Computers & Geosciences Vol. 16, No. 6, pp. 833–846, 1990
Printed in Great Britain.

0098-3004/90 $3.00 + 0.00

SEDRULE: A RULE-BASED SYSTEM FOR INTERPRETING SOME MAJOR SEDIMENTARY ENVIRONMENTS

MICHAEL N. DEMERS
Department of Geography, The Ohio State University, Columbus, OH 43210, U.S.A.

(*Received* 6 *February* 1990)

Abstract—SEDRULE is a simple expert system for identification of major sedimentary environments from good outcrops. Although rudimentary, the program, written in muLISP, illustrates the forward-chaining search method. Initially designed as a teaching aid for the novice field sedimentologist, it also can be used to teach some basic principles of LISP programming within sedimentology. Furthermore, this program helps analyze the basic tenets of interpreting depositional environments, thereby helping to examine expert reasoning, which is a goal of AI research. Finally, the program expands its knowledge base as the student's own abilities grow, so it provides a lifelong tool for field work.

Key Words: Expert systems, LISP, Sedimentary environments.

INTRODUCTION

Many fields of geological inquiry do not readily lend themselves to numeric or algorithmic analysis. Instead, much of the knowledge acquired by the practicing geologist is conceptual, an integrated understanding of complex physical processes which combine to explain phenomena observed in the field or laboratory. Paramount among such approaches is the experimental acquisition and storage of classification rules of sedimentary environments. Such knowledge must be transfered piecemeal to the student through continued field experiences, frequently resulting in informational hiatuses because of lack of available environmental settings. A mechanism is needed to catalog such expertise, to deliver adequate classifications from it, to correct erroneous classifications and to learn from the user as experience dictates.

SEDRULE is one of a new generation of nonprocedural computer programs known as expert systems which provide such a mechanism, delivering solutions through symbolic logic rather than numeric data processing. Such declarative programs store facts and rules within a specified knowledge domain by employing user-supplied information, and applying built-in logic to select a reasonable solution to a query (Hayes-Roth, Waterman, and Lenat, 1983). Additionally, the expert's knowledge can be augmented by interactively adding to the knowledge base during the query process, thereby enhancing the utility of the system to subsequent users.

The geological domains have been fertile ground for many expert consultation programs. One early successful consultative expert systems was PROSPECTOR (Duda, Gaschnig, and Hart, 1979), which assists exploration geologists in locating areas of potential mineralization. Claims of system success have been reported widely (Campbell and others, 1982), and a microcomputer-based version subsequently developed (McCammon, 1986). Other geologically oriented knowledge based systems include DIPMETER (Davis and others, 1981) and LITHO (Bonnet and Dahan, 1983), assessment tools for oil well-log interpretation, muPETROL (Miller, 1986), an expert system for the classification of oil-bearing capability of sedimentary basins, EXPAL (Conrad and Beightol, 1988), an expert system for the identification and management in large paleontological databases, and MICA (Hart, McQueen, and Newmarch, 1988), a mineral identification advisor for geology students. Additionally, nongeology related knowledge-based systems for the earth sciences are summarized by Fisher and others (1988) and an expert system for soil taxonomy (STAX) has been produced by Fisher and Balachandran (1989). Such examples give ample evidence of the abundance of knowledge-rich, algorithmically poor domains within the geosciences that are well suited to application of nonprocedural programming techniques.

Most of the geological applications, with the exception of MICA, are designed for the practicing geologist rather than the student. As such they may be large, requiring mainframe computers or microcomputers with large harddisk drives. This limitation precludes their use in the field. The advent of small yet powerful laptop computers allows the power of such systems to be brought into the field. Similar applications of portable microcomputers have been performed successfully in soil science (Fisher and others, 1987), however no attempt to implement expert systems in field analysis of sedimentary environments has as yet, been performed, particularly as a pedagogical tool.

KNOWLEDGE BASE

Expert systems require the formalization of a rule base. As Pettijohn (1975) points out, a classification scheme represents, in shorthand form, an attempt to organize the knowledge of an expert about a particular subject. Thus, a classification of clastic sediments should embody an understanding of the physical properties and processes by which they are defined. This body of knowledge, or knowledge base, is the distillation of experience and accumulated learning of an expert or group of experts within the fields of sedimentology, sedimentary petrology, and geomorphology.

For pedagogical purposes the body of knowledge need not be complete, but it should be self-explanatory. Clayton (1973, 1988) developed printed classification systems for use by students of geomorphology in field identification exercises. His classifications merge both descriptive and genetic concepts regarding clastic sediments and frequently uses nomenclature which is itself more process oriented. He integrates his knowledge of sedimentary textures, sedimentary structures, hydraulic regimine, and depositional environment into a tree-based classification not unlike that used in the taxonomy of organisms. Its design rquires that the user describe fundamental physical properties of a sediment in the field and to recognize processes and relationships among depositional environments. In addition, the strong hierarchical structure of Clayton's classification makes it ideal for incorporation into a declarative knowledge-based system such as SEDRULE.

Clayton's original (1973) version of the knowledge base included 26 sediment types (Table 1), whereas his newer (1988) knowledge base includes 15 types (Table 2), which are more generic, more globally descriptive of the depositional environment, and more closely akin to the needs of the entering student. SEDRULE could be designed to use either approach, but the examples in SEDRULE center on the former, more detailed classification in which classificatory redundancy is minimized. This more explicit classification also permits more adaptability to expansion as the student's experience and knowledge grow.

The general categories presented in Clayton's (1988) classification can look extremely different in the field, depending primarily on the flow regime, but the classification scheme provides a variety of search methods by which each could be identified. Although this makes the system redundant, it also allows the student to recognize the complex relationship

Table 1. List of sediment types included in Clayton's (1973) classification system

Sediments lacking bedding:

HIGHLY BURROWED DEPOSIT OR HILLSLOPE DEPOSIT
GLACIAL DEPOSIT
PLASTIC FLOW SEDIMENT
FLUVIAL OVERBANK SEDIMENT
HIGHLY BURROWED NEARSHORE SEDIMENT

Sediment with bedding but lacking crossbedding:

DISCONTINUOUS TURBIDITY CURRENT SUSPENSION LOAD SEDIMENT
OFFSHORE NON-BOTTOM CURRENT SUSPENSION LOAD SEDIMENT
CONTINUOUS TURBIDITY CURRENT SUSPENSION LOAD SEDIMENT
FLUVIAL OVERBANK SEDIMENT
VOLCANIC ASH OR LOESS

Sediment with low angle crossbedding:

DEEP NEARSHORE RIPPLE SEDIMENT
SHALLOW NEARSHORE RIPPLE SEDIMENT
SLOPEWASH SEDIMENT
UPPER FLOW REGIME FLUVIAL SEDIMENT
LOWER FLOW REGIME TRACTION LOAD OR UPPER FLOW REGIME FLUVIAL SEDIMENT
SHALLOW NEARSHORE BAR SEDIMENT
TOPBEACH SEDIMENT
FOREBEACH SEDIMENT
EOLIAN TRACTION LOAD SEDIMENT

Sediment with high angle crossbedding:

DEEP NEARSHORE RIPPLE SEDIMENT
SHALLOW NEARSHORE RIPPLE SEDIMENT
FLUVIAL RIPPLE SLIPFACE SEDIMENT
TALUS
DELTA SLIPFACE SEDIMENT
BACKBEACH SEDIMENT
FOREBEACH SEDIMENT
SHALLOW NEARSHORE BAR SEDIMENT
EOLIAN DUNE SLIPFACE SEDIMENT
OCEAN CURRENT DUNE SLIPFACE SEDIMENT

Table 2. List of sediment types included in Clayton's (1988) classification system

```
Sediments lacking bedding:

     GLACIAL SEDIMENT
     MASS MOVEMENT SEDIMENT

Sediments with plane bedding:

     A. Gravel:
          FLUVIAL CHANNEL SEDIMENT
          TOPBEACH SEDIMENT
          FOREBEACH SEDIMENT
          CONTINUOUS BOTTOM CURRENT SEDIMENT

     B. Sand:
          EOLIAN SEDIMENT
          FOREBEACH SEDIMENT
          TOPBEACH SEDIMENT
          SLOPEWASH SEDIMENT
          FLUVIAL CHANNEL SEDIMENT
          CONTINUOUS BOTTOM CURRENT SEDIMENT
          DISCONTINUOUS BOTTOM CURRENT SEDIMENT

     C. Silt or clay:
          FLUVIAL OVERBANK SEDIMENT
          SLOPEWASH SEDIMENT
          EOLIAN SEDIMENT
          OFFSHORE SEDIMENT

Crossbedded sediments:

     A. Solitary sand or gravel:
          BACKBEACH SEDIMENT
          DELTA FORESET SEDIMENT
          TALUS

     B. Sand, grouped small scale crossbedding:
          FLUVIAL CHANNEL SEDIMENT
          FLUVIAL OVERBANK SEDIMENT
          TOPBEACH SEDIMENT
          NEARBEACH SEDIMENT
          DISCONTINUOUS BOTTOM CURRENT SEDIMENT
          CONTINUOUS BOTTOM CURRENT SEDIMENT

     C. Sand, grouped large scale crossbedding:
          EOLIAN SEDIMENT
          FLUVIAL CHANNEL SEDIMENT
          DISCONTINUOUS BOTTOM CURRENT SEDIMENT
          CONTINUOUS BOTTOM CURRENT SEDIMENT
```

between descriptive and genetic classifications. It is suggested that the reader contact Clayton directly and request his document and then use it as an example knowledge-base development project (see "Acknowledgment").

PROGRAMMING CONSIDERATIONS

SEDRULE is written in muLISP-86 (The Soft Warehouse, 1986), a dialect of the LISP language available for either PC or generic MS-DOS machines. This particular version of muLISP is postdated by muLISP-87 (The Soft Warehouse, 1987) which also interacts with a separate compiler for delivery of the final product. The program is not compiled here, nor does it employ any graphics, thus allowing SEDRULE to run on any MS-DOS machine. Although at $300 muLISP is more expensive than languages such as Turbo-Prolog, it allows greater flexibility for development of dynamic rule bases because both rules and facts can be changed easily.

The term LISP stands for LISt Processing and accurately describes the strength of the language to manage textual information. As a symbol processing language LISP is probably the most used. A number of arguments for this have been put forward, but perhaps the most important is that LISP is easy to learn (Winston and Horn, 1981). In fact, for many who have difficulty with procedural languages such as FORTRAN or Pascal, LISP may be the ideal language, at least for nonprocedural problems. Its simplicity and the uniformity between function definitions and data make it easy to begin programming without concern for data declaration statements.

Among LISP's many strengths are its ability to grow as a language, a trait now present within the procedural language C. Once a function has been defined, it is simple to transfer that same function to another program. Therefore, as the programmer's

knowledge of LISP programming becomes more refined and the problem domains become more difficult, the language can grow to keep pace.

Based on this premise, SEDRULE is a simple program with easy to understand functions. The program is annotated heavily for the user so that innovations and additions can be made, and functional capabilities can be added. Additionally, I have included a section on creating your own database so that using the program in its existing form is possible within any number of classification domains.

As with all AI programming languages there is a two-part nature to LISP programs, the knowledge base and the inference engine. The knowledge base used in this program is based on production rules, which are a set of questions and their associated answers from which classification problems can be solved easily. The knowledge base therefore is a recipe for classification based on the known relationships between these questions and answers. An inference engine is a set of procedures which are designed to trigger the proper questions in the appropriate order to achieve the desired result.

INFERENCE ENGINE

The development of a set of production rules is only an initial step in creating an expert system. It also is necessary to develop the organizational procedure to trigger the rules at the proper time. Such an organization (inference engine) can be based on a wide variety of available search procedures.

SEDRULE uses one of the simplest search strategies, known as forward chaining. In this search strategy the computer works from a set of facts to a possible conclusion. The program tries all available rules, eliminating those which do not apply and adds those which do. In essence the program looks for rules based on already known facts. In SEDRULE these facts are provided by the user from yes or no responses to questions.

The search strategy is a decision tree based on Clayton's (1973) classification. The search through the decision tree is data-driven, that is, the solution is carried out in a forward manner via the user's responses. Based on the decision tree, all positive responses to a query result in a movement along the positive response branches of the tree. A negative response prompts a movement along the opposing branches. The advantage of such a search methodology lies in its ability to rapidly eliminate large portions of the search-space. For example, the first query is always ... "Does the sediment show clear evidence of bedding?" A positive response effectively eliminates approximately 25% of the search space, whereas a negative response eliminates approximately 75%. Two consecutive positive responses eliminates approximately 40% of the search space, and a positive followed by a negative response eliminates about 80%, and so on. The search-space is a measure of the number of questions and answers which remain to be answered.

The sediment types are organized into three major groups: unbedded, parallel bedded, and crossbedded. In order to maintain the integrity of the rule-base structure, a sediment under consideration must be relegated to one group or another within the first two query responses. Thus, unbedded sediments are identified with one initial negative response, parallel-bedded sediments are grouped after one positive response and one negative response, and crossbedded sediments are grouped after two positive responses.

The ordering of the queries within the rule base is critical to the selection of proper sediment types. They are arranged in a specific hierarchy by group, level, and lateral position. For example, the group designating unbedded sediments is situated first in the rule base because fewer responses are needed to narrow the search space within which a proper answer can be derived. Within each group, sediment types which are situated at the higher levels are ordered next, and within each level, the sediment types also are prioritized. Details on how to organize such a rule base are given in a later section of this paper.

PROGRAM IMPLEMENTATION AND ANATOMY

The program is composed of twelve functions which provide most of its operative capabilities (Table 3). These functions are created by using the LISP command "DEFUN" which stands for define function and allows for the recursive definition of the function's attributes and parameters. Paramount among these functions are the following eight which are designed to create, test, manipulate, and write the knowledge base as it is used.

CONSULT is the first function encountered and provides an opening menu, requests a user name and a consultation identification name (used for later file storage and retrieval), and triggers appropriate questions determined by the user's response. MEMBER-BASE then checks to see if the consultation name exists. If it does, this indicates that a previous consultation had taken place and a database had been saved under that name. CONSULT then asks the user whether they want to use this knowledge base and subsequently calls the appropriate data sources supplied by the next function, SEDIMENT, which is designed to allow the user to select among the knowledge bases *INIT-BASE*, *SEDIMENT.MEM*, and *UPDATE.MEM*. By using an and/or structure, the program selects which database is to be used by the response of the user. Once this response has been given, the program searches for the existence of a consultation name selected by the user by triggering the function MEMBER-BASE which then sends program control to the appropriate knowledge base.

At this point, control is directed by MAKE-BASE to the knowledge base questions and answers to determine the correct sediment type. The pattern of user responses prompts a selected answer which also asks whether it is correct. A negative response triggers CONSULT to request an alternative answer from the user. ALREADY-EXISTS will determine whether the user supplied answer already is in the database, and if it is will return control to CONSULT to deliver this information to the user. If, at that time, the user is sure of his or her thinking, the function REMOVE-SEDIMENT will remove the sediment name from the database, and replace it with the user's answer.

Conversely, if the user supplied answer is not in the knowledge base, the function DISPLACE will place a new question and a sediment associated with a yes response into the knowledge base, thus relegating the old answer to a no response. When the consultation is complete, CONSULT will ask whether the program should save the additional information. If yes, PRT-BASE merges and "pretty prints" to disk, the database used together with the new questions and answers supplied. Pretty printing merely places the appropriate parentheses and indentations into the new database.

The functions REMOVE-SEDIMENT and DISPLACE, the two most powerful functions in SEDRULE, allow for the creation of a dynamic database structure, thereby giving the program rudimentary learning capabilities. They rely on two of LISP's most frequently used command structures, the constructor (CONS) which adds items to a list, and the (CAR and CDR) functions which pull lists apart. The CAR, for example, returns the first item of a list, whereas a CDR returns everything but the first item of a list. CONS adds items of one list to those of another. By a thoughtful combination of these functions, items can be moved easily to both add new facies to our database or remove them.

PROGRAM EXECUTION

To run the program from the system prompt simply type the command "MULISP SEDRULE" after which the muLISP banner will be displayed. Alternatively, muLISP may be loaded separately followed by the LISP command "(RDS SEDRULE)" which implies read data-source SEDRULE. Although muLISP accepts both upper and lower case responses, the language is oriented toward capital letters and it is good practice to issue commands to the language in all capitals.

After the program loads, the SEDRULE banner appears with instructions to strike any key to continue. The program will then ask the user for his or her first name. This will allow SEDRULE to respond to the user by addressing him or her by name. The progam will also ask for the name of either a current or past consultation. This request allows SEDRULE to recall any changes made to the database during a past session, and it is important that the spelling be correct. If no changes were made during a past session, or if a new consultation name is indicated the program will not use past knowledge, but will reserve the present consultation name if changes are made to the database during the current session.

Once the consultation name has been provided and stored internally, SEDRULE asks the user whether additional information about sediments should be consulted. If the consultation name is recognized from a past session, for example, SEDRULE will ask the user whether he or she wants to use this information. If the user responds in the

Table 3. SEDRULE functions and their descriptions

Function	Utility
SEDIMENT	Defines knowledge bases and menus
CONSULT	Trigger questions and answers
ALREADY-EXISTS	Tests for existence of sediment type
REMOVE-SEDIMENT	Removes an existing sediment
MEMBER-BASE	Tests for existence of sediment base
MAKE-BASE	Creates the knowledge base selected
PRT-BASE	Merges and pretty prints database updates
DISPLACE	Adds new sediment types and questions
Y-OR-N-P	Accepts yes or no responses
READ-KEY	Accepts "any key" response
PRINTC	Prints the first line of a statement
CENTER	Centers text

affirmative the knowledge base "*UPDATE*.MEM" will be used during the current session, where *UPDATE*.MEM is the name of the database produced during a previous session. Otherwise an alternative knowledge base, either *INITBASE* or *SEDIMENT* will be used, where *INITBASE* is a small, built-in database contained in SEDRULE and *SEDIMENT* is a prebuilt knowledge base stored as SEDIMENT.MEM.

If the consultation name provided by the user is not recognized as a previous consultation the program will ask the user whether he or she wants to refresh the program's memory from the prebuilt database. A positive response will load SEDIMENT.MEM and use that database during consultation. Negative response will ignore SEDIMENT.MEM and will then use the small database *INITBASE* contained in the program code.

Once these questions have been asked and answered SEDRULE tells the user that he or she will be asked some yes-or-no questions about sediments. The pattern of the yes and no responses will determine which sediment type is selected. As SEDRULE arrives at an answer the user is prompted with a (Y/N) preceded by a message asking whether the user agrees with the response. If the response is Y for yes the program thanks the user and asks whether another consultation is requested. A negative response terminates the program and it returns to the system prompt.

If the user does not agree with the response SEDRULE asks for assistance by prompting the user to specify what sediment type he or she thinks it is. The user then may enter a new or more detailed sediment type by simply typing it. It does not matter whether capitals are used. If the new sediment type is identical to the answer just given SEDRULE gives an appropriate scolding and asks whether another consultation is requested. On the other hand, if the user provided sediment is contained elsewhere in the database, a suggestion is made that a particular question was answered incorrectly, asking the user if he or she is sure of their response to the question mentioned. If the user responds in the affirmative, the database is changed by replacing the sediment in the database by the user supplied sediment. A negative response prompts the program to ask whether another consultation is desired.

If the user supplied sediment name is not included in the database SEDRULE thanks the user for the response and asks what question could be asked for which a yes answer indicates the new sediment type rather than the one the program specified. Supplying such a question will allow the new sediment type to be retrieved if the answer to the supplied question is yes, whereas the old sediment type originally provided by SEDRULE will be triggered if the answer is no. If the sediment type supplied by SEDRULE has been modified in any way the program will ask the user whether they wish to save the changes provided. Such changes will be saved by merging the current database with the new information under the name *UPDATE*.MEM, where *UPDATE* is the user supplied consultation name. Finally SEDRULE returns the user to the system prompt if the program was called from the system prompt or to muLISP if it was loaded from within muLISP. From within muLISP simply type "(SYSTEM)" to return to the system prompt.

CREATING YOUR OWN DATABASE

A major feature of SEDRULE is that it is readily adaptable to the needs of any regional setting by allowing the creation of knowledge bases specific to the area. The knowledge base can be produced by using the muLISP editor or any text editor under nondocument mode. It is better to use the muLISP editor, however, because it uses a flashing mechanism to indicate the relationship between *left* and *right* parentheses. This is especially useful when creating a new knowledge base.

Such an external knowledge base includes only questions and their associated answers organized to correspond to the proper response pattern. One also will note that there are fewer parentheses in the external knowledge base than in the internal one. This is because SEDRULE provides the necessary parentheses for external knowledge bases.

To demonstrate how to produce an external knowledge base, I have simplified the existing internal knowledge base (Table 4). The general flow of the knowledge-base question and answer sequence can be followed by paying particular attention to the corresponding column locations of parentheses. A yes response redirects the linear flow to questions which are vertically aligned, whereas a no response continues the linear flow. This corresponds to the general flow of a procedural language IF-THEN-ELSE structure.

In its basic hierarchy, the first breakdown determines whether the sediment type is bedded. If the response to this question is no the program then goes to the next question in order. In this example the question asks whether the sediments have a mixture of textures. If the response is no, once again the program goes to the next line which is the answer VOLCANIC ASH. Alternatively, if the response to whether the sediments have a mixture of textures is yes, the program skips down to the matching location which in this example delivers the answer TILL, thus completing the questions under the first level of hierarchy based on the condition of the sediments not being bedded.

Should the response to the initial question concerning bedding be yes, the program bypasses the questions and answers about nonbedded sediments and asks whether the bedding is crossbedded. If not, SEDRULE asks whether the noncrossbedded

Table 4. Simplified internal knowledge base for SEDRULE

```
("Are the sediments bedded"
        ("Do the sediments contain a mixture of textures"
          "VOLCANIC ASH"
        ("TILL") ("Is there crossbedding"
     ("Is the material mostly silt or clay"
        ("Do the sediments occur only in depressions"
       "FLUVIAL OVERBANK SEDIMENT"
  ("LOESS")
     ("FLUVIAL SUSPENSION LOAD SEDIMENT")
("Is the crossbedding high angle (17 to 34 degrees)"
     "BEACH DEPOSIT"
("DUNE DEPOSIT")
```

material is mostly silt or clay. A yes response triggers the answer FLUVIAL SUSPENSION LOAD SEDIMENT, whereas a no response asks whether the materials occur only in depressions. An affirmative response triggers the answer LOESS, whereas a negative response triggers the remaining answer FLUVIAL OVERBANK SEDIMENT, thus eliminating all possible responses based on sediments which are bedded but which lack crossbedding.

The final set of questions is triggered if the response to whether the sediments are crossbedded is affirmative. SEDRULE asks whether the crossbedding is high angle, which, if true triggers DUNE SEDIMENT, and if false triggers the remaining BEACH DEPOSIT answer. This completes the possible answers contained in this highly limited knowledge base.

Creating a knowledge base of facts based on any selected set of production rules can be carried out in this fashion and the file created should be given the same name as that included in the program. For example, the external database in SEDRULE is termed SEDIMENT.MEM, which corresponds to *SEDIMENT* in the program. The database also must be contained on the same disk and directory in which SEDRULE resides, although modifications to this restriction can be made.

A final note for the first time user attempting to develop a knowledge base is that in many hierarchical classifications there is one question for each response. SEDRULE will not respond properly if there are an identical number of questions and responses because it assumes that a no response to the last question will trigger automatically the remaining sediment type. Attempts at creating external databases without this in mind will frequently yield a "NIL" response from the program indicating that there are too many questions and not enough answers.

SUMMARY

SEDRULE is a simple expert system with elementary learning capabilities. It is designed primarily as a pedagogical tool to assist beginning students in the field identification of sediments in locations where good outcrops occur. As the student's knowledge of sedimentary environments grows, the new and more specific sediment types can be included interactively, one at a time, as they are encountered.

The paucity of LISP code necessary to produce this program also will act as an incentive to the student to both examine the logic of sediment deposition and to produce additional databases and program enhancements to further its utility. By examining the construction of such databases the student will become more familiar with both their own knowledge domain and with those of other classificatory endeavors. Additionally, the exposure to manipulating such systems with microcomputer AI languages should result in more familiarity with the already growing expert systems in the geosciences (Grashion, 1987).

Acknowledgments—I wish to thank Dr. Lee Clayton for providing updated material for his sediment classification. Dr. Clayton may be contacted at the Wisconsin Geological and Natural History Survey, 3817 Mineral Point Road, Madison, WI 53705.

REFERENCES

Bonnet, A., and Dahan, C., 1983, Oil-well data interpretation using expert systems and pattern recognition techniques: Proc. Eighth Intern. Joint Conference on Artificial Intelligence, Karlshruhe, West Germany, p. 185–189.

Campbell, A. N., Hollister, V. F., Duda, R. O., and Hart, P. E., 1982, Recognition of a hidden mineral deposit by an artificial intelligence program: Science, v. 217, no. 4563, p. 927–929.

Clayton, L., 1973, Key for the identification of the major sites of sediment deposition based on characteristics seen in good outcrops: unpubl. lecture material, Univ. North Dakota, Grand Forks, 6 p.

Clayton, L., 1988, Guide to the identification of clastic sediment: unpubl. lecture material, Univ. Wisconsin, Madison, 11 p.

Conrad, M. A., and Beightol, D. S., 1988, Expert systems identify fossils and manage large paleontological databases: Geobyte, v. 4, no. 2, p. 42–46.

Davis, R., Austin, H., Caribom, I., Frawley, B., Pruchnik, P., Sneiderman, R., and Gelreath, J. A., 1981, The DIPMETER ADVISOR: interpretation of geological signals: Proc. Seventh Intern. Joint Conference on Artificial Intelligence, Vancouver, p. 846–849.

Duda, R. O., Gaschnig, J. G., and Hart, P. E., 1979, Model design in the PROSPECTOR consultant system for mineral exploration, *in* Michie, D., ed., Expert systems in the micro-electronic age: University Press, Edinburgh, p. 153–167.

Fisher, P., and Balachandran, S. C., 1989, STAX: A Turbo Prolog rule-based system for soil taxonomy: Computers & Geosciences, v. 15, no. 3, p. 295–324.

Fisher, P. F., Mackaness, W. A., Peacegood, G., and Wilkinson, G. G., 1988, Artificial intelligence and expert systems in geo-data processing: Progress in Physical Geography, v. 12, no. 3, p. 371–388.

Fisher, P. F., Pearson, M. D., Clarke, S. R., and Ragg, J. M., 1987, Computer programs to assist the automation of soil description: Soil Use and Management, v. 3, no. 1, p. 26–31.

Grashion, A. R., 1987, Expert systems: who needs them? Geology Today, v. 3, no. 6, p. 202–205.

Hart, A. B., McQueen, K. G., and Newmarch, J. K., 1988, A computer program which uses an expert systems approach to identifying minerals: Jour. Geol. Education, v. 36, no. 1, p. 30–36.

Hayes-Roth, F., Waterman, D. A., and Lenat, D. B., eds. Building expert systems: Addison-Wesley, Reading, Massachusetts, 444 p.

McCammon, R. B., 1986, The muPROSPECTOR mineral consultant system: U.S. Geol. Survey Bull. 1697, 35 p.

Miller, B. M., 1986, Building an expert system helps classify sedimentary basins and assess petroleum resources: Geobyte, v. 2, no. 2, p. 44–48.

Pettijohn, F. J., 1975, Sedimentary rocks (3rd ed.): Harper & Row, New York, 628 p.

Robinson, V. B., and Frank, A. U., 1987, Expert systems for geographic information systems: Photogrammetric Engineering and Remote Sensing, v. 53, no. 10, p. 1435–1441.

The Soft Warehouse, 1986, muLISP-86 reference manual: Honolulu, Hawaii, 408 p.

The Soft Warehouse, 1987, muLISP-87 reference manual: Honolulu, Hawaii, 439 p.

Winston, P. H., and Horn, B. K. P., 1981, LISP: Addison-Wesley, Reading, Massachusetts, 430 p.

APPENDIX

Program Listing

```
;File:  SEDRULE.LSP

(LOOP (PRIN1 '*) (EVAL (READ)) ((NULL RDS) (SYSTEM)) )

;Define database types and create opening menus.

(DEFUN SEDIMENT (
    *BASE* *UPDATE* *NEW* ) ;Databases used
  (CLEAR-SCREEN)
  (TERPRI 25) ;25 blank lines create scrolling action
  (CENTER "* * * * * * * * * * * * * * * * * * * * * *")
  (CENTER "* * *                                 * * *")
  (CENTER "* * *  SEDRULE: BY M. N. DEMERS       * * *")
  (CENTER "* * *                                 * * *")
  (CENTER "* * * * * * * * * * * * * * * * * * * * * *")
  (CENTER "* * * * * * * * * * * * * * * * * * * * * * * * * * * * *")
  (CENTER "* * *                                               * * *")
  (CENTER "* * *    A RULE BASED FIELD ORIENTED TUTORIAL       * * *")
  (CENTER "* * *                                               * * *")
  (CENTER "* * *  DESIGNED TO TEACH THE BASICS OF INQUIRY      * * *")
  (CENTER "* * *                                               * * *")
  (CENTER "* * *    FOR CLASTIC SEDIMENT IDENTIFICATION        * * *")
  (CENTER "* * *                                               * * *")
  (CENTER "* * * - - - - - - - - - - - - - - - - - - - - - - - * * *")
  (CENTER "* * *                                               * * *")
  (CENTER "* * *        BASED ON CLAYTON (1973, 1988)          * * *")
  (CENTER "* * *                                               * * *")
  (CENTER "* * * * * * * * * * * * * * * * * * * * * * * * * * * * *")
  (TERPRI)
  (PRINTC "Press any key to continue.")
  (READ-KEY) ;Check to see if key has been pressed.
  (CLEAR-SCREEN)
  (PRINTC "Please type in your first name and press the RETURN key so I")
  (PRINC "will know what to call you. ---> ")
  (CLEAR-INPUT)
  (SETQ NAME (STRING-CAPITALIZE (READ-LINE))) ;Cap first letter of name.
  (TERPRI 2)
  (PRINTC "Please give a current or past consultation name of eight letters")
```

```
  (PRINC "or less for storage and recall. ——> ")
  (SETQ *UPDATE* (READ-LINE)) ;Create a name for user data base.
  (CLEAR-INPUT)
  (TERPRI 2)
  (PRINTC "Thanks " NAME ", now think of the characteristics of your")
  (PRINTC "sediment and I will try to assist you in its identification")
  (PRINTC "by asking you some yes-or-no questions.")
  (TERPRI)

;Select the database to use for classification.

  (SETQ *BASE* *INIT-BASE*)
  ( ((OR
       (AND (RDS (PACK* *UPDATE* '.MEM))
         (PRINTC "Would you like to refresh my memory about sediments")
         (Y-OR-N-P "from our previous session?") )
       (AND (RDS 'SEDIMENT.MEM)
         (PRINTC "Would you like to refresh my memory about sediments")
         (Y-OR-N-P "from the pre-built sediment data file?") ) )
     (SETQ *BASE* (MAKE-BASE (READ))) ) )
(CLEAR-SCREEN)


;Or get out of the program entirely

(OR
    (AND
    (PRINTC "Are you sure you wish to continue this consultation. ")
    (Y-OR-N-P "If not I will return you to the operating system?") )
        (SYSTEM)
  (RDS)


;Trigger function consult, query user.

  (LOOP
    (TERPRI)
    (PRINC "When you are ready please press any key to begin consultation. ")
    (READ-KEY)
    (CLEAR-SCREEN)
    (TERPRI 2)
    (CONSULT *BASE*) ;Consult the current knowledge base.
    ((NOT (Y-OR-N-P "Would you like to do another consultation?")))
    (CLEAR-SCREEN) )
  (TERPRI)
  ( ((NOT *NEW* )) ;If no new input skip all of this.
   (PRINTC NAME ", do you want me to save the new sediments in " *UPDATE* " ")
    ((Y-OR-N-P "for use next time you consult SEDRULE?")
      (WRS (PACK* *UPDATE* '.MEM)) ;Otherwise add it to the user knowledge.
      (PRT-BASE *BASE* 0) ;And pretty print it.
      (TERPRI)
      (WRS)
      (SETQ *NEW*) ;Set the knowledge base status to new.
      (TERPRI) )
    (TERPRI) ) )

;Define the function which performs the consultation.

(DEFUN CONSULT (BASE
    SEDIMENT ANSWERS NEW-BASE QUESTION)
  (LOOP
    ((ATOM (CDR BASE))


;Deliver answer and query as to its correctness.

(CLEAR-SCREEN)
  (PRINTC "Would you agree that the sediment described could be classed")
  ((Y-OR-N-P (PACK* "as" (CAR BASE) "?"))
(TERPRI)


;If answer is acceptable thank the user.

(PRINTC "Thanks for the consultation " NAME ".")
(TERPRI) )


;If answer is not acceptable ask for alternative answer.
```

```
(PRINTC "I need assistance, what sediment do you think it is?  ")
  (PRINC "")
  (SETQ SEDIMENT (STRING-UPCASE (READ-LINE))) ;Capitalize sediments.
(TERPRI)


;If suggested answer same as SEDRULE answer tell user.

((EQ SEDIMENT (CAR BASE))
  (PRINTC "Check again " NAME ", that was my suggestion.")
(TERPRI) )


;If suggestion is defined elswhere check for correct response.

(SETQ NEW-BASE (ALREADY-EXISTS SEDIMENT (REVERSE ANSWERS) *BASE*))
  ( ((NOT NEW-BASE))
      (PRINTC "I think you may have incorrectly answered the question")
      (TERPRI)
      (SPACES 10)
  (PRINTC """ (CAR NEW-BASE) "?""")
(TERPRI)


;If the questions were answered correctly change rule base.

((Y-OR-N-P (PACK* NAME
" are you sure you answered this question correctly?"))
  (REMOVE-SEDIMENT SEDIMENT *BASE*)
  (PRINTC "OK, I fixed by memory so I won't make that mistake again.")
  (TERPRI) )
(TERPRI)
(RETURN) )
      (PRINTC "What question can I ask for which a YES answer indicates")
      (PRINTC (@ SEDIMENT) " rather than " (@ (CAR BASE)) "?")
      (SETQ QUESTION (STRING-RIGHT-TRIM "?.! " (READ-LINE)))
      (TERPRI)
      (DISPLACE BASE (LIST QUESTION (LIST (CAR BASE)) (LIST SEDIMENT)))
      ;Adds new question and sediment during this session only.
      (PRINTC "Thanks for the information.  I'll store it for later use.")
      (SETQ *NEW* T) )
    (PUSH (Y-OR-N-P (PACK* (CAR BASE) "?")) ANSWERS)
    (TERPRI)
    ( ((CAR ANSWERS)
        (SETQ BASE (CADDR BASE)) )
     (SETQ BASE (CADR BASE)) ) ) )


;Define the function to test for existence of sediment type in fact base.

(DEFUN ALREADY-EXISTS (SEDIMENT ANSWERS BASE)
  ((ATOM (CDR BASE)) NIL)
  ((CAR ANSWERS)
    ((MEMBER-BASE SEDIMENT (CADR BASE)) BASE)
    (ALREADY-EXISTS SEDIMENT (CDR ANSWERS) (CADDR BASE)) )
  ((MEMBER-BASE SEDIMENT (CADDR BASE)) BASE)
  (ALREADY-EXISTS SEDIMENT (CDR ANSWERS) (CADR BASE)) )


;Define the function to remove an existing sediment from the fact base.

(DEFUN REMOVE-SEDIMENT (SEDIMENT BASE)
  ((MEMBER-BASE SEDIMENT (CADR BASE))
    (SETQ SUB-BASE (CADR BASE))
    ((EQ SEDIMENT (CAR (CADR SUB-BASE)))
      (RPLACA (CDR BASE) (CADDR SUB-BASE)) )
    ((EQ SEDIMENT (CAR (CADDR SUB-BASE)))
      (RPLACA (CDR BASE) (CADR SUB-BASE)) )
    (REMOVE-SEDIMENT SEDIMENT (CADR BASE)) )
  (SETQ SUB-BASE (CADDR BASE))
  ((EQ SEDIMENT (CAR (CADR SUB-BASE)))
    (RPLACA (CDDR BASE) (CADDR SUB-BASE)) )
  ((EQ SEDIMENT (CAR (CADDR SUB-BASE)))
    (RPLACA (CDDR BASE) (CADR SUB-BASE)) )
  (REMOVE-SEDIMENT SEDIMENT (CADDR BASE)) )


 ;Define the function which tests if sediment base exists.
```

```
(DEFUN MEMBER-BASE (SEDIMENT BASE)
  (LOOP
    ((ATOM (CDR BASE))
      (EQ (CAR BASE) SEDIMENT) )
    ((MEMBER-BASE SEDIMENT (CADR BASE)))
    (SETQ BASE (CADDR BASE)) ) )


;Define the function which creates the data base selected.

(DEFUN MAKE-BASE (BASE)
  ((ATOM BASE)
    (LIST BASE) )
  (LIST (CAR BASE) (MAKE-BASE (CADR BASE)) (MAKE-BASE (CADDR BASE))) )


;Define the function which merges existing rule base to user memory file.
;Also add appropriate indentations and parentheses.

(DEFUN PRT-BASE (BASE TAB PRIN1)
  (SPACES TAB)
  ((ATOM (CDR BASE))
    (PRIN1 (CAR BASE)) )
  (PRINC '"(")
  (PRINT (CAR BASE))
  (PRT-BASE (CADR BASE) (+ TAB 2))
  (TERPRI)
  (PRT-BASE (CADDR BASE) (+ TAB 2))
  (PRINC '")") )


;Define the initial (default) rule base.

(SETQ *INIT-BASE*
 '("Are the sediments bedded"
     ("Do the sediments contain a mixture of textures"
("VOLCANIC ASH")
  ("TILL"))
   ("Is there crossbedding"
     ("Is the material mostly silt or clay"
("Do the sediments occur only in depressions"
 ("FLUVIAL OVERBANK SEDIMENT")
("LOESS"))
     ("FLUVIAL SUSPENSION LOAD SEDIMENT"))
   ("Is the crossbedding high angle (17 to 34 degrees)"
     ("BEACH DEPOSIT")
       ("DUNE DEPOSIT")) )))

;Define the Displace function to add new sediment and question.


(DEFUN DISPLACE (LST1 LST2)
  (RPLACA LST1 (CAR LST2))
  (RPLACD LST1 (CDR LST2)) )


;Define the function which accepts yes or no responses.

(DEFUN Y-OR-N-P (MSG
    CHAR READ-CHAR RDS WRS )
  ( ((NULL MSG))
    (FRESH-LINE)
    (WRITE-STRING (PACK* MSG " (Y/N) ")) )
  (CLEAR-INPUT)
  (LOOP
    (SETQ CHAR (CHAR-UPCASE (READ-CHAR)))
    ((EQ CHAR 'Y)
      (WRITE-LINE CHAR)
      T )
    ((EQ CHAR 'N)
      (WRITE-LINE CHAR)
      NIL )
    (WRITE-BYTE 7) ) ) ;Beep if not yes or no response.


;Define the read-key function to accept user response.

(DEFUN READ-KEY (
```

```
    READ-CHAR RDS)
  (READ-CHAR) )


;Define the function which prints the first line of a statement.

(DEFUN PRINTC LST
  (PRINC (PACK LST))
  (TERPRI)
  'T )


;Define the function which centers opening text.

(DEFUN CENTER (MSG)
  (SET-CURSOR (ROW)
(TRUNCATE (- (CADDDR (MAKE-WINDOW)) (LENGTH MSG)) 2))
  (WRITE-LINE MSG) )

(SEDIMENT (RDS))

;SEDIMENT.MEM    DATA BASE FILE FOR SEDRULE

("Does it show clear evidence of bedding"
;If it does go to crossbedding question.  If not, ask the next question.

  ("Is it composed mostly of sand"
    ("Is it composed mostly of silt"
     ("Is it composed mostly of clay"
      ("Is it composed mostly of poorly sorted mixtures of material"
       "HIGHLY BURROWED DEPOSIT OR HILLSLOPE DEPOSIT"
      "GLACIAL DEPOSIT")
     "PLASTIC FLOW SEDIMENT")
    "FLUVIAL OVERBANK SEDIMENT")
   "HIGHLY BURROWED NEARSHORE SEDIMENT")

   ("Is crossbedding present"
   ;If yes, go to high angle question.  If not, ask next question.

("Does it occur draped up and down hills and depressions"
   ("Is the bedding indistinct, with buried soils common"
    ("Is it uniformly and rhythmically bedded upward from silt to clay"
     ("Is the material mostly coarse silt, with indistinct bedding"
        "DISCONTINUOUS TURBIDITY CURRENT SUSPENSION LOAD SEDIMENT"
        "OFFSHORE NON-BOTTOM CURRENT SUSPENSION LOAD SEDIMENT")
     "CONTINUOUS TURBIDITY CURRENT SUSPENSION LOAD SEDIMENT")
   "FLUVIAL OVERBANK SEDIMENT")
 "VOLCANIC ASH OR LOESS")

("Is the crossbedding high angle, i.e. .3 to .6 radians"
 ;If yes, ask about the size of the beds.  If no, ask next question.

("Are the sets greater than 50 cm thick"
   "DEEP NEARSHORE RIPPLE SEDIMENT"
     "SHALLOW NEARSHORE RIPPLE SEDIMENT")
   ("Is the material well sorted"
      ("Is the material mostly gravel or sandy gravel"
         "SLOPEWASH SEDIMENT"
           "UPPER FLOW REGIME FLUVIAL SEDIMENT")
("Are the beds grading upward from fine to medium sand"
   ("Are the beds grading upward from gravel to sand"
      ("Is the sediment body grading seaward into forebeach sediment"
        ("Do the bedding planes have long, straight, parallel ripple marks"
    "LOWER FLOW REGIME TRACTION LOAD OR UPPER FLOW REGIME FLUVIAL SEDIMENT"
  "SHALLOW NEARSHORE BAR SEDIMENT")
  "TOPBEACH SEDIMENT")
 "FOREBEACH SEDIMENT")
"EOLIAN TRACTION LOAD SEDIMENT")

;Move here if answer to high angle crossbedding question is no.
   ("Are the largest sets thicker than 50 cm"
      ("Are the sets very trough-shaped"
           ("Do the sets have some associated nearshore bar sediment"
              "DEEP NEARSHORE RIPPLE SEDIMENT"
            "SHALLOW NEARSHORE RIPPLE SEDIMENT")
         "FLUVIAL RIPPLE SLIPFACE SEDIMENT")
   ("Are the sets grouped"
```

```
    ("Do the beds abut up dip against a steep unconformity under them"
       "TALUS")
("Is the sediment body elongated perpendicular to the regional slope"
   "DELTA SLIPFACE SEDIMENT"
     ("Is the material primarily gravel"
        "BACKBEACH SEDIMENT"
          "FOREBEACH SEDIMENT")
("Are the sets primarily trough shaped sand or gravel"
   ("Are the sets from 50 mm to as much as 3 m thick"
        "SHALLOW NEARSHORE BAR SEDIMENT")
          "FLUVIAL DUNE SLIPFACE SEDIMENT")
("Do the sets have associated small scale crossbedding"
     "EOLIAN DUNE SLIPFACE SEDIMENT"
     "OCEAN CURRENT DUNE SLIPFACE SEDIMENT")) )))
```