Knowledge Acquisition Using Linguistic-Based Knowledge Analysis

Daniel L. Schmoldt
USDA Forest Service
Brooks Forest Products Center
Virginia Tech
Blacksburg, Virginia 24061-0503
schmoldt@vt.edu

ABSTRACT. Most knowledge-based system development efforts include acquiring knowledge from one or more sources. Difficulties associated with this knowledge acquisition task are readily acknowledged by most researchers. While a variety of knowledge acquisition methods have been reported, little has been done to organize those different methods and to suggest how to apply them within a conceptual framework. The *linguistic-based knowledge analysis* apreach described here offers a conceptual approach wherein knowledge in a subject area is analyzed as (1) lexical, (2) syntactic, and (3) semantic knowledge. Analysis of these three separate components of knowledge creates a domain "language" that completely describes the knowledge of a particular subject area in logical detail. This "language" serves as a knowledge model that can be readily translated into a software implementation. The resulting knowledge-based system is able to "converse" in the language of its domain about particular types of problems. Many previously reported knowledge acquisition techniques are effective, to varying degrees, for acquiring lexical, syntactic, and semantic knowledge.

A major obstacle to the development of knowledge-based systems (KBS) is the expensive work of acquiring expertise from various sources and encoding it into a computer representation of that knowledge. Ideally, of course, computers should learn from experience just as people do. That is, a human specialist teaches a computer what is correct and the computer adjusts its understanding" and behavior in order to be "smarter" (Michie and Johnston 1985). We want to be able to show a computer examples of things, be they statements of fact, pictures, sample actions, etc. and have it, by a process of discovery, formulate ideas that connect those examples, finding patterns within these events and ideas (Langley and Carbonell 1984). Only in this way can the bottleneck of encoding knowledge into knowledge representation structures be overcome (Michie and Johnston 1985). Unfortunately, it will likely be some time before our "intelligent" creations are capable of learning in that manner. For the present, we must be content to work closely, either directly or indirectly, with those sources of expertise we desire to cap ture.

There seems to be general agreement among KBS developers that knowledge acquisition (KA) is critical to any system development project. Consensus regarding the importance of knowledge acquisition, however, is not generally reflected in the level of effort committed to this task, nor to the extent that such activities are reported in the literature on KBS applications. Most reports of system development only contain a very cursory presentation of KA activities, or none at all. Descriptions of system design and architecture, programming tools, and system operation constitute most of all the system development efforts reported.

Some KA reports have appeared in the natural resources literature. Gordon (1989) provides an overview of KA methods, in general. Hoffman (1987) also reviews a number of KA methods, but within the context of different problem-solving tasks and the data required by them. Windon and Massey (199 1) describe the use of unstructured interviews, structured interviews, and protocol analysis, while Evanson (1988) presents techniques and guidelines for the interview process. Others have reported on individual techniques, such as question probes (Gordon and Gill 1989), multi-expert elicitation using a modified Delphi procedure (Schmoldt and Bradshaw 1988), hypothetical test cases (Senjen 1988), multi-expert elicitation using questionnaires (Schmoldt and Peterson 1991), dual expert elicitation (Finkelstein and Enksson 1994), graph-based elicitation of suitability factors (Nevo *et al.* 1990), and multiple-expert elicitation within a blackboard architecture (Clarke *et al.* 1990). Still others have provided some details of knowledge acquisition applied to specific system development efforts (Bridges et al. 1995, DeMeers 1989, Downing and Bartos 1991, Ekblad et al. 1991, Goforth and Floris 1991, Haas 1992, O'Hara et al. 1990, Rust 1988). Nevertheless, the number of reports about knowledge acquisition efforts and techniques represents only a small fraction of the KBS projects reported.

Many natural resources domains present unique sets of problems with respect to acquiring existing expertise. Despite their uniqueness, however, they also share some commonalties that can be exploited for successful knowledge acquisition. Sharing KA experiences through more extensive and comprehensive communications of KA activities can benefit all those involved in development projects. This report takes one step in that direction by providing a theoretical approach called *linguistic-based knowledge analysis*, and demonstrating how many existing KA methods can be applied within this theoretical framework.

Knowledge Acquisition in Overview

Four primary sources of knowledge are: literature, human specialists, existing models (e.g., mathematical models, existing problem-solving procedures), and examples (Sell 1985). Knowledge acquisition refers to the process of locating, collecting, organizing, synthesizing, and formalizing the information, concepts, and strategies that pertain to some subject area of interest. Although more technical definitions of knowledge can be made (q.v., Schmoldt and Rauscher 1994), in this paper, knowledge is used generally to refer to any data, information, or justified true belief (knowledge).

Figure 1 illustrates how the KA process fits into KBS development. The goal of the system developer is to combine the knowledge that is scattered

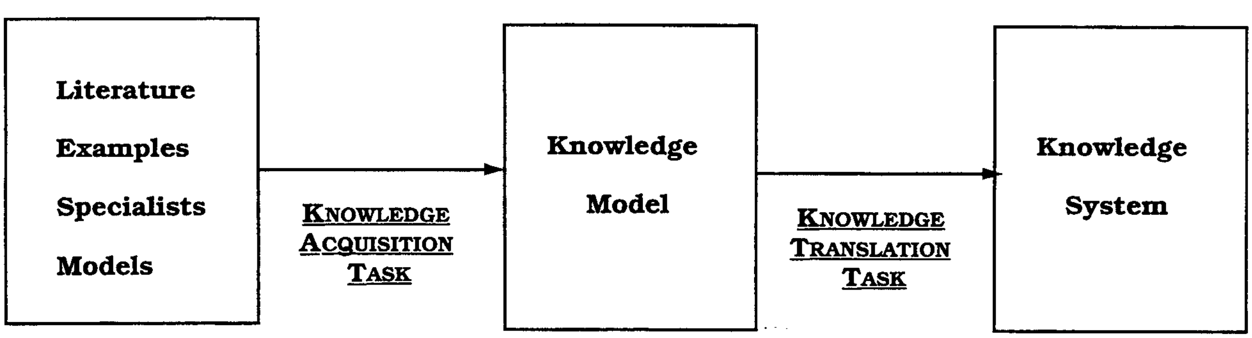

Figure 1. *Knowledge system development contains two subtasks. The first creates a knowledge model through a knowledge acquisition process and the second produces a KBS through a knowledge translation/implementation process.*

throughout these four sources to produce a useful knowledge system. To do this, he or she must complete a KA task to create a *knowledge model* and must also perform a knowledge translation task (implementation) to encode that model into an executable computer program.

A knowledge model includes the knowledge and rationale that must be applied to solve problems in the subject domain (Schmoldt and Rauscher 1996). An example knowledge model appears in Rauscher et al. (1995). A knowledge model can be represented in many ways, e.g., tables, diagrams, outlines, and at many levels of detail, e.g., goals and subgoals only, or case-specific reasoning. But eventually a knowledge model must completely and specifically outline the domain knowledge so that someone can apply knowledge representation methods and search procedures to create a knowledge-based system.

Methods of collecting, organizing, and formalizing knowledge vary substantially depending on the knowledge source. For example, knowledge in literature has already been collected and organized, but someone must still synthesize and formalize it so that it can be implemented in a KBS. Human expert knowledge, on the other hand, is woven from human experience and training into internal mental models of how the world works and therefore requires collection, organization, synthesis, and formalization. Knowledge that exists in models is already collected, organized, synthesized, and formalized, but it may be necessary to re-formalize it or to integrate it with other knowledge. By carefully analyzing knowledge to create a domain "language," it becomes possible to combine knowledge from several different sources in a cohesive and coherent way.

Linguistic-Based Knowledge Analysis

Linguistic-based knowledge analysis approaches KA by categorizing knowledge as belonging to one of three major types: lexical knowledge, syntactic knowledge, and semantic knowledge. Much as a linguist examines a language as lexicons connected by syntax formed into semantic expressions, this type of knowledge analysis creates a language for the subject domain of interest. It does so by examining each type of knowledge separately. These three types of knowledge also exhibit an implicit sequential ordering to their analysis—lexical analysis must precede syntactic analysis, and syntactic analysis must precede semantic analysis. Each is designed to generate different features of a knowledge model. Any KA effort should include an analysis of each of these types of knowledge.

Identifying, labeling, and describing conceptual and physical "objects" in a domain is the first step. These "objects" are decision *factors,* i.e., conditions, variables, and domain entities that an expert talks about when solving problems in the subject area. *Lexical knowledge analysis* creates the lexicons that make up the domain language, from which it becomes possible to subsequently discuss knowledge structure (syntax) and tactical and strategic knowledge (semantics). *Syntactic knowledge analysis* identifies, labels, and describes the relationships among the factors identified in lexical analysis. *Semantic knowledge analysis* focuses on specific combinations of factors (lexicons) and relationships (syntax) to indicate plausible avenues of search toward problem solutions. Semantic analysis adds meaning to our lexicons and syntax to produce a domain language, telling us when and how various concepts and structures can be used to solve problems—to make domain "sense" out of application problems. Because system development is an iterative process of building, revising, and rebuilding, knowledge analyses will be expanded continually as new prototype versions are constructed.

Lexical Knowledge Analysis

During any preliminary KBS planning phase, system developers should have characterized typical subject matter data and should have identified the sources of those data (Schmoldt and Rauscher 1996). Now, as KA is begun, the details of the problem-solving process must be analyzed and described. One of the key tasks to help understand a problem-solving process is to analyze the factors that are important for decision making; that is, what pieces of information are used to solve a problem? These will become the lexicons in the constructed domain language.

First of all, any objects in the domain that will be part of the decision-making process are given a name. This may not be as trivial as it sounds. Some abstract concepts are difficult to label (Benfur and Furbee 1989). Most experts encounter substantial difficulty with the task of creating lists of decision factors off the top of their heads. When presented with a partial list a priori, however, it is much easier for them to focus on important factors, refining those factors already present on the list, and adding others to it. Therefore, problem examples (actual or hypothetical), books, manuals, etc. are often the most helpful sources for enumerating decision factors and finding unique names for them. Schmoldt (1987) used this approach to enumerate symptoms, signs, and predisposing conditions of various red pine insects and diseases. At least initially, objects present on a decision-factors list should be reduced to the most general terms possible. Introducing more detail during this step forces the knowledge engineer to consider how the factors are used in decision making and how they will be represented later during implementation. Some implementation issue may need to be addressed at this stage, but others that surface cannot be resolved yet given the current stage of model development.

Once a list has been created it may contain some redundancies. As each object is labeled, it is given a definition to distinguish it from all other terms. While doing this, it is possible to identify terms that duplicate others.

When a list of reasonably unique terms has been created, each term is assigned a set of possible values that it may assume. Terms with numerical values may have some range of values that is reasonable; non-numeric terms may have discrete values such as high-medium-low or red-yellow-brown. In situations where the entities are more complex than single terms, i.e., each entity possesses several attributes, then each of these attributes can be assigned values. For example, we could describe facts about trees within a forest in a number of ways. Each different attribute of a tree may have a separate label, definition, and value, or, alternatively, if there are many characteristics of a typical tree that the KBS should be able to reason about, it would be more elegant and efficient to regard each tree as distinct, but each with a standard set of descriptors or attributes. This begins to sound very much like a frame-based representational structure; hence, rather than just describing entities, we've begun to consider how they will be used. Consequently, issues of knowledge representation begin to surface. Often different aspects of KBS design are intricately interwoven and ideally must be considered in parallel.

One final aspect of the knowledge-term vocabulary can be clarified during this listing process. Some terms constitute the basic data that describe a problem, for example, "tree diameters" or "forage species present." These objects are often referred to as *atomic* factors because they are elemental (not composite, i.e., unable to be further reduced to other terms) and observable. In general, all decision factors can be either observed directly (atomic) or derived from atomic factors (hypotheses). Some atomic factors are unusual in that they either can be observed or, if the necessary observation cannot be made, can also be derived from other atomic factors. For example, a tree can be observed as dead, or alternatively its "death" state can be inferred from the observations of dead shoots, buds, and needles. When a factor is derived from other atomic factors, it is often known with much less certainty than if it has been directly observed.

Factors, that represent collections of atomic facts in a conceptual manner are often called *hypotheses.* Hypotheses represent higher-level concepts that must be inferred and cannot be observed directly, e.g., "moisture stress," "stand stocking level," or "germination condition." Hypotheses and their formation are often very important in the problem-solving process. Hypotheses—sometimes also called *intermediate factors* — often serve to organize and consolidate information represented by atomic factors and other lower-level hypotheses into a solution-building and reasoning process.

At this stage of lexical analysis, it may be difficult to define a possible range of values for some of the intermediate hypotheses. Possible values for intermediate hypotheses are often dependent on how particular factors combine to infer these hypotheses. Consequently. some lexical knowledge analysis may need to be completed later after some knowledge structure (syntax) has been imposed on these factors and hypotheses.

Syntactic Knowledge Analysis

A list of terms without any structure conveys little knowledge about a domain. Just as words that don't adhere to any grammatical rules (syntax) result in poorly formed sentences, so too knowledge terms without any structure make it difficult to represent complex ideas. The second important aspect of knowledge analysis is the understanding of relationships among decision factors. Syntactic knowledge expands the number and types of concepts that can be represented, because it is possible to combine individual factors in different ways. Then knowledge begins to take on an aspect of depth, rather than the one-dimensional nature of knowledge terms.

Structure specification requires two properties: (1) a description of the relationship, i.e., type, and (2) the strength of the relationship, e.g., proximity between two terms. Relationship type is most important; strength becomes important when it is necessary to make finer judgments based upon a specified relationship. For example, two different insects that damage tree roots may both girdle large roots, which is very important *descriptive* information relating particular insects with tree symptoms. One of these insects, however, may girdle large roots more frequently or during certain times of the year. This additional information further describes the insect/symptom relationship by *qualifying* the association. The distinction between relationship type and relationship strength implies that for KBS development, type structures should be defined first and, if necessary, discriminate terms and their usage in decision making on the basis of strength of association or conceptual proximity.

Because relationships among several terms can sometimes indicate a new, additional concept (an intermediate hypothesis), structure analysis may help illuminate these aggregate concepts to fill in any omissions remaining from the lexical analysis stage. All knowledge pieces are now present for an analysis of semantic knowledge, the next step.

Semantic Knowledge Analysis

The third, and probably most difficult, aspect of knowledge analysis is the elicitation of problem-solving (tactical and strategic) knowledge. Eliciting and applying semantic knowledge to the knowledge acquired in lexical and syntactic analyses is at least as important as those earlier types of knowledge. Semantic knowledge is the applicative component of knowledge; that is, it tells us how to exercise our static—factor and structure—knowledge to make it work for us in solving problems.

Semantic knowledge typically includes a number of aspects. It can specify: (1) how to combine various pieces of static knowledge for reasoning; (2) when to use particular pieces of static knowledge; (3) what possible solutions or goals to pursue in what order; and (4) what type of search to conduct, e.g., certain rules sets may be evaluated in a particular order or a certain type of control strategy or inference method may be selected. The first use is tactical knowledge; i.e., it tells us *what* to do given those pieces of static knowledge. The other three uses are strategic knowledge in that they tell us *how* to apply our other knowledge (which may include the application of tactical knowledge). There may be other applications of semantic knowledge besides these, but they all function similarly in that they orchestrate how solutions are searched for, selected, and evaluated.

Semantic analysis seems difficult because an expert often has a better understanding of *what* knowledge is used for problem solving rather than *how* it is used. It is often quite readily apparent what specific pieces of knowledge are applied to a problem. To understand how that knowledge is actually used requires that an expert introspect about internal thought processes or retrospect about previously solved problems—activities with which most people have great difficulty. Although a large portion of factor and structural knowledge resides in the public arena, most problem-solving (semantic) knowledge is private to each individual specialist. Consequently, most acquisition methods for problem-solving knowledge acquisition employ indirect approaches to infer underlying strategies and tactics. Strategies surface from observed or proposed problem-solving behavior, possibly supplemented with explanatory discourse.

The following rule from a rule-based knowledge representation (Table 1) illustrates these different types of knowledge:

```
IF      percent_slope > 20
        AND percent_slope < 80
        AND burn_pattern = spotty
        AND percent_area_burned > 40
THEN    erodibility = moderate
```

Lexical knowledge analysis produces the various terms and the potential values that one can use to reason in the problem domain, e.g., the factors, *percent_slope* and *burn_pattern,* and their ranges of values. Syntactic knowledge analysis indicates that *percent_slope, burn_pattern,* and *percent_area_burned* are related in a causal way to post-fire *erodibility.* Therefore, a rule relating these factors will be useful for predicting *erodibility*. Semantic knowledge analysis indicates how those factors are related and what their relationship indicates about *erodibility* (tactical knowledge). That is, in this case, the conjunction of particular factor-value pairs implies a particular prediction for *erodibility*. Therefore, semantic knowledge provides applicative specifics for the lexical and syntactic knowledge. Other forms of knowledge representation will have slightly different examples for the different knowledge

Related Terminology

It is useful to examine how linguistic-based knowledge analysis compares with the varied (and often confusing) body of terminology introduced elsewhere. Table 2 identifies some of the relationships between lexical, syntactic, and semantic knowledge and other authors' descriptions. Most other delineations distinguish only between semantic knowledge and "the rest" (often called "declarative"). Knowledge-analysis terminology goes one step further by discriminating lexicons and the relationships between lexicons (syntax). While this may seem dubious from a taxonomic standpoint, this distinction helps guide the acquisition of knowledge because many KA methods may elicit specifically one or the other type of knowledge. Then, rather than just acquiring knowledge in a domain, the knowledge engineer explicitly elicits certain types of knowledge (using type-specific methods) that fulfill specific epistemological needs. This results in a more deliberate and rigorous knowledge acquisition process.

Table 1. *Each row of a knowledge table contains a set of values for each of the independent factors and a corresponding value for the dependent factor. This table format allows one to code rules directly from the entries of the table.*

Percent Slope	Burn Pattern	Percent Riparian Area Burned	Erodibility
0 - 20	?	?	low
20 - 80	mosaic	?	moderate
> 80	extensive	?	high
?	spotty	0 - 40	low
20 - 80	spotty	> 40	moderate
> 80	mosaic	> 20	high
> 80	mosaic	0 - 20	moderate
20 - 80	extensive	0 - 40	moderate
20 - 80	extensive	> 40	high

Table 2. *Previous knowledge-type terminology can be related to the three types of knowledge described in linguistic-based knowledge analysis. Categorization of knowledge acquisition methods within knowledge analysis appears in Table 4.*

Lexical Knowledge Syntactic Knowledge	Semantic Knowledge
Facts	heuristics
declarative knowledge	procedural knowledge
"deep" knowledge (first principles)	"surface" knowledge (compiled)
"public" knowledge[a]	"private" knowledge
"textbook" knowledge[a]	experiential knowledge
properties/attributes	methods/procedures
problem-specific knowledge	problem-solving knowledge
explanatory knowledge	deductive knowledge

[a] In some cases this may also include semantic knowledge.

Knowledge Acquisition Methods

Most of the KA techniques mentioned in the following pages aim to systematize an expert's knowledge into a form that is universal and, hence, less anecdotal. Anecdotal knowledge is often limited in application to situations that are very similar to the prior experiences that produced that knowledge. On the other hand, universal knowledge is more general and can be applied to a broader range of problems, many of which might only be vaguely similar to an expert's original experiences.

Certain acquisition methods do a better job of eliciting and organizing particular types of knowledge. Because the purpose here is to show how different KA methods apply to knowledge analysis, interested readers are urged to seek out the respective authorities listed in the references for additional details about particular KA methods. Figure 2 depicts a taxonomy of the various methods presented in the following sections.

Unstructured Interviews

Unstructured interviews are characterized largely by a lack of organization. A knowledge engineer (interviewer) and expert sit down and the expert responds to questions posed by the interviewer. Questions follow no preset or designed format and topics are pursued in whatever order or at whatever length seems best to the knowledge engineer. Successive interviews tend to expand on topics covered in prior sessions. Unstructured interviews often become the primary method of elicitation employed to develop a quick prototype for a feasibility study.

These types of interviews maybe useful during initial stages of KA. Their lack of structure permits a sort of free association dialogue that may illuminate many of the major issues that are important, but they also have some drawbacks. They are by nature unstructured and, hence, may be very inefficient at collecting knowledge because of redundancies and omissions. To work effectively, the interviewer must be a very skilled communicator and have an ability to identify key issues through incisive questioning. This level of interviewer talent implies, instead, some unconscious systematic methodology by the interviewer, which means that he or she may, in essence, be conducting a structured interview.

Because of the broad nature of unstructured interviews, it is possible to elicit lexical, syntactic, and semantic knowledge. The loose structure of this technique, however, makes it difficult for the knowledge engineer to know how completely each of the three types of knowledge has been acquired. Except as an ice-breaker technique, unstructured interviews cannot be strongly recommended as a valuable or effective elicitation method.

Structured Interviews

Within the general realm of structured interviews, a large number of methods may be utilized (Gordon 1989). A fundamental intent of these methods is to provide the expert with guidance for his or her responses, thereby increasing interview efficiency. Explicit guidance allows an expert to focus on the subject matter rather than on how responses should be formatted. The following paragraphs briefly describe a number of these techniques.

Free Association

Anderson (1983) proposed a model of human memory called *spreading activation theory.* This theory proposes that whenever one thinks of a particular concept, all concepts that are closely associated with that initial concept can very easily be re-

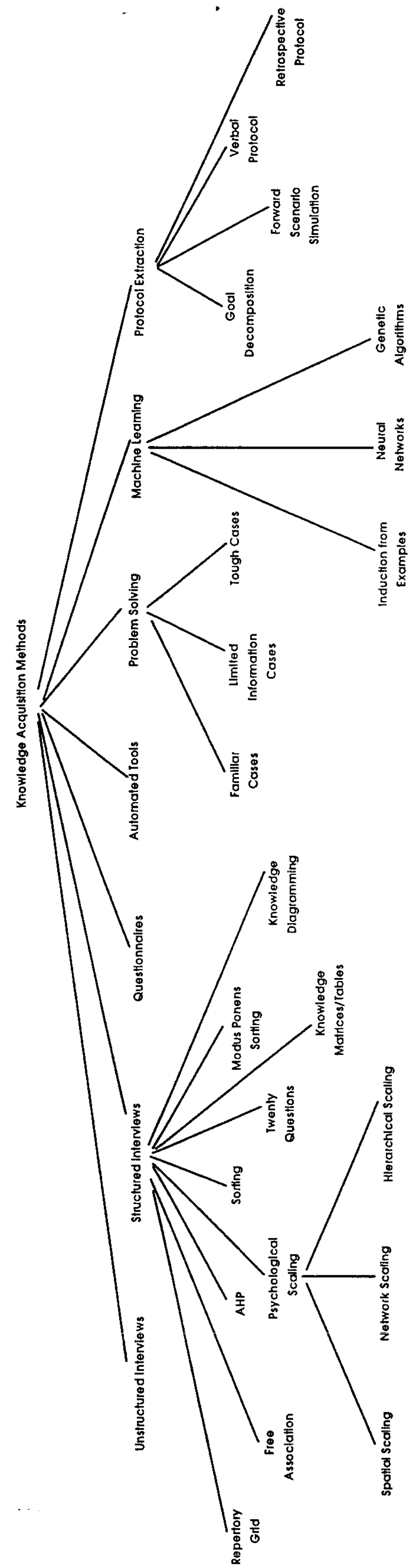

Figure 2. *A taxonomy of knowledge acquisition methods identifies the major acquisition categories and also some particular techniques within each category.*

called (activated). On the basis of this theory, Mitchell (1987) suggests a *free association* KA technique. Various key concepts in the subject area are posed to an expert one at a time and he or she is asked to respond with other concepts that are in any way related to each of them. It then becomes possible for the knowledge engineer to construct a matrix or graphical road map of terms/concepts in the subject area.

An example of such a terminology graph appears in Figure 3. Three defoliators of red pine are linked with terms that are related to these insects. The knowledge engineer might ask an expert to respond with all pest diagnosis terms that are related to these three insects. This method elicits not only concepts that are part of the general theory of a subject area, but also those things that are idiosyncratic and, hence, private to an expert's knowledge (Mitchell 1987).

Obviously, free association generates decision factors, but it also identifies which ones are related to which other ones. Free association does not, however, elicit either the type of relationship (which is very important) or how closely pairs of factors are related so syntactic knowledge is only partially uncovered.

Psychological Scaling

Results from the use of free association could be subsequently enhanced by one of several *psychological scaling* techniques. Cooke and McDonald (1987) discuss a number of scaling methods. Proponents of scaling methods claim that in addition to associations among concepts in memory there also exist strengths of association. These strengths have some relation to cognitive proximity. So the stronger an association is, the closer, or more related, two concepts are. Whereas free association can provide a graph or matrix of all concepts and whether they are related, scaling methods go one step further and require that the relationships be assigned some value from a numerical scale to indicate relatedness. This association matrix (or dis-

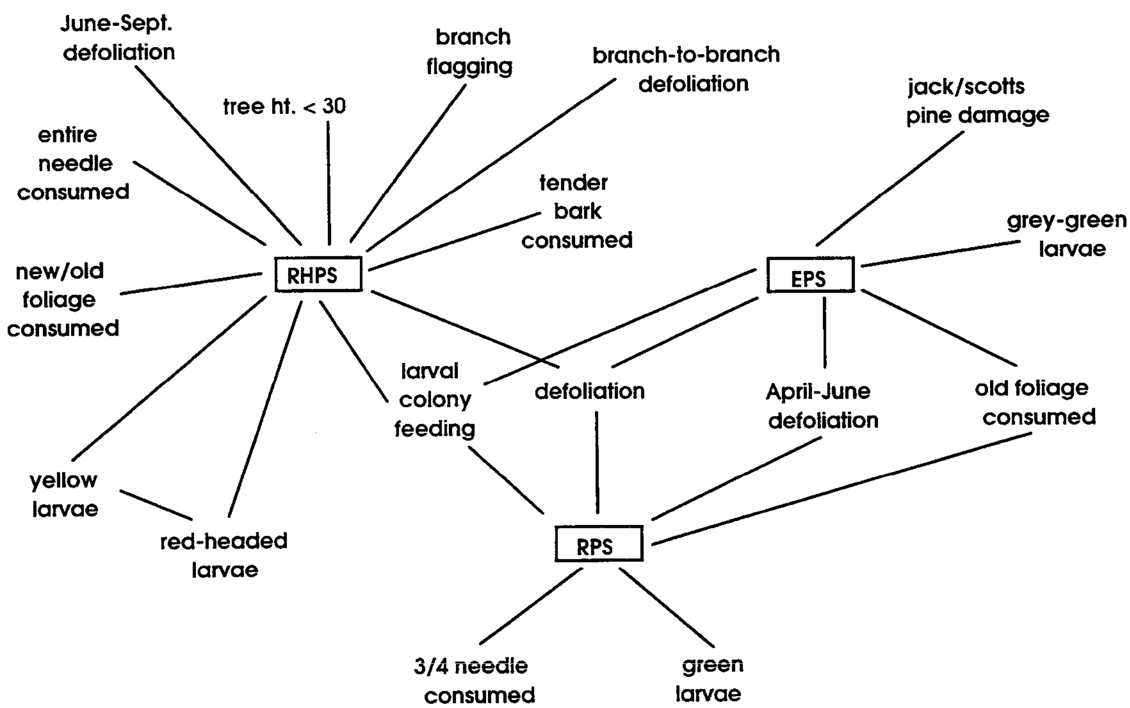

Figure 3. *Free association by a subject matter expert can identify many of the important concepts within his or her knowledge base. Associations among these concepts can then be represented graphically or in a tabular manner using a matrix. RPS, EPS, and RHPS refer to red pine sawfly, European pine sawfly, and red-headed pine sawfly, respectively.*

tance matrix) can then be transformed into a spatial representation (using multidimensional scaling, Shepard 1962), a hierarchical representation (using cluster analysis, Johnson 1967), or a network representation (e.g., Cooke and McDonald 1987) of an expert's knowledge in a subject area. The graph in Figure 3 can be converted into a matrix with numerical values that represent closeness (Figure 4). This representational transformation accentuates any lexical analysis information and also elucidates knowledge structure.

Anthropological Methods

In addition to psychology research, a number of useful elicitation techniques have also been developed in anthropology (Benfur and Furbee 1989). To understand the language and thought patterns of vastly dissimilar cultures, anthropologists have been forced to develop numerous methods to describe the acquisition of personal/cultural knowledge—knowledge that is quite difficult for indigenous peoples to impart to others from outside their particular culture. Such cultural roadblocks are analogous to the situation of a domain expert and his or her subject matter specialty, which can also be viewed as a culture of sorts. Many of these methods are detailed in a book by Brokenshaw et al. (1980).

Sorting. One technique, referred to as *sorting,* uses a stack of cards, each with a term/concept from the domain written on it. The process of generating these terms constitutes factor knowledge analysis. An expert is instructed to sort the cards into different piles using whatever criteria seem appropriate. Following a sort, the expert is asked to provide a verbal description of the sort criteria used. When the cards contain only atomic decision factors and intermediate hypotheses, then the subject matter expert can group cards to indicate composite terms, i.e., additional intermediate hypotheses, or can group cards based on concept proximity. In this manner, some lexical knowledge analysis occurs after the original cards have been created, but in most cases the act of sorting produces syntactic knowledge only. When the cards contain different possible solutions only and no intermediate or atomic factors, this method works very similarly to the repertory grid technique (see below). By organizing concepts into "similar" stacks using different sorting criteria on each pass, the expert is essentially performing a psychological scaling exercise identical to that described above (except perhaps more interactive in this case). With each pass the expert is scaling the concepts along a different dimension.

Twenty Questions. Twenty questions is a game familiar to most readers. The knowledge engineer collects several task scenarios prior to the interview. Then, during the interview, the expert is instructed to ask questions about a particular task much like would transpire in an actual problem-solving situation. The expert also supplies rationale for asking each question. Our defoliator example from above is used to illustrate the 20 questions technique in the dialog excerpt of Figure 5.

The questions that the expert asks and the rationale provided for that questioning can elicit de-

	green larvae	yellow larvae	red-headed	new/old foliage	old foliage	entire needle	branch flagging	grey-green larvae	bark consumed	3/4 needle consumed
green larvae	0									
yellow larvae		0								
red-headed			0							
new/old foliage				0						
old foliage					0					
entire needle						0				
branch flagging	8	3	3	2	8	2	0			
grey-green larvae	2	6	8	9	4	3	8	0		
bark consumed							1	9	0	
3/4 needle consumed							9	7		0

Figure 4. *By creating a matrix of pairwise relatedness values (distances), one can extend the graphical representation in Figure 3 produced by free association. Ten concepts were selected from the free-association graph and values (0-9) have been entered for two rows of the matrix. Because the matrix is symmetrical, it can be displayed as upper or lower triangular Distances between concepts can be used to calculate clusters of similar concepts.*

Where is the stand located? (*Certain insects are limited to particular geographic areas):* **Portage county**

How large are the frees? (*Some insects only affect trees of a certain height.):* **25 ft**

What type of damage has occurred? *(Gross symptoms can indicate certain classes of pests.):* **Defoliation**

Are new needles or old needles or both affected? *(Different defoliators consume either old needles or both new and old):* **New**

When did defoliation occur? *(The timing of defoliation can be very diagnostic):* **June**

Are larvae present. if so what color are they? (*A good larval description can identify a particular defoliator absolutely):* **No**

Figure 5. *When example test cases are available for knowledge acquisition, a subject matter expert can be asked to solve particular cases by having the expert ask probing questions and provide a rationale for each such question. This is often referred to as the 20 questions technique, similar to the game of the same name.*

cision factors, their relationships, and how those factors are used in problem solving. Therefore, this technique is able to elicit all three types of knowledge. While the 20-question format is very natural for the expert, the knowledge engineer is forced to record session dialog and analyze the transcripts later for knowledge content.

Modus-Ponens Sorting. In modus ponens sorting—referred to as a "structured interview" by Schweickert et al. (1987)—an expert is asked to list factors important to making a decision in the subject area and also to list all possible outcomes (final decisions). Then an expert must connect, in the form of if-then rules, factors to each other, outcomes to each other, and factors to outcomes. This is a sorting method based on if-then relationships, hence the use of the term *modus ponens* sorting. This technique includes lexical analysis (lists of terms are created), syntactic analysis (factors are linked to each other), and semantic analysis (factors are linked to outcomes, i.e., decisions, via tactical rules).

If the number of factors pertinent to each solution is small, it may be possible for the knowledge engineer to construct all "reasonable" combinations of factors for each solution and present these to an expert for critiquing; Schmoldt (1987) used this method. When Schweickert et al. (1987) compared sorting, 20 questions, and modus ponens sorting for producing if-then rules, they found modus ponens sorting and 20 questions superior to sorting. Their sorting method, however, was not formulated to explicitly create if-then rules as part of the interview process; hence, it exhibited a poor showing in their comparison.

Knowledge Diagraming

Graphing domain concepts and the relationships between those concepts, as revealed by an expert, can be a useful visual aid. This *knowledge diagram* creates a sort of road map of an individual's cognitive structures. Then, in subsequent interviews, the interviewer and expert have a record of where they have been and what topics may need to be expanded further. Figure 3 can be expanded into a knowledge diagram by adding connecting phrases to the links and by expanding and grouping various concepts (Figure 6). For example, the relationship between "larval colonies" and "defoliation" could include the connector <cause>, as in "larval colonies" <cause> "defoliation." Also, the different types of needle consumption, e.g., "entire needle," "¾ths of needle," "branch to branch," might be grouped under the concept "needle consumption." Explicit labeling of terms and their relationships creates both lexical and syntactic knowledge.

Knowledge diagraming has also been used with question probes by Gordon and Gill (1989) and Graesser and Clark (1985), and also by Schmoldt and Bradshaw (1988) as part of multi-expert elicitation. Actual diagram creation is highly recommended even if the knowledge diagram is not utilized in successive interview sessions. The activity of creating a diagram helps the knowledge engineer visualize and better understand the domain's knowledge structure.

Repertory Grid

The *repertory grid* technique originated with Kelly (1955) and his theory of personal constructs. This theory posits that each of us operates somewhat like a "personal scientist," i.e., we attempt to organize, predict, and control our own world by categorizing and classifying our experiences. This is similar to a scientist who develops and tests theo-

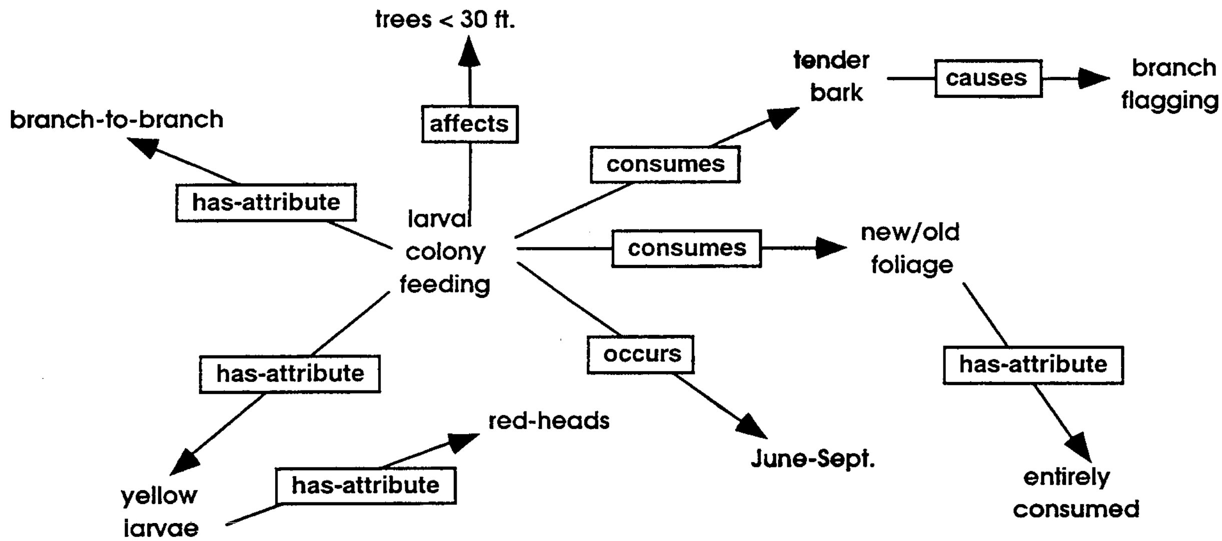

Figure 6. *A portion of the free-association graph of Figure 3 appears here with different concepts connected by labeled links. This graph provides more structure and more detail than the free-association graph.*

ries about the physical world. A repertory grid is a clinical technique that Kelly developed to identify and analyze personal constructs, i.e., mental models. Boose (1985, 1986) automated and applied this technique to the acquisition of expert knowledge. Boose's process attempts to fill in a matrix, where each column corresponds to an element that is to be discriminated (i.e., final solutions), and each row represents a personal construct (decision factor) that differs across several of the column elements. Each construct has a diametric description, e.g., short/tall or good/bad. Then each column element (solution) can be assigned a value for each construct (decision factor) that indicates the association between it and each diametric construct.

Figure 7 contains part of a repertory grid for our defoliator example. Each element receives a score in the range 1-5 depending on how much of the construct it possesses. Many of the constructs for this example, present in Figures 3 and 6, do not necessarily have a natural range of values, but rather have discrete values, e.g., the location of the stand or the description of the larvae. These types of constructs are not easily incorporated into the repertory grid and must be added to the knowledge representation later.

The repertory grid has the nice advantage of forcing an expert to create a tabular representation of his or her internal concepts about a subject area. Decision factors are enumerated and discrimination rules are created as the grid is constructed. Then the grid can be used for syntactic analysis of knowledge structure or for semantic analysis and actual rule construction (Boose 1986). Boose (1986) notes, however, that it is limited in application to declarative types of knowledge (*declarative* knowledge, in this case, includes tactical knowledge). Procedural, strategic, and causal knowledge are difficult to represent with this technique.

Knowledge Matrices/Tables

A technique that is very similar to the repertory grid is the use of *knowledge matrices* or, equivalently, *knowledge tables.* Knowledge matrices/tables allow the knowledge engineer to explicitly describe the associations between a particular hypothesis and all the factors upon which it depends. Various com-

Elements			Constructs
RHPS	RPS	EPS	
2	5	4	new/old foliage
2	4	4	spring/summer
5	4	2	% needle consumption
3	1	1	no flagging/flagging

Figure 7. *The repertory grid technique can effectively discriminate among a set of elements (solutions) based on various bipolar constructs. All the solutions are scored for each construct on some arbitrary scale, e.g., 1-5.*

binations of values for the factors are associated with particular values of the hypothesis. These associations can then be translated directly into if-then rules, if desired.

Most hypotheses are established by the interaction of several different factors. For example, a hypothesis, *Y,* maybe inferred from the values of factors X_1, X_2, and X_3. Using the hypothesis and factors, a matrix can be constructed to ensure that the correct relationships have been specified and that all possible combinations of the factors have been accounted for (Braun 1989). The entries in the matrix represent the values of *Y;* the row and column headings correspond to particular values of the *X's.*

Table 3 contains a more concrete example. Here the hypothesis we are interested in is *slope erodibility* of a tract of land following a wildfire. The factors identified as influencing erodibility are *percent slope, percent of riparian area burned,* and the *pattern of the burn* in the watershed. This matrix representation allows us to easily see what the relationships among all pertinent factors and *erodibility* are.

An equivalent, and possibly more useful, representation of the knowledge matrix in Table 3 ap-

Table 3. *A knowledge matrix creates an alternative representation to the knowledge table. This knowledge matrix displays the relationships between three independent factors and their impact on a dependent factor, erodibility. The values, "low", "moderate", and "high" are possible erodibility values.*

		Percent Riparian Area Burned			
Erodibility		**0-20**	**20-40**	**> 40**	**Burn Pattern**
	0-20	low	low	low	
Slope	20-80	low	low	moderate	spotty
	> 80	low	low	moderate	
	0-20	low	low	low	
Slope	20-80	moderate	moderate	moderate	mosaic
	> 80	moderate	high	high	
	0-20	low	low	low	
Slope	20-80	moderate	moderate	high	extensive
	> 80	high	high	high	

pears in Table 1. This knowledge *table* contains a column for each of the independent factors *(percent slope, percent riparian area burned,* and *burn pattern)* and one for the dependent factor *(erodibility).* There is one row for each possible combination of values for the independent factors, where the dependent factor entry for that row is the logical conjunction of the values for the independent factors. Although it would be difficult to include more than four factors in a knowledge matrix (one on each side and top/bottom), in a knowledge table there is no similar organizational dilemma because each factor is contained in a separate column.

In this particular case, the total number of rows needed would be 3 x 3 x 3 = 27. While this may seem like a more verbose way to represent the same contents as Table 3, it is really not that onerous. For example, in Table 3, notice that when *slope* = 0-20%, *erodibility* is *low* regardless of the values of the other two factors. Table 1 represents this knowledge by using a "?" in the columns corresponding to *percent riparian vegetation burned* and *burn pattern* to indicate "we don't care what the values of those factors are, it's unimportant." Therefore, this one row contains knowledge that would otherwise be spread over nine rows (3 x 3). As a result of this and other shortcuts, a total of nine rows in Table 1 covers all 27 cases. The real advantage of the representation in Table 1 is that if-then rules can immediately be written to correspond to each row of the table. For the example above, we could write the concise rule:

IF 0 ≤ *percent_slope* ≥ 20 THEN *erodibility* = low.

By specifying decision factors and the values of factors that imply values of other hypotheses, all three types of knowledge are acquired and recorded.

Analytic Hierarchy Process

Another aid to the analysis of a decision process is the *analytic hierarchy process* (AHP) developed by Saaty (1980). It allows persons with decision-making expertise to structure a complex problem in the form of a hierarchy. The process requires an ability to enumerate all possible decisions, i.e., alternative solutions, a priori. Then criteria are established to evaluate those decisions. Likewise, there may also be criteria to evaluate each of the previous criteria. This forms a hierarchy (Figure 8).

At each level, pair-wise comparisons are made regarding the relative likelihood, relative preference, or the relative importance of each criterion versus each of the other criteria at the same level. For our example, erosion hazard rating, establishment likelihood, political/social impacts, and downstream values would be compared in a pairwise

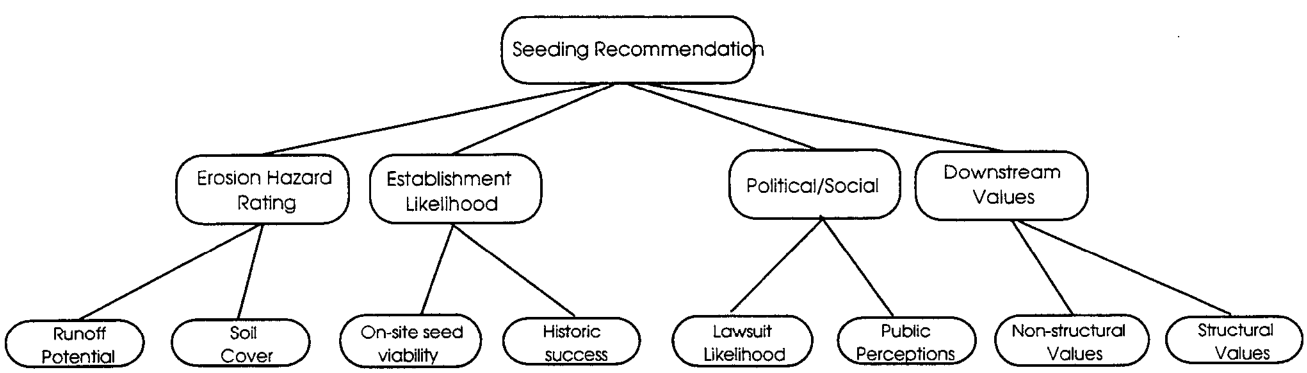

Figure 8. *The analytic hierarchy process (AHP) dissects a decision process into a hierarchy of criteria, where each criterion may have subordinate criteria. At each level of the hierarchy, pairwise comparisons are used to generate a priority value for each of the criteria at each level. A finite and known set of decisions can then be compared by scoring each leaf node decision with respect to preceding criteria and then multiplying those values by successive priority values between the leaf node and the root node.*

manner. Similarly, runoff potential and soil cover would be compared, as would on-site seed viability and historic success, etc.

In the AHP, these comparisons can then be converted into numbers that represent each criterion's contribution to the overall decision. At each of the leaf nodes, a score is recorded by an expert for each possible decision with respect to the criteria immediately above that node. Suppose that, in our example, the two possible decisions are *seed* or *do not seed.* Then both of these decisions are scored according to their impact on runoff potential and soil cover (the two lowest-level criteria). The score for each, *seed* and *do not seed,* is then multiplied by the priority value for each of the criteria above, runoff potential and soil cover. Each of these two scores is then multiplied by the priority value for the next higher level node, i.e., erosion hazard rating. A similar process proceeds from each set of leaf nodes up to the root node, where the scores for each possible decision are summed across the top-level criteria. The process of combining and propagating scores up through the hierarchy to the root node results in an overall likelihood, preference, or importance, score for both of our possible decisions, *seed* or *don't seed.*

For KA purposes, it would probably not be useful to duplicate exactly this behavioral model of decision making in a KBS. The AHP produces a decision structure that is valid *at one point in time* and *for a particular instance.* This general analytic process, however, does provide a great deal of knowledge about how an expert might structure and think about a class of problems. The primary advantage of this analytic method is that it provides a systematic and detailed description of an expert's decision-making criteria and each criterion's relative contribution in the decision process. Unless the AHP that is developed is used as the actual decision model, the AHP does not elicit any general-purpose semantic knowledge.

Several drawbacks to this approach are immediately apparent. First, not all decision-making situations can be structured in a hierarchical manner. Second, it is not clear what to do when certain criteria are not known in an actual problem instance. Third, priority values in the AHP are static; however, in many real-world situations the relative contribution of different factors may vary over time or with different users and their specific problems.

Questionnaires

Occasionally, it may be difficult to physically meet with an expert or it may be necessary to extract very detailed and specific knowledge about some topic. *Questionnaires* can be quite useful in both circumstances (Olson and Rueter 1987, Schmoldt and Peterson 1991). Also, when several experts are to be interviewed, the time and effort of the knowledge engineer can be reduced if questionnaires can be used in place of face-to-face meetings. This situation was encountered with the KBS development effort fo seeding recommendations (Figure 8). The content of questionnaires may be very specific and require only short answers or they may be more general and intended to elicit longer prose. Multiple-choice or short-answer questions can often help an expert focus on the specific details that a questionnaire is designed to address. It may also be desirable to allow an expert the flexibility to add comments for explanation or emphasis. Schmoldt (1987) successfully used short-answer questionnaires to verify and augment lists of factors related to pest diagnosis; these factors were categorized as either predisposing conditions, symptoms, or signs. Using categories of lists helps an expert narrow his or her attention to a specific task with specific types of output. Schmoldt and Peterson (1991) used tables and short-answer questionnaires to elicit knowledge about air pollution impacts in wilderness areas from a large group of scientists and land managers. These questionnaires were completed in a workshop setting where: (1) questionnaires provided a focus for discussion and (2) their presence ensured that participants were aware of tasks to be completed in the time allotted. By using other methods in conjunction with questionnaires, it maybe possible to elicit semantic knowledge, but, in general, only factors and their relationship are acquired.

Protocol Extraction

Encouraging an expert to explicitly detail how either typical or specific problems are solved can elucidate many specifics of his or her decision-making process. Explication of an expert's problem-solving *protocol* identifies factors that are important, relationships among them, hypotheses that can be inferred, and strategies of how and when these factors are applied. A number of different protocol extraction methods have been utilized.

Goal Decomposition

Goal decomposition is one of the most basic methods of formalizing problem-solving strategies. An expert is asked to enumerate the steps to be followed (subgoals) as a problem is solved. A particular problem scenario may be used as an example, or general types of problems maybe discussed more in an abstract manner. These subgoals can then be used in recursive goal decomposition until the subgoals become fairly simple and readily accomplished tasks. This approach may be used in combination with knowledge diagraming or other methods presented above.

Forward Scenario Simulation

Another method, *forward scenario simulation* (Grover 1983), is almost identical to goal decomposition but prompts an expert for additional information, such as decision factors and explanations, in addition to subgoals. It is referred to as a simulation because an expert does not actually solve a problem scenario, but only describes how it might be done (Gordon 1989). Forward scenario simulation represents a very general protocol method because it elicits both decision factors and problem-solving strategies.

Verbal Protocol

In *verbal protocol*, an expert is asked to solve a particular problem in the domain and to verbalize his or her rationale at the same time. Hoffman (1987) suggests the use of three different types of tasks for verbal protocol: (1) typical and familiar tasks, (2) limited information tasks, and (3) rare or tough cases. Gordon (1989) mentions a variant of typical verbal protocol, *retrospective protocol*, in which the actual problem solving and explanation portions of the protocol are performed separately. That is, an expert first solves a particular problem and then reflects on the methods and rationale used for problem solving. The technique in the following section also requires that an expert solve example cases, but without any accompanying explanatory rationale.

Observed Problem Solving

An expert may work differently when he or she is not required to justify problem-solving steps. To avoid overly self-conscious and, hence, unnatural decision-making situations, an expert maybe asked to solve problems without providing explanations. *Observed problem solving* may occur either in its natural environment, i.e., on the job, or in an artificial situation. Different types of tasks, i.e., familiar cases, limited information cases, and tough cases, may also be used with this method. Because problem-solving steps are not made explicit by the expert when using this KA method, the knowledge engineer must infer implicit strategies that are employed to solve various types of problems (Gordon 1989).

Machine Learning

Most of the extraction methods described in the previous sections collect knowledge from experts. But, as noted earlier, some knowledge also exists in real-world examples. That is, much can be inferred from prior situations and then applied to new or similar circumstances. Machine learning methods are automated techniques for discerning patterns in data sets. In the three methods presented below, a decision procedure—e.g., decision tree or decision function—results from learning.

Machine Induction

Machine induction methods attempt to discern patterns present in collections of decision factors and their corresponding solutions (e.g., Langley and Carbonell 1984, Michalski et al. 1986). Many of these programs attempt to induce rules from examples using an algorithm similar to the ID3 algorithm of Quinlan (1983). This bottom-up approach creates general and universal associations from specific problems and their attributes. The result is often a discriminating hierarchy similar to a decision tree. Like empirical relationships derived using traditional statistical approaches (such as least-squares regression), however, decision rules resulting from machine induction are very sensitive to, and only applicable to, the range of problems from which they were developed. This implies that examples used for learning must be selected very carefully (Hart 1986). Julien et al. (1992) review a number of machine induction techniques and Jeffers (1991) reviews of number of machine induction software packages.

Clustering methods, on the other hand, do not use any training set. They attempt to organize similar individuals into groups, or clusters, that are alike in some (possibly more than one) way. By examining how individuals align themselves into clusters and what attributes of individuals are important for each cluster, it is possible to understand more about the population of individuals. Often, individuals are labeled as similar to other individuals if they are close to each other with respect to some metric (e.g., Michalski and Stepp 1983). Others (e.g., Matthews and Hearne 1991) have used nonmetric clustering, where clusters are formed based on some characteristic of the clusters and not on proximity among individuals. These techniques have also been referred to as *conceptual* clustering because the attributes used for clustering are not necessarily arithmetic.

Artificial Neural Networks

Another approach to machine learning uses *artificial neural network* (ANN) structures to represent the association between decision factors and possible solutions. The first description of the operation of an ANN was probably given by Hebb (1949). They were studied extensively in a theoretical fashion by Minsky and Papert (1969) and received a thorough empirical treatment in Rumelhart and McClelland (1986) and McClelland and Rumelhart (1986). An ANN consists of one or more layers of simple processing units. Each processing unit is connected to one or more processing units or input variables or output solutions. The output of any processing unit is determined by the inputs of its connections and the activation function that it uses to combine inputs. A network is then exposed to a variety of paired input and output combinations that constitute a training set. If network output does not agree with expected output from the training set pair, a learning algorithm modifies the strength values associated with processing unit connections. Different network architectures, different learning algorithms, and different activation functions cause networks to exhibit vastly different properties.

Machine induction and ANNs possess several similar characteristics. Both are dependent on the quality and quantity of examples used for system construction. Each requires that decision factors are selected a priori; this may necessitate the use of other KA methods initially. Certain decision factors may be eliminated later, however, if they are found to be redundant or ineffective. Syntactic knowledge, represented as relationships among concepts in the domain, remains hidden in the discrimination tree or in processing unit connections. Semantic knowledge can possibly be inferred from a discrimination tree but it has no direct representation in a neural network. Nevertheless. there have been some efforts, e.g., Narazaki et al. (1996), to generate classification rules from ANNs. Machine learning can often produce very applicative systems, but they are opaque with respect to providing explicit accounts of domain knowledge.

Genetic Algorithms

A third machine learning approach is *genetic algorithms* (Goldberg 1989). This technique borrows ideas from traditional biological evolution theory. In a population of individuals, those with the “best” score, along some measure of usefulness or fitness, survive and reproduce. Offspring are better adapted because they contain the best attributes of their parents. These new genotypes possess combinations of attributes that were not previously present in any other individuals. In this sense, the genetic system has produced, or “learned,” a new and unique individual.

When applied to a machine learning task, genetic algorithms are generally constructed as classifier systems. That is, a genetic-based machine learning system attempts to create a set of classifiers that are useful for describing a particular data set. A genetic algorithm begins with a set (population) of candidate classifiers. In most implementations of genetic algorithms for machine learning, classifiers are represented by strings of 0’s and 1’s. Classifiers that are useful, i.e., successfully describe some aspect of the data set, receive a payoff that determines how well they perform in their environment in the future. Members of this classifier population reproduce in the next cycle proportional to their “usefulness” score. After reproduction, patterns of 0’s and 1’s in members of the new population cross over, i.e., two population members contribute a portion of their pattern to produce a new member (classifier) of the population. In this way, new classifiers are created from the best classifiers of the previous population. A random mutation step is also applied to ensure that populations do not stagnate (a random 0 or 1 is changed with a very low frequency). Over several generations, populations produce individuals (classifiers) that become more "useful" in describing a data set than individuals of previous populations.

Genetic-based machine learning can, therefore, derive new classifiers that were not present in the original set of classifiers (often a randomly generated set). As with the other two machine learning methods, decision factors must be selected a priori. Because input factors and the population of classifiers are encoded as 0's and 1's, however, any syntactic or semantic knowledge is completely obscured.

Automated Tools

Various computer programs have been developed to perform many of the interview tasks described above, thereby relieving a knowledge engineer of some of those chores. These automated KA tools, such as ETS (Boose 1986), lead an expert through various on-line exercises to identify decision factors, knowledge structures, and problem-solving strategies. Other tools, such as AKT (walker et al. 1995) and NetWeaver[1], lack a formal interviewing facility, but provide for recording, organizing, and executing knowledge bases. They can be viewed as knowledge base processors, analogous in intent to word processors. Some of these tools even create an operational system directly from interviews. Many rely on graphical aides to help an expert visualize his or her knowledge structures. Several such tools are available commercially, but most have been developed and are used in-house or represent academic research systems. Gordon (1989) provides a brief survey of several of the tools available.

Summary and Discussion

Acquiring the necessary knowledge for KBS development is a difficult task because of the abstract and complex nature of knowledge, especially private, human knowledge. Performing a good job during this phase of system development is essential, however, to produce an adequate knowledge model. This model will later be implemented as computer code in a KBS. A poor job of KA will produce a faulty knowledge model, which will, in turn, result in a faulty KBS.

Linguistic-based knowledge analysis involves a careful examination of the knowledge applied in a problem domain and is intended to guide knowledge acquisition. The knowledge analysis process is analogous to lexical, syntactic, and semantic analysis of natural or formal languages. The conceptual basis provided by the knowledge analysis framework can help a knowledge engineer systematically acquire and understand expertise in an unfamiliar domain. Because knowledge analysis subdivides the KA task into smaller and logically connected subtasks, it allows the knowledge engineer to select various KA methods that are appropriate to the separate steps, and still to combine their results to develop a knowledge model. Used in combination with a variety of KA methods, knowledge analysis can increase the completeness and reliability of the knowledge model created.

Knowledge acquisition methods have been developed in many different disciplines, such as decision theory, psychology, management science, computer science, and anthropology. Each method has its own strengths and weaknesses, and is variously effective for acquiring certain types of knowledge. Consequently, it is best to apply several different methods to obtain a complete picture of any subject area and to find effective methods for reasoning about that knowledge.

From the descriptions of acquisition methods above we can extrapolate and suggest some methods that may be particularly useful for each of the three knowledge analysis steps. Table 4 summarizes the application of particular acquisition methods to the three components of a domain "language." Several techniques, such as free association, modus ponens sorting, repertory grid, analytic hierarchy process, questionnaires, knowledge tables/matrices, and automated tools, specifically request decision factor information. Others, such as 20 questions, unstructured interviews, protocol extraction methods, and machine learning methods, often extract decision factors as part of their acquisition process. Literature and examples are also very useful for identifying factors and should be considered as a first source. Because a knowledge engineer will want to review the available literature for his or her own background education in the subject area, ini-

[1] Michael Saunders, Dept. of Entomology, Pennsylvania State University.

tial acquisition via literature can occur simultaneously with preparation for interview sessions.

Most of the acquisition methods that elicit syntactic knowledge do so on the basis of relationship type. Sorting, modus ponens sorting, knowledge diagraming, repertory grid, questionnaires, and automated tools usually attempt to explicitly identify and label relationship types. Using a hierarchical structure and priority values, the analytic hierarchy process elicits both the type and the strength of relationship knowledge. In the process of completing knowledge matrices/tables it becomes necessary to stipulate explicitly which decision factors are related to which hypotheses and what combinations of factors are associated in particular ways. Unstructured interviews, protocol extraction methods, and 20 questions illuminate structure less directly and often less completely. Psychological scaling methods, however, highlight the proximity between terms, rather than actual types of relationships that exist.

Several acquisition methods utilize a problem-solving scenario. To facilitate problem solving, an expert is asked to consider either a generic example problem or a specific problem instance. While solving a problem, the expert provides explanations, as in 20 questions and protocol extraction. Retrospective protocol splits the activities of problem solving and exposition into two separate tasks. Problem solving and forward scenario simulation eliminate exposition altogether, where in the former an expert solves a problem and in the latter only problem-solving steps are elicited. When using knowledge matrices/tables, it may be useful to have the expert consider past problem-solving scenarios to fill in entries in the matrix. Goal decomposition also extracts various components of solving a problem without actually asking an expert to solve one. In the analytic hierarchy process, criteria and their priorities explicitly specified in a hierarchical structure precisely define a decision-making strategy; in a sense, the AHP is the strategy. Induction from examples empirically creates tactical semantic knowledge from previously solved problems. Semantic knowledge may crystallize during modus ponens sorting, unstructured interviews, knowledge diagraming, and automated tool usage, but it results from verbalization rather than from performance.

From Table 4 it is apparent that several KA methods can be applied to all three components of a knowledge analysis. Although certain methods are more effective for particular analyses, the knowledge engineer can carefully apply one of several methods and, in so doing, extract all three forms of knowledge to some extent.

While reports of KBS development activities often acknowledge the difficult task of knowledge acquisition, in many cases less effort has been expended to share KA ideas and to suggest new and effective approaches to the task. The level of activity devoted to understanding how to effectively capture human knowledge has lagged behind efforts in hardware and software implementation. This knowledge analysis approach provides a framework to organize existing KA methods and outlines a theoretical basis for developing new ideas and techniques. By developing lexicons, syntax, and semantics for a domain, it becomes possible to converse unambiguously and knowledgeably about that subject area.

Table 4. *Many different knowledge acquisition methods can be applied to each of the three components of linguistic-based knowledge analysis. Several methods can be applied to all three aspects of knowledge analysis.*

	Type of Knowledge		
Acquisition Method	**Lexical**	**Syntactic**	**Semantic**
Unstructured Interview	×	×	×
Free Association	×	×	
Psychological Scaling		×	
Sorting	×	×	
20 Questions	×	×	×
Modus Ponens Sorting	×	×	×
Knowledge Diagramming	×	×	
Repertory Grid	×	×	
Knowledge Matrices	×	×	×
AHP	×	×	
Questionnaires	×	×	
Protocol Extraction[a]	×	×	×
Goal Decomposition			×
Problem Solving			×
Machine Learning	×		×
Automated Tools	×	×	×

[a] Includes forward scenario simulation and verbal protocol.

Acknowledgment

The author gratefully recognizes the valuable contribution of an anonymous reviewer for the journal, who pointed out several shortcomings of the original manuscript—in particular, the need to position *knowledge analysis* within existing knowledge acquisition terminology.

References

Anderson, J.R. 1983. A spreading activation theory of memory. Journal of Verbal Learning and Verbal Behavior 22: 261-295.

Benfur, R. A., and L. Furbee. 1989. Knowledge acquisition in the Peruvian Andes. AI Expert 4(11): 22-29.

Boose, J.H. 1985. A knowledge acquisition program for expert systems based on personal construer psychology. International Journal of Man-Machine Studies 23: 495-525.

Boose, J.H. 1986. Expertise Transfer for Expert System Design. Elsevier, New York.

Braun, R. 1989. Expert system tools for knowledge analysis. AI Expert 4(10): 22-29.

Bridges, S.M., C. Higginbotham, J.M. McKinion, and J.E. Hodges. 1995. Fuzzy descriptor of time-varying data: Theory and applications. AI Applications 9(2): 1-14.

Brokenshaw, D.M. Warren, and O. Werner. 1980. Indigenous Knowledge Systems and Development. University Press of America, New York.

Clarke, N.D., J.A. Stone, and T.J. Vyn. 1990. Conservation tillage expert system for southwestern Ontario: Multiple experts and decision techniques. AI Applications 4(2): 78-84.

Cooke, N. M., and J.E. McDonald. 1987. The application of psychological scaling techniques to knowledge elicitation for knowledge-based systems. International Journal of Man-Machine Studies 26: 533-550.

DeMeers, M.N. 1989. Knowledge acquisition for GIS automation of the SCS LESA model: An empirical study. AI Applications 3(4): 13-22.

Downing, K., and D.L. Bartos. 1991. AI methods in support of forest science: Modeling endemic level mountain pine beetle population dynamics. AI Applications 5(2): 105-115.

Ekblad, S.L., W.T. Hamilton, J.W. Stuth, and J.R. Conner. 1991. RangeCon: A management evacuation system for assessing success of selected range improvement practices. AI Applications 53(9-20):

Evanson, S.E. 1988. How to talk to an expert. AI Applications 2(4): 26-30.

Finkelstein, D.B., and M. Eriksson. 1994. A dual expert knowledge acquisition technique applied to plant hormone action. AI Applications 8(2): 65-73.

Goforth, G.F., and V. Floris. 1991. OASIS: An intelligent water management system for South Florida. AI Applications 5(1): 47-55.

Goldberg, D.E. 1989. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Publishing Company, Inc., Reading, Massachusetts.

Gordon, S.E. 1989. Theory and methods for knowledge acquisition. AI Applications 3: 19-30.

Gordon, S.E., and R.T. Gill. 1989. Question probes: A structured method for eliciting declarative knowledge. AI Applications 3(2): 13-20.

Graesser, A.C., and L.F. Clark. 1985. Structures and Procedures of Implicit Knowledge. Ablex Publishing, Norwood, New Jersey.

Grover, M.D. 1983. A pragmatic knowledge acquisition methodology. Pages 436-438 in: Proceedings, Eighth International Joint Conference on Artificial Intelligence, A. Bundy, editor. William Kaufman, Inc., Los Altos, California.

Haas, T.C. 1992. A Bayes network model of district ranger decision making. AI Applications 6(3): 72-88.

Hart, A. 1986. Knowledge Acquisition for Expert Systems. Kogan Page Ltd., London.

Hebb, D.O. 1949. Organization of Behavior. Science Editions, New York.

Hoffman, R.R. 1987. The problem of extracting the knowledge of experts from the perspective of experimental psychology. AI Magazine 8(2): 53-67.

Jeffers, J.N.R. 1991. Rule induction method in forestry research. AI Applications 5(2): 37-44.

Johnson, S.C. 1967. Hierarchical clustering schemes. Psychometrika 32: 241-254.

Julien, B., S.J. Fenves, and M.J. Small. 1992. Knowledge acquisition methods for environmental evaluation. AI Applications 6(l): 1-20.

Kelly, G.A. 1955. The Psychology of Personal Constructs. Norton, New York.

Langley, P., and J.G. Carbonell. 1984. Approaches to machine learning. Journal of the American Society for Information Science 35: 306-316.

Matthews, G., and J. Hearne. 1991. Clustering without a metric. IEEE Transactions on Pattern Analysis and Machine Intelligence 13(2): 175-184.

McClelland, J. L., and D E. Rumelhart. 1986. Parallel Distributed Processing: Explorations in the Microstructure of Cognition. MIT Press, Cambridge, Massachusetts.

Michalski, R. S., J.G. Carbonell, and T.M. Mitchell. 1986. Machine Learning: An Artificial Intelligence Approach. Morgan Kaufmann Publisher, Los Altos, California.

Michalski, R. S., and R.E. Stepp. 1983. Learning from observation: Conceptual clustering Pages 331-363 in Machine Learning, An Artificial Intelligence Approach. Morgan Kaufmann, Los Altos, California

Michie, D., and R. Johnston. 1985. The Knowledge Machine: Artificial Intelligence and the Future of Man. William Morrow and Company, Inc., New York.

Minsky, M.L., and S. Papert. 1969. Perceptions. MIT Press, Cambridge, Massachusetts.

Mitchell, A.A. 1987. The use of alternative knowledge-acquisition procedures in the development of a knowledge-based media planning system. International Journal of Man-Machine Studies 26: 399-411.

Narazaki, H., T. Watanabe, and M. Yamamoto. 1996. Reorganizing knowledge in neural networks An explanatory mechanism for neural networks in data classification problems. IEEE Transactions on Systems, Man, and Cybernetics 26(1): 107-117.

Nevo, A., I. Amir, and J. Dag. 1990. Knowledge acquisition for a crop-plot suitability expert system. AI Applications 4(3): 53-57.

O'Hara, K.L., S.D. Oliver, S.G. Pickford, and J.J. Townsley. 1990. A prototype decision rule base for planning and anticipating effects of silvicultural activities over broad areas. AI Applications 41): 25-34.

Olson, J.R., and H.H. Rueter. 1987. Extracting expertise from experts: methods foreknowledge acquisition. Expert Systems 4(3) 152-168.

Quinlan, J.R. 1983. Induction of decision trees. Tioga, Palo Alto, California.

Rauscher, H.M., D.A. Perala, and C.V. Worth. 1995. The ecology and management of aspen. AI Applications 9(3): 59 (2 diskettes).

Rumelhart, D.E., and J.L. McClelland. 1986. Parallel Distributed

Processing: Explorations in the Microstructure of Cognition. MIT Press, Cambridge, Massachusetts.

Rust, M. 1988. White pine blister rust hazard rating: An expert systems approach. AI Applications 2(2./3): 47-50.

Saaty, T.L. 1980. The Analytic Hierarchy Process. McGraw Hill, New York.

Schmoldt, D.L. 1987. Evaluation of an Expert System Approach to Forest Pest Management of Red Pine (Pinus resinosa). Ph. D. dissertation. University Microfilms International.

Schmoldt, D.L., and W.G. Bradshaw. 1988. A cumulative Delphi approach to knowledge acquisition. Pages 149-162 in: Third Annual Rocky Mountain Conference on Artificial Intelligence, Y. Wills, editor. Rocky Mountain Society for Artificial Intelligence, Boulder, Colorado.

Schmoldt, D.L., and D. L. Peterson. 1991. Applying knowledge-based methods to the design and implementation of an air qualit y workshop. Environmental Management 15(5): 623-634.

Schmoldt, D.L., and H. M. Rauscher. 1994. A knowledge management imperative and six supporting technologies. Computers and Electronics in Agriculture 10(1): 11-30.

Schmoldt, D.L., and H. M. Rauscher. 1996. Building Knowledge-Based Systems for Natural Resource Management. Chapman & Hall, New York.

Schweickert, R., A.M. Rurton, N.K. Taylor, E.N. Corlett, N.R. Shadbold, and A.P. Hedgecock. 1987. Comparing knowledge elicitation techniques: A case study. Artificial intelligence Review 1: 245-253.

Sell, P. 1985. Expert Systems: A Practical Introduction. John Wiley & Sons, Inc., New York.

Senjen, R. 1988. Knowledge acquisition by experiment: Developing test cases for an expert system. AI Applications 2(2/3): 52-55.

Shepard, R.N. 1962. Analysis of proximities: Multidimensional scaling with an unknown distance function. Psychometrika 27: 125-140.

Walker, D.H., F.L. Sinclair, and G. Kendon. 1995. A knowledge-based systems approach to agroforestry research and extension. AI Applications 9(3): 61-72.

Windon, D.R.J., and J.G. Massey. 1991. Translating knowledge form expert to expert system. AI Applications 5(1): 33-39.

Since 1991, Daniel L. Schmoldt has worked as a Research Forest Products Technologist with the Southern Research Station, USDA Forest Service, Thomas M. Brooks Forest Products Center at Virginia Tech. He received degrees in mathematics, computer science, and forest biometry from the University of Wisconsin-Madison. Prior to his present position with the Research Unit "Integrated Lifecycle of Wood: Tree Quality, Wood Processing, and Recycling," he worked as a Research Forester for the Fire Management Planning and Economics project and also the Atmospheric Deposition project of the Pacific Southwest Research Station, USDA Forest Service. Current research activities include (1) tree quality assessment—using imaging methods and nondestructive evaluation techniques—and (2) wood-processing automation for the hardwood industry—applying ultrasonic, computed tomography, and optical scanning coupled with machine vision systems. He also works on various projects dealing with forest management and decision making.