problem lists,

cold sore head cold

serves as a good test case for idiomatic word usage and at the same time has a number of possible parsings.

"Cold" in the example takes on two meanings: "cold sore" implies a herpetic lesion and "head cold" generally means a viral respiratory infection. Within the Connecticut Health Care Plan problem lists "cold" is also used in the phrases "cold feet" and "cold thyroid nodule". Only in the phrase "cold feet" does "cold" take on its normal meaning.

The other two words in the example appear to take on their common meanings. "Sore" is usually taken to mean inflammed and "head" usually means the body region above the neck. However, "head" in "head cold" really only reffers to to upper respiratory involvement and should be taken idiomatically. "Sore" similarly should be understood idiomatically.

To deal with idiomatic word usage three types of adjacent word interactions were recorded in the dictionary. Hyphenation is the first of these. It is used in defining the words in the example. "Cold" is entered in the dictionary with a number of definitions. One of these begins with the term "H_SORE" which indicates that this particular definition is only valid if "cold" is parsed in the context "cold-sore". Correspondingly, "sore's" definitions include one begining with "Y_COLD" indicating that this is the second half of a HYphenated definition. Similar entries define "head-cold".

The other types of adjacent word interactions are forward linking and back linking. Foward linking is used in creating the dictionary entry for "rheumatic" to allow the encoder to chose the appropriate definition when "rheumatic" is followed by "heart". A different definition should be chosen if "rheumatic" is followed by "arthritis". This later definition begins with the term "F_ARTHRITIS". Back linking is used in the definition of "mellitus" to indicate that the definition is only meaningful if the previous word was "diabetes". The definition for "mellitus" begins with the term "B_DIABETES".

The presence of context sensitive word definitions is by no means a

complete solution. The fact that "... head cold ..." may be parsed as "... head-cold ..." does not guarantee that that is the proper parsing. Consider

hit head cold limbs.

An unlikely problem list entry, but, it should be parsed as if it were an elipsis of "the patient hit his head and presently his limbs are cool".

In parsing any but the simplest subsets of natural language ambiguous phrases will arise. The simple phrase "head cold" is generally taken to mean viral repiratory infection. There must be some way for the people to eliminate the parsing of "head cold" as short for "the patient's head feels cold". Presumebly, interpretations of "head cold" other than the common are eliminated on the basis of knowledge that it is rather uncommon for someone's head to be cool to touch. It is not impossible, but uncommon enough to ignore that interpretation of "head cold" unless the context of the utterance forces it.

A direct approach was chosen to resolve amiguities as noted above. the encoding program includes a routine to reject unlikely or impossible clinical events. This also means that a parsing would be rejected only after it was complete and the clinical events formulated. The relative inefficiency was hoped to be small. The advantage of waiting until the clinical events had been formulated was that medical knowledge to be drawn upon could also be formulated in terms of clinical events.

A large amount of simple, "common sense", medical knowledge can be formulated in clinical event codes. The phrase

social bleeding

is either a mistake or it means "socially triggered bleeding". The clinical event that corresponds to "disruption of blood vessels supplying the social relationships" can be eliminated by a simple matrix which indicates which function codes are reasonable with which system codes. "Bleeding" would not be allowed with "psychological" or "social" system codes. "Cold", meaning reduced temperature, would only be allowed with extremities.

Even without a matrix of permitted function and system codes some very simple constraints on clinical events can be formulated. A function code must be present. This permits the terse problem list entry

trauma

while disallowing

feet

which indicates a body part but nothing else. Then when parsing

cold feet

the reading "upper respiratory infection and feet" can be eliminated because it would require a body part to stand on its own as a problem!

IV.C -- The Dictionary for Encoding

The requirements for the dictionary were that it allow rapid access to definitions of words for encoding and that it also allow rapid revision of definitions when mistakes were noted. Storage and retrieval based on open hashing was chosen to provide rapid access. Dictionary entries were to be stored as variable length text strings to provide

flexibility in dictionary content.

The basic format for a dictionary entry was

Word definition

and if there was more than one definition

Word First definition; Second definition ...

Each definition consisted of one or more parts separated by spaces. Normally a definition would consist of codes such as "F4420" (Function code, infection, not otherwise specified). If a definition was context sensitive the hypenation or linking term would be included with the codes. Definition terms which referenced other words simply included the word as text. No file pointers were permitted within dictionary entries. Since the dictionary used hashed indexing no significant speed improvement would result from using direct file pointers. A benefit of eliminating file pointers within definitions was that the dictionary could be saved for backup storage or shipment to other systems in simple readable form.

IV.D -- Preprocessing

The problem lists had been stripped of extraneous punctuation during previous investigations. This had removed quote marks, parenthesis, brackets, and other symbols that were assumed to be keypunch errors. The preprocessing did leave semicolons in entries such as

hypertension; no change.

It also left periods in abreviations such as

The physicians' use of punctuation was not uniform. While it was hoped that punctuation would provide clues to phrases comprised of more than one clinical event, this did not appear to be the case. In the examples above, the periods and semicolons are in the middle of single events. Since a meaningful role for punctuation could not be discerned, punctuation symbols were entered in the dictionary as words to be ignored. The dictionary was programmed to accept any non-blank sequence of characters as a "word". No difficulty occurs in ignoring the punctuation in the hypertension phrase above, but abreviations proved to be somewhat of a problem as will be described later.

IV -- General Approach (cont)

E -- Spelling Correction

As previously mentioned, a list of misspellings paired with correct spellings had been manually prepared. To incorporate this into the dictionary with a minimum of effort, dictionary definitions were allowed to indicate the correct spelling. This same mechanism would also function for synonyms such as "finger" and "digit".

A dictionary entry for the misspelling "NECL" might be

M_NECK.

The "M_NECK" implies that the dictionary should be referenced again for the word "NECK". For flexibility, definitions found while referencing the correct spelling are appended to the misspelling indicator. This allows abbreviations like "inf" to be defined as

M_INFARCTION; M_INFECTION; ...

The encoder had to be designed so that when a word was being treated as a misspelling or synonym, and thus as if it were another word, context sensitive definitions could function as if the correct spelling were actually present. This too was facilitated by the technique of appending the definition of the word referenced to the misspelling indicator.

IV.F -- Expected Shortcomings

Since the approach chosen ignores syntax it is unable to handle sentences expressing relationships between clinical events. This should not present any difficulty coding problem lists, but does represent a limitation on other possible coding tasks.

The encoder's "knowledge" is initially to be programmed-in. There is no file of information such as the fact that psychological systems can not bleed (in reference to the example in section IV.B). Instead there will be programmed tests for this type of clinical event. This makes it necessary to reprogram and debug any changes. The dictionary is very easy to correct and critera for reasonable clinical events should be equally easy to modify.

As has perhaps not explicitly been mentioned, the encoder must test all possible parses of a clinical event. This may be desirable in that problem list entries should be unambiguous. However, much of the ambiguity found by the encoder will be due to lack of knowledge about improbable clinical events. The most serious problem that will arise is that phrases which at one point were encoded properly may no longer encode properly because new definitions have been added.

IV.G -- Possible Improvements

It may seem premature to discuss improvements to a program even before its performance has been reported. However, on the basis of the design alone one should be able to predict limitations on program capabilities, as noted above, and know which of these may easily be removed.

The simpler syntactic forms such as "... is secondary to ..." and "... metastatic to ..." could be processed in much the same way negation and conjunction are to be processed. First the surrounding clinical events or partial clinical events are encoded and then a separate

routine may attempt to combine or relate them.

A spelling correction algorithm could be implemented. Some spelling correction algortihms would be very difficult to implement using the present dictionary because it is based on hashed indexing. The most interesting possibility would be to develop a method which proposes a number of possible corrected spellings and then determines which leads to the most likely encoding. Presumably some "corrections" would lead to absurd encodings or none at all. A very clever technique would be to examine other parts of the medical record, especially past problems, to determine the most likely meaning of a misspelling.

Suffix processing should be reasonably easy to add as a prepocessing step. Words could be reduced to their root forms and then checked in the dictionary. This might greatly reduce the size of the dictionary. The suffix processing algorithm could be written as in [PRA69].

IV.H -- Notable Features

As was done with the works reviewed, the salient features of the approach outlined are listed:

works with a previously tested coding system

designed for a general medical clinic it should also be applicable to most specialty practices

the vocabulary is not limited to any pre-existing document; words can easily be added

there is "leverage"; each new dictionary entry may make a

number of new phrases understandable

it is possible to reject some nonsense input

V -- Methodology

A -- Hardware and Software Environment

The programming was done in "C", an Algol-like language. "C" is actually a "BCPL" derivative. Unlike "BCPL" it does recognize a few variable types. Like "BCPL" it is not block-stuctured. Only local and global scopes are available.

"C" and all the other software used ran under the UNIX timesharing system on a PDP-11/45. The UNIX system and a goodly number of utility programs were developed at Bell Laboratories [RIT74]. The encoding program was implemented to run as a normal user job which meant it was to run in less than 64K bytes.

A number of string-handling routines were written as well as some list processing routines which operated on strings. Natural language processing is usually more effectively programmed in a list or string based language. LISP or SNOBOL would be the most appropriate. At the time the programming was undertaken, LISP was not available and the only SNOBOL was a subset of version 3 and unsupported. In retrospect it might have been more efficient to have written a simple LISP interpreter rather than programming a package of many small string and list routines.

V.B -- "Encode": The Encoder's Main Control Routine

The top level routine for the encoding program was written as a subroutine to allow the encoder to be used in a variety of circumstances. For initial debugging "encode" was called by a test

routine which accepted a problem list entry from the programmer's console. Later on, "encode" was called by a routine that read successive problem list entries from a file. In the future it will be called by a routine that retrieves problem lists from the CHCP data-base and places the encoded results back into the data-base.

"Encode" takes as its input arguments an entry from a medical problem list and the file number of the dictionary file. It tests all combinations of the definitions of the words in the problem list entry. The logic to run through all the combinations is fully contained within the "encode" routine. The logic for most of the rest of the encoding process is in subroutines which "encode" invokes in the appropriate sequence.

"Encode" owns the main (global) variables which collectively comprise the parse state. At any point in time, examination of the values of the parse state variables will show which definitions the encoder is considering and what clinical events it has formulated. Processing first centers around the Word vector which holds the words of the phrase and their dictionary entries. The simple variable Wordnumber indicates which specific word is being considered. The Clinical_event vector contains one entry for each word. Its entries come to contain the clinical events as they are built-up word by word. The Merged_event vector contains one entry for each word to indicate whether the Clinical_event entry for the word has been merged with the Clinical_event entry for the previous word. If a word's Clinical_event entry has not been merged then either the word defines a complete

clinical event by itself or it is the first of a series of words describing a clinical event which has been built-up. Since the dictionary entry for a word may include a number of definitions the Definition_active vector contains one entry for each word to indicate which definition is being considered.

The actual program is complicated by the fact that "C" does not recognize strings. The Word and Clinical_event vectors are declared as arrays of characters to simulate a vector of strings. Some further confusion results from the fact that "C" starts all array indices at zero. When reference is made to the "first" element of a vector the "zeroth" element is implied.

The basic steps of encoding are as follows --
"Encode" invokes the subroutine "findwords" which arranges the words, punctuation, and numbers making up the problem list entry into the Word vector. Words and punctuation are normally found in the dictionary and the entire dictionary entry is placed into the vector. It is during this stage that misspellings are handled. "Findwords" invokes the "expand_definition" routine to actually look-up the correct spellings or synonyms.

If any of the words do not appear in the dictionary, "findwords" returns an error indication. It places markers in the encoding string so that the encoding string, when printed underneath the input phrase, shows which words could not be found.

Assuming "findwords" was sucessful, "encode" begins the actual parsing using the first definition of each word. "Local_context_ok" is

called to verify that the use of each word's definition does not conflict with any context sensitive terms in the definition. "Local_context_ok" operates in a somewhat obscure manner because it is actually called separately for each word and it does not assume that it can examine the words to the right of the one it has been called to check. This causes no difficulty in testing back links or the second half of a hyphenation definition. However, forward links and the first half of a hyphenation definition are tested during the check on the next word. An exception is made for the last word in a phrase whose forward linked definitions cannot possibly be used in a parsing. "Local_context_ok" is programmed not to look to the right so that "encode" does not have to commit itself to an entire parsing at once. Later-on "encode" will backup and try a different definition for the last word. Since "local_context_ok" approved the parsing one step at a time it will need only check the new definition for the word that is taking on a different meaning.

After the use of a particular definition has been "okayed", "make_clinical_event" copies relevant information from the definition into the corresponding entry in the Clinical_event vector. The "cl_dimension" routine determines which parts of the definition are clinical event coding values which must be copied. Context sensitive terms, gramatical flags for negation and conjuction, and "X_IGNORE" terms are not meant to be part of a clinical event. The key to a term's type is its first two characters. If the second character is an underscore ("_") then what follows the underscore is a word and not

meant to be a clinical event code. The first character indicates the type of value for clinical event codes. The "F" in "F4420" implies that this is a function value.

"Try_to_merge_clinical_events" is then invoked to build_up a more complete coding of the clinical event if possible. If the last Clinical_event entry contained only a system value and the present definition for the word currently being scanned specifies just a function code then presumably these words are acting together and the information can be merged. If the Clinical_event entry built-up for the previous word already specifies a function value then presumably another, separate, clinical event is being described and merging should not take place. The corresponding Merged_event entry is marked true or false accordingly.

After "local_context_ok", "make_clinical_event", and "try_to_merge_last_clinical_events" have been invoked for each of the words an encoding has hopefully been obtained. "Encode" now calls "reasonable_encoding" to determine if the clinical event(s) generated for this parsing is plausible. "Reasonable_encoding" can examine the Clinical_event and Merged_event vectors to determine what the encoding is. This is the routine which embodies whatever medical knowledge or "common sense" "encode" can demonstrate in its work. At present "reasonable_encoding" simply checks each individual coded clinical event to assure that at a minimum it specifies a function. It also tests explicitly for events such as

head cool.

Eventually it should include a matrix of allowed system and function combinations.

The reason that "encode" waits until the very end of a parse to test that its work is reasonable is to allow negation and conjunction processing to occur after the entire basic parse. Negation and conjunction were not actually programmed in the present implementation, but they could be effected as described in the general approach. At the point "reasonable_encode" is invoked the problem list entry has been parsed into simple clinical events or parts of clinical events. The negation and conjunction markers appear between these events for a routine to process. In the parse of

back and neck pain

"back" would appear as a partial clinical event separated by a conjunction marker from the clinical event "neck pain". The conjunction module would note that "back" defines a typography as does "neck". The conjunction must therefore imply that "pain", a function, is to be copied to complete the partial event. Invoking "reasonable_encode" prior to forming the conjunction would eliminate the parsing on the grounds that "back" cannot act by itself as a clinical event. If efficiency demanded it, an "unreasonable_event" routine could be programmed to check much earlier for events or partial events which are totally out of the question. This could be done after "merge_last_clinical_events" is invoked and could cause "encode" to move right on to the next definition.

As "encode" works through all the possible parsings it keeps track

of all the reasonable encodings. It uses the rule that "less is more" to pick the encoding it will finally return. At any one time it really only stores the encoding(s) comprised of the fewest number of separate clinical events. When all the parsings are exhausted "encode" returns the shortest encoding(s) and indicates an error if none was obtained or more than one was obtained. It is an unresolved question as to whether or not there is some better way to chose among alternative encodings of a phrase. If "encode" were being used to code phrases as they were typed-in then perhaps the typist could decide between alternative encodings. To this time, each phrase that has resulted in alternative encodings has demontrated a bug or shortcoming in "reasonable_event".

V -- Methodology (cont)

C -- Auxiliary Routines

A number of auxiliary routines were programmed in the course of developing the encoding program. The most important of these are described here. The routines fall into two categories: those to aid in debugging the encoder and those for dictionary maintenance.

"Test_encode" was the most advanced of the routines used to run "encode". "Test_encode" was a main program which could be run under UNIX given a number of parameters. If no parameters were specified "test_encode" accepted phrases directly from the program console and passed them to "encode". The encoding returned, along with the error code if any, would then be printed. If a file name was given as a parameter "test_encode" would read successive problem list entries from the file and pass them to "encode". Both the problem list entries and "encode's" results were then printed. For further flexibility a beginning and ending line number could be specified along with the file name. If the letter "p" was also specified "test_encode" would pause between lines from the file. It would proceed when a carriage return was struck on the program console. With these options "encode's" performance could be tested against a standard file of phrases while working on a cathode-ray (television) terminal.

The other important debugging routine was "testpt" (test point). "Testpt" served the same purpose as electrical test points in circuit checking. At any point in "encode" or any of its subroutines, if a particular condition should hold or it was simply important that

processing had reached that point "testpt" could be invoked. "Testpt" took as its first argument the name of the subroutine and the name of the particular point within the subroutine separated by a space. Its second argument was a boolean expression testing the condition that should hold. If either the test point had been named at the beginning of "encode's" execution or the condition was found not to hold then "testpt" returned the value true. "Testpt" is actually a function. When it is very first invoked it would inquire at the program console which, if any, test points should always be turned on. When a subroutine invoked "testpt" it checked the value returned and if true would print any appropriate variables. "Testpt" would print the name of the test point before returning to the subroutine so that it was clear where the printing was occuring.

"Testpt's" utility lay in the fact that normally all the test points were off and no debugging printout occured. However, if some previously valid condition was no longer true, printing would occur immediately rather than the program silently continuing with incorrect intermediate values. An additional feature was that if any question arose as to the source of an incorrect encoding, the encoder could simply be run again with appropriate test points turned on to display intermediate steps in the parsing. "Testpt" was not just used with "encode", but with all routines. Normally, test points were established at the entry to a routine and at its exit. At these points the input and output conditions could be checked.

The main dictionary maintenance routine was "update" which could be

used to change entries in the dictionary and also to list the total contents of the dictionary. "Update" operated by invoking the following routines:

"define" -- creates a new dictionary entry or replaces an old one

"delete" -- removes a dictionary entry

"lookup" -- finds the dictionary entry for a word; also used by "findwords" for encoding

"scan" -- retrieves successive entries from the dictionary for listing (the order of the entries appears quite random because hashed indexing was used)

"i_dictionary" -- initializes a file to serve as a new dictionary

"hash_word" -- returns a value in the range [0,1) for any non-blank text string to serve as its index value

As an aside, a "dictionary" was kept of clinical event codes. The value of the code, "F4420" for example, took the place of a word in an entry and instead of a coded definition the english phrase describing the code appeared. This "dictionary" was never used by "encode", but was used by "test_encode" and the dictionary listing formatter to print understandable english along side the numeric codes. Since the dictionary maintenance routines really only expected text strings and not any particular content, both the regular word dictionary and the "code dictionary" could be maintained by the same "update" program. Only the name of the dictionary file had to be supplied.

Readable listings of the dictionary and the clinical event codes were produced by programs which sorted and then formatted the "scan" list produced by "update". The UNIX "pipe" facility was used to take the "scan" listing and run it through the UNIX permuted-index utility program "PTX". The "pipe" facility then carried the permuted listing to the "formatter" routine. The permuted-index utility program is meant for producing keyword-in-context listings. When applied to a dictionary "scan" listing it results first in an alphabetically sorted list of words and then a list of all the words whose definition includes a certain code value. The "formatter" program took this listing and converted it to a form more appropriate for a dictionary.

The UNIX "sort" utility and a second formatting program were used to process the "scan" listing of the "code dictionary". The result was a listing similar to the one shown in section II.B.

VI -- Results

A -- Overall Performance

Manual verification of "encode" indicated proper coding of >80% of the problem list entries from the Community Health Care Plan. For most phrases "encode" requried less than one second to perform the encoding.

Determination of the true number of correct encodings was limited by the manual effort required to check them. Approximately 1000 different phrases where checked. These were the phrases that occured most commonly. If more phrases were examined the percentage of verfied phrases would be greater. The majority of the errors noted were due to incorrect dictionary entries.

As noted, conjunction processing was not implemented. This adversly affects about 0.5% of the problem list entries. About half the time "and" occurs, it occurs in a phrase like

sore throat and runny nose

which "encode" correctly interpretes by ignoring "and".

Negation processing also has not been implemented. This mainly affects phrases like

hypertension; no change.

Only a minor dimension is affected so that the basic coding is correct. The basic clinical event is increased blood pressure. The "no" applies only to the function modifier "change".

VI.B -- Sample Encodings

To demonstrate the operation of "encode" three examples are given below. The shortest is desribed in step by step detail. The steps described are a summary of the information printed when test points are turned on in "encode's" main subroutines.

VI.B.1 -- "Foot pain" is a straight foward phrase for "encode" to process.

a. "Encode" invokes "findwords" to obtain the following from the dictionary:

```
  Word[0]=FOOT SC5C0;
               Y_FLAT SC5CE;
               Y_ATHLETES SA768;
  Word[1]=PAIN F320F;
```

b. "Local_context_ok" will report favorably on the use of "FOOT's" first definition, SC5C0, since no context exits yet for conflict.

c. "Make_clinical_event" will copy the system code SC5C0 into Clinical_event[0].

d. "Try_to_merge_last_two_clinical_events" has nothing to do when invoked by "encode" since only Clinical_event[0] has been set-up.

e. Moving foward to "PAIN" "local_context_ok" will again report favorably. The definitions for "FOOT" and "PAIN" presently under consideration are not context sensitive.

f. "Make_clinical_event" will copy the function code F320F into Clinical_event[1].

g. "Try_to_merge_last_two_clinical_events" will merge the clinical event value set-up by "FOOT" into the clinical event set-up by "PAIN" resulting in SC5C0 F320F being stored in Clinical_event[1]. It will also indicate that Clinical_event[1] now subsumes Clinical_event[0] by marking Merged_event[1] as TRUE.

h. "Reasonable_encode", using Merged_event as the index to the most inclusive clinical events, invokes "event_reasonable". In this case only Clinical_event[1] is examined and with favorable results.

i. "Encode" invokes "record_encoding" to save SC5C0 F320F as a possible encoding. "Encode" notes that it now has an encoding which sums up the entire phrase in one clinical event eliminating the need to save any future encodings requiring more than one clinical event.

j. For completeness "encode" proceeds to consider "FOOT's" alternate definitions. Both of these will be rejected by "local_context_ok." The second and third definitions of "FOOT" are only applicable in the contexts "flat foot" and "athletes foot".

k. "Encode" will return the single encoding SC5C0 F320F and an errorcode indicating that no error occured.

VI.B.2 -- "Cold sore head cold" requires simultaneous recognition of idiomatic word usage and multiple clinical events. While this phrase never appeared in the CHCP problem lists it best demonstrates "encode's" capabilities.

a. "Findwords" determines the following from the dictonary:

```
Word[0]=COLD S4000 F4420;
            Y_HEAD S4000 F4420;
            FA140;
            F_THYROID F2220;
            H_SORE X_IGNORE;
Word[1]=SORE F3230;
            Y_COLD SA238 F4540;
Word[2]=HEAD SC100;
            H_COLD X_IGNORE;
Word[3]=COLD S4000 F4420;
            Y_HEAD S4000 F4420;
            F_THYROID F2220;
            H_SORE X_IGNORE;
```

b. Listed below are the encodings passed to "reasonable_encode". This is done rather than review step by step the operation of "local_context_ok", "make_clinical_event", and "try_to_merge_last_two_clinical_events". The words themselves are used rather than the numberic codes. Brackets are used to indicate the effects of merging into clinical events. "COLD" is to be taken as "head cold". "COOL" is written when "COLD" is to

be understood as in "cold feet".

1) COLD [SORE HEAD] COLD

2) " " " COOL

3) " " [HEAD--COLD]

4) COOL [SORE HEAD] COLD

5) " " " COOL

6) " " [HEAD--COLD]

7) [COLD--SORE] HEAD COLD

8) " " [HEAD COOL]

9) " " [HEAD--COLD]

c. "Encode" dutifully saves #1 - #6 as possible encodings which parse the phrase into three clinical events.

d. "Reasonable_encode" will reject #7 and #8.

e. "Reasonable_encode" will accept #9. "Encode" then notes that this encoding parses "cold sore head cold" into only two clinical events and therefore it can discard all the previous encodings.

VI.B.3 -- "U R I" demonstrates the use of the misspelling definition for abbreviations or synonyms.

a. Because "U", "R", and "I" are separated by spaces each is considered an individual word by "findwords".

b. Initially "findwords" retrieves the following from the dictionary:

```
Word[0]=U M_UPPER;
Word[1]=R M_RESPIRATORY;
          M_ROUTINE;
```

Word[2]=I M_INFARCTION;

M_INFECTION;

M_INFLAMATION;

After "expand_definition" has processed the definitions:

Word[0]=U M_UPPER H_RESPIRATORY X_IGNORE;

Word[1]=R M_RESPIRATORY S4000;

M_RESPIRATORY Y_UPPER S4400;

M_ROUTINE U0000;

Word[2]=I M_INFARCTION F6448;

M_INFECTION F4420;

M_INFLAMATION F420F;

c. The presence of the M_"correct spelling" in the expanded definition allows context checks to work as if a word had truly been replaced by its correct spelling, synonym, or full spelling.

d. Another interesting aspect of this encoding is that three possible clinical events result:

#1 UPPER--RESPIRATORY INFARCTION

#2 UPPER--RESPIRATORY INFECTION

#3 UPPER--RESPIRATORY INFLAMATION

None of these is shorter than the others so "reasonable_event" must be able to eliminate #1 and #3.

e. It would be well to consider preprocesing to condense "U_R_I" and "U.R.I." to "URI" which would be considered one word.

VI -- Results (cont)

C -- Difficulties

A problem arose while developing dictioary entries to code the following problem list phrases:

1 Allergy Shot	7 Allergic Conjunctivitis
2 Allergic Reaction	8 Allergic Rhinitis
3 Pennicillin Allergy	9 Allergic Rash
4 Allergic to Pennicillin	10 Atopic Rhinitis
5 Allergy	11 Atopic Dermatitis
6 Allergies	12 Contact Dermatitis
	13 Dermatitis

The first and second phrases are quite different in meaning from all the rest. #3 through #6 imply that "allergy", "allergies", and "allergic" can be used to describe a clinical event which may optionally have an etiology specified. #7 through #11 show "allergic" and "atopic" acting to modify a word which by itself specifies a clinical event.

Initially "allergic rhinitis" was coded with a different function code than "rhinitis". This meant that for the encoder "allergic rhinitis" was idiomatic and a context sensitive definition required for both words involved. If this tack is taken for all phrases like #7 through #12 a very large number of context sensitive definitions must be stored.

The problem can be resolved if the following situation can be represented:

System: Nasal Passages

Function: Inflamation

Etiology: see clinical event -->System: Immunological

Function: Increased

Etiology: Ragweed

which corresponds to the statement "patient suffers from rhinitis due to ragweed allergy. A similar representation would be used for "diabetic neuritis" which also appears among the CHCP problem lists.

Phrase #12 above, "contact dermatitis" is still problematic. It may mean "dermatitis due to contact with an allergin" in which case it is similar to #11, "atopic dermatitis". Or, it may mean "mild chemical burn" in which case it is best dealt with as an idiomatic expression.

A less serious problem was that of implicit values for dimensions. "Rash" very definitely indicates a function value. However, if no value is specifically indicated for the system dimension, skin should be assumed. A tempting solution would be to implement a new type of context sensitive definition, one that could be used whenever a dimension had not been specified. The tenitive solution was to let the system dimension go unspecified. The best solution is probably to add another step just before an encoding is returned. At this new step default system and typography, perhaps even etiology and other dimensions, could be filled-in. This relates to the next problem to be discussed, coding for effective case retrieval.

"Diabetes" is encoded as:

System: Islets of Langerhans

Function: Decreased

which is technically correct. The problem which arises is, when should a record be retrieved if one of the listed medical problems is "diabetes". With the present encoding it would be retrieved if "diabetes" were requested or, the subsuming category, endocrine disorders. But within a health maintenance organization it should be possible to retrieve all the diabetic cases when scheduling opthamologic exams. The question becomes one of how clever can the data-base be. Certainly it is resonable to make sure that all cases of "rash" are retrieved when dermatologic cases are reviewed. And this could easily be facilitated by added the step described above which would fill-in skin as the default system. For "diabetes" the situation is more complex. Perhaps entire events should be created by default to flag the patient's record for opthamology, podiatry, and the renal unit if these common complications are not explicitly mentioned.

Parenthetically, this last problem raises once again questions of knowledge and language understanding. A common expression on a medical teaching ward is "as soon as you hear 'diabetes' you should immdiately think 'retinopathy, nephropathy, and neuropathy'." Is this also true for an effective encoding program? Is this one of those "frames" of knowledge described by Minsky [MIN75]?

VII -- Conclusions

It is possible to automatically encode problem lists from a general medical practice. This work did achieve its objective of programming a functioning encoder for the Community Health Care Plan problem lists. The estimate of the encoder's accuracy is very conservitively 80%. It will probably demostrate an accuracy well above 90% with further manual verification and minor corrections to the dictionary.

The encoding program presently runs on a time-sharing minicomputer. Processing problem list entries in under one second, it is very inexpensive and could be used interactively for data entry.

The coded clinical event model functions well as a structure for organizing the encoding process. Serving as the encoder's semantic model it allows medical information to be easily manipulated and tested. This is critical if the encoding program is to be "intelligent" in its operation. It must have some store of medical knowledge, however mundane, to be able to eliminate uncommon or absurd interpretations of its input.

Extension of this work is possible to provide an even more comprehensive program.

ALB67 Alberga, C.N.: String Similarity and Misspellings. Communications of the ACM 10(1967)302-313

BOR68 Borko, H.: Automated Language Processing. (New York: John Wiley and Sons 1968)

BRU71 Brunjes, S.D.: An Anamnestic Matrix Towards a Medical Language. Computers and Biomedical Research 4(1971)571-584

CAP65 College of Ammerican Pathologists: Systematized Nomenclature of Pathology. (Chicago, Illinois 1965)

CHC74 Conneticut Health Care Center Plan, Inc.: Clinical Visit Form Version 5. (New Haven, Conneticut 1974)

CHO57 Chomsky, N.: Syntactic Structures. (The Hague: Mouton, 8th printing 1969)

CPH73 Commission on Professional and Hospital Activities: Hospital Adaptation of ICDA. (Ann Arbor, Michigan 1973)

DAM64 Damerau, F.J.: A Technique for Computer Detection and Correction of Spelling Errors. Communications of the ACM 7(1964)171

DAM76 Damerau, F.J.: Automated Language Processing. Annual Review of Information Science and Technology (Vol. 11). (Washington, D.C.: American Society for Information Science 1976)

DUN77 Dunham, G.S., Pacak, M.G., Pratt, A.W.: Automatic Indexing of Pathology Data. (Bethesda, Maryland: NIH Division of Computer Research and Technology) to appear in the Journal of the American Society for Information Science

GRE63 Green, B.F., Wolf, A.K., Chomsky, C., Laughery, K.: Baseball: An

Automatic Question Answerer . Computers and Thought, Feigenbaum, E.H. and Feldman, J. editors. (New York: McGraw-Hill 1963)

GRE72 Greenwood, R.M.: Kodiac: A System for Disease Coding by a Medium-sized Computer. Bio-Medical Computing 3(1972)128-134

GRI73 Grishman, R., Sager, N., Raze, C., Bookchin, B.: The Linguistic String Parser. AFIPS Conference Proceedings Vol. 42, 1973 National Computer Conference. (Montvale, New Jersey: AFIPS Press)

HIR76 Hirschman, L., Grishman, R., Sager, N.: From Text to Structured Information -- Automatic Processing of Medical Reports. AFIPS Conference Proceedings Vol. 45, 1976 National Computer Conference. (Montvale, New Jersey: AFIPS Press)

HOW68 Howell, R.W., Loy, R.M.: Disease Coding by Computer: The "Fruit Machine" Method. British Journal of Preventive Social Medicine 22(1968)178-181

JOR77 Jordan, S.R., Brown, A.F.R., Hutton, F.C.: Computerized Russian Translation at ORNL. Journal of the American Society for Information Science 28(1977)26-33

KIM73 Kimball, J.P.: The Formal Theory of Grammar. (Englewood Cliffs, New Jersy: Prentice-Hall 1973)

LAM66 Lampson, B.G., Glinski, B.C., Hawthorne, G.S., Soutter, J.C., Russell, W.S.: Storage and Retrieval of Uncoded Tissue Pathology Diagnoses in the Original English Free Text Form. User's Manual I -- A Natural Language Information Retrieval System. (IBM#320-2606: 1966)

KLE63 Klein, S., Simmons, R.F.: A Computational Approach to Gramatical Coding of English Words. Journal of the ACM 10(1963)334

LIN63 Lindsay, R.K.: Inferential Memory as the Basis of Machines Which Understand Natural Language. Computers and Thought, Feigenbaum, E.H. and Feldman, J. editors.

LYN75 Lynch, J.T.: Prevalence of Medical Problems in an HMO Population: A Computerized Medical Record Approach. MPH Thesis, Yale University (1975)

MCC68 McCawley, J.D.: The Role of Semantics in Grammar. Universals in Linguistic Theory, Bach, E. and Harms, R.T. editors. (Holt, Rinehart, and Winston, Inc. 1968)

MIN75 Minsky, M.: A Framework for Representing Knowlege. The Psychology of Computer Vision, Winston, P.H. editor. (New York: McGraw-Hill 1975)

MOR70 Morgan, H.L.: Spelling Correction in Systems Programs. Communications of the ACM 13(1970)90

PRA69 Pratt, A.W., Pacak, M.G.: Identification and Transformation of Terminal Morphemes in Medical English. Methods of Information in Medicine 8(1968)84-90

PRA73 Pratt, A.W.: Medicine, Computers, and Linguistics. Advances in Biomedical Engineering (Vol. 3). (New York: Academic Press 1973)

RAP76 Raphael, B.: The Thinking Computer: Mind Inside Matter. (San Fransico, California: W.H. Freeman and Co. 1976)

RIT74 Ritchie, D.M., Thompson, K.: The UNIX Timesharing System. Communications of the ACM 17(1974)365-375

SAG72 Sager, N.: Syntactic Formatting of Science Information. AFIPS Conference Proceedings Vol. 41, 1972 Fall Joint Computer Conference. (Montvale, New Jersy: AFIPS Press)

SAG75 Sager, N, Grishman, R.: The Restriction Language for Computer Grammers of Natural Language. Communications of the ACM 18(1975)390-400

SAM69 Sammet, J.E.: Programming Languages: History and Fundamentals. (Englewood Cliffs, New Jersey: Prentice-Hall, Inc. 1969)

SCH73 Schank, R.C: Identification of Conceptualizations Underlying Natural Language. Computer Models of Thought and Language Schank, R.C. and Colby, K.M. editors (San Fransico, California: W.H. Freeman and Co. 1976)

STO66 Stone, P.J., Dunphy, D.C., Smith, M.S., Ogilvie, D.M.: The General Inquirer: A Computer Approach to Content Analysis. (Cambridge, Massachusetts: The MIT Press 1966)

THO75 Thompson, F.B., Thompson, B.H.: Practical Natural Language Processing: The REL System as Prototype. Advances in Computers (Vol. 13), Rubinoff, M. and Yovits, M.C. editors. (New York: Academic Press 1975)

WEE69 Weed, L.L.: Medical Records, Medical Education, and Patient Care (Cleveland, Ohio: The Press of Case Western Reserve 1969)

WHI77 White, W., Barkman, B., Bernier-Bonneville, L., Cousinequ, L.: A Method for Automatic Coding of Medical Information in Patient Records. Methods of Information in Medicine 16(1977)1-10

WIN72 Winograd, T.: Understanding Natural Language. (New York:

Academic Press 1972)

WHO77 World Health Organization: Manual of the International Statistical Classification of Diseases, Injuries, and Causes of Death, (Ninth Revision). (Geneva 1977)

WOO70 Wood, W.A.: Transition Network Grammar for Natural Language Analysis. Communications of the ACM 13(1970)591-606

YNG72 Yngve, V.H.: COMIT II. (Cambridge, Massachusetts: The MIT Press 1972)

YALE MEDICAL LIBRARY

Manuscript Theses

Unpublished theses submitted for the Master's and Doctor's degrees and deposited in the Yale Medical Library are to be used only with due regard to the rights of the authors. Bibliographical references may be noted, but passages must not be copied without permission of the authors, and without proper credit being given in subsequent written or published work.

This thesis by has been used by the following persons, whose signatures attest their acceptance of the above restrictions.

NAME AND ADDRESS	DATE
Mitchell Jay Sklar	4/24/85