AN INFORMATION AND ADVISING SYSTEM FOR AN AUTOMATED INTRODUCTORY COMPUTER SCIENCE COURSE

BY

DAVE RONALD ELAND
B.A., Oral Roberts University, 1969
M.S., University of Tulsa, 1972

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1975

Urbana, Illinois

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

THE GRADUATE COLLEGE

June, 1975

WE HEREBY RECOMMEND THAT THE THESIS BY

DAVE RONALD ELAND

ENTITLED AN INFORMATION AND ADVISING SYSTEM FOR AN AUTOMATED INTRODUCTORY COMPUTER SCIENCE COURSE

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Director of Thesis Research

Head of Department

Committee on Final Examination†

Chairman

† Required for doctor's degree but not for master's.

D517

ACKNOWLEDGMENTS

The author wishes to express his sincere appreciation to his thesis advisor, Professor Jurg Nievergelt, for his guidance throughout the course of the research reported in this thesis.

He would also like to thank the members of his thesis committee: Professors H. G. Friedman, W. J. Hansen, S. R. Ray, and B. A. Sherwood for their interest in this project. Thanks is also due to the Department of Computer Science and the National Science Foundation (Grant EC-41511) for their financial support.

In addition, a word of thanks must be given to Ron Danielson, Dave Embley, and Prabhaker Mateti for their many stimulating discussions and helpful suggestions. Special thanks is also in order to Connie Slovak and June Wingler for their excellent work in typing the manuscript.

Finally, the author is deeply indebted to his wife, Patricia, for her kind understanding and support during all phases of this work.

TABLE OF CONTENTS

Chapter Page

LIST OF FIGURES

Figure | Page

1. INTRODUCTION

1.1 Background

The computer revolution is making itself felt in every area of our society, and the educational world is certainly no exception. Much of the administrative tedium associated with education is rapidly being taken over by the computer, and the various disciplines are finding the computer to be a valuable (and increasingly irreplaceable) problem solving tool. In fact, the computer revolution is making inroads into the very process of teaching, in the form of computer managed instruction (CMI) and computer assisted instruction (CAI).

Every revolution brings its own special set of problems, and the educational world is faced with an unprecedented demand for courses which provide students with an opportunity to learn about computers. This demand is particularly noticeable at large universities such as the University of Illinois where about 2000 students per semester are involved in introductory computer science courses. Ironically, the very movement causing this "traffic jam" may be able to help provide a solution to the problem. While the volume in computer use has been going up, the price has been coming down. Consequently, CAI is rapidly becoming financially feasible. This fact has been a fundamental element in the PLATO CAI system developed by the Computer-Based Education Research Laboratory at the University of Illinois [1].

PLATO (an acronym for Programmed Logic for Automatic Teaching Operations) is a timesharing computer system designed to support up to 1000 simultaneous users. Each user interacts with PLATO through a terminal consisting of a keyboard and a plasma display panel [2,3]. Input from the user is made through the keyboard which is essentially a typewriter keyboard augmented by some special function keys. Output to the user is presented on the plasma display panel which produces text and line drawings (on a screen which holds 32 lines of 64 characters) with better resolution than that produced on a typical television screen. The network of 1000 terminals is supported by a CDC Cyber-73 computer system which includes two CPU's (each capable of performing one million instructions per second), 65K words of central memory (60 bit words), two million words of Extended Core Storage (ECS), and over 10^8 words of disk storage [4].

The PLATO system has made it possible for the Department of Computer Science to undertake the ACSES (Automated Computer Science Education System) project [5,6]. The main purpose of this project is to develop a CAI system which can effectively support a significant portion of the task of educating about 2000 students per semester in introductory computer science courses. The essential components of ACSES include the following:

(1) a library of instructional lessons which covers topics on programming concepts, various programming languages, computing techniques, and application areas

(2) a compiler system which supports the interactive preparation, execution, and debugging of programs written in one of several languages [7,8,9]

(3) a communications system to manage comments between students, instructors, and lesson authors

(4) an exam system for the presentation, grading, and analysis of exams [10]

(5) an information retrieval system and guide to the library of lessons (the topic of this thesis).

1.2 Motivation for the GUIDE

One of the dominant characteristics of the lesson material developed for ACSES is that it is quite modular in nature. There are many reasons for this fact: common material can be factored out and treated thoroughly in one place; the needs of a diverse audience (from the novice to the expert, from the casual observer to the serious student) can be met by the same library of lessons; a student or instructor is not constrained to work with a fixed lesson structure, but is free to select the content and lesson sequencing most appropriate for his particular situation; the lesson material has been (and will continue to be) contributed over a long span of time by a large number of different authors who have many different backgrounds and are distributed over a wide geographical area.

The ultimate scope of the ACSES project is quite broad in terms of the amount of material developed for the system and the number of people who would be able to utilize that material. Already the lesson library contains over 150 lessons which are available to anyone who has access to one of the PLATO terminals which are located both locally and at remote geographical locations. Furthermore, plans are being made for

several PLATO systems distributed throughout the country, each capable of supporting approximately 1000 terminals and having some means of sharing lesson material with the other systems. (Currently, PLATO systems are located at the University of Illinois in Urbana, Illinois; Control Data Corporation in Minneapolis, Minnesota; and Florida State University in Tallahassee, Florida.) Thus, there will be potentially many thousands of people of very diverse backgrounds and interests who might want to utilize the library of computer science lessons.

This brings us to the problem of central interest: the size and complexity of the library of lessons make it very difficult for someone to rapidly locate a particular lesson (or set of lessons) which corresponds to his interests at a given moment in time. What is needed is an expert advisor and reference librarian who is available to guide people in the use of these computer science PLATO lessons. This librarian would be intimately acquainted with the lessons and the discipline of computer science, and would be available to guide a student through a course he is taking, or to help someone browse through the lesson modules, pursuing topics of interest. In addition, the librarian would be able to assist an instructor in selecting lesson modules that would be appropriate for a particular course of study.

The PLATO environment in which the ACSES system is to function demands that this librarian be automated. A written description of the lesson library could not be properly distributed and would constantly be outdated as new lessons are added and old ones modified. It would not be feasible to make a human librarian available during all the hours the PLATO system is operational (essentially 24 hours a day) nor at all

the geographical locations where people would have access to a PLATO terminal. Furthermore, a person would be unable to meet the demands of many simultaneous users during peak hours of use.

This thesis is a presentation of the GUIDE -- an information system which was developed to guide people in the use of the computer science library of PLATO lessons. By typing requests in the English language, a student can ask questions about the library of lessons, the lessons and performance required for a particular course, past performance in lessons previously taken, or requests for lessons on the various topics in computer science. The GUIDE will then respond with the desired information, or explain why the request cannot be accepted, or ask for some feedback to help clarify the intent of the request. In addition, the student can initiate a graphics mode of interaction with the GUIDE which enables him to freely explore the information which is maintained by the system. We will demonstrate that the GUIDE successfully fulfills the role of the advisor and reference librarian described above.

1.3 Sample of the GUIDE's Capabilities

The GUIDE is a system which provides some basic tools for structuring and retrieving information. In an abstract sense, the information managed by the GUIDE is a set of objects, a set of relations between those objects, a set of attributes possessed by those objects, and a set of relations between those attributes. In concrete terms, the objects described above are lessons; the relations between the lessons are relationships like prerequisite, sequel, etc.; the attributes of the lessons are concepts in the discipline of computer science; and the

relations between the concepts are relationships like generic, synonym, etc. Conceptually, this collection of information is viewed as an abstract "information space" which a student can explore when searching for lessons which deal with a topic he is interested in studying. Actually, it is useful to view this information space as a sort of "dual space": the lessons and relations between lessons form a "lesson space," the concepts and relations between concepts form a "concept space," and the two spaces are interconnected through keywords (i.e., concepts which have been explicitly attached to a lesson as a description of the content of that lesson).

It should be mentioned that the GUIDE functions as an administrative tool as well as serving to assist students in finding lessons of interest. The information structuring and retrieving capabilities of the GUIDE can be used effectively in managing any relatively large body of information which possesses a rich structure of interconnections.

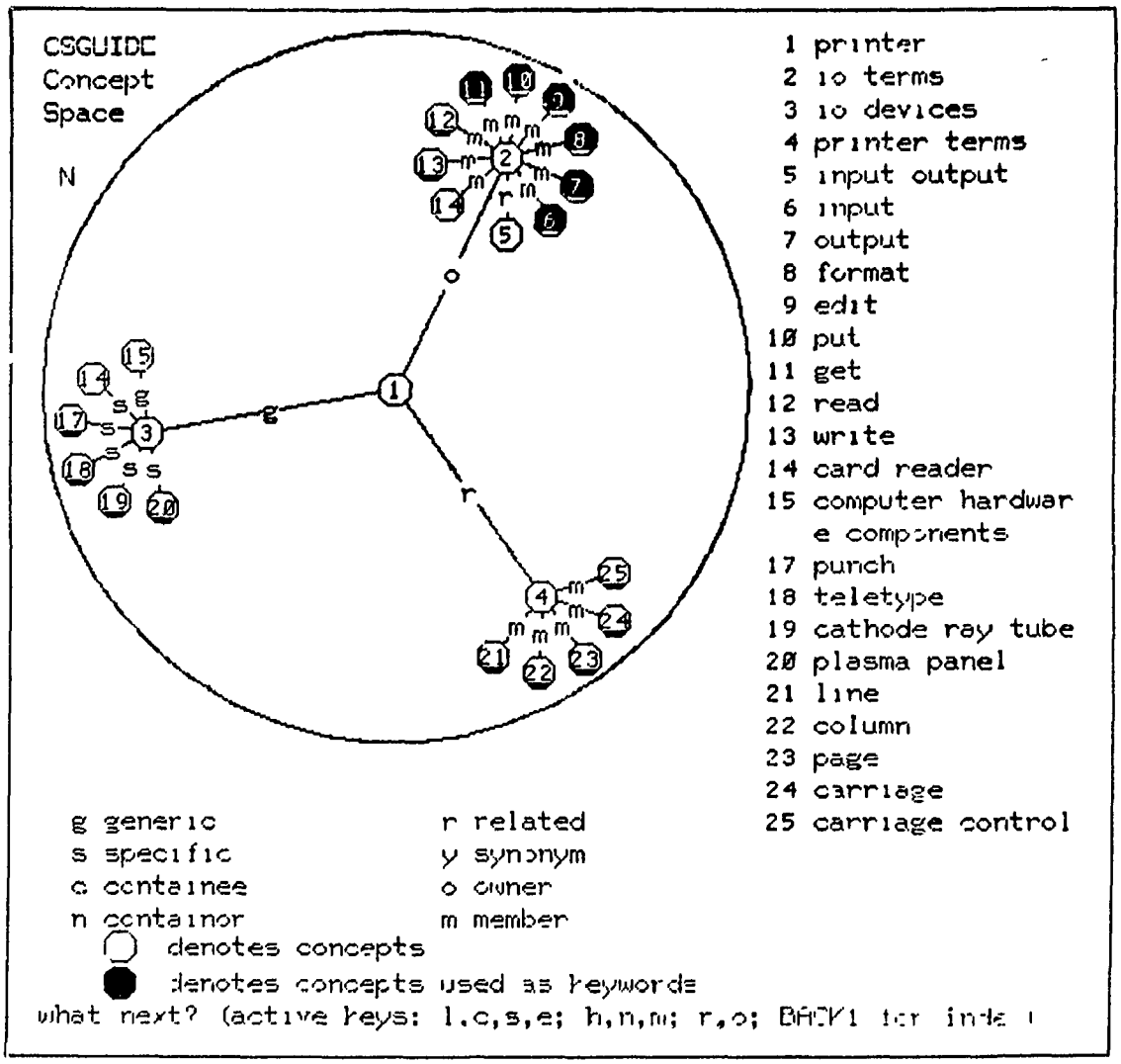
The following sequence of figures presents a hypothetical "dialog" between a student and the GUIDE which highlights the capabilities of the system in guiding a person on a search through the GUIDE's information space.

```
Please type your request:

  give me a lesson about control of spacing on the printer


If I've understood things correctly, I believe you want
something specified by:
    the list of keywords: printer
-- is that right?

(Press NEXT to proceed, BACK to restate request)

Sorry, no lessons matching your specification were found
```

Assume that a search for information begins as illustrated above:

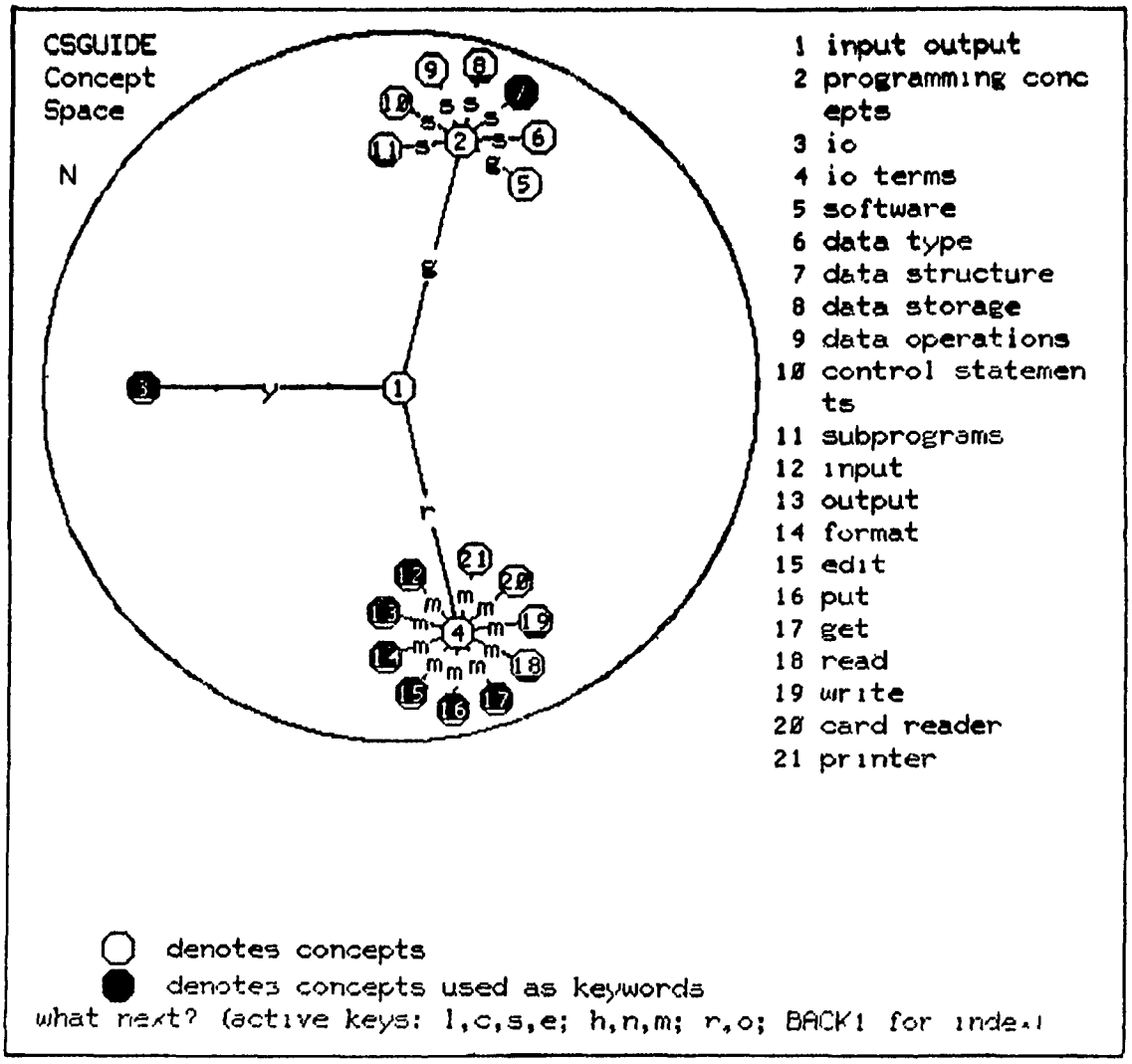
1. The GUIDE asks for the student's request.
2. The student types his request.
3. The GUIDE analyzes the request (underlining marks the progress of the process), paraphrases it to be sure it is properly understood, and asks whether to proceed or wait for a restatement of the request.
4. The student notices that some of the important words of his request were ignored by the GUIDE ("control" and "spacing"). Nevertheless, he decides to see what the GUIDE has on "printer" and presses NEXT.
5. The GUIDE reports that "printer" is not used as a lesson keyword.
6. The student observes that although no lessons were found, the word "printer" is the name of a concept known to the GUIDE. He decides to get a display of terms in the neighborhood of "printer" and look for keywords which will lead him to a useful lesson.

Figure 1. Sample Request for Information



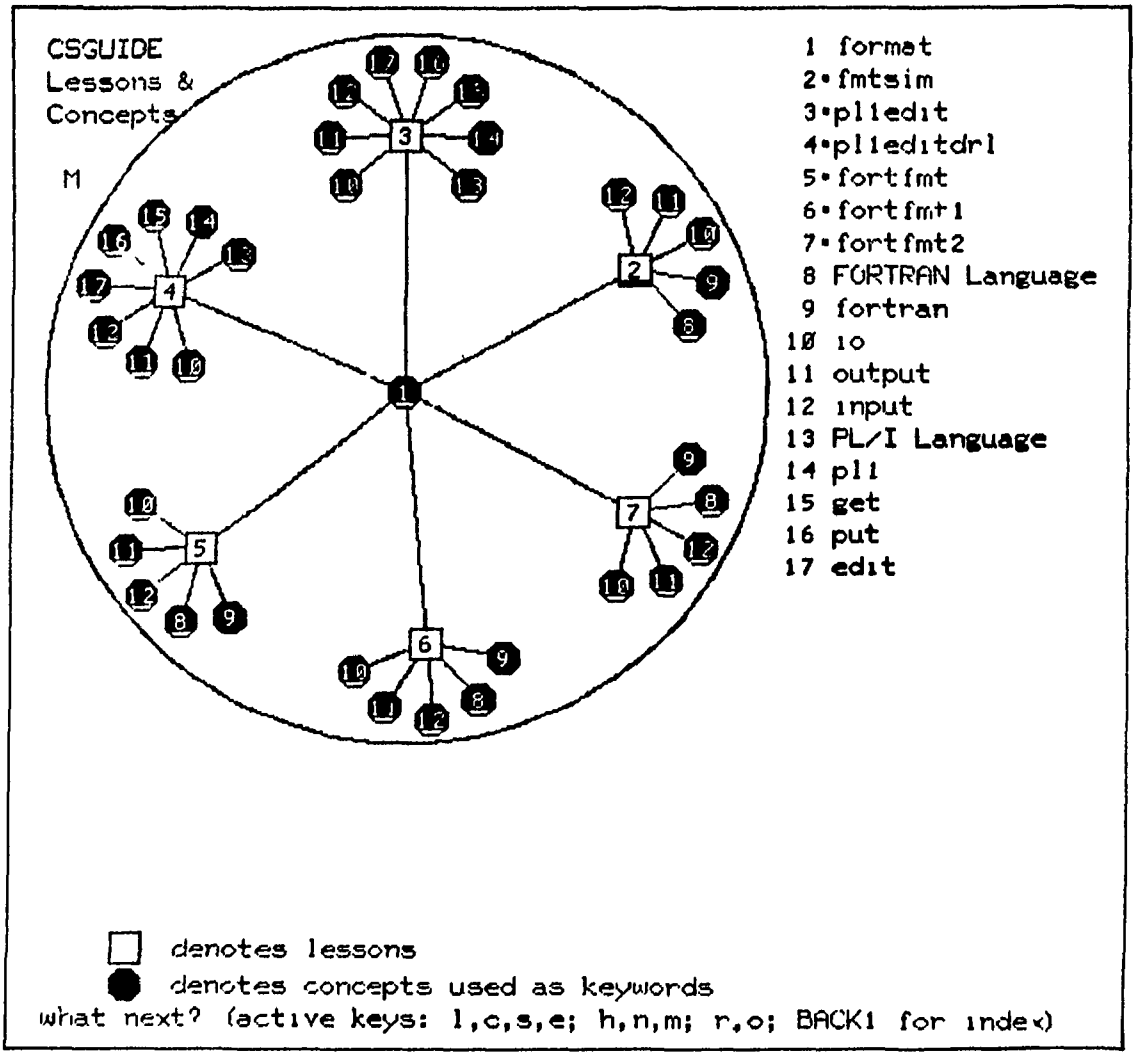
Having obtained a display of concepts in the neighborhood of "printer," the student now observes that the concept "input output" is in the general area of his interest. He decides to move in that direction through the concept space by requesting a neighborhood display with "input output" at the center.

Figure 2. Concepts in the Neighborhood of "printer"



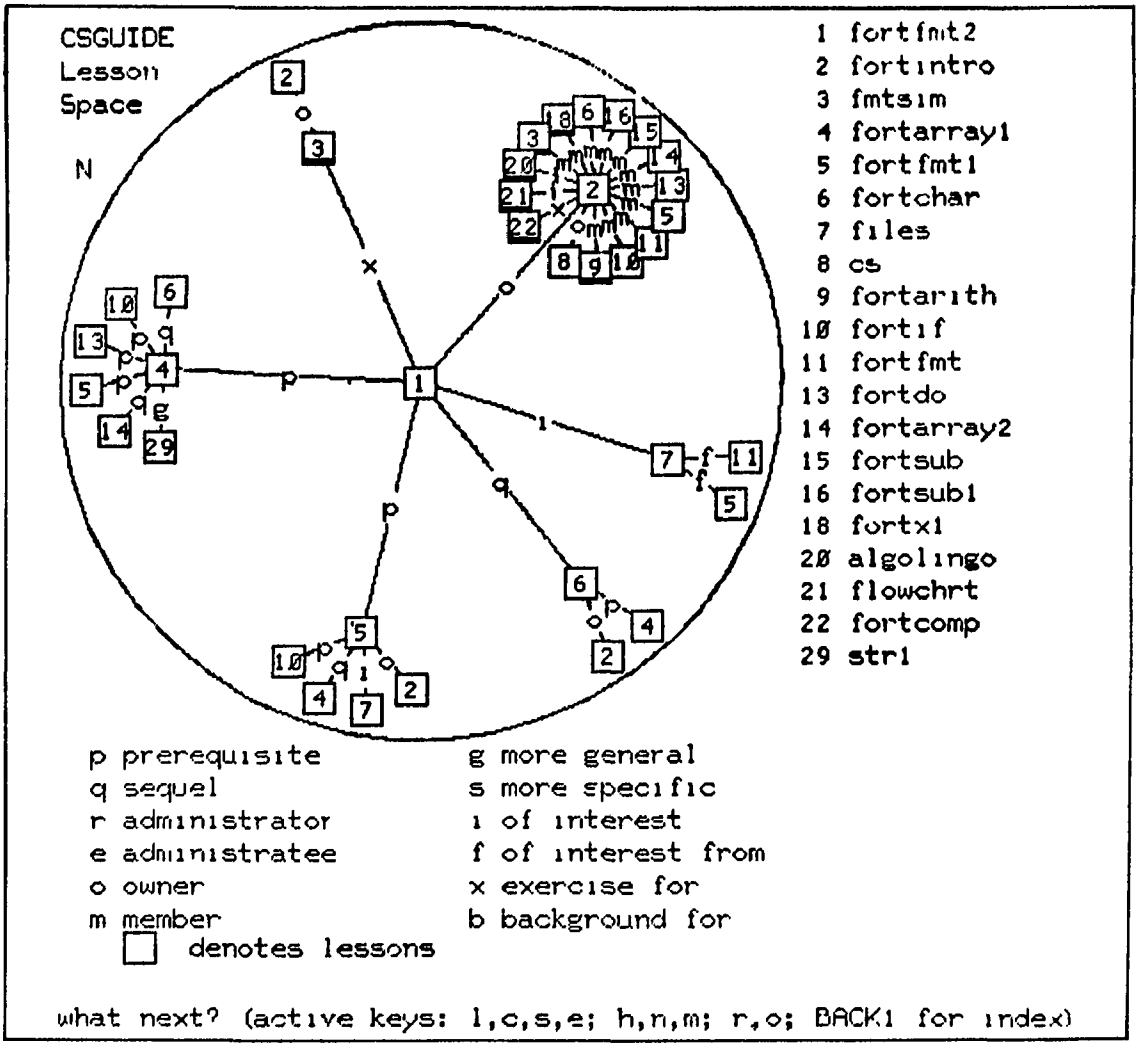
From this display of concepts in the neighborhood of "input output," the student observes that he is not getting any closer to keywords which seem to be in his area of interest. He decides that the keywords in the neighborhood of "io terms" are his most promising lead, and he selects "format." In order to obtain an overall perspective of the lessons which have "format" as a keyword, he requests a "mixed mode" display which shows both lessons and concepts.

Figure 3. Concepts in the Neighborhood of "input output"



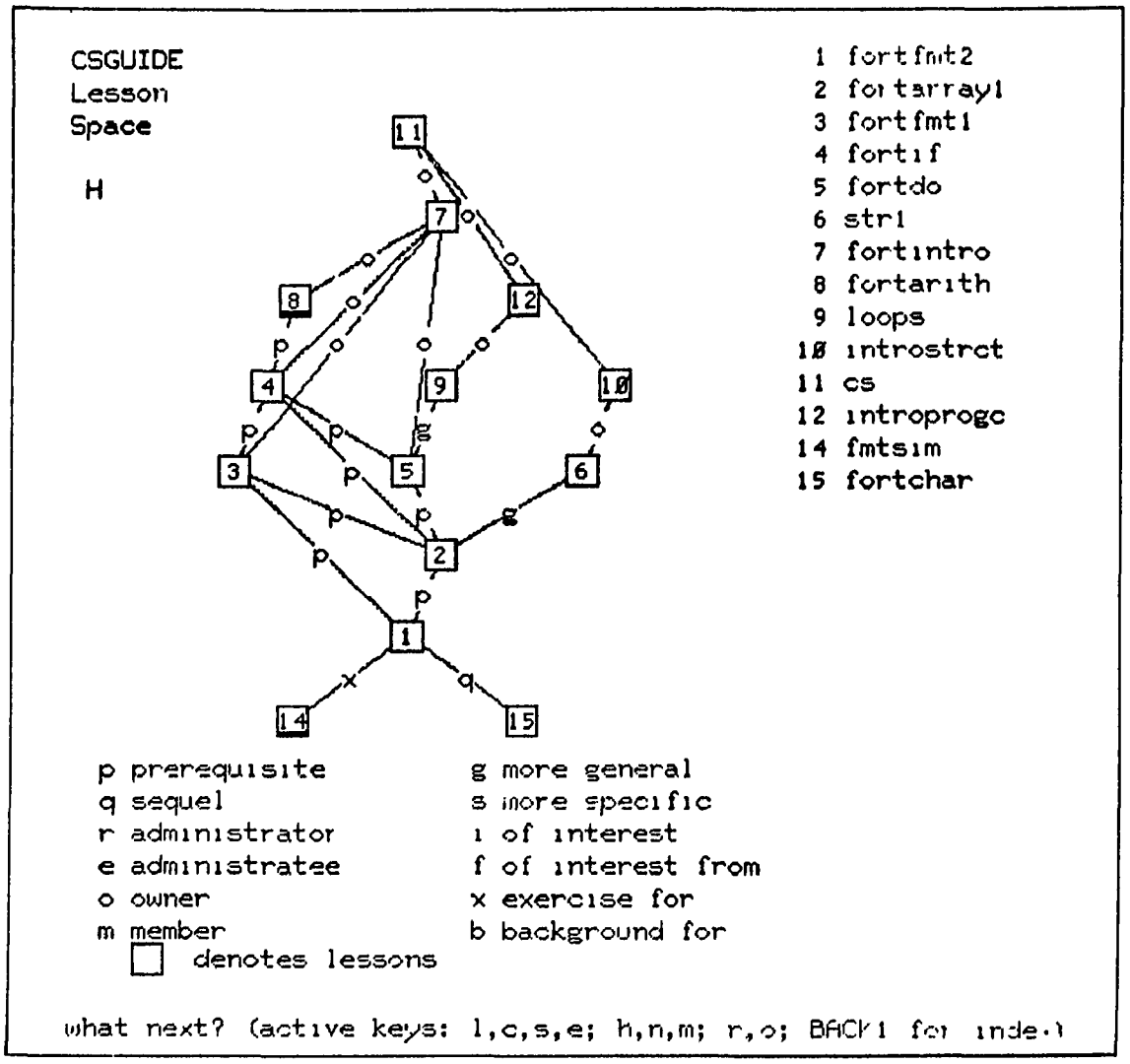
This display shows all the lessons in the neighborhood of "format," and in turn all the concepts in the neighborhood of those lessons. The student deduces that the lesson "fortfmt2" is a promising candidate for study. He decides to take a look at the lessons which are in the neighborhood of "fortfmt2" in the lesson space.

Figure 4. Lessons and Concepts in the Neighborhood of "format"



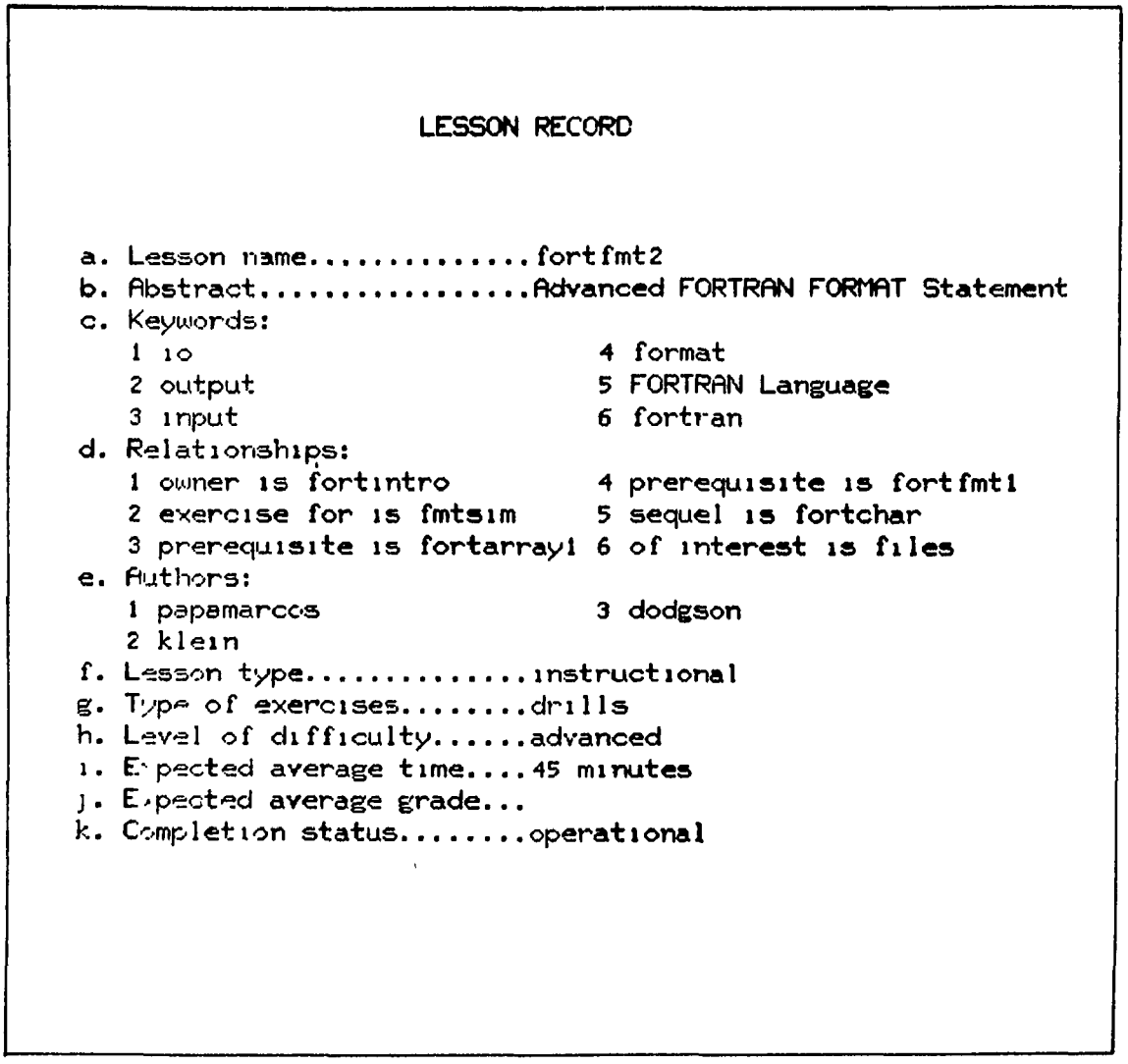
This display shows that there is a rich prerequisite-sequel structure around "fortfmt2," and the student requests a hierarchical display of the lesson space for a better perspective.

Figure 5. Lessons in the Neighborhood of "fortfmt2"



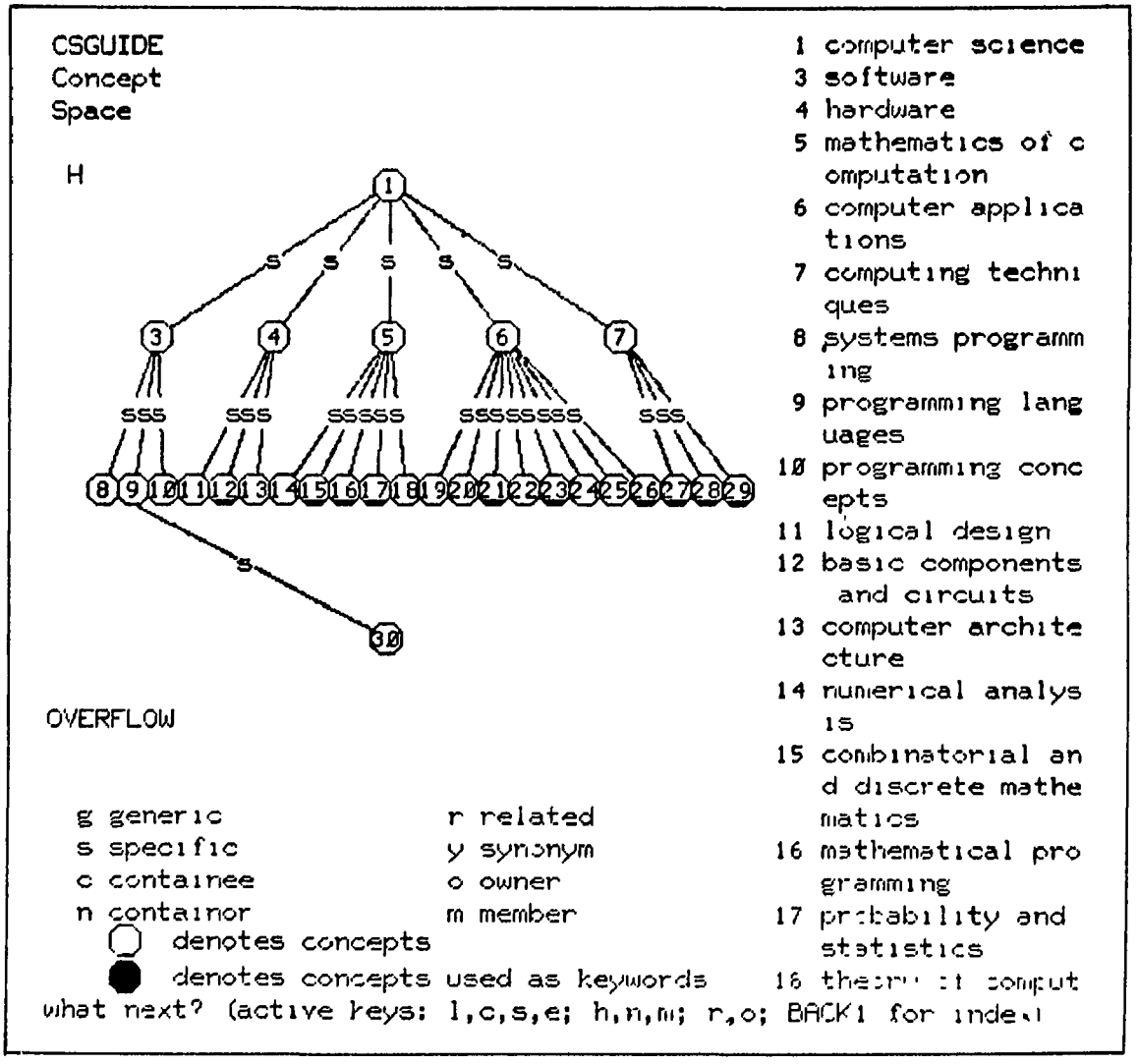
The student feels that he is acquainted with the prerequisites, and is now fully satisfied that "fortfmt2" is the lesson he wants to study. He decides to take a look at the lesson record just prior to leaving the GUIDE and entering the lesson "fortfmt2."

Figure 6. Hierarchical Display of Lesson Space around "fortfmt2"



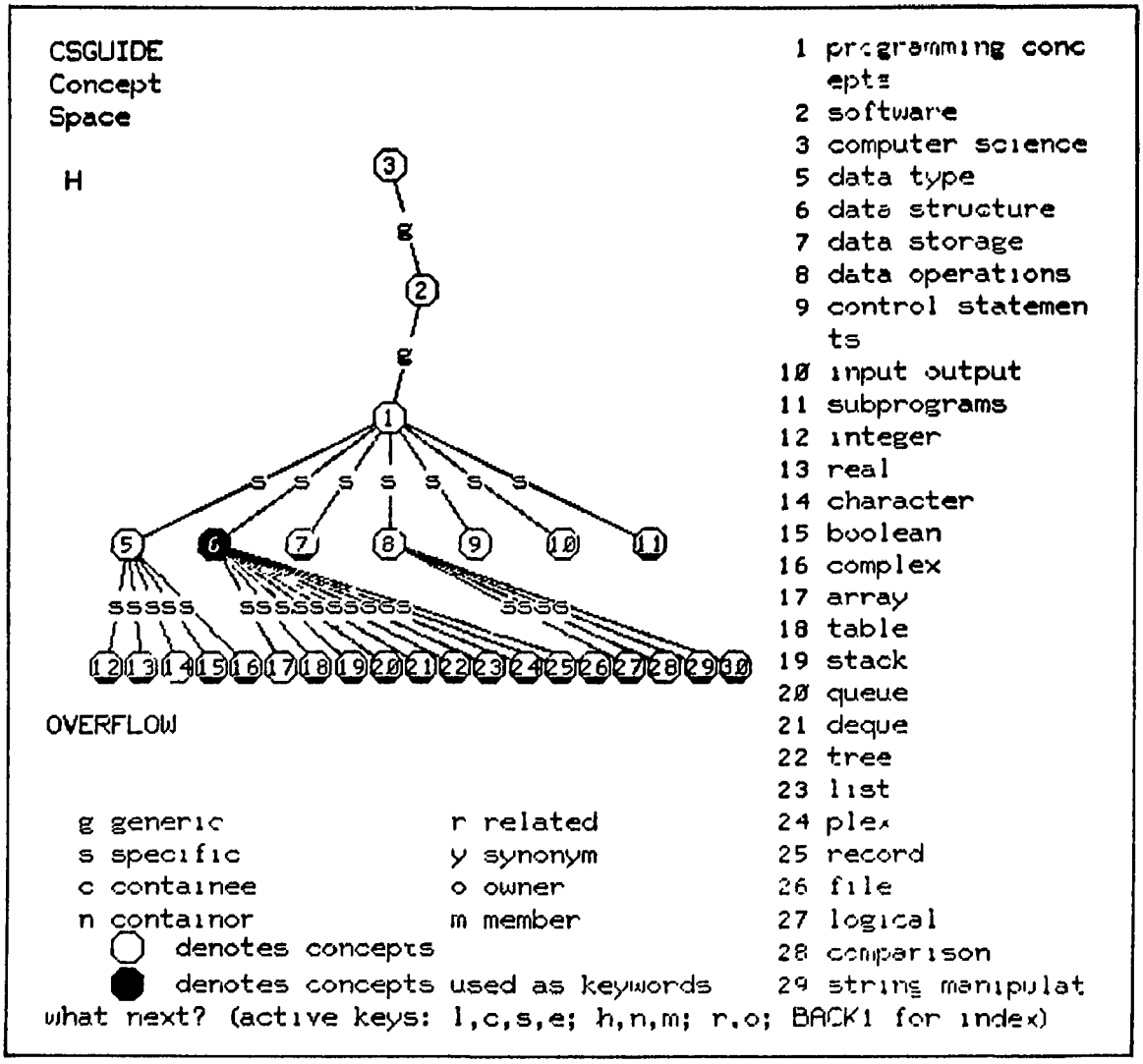
After looking at the lesson record, the student exits from the GUIDE and studies "fortfmt2."

Figure 7. Lesson Record for "fortfmt2"



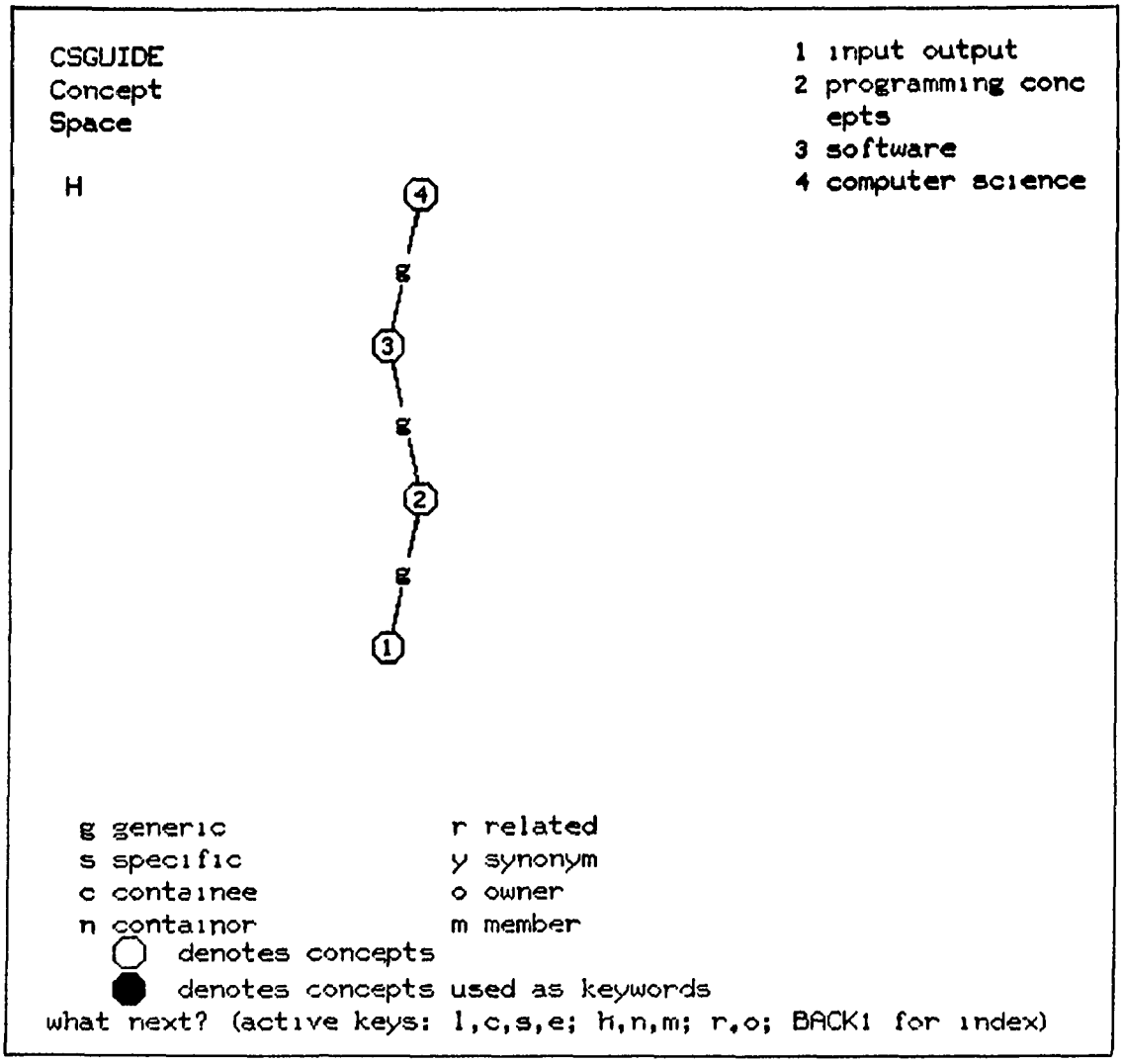
As an alternative to the "bottom up" approach illustrated in Figures 2 and 3, the student could have taken a "top down" approach by starting at the root of the classification tree in the concept space -- "computer science" -- and gradually refining his area of interest. From the above display, he selects "programming concepts."

Figure 8. Hierarchical Display of Concept Space around "computer science"



From this display, the student selects "input output."

Figure 9. Hierarchical Display of Concept Space around "programming concepts"



Having reached the "end of the line" in the classification tree, the student switches to a display of all the terms in the neighborhood of "input output."

Figure 10. Hierarchical Display of Concept Space around "input output"

1.4 Research Problems Posed by the GUIDE

The problem of rapid translation of a request from English to an intermediate representation for further processing has been solved by Pradels in his doctoral dissertation [11]. Some of the interesting research problems which remained to be solved in implementing the GUIDE include the following:

(1) In order to effectively guide students who want to explore topics of interest in the area of computer science, the GUIDE had to be given an understanding of the discipline of computer science. The database which was developed to provide this understanding is called the "concept space." The design of the concept space was based on research into methods for organizing knowledge, and techniques for the machine representation of knowledge. In building the content of the concept space, the field of computer science was thoroughly reviewed, topics and concepts pertinent to an introductory view of computer science were carefully selected, and the interrelationships among these concepts were defined. Finally, techniques which made use of the graphics capabilities of the PLATO terminal were developed for communicating the content of the concept space to the student.

(2) The PLATO lessons in the ACSES library had to be reviewed and cataloged for the GUIDE. The lesson catalog which was developed is in some ways like an automated card catalog. The fact that it is automated makes it easier for a student to retrieve a given entry in the catalog, and it is possible to pursue longer "chains of relationships" in less time than that required by manual methods. Furthermore, the nature of the entries in the GUIDE's "automated catalog" -- CAI lessons instead of books -- leads to a richer structure of interrelationships than is found

in a typical card catalog. A conceptual view of this aspect of the catalog was developed in which the lessons and the relations between them are viewed as an abstract "lesson space." It turned out that the lesson space was structurally equivalent to the concept space, and this fact was fully exploited in applying to the lesson space the graphics techniques which had been developed for the concept space.

(3) The possibilities of integrating an "advising function" together with more traditional information retrieval functions was an innovative aspect of the GUIDE. The databases which would support this function (course outlines and student records) had to be designed and incorporated into the GUIDE, and the mechanism to recognize and respond to requests about past performance or specified goals was also provided. In addition, routines were written which compare a student's performance against the standard of performance specified in a course outline. (These routines are invoked at the initiative of the student.)

(4) Traditional information retrieval and data management problems had to be solved within the severe constraints of the PLATO CAI environment. (A data management system was designed which supports the entry, deletion, modification, and security of the data for the GUIDE system.) More specifically, the data for the system had to be represented in a manner which is conducive to processing within the PLATO memory hierarchy (disk, ECS, central memory), and occupies less than 8K 60 bit words. The processing algorithms are limited to 10K words of storage and are allowed to utilize an average of less than 2 to 5 milliseconds per second of processing time (this is on the order of 10^5 machine instructions per request presented to the GUIDE). By way of contrast, typical information retrieval and data management systems

described in the literature are often implemented on a computer dedicated to the sole purpose of supporting that system. Relative to the resources available to the GUIDE, these systems are able to utilize roughly 10 - 10^2 times as much central memory, and 10^2 - 10^3 times as much processing power.

2. DESIGN OF THE GUIDE

2.1 Introduction

The design of the GUIDE provided a challenging task in software engineering. The stringent time and space constraints of PLATO intensified the struggle of time-space trade offs which is inherent in a system as large and complex as the GUIDE. In addition, the PLATO system itself has evolved considerably since the inception of the GUIDE project. New features and capabilities of PLATO were consistently monitored and incorporated into the GUIDE whenever an improvement could be realized. Also, the TUTOR language in which the GUIDE was implemented provides few structural facilities in the areas of data management and flow of control. Thus, it was necessary to develop a systematic methodology for data management as well as techniques for imposing a discipline of structured programming. The effort expended in developing and consistently using techniques for structured programming has already been vindicated during the implementation and debugging of the GUIDE, and should be of immense value for the maintenance of the system.

The driving force behind the design process has been to build a system which will not become a "museum piece," but will be of practical use to people interested in the computer science PLATO lessons. This pragmatic approach is characteristic of work done in the PLATO environment where a recurring theme is "try it out and see how it works." The design of the GUIDE has been through several iterations. Previous work by

Murai [12] and Pradels [11] was carefully reviewed. Tentative designs were implemented and then modified or rewritten until a workable configuration was achieved.

The essential components of the GUIDE which have evolved from this design process are shown in the block diagram in Figure 12. The blocks enclosed by the dotted line represent the various areas of the database, those outside the dotted line represent the different processing algorithms of the system, and the solid lines and arrows represent the flow of information. In addition to generating the final response, the processing algorithms will also provide appropriate feedback messages when additional information is needed to process a request. This chapter consists of a presentation of each of the components outlined in the diagram.

2.2 The GUIDE Database

2.2.1 Introduction

The topic of databases and data management has been receiving a great deal of attention. This is evident in the large number of recent publications on this topic (e.g. various ACM publications, Proceedings of AFIPS, Proceedings of IFIPS, etc.), as well as the activity of such groups as CODASYL (Committee on Data Systems and Languages) [13-15], the Joint GUIDE-SHARE Data Base Requirements Group [16], and the ACM SIGMOD (Special Interest Group on Management of Data) [17,18].

Several of the important issues and concepts emerging from this activity have been applicable in the development of the GUIDE. For example, the concept of data integration was a significant factor in making the GUIDE a workable system. In the early days of data processing,

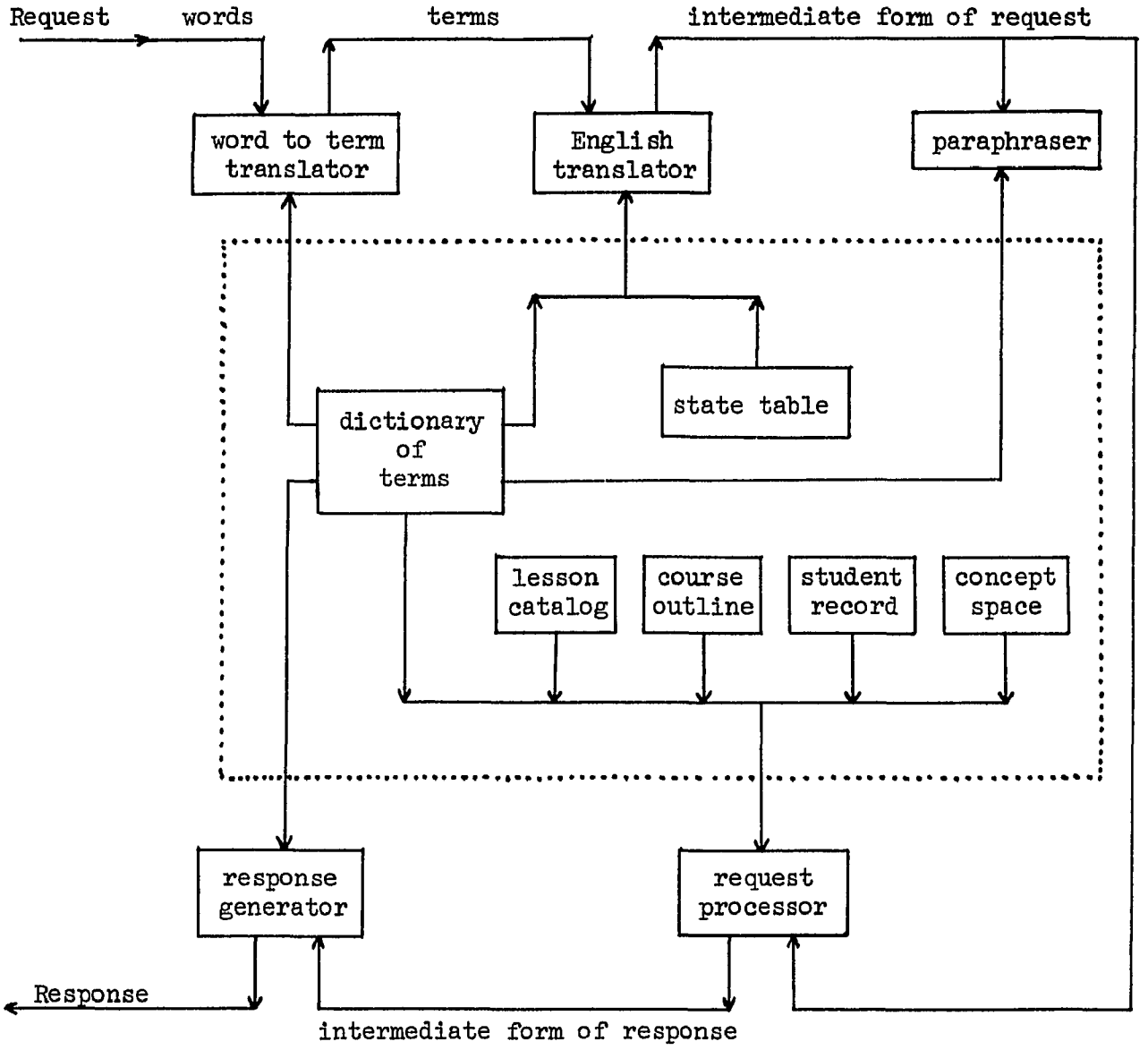


Figure 12. Block Diagram of the GUIDE

the typical mode of operation was for each programming application to design and maintain its own distinct set of data. As systems become larger and more complex, the task of managing all the different sets of data became increasingly difficult. Eventually, there emerged the idea of integrating all the data maintained by a system so that essentially one copy of each data item is stored in the system, and all applications obtain their data from this single "database." This same phenomena occured in the evolution of the GUIDE. In the early design stages, several distinct files of information were envisioned for the system, such as the lesson catalog, course outlines, a classification schedule, and the translator's vocabulary. However, the success of the final design was dependent to a large extent on the integration of all this data into a single database. The performance of the GUIDE would suffer appreciably if the system were to lose the savings in memory space and processing time which have been achieved by utilizing an integrated database.

Another valuable concept utilized in the GUIDE system is the idea of a subschema [15]. The schema is a description of the entire database maintained by a system, whereas a subschema is a description of a portion of that database utilized by a particular application. This idea was applied in implementing two PLATO lessons which are distinct from the GUIDE, but utilize information from the GUIDE database. One of these lessons provides an index to the computer science PLATO lessons; the other provides access to the course outlines. In each case, only a portion of the entire GUIDE database is used. In actual practice, the appropriate portion of the database is extracted, reformatted, and stored in a separate file. Conceptually, however, this process can be

viewed as a software mapping which maps a subset of the schema onto a subschema. Thus, the maintenance of the data is still handled in one place -- the database -- and the applications depend on this one database for their data.

An important issue in any computer application is that of data integrity. The problems in this area are compounded in a database system, and complicated even further in a timesharing environment. In the GUIDE, checks are made on new data coming into the database to be sure that all items have valid values. Special "lock" and "unlock" primitives were implemented to guarantee mutual exclusion; that is, attempts to simultaneously modify the same data are blocked and access permitted to only one user at a time. (Whereas some systems solve the mutual exclusion problem by allowing only one user at a time to edit the entire database, it is possible for several people to simultaneously edit the GUIDE database. This was achieved by careful analysis of the editing operations and identification of certain critical operations for which mutual exclusion must be provided.) In spite of all precautions, errors in the database will inevitably occur. Analysis routines were written which check the internal consistency of the database and assist in making corrections if any damage is detected.

One more item to be mentioned is the problem of security. Many hundreds of man-hours have been invested in the construction of the GUIDE database, and special precautions have been taken to preserve that information. For protection against accidental damage to the database (PLATO system crashes, bugs in new database management routines, careless editing, etc.) a disk-file backup copy of the database is maintained at all times, and printed hardcopy of the database is also produced

periodically. For protection against deliberate tampering with the database, a scheme of passwords and access keys was implemented. Access to edit various areas of the database is granted on the basis of a password entered by the user. Any attempt to systematically break a password is recorded on a special audit trail which also records any changes actually made to the database. In addition to the above security measures, the PLATO system provides backup copies of files in case of a system crash, and a password scheme provides protection against illegal access to a PLATO file.

The construction and maintenance of the GUIDE database is accomplished through a special editor. The database editor handles all entry and deletion of data, makes all appropriate interconnections between data items, sets all internal information required for proper storage and retrieval, and manages the allocation of memory space. In addition, several utility routines provide for the display, analysis, and repair of the various parts of the database.

2.2.2 Dictionary of Terms

The primary mechanism by which the database has been integrated is through the introduction of the concept of "terms." Any word or phrase known to the system (e.g. the lesson name "fortif" or the grammatical phrase "do you have" or the concept "do loops") is considered to be a term. All information known about a term is stored physically adjacent in memory. (Thus, knowing the name of a term makes all information about that term immediately available.) This packet of information -- a term and everything known about it -- forms a logical "record." The different types of terms (lesson names, concepts, course names, author names, and

grammatical words) each have a different record format since the detailed information relevant to each is different. All records are of variable length so that optimum space utilization can be achieved.

Terms are located in the dictionary by the method of hashing in order to provide the fastest possible retrieval mechanism. New terms are added at the end of currently used storage. When a term is deleted, the memory space is reclaimed immediately by sliding up all subsequent term records. (This operation is relatively inexpensive because of special data movement capabilities which are available on PLATO.) At all times, the database is in a configuration which occupies the minimum amount of memory space -- there are no unused areas lying around waiting for special garbage collection routines to make them available.

Term records are arranged in memory in chronological order (although any other convenient ordering -- alphabetical, or chronological (or alphabetical) by term type, etc. -- could also be implemented simply by changing the algorithm which selects the location where a new term record is to be placed). Associated with each term is a unique number which could be viewed conceptually as an accession number, but as a practical matter is the location of a slot in an address table (the record directory). This number remains invariant as long as the term is in the database and serves as a logical pointer to the term record. All interconnections between terms are made via this term number (a lesson record references concepts, lessons, and authors; a concept record references lessons and concepts; author and course records reference lessons). This technique permits references to other records to be packed very compactly within a term record. Furthermore, records can be

freely moved for memory management with no need to make any changes at the record level -- only the record directory needs updating when this is done.

The result of using this "term concept" is that the entire GUIDE database (except for the state table and student record) is simply a sequence of term records. This makes it possible to use a small, carefully designed set of routines to manage and utilize the database (all records share the same retrieval mechanism, the same memory management, etc.). Furthermore, this helps separate the physical and logical views of the database. There is no physical entity which forms the lesson catalog, course outlines, or concept space. Rather, these are logical entities defined by the set of lesson records, course records, and concept records respectively.

The details of the term record formats for lessons, courses, and concepts will be discussed in sections 2.2.4, 2.2.5, and 2.2.7. A sample author record is shown in Figure 13. The significance of each field of information is self-explanatory. The author record was implemented to facilitate the retrieval of lessons specified by author name.

A sample grammatical word record is shown in Figure 14. The "name index" field is used primarily to distinguish words which are a member of a class of terms which form an ordered set. For example, the name index for "January" is 1, "February" is 2, etc. In the example of Figure 14, "sequel" happens to be the second relation in the set of relations which are defined between lessons. (The full set of relations is discussed in section 2.2.4.) Most words simply have a value of 0 for the name index. The "input categories" field is used by the English

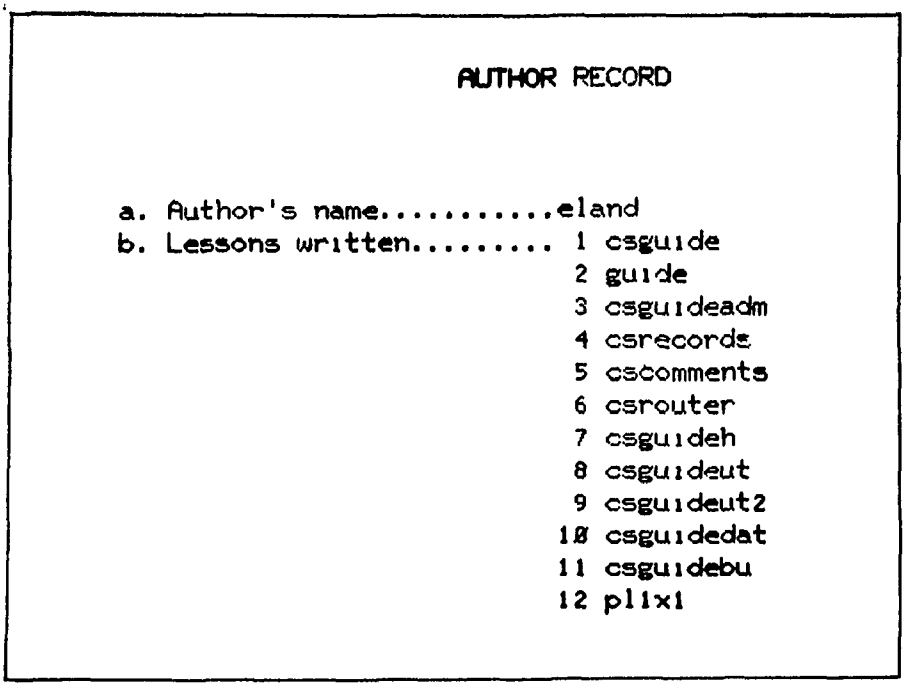


Figure 13. Sample Author Record

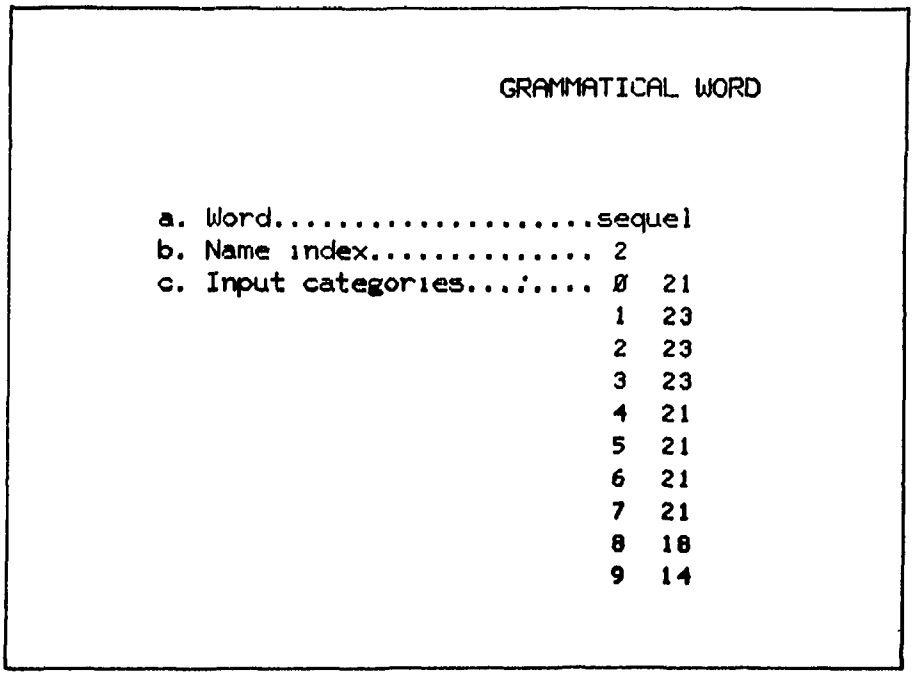


Figure 14. Sample Grammatical Word Record

translator. Whereas lesson names, concepts, course names, and author names have an input class of 1, 2, 3, or 4, respectively, regardless of the current state of the English translator, the input category of grammatical words is a function of the current state of the translator (there are 10 possible states, numbered 0 through 9). This mechanism was introduced to handle the situation of words which have a different meaning in a different context. (The translator is discussed in section 2.3.3.) The input categories of all grammatical words are documented in the Appendix.

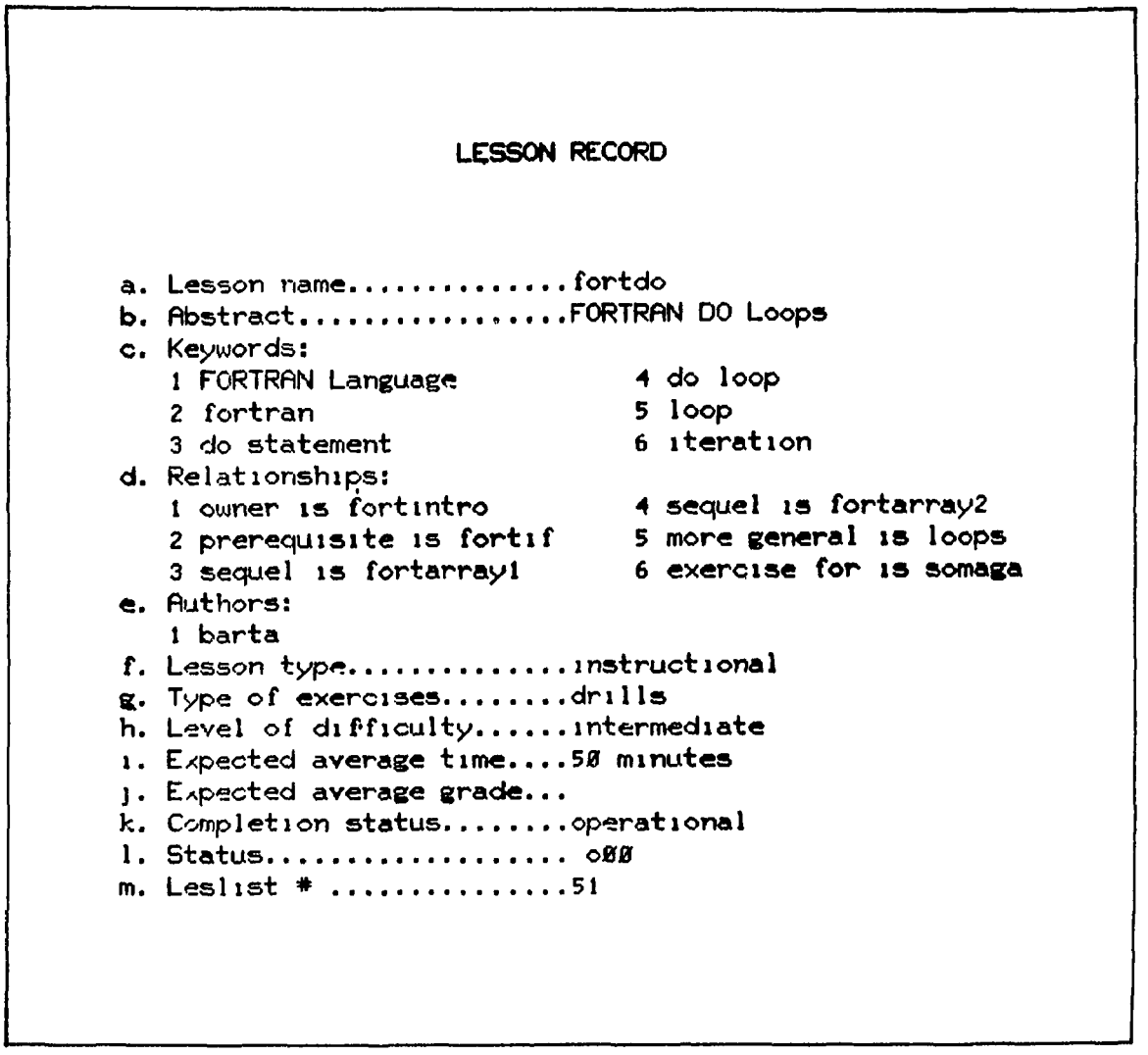
2.2.3 State Table

The state table (shown in the Appendix) is utilized by the English translator as it transforms the original request into an intermediate representation for further processing. Corresponding to each possible term input class and each state of the finite state automaton which drives the translator is a table entry which determines the next state to be entered by the automaton and the output symbol to be generated (the output symbol becomes a part of the intermediate representation of the original input request). There is also a special level of confidence flag used to mark certain situations for special processing. A discussion of the translator is presented in section 2.3.3.

2.2.4 Lesson Catalog

The lesson catalog is simply the set of all lesson records. A sample lesson record is shown in Figure 15. The meaning of each field of information is discussed below:

(a) "Lesson name" is the name of the lesson as known to the PLATO system. At the present time, the name of a PLATO file may contain



LESSON RECORD

a. Lesson name.....fortdo
b. Abstract.....FORTRAN DO Loops
c. Keywords:
   1 FORTRAN Language
   2 fortran
   3 do statement
   4 do loop
   5 loop
   6 iteration
d. Relationships:
   1 owner is fortintro
   2 prerequisite is fortif
   3 sequel is fortarray1
   4 sequel is fortarray2
   5 more general is loops
   6 exercise for is somaga
e. Authors:
   1 barta
f. Lesson type.....instructional
g. Type of exercises.....drills
h. Level of difficulty.....intermediate
i. Expected average time....50 minutes
j. Expected average grade...
k. Completion status.....operational
l. Status..... o00
m. Leslist * 51

Figure 15. Sample Lesson Record

at most 10 characters, and consequently a PLATO lesson name is not always fully descriptive of the lesson's content.

(b) "Abstract" gives a one line abstract of the lesson. Perhaps "title" would be a more appropriate name for this field of information, since the one line limit results in more of a descriptive title than a complete abstract. Conceptually, there is no reason why the abstract field could not be extended to a full page of text or even more; but as a practical matter, the one line limit is necessary due to limitations on memory space.

(c) "Keywords" is a list of words and phrases which are descriptive of the lesson's content. The words included in this field are those which enable one to retrieve a lesson specified by its subject matter.

(d) "Relationships" is a list of relation-lesson pairs which define the relationships that exist between this lesson and other lessons in the lesson catalog. The interpretation of this structure as an abstract "lesson space" is described in section 2.3.6. The full set of possible relations is discussed below:

prerequisite-sequel: If lesson A contains material that should be mastered before lesson B can be attempted, then lesson A is a prerequisite of lesson B. For example, "fortif" is a prerequisite of "fortdo." Conversely, "fortdo" is a sequel of "fortif."

administrator-administratee: If lesson A provides administrative services for lesson B, then lesson A is an administrator of

lesson B. Conversely, lesson B is an administratee of lesson A.

owner-member: This relationship was introduced for the purpose of imposing an administrative structure on the library of lessons. All of the lessons on a particular topic are grouped into sets containing ten to fifteen lessons. A specific individual is then appointed the manager of a given set of lessons. This individual fills a role much like that of the editor of a scholarly journal, maintaining quality control and initiating new material as necessary. A special "entry" lesson (see field f below) is created which is the "owner" of each lesson that is a "member" of this set of lessons.

more general-more specific: This relates lessons which deal with a given topic at a level which is "more general" or "more specific" than the treatment in this lesson.

of interest-of interest from: This is a loose relationship used to point out lessons which may be "of interest" to someone who is interested in this lesson.

exercise for-background for: This is a relation which points the way to lessons which provide "exercise for" the theory of this lesson, or which provide "background for" the exercises in this lesson.

(e) "Authors" is simply a list of the authors of the lesson.

(f) "Lesson type" indicates the nature of the lesson being described. The different types of lessons are as follows:

instructional: presents subject matter material

exam: provides test questions and grades responses

communication: a file for comments, bulletins, or perhaps on-line interactive communication

administration: a lesson which provides administrative services (often a file for handling data management)

compiler: a lesson which allows one to enter and execute programs

entry: a special administrative lesson which provides a forum for the manager of a set of lessons to set forth the general philosophy and special conventions of that set of lessons

(g) "Type of exercises" gives an indication of the interaction provided by the lesson. Possible values are none, drills, simulation, and programming.

(h) "Level of difficulty" can be introductory, intermediate, or advanced.

(i) "Expected average time" gives an estimate of the time (in minutes) required to complete the lesson.

(j) "Expected average grade" is the grade expected for the lesson (based on a scale of 100). Most lessons in the lesson catalog do not currently evaluate student performance, and thus this field is usually empty.

(k) "Completion status" indicates the current level of development of the lesson. Possible values are just started, work in progress,

nearly complete, operational, under revision, obsolete, and revision needed.

(ℓ) "Status" provides a set of flags used by the GUIDE to mark certain lessons for special handling (for example, access to certain lessons may be restricted to some users).

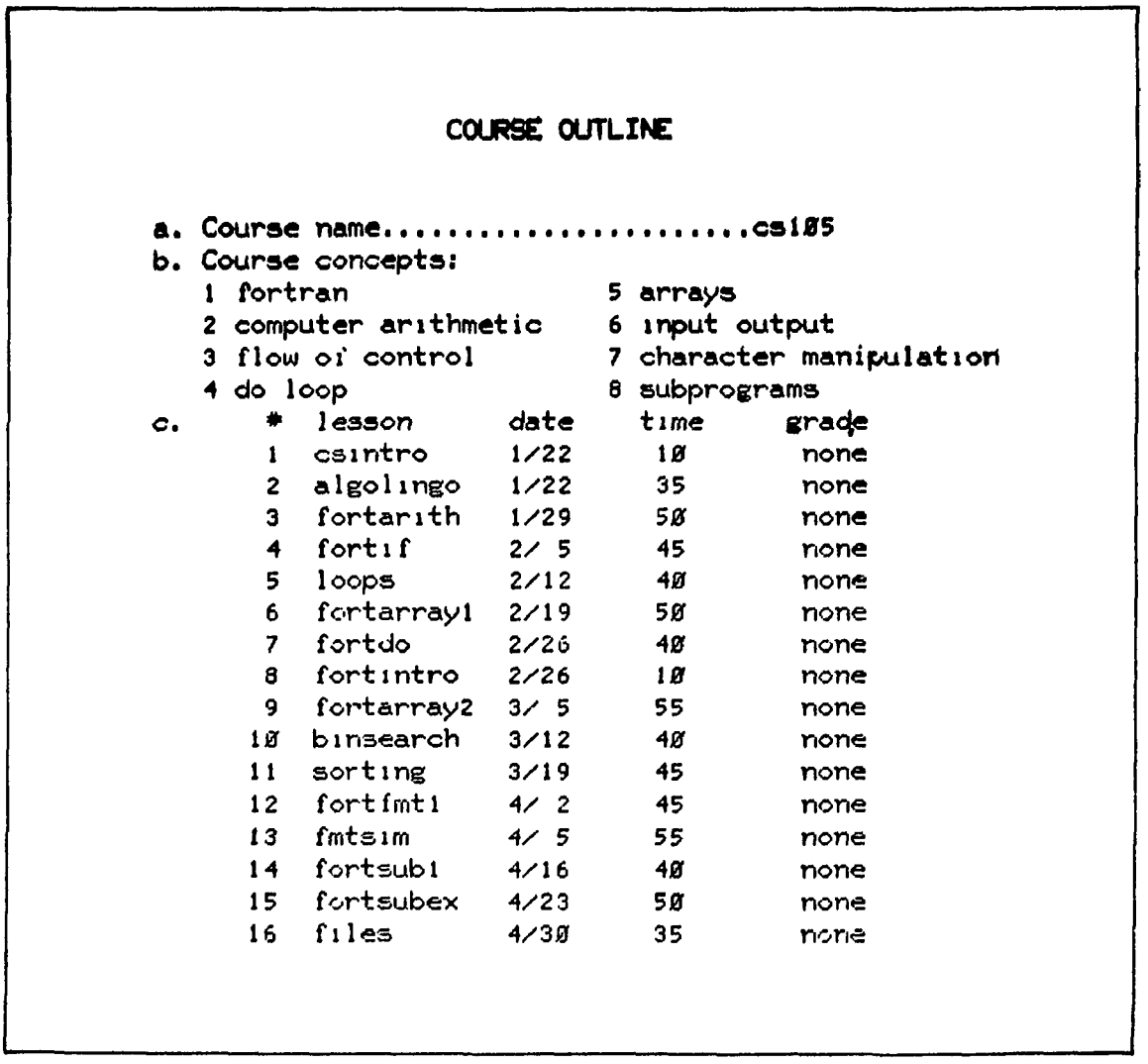
(m) "Leslist #" is the location of this lesson in an address table maintained by the PLATO system. This address table is used by PLATO so that references to a file can be made independent of the actual file name.

2.2.5 Course Outlines

A sample course outline is shown in Figure 16. The meaning of each field of information is as follows:

(a) "Course name" is the GUIDE name of the course. This name may or may not correspond to the PLATO course in which a student is enrolled. (In order to sign onto the PLATO system, one must be enrolled in a PLATO course.)

(b) "Course concepts" is a list of concepts in the concept space. (The concept space is discussed in Section 2.2.7.) This allows an instructor to describe a set of concepts which are relevant to a particular course. This model of the course expressed in the concept space could supplement the list of lessons described in field (c) below, or it might be the only description of the course known to the GUIDE. In either case, the student can locate lessons appropriate for study by exploring the concept space (as illustrated in Section 1.3).



COURSE OUTLINE

a. Course name.....cs105

b. Course concepts:

1 fortran	5 arrays
2 computer arithmetic	6 input output
3 flow of control	7 character manipulation
4 do loop	8 subprograms

c.

#	lesson	date	time	grade
1	csintro	1/22	10	none
2	algolingo	1/22	35	none
3	fortarith	1/29	50	none
4	fortif	2/ 5	45	none
5	loops	2/12	40	none
6	fortarray1	2/19	50	none
7	fortdo	2/26	40	none
8	fortintro	2/26	10	none
9	fortarray2	3/ 5	55	none
10	binsearch	3/12	40	none
11	sorting	3/19	45	none
12	fortfmt1	4/ 2	45	none
13	fmtsim	4/ 5	55	none
14	fortsub1	4/16	40	none
15	fortsubex	4/23	50	none
16	files	4/30	35	none

Figure 16. Sample Course Record

(c) This field of information gives a list of lessons and the date by which each lesson should be completed. Essentially, this imposes a prerequisite-sequel relation on a set of lessons. This ordering is not constrained to match the ordering specified in the lesson catalog, so an instructor is free to select whatever is most appropriate for his situation.

2.2.6 Student Records

The box labeled "student record" in the block diagram of the GUIDE in Figure 12 is deceptively simple in its appearance. In reality, there is a great deal of machinery behind the scenes which makes that student record available to the GUIDE. There are three aspects of the student record for which provision had to be made: storage, collection, and utilization of the data.

Storage: From a conceptual point of view, a student's name could have been viewed as simply another term in the database, and at one point in time this approach was seriously considered. However, the overwhelming amount of data which had to be maintained precluded the implementation of this approach (data was maintained for over 700 students this past semester). The most logical alternative was to utilize the storage space which the PLATO system automatically allocates to each student registered on the system. This includes both disk storage and storage which is available while the student is studying a lesson (both "student" and "router" variables fall in this class; the GUIDE uses the router variables). The drawback to this approach is that data is maintained only for those students in PLATO courses under the control of the Department of Computer Science. In practice, the GUIDE is utilized

by a much broader audience. This means that for these people, the GUIDE is unable to answer requests which require data from a student record.

Collection: The collection of data for the student record is made possible by a special "router" lesson provided by the PLATO system. The router lesson controls the student before and after he enters an instructional lesson, and makes it possible to store information about what lesson was entered, how much time was spent in it, and what grade (if any) was received. Storing data in the router variables is quite convenient in a router lesson, and can often be accomplished when other methods of storage will not succeed (e.g. writing into a disk file can fail if a student has made an "abortive" exit). This was another reason for utilizing the facilities provided by the PLATO system.

Utilization: There are three types of utilization of the student data.

(1) The GUIDE uses the data in responding to certain types of requests.

(2) An administrative PLATO lesson was written which makes the data available to an instructor. This lesson allows the instructor to scan through all the data for his class, or to review the data available on any particular student. (A student is also permitted to view his own record.)

(3) It is anticipated that various analysis routines will be written to provide cross-sectional statistics on all the data which is available. These statistics could be used to analyze student performance (e.g. time spent, grades achieved) as well as lesson characteristics (e.g. average time required to complete a lesson).

A sample student record is shown in Figure 17. The meaning of each field of information is as follows:

(a) "Name" is the student's name. The GUIDE uses the same name as used by the student in signing onto the PLATO system.

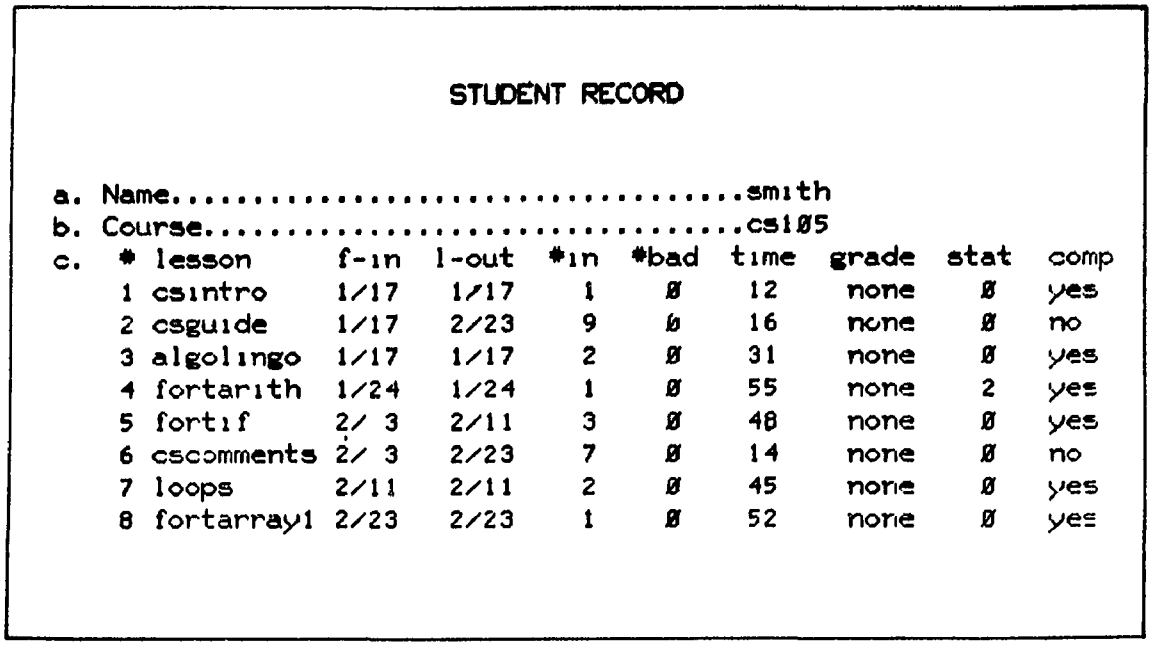
(b) "Course" is the name of a course in the GUIDE database. If a request does not specify a specific course, the course specified in this field is assumed. If this field is empty, the GUIDE will assume the appropriate course is the PLATO course in which the student is enrolled.

(c) This field is a list of each lesson entered by the student (in chronological order). The data on each lesson includes the name of the lesson, date of first entry to the lesson, date of last entry to the lesson, number of times the lesson was entered, number of times the student did not return to the router upon exit from the lesson (in which case the time is not accurate, and the grade and status may be inaccurate), total elapsed time spent in the lesson (in minutes), a flag marking whether the lesson was properly completed, grade received in the lesson (on a scale of 100), and a 5-bit field of status information which can be utilized in any way deemed appropriate for a given lesson.

2.2.7 Concept Space

2.2.7.1 Introduction

Every information retrieval system is faced with the problem of bridging the gap between a user's perception of the subject matter and the actual information which is stored in the system. The concept space was introduced into the design of the GUIDE to help bridge that gap.



STUDENT RECORD

a. Name.....smith
b. Course.....cs105

c. #	lesson	f-in	l-out	#in	#bad	time	grade	stat	comp
1	csintro	1/17	1/17	1	0	12	none	0	yes
2	csguide	1/17	2/23	9	0	16	none	0	no
3	algolingo	1/17	1/17	2	0	31	none	0	yes
4	fortarith	1/24	1/24	1	0	55	none	2	yes
5	fortif	2/ 3	2/11	3	0	48	none	0	yes
6	cscomments	2/ 3	2/23	7	0	14	none	0	no
7	loops	2/11	2/11	2	0	45	none	0	yes
8	fortarray1	2/23	2/23	1	0	52	none	0	yes

Figure 17. Sample Student Record

The first step in providing the GUIDE with an understanding of the discipline of computer science was to find an appropriate model for that knowledge. The attempt to systematically organize knowledge has a long history, and an investigation was made into some of the basic philosophies behind traditional library classification schemes. Three of the important approaches which have evolved in that area are the Dewey Decimal System (based on a classification tree of subjects arranged from general to specific), Cutter's Subject Headings (the basis of that used in the Library of Congress, where subject headings are assigned on the basis of a document's content, and alphabetized listings of subjects are maintained to help locate a given topic), and Ranganathan's faceted classification scheme (in which the fundamental elements and facets of a subject are carefully analyzed, and then the more complex topics are specified as combinations of these basic elements).

In spite of the many centuries devoted to the task, no one has yet developed a universally accepted technique for organizing a body of knowledge for use as a library classification. Part of the reason for this failure is probably due to the fact that these traditional approaches are not able to fully reflect the way knowledge is organized in the human mind. Borden and Nelson assert that rules for knowledge classification must come from rules generated by the human conceptual system [19]. Similarly, Farradane argues that a sound classification theory must be based on studies of the nature of knowledge and learning [20,21]. This brings one into contact with studies in such areas as epistemology and psychology, and some interesting ideas have emerged in these areas. Meredith expresses the idea that the cognitive structure in the mind is

essentially a network of relations of various orders [22]. He goes on to develop a theory about the nature of concepts based on this idea of relations [23], and asserts that to understand a concept is to apprehend correctly all the relations which determine its structure [24]. Shera reflects some of these same ideas when he states that a concept is a network or pattern of inferences, associations, and relationships [25]. Farradane expresses the thought that a classification constitutes a theory of the structure of knowledge [26], and he asserts that all analysis of information for storage and retrieval should be in terms of concepts and the relations between them [27]. Levy presents a thorough discussion on the actual relations between concepts which have been proposed by Farradane and many others [28]. Vickery also discusses many different such relations which can be defined [29].

The important point to be extracted from this discussion is not some specific set of relations to be used in modeling knowledge, but to observe that there is some consensus that the method of modeling knowledge as a network built of concepts and relations is a useful point of view. This fact is particularly emphasized in the work of Quillian and his model of semantic memory [30], and is further substantiated by experimental work which seems to confirm that this is a viable model [31,32].

For the design phase of the GUIDE concept space, the basic question to be settled was not exactly what knowledge was to be stored, but how it was to be represented. The data structure which ultimately evolved from the investigation described above is simply that of an abstract graph where the nodes of the graph are concepts and the arcs are relations between concepts. The choice of this extremely simple

yet powerful model was fully vindicated when it was put to use. It was found to be adequate to incorporate the synonym dictionary, the hierarchical classification scheme, and the term clusters which were originally proposed as separate components of the concept space. Also, it serves as a keyword index (holding all keywords which have been attached to lessons) and a thesaurus (holding all subject-matter terms known to the system and showing how they are related). Furthermore, when it was desired to build an index to the library of lessons, the concept space already provided the necessary mechanism to do so. And finally, it was possible to specify the structure of a course by means of concepts rather than a listing of lessons, again simply by utilizing the mechanisms available in the structure of the concept space.

It should be emphasized that the word "concept" is used rather loosely. Any word or phrase which is the name of a node in the concept space is called a "concept" -- even though a description such as "term," "jargon," "topic," "subject," "index heading," etc. might be more appropriate in certain cases.

The actual relations which have been used in building the concept space are presented in Section 2.2.7.2. The set of relations described there could be readily extended if it proved desirable to do so. In assembling this set of relations, the use of obscure types of relations based on elaborate theories which are not universally accepted (such as those presented by Levy [28]) was avoided in favor of using simple relations which would be intuitively clear to the user. However, if a universally accepted set of relations were to evolve, they could easily be incorporated into the structure of the concept space.

The graphics techniques which were developed for communicating the structure of the concept space to the user are presented in Section 2.3.6, and a sample of the use of these techniques was shown in Section 1.3.

2.2.7.2 Concept Record

A sample concept record is shown in Figure 18. The meaning of each field of information is as follows:

(a) "Concept name" is the word or phrase naming the concept.

(b) "Lessons used in" is a list of lessons in which this concept has been used as a keyword. This list permits the immediate retrieval of lessons related to this concept.

(c) "Relationships" is a list of relation-concept pairs which define the relationships that exist between this concept and other concepts in the concept space. The full set of possible relations is as follows:

generic-specific: This relation is used in building a classification of the discipline of computer science. Items are arranged in a "classification tree," proceeding from the general to specific as is traditionally done in library classification schemes. In a sense, this is the most important relation in the concept space, for it forms what may be thought of as a "spanning tree" over the entire graph structure which constitutes the concept space. It is intended that by following the generic-specific relation, one can eventually reach any neighborhood of the concept space.

CONCEPT RECORD

a. Concept name.....logical design
b. Lessons used in:
   1 intrologic
   2 boolex
   3 logicflow
   4 logicdrill
   5 logicblk
   6 logicgate
c. Relationships:
   1 synonym is logic design
   2 generic is hardware
   3 specific is physical properties
   4 specific is gates
   5 specific is combinational circuits
   6 specific is sequential circuits
   7 specific is logical components

Figure 18. Sample Concept Record

containor-containee: This relation was introduced in order to impose an index structure on the library of lessons. The special concept "C S Lessons Index" has as its "containees" the concepts which are subheadings for the index. These concepts in turn contain listings of the lessons included under that heading. This structure is utilized primarily by a utility routine which extracts the index headings and lesson listings from the GUIDE database and places them in another data file for use by an administrative lesson which presents the lesson index.

related: This relation links together two concepts which are in some sense related. The exact nature of the relationship is not specified.

synonym: The synonym relation indicates that two concepts are to be considered entirely equivalent by the GUIDE. This relation provides the mechanism for building a synonym dictionary within the concept space.

owner-member: This relation is used to group together clusters of terms which are often used in a particular context. For example, the terms "print," "line," "page," "column," "carriage," etc. tend to occur in discussions on printed output. The purpose of these clusters is to provide a context for terms unfamiliar to the student, and to act as entry points into the concept space (the student may request a lesson on some obscure term which is known to the GUIDE but not used as a lesson keyword; the GUIDE can then direct

the student to lesson keywords which are in the neighborhood of his original term). Whenever possible, these term clusters are linked into the classification tree at the appropriate point to facilitate full exploration of the concept space.

prerequisite-sequel: This relation is used primarily in building the model of a course which is specified by concepts rather than by a list of lessons (see Section 2.2.5). With this relation, an instructor is able to specify certain concepts which should be mastered before subsequent concepts can be studied.

2.3 The GUIDE Processing Algorithms

2.3.1 Introduction

In a very real sense, the GUIDE can be viewed as an information transformation machine. The solid lines and arrows along the outside edge of the block diagram in Figure 12 can be thought of as a conveyor belt along which pieces of information are transported. At four places along that path (word to term translator, English translator, request processor, and response generator) the information currently on the conveyor belt is combined with additional raw materials (data from the database) and transformed to a new form. Eventually, the final product (the response) has been produced. The entities which perform this "transformation" process are the GUIDE processing algorithms.

2.3.2 Word to Term Translator

The word to term translator accomplishes the task of transforming a sequence of input words into a sequence of terms recognized by the

system. This is done by extracting the leftmost word of an input request, applying a hash function to that word, and looking in the hash table in the appropriate location. Entries are arranged in the hash table in such a way that it is possible to extract from the original request the longest possible substring which matches a term known to the system. The progress of the translation process is communicated to the student by underlining each term when it is found in the term dictionary. This underlining also serves to mark groups of words which are considered to be a single term. Words not found are also underlined and the first letter is marked with an underscore to indicate it is not known to the system. Words not found are simply ignored by the translator.

2.3.3 English Translator

The English translator accomplishes the task of transforming a sequence of input terms into an intermediate representation for further processing. The purpose of the translator is not to make an elaborate linguistic analysis of a request, but to assist in mapping a request onto the proper response as quickly as possible. The translator's approach to dealing with natural language can be likened to a person who is hard of hearing. Even though such a person does not always "catch" all the words someone speaks, he can usually guess the meaning of a sentence on the basis of his knowledge of the general topic of conversation and the few words he did hear clearly (including the ordering and context of those words).

The computational model used in the translation process is that of a nondeterministic finite state automaton. Based on the current

state of the automaton and the input class assigned when a term is encountered, the appropriate entry in the state table determines any addition to be made to the intermediate representation of the request, and also establishes the next state to be entered by the automaton. Each state of the automaton can be associated with a partial understanding of the request. This understanding is based on that portion of the request which has been analyzed up to that point. If the translator reaches a dead end in its search for a meaningful interpretation of the request, it backs up to the previous term and looks in the state table for an alternative interpretation. This process is continued until all choices are exhausted or a consistent interpretation has been found. A complete discussion of the translator can be found in Pradel's Ph.D. thesis [11].

2.3.4 Paraphraser

The paraphraser produces a paraphrase of the original request based on the intermediate representation produced by the English translator. The purpose of this paraphrase is twofold:

(1) The student can confirm whether his request has been properly understood by the system. If so, he can proceed to the response. If not, he can immediately rephrase his request. (Also, in many cases, he can deduce what caused the system to misunderstand his request.)

(2) While the paraphrase is being read, the GUIDE begins immediately to work on the response. Thus, in most cases there will not be a noticeable delay before the response can be generated.

Furthermore, in cases when the response has been quickly prepared, there will be some idle processor time while the student finishes reading the paraphrase. This delay will help lower the average processing time consumed by the GUIDE.

2.3.5 Request Processor

The request processor accomplishes the task of transforming the intermediate representation of a request into a specification of the type of response to be generated. By analogy with the output of the English translator, this specification has been called the "intermediate form of the response." As with all phases of the GUIDE's operation, time is of the essence. Once again, the application of some simple heuristics have made it possible for the GUIDE to operate within a reasonable time frame.

The heuristics used in the request processor are based on the principle of determining as quickly as possible which area of the database contains the answer to the original request, and what possible response of the system will display that area of the database. In some cases, the request processor simply indicates to the response generator the area of the database to be displayed (for example, the term number of a course record). In other cases, the request processor assembles some data from the database and passes that information to the response generator (for example, a list of term numbers of lesson records which match a given specification).

Probably the most time-consuming task performed by the request processor is assembling a list of lessons which match a given specification.

This is done by starting with a list of lessons and then systematically removing from that list all lessons which do not match the full specification. The starting list is the union (or intersection) of all lessons attached to the or-list (or and-list) of keywords mentioned in the request. (An "or-list" is keywords connected by "or"; for example, "What do you have on trees or stacks or queues?" Similary, an "and-list" is keywords connected by "and"; for example, "Give me some lessons on fortran and arrays.") If no keywords were used in the request, then the list of lessons attached to the author name specified in the request is used. If no author name was mentioned, then the list of lessons in the course outline or the student record is used. If none of the above specifications have been made, then the starting list is simply the list of all lessons in the lesson catalog. The specifications which can be made in a request include subject matter keywords, relationships to other lessons (e.g., prerequisite), author name, course name, lesson type, type of exercises, level of difficulty, and relative sequence (first, next, or last).

2.3.6 Response Generator

The response generator accomplishes the task of transforming the intermediate form of the response into a display which can be presented to the user. There are four basic types of response which will be discussed below: list of lessons, record display, graphics display, and feedback message.

2.3.6.1 List of Lessons

The response to a large number of requests presented to the GUIDE is a list of lessons matching a given specification. For this

response, the response generator simply produces a listing of the name and abstract of the lessons which have been retrieved. A sample list of lessons found in response to the question "What do you have on arrays?" is shown in Figure 19.

2.3.6.2 Record Displays

Record displays are generated for lessons, courses, and students. Examples of each type of display have already been shown in Figures 15, 16, and 17 respectively. In the original design of the response generator, it was intended that most requests would receive a prose response. For example, the request "Who wrote fortdo?" would receive the response "The author of fortdo is Barta." The reply to "What is my next lesson in CS 105?" would be "Your next lesson in CS 105 is fortif." As an intermediate step in the development toward that end, it was decided to display the entire record which contained the piece of information which had been requested. For example, the lesson record for fortdo would be displayed in response to all of the following questions: "Tell me about fortdo," "Who wrote fortdo," "What is the expected grade for fortdo," etc. Similarly, the course record for CS 105 would be displayed as the response to questions like "What lesson is next in CS 105," "Give me a list of lessons for CS 105," "Is binsearch a lesson for CS 105," etc. This approach to the response proved to be so successful that implementation of the prose approach was abandoned. Some of the reasons for this decision are as follows:

(1) Presentation of the full set of available information anticipates in advance a whole sequence of potential questions. For

The lessons matching your specification include:

#	Lesson	Abstract
1	basicarray	Arrays in BASIC
2	oldarray	PL/I Arrays
3	pliarrayx	Advanced Examples of PL/I Arrays
4	pliarray	PL/I Arrays
5	fortarray2	Two Dimensional Arrays
6	fortarray1	One Dimensional Arrays.
7	aplvector	APL Vectors

Figure 19. Sample List of Lessons

example, after finding out what a given lesson is about it is likely that one might then ask who wrote it, and then what type it is, and then how long it takes, etc. With the entire lesson record already displayed on the screen, these questions have already been answered.

(2) It is possible to handle a large class of poorly phrased questions ("poor" in terms of the translator's ability to understand it). It is fairly easy to deduce one is asking for something about the lesson "fortdo" even though exactly what is desired is unclear.

(3) Additions or modifications in the type of data stored about lessons can be retrieved without changes to the translator.

(4) Each request presented to the GUIDE receives a generous amount of information in response. Information which the student does not want can simply be ignored. This approach is more effective than the opposite mode of behavior where one receives only exactly what he asks for: it would be difficult to keep track of all the information that could be made available, and even more difficult to remember exactly how to ask for it.

(5) Since a generous amount of information is given for every request, it is possible to write shorter requests and still obtain the desired information. Analysis of requests presented to the GUIDE (discussed more fully in Section 3.3) show that students move quickly in the direction of shorter inputs. In recognition of this trend, shortcuts were placed in the translator which allowed a single word (e.g., the lesson name "fortdo" or the keyword "fortran," or the course name "CS105") to constitute a valid request. It was observed

that students made generous use of this capability as soon as it was implemented.

(6) It is possible for a small, compact set of routines to generate the necessary responses. (In fact, it was possible to use the same routines as used in the database editor.) This constitutes a significant savings in the space consumed by the display generation routines.

2.3.6.3 Graphics Displays

One of the challenging research tasks in implementing the GUIDE was the development of an effective means of communicating the structure and content of the concept space. The network of concepts possesses a very rich structure of interrelationships which is difficult to describe. Fortunately, the PLATO terminal provides a graphics capability which helped solve that problem. The adage that "a picture is worth ten thousand words" reveals the principle which was utilized in communicating the concept space to the user. The human being is a superb information processor, and the GUIDE's problem is simply to get that information out in a form which can be readily assimilated.

The general problem of presenting visual displays of a graph structure is quite difficult, as evidenced in the work of Maguire [33]. However, by utilization of some appropriate heuristics, it was possible to develop three different types of graphical displays which help present different points of view of the concept space: the neighborhood, hierarchical, and mixed mode displays.

The neighborhood display shows the concepts which lie in the immediate neighborhood of a given concept. It gives sort of a "worm's eye view" or "bottom-up view" of the concept space. Figure 20 shows a sample neighborhood display. The first circle of nodes shows the "first generation" of concepts--those that are directly related to the central concept of the display. The secondary circles of nodes show the "second generation" of concepts--those that are directly related to the first generation concepts (and hence are two generations away from the central concept). Exploration of the concept space can be accomplished by requesting successive displays where the central node in each new display has been selected from the first or second generation of the previous display. This movement can be illustrated as in Figure 21 where one can envision the movement of a locus of attention gradually over an area, as in moving a magnifying glass over a page. This capability of exploring neighborhoods in the concept space deals quite effectively with a fundamental problem identified by Ranganathan when he said that classification amounts to mapping on a line the many dimensions of the universe of knowledge, and the crucial problem is that the "immediate-neighborhood" relation be preserved [34]. The GUIDE is able to extract an "immediate-neighborhood" from a multidimensional knowledge space (the concept space, where in a sense each relation in the space defines a new dimension) and project that neighborhood onto a two-dimensional plane for observation.

Whereas the neighborhood display gives a sense of "distance" in the concept space, the hierarchical display imparts a sense of "direction." It gives sort of a "bird's eye view" or "top-down view"

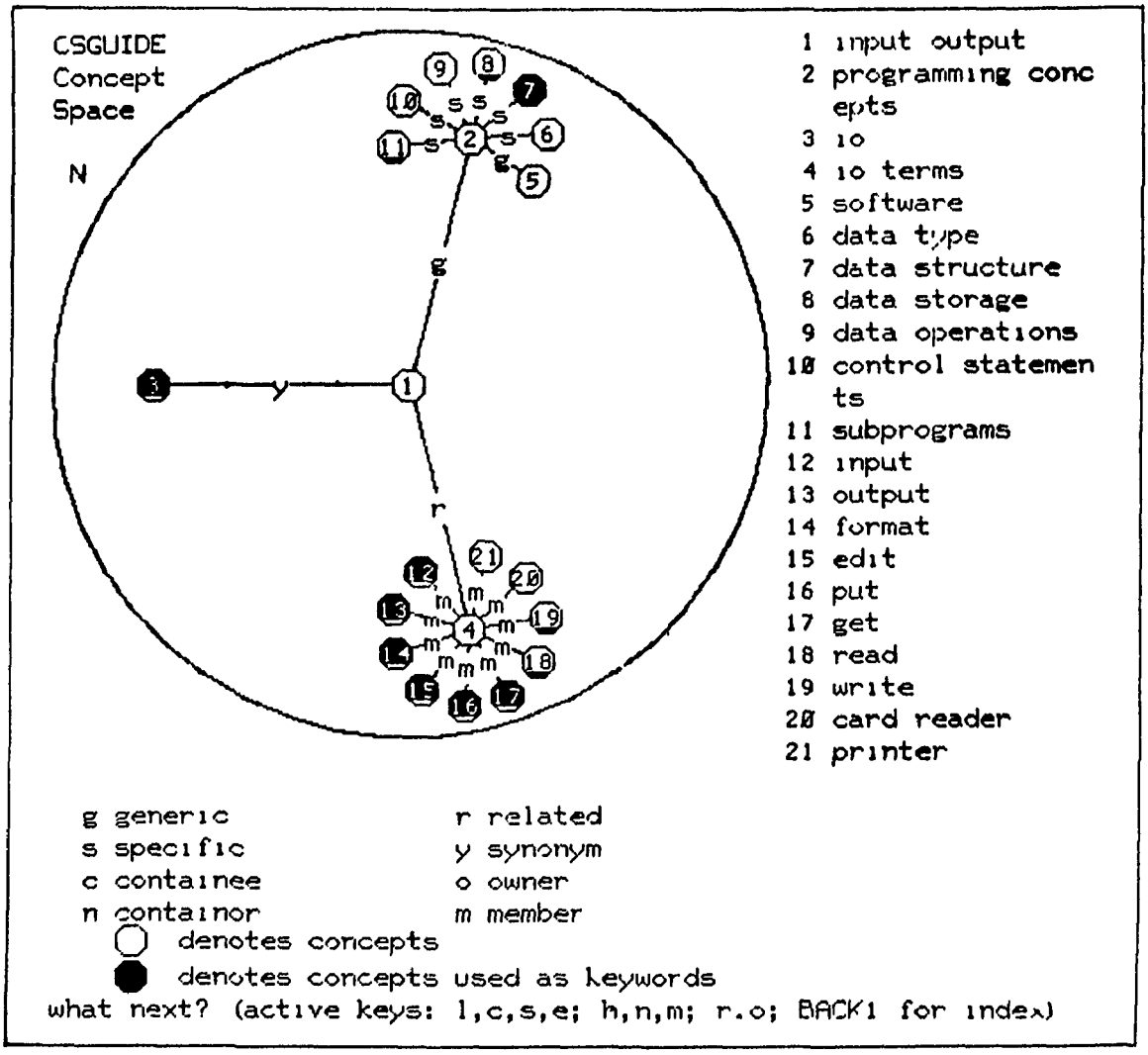


Figure 20. Sample Concept Space Neighborhood Display

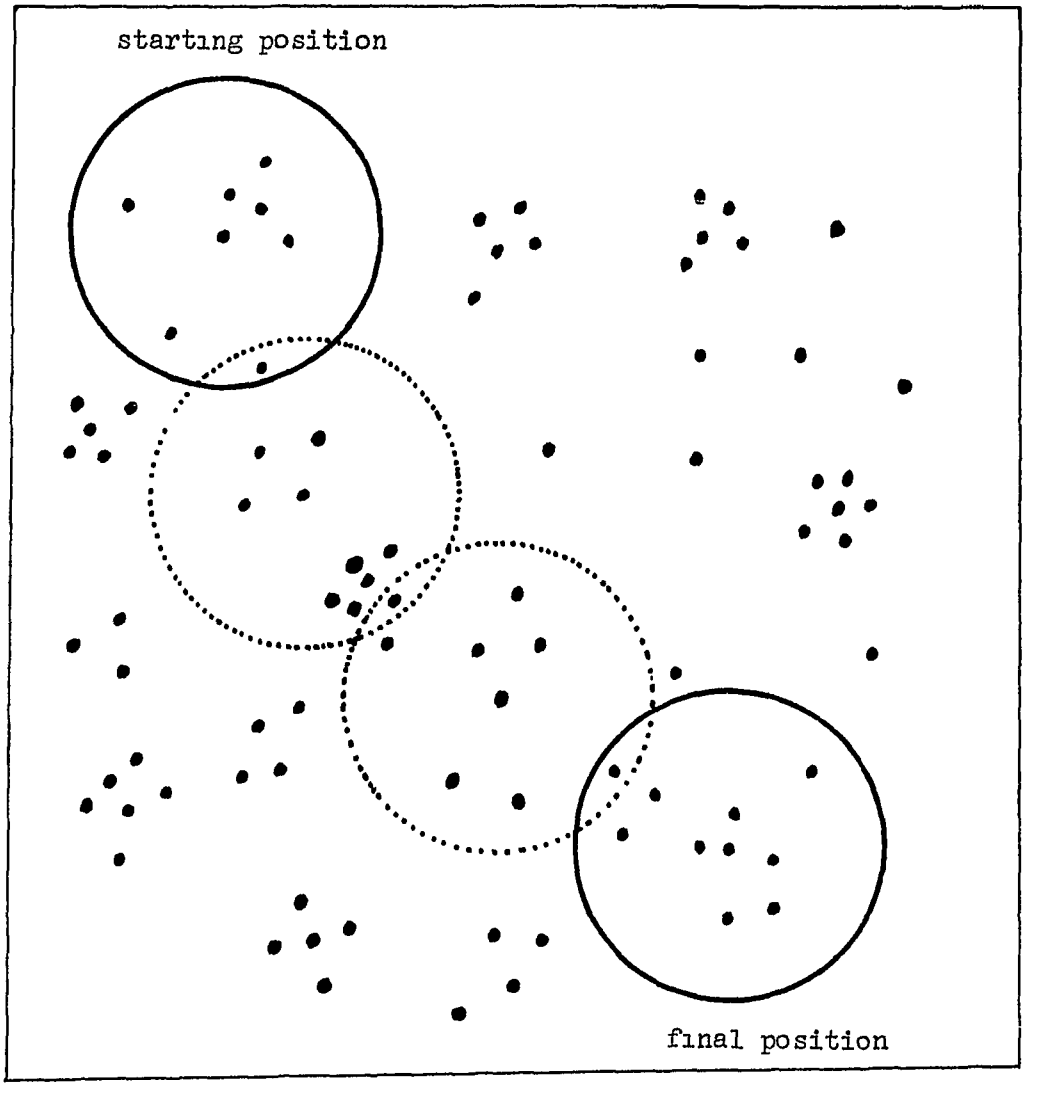


Figure 21. Illustration of Movement through Neighborhoods in Concept Space

of the concept space. Figure 22 shows a sample hierarchical display. Basically, the hierarchical mode enables one to traverse the classification tree, pursuing topics by narrowing or expanding one's scope of interest. The sense of direction imparted by the hierarchical mode is how "high" or "low" a given concept is in the classification tree, and where it is relative to the "root" of the tree--the concept "computer science."

The mixed mode display was introduced to facilitate the graphical presentation of the set of lessons which are attached to a given concept. It is called "mixed" since both lessons and concepts appear in the display. An example of such a display is shown in Figure 23. An interesting interpretation of this display is that in a very real sense, a lesson can be viewed as a relation in the concept space, providing a link between various concepts. For example, in Figure 23, the concept "do loop" has the "fortdo" relation with the concept "fortran."

An extremely interesting by-product of the development of these graphics displays was the observation that the lesson catalog possesses a structure which is identical to that of the concept space. From an abstract point of view, there is no difference between concepts which are connected by relations, and lessons which are connected by relations. The idea of a "lesson space" was developed, and the graphics capabilities were extended to include the display of lessons as well as concepts.

A sample of a neighborhood display in the lesson space is shown in Figure 24. This very useful display enables one to quickly locate lessons which are related in some way to a given lesson. Although the actual relations used in defining the lesson space are different from

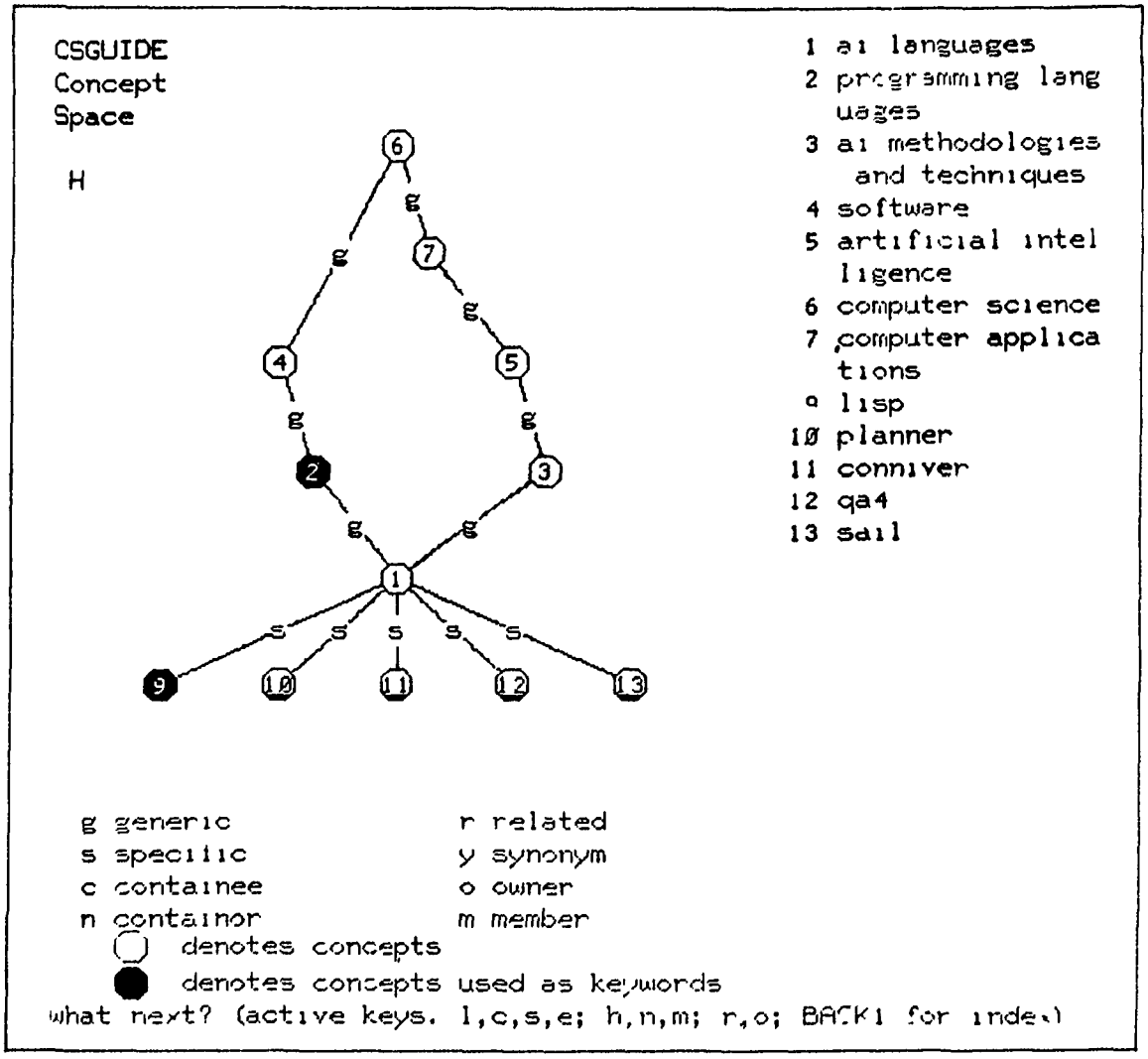


Figure 22. Sample Concept Space Hierarchical Display

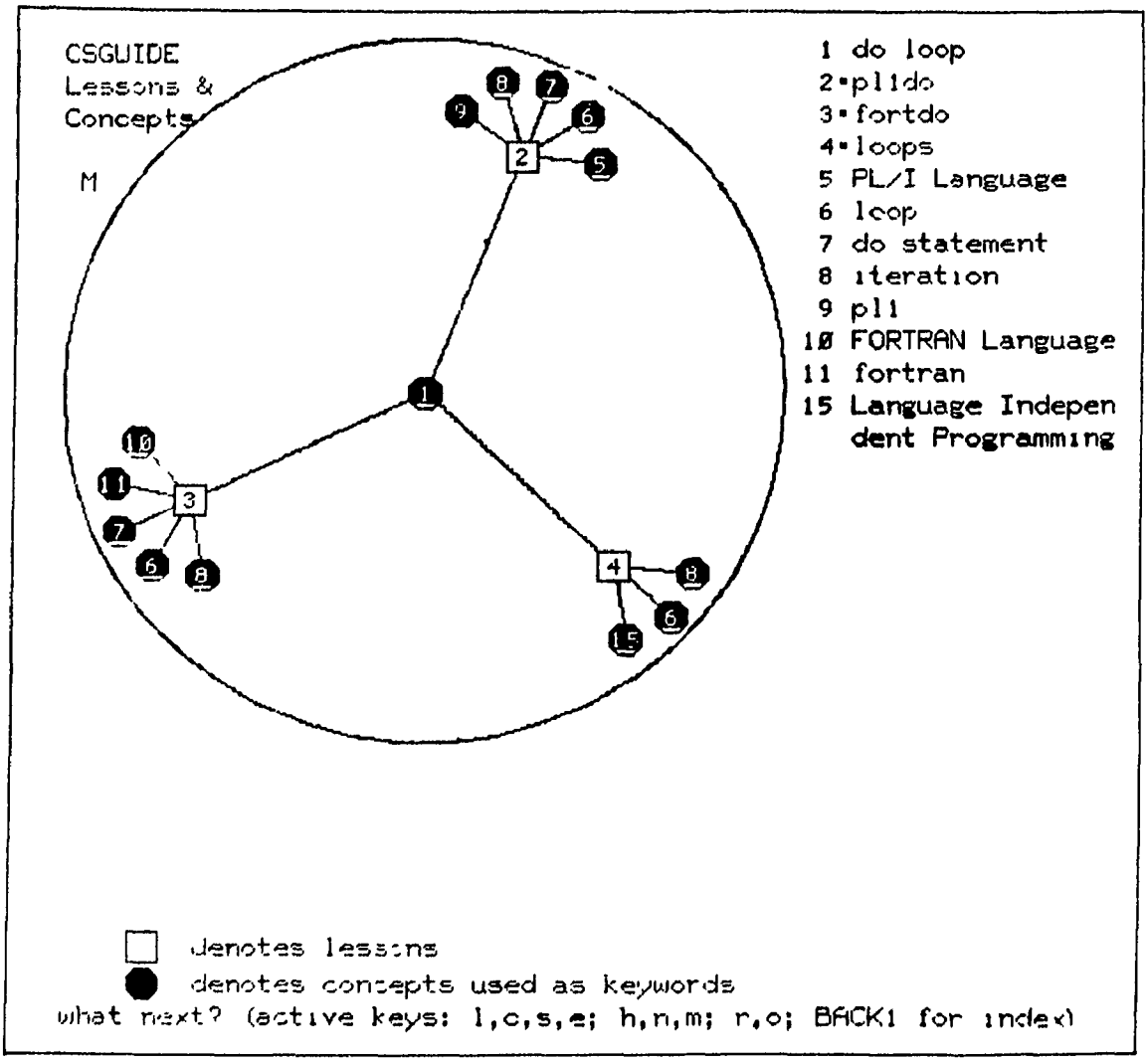


Figure 23. Sample Concept Space Mixed Mode Display

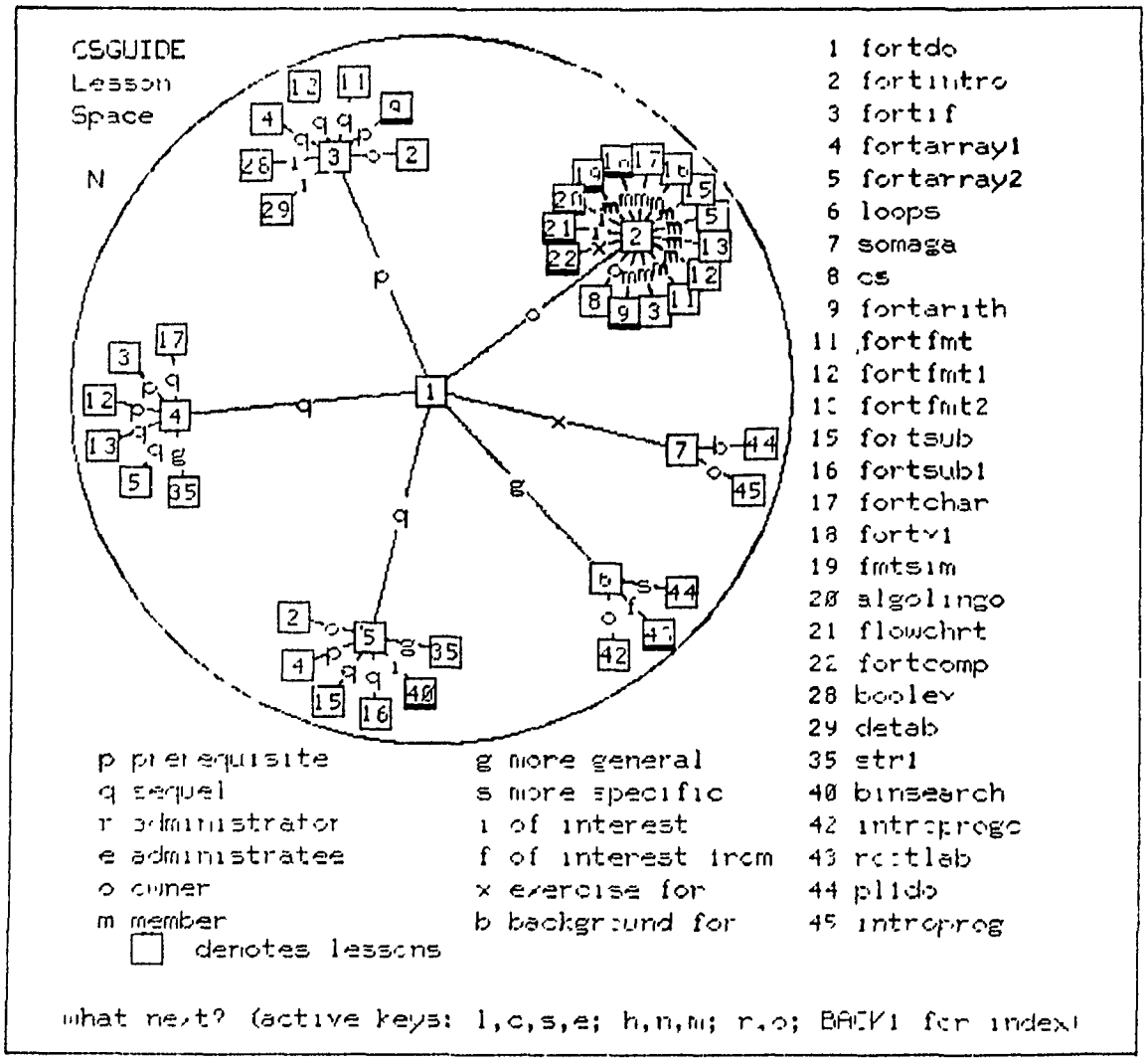


Figure 24. Sample Lesson Space Neighborhood Display

those in the concept space, the idea of showing neighborhoods and exploring the space is equally applicable to both lessons and concepts.

A sample of a hierarchical display in the lesson space is shown in Figure 25. This reveals the richer hierarchical nature of the lesson space due to the prerequisite-sequel structure of lessons. The "root" node of the lesson space is the lesson "cs" which owns all the "entry" lessons, which in turn own the other lessons of the library (see the discussion of the owner-member relation in Section 2.2.4).

A mixed mode display of the lesson space is shown in Figure 26. Again, an interesting interpretation of this display is that a concept can be viewed as a relation in the lesson space. For example, in Figure 26, the lesson "fortdo" has the "iteration" relation with the lesson "pldo."

The basic steps involved in generating the graphics displays are as follows:

(1) The nodes to be included in the display are collected by a breadth-first traversal of the graph structure which forms the concept (or lesson) space, starting at the node which the user has specified. During this process, relations which are not desired are simply skipped. (For example, when traversing upward in the classification tree, nodes related by the "generic" relation would be included, and those related by the "specific" relation would be skipped; when traversing downward, the situation is reversed. The nodes to be included in a given traversal are indicated by setting the n^{th} bit in a special "flag word" if the n^{th} relation is desired. This "flag word" can be set by the user, or automatically set by the GUIDE.) The traversal process is terminated

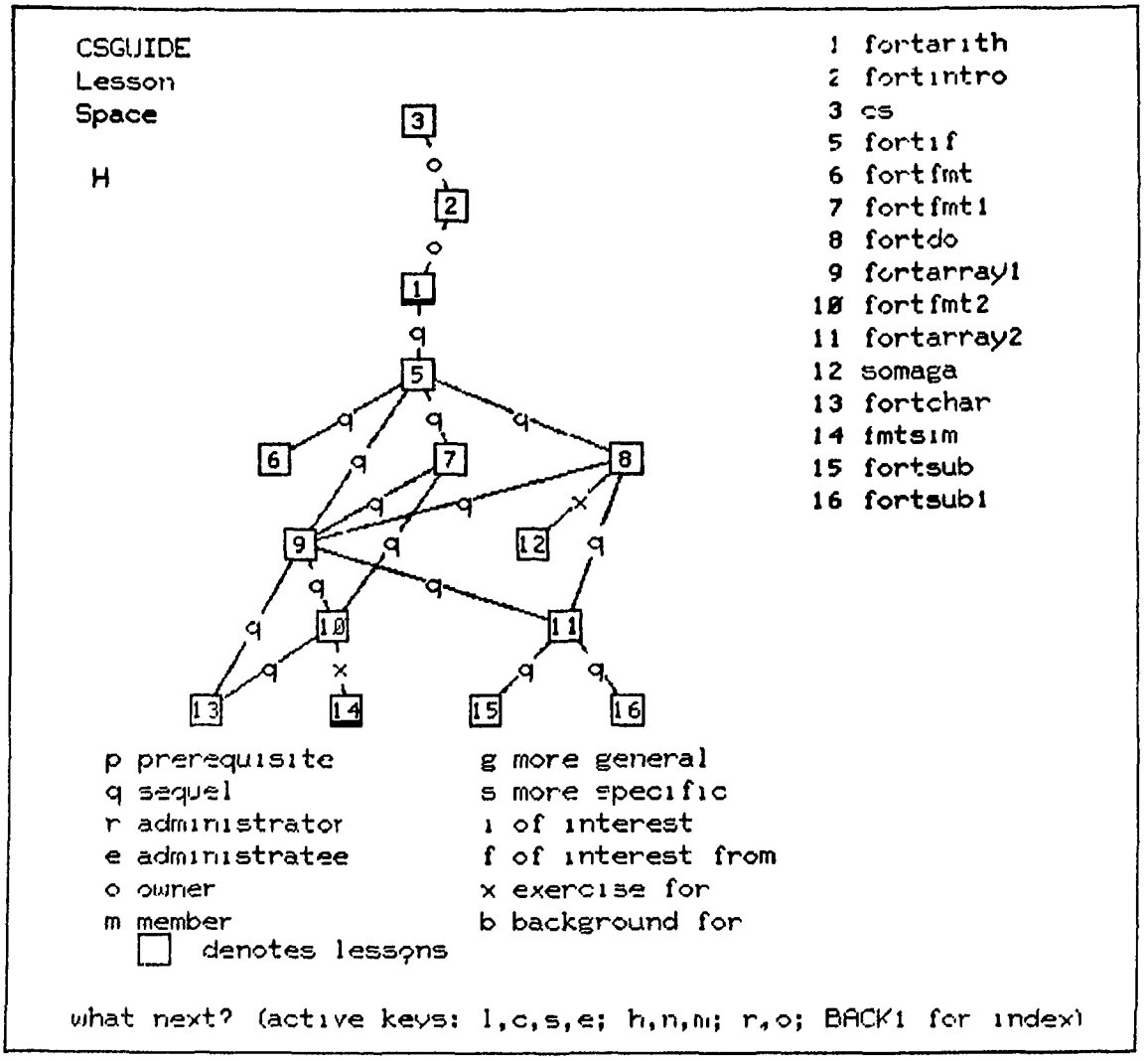


Figure 25. Sample Lesson Space Hierarchical Display

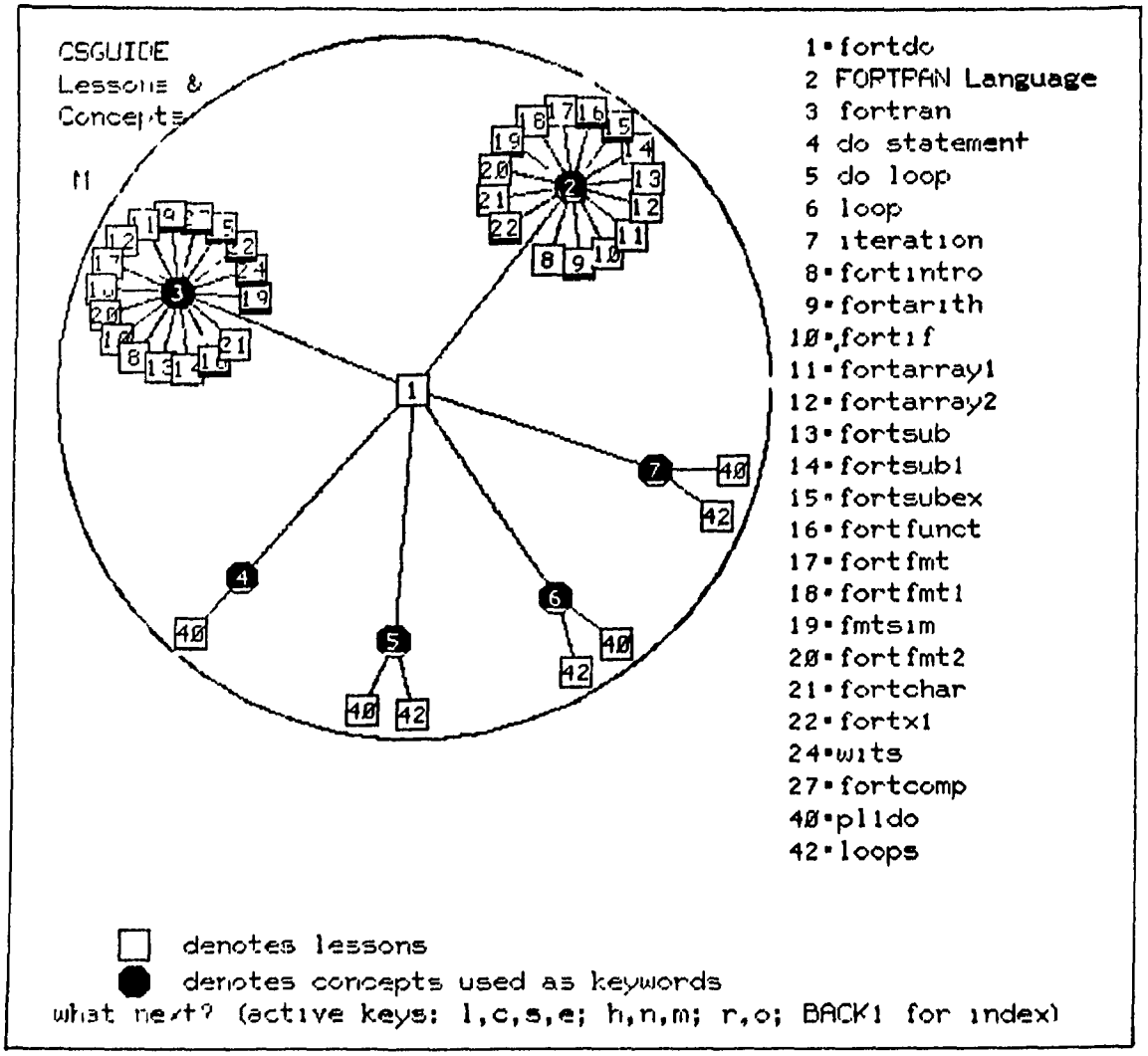


Figure 26. Sample Lesson Space Mixed Mode Display

in the neighborhood and mixed mode displays when the distance from the starting node is equal to two, that is, when the "second generation" nodes have all been collected. The process is terminated in the hierarchical display when there are no more relations to pursue, or when 30 nodes have been encountered (the limit of 30 nodes was selected somewhat arbitrarily on the basis of the fact that a display with more than about 30 nodes tends to become overly cluttered). During the traversal process, the distance from the starting node is recorded, and flags are set for each node indicating whether it is a lesson or concept and whether it is a "terminal node." (A concept is terminal if it is related to at most one other concept; a lesson is terminal if it is related to at most two other lessons. Terminal nodes are marked with an underscore in the graphics display.)

(2) After all the nodes have been collected, the coordinates for each node are computed. For the neighborhood and mixed mode displays, the first generation nodes are all placed a fixed distance from the central node, and the second generation nodes are all a fixed distance from the first generation node to which they are are related. The angle allocated to a first generation node is based on the number of other nodes which must be plotted around it, and the angle allocated to a second generation node is determined on the same basis. The more nodes to be plotted, the smaller the angle which is available: this leads to a "graceful degradation" of the display when one attempts to plot too many nodes. A display which is too crowded can be resolved by moving to a more sparsely populated region of the concept (or lesson) space, or by excluding some of the relations from the display. In the case of the

hierarchical display, the vertical coordinate is based on the level of the node (distance from the starting node), and the horizontal coordinate is based on whether a node is the first, second, or third, etc. node encountered on that level. The vertical spacing between nodes is equally divided among all the levels which need to be plotted, and the horizontal spacing on a given level is determined in a similar manner. In addition, the nodes on all even levels are offset slightly to the left, and those on odd levels to the right in order to avoid conflicts with nodes which have predecessors more than one level above.

(3) After the coordinates have been computed, the nodes and arcs between them are plotted on the plasma panel. The midpoint of each arc is labeled with a letter which indicates the relation which exists between the two nodes.

2.3.6.4 Feedback Response

There are three situations in which a feedback response is presented to the user

(1) If an ambiguous term is used (e.g., the word "lisp" is the name of a lesson and the name of a concept), the user is asked to clarify his intended use of the term.

(2) When the translator cannot find a valid interpretation of the request, a message describing the problem is presented (e.g., the message might suggest an alternative syntax for the request, or perhaps suggest that additional information be included in the request). Quite often, the problem is simply that a significant word of the request (e.g., a name thought to be a valid lesson or concept) is not in the GUIDE database.

(3) When the request processor does not have sufficient information to process the request (e.g., there is no student record available, or the name of the course outline to consult cannot be deduced) a message stating the problem is presented. Also, an appropriate message is given when the GUIDE is unable to locate any lessons which match the specification given in a request.

3. PERFORMANCE OF THE GUIDE

3.1 Introduction

The overall structure of the GUIDE is shown in Figure 27. The title page identifies the GUIDE and depicts its purpose by means of an entertaining animation. The index page provides ready access to the basic areas of the GUIDE and serves as a "base of operations" in that from nearly every place in the GUIDE, one can reach the index by a single keypress.

The help index provides a listing of the various types of assistance which is available (e.g., an explanation of the purpose of the GUIDE, examples of typical valid requests, listings of the terms known to the system, an explanation of the relations which exist in the concept space and lesson space, etc.). The help pages present the actual display of that information.

The advice page presents a comparison between a student's record and the course outline of the course in which he is enrolled. The "advising function" consists of highlighting any differences which may exist. For example, the student may have spent less than the suggested amount of time on a lesson, or achieved a lower grade than the specified level of achievement. In addition, the GUIDE identifies which lesson from the course outline the student should study next.

The greatest portion of a user's interaction with the GUIDE occurs in the areas of requests and responses. On the request page,

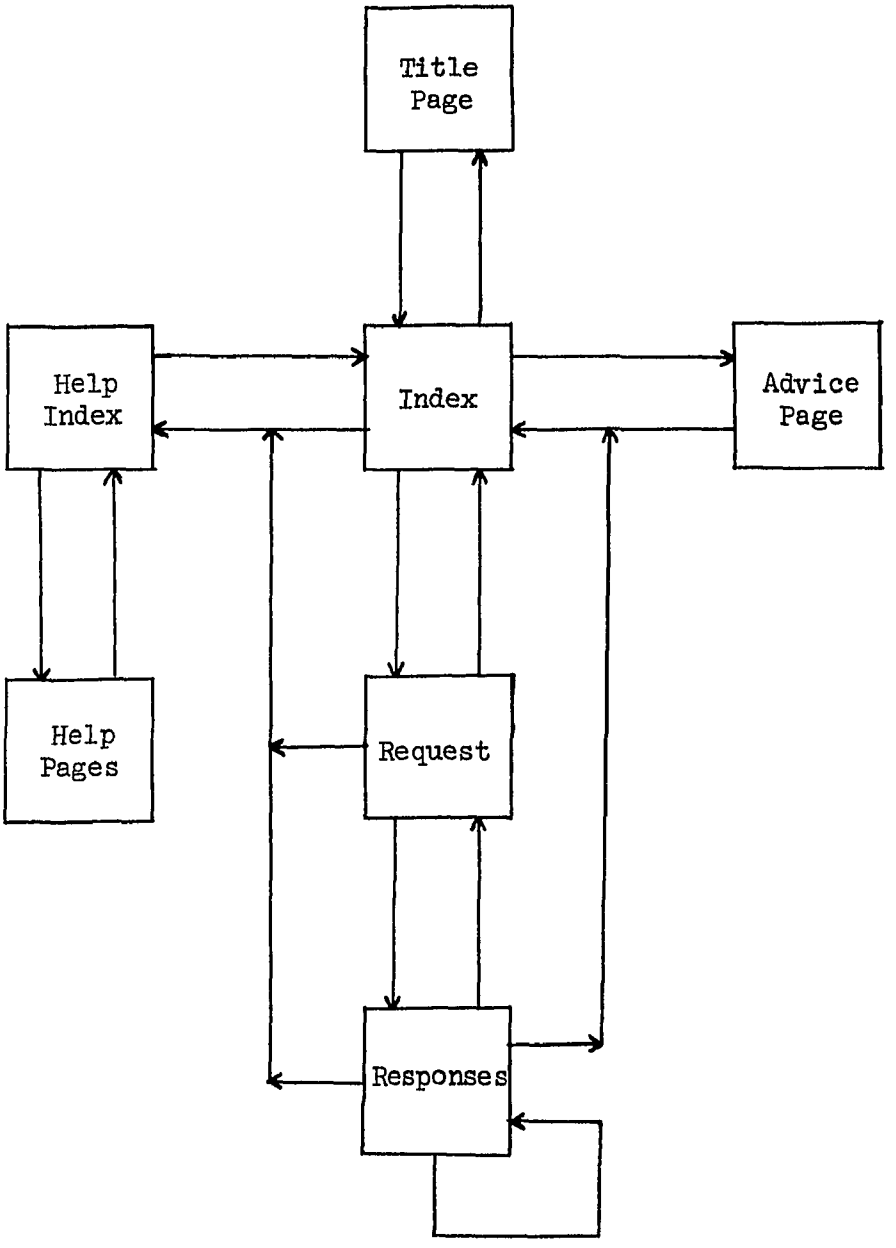


Figure 27. Overall Structure of the GUIDE

the student types in his question, sees the paraphrase, and receives most of the feedback messages which might be generated. The possible responses which can be made were discussed in Section 2.3.6.

Note that there are ample access paths between all the areas of the GUIDE. Special mention should be made of the loop between responses. To facilitate rapid exploration of the data which can be made available, provision was made for the user to elicit a sequence of responses from the system with the fewest possible number of keypresses. After every response has been displayed, it is possible to get a further response by pressing just 2 or 3 keys. (The user is prompted on which actions are available so that it is not necessary to memorize which keys must be pressed.) This feature is representative of the human engineering factors which were considered in the design of the user interface with the GUIDE. Extensive experience with the PLATO system has shown that features which make life easier for the user are a large factor in the usefulness and success of a project.

This chapter will be devoted to examples of requests and the responses made by the GUIDE, illustrating the broad range of requests which can be handled by the translator and the ability of the paraphraser to restate the user's request. (An example of the "exploration mode" of interaction with the GUIDE has already been presented in Section 1.3.) Finally, an analysis of actual student requests presented to the GUIDE will be discussed in Section 3.3.

3.2 Sample Requests and the GUIDE's Response

3.2.1 List of Lessons

A large number of the people who come to the GUIDE for help are seeking a lesson (or a set of lessons) which matches a given specification. The attributes which can be included in specifying a lesson were discussed in Section 2.3.5. Some examples of typical requests are as follows:

```
   give me something on arrays in fortran

If I've understood things correctly, I believe you want
something specified by:
    the list of keywords:  array, fortran
-- is that right?
```

```
   do you have anything on stacks deques or queues

If I've understood things correctly, I believe you want
to know whether there exists
something specified by:
    the or-list of keywords:  stack, deque, queue
-- is that right?
```

```
   give me some introductory instructional material on arrays

If I've understood things correctly, I believe you want
something specified by:
    the list of keywords:  array
    the module type:  instructional
    the level of difficulty:  introductory
-- is that right?
```

what are the prerequisites of an exam on fortran

If I've understood things correctly, I believe you want
the prerequisites of
something specified by:
    the list of keywords: fortran
    the module type: exam
-- is that right?

is fortdo on do loops

If I've understood things correctly, I believe you want
to know whether
something named fortdo is among the set of
something specified by:
    the list of keywords: do loop
-- is that right?

i want an introductory level exam by barta

If I've understood things correctly, I believe you want
something specified by:
    the author name: barta
    the module type: exam
    the level of difficulty: introductory
-- is that right?

give me some practice in programming fortran

If I've understood things correctly, I believe you want
something specified by:
    the list of keywords: fortran
    type of exercises: programming
-- is that right?

do you have anything on lisp

HELP! There is some ambiguity here...

Please press the number corresponding to your intended use of the word (or phrase beginning with) "lisp"

1. lesson name
2. concept (subject matter keyword)
3. course name
4. author name
5. other...

is fortcomp a compiler

If I've understood things correctly, I believe you want to know whether
something named fortcomp is among the set of
something specified by
the module type: compiler

The examples above illustrate the flexibility of the translator in recognizing the intent of various forms of requests.

The request where the GUIDE asks for help shows how ambiguity can be resolved by interaction with the student. (The word "lisp" is used both as the name of a lesson and as the name of a concept in the GUIDE database. The intended use of the term must be determined before the GUIDE can deduce the proper response to be given.) Each of these requests would lead to a listing of lessons (as shown in Figure 19), or to a message indicating that no lessons with the given specification were found.

3.2.2 Lesson Record

Questions asking for information about a specific lesson lead to the display of a lesson record. The following are some examples of such requests and their paraphrase:

tell me about fortdo

If I've understood things correctly, I believe you want
something named fortdo
-- is that right?

give me the abstract of fortdo

If I've understood things correctly, I believe you want
the abstract of
something named fortdo
-- is that right?

how long does it take for fortdo

If I've understood things correctly, I believe you want
the schedule to achieve in
something named fortdo
-- is that right?

what is the expected grade for fortdo

If I've understood things correctly, I believe you want
the grade required in
something named fortdo
-- is that right?

fortdo

If I've understood things correctly, I believe you want
something named fortdo
-- is that right?

tell me about fortx

Sorry, your request is not clear to me. Are you
sure it is within the scope of requests the GUIDE
can answer -- or were significant words in your
request not in the GUIDE's vocabulary?

The last example illustrates the GUIDE's response if the student tries to use a lesson name which is not in the database. The response to the other questions would be a lesson record display like that in Figure 15.

3.2.3 Course Outline

A request which requires information from a course outline leads to the display of a course record. Requests in this category include anything asked about the future (one's goals as specified in a course), as well as requests explicitly mentioning a particular course. Some examples of this type of request and their paraphrase are shown below:

what do i do next for cs105

If I've understood things correctly, I believe you want
something specified by:
    the course: cs105
    it is yet to be taken
    the relative sequence of: next
-- is that right?

list the lessons in cs105

If I've understood things correctly, I believe you want
something specified by:
    the course: cs105
-- is that right?

do i take any advanced material by barber on fortran
before may 7

If I've understood things correctly, I believe you want
to know whether there exists
something specified by:
    the list of keywords: fortran
    the author name: barber
    the level of difficulty: advanced
    it is yet to be taken
    the time period ending: May 7
-- is that right?

what grade should i get in fortdo

If I've understood things correctly, I believe you want
the grade required in
something named fortdo
-- is that right?

cs105

If I've understood things correctly, I believe you want
something specified by:
    the course: cs105
-- is that right?

what should i do next

If I've understood things correctly, I believe you want
something specified by:
    it is yet to be taken
    the relative sequence of: next
-- is that right?

(Press NEXT to proceed, BACK to restate request)

Course outline to consult cannot be determined --
please include the course name in your request.

The last example shows how the GUIDE responds when a course outline is not available. There are three ways to deduce which course outline to consult. (1) the information is explicitly stored in a student's record, (2) the PLATO course in which the student is enrolled has the same name as a course in the GUIDE database, or (3) a course name is explicitly mentioned in the request. The other examples shown above would lead to the display of a course outline like that shown in Figure 16.

3.2.4 Student Record

Any requests about past performance lead to the display of a student record. Some examples of such requests are as follows:

what did i do yesterday

If I've understood things correctly, I believe you want
something specified by:
    it has already been taken
    the time: May 25
information is from your student record
-- is that right?

have i taken anything on fortran yet

If I've understood things correctly, I believe you want
to know whether there exists
something specified by:
    the list of keywords: fortran
    it has already been taken
information is from your student record
-- is that right?

what grade did i get in fortdo

If I've understood things correctly, I believe you want
your grade obtained in
something named fortdo
information is from your student record
-- is that right?

how much time did i spend in fortarith

If I've understood things correctly, I believe you want
the schedule you achieved in
something named fortarith
information is from your student record
-- is that right?

what did i do since march 23

If I've understood things correctly, I believe you want
something specified by:
    it has already been taken
    the time period beginning: March 23
information is from your student record
-- is that right?

(Press NEXT to proceed, BACK to restate request)

Sorry, you have no student record available and there is
insufficient information to answer your request.
[At the present time, only students routed by
csrouter have a student record.]

The last example illustrates the response given when the GUIDE does not maintain data for the student who has asked the question. The other examples would receive a student record display similar to that shown in Figure 17.

3.2.5 Graphics

Most of the graphics responses are explicitly requested by the student when he is exploring the concept space or lesson space as illustrated in Section 1.3. However, the GUIDE does initiate a graphics response for requests about lessons related in some way to a given lesson. Originally, the list of lessons response was given in such cases, but it was felt that the graphics response communicated a more effective package of information, especially in the case of inquiries about the prerequisites and sequels of a given lesson. Some examples of this type of request are shown below:

```
what comes before fortdo
```

```
If I've understood things correctly, I believe you want
the prerequisites of
something named fortdo
-- is that right?
```

```
what do you have like fortdo
```

```
If I've understood things correctly, I believe you want
something similar to
something named fortdo
-- is that right?
```

i want the sequels of the owner of fortarray2

If I've understood things correctly, I believe you want
the sequels of
the owners of
something named fortarray2
-- is that right?

(Press NEXT to proceed, BACK to restate request)

No lessons having the specified relationship were found.

The last example shows the response when there are no lessons having the specified relationship. The other requests would receive either a hierarchical display like that in Figure 25, or a neighborhood display as in Figure 24, depending on the nature of the relations which were mentioned in the request.

3.3 Analysis of Requests

For a period of 9 months, a record was kept of all requests presented to the GUIDE along with the response which was given. Over this span of time, more than 4000 requests were recorded and analyzed. These requests came from people registered in 77 different PLATO courses. Of these, only 8 courses were managed by the Department of Computer Science. However, 80% of the requests were from those in the computer science courses. Nearly 15% of all the requests came in during the spring semester alone from computer science students registered for formal course work on PLATO. Thus, even though these students were already provided with a listing of lessons which they should take, many of them went to the GUIDE for further information about the library of computer science PLATO lessons. A summary of the nature of the requests which were received is shown below:

Within domain of GUIDE.....81%

Within translator's domain of discourse.....67%

List of Lessons.....30%
Lesson Record.....16%
Course Record.....17%
Student Record..... 3%
Related Lessons..... 1%

Outside translator's domain of discourse.....14%

Too General..... 5%
Beyond Scope..... 6%
Misspelled..... 2%
Other..... 1%

Outside domain of GUIDE.....19%

Other Subjects..... 2%
Games..... 4%
Test Sequences..... 6%
Conversation..... 3%
Graffiti..... 2%
Other..... 2%

A request was considered to be within the domain of the GUIDE if it dealt with something in the area of computer science, or with the other types of data maintained by the GUIDE. A request was counted within the translator's domain of discourse if it could be handled by the translator--including the case of requests which were not of the proper form, but were clearly explained as such by an appropriate feedback message. For example, the question "Did Barta write fortdo" must be phrased in the form "Is fortdo by Barta" or "Was fortdo written by Barta," and a feedback message explains that the name of a lesson must precede its specification. Questions within the translator's domain of discourse are classified according to the type of response which would be given. Those outside the translator's domain of discourse include requests which were too general (for example, "What lessons

are available," "Let me see some of your lessons," "I want to see a catalog," and "Show me an index"), and some which went beyond the scope of requests which the translator was originally designed to handle (for example, "Elaborate on the differences between fortran 2 and fortran 4 please," "What lesson am I working on," "Show me my student record," "How can I run a lesson," "What courses are available," and "How is a computer made").

Those questions outside the domain of the GUIDE include requests for information on other subjects (for example, "Give me some accounting lessons," and "What do you have in English"), and a relatively large number of requests for games. (There are several recreational and educational games implemented on PLATO which are not usually made available to students enrolled in formal course work. However, the students quickly learn about the existence of these games and try to get to them.) In addition, a number of requests are simply "test sequences" in which an individual is tinkering around and watching how the GUIDE responds. (The following sequence is an example: "Exam" "CS101 exam" "Fortdo exam" "When do I take the fortdo exam" "Fortdo exam when" "When fortdo exam".) Finally, some requests are simply conversation with the GUIDE (e.g., "What is your name," "How many people are on the system tonight," "Why are you so dumb," "What time is it," and "How do you work"), and of course there are the inevitable graffiti (e.g., "In feet or meters how high is up," and "This statement is false").

The effort of monitoring the requests received by the GUIDE was quite profitable. It was possible to identify additions which needed to be made to the translator's vocabulary (e.g., the phrase

"next lesson after" was made equivalent to "sequel"), and to note that special provision needed to be made for requests which were too general (a feedback message was added which directs the student to an alphabetical listing of the lessons known to the GUIDE). Also, it was observed that approximately 30% of the requests received consisted of a single word, and the translator was adjusted to recognize this as a valid form of request.

4. KEY FEATURES OF THE GUIDE AND COMPARISON WITH OTHER SYSTEMS

4.1 Concept Space and Lesson Space

One of the key features of the GUIDE is the conceptual point of view of an abstract information space which can be explored by a student who is searching for some information. Some of the distinctive features of this space are the use of a graph-theoretic model to represent the information, the dual nature of the "concept space" and "lesson space" within the total information space, and the graphical methods developed for displaying the space. This section will be devoted to a discussion of some of the relevant literature in these areas.

The idea of an "information space" which is explored in the search of a particular item of information can be traced back to a paper by Bush in which he postulates a "memex" machine which would be used by a research worker to store his personal library [35]. An actual working system which probably comes the closest to implementing these ideas is the "augmented knowledge workshop" of Englebart [36]. By use of a CRT display, a typewriter keyboard, a special function keyboard, and a "mouse" (a device for moving a cursor around on the CRT display for specifying certain screen positions) an individual can rapidly move through files of information. The predominant structure of these files is hierarchical in nature, although provision is made for cross-links which the user may wish to make.

An alternative to a hierarchically structured information space was introduced by Taube in his development of coordinate indexing [37-39]. He presented coordinate indexing as a means of forming associations which are severed by a classification scheme. One can view the set of all n keywords used in indexing documents as forming an n-dimensional vector space. The content of a given document would be specified as a vector in this space. An actual implementation of these ideas is realized in Salton's SMART system [40]. He has exploited the vector space analogy by introducing the usual distance metric for such a space as a measure of the "distance" between documents. Queries for information are mapped onto a point in this abstract space, and documents within an appropriate distance are retrieved as the response.

Yet another approach to the structure of an information space forms the basis of Doyle's ideas for allowing a user to pursue associative trails of thought while searching for documents [41,42]. He suggests that a statistical analysis of word associations could form an association map which would be displayed to help a user pursue interesting avenues of thought. Some graphics displays for the display of such a map are suggested by Doyle [41] and Borko [43]. Unfortunately, such an arrangement of terms sometimes lacks a sense of orientation, and Doyle does not advocate that a classification approach be abandoned entirely, since a classification scheme provides useful "conceptual grooves" which help give a "sense of direction" during a literature search. The work of Treu is based on this idea of building an association map [44-46]. The association between terms is determined in his system by an analysis of the co-occurrence frequencies of terms within a document.

The idea of building a machine representation of knowledge which attempts to model the way knowledge is stored in the human mind was introduced in Quillian's model of semantic memory [30]. The application of this type of model was brought into the environment of computer assisted instruction by Carbonell in his SCHOLAR system [47]. He developed the idea of information-structure-oriented CAI whereby an automated instructional system is given some useful knowledge about the domain of discourse so that it can take some initiative in generating questions and responses which have not been explicitly programmed in advance. Wexler has developed a similar idea based on building an information net from which useful dialog can be derived on the basis of interaction with a student [48]. Koffman also presents a system based on this principle of generative CAI [49].

A point of view which corresponds to the orientation of the GUIDE is introduced by Osin in his SMITH system [50]. His approach to generative CAI is called IRO-CAI (information-retrieval-oriented CAI). The essential components of the database in SMITH are a set of frames, a set of relations between the frames, and a set of topics. A frame is a segment of text or an illustration which presents instructional material. The relations between frames are the precedence relation and the immediate successor relation which are used to specify the order in which frames are to be presented. The topics are used for indexing the frames by subject matter. The indexing by topic is qualified by a "pedagogical qualifier" which indicates how a topic is presented in a frame (e.g., the frame may introduce the topic, or contain necessary material for the presentation of the topic, or give an exercise on the topic, etc.).

In comparison with the GUIDE, the "frames" correspond to "lessons," the relations between frames to the relations between lessons, and the "topics" to "concepts." Whereas a frame in SMITH is a single display, a lesson in the GUIDE is an entire instructional sequence of displays which requires 30-60 minutes to complete. The GUIDE has a much richer structure of relations between lessons than SMITH has between frames. The GUIDE has no analogue to the "pedagogical qualifier," although such a device could be included--it would correspond to the "roles" which have been introduced to augment coordinate indexing [51]. The information which can be retrieved in SMITH is a list of topics on a given frame (analogous to the GUIDE's list of concepts for a given lesson), or a frame (or sequence of frames) on a given topic (corresponding to the GUIDE's list of lessons on a given concept). SMITH's information space has far less structure than the GUIDE, having a very simple "frame space," and having no relations at all between nodes in the "topic space."

The system which is probably the closest to the GUIDE in its structure and philosophy is that described by Brunnstein and Schmidt [52]. The system they propose is of the genre of generative CAI in some respects, but with some important differences. Whereas the systems of Wexler and Carbonell basically make "internal" use of the semantic network of information which is available, Brunnstein and Schmidt argue that a very profitable approach is to give the student the information he needs to pick his own path through the material being presented. In other words, the system's model of knowledge for the subject being studied should be displayed explicitly to the student rather than only being utilized "behind the scenes." They believe that this approach

would be valuable in helping the student form his own internalized model of the area of knowledge which the system knows about.

Brunnstein and Schmidt propose an information network where the nodes are units of information, and the arcs show the paths leading from simple to more complex elements of information. This network would be displayed to the user on a graphical output device like a CRT display. The graphics displays are not dynamically generated (as they are in the GUIDE) but are explicitly specified in advance. The authors discuss techniques for specifying such an information network for a system, using either interactive on-line methods, or a special-purpose language that could be processed off-line.

In comparison with the GUIDE, the information network corresponds to the concept space. The relations in the information network would correspond to the generic-specific, owner-member, and prerequisite-sequel relations of the concept space. Movement through the information network would be similar to that through the concept space as one searches for a desired piece of information. There is no analogue of the lesson space in their system--the information network is in a sense a self-contained entity in that the objects being sought are the pieces of information which constitute the network.

The idea of a dual "lesson space" and "concept space" wherein one can move freely between the two spaces searching for information does not seem to be present in the literature. In general, the objects retrieved in an information retrieval system are documents (or document citations) and with the possible exception of the "cites-cited by"

relation of citation indexing, the documents are not interconnected by a set of relations like those between lessons in the GUIDE. It was not until the advent of the PLATO system that an entire library of CAI lessons with a rich structure of interconnections could be manipulated in an on-line environment. Thus, the GUIDE has had a unique opportunity in developing the idea of a "lesson space."

Certain aspects of the graphics techniques developed for the GUIDE are also somewhat unique among actual working systems, although Soergel describes possible ways of picturing relationships which resemble the displays produced by the GUIDE [53]. A hierarchical mode of display essentially similar to that in the GUIDE has been implemented in several systems. Examples are the work of Thompson [54-56] and Lyman [57].

4.2 Natural Language

Another key feature of the GUIDE is the ability to deal with natural language input. At one extreme in the possible approaches to this problem is the work of those like Woods [58], Winograd [59], and Schank [60] who have devised systems which are able to perform an elaborate linguistic analysis of natural language. At the other extreme are keyword systems like that of Chai [61] which ignore syntactic constructs of the language and recognize only certain special system words. Moyne argues that natural language systems based on careful syntatic analysis are superior to simplistic keyword systems [62]. However, in a very limited domain of discourse such as that of the GUIDE, it is possible to devise computationally efficient techniques

which lead to an adequate understanding of a request without a complex syntactic analysis [11]. The basis of the GUIDE's approach is a formal model of the semantics of the domain of discourse which allows the GUIDE to follow the user's thought process as he sequentially formulates his request. The translation process of the GUIDE has the efficiency of a keyword matching scheme, but the understanding of the GUIDE is deeper because the keyword matching is driven by a semantic model of the domain of discourse.

4.3 Paraphrase of Request

In an interactive environment, effective feedback is an important component of a successful system. The method of paraphrasing a request is a distinctive feature of the GUIDE which deals with that issue. For most systems, the type of feedback available is "relevance feedback" (e.g., the SMART system [40]) whereby the system tells the user how it interpreted his requests by showing a sample of the information which was retrieved. The user then indicates what parts of the retrieved material were relevant and the system modifies its understanding of the request on that basis. In contrast, a semantic model of the domain of discourse allows the GUIDE to return to the user its understanding of his request directly.

4.4 Student Record and Course Outline

Most information retrieval systems do not maintain data which corresponds to the GUIDE's student record and course outline. This data is the basis for the GUIDE's "advising functions." Probably the

closest analogue to this data is the "student model" of generative CAI systems such as that of Koffman [49]. In a sense, the course outline is the GUIDE's model of an ideal student, while a student record is the model of an actual student. The reporting of discrepancies between these two models constitutes the GUIDE's advising function. Unfortunately, the GUIDE's student model (both ideal and actual) is somewhat weak. This is because the individual lessons of the lesson library do not systematically or consistently evaluate the performance of students. Without this type of data, an instructor cannot, of course, give detailed specifications of how a student should perform in a given lesson (i.e., with no means of describing an actual student in detail, the instructor has no model for describing an ideal student).

4.5 Aspects of the Implementation

One of the problems which was faced in the development of the GUIDE was the stringent environment in which it must function. The techniques which enabled the GUIDE to overcome this problem include the following:

(1) Decisions which are difficult or time consuming to automate were placed in the hands of the user. For example, ambiguous terms are clarified by asking the user to indicate his intended use of the ambiguous word or phrase; questions about any part of a lesson record, course record, or student record receive the entire record in response; the paraphrase expresses how the GUIDE has interpreted a request, and the user can either accept it as is or try to rephrase his request from scratch; a "term explosion" in the attempt to find useful keywords is controlled by the user as he selects the graphics displays he wants to see.

(2) The algorithms used in generating the graphics displays take advantage of simplistic techniques which do not require a great deal of computation.

(3) The English translator which was developed for the GUIDE [11] performs its task very rapidly.

(4) Space problems were solved by packing data as tightly as possible. Variable length records were used, and in most cases a field of information was just large enough to contain the maximum possible range of values. Time problems were solved by using essentially an inverted file for retrieving the lessons attached to a given keyword, and both space and time were saved by representing the concept space and lesson space graphs in the form of adjacency lists.

5. SUMMARY AND CONCLUSIONS

5.1 Summary

In Chapter 1 we described a unique educational environment which has been made possible by the PLATO CAI system and the Department of Computer Science ACSES project. This environment has created the need for an automated advisor and reference librarian to help guide students in the use of the lesson material which has been developed as a part of ACSES. The capabilities of the GUIDE system were highlighted, and the research problems posed by the GUIDE were discussed.

In Chapter 2 the design of the GUIDE system was presented. The major components of the database and the philosophy of its design were discussed in detail. In addition, the major processing components of the system were also described, and the basic interactive graphics capabilities of the system were shown.

Chapter 3 contained extensive examples of the GUIDE's performance, illustrating the power and flexibility of the English translator, and the ability of the paraphraser to express the intent of a student's request. An analysis of over 3000 requests received by the GUIDE was also presented.

Finally, Chapter 4 presented some of the key features of the GUIDE and discussed some of the relevant literature.

5.2 Suggestions for Further Work

It would be difficult to say that a system like the GUIDE is ever completely finished. There are constant changes, improvements, and adjustments which could be made.

One useful activity would be in the area of improving the concept space. The requests presented to the system could be monitored to determine areas of interest and then a special effort made to improve the structure of the concept space in those areas. If a serious audience could be found, it would perhaps be reasonable to let them modify the concept space, making additions and interconnections which they felt were appropriate (the mechanism for this was at one time present in the system, but had to be removed because too much "graffiti" were coming in and cluttering the system). This would make use of the principle discussed by Batty that when several people analyze a subject, certain identifiable networks of terms begin to emerge [63].

The translator could be extended to allow direct specification of graphics displays to be shown (e.g., "show me the prerequisites, sequels, and lessons with exercises for fortdo"). Also, additional syntactic forms which are constantly used but not currently allowed could be permitted.

The concept space could be enriched by adding definitions to the concept nodes, or perhaps the concept space could be used as the basis of an instructional lesson on the discipline of computer science. In other words, one could use the concept space as an end in itself in learning about computer science, rather than using it primarily as a tool for retrieving lessons.

Further extensions of the graphics capabilities could be explored. For example, one might add the ability to specify a neighborhood to be displayed around a given set of lessons or concepts instead of only a single given lesson or concept. Also, some work might be done on specifying the displays in advance (as is proposed by Brunnstein and Schmidt [52]) rather than generating the displays dynamically. This has two advantages: (1) it would take less time to plot the display, and (2) given nodes would always be shown in the same way relative to each other, rather than having no constant spatial relationship. However, there are two disadvantages: (1) it would take more space to store the concept space in memory, and (2) it would take quite a bit more time to enter data into the concept space.

The lesson space could become more "finely tuned" in its structure as the content of the lessons begins to settle into a fairly fixed state and the interrelationships between lessons become clear to those in charge of managing the library. Also, a refinement of the terms used as keywords for a given lesson would help improve the system's ability to retrieve the proper lessons.

Work could be done in utilizing the GUIDE in different subject areas. For example, one could use the GUIDE for retrieving PLATO lessons in other subject areas simply by changing the content of the database. (Work is already in progress toward adapting the GUIDE for use by the foreign languages PLATO projects.) Also, the GUIDE could be applied in different contexts. This past semester two students worked on a project to use the GUIDE as an aid for retrieving books from the

Department of Computer Science library. Basically, the only change really necessary to the GUIDE was to interpret a lesson record as a "book record." The students defined their own classification tree for the discipline of computer science, indexed several books under the appropriate keywords, and had a retrieval system with the natural language and graphics capabilities possessed by the GUIDE.

The advising capabilities of the GUIDE could be enriched in several ways. The instructional lessons could provide an evaluation of the student's performance--perhaps at least indicate whether all areas of the lesson have been studied. It might be reasonable to allow the student to evaluate his own performance; for example, by ranking his understanding of the subject of a lesson on a scale from 1 (don't understand at all) to 7 (understand well). In the course outline, one could provide some means for an instructor to specify "counseling actions"; for example, specify that in the third week of classes, students should be shown a progress report on their performance, in the fourth week, display a reminder of an upcoming exam, etc.

5.3 Conclusions

We have demonstrated that the GUIDE successfully fulfills the role of the advisor and automated librarian described in Chapter 1. By introduction of the concept space and lesson space, we have been able to greatly enhance the ability of the GUIDE to assist people in finding the appropriate lessons. We have successfully implemented the type of scheme postulated by Johnson wherein one is able to move freely in a polydimensional information space, pursuing vertical, horizontal,

and oblique relationships [64]. Already, students are learning to utilize the GUIDE effectively, and in the future the GUIDE should be a valuable asset in making the ACSES project useful to anyone having access to one of the many PLATO terminals scattered throughout the country.

LIST OF REFERENCES

[1] Alpert, D. and D. L. Bitzer, "Advances in Computer Based Education," Science 167 (1970), pp. 1582-1590.

[2] Stifle, Jack, "The Plato IV Student Terminal," University of Illinois, CERL Report X-15, January 1973.

[3] Stifle, Jack, "The Plato IV Terminal: Description of Operation," University of Illinois, CERL Report, March 1973.

[4] Stifle, Jack, "The Plato IV Architecture," University of Illinois, CERL Report X-20, May 1972.

[5] Nievergelt, J. and E. M. Reingold, "Automating Introductory Computer Science Courses," SIGCSE Bulletin 5, No. 1 (1973), pp. 24-25.

[6] Nievergelt, J., E. M. Reingold, and T. R. Wilcox, "The Automation of Introductory Computer Science Courses," SIGCUE Bulletin 8, No. 1 (1974), pp. 15-21.

[7] Wilcox, T. R., "The Interactive Compiler as a Consultant in the Computer Aided Instruction of Programming," Proceedings of the Seventh Annual Princeton Conference in Information Sciences and Systems, March 1973.

[8] Tindall, Michael H., "Table-driven Compiler Error Analysis," Ph.D. Thesis to be published June, 1975, Department of Computer Science, University of Illinois, Urbana, Illinois.

[9] Davis, Alan M., "An Interactive Analysis System for Execution-Time Errors," Ph.D. Dissertation, Technical Report UIUCDCS-R-75-695, University of Illinois, January 1975.

[10] Whitlock, Larry R., "A Generative Tailored Exam System," unpublished Ph.D. Thesis Proposal, Department of Computer Science, University of Illinois, Urbana, Illinois, April 1975.

[11] Pradels, Jean, "The Guide: An Information System," Ph.D. Dissertation, Technical Report UIUCDCS-R-74-626, University of Illinois, March 1974.

[12] Murai, Shinnichi, "Guide-0: An Experimental Information System," Master's Thesis, Technical Report UIUCDCS-R-73-587, University of Illinois, August 1973.

[13] CODASYL Systems Committee, A Survey of Generalized Data Base Management Systems, ACM, May 1969.

[14] CODASYL Data Base Task Group, Feature Analysis of Generalized Data Base Management Systems, ACM, New York, May 1971.

[15] CODASYL Data Base Task Group, April 1971 Report, ACM, New York, 1971.

[16] GUIDE-SHARE Data Base Requirement Group, Requirements for a Data Base Management System, November 1971, available from GUIDE or SHARE, New York.

[17] Proceedings of the ACM SIGFIDET Workshop on Data Description, Access, and Control, 1971.

[18] FDT Bulletin, Quarterly publication of the ACM SIGMOD.

[19] Borden, G. A. and W. F. Nelson, "Toward a Viable Classification Scheme: Some Theoretical Considerations," American Documentation 20, No. 4 (1969).

[20] Farradane, J. E. L., "The Psychology of Classification," Journal of Documentation 11, No. 4 (1955).

[21] Farradane, J. E. L., "Relational Indexing and Classification in the Light of Recent Experimental Work in Psychology," Information Storage and Retrieval 1, No. 1 (1963).

[22] Meredith, G. P., "The Analysis of Relations," Proceedings of Leeds Philosophical Society (Scientific Section) 5 (1949), pp. 269-278.

[23] Meredith, G. P., "The Formulation of Epistemic Relations," Proceedings of Leeds Philosophical Society (Scientific Section) 6 (1952), pp. 33-42.

[24] Meredith, G. P., "Semantic Matrices," International Conference on Scientific Information 5 (1958), p. 171.

[25] Shera, J. H., "Pattern, Structure, and Conceptualization in Classification," Proceedings of the International Study Conference on Classification for Information Retrieval, 1957.

[26] Farradane, J. E. L., "A Scientific Theory of Classification and Indexing," Journal of Documentation 6 (1950), pp. 83-99, and 8 (1952), pp. 73-92.

[27] Farradane, J. E. L., "Concept Organization for Information Retrieval," Proceedings of the International Symposium on Relational Factors in Classification in Information Storage and Retrieval 3, No. 4 (1967).

[28] Levy, F., "On the Relative Nature of Relational Factors in Classifications," Information Storage and Retrieval 3 (1966-67).

[29] Vickery, B. C., On Retrieval System Theory, London, Butterworths, 1965.

[30] Quillian, M. R., "Semantic Memory," in Martin Minsky (ed.) Semantic Information Processing, MIT Press, 1968.

[31] Collins, A. and M. R. Quillian, "Retrieval Time from Semantic Memory," Journal of Verbal Learning and Verbal Behavior 9 (1970).

[32] Simmons, R. F., "Semantic Networks: Their Computation and Use for Understanding English Sentences," in R. Schank and K. Colby (eds.) Computer Models of Thought and Language, W. H. Freeman and Company, San Francisco, 1973.

[33] Maguire, R. B., "Methods for Producing Visual Displays of Linear Graphs," Technical Report CS-73-29, University of Waterloo, Department of Computer Science, November 1973.

[34] Ranganathan, S. R., "Hidden Roots of Classification," Information Storage and Retrieval 3, No. 4 (1967).

[35] Bush, V., "As We May Think," Atlantic Monthly 176 (1945).

[36] Engelbart, Douglas, et al., "The Augmented Knowledge Workshop," AFIPS, 1973.

[37] Taube, M., "Unit Terms in Coordinate Indexing," American Documentation 3 (1952).

[38] Taube, M. and I. Wachtel, "The Logical Structure of Coordinate Indexing," American Documentation 4, No 2 (1953).

[39] Taube, M. et al., "Storage and Retrieval of Information by Means of the Association of Ideas," American Documentation 6 (January 1955), pp. 1-18.

[40] Salton, G. (ed.), The SMART Retrieval System, Prentice-Hall, 1971.

[41] Doyle, L., "Semantic Road Maps for Literature Searches," Journal of the ACM 8 (1961), p. 553.

[42] Doyle, L. B., Indexing and Abstracting by Association," American Documentation 13, No. 4 (1962).

[43] Borko, H., Interactive Displays for Document Retrieval, System Development Corp., May 1965.

[44] Treu, S., "The Browser's Retrieval Game," American Documentation 19 (October 1968), p. 404.

[45] Treu, S., "Supplementing Human Memory by Means of Interactive, Computer-Based Associative Storage and Retrieval," Ph.D. Dissertation, University of Pittsburgh, 1970.

[46] Treu, S., "A Conceptual Framework for the Searcher-System Interface," in D. E. Walker (ed.) Interactive Bibliographic Search: The User/Computer Interface, AFIPS Press, 1971.

[47] Carbonell, J., "Mixed-Initiative Man-Computer Instructional Dialogues," Bolt, Beranek and Newman, Inc., Report No. 1971, 1970.

[48] Wexler, J. D., "Information Networks in Generative Computer-Assisted Instruction," IEEE Transactions on Man-Machine Systems, MMS-11, 4 (1970).

[49] Koffman, E. B., "A Generative CAI Tutor for Computer Science Concepts," Spring Joint Computer Conference 1972.

[50] Osin, L., "A System to Produce CAI courses from Logically Structured Educational Material," Doctoral Dissertation, June 1974, Department of Computer Science, Israel Institute of Technology (Technion), Haifa.

[51] Lancaster, F. W., "On the Need for Role Indicators in Postcoordinate Retrieval Systems," American Documentation 19, No. 1 (1968).

[52] Brunnstein, Klaus, and Joachim Schmidt, "Structuring and Retrieving Information in Computer-Based Learning," International Journal of Computer and Information Sciences 2, No. 2 (1973).

[53] Soergel, D., Indexing Languages and Thesauri: Construction and Maintenance, Melville Publishing Co., Los Angeles, 1974.

[54] Thompson, D. A., L. A. Benningson, and D. Whitman, "A Proposed Structure for Displayed Information to Minimize Search Time Through a Data Base," American Documentation 19, No. 1 (1968).

[55] Thompson, David A., "Interactive Computer Graphics in Medical Information Retrieval and Classification," Computer Graphics 6, No. 4 (Winter 1972).

[56] Seastrom, Dale E. and David A. Thompson, "ADMIRE--A Study of an Adaptable Document Retrieval System with Assistive Displays," ASIS 10 (1973), Proceedings of the American Society for Information Science.

[57] Lyman, E., "An On-Line Document Retrieval Strategy Using the PLATO System," University of Illinois, CERL Report X-21, May 1971.

[58] Woods, W. A., "An Experimental Parsing System for Transition Network Grammars," in R. Rustin (ed.) Natural Language Processing, Algorithmics Press, Inc., New York, 1973.

[59] Winograd, Terry, "Procedures as a Representation for Data in a Computer Program for Understanding Natural Language," Al TR-17 MIT Artificial Intelligence Laboratory, February 1971.

[60] Schank, R., "Identification of Conceptualizations Underlying Natural Language," in Schank and Colby (eds.) Computer Models of Thought and Language, W. H. Freeman and Company, San Francisco, 1973.

[61] Chai, D. T., "An Information Retrieval System Using Keyword Dialog," Information Storage and Retrieval 9 (1973).

[62] Moyne, J., "Information Retrieval and Natural Language," American Society for Information Science 6 (1969).

[63] Batty, C. David, "The Instant Thesaurus: Synthesizing User's Concept Systems to Reduce Cost and Time in Index Language Development," American Society for Information Science Proceedings 10 (1973).

[64] Johnson, H. T., "A Polydimensional Scheme for Information Retrieval," American Documentation 13, No. 1 (1962).

APPENDIX A

STATE TABLE UTILIZED BY THE ENGLISH TRANSLATOR

STATE TABLE — NEXT STATE = OUTPUT = LEVEL OF CONFIDENCE

INPUT	STATE 0	1	2	3	4	5	6	7	8	9
0	0=0=3	1= 0=3	2= 0=3	3= 0=3	4= 0=3	5= 0=3	6= 0=3	7= 0=3	8= 0=3	9= 0=3
1	11=30=3 6=16=3	4=16=3	9=14=3 2=14=3	6=16=3	4=16=3	11= 0=3	6=16=3	11= 0=3	8=16=3	11= 0=3
2	11=30=3 5=15=3	5=15=3	2=26=2	7=15=3 3= 0=3	4= 0=3	5=15=3	6= 0=3	7=15=3	8=29=3	9=15=0
3	11=30=3 5=15=3	5=15=3 4=16=3	2= 0=2	7=15=3 6=16=3	4=16=3	5=15=3	6=16=3	7=15=3	8=16=3	9=15=0
4	11=30=3 5=15=3	11=31=3	2=26=2	11=31=3	4= 0=3	5=15=3	6= 0=3	7=15=3	8=29=3	9=15=0
5	5=15=3 0= 0=3 1= 0=3	5=15=3 1= 0=3	2= 0=1	7=15=3 3= 0=3	4= 0=3	5=15=3	6= 0=3	7=15=3	8= 0=3	9=15=3
6	0=15=3	5=15=3 1= 0=3	2=26=3	7=15=3 3= 0=3	4= 0=3	5=15=3	6= 0=3	7=15=3	8=29=3	9=15=3
7	5=15=3 0= 0=3 1= 0=3	5=15=3 1= 0=3	2=26=2	7=11=3 6=12=3	4= 0=3	5=15=3	6= 0=3	7=15=3	8=29=3	9=15=3
8	5=15=3 0= 0=3 1= 0=3	5=15=3 1= 0=3	2= 0=3	7=15=3 3= 0=3	4= 0=3	5=15=3	6= 0=3	7=15=3	8= 0=3	9=15=3
9	0=15=3	5=15=3 1= 0=3	2=26=2	7=15=3 3= 0=3	4= 0=3	5=15=3	6= 0=3	7=15=3	8=29=3	9=15=0
10	11=30=3	5=15=3 1= 0=3	2=26=2	7=15=3 3= 0=3	4= 0=3	5=15=3	6= 0=3	7=15=3	8=29=3	9=15=3
11	1= 4=3 0= 0=3	1= 0=3		3= 0=3	4= 0=3	5=14=3	6= 0=3	7=14=3		
12	1= 5=3 3= 6=3	1= 0=3		3= 0=3	4= 0=3	5=15=3	6= 0=3	7=15=3		
13	3= 6=3	8= 0=3	8= 0=3	8= 0=3	8= 0=3	8= 0=3				
14	1= 2=3 3= 2=3	11= 0=3	11= 0=3	11= 0=3	11= 0=3	11= 0=3	6=30=3	7=30=3		11= 0=3
15	1= 3=3 3= 3=3 0= 0=3	5=17=3	2=29=3	7=15=3	4=29=3	5=17=3	6=29=3	7=17=3	8= 0=3	9=17=3
16	1= 4=3 3= 4=3 0= 0=3	1=29=3	2=29=3	3=29=3	4= 0=3	5=18=3	6= 0=3	7=18=3	8=29=3	9=18=3
17	1= 5=3 3= 6=3 0= 0=3	1=29=3	2=29=3	3=29=3	4= 0=3	5=19=3	6= 0=3	7=19=3	8=29=3	9=19=3
18	3= 6=3 0= 0=3	5=20=3 1= 0=3	9=10=2 8=10=2	7=20=3	4=29=3	5=20=3	6=29=3	7=20=3	8=10=3	9=20=3
19	1= 7=3 3= 8=3	5=21=3 1= 0=3	9= 9=2 8= 9=2	7=21=3	4=29=3	5=21=3	6=29=3	7=21=3	8= 9=3	9=21=3
20	1= 9=3 0= 0=3	5= 0=3 4= 0=3	2= 0=3	3= 0=3		5=22=3	6=29=3	7=22=3		
21	1=10=3 0= 0=3				4=29=3	5=29=3	6=29=3	7=29=3	8=29=3	9=29=3
22	0= 0=3 5= 0=3 1= 2=3	1= 9=3	9= 9=2 8= 9=2	3= 9=3	4=30=3	5=30=3	6=30=3	7=30=3		
23	0= 0=3 1= 0=3	1=10=3	9=10=2 8=10=2	3=10=3		5=23=3		7=23=3		9=23=3
24	9= 1=3 2=11=3 0= 0=3		2=26=3						8=26=3	9=26=3
25	0=30=3									
26	0=31=3								8=26=3	
27	1=12=3 0= 0=3	1=12=3	9=12=2 8=12=2 2=12=2	3=12=3						
28	0=29=3									
29	1=13=3 0=13=3	1=13=3	9=13=2 8=13=2 2=13=2	0= 0=0	0= 0=0	0= 0=0	7= 115	0= 0=0	0= 0=0	6= 1=1
30										
31	0=25=3	1=25=3	2=25=3	3=25=3	4=25=3	5=25=3	6=25=3	7=25=3	8=25=3	9=25=3

APPENDIX B

TERM INPUT CLASSES

lesson = 1, concept = 2, course 3, author = 4

INDX	STATE DEPENDENT INPUT CLASS 0	1	2	3	4	5	6	7	8	9	WORD
0	23	0	0	0	0	0	0	0	0	0	ABOUT
0	14	16	14	16	21	21	21	21	21	21	ABSTRACT
4	29	29	29	29	29	29	29	29	29	29	ADMINISTRATEE
4	5	5	5	5	5	5	5	5	5	5	ADMINISTRATION
3	29	29	29	29	29	29	29	29	29	29	ADMINISTRATOR
3	6	6	6	6	6	6	6	6	6	6	ADVANCED
0	31	31	31	31	31	31	31	31	31	31	ADVICE
0	21	18	18	18	18	18	18	18	18	18	AFTER
0	31	31	31	31	31	31	31	31	31	31	AFTERNOON
0	26	16	16	16	16	16	16	16	16	16	AND
0	24	0	0	0	0	0	0	0	0	0	ANY
4	10	10	10	10	10	10	10	10	10	10	APRIL
0	24	0	0	0	0	0	0	0	0	0	ARE
8	10	10	10	10	10	10	10	10	10	10	AUGUST
0	15	14	14	14	21	0	21	0	21	0	AUTHOR
0	31	31	31	31	31	31	31	31	31	31	AVERAGE
12	29	29	29	29	29	29	29	29	29	29	BACKGROUND FOR
0	31	31	31	31	31	31	31	31	31	31	BAD
0	20	19	19	19	19	19	19	19	19	19	BEFORE
1	6	6	6	6	6	6	6	6	6	6	BEGINING
1	6	6	6	6	6	6	6	6	6	6	BEGINNING
0	31	31	31	31	31	31	31	31	31	31	BEST
0	0	0	0	0	0	0	0	0	0	0	BETWEEN
0	0	0	0	0	0	20	0	20	0	20	BUT
0	23	0	20	0	0	0	0	0	0	0	BY
0	24	0	0	0	0	0	0	0	0	0	CAN
3	5	5	5	5	5	5	5	5	5	5	COMMUNICATION
0	5	5	5	5	5	5	5	5	5	5	COURSE
0	31	31	31	31	31	31	31	31	31	31	DAY
12	10	10	10	10	10	10	10	10	10	10	DECEMBER
0	14	16	14	16	21	21	21	21	21	21	DESCRIPTION
0	24	14	0	0	0	14	0	0	0	0	DID
31	24	14	0	0	0	14	0	0	0	0	DID I
0	31	31	31	31	31	31	31	31	31	31	DIFFERENT
7	6	6	6	6	6	6	6	6	6	6	DIFFICULT
0	24	0	0	0	0	0	0	0	0	0	DO
0	24	0	0	0	0	0	0	0	0	0	DOES
0	24	0	0	0	0	0	0	0	0	0	DO YOU HAVE
1	6	6	6	6	6	6	6	6	6	6	EASY
0	31	31	31	31	31	31	31	31	31	31	EIGHT
1	6	6	6	6	6	6	6	6	6	6	ELEMENTARY
6	5	5	5	5	5	5	5	5	5	5	ENTRY
0	31	31	31	31	31	31	31	31	31	31	EVENING
2	5	5	5	5	5	5	5	5	5	5	EXAM
0	31	31	31	31	31	31	31	31	31	31	EXCELLENT
0	0	0	0	0	0	20	0	20	0	20	EXCEPT
2	7	7	7	7	7	7	7	7	7	7	EXERCISE
11	29	29	29	29	29	29	29	29	29	29	EXERCISE FOR
2	10	10	10	10	10	10	10	10	10	10	FEBRUARY
0	31	31	31	31	31	31	31	31	31	31	FIFTEEN
0	31	31	31	31	31	31	31	31	31	31	FIFTH
0	31	31	31	31	31	31	31	31	31	31	FIFTY
0	8	8	22	8	22	22	22	22	22	22	FINISH
1	9	9	9	9	9	9	9	9	9	9	FIRST
1	9	9	9	9	9	9	9	9	9	9	FIRST TIME
0	31	31	31	31	31	31	31	31	31	31	FIVE
0	21	18	18	18	18	18	18	18	18	18	FOLLOW
0	31	31	31	31	31	31	31	31	31	31	FOUR

INDX	STATE DEPENDENT INPUT CLASS 0	1	2	3	4	5	6	7	8	9	WORD
17	10	10	10	10	10	10	10	10	10	10	FRIDAY
0	21	18	18	18	18	18	18	18	18	18	FROM
0	0	0	0	0	0	0	0	0	0	0	GET
0	23	0	0	0	0	0	0	0	0	0	GIVE
31	8	8	8	8	8	8	8	8	8	8	GONE THROUGH
0	31	31	31	31	31	31	31	31	31	31	GOOD
31	8	14	8	0	14	14	0	0	0	0	GOT
0	8	8	8	8	8	8	8	8	8	8	GO THROUGH
0	17	14	14	18	21	21	21	21	21	21	GRADE
0	31	31	31	31	31	31	31	31	31	31	GUIDE
7	6	6	6	6	6	6	6	6	6	6	HARD
31	24	0	0	0	0	0	0	0	0	0	HAS
0	24	0	0	0	0	0	0	0	0	0	HAVE
0	19	14	14	14	21	21	21	21	21	21	HOUR
0	0	0	0	0	0	0	0	0	0	0	HOW ARE
0	28	16	16	16	21	21	21	21	21	21	HOW BAD
0	26	16	16	16	21	21	21	21	21	21	HOW CAN
31	18	14	14	16	21	21	21	21	21	21	HOW DID I
0	26	16	16	16	21	21	21	21	21	21	HOW DO
0	28	16	16	16	21	21	21	21	21	21	HOW GOOD
0	26	16	16	16	21	21	21	21	21	21	HOW HAVE
0	0	0	0	0	0	0	0	0	0	0	HOW IS
0	19	16	14	16	21	21	21	21	21	21	HOW LONG
0	16	16	14	16	21	21	21	21	21	21	HOW MANY
0	28	16	16	16	21	21	21	21	21	21	HOW WELL
0	31	31	31	31	31	31	31	31	31	31	IMPROVE
0	23	0	0	0	0	0	0	0	0	0	IN
1	5	5	5	5	5	5	5	5	5	5	INSTRUCTIONAL
0	31	31	31	31	31	31	31	31	31	31	INTERESTING
2	6	6	6	6	6	6	6	6	6	6	INTERMEDIATE
1	6	6	6	6	6	6	6	6	6	6	INTRODUCTORY
0	24	0	0	0	0	0	0	0	0	0	IS
0	0	0	0	0	0	0	0	0	0	0	I CAN
31	8	8	0	8	8	8	8	8	8	8	I DID
0	8	8	8	8	8	8	8	8	8	8	I DO
0	0	0	0	0	0	0	0	0	0	0	I HAVE
0	0	0	0	0	0	0	0	0	0	0	I LIKE
1	10	10	10	10	10	10	10	10	10	10	JANUARY
7	10	10	10	10	10	10	10	10	10	10	JULY
6	10	10	10	10	10	10	10	10	10	10	JUNE
2	9	9	9	9	9	9	9	9	9	9	LAST
2	9	9	9	9	9	9	9	9	9	9	LAST TIME
0	8	8	8	8	8	8	8	8	8	8	LEARN
31	8	8	8	8	8	8	8	8	8	8	LEARNED
0	5	5	5	5	21	21	21	21	21	21	LESSON
0	17	14	14	18	21	21	21	21	21	21	LEVEL
0	27	27	27	27	21	14	21	21	21	14	LIKE
0	23	0	0	0	0	0	0	0	0	0	LIST
3	10	10	10	10	10	10	10	10	10	10	MARCH
5	24	10	10	10	10	10	10	10	10	10	MAY
6	29	29	29	29	29	29	29	29	29	29	MEMBER
13	10	10	10	10	10	10	10	10	10	10	MONDAY
0	31	31	31	31	31	31	31	31	31	31	MONTH
7	29	29	29	29	29	29	29	29	29	29	MORE GENERAL
8	29	29	29	29	29	29	29	29	29	29	MORE SPECIFIC
0	31	31	31	31	31	31	31	31	31	31	MORNING
0	18	16	14	16	21	21	21	21	21	21	MY GRADE

INDX	STATE DEPENDENT INPUT CLASS 0	1	2	3	4	5	6	7	8	9	WORD
0	18	16	14	16	21	21	21	21	21	21	MY LEVEL
0	18	16	14	16	21	21	21	21	21	21	MY PERFORMANCE
0	18	16	14	16	21	21	21	21	21	21	MY STANDING
0	23	0	0	0	0	0	0	0	0	0	NEED
3	9	9	9	9	9	9	9	9	9	9	NEXT
0	21	18	18	18	18	18	18	18	18	18	NEXT AFTER
0	21	18	18	18	18	18	18	18	18	18	NEXT LESSON AFTER
3	9	9	9	9	9	9	9	9	9	9	NEXT TIME
0	31	31	31	31	31	31	31	31	31	31	NIGHT
0	31	31	31	31	31	31	31	31	31	31	NINE
0	26	15	15	15	15	15	15	15	15	15	NO
0	26	15	15	15	15	15	15	15	15	15	NOT
11	10	10	10	10	10	10	10	10	10	10	NOVEMBER
20	10	10	10	10	10	10	10	10	10	10	NOW
0	16	16	14	16	21	21	21	21	21	21	NUMBER
31	8	14	8	0	14	14	0	0	21	14	OBTAINED
10	10	10	10	10	10	10	10	10	10	10	OCTOBER
9	29	29	29	29	29	29	29	29	29	29	OF INTEREST
10	29	29	29	29	29	29	29	29	29	29	OF INTEREST FROM
0	23	0	20	0	0	0	0	0	0	0	ON
0	31	31	31	31	31	31	31	31	31	31	ONCE
0	31	31	31	31	31	31	31	31	31	31	ONE
0	26	17	17	17	17	17	17	17	17	17	OR
0	26	15	15	15	15	15	15	15	15	15	OTHER THAN
0	14	16	14	16	21	21	21	21	21	21	OUTLINE
0	0	0	0	0	0	0	0	0	0	0	OUTPUT
0	31	31	31	31	31	31	31	31	31	31	OUTSTANDING
5	29	29	29	29	29	29	29	29	29	29	OWNER
0	17	14	14	18	21	21	21	21	21	21	PERFORMANCE
6	7	7	7	7	7	7	7	7	7	7	PRACTICE
4	7	7	7	7	7	7	7	7	7	7	PRACTICING
0	20	19	19	19	19	19	19	19	19	19	PRECEDING
1	20	22	22	0	21	21	21	21	19	14	PREREQUISITE
0	20	19	19	19	19	19	19	19	19	19	PREVIOUS
2	5	5	5	5	5	5	5	5	5	5	QUIZ
0	31	31	31	31	31	31	31	31	31	31	REASON
31	8	14	8	0	14	14	0	0	8	8	RECEIVED
0	27	27	27	27	21	14	21	21	21	14	SAME
18	10	10	10	10	10	10	10	10	10	10	SATURDAY
0	31	31	31	31	31	31	31	31	31	31	SECOND
31	8	8	8	8	8	8	8	8	8	8	SEEN
0	31	31	31	31	31	31	31	31	31	31	SEMESTER
9	10	10	10	10	10	10	10	10	10	10	SEPTEMBER
2	21	23	23	23	21	21	21	21	18	14	SEQUEL
0	31	31	31	31	31	31	31	31	31	31	SEVEN
0	24	0	0	0	0	0	0	0	0	0	SHOULD
0	23	0	0	0	0	0	0	0	0	0	SHOW
0	27	27	27	27	21	14	21	21	21	14	SIMILAR
1	6	6	6	6	6	6	6	6	6	6	SIMPLE
0	21	18	18	18	18	18	18	18	18	18	SINCE
0	31	31	31	31	31	31	31	31	31	31	SIX
31	0	14	0	0	0	0	0	0	0	0	SPENT
0	17	14	14	18	21	21	21	21	21	21	STANDING
0	17	14	14	18	21	21	21	21	21	21	STUDENT RECORD
31	8	8	8	8	8	8	8	8	8	8	STUDIED
0	8	8	8	8	8	8	8	8	8	8	STUDY
0	14	16	14	16	21	21	21	21	21	21	SUBJECT

INDX	STATE DEPENDENT INPUT CLASS 0	1	2	3	4	5	6	7	8	9	WORD
0	21	18	18	18	18	18	18	18	18	18	SUCCEEDING
0	14	16	14	16	21	21	21	21	21	21	SUMMARY
19	10	10	10	10	10	10	10	10	10	10	SUNDAY
0	8	8	8	8	8	8	8	8	8	8	TAKE
31	8	14	8	8	8	8	8	8	8	8	TAKEN
0	23	0	0	0	0	0	0	0	0	0	TEACH
0	23	0	0	0	0	0	0	0	0	0	TELL
0	22	0	0	0	0	0	0	0	24	24	TELL ME ABOUT
0	31	31	31	31	31	31	31	31	31	31	TEN
2	5	5	5	5	5	5	5	5	5	5	TEST
0	31	31	31	31	31	31	31	31	31	31	THESE
0	31	31	31	31	31	31	31	31	31	31	THIRD
0	31	31	31	31	31	31	31	31	31	31	THIRTY
0	31	31	31	31	31	31	31	31	31	31	THIS
0	31	31	31	31	31	31	31	31	31	31	THOSE
0	31	31	31	31	31	31	31	31	31	31	THREE
14	10	10	10	10	10	10	10	10	10	10	THURSDAY
0	19	16	14	16	21	21	21	21	21	21	TIME
0	0	19	19	19	19	19	19	19	0	19	TO
21	10	10	10	10	10	10	10	10	10	10	TODAY
23	10	10	10	10	10	10	10	10	10	10	TOMORROW
21	10	10	10	10	10	10	10	10	10	10	TONIGHT
0	31	31	31	31	31	31	31	31	31	31	TOO
31	8	14	8	8	8	8	8	8	8	8	TOOK
0	14	16	14	16	21	21	21	21	21	21	TOPIC
4	7	7	7	7	7	7	7	7	7	7	TO PROGRAM
14	10	10	10	10	10	10	10	10	10	10	TUESDAY
0	31	31	31	31	31	31	31	31	31	31	TWENTY
0	31	31	31	31	31	31	31	31	31	31	TWICE
0	31	31	31	31	31	31	31	31	31	31	TWO
0	20	19	19	19	19	19	19	19	19	19	UNTIL
0	23	0	0	0	0	0	0	0	0	0	WANT
0	24	14	0	0	0	14	0	0	0	0	WAS
19	10	10	10	10	10	10	10	10	10	10	WEDNESDAY
0	31	31	31	31	31	31	31	31	31	31	WELL
31	8	8	8	8	8	8	8	8	8	8	WENT THROUGH
0	22	0	0	0	0	0	0	0	24	24	WHAT
0	22	0	0	0	0	0	0	0	24	24	WHAT ARE
0	22	0	0	0	0	0	0	0	24	24	WHAT CAN
31	0	0	2	0	0	0	0	0	24	24	WHAT DID
0	22	0	0	0	0	0	0	0	24	24	WHAT DO
0	28	16	14	16	21	21	21	21	21	21	WHAT ELSE
0	22	0	0	0	0	0	0	0	24	24	WHAT HAVE
0	22	0	0	0	0	0	0	0	24	24	WHAT IS
0	22	0	0	0	0	0	0	0	24	24	WHAT SHOULD
0	19	16	14	16	21	21	21	21	21	21	WHEN
0	22	0	0	0	0	0	0	0	24	24	WHERE DO
0	15	14	14	14	21	21	21	21	21	21	WHO
0	31	31	31	31	31	31	31	31	31	31	WHY
0	26	15	15	15	15	15	15	15	15	15	WITHOUT
0	31	31	31	31	31	31	31	31	31	31	WORSE
0	31	31	31	31	31	31	31	31	31	31	WORST
0	0	0	0	0	0	0	0	0	0	0	WOULD LIKE
0	31	31	31	31	31	31	31	31	31	31	YEAR
22	10	10	10	10	10	10	10	10	10	10	YESTERDAY
0	0	0	0	0	0	0	0	0	24	24	YOU ARE
0	0	0	0	0	0	0	0	0	24	24	YOU CAN
0	0	0	0	0	0	0	0	0	24	24	YOU HAVE

APPENDIX C

STATISTICS ON THE GUIDE DATABASE

	# records	# memory words	# words/record
Lessons.....	152	1412	9.3
Concepts.....	740	2820	3.8
Courses.....	8	99	12.4
Authors.....	54	125	2.3
Grammatical words.....	230	717	3.1
TOTAL	1184	5173	4.4

Lesson interconnections:

keywords	links to other lessons
0.....12	0.....40
1.....12	1.....40
2.....36	2.....20
3.....31	3.....14
4.....15	4.....11
5.....27	5..... 7
6.....14	6.....11
7..... 6	7..... 3
8..... 2	8..... 1
9..... 2	9..... 2
	11..... 1
3.3 keywords/ lesson	16..... 1
	17..... 1
	2.5 links/lesson

Concept interconnections:

lessons	links to other concepts
0.....616	0.....499
1..... 43	1.....176
2..... 18	2..... 13
3..... 13	3..... 15
4..... 10	4..... 10
5..... 6	5..... 6
6..... 7	6..... 10
7..... 7	7..... 4
8..... 5	8..... 1
9..... 3	9..... 1
10..... 4	10..... 2
11..... 2	11..... 2
12..... 1	18..... 1
14..... 1	
16..... 1	0.63 links/concept
17..... 1	
19..... 1	
21..... 1	

0.69 lessons/keyword

VITA

The author, Dave Ronald Eland, was born in Denver Colorado, on February 17, 1947. He graduated from Oral Roberts University with a Bachelor of Arts degree in mathematics in May 1969. In July of 1971, he received a Master of Science degree in physics from the University of Tulsa. Since September, 1972, he has been a research assistant in the Department of Computer Science of the University of Illinois. He is a member of the Association for Computing Machinery.