Paper 135-26

Telling Aardvarks from Zebras with an Expert System Written in SAS®

Anthony Dymond, Dymond and Associates, LLC, Concord, California

ABSTRACT

Expert systems are a branch of artificial intelligence that encapsulate the knowledge and behaviors of a subject matter expert in their domain of expertise. The expert's automated services then become available throughout the enterprise. Classification and diagnosis represent an important class of problems that have traditionally been a major objective of expert systems.

An object-oriented expert system is now available that will allow a user to create knowledge bases written in Base SAS or a combination of Base SAS and SAS Component Language (SCL). Implementation of a knowledge base to classify mammals is described here as an example of the expert system used for this general class of knowledge bases. The discussion includes: identifying objects and placing them in a search tree, object design, method design, method inheritance, blackboard databases, and search engine control. Mammals are chosen as a subject because their classification is not an overly complex design, and because everyone is to some degree a subject matter expert.

INTRODUCTION

Expert systems are a branch of artificial intelligence that encapsulate the knowledge and behaviors of a subject matter expert in their domain of expertise. The expert can serve a variety of functions, such as an expert consultant, a teacher, a support center technician, a librarian, an underwriter, or an expert assistant who reliably performs a repetitious task. Some successful expert systems include MYCIN, which used diagnostic information to recommend antibiotics; PROSPECTOR, which used geological data to recommend drilling sites; and R1, which could recommend the layout of VAX computer hardware (Jackson, 1999).

Classification and diagnosis represent a general group of problems that can be addressed using expert systems. Domain knowledge can often be decomposed into a taxonomic-like structure that is amenable to inclusion in an expert system's knowledge base. For example, equipment failures decompose from general systems to intermediate modules to specific components; types of animals decompose through class, phylum, order, family, genus, and species. This general decomposition can often be extended to include less obvious subjects; a record in a database can be classified as normal or in error, and in turn subclassified to types of error or types of fraud.

In each example we would expect to produce an expert system that could provide expert advice within its domain. This is not to say that these expert systems would be of similar complexity. Although of similar general form, an expert system classifying trees will be substantially easier to build than one attempting medical diagnoses.

An object-oriented expert system is now available that will allow a user to create knowledge bases written in Base SAS or a combination of Base SAS and SCL (Dymond, 2000). Implementation of a knowledge base to classify mammals is described here as an example of this general class of knowledge bases. Mammals are chosen as a subject because their classification is not an overly complex design, and because everyone is to some degree a subject matter expert.

KNOWLEDGE BASE DESIGN

Objects and Trees

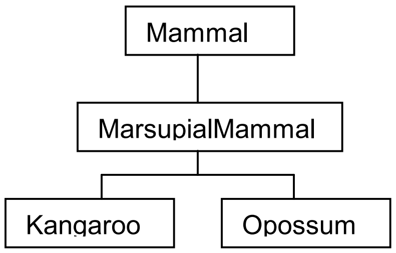
An expert system is composed of three main parts: an inference engine, a database, and one or more knowledge bases (Figure 1). The inference engine provides both development tools and a run-time environment. The knowledge bases serve a role similar to separate applications and are executed under the control of the inference engine. The database contains the current data from an ongoing session.

A subject matter expert's knowledge is captured in the knowledge base as objects in a graph. The graph provides a mechanism to navigate between the objects, and it is implemented in this expert system as a search tree explored by well-known tree search algorithms such as depth-first search.

A general approach to designing a knowledge base is to arrive at a set of objects and then consider how the resulting objects should be located in the tree. For knowledge in the general area of classification and diagnosis, this process is often straightforward, with both the identification of objects and their location in the tree closely resembling a class diagram of the subject area. The task is simplified for mammal classification since the class diagram is essentially the taxonomic structure found in numerous zoology books. The top levels of this tree would look like:

```
Mammal
  |-EggLayingMammal
  |   |-DuckBilledPlatypus
  |   |-SpinyAntEater
  |
  |-MarsupialMammal
  |  |-Kangaroo
  |  |-Opossum
  |
  |-PlacentalMammal
  |  |-Aardvark
  |  |-Carnivore
  |  |  |-Bear
  |  |  |-Cats
  |  |  |  |-MountainLion
  |  |  |  |-Tiger
```

After the objects have been identified and located in the search tree, the next task is to design the attributes and methods within the objects. This design will be influenced by what the object represents, how its purpose will be satisfied during knowledge base execution, and how objects will interact with or be influenced by each other. The short tree segment surrounding the object MarsupialMammal will be used to illustrate this design process.



Object Design

The purpose of the object MarsupialMammal is to conclude if an animal is or is not a marsupial mammal. This fact can have values of "true," "false," or "unknown." We can conclude the animal is a marsupial mammal if it is either a kangaroo or an opossum. However, we could also reach this conclusion if we know the animal is a mammal and has a marsupial pouch. The design of the object MarsupialMammal should then be such that the attribute (MarsupialMammal.marsupialMammal) has values in ('true','false','unknown') as determined by the tests:

```
 :- (Mammal.mammal) and
        (MarsupialMammal.marsupialPouch)
     or
 :- (Kangaroo.kangaroo)
     or
 :- (Opossum.opossum)
```

Blackboard Database

The first test needs information about "marsupialPouch," which is an attribute within the object MarsupialMammal. Other tests require values for "mammal," "kangaroo," and "opossum." These are attributes of their corresponding objects. Values for all attributes will be read from the database when they are needed. This expert system uses a memory-resident "blackboard" database, and all objects in the knowledge base can read and write against the database. The database has an object-attribute-value structure, with each row also containing an index field.

Search Engine

The purpose of the search tree is to provide a navigational framework between the objects. For example, if (MarsupialMammal.marsupialMammal='unknown') after the tests have executed, then the search engine should next execute tests in the object Kangaroo. This would cause a value to be written in the database for the attribute (Kangaroo.kangaroo). The search engine would then return to the object MarsupialMammal and execute the tests there again. If the value for kangaroo had been changed to (Kangaroo.kangaroo='true'), then it would now be possible to set (MarsupialMammal.marsupialMammal='true'), and the purpose of the object MarsupialMammal would be accomplished.

Note that this search strategy is just what would be expected from a depth-first search algorithm operating in this region of the tree. All that is necessary is for the objects to pass information to the search algorithm about whether or not the objects' goals have been satisfied. This is done through an attribute (ObjectName.xostatus) within each object. By manipulating the xostatus attribute values, the search engine can be directed to return as soon as possible to the beginning goal node, or to do an exhaustive search of some of the leaf nodes containing the detailed classifications. These search algorithms can be very efficient, gathering only the necessary information to accomplish their tasks within large knowledge bases.

Inheritable Methods

All the tests are contained in labeled blocks of code call methods. The object Mammal has methods that attempt to provide a value for (Mammal.mammal).

One of the advantages of using search trees as a navigational tool is that the search can begin at any node in the tree. However, one of the attribute values needed by methods in the object MarsupialMammal is "mammal," which is instantiated in the object Mammal. If search starts below the object Mammal, then Mammal's methods will not have run and this attribute will not have a value in the database.

The solution to this lies in inheritable methods. Although the methods to provide a value for (Mammal.mammal) belong to the object Mammal, the object MarsupialMammal can ask to run these methods. If a method is not found within an object, the method inheritance process will travel up the tree looking for a method matching the name. If one is found, it is taken and run within the scope of the object being executed.

Coding the Methods

Methods in this expert system are labeled blocks of Base SAS or SCL code. There can be one or more methods in an object, and each method can contain one or more data steps and PROCs. An example of a user written method could be:

```
XOMETHOD=hiWorld
   data _null_;
      put 'hiWorld';
   run;
XOENDMETHOD
```

There are also a number of system methods that can be used anywhere in the code. They have the general form:

```
   %methodName(ObjectName,<args>)
```

For example, in the code fragment

```
%getval('Mammal','mammal',mammal);
%getval('MarsupialMammal','marsupialPouch',
         pouch);
if mammal='true' and pouch='true' then do;
   %setval('MarsupialMammal','marsupialMammal',
           'true');
```

%getval() is a method that reads an attribute value from the database and returns it in a local variable. The code fragment tests mammal and pouch and, if both are true, the system method %setval() writes (MarsupialMammal.marsupialPouch='true') in the database.

Since many of the methods in this knowledge base are very similar, it is possible to provide some template methods to simplify the method coding process. Figure 2 shows such a template for a method that would conduct a proof with one "and clause." The template also allows the user to be queried for information using a system method %xoask().

RUNNING THE EXPERT SYSTEM

A session starts by using the mouse to highlight a node in the tree to mark it as the goal node, and then selecting "run" from the main menu. A search algorithm is chosen, a box is checked if this is to be an exhaustive search, and the search process begins. As the session progresses, the object currently being explored is highlighted in the tree, and the method being run is displayed on the message line. User query windows controlled by %xoask() methods open as needed to gather information from the user.

When the session completes, the system displays a message box restating the goal node and search algorithm, and providing information on the completion status. At this time, a database editor can be used to review or change entries in the blackboard database. The tree, methods, and database can be stored as items in a SAS catalog, allowing the run-time state of the system to be saved and reopened later.

CONCLUSIONS

Classification and diagnosis represent an important class of problems, and have traditionally been a major objective of expert systems. In this example, an expert system to help identify mammals is constructed by placing objects into a search tree and filling out method templates. The knowledge base can be rapidly built and can be mostly maintained by end users who are not programmers. Many practical knowledge bases, such as for business rules or call center support, can be built using similar techniques.

REFERENCES

Dymond, A.M. (2000), "An Object-Oriented Expert System for the SAS Environment," *Proceedings of the Eighth Annual Western Users of SAS Software Regional Users Group Conference*, 8, 454-458.

Jackson, P. (1999), *Introduction to Expert Systems*, Harlow, England: Addison Wesley Longman Limited.

ACKNOWLEDGEMENTS

Automated Consultant^{tm} is a trademark of Dymond and Associates, LLC.

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

CONTACT INFORMATION

Anthony M. Dymond, Ph.D.
Dymond and Associates, LLC
4417 Catalpa Ct.
Concord, CA 94521

(925) 798-0129
(925) 680-1312 (FAX)
amdymond@dymondassoc.com

A SAS Institute Inc. Quality Partner

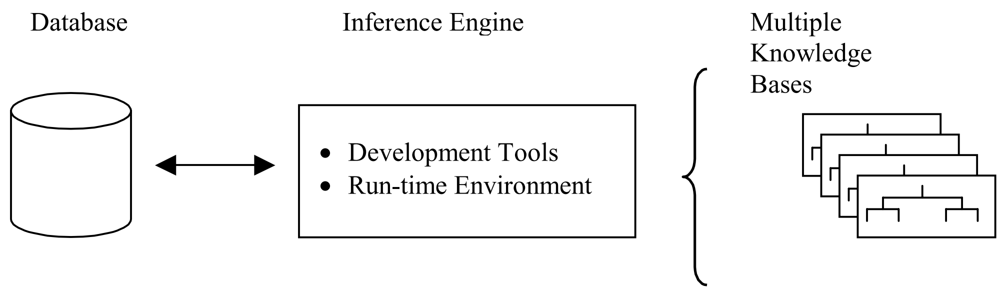


Figure 1: An expert system is composed of an inference engine, a database, and multiple knowledge bases.

```
PROOF01=proveMarsupial;

TARGET={MarsupialMammal,marsupialMammal};

ANDCLS={Mammal,mammal}                  eq 'true';
       {MarsupialMammal,marsupialPouch} eq 'true';
ENDCLS;

ASKUSR=%xoask3(xodb,MarsupialMammal, marsupialPouch,work.dsname,1,
        N,Does the animal have a marsupial pouch?);
ENDASK;

ENDPROOF01;
```

Figure 2: Method template that codes the AND clause in the object MarsupialMammal.