Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
FHWA/VA-87/32		

4. Title and Subtitle	5. Report Date
A Demonstration of Expert Systems Applications in Transportation Engineering, Volume I: Transportation Engineers and Expert Systems	June 1987
	6. Performing Organization Code
	8. Performing Organization Report No.
7. Author(s)	
Ardeshir Faghri, Michael J. Demetsky	VTRC 87-R32
9. Performing Organization Name and Address	10. Work Unit No. (TRAIS)
Virginia Transportation Research Council Box 3817 University Station Charlottesville, Va. 22903-0817	11. Contract or Grant No. HPR 2264
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address	
Virginia Department of Transportation 1221 E. Broad Street Richmond, Va. 23219	Final Report
	14. Sponsoring Agency Code

15. Supplementary Notes

In cooperation with the U. S. Department of Transportation, Federal Highway Administration

16. Abstract

Expert systems, a branch of artificial-intelligence studies, is introduced with a view to its relevance in transportation engineering. Knowledge engineering, the process of building expert systems or transferring knowledge from human experts to computers, is described. The general differences between expert systems and conventional computer programs are summarized. The architecture of the expert system is shown to separate knowledge of the problem domain (knowledge base) from general problem solving knowledge (inference engine). Different approaches to each of these tasks are described. Recent developments in computer software that support and simplify the development of expert systems are presented, and recent applications to three transportation engineering problems are described. Background information is given to recommend the development of a prototype expert system for traffic control in construction zones.

17. Key Words	18. Distribution Statement
Transportation, expert systems, software, knowledge engineering, computers	No restrictions. This document is available to the public through the National Technical Information Service, Springfield, Virginia 22161

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
unclassified	unclassified	37	

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

FINAL REPORT

A DEMONSTRATION OF EXPERT SYSTEMS APPLICATIONS IN TRANSPORTATION ENGINEERING

VOLUME I

TRANSPORTATION ENGINEERS AND EXPERT SYSTEMS

by

Ardeshir Faghri
Research Scientist Assistant

and

Michael J. Demetsky
Faculty Research Scientist

Virginia Transportation Research Council
(A Cooperative Organization Sponsored Jointly by the Virginia Department of Transportation and the University of Virginia)

In Cooperation with the U. S. Department of Transportation
Federal Highway Administration

Charlottesville, Virginia

June 1987
VTRC 87-R32

TRANSPORTATION PLANNING RESEARCH ADVISORY COMMITTEE

D. W. BERG, Chairman, Assistant Rail & Public Transportation Division Administrator , VDOT

E. D. ARNOLD, JR., Research Scientist, VTRC

B. R. CLARKE, Assistant Transportation Planning Division Administrator, VDOT

G. R. CONNER, Assistant Rail & Public Transportation Division Administrator, VDOT

R. A. DRUMWRIGHT, Transit Manager, James City County Transit Company, Williamsburg

T. F. FARLEY, Assistant District Engineer, Northern Virginia Division, VDOT

D. L. FARMER, Chief Transportation Planner, Southeastern Virginia Planning District Commission, Chesapeake

J. N. HUMMEL, Chief, Planning & Engineering Division, Arlington Department of Public Works

A. F. LAUBE, Assistant Urban Division Administrator, VDOT

ANGELA H. MOORE, Principal Planner, County of Henrico, Planning Office, Richmond

A. J. SOLURY, Division Planning & Research Engineer, FHWA

G. R. STILL, Associate Planner, City of Danville

M. S. TOWNES, Assistant to the Executive Director, Peninsula Transportation District Commission, Hampton

TRAFFIC RESEARCH ADVISORY COMMITTEE

A. L. THOMAS, JR., Chairman, Transportation Traffic Safety Division Administrator, VDOT

J. B. DIAMOND, District Traffic Engineer, VDOT

C. F. GEE, Assistant Construction Division Administrator, Central Office

T. A. JENNINGS, Safety/Technology Transfer Coordinator, FHWA

C. O. LEIGH, Maintenance Division Administrator, VDOT

T. W. NEAL, JR., Chemistry Laboratory Supervisor, VDOT

W. C. NELSON, JR., Assistant Traffic & Safety Division Administrator, VDOT

H. E. PATTERSON, Senior Traffic Engineer, Department of Public Works, Norfolk, Virginia

C. B. PERRY II, District Engineer, VDOT

R. L. PERRY, Assistant Transportation Planning Division Administrator, VDOT

F. D. SHEPARD, Research Scientist, VTRC

L. C. TAYLOR II, District Traffic Engineer, VDOT

ABSTRACT

Expert systems, a branch of artificial-intelligence studies, is introduced with a view to its relevance in transportation engineering. Knowledge engineering, the process of building expert systems or transferring knowledge from human experts to computers, is described. The general differences between expert systems and conventional computer programs are summarized. The architecture of the expert system is shown to separate knowledge of the problem domain (knowledge base) from general problem solving knowledge (inference engine). Different approaches to each of these tasks are described. Recent developments in computer software that support and simplify the development of expert systems are presented, and recent applications to three transportation engineering problems are described. Background information is given to recommend the development of a prototype expert system for traffic control in construction zones.

A DEMONSTRATION OF EXPERT SYSTEMS APPLICATIONS IN TRANSPORTATION ENGINEERING

VOLUME I

TRANSPORTATION ENGINEERS AND EXPERT SYSTEMS

by

Ardeshir Faghri
Research Scientist Assistant

and

Michael J. Demetsky
Faculty Research Scientist

INTRODUCTION

Expert systems are derived from studies of artificial intelligence (AI), which has been defined as "The study of mental faculties through the use of computational models"(1). AI can be subdivided into three relatively independent research areas (Figure 1): (1) natural language processing, (2) robotics, and (3) expert systems (2).

Edward Feigenbaum of Stanford University defined an expert system as:

> . . . an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solution. Knowledge necessary to perform at such a level, plus the inference procedures used, can be thought of as a model of the expertise of the best practitioners in the field.
>
> The knowledge of an expert system consists of facts and heuristics. The facts constitute a body of information that is widely shared, publicly available, and generally agreed upon by experts in a field. The heuristics are mostly private, little-discussed rules of good judgement (rules of plausible reasoning, rules of good guessing) that characterise expert-level decision making in the field. The performance level of an expert system is primarily a function of the size and the quality of a knowledge base it possesses (2).

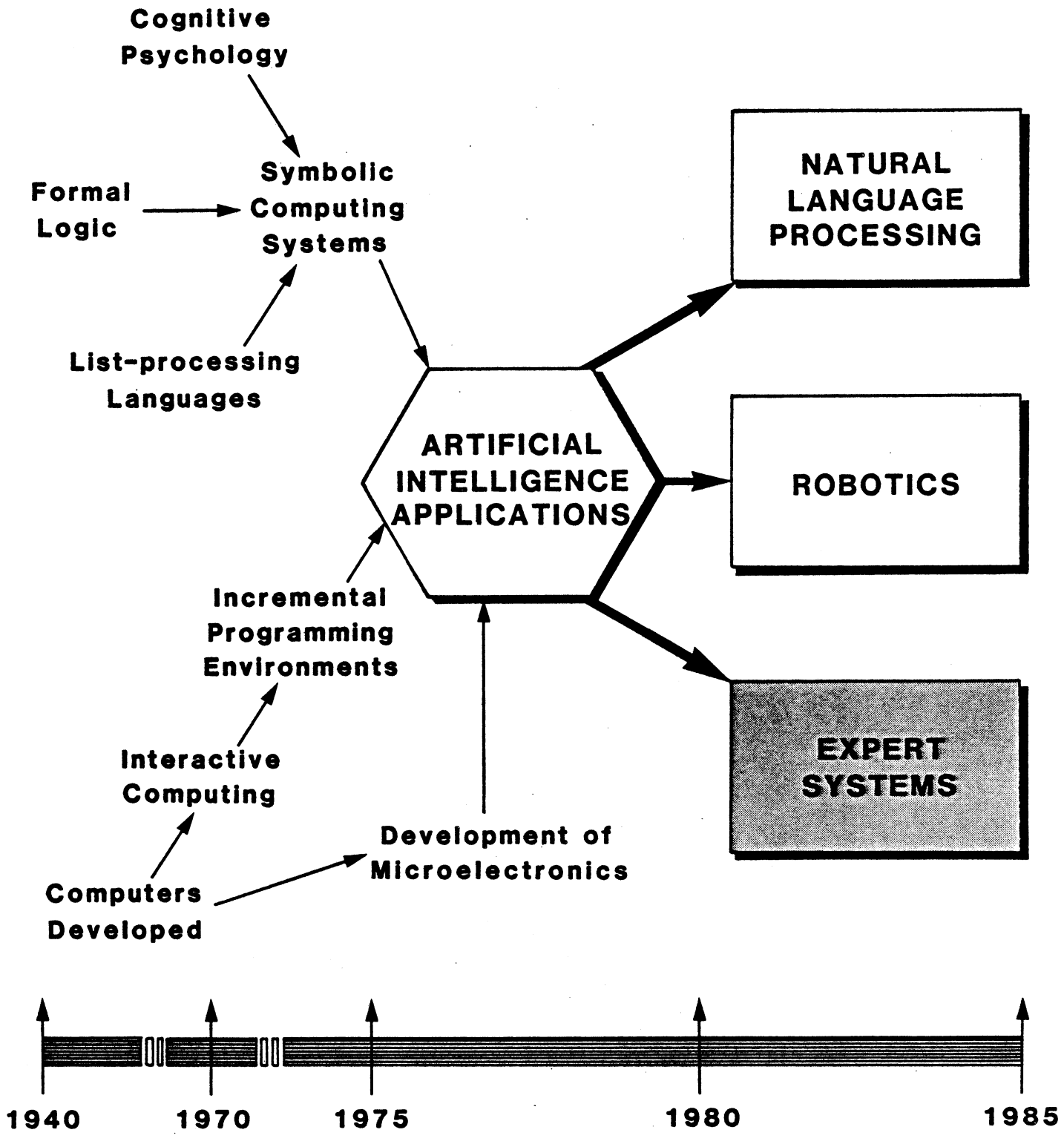


Figure 1. The evolution of expert systems

The process of building expert systems is referred to as knowledge engineering, and those who design and build expert systems are called knowledge engineers (3).

The first knowledge-based expert systems (KBES) were built by interviewing a recognized human expert and attempting to extract that expert's knowledge, hence the term "expert systems." Recently, however, several quite useful systems that contain knowledge of a difficult decision-making situation were built, but they are hardly the equivalent of a human expert. To avoid suggesting that all systems built by means of knowledge-engineering techniques capture the knowledge of a human expert, "knowledge systems" is rapidly becoming the preferred name, but both terms are used synonymously (2).

Knowledge engineers are concerned with identifying the specific knowledge an expert uses in solving a problem. First, he/she determines what facts and rules of thumb a human expert uses. Then the knowledge engineer determines the inference strategy the expert uses in an actual problem-solving situation. Finally, a system is developed that simulates the expert's judgment.

If a knowledge system program is to function like a human expert, it must be able to do the things human experts commonly do. For example, experts consult with others to help solve problems. Thus, most knowledge systems ask questions, explain their reasoning if asked, and justify their conclusions. Moreover, they typically do this in language the user can understand. They allow the user to skip questions, and most can function even when the user provides incomplete or uncertain data. In other words, knowledge systems interact with a user in the same way a human consultant does.

The best expert systems can solve difficult problems, within a very narrow domain, as well as or better than human experts can (2). This is not to suggest that today's expert systems are as effective as human experts. The technology is new and just beginning to be applied. Knowledge systems are confined to well-defined tasks (2); they are not able to reason broadly over a field of expertise. They cannot reason from axioms or general theories. They do not learn on their own; thus, they are limited to using the specific facts and heuristics they were "taught." They lack common sense; they cannot reason by analogy; and their performance deteriorates rapidly when problems extend beyond the narrow task they were designed to perform.

It is reasonable, for example, to consider developing an expert system to guide, tutor, and consult a transportation engineer in analyzing the capacity of signalized intersections (4). This problem is sufficiently well defined and adequately limited so as to result in a useful, small system. One would not, however, want to try to develop a system for the entire field of traffic engineering. This would be too unwieldy--because it would be too broad--to be of any use.

On the other hand, knowledge systems do not display biased judgments, nor do they jump to conclusions and then seek to maintain those conclusions in the face of disconfirming evidence. They do not have "bad days"; they always attend to details; they always systematically consider all of the possible alternatives. The best of them, equipped with thousands of heuristic rules, are able to perform their specialized tasks better than a human specialist (2).

CONVENTIONAL PROGRAMS VERSUS EXPERT SYSTEMS

The most significant difference between expert systems and conventional programs is that expert systems manipulate knowledge while conventional programs manipulate data. Table 1 shows the general differences between expert systems and conventional programs.

Table 1

Comparison of data processing and knowledge engineering

Conventional Programs	Expert Systems
-Representation and use of data	-Representation and use of knowledge
-Algorithmic	-Heuristic
-Repetitive process	-Inferential process
-Sequential, batch processing	-Highly interactive processing
-Mid-run explanation impossible	-Mid-run explanation possible
-Effective manipulation of large data bases	-Effective manipulation of large knowledge bases

For AI researchers, an expert system is defined as a computer program that possesses expertise, symbolic reasoning, depth, and self-knowledge (5). An expert system represents knowledge symbolically, as sets of symbols that stand for problem concepts. In AI jargon, a symbol is a string of characters that stands for some real-world concept (3). The following are symbols: "product," "defendant," and "0.8." These symbols can be combined to express relationships. When these relationships are represented in an AI program, they are called "symbol structures" (3). The following are examples of symbol structures: "defective product," "leased-by product defendent," and "equal (liability defendent) 0.8." These structures can be interpreted to mean "the product is defective," "the product is leased by the defendant," and "the liability of the defendant is 0.8."

An expert system has depth; that is, it operates effectively in a narrow domain containing difficult problems. Thus, the rules in an expert system are numerous and often individually complex.

Most current expert systems have what is called an explanation facility for explaining how the system arrived at its answers. Most of these explanations involve displaying the inference chains and explaining the rationale behind each rule used in the chain. The ability to examine their reasoning process and explain their operation is one of the most important qualities of expert systems (3).

THE ARCHITECTURE OF EXPERT SYSTEMS

The heart of an expert system is its knowledge, which is structured to support decision making. When AI scientists use the term "knowledge," they mean the information a computer program needs before it can function properly (3). This information can take the form of facts or rules.

Facts: Responses to a brake light from a leading vehicle require 0.4 second to above 1.0 second for some drivers (6).

All physical motor capabilities deteriorate with age.

Rules: If forced flow and low speeds exist on a segment of a highway a level of service F is achieved.

If the degree of congestion and/or vehicle delay caused by daytime lane closures is severe, nighttime construction and/or maintenance should be considered.

Facts and rules in an expert system are not always true or false; sometimes there is a degree of uncertainty about the truth of a fact or the validity of a rule. When this doubt is made explicit, it is called a "certainty factor"(3).

Fact: Rail-highway crossings that are located near major employment centers experience more accidents with certainty 0.7.

Rule: If the average speed increases by 10 m.p.h. with certainty 1.0, the number of accidents will increase by 10 percent with certainty 0.6.

The organization of knowledge in an expert system separates the knowledge about the problem domain from the system's other knowledge, such as general knowledge about how to solve problems or knowledge about how to interact with the user. The collection of domain knowledge is called the knowledge base; the general problem-solving knowledge is called the inference engine."

The knowledge base in an expert system contains facts (data) and rules (or other representations) that use those facts as the basis for decision making. The inference engine contains an interpreter that decides how to apply the rules to infer new knowledge and a scheduler that decides the order in which the rules should be applied (See Figure 2).

KNOWLEDGE REPRESENTATION TECHNIQUES

The process of formulating or viewing a problem so it will be easy to solve in an expert system is called knowledge representation (3). There are a standard set of knowledge representation techniques, any of which can be used alone or in conjunction with others to build expert systems. Each technique provides the program with certain benefits making it more efficient, more easily understood, or more easily modified (7). The three most widely used in current expert systems are rules (the most popular by far), semantic nets, and frames. Other methods are object-attribute-value triplets, and logical expressions. Each method formalizes the knowledge of the expert and the facts of the problem domain so as to contain all of the information required to make an intelligent decision.

Knowledge Representation Using Rules

The most popular type of knowledge representation technique is rule-based representation (3). Rules provide a formal way of representing recommendations, directives, and strategies; they are often appropriate when the domain knowledge results from empirical associations developed through years of problem-solving experience. Rules are expressed as conditionals ("if-then" statements) (3).

1. If a flammable liquid is spilled, the fire department should be called.
2. If the pH of the spill is less than 6, the spill material is an acid.
3. If the spill material is an acid and smells like vinegar, it is acetic acid.

These are rules that might exist in an expert system for containing oil and chemical spills (8).

Each of the two parts of the antecedent in rule 3 is called an "expression" or an "if clause." The consequent contains a single expression or "then clause," although it could just as well contain more than one. The clauses in the antecedent can be connected with the logical operators "and" or "or " (2).

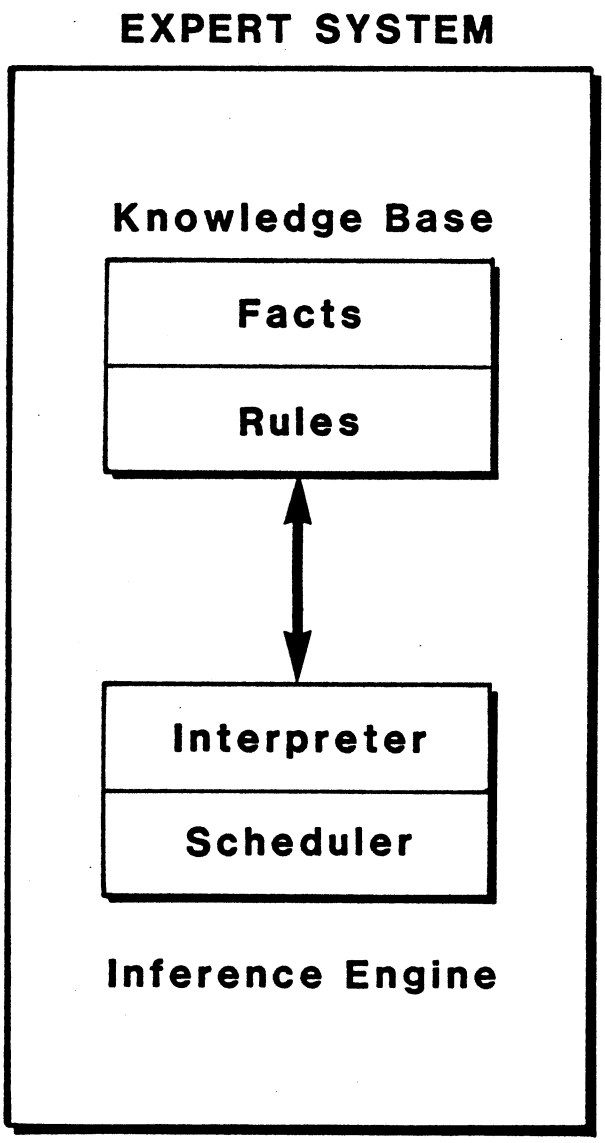


Figure 2. The architecture of expert systems

In a rule-based expert system, knowledge is represented as sets of rules that are checked against a collection of facts or knowledge about the current situation. When the antecedent of a rule is satisfied by the facts, the action specified by the consequent is performed. When this happens, the rule is said to fire or execute (3). A rule interpreter compares the antecedents with the facts and executes the rule whose consequent matches the facts, as shown below:

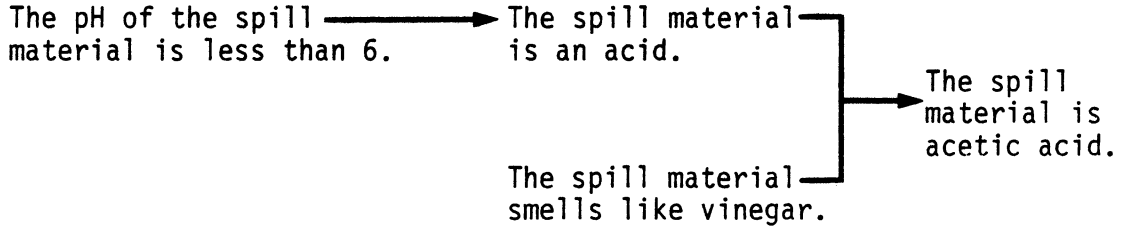
Facts: A flammable liquid was spilled. The pH of the spill material is less than 6. The spill smells like vinegar.

Rules: If the pH of the spill is less than 6, the spill material is an acid.

The new fact added to knowledge base: The spill material is an acid.

The rule's action may modify the set of facts in the knowledge base, by adding a new fact. The new facts added to the knowledge base can themselves be matched with the antecedent of rules.

This matching of rule antecedents to the facts can produce what are called inference chains (3). The inference chain for this example is as follows:



This inference chain shows how the system used the rules to infer the identity of the spill material. An expert system's inference chains can be displayed to the user to help explain how the system reached its conclusions.

Knowledge Representation Using Semantic Nets

Another approach to the representation of knowledge is based on a network structure, called a semantic net (also called a semantic network). Semantic nets were originally developed for use as psychological models of human memory but are now a standard method of representation for AI and expert systems (9). A semantic net consists of points called nodes connected by links called arcs describing the relations between the nodes. The nodes in a semantic net stand for

objects, concepts, or events. Arcs can be defined in a variety of ways, depending on the kind of knowledge being represented:

1. Isa arcs are most often used to establish a property-inheritance hierarchy; that is, instances of a class have all properties of more general classes of which they are members. The is-a relation is transitive: items lower in the net can inherit properties from items higher up in the net.

2. Has-part arcs identify nodes that are properties of other nodes.

Figure 3 illustrates both isa and has-part arcs in a simple net for the concept of public transit mode. Semantic nets used to describe natural languages use arcs such as "agent," "object," and "recipient" (3).

The isa relation (like the has-part relation) establishes an inheritance hierarchy for properties in the net (3). This means that items lower in the net inherit properties from items higher up in the net. This saves space since information about similar nodes does not have to be repeated at each node and can be stored in one central location. For example, in the public-transit-mode semantic net the common parts of each mode, such as passenger seat and engine, are stored once at the mode level, rather than repeatedly at lower levels like bus or particular bus system. The net can be searched, using knowledge about the meaning of the relations in the arcs, to establish facts like "Washington Metro has passenger seats." Semantic nets are a useful way to represent knowledge and to simplify problem solving in domains that use well-established taxonomies (3).

Representation of the Knowledge Using Frames

Frames provide another method for representing facts and relationships. A frame is a description of an object that contains slots for all the information associated with the object. Slots may store values (2). Each slot can have any number of procedures attached to it. Three useful types of procedures often attached to slots are listed below (3).

1. If-added procedure: Executes when new information is placed in the slot.
2. If-removed procedure: Executes when information is deleted from the slot.
3. If-needed procedure: Executes when information is needed from the slot, but the slot is empty.

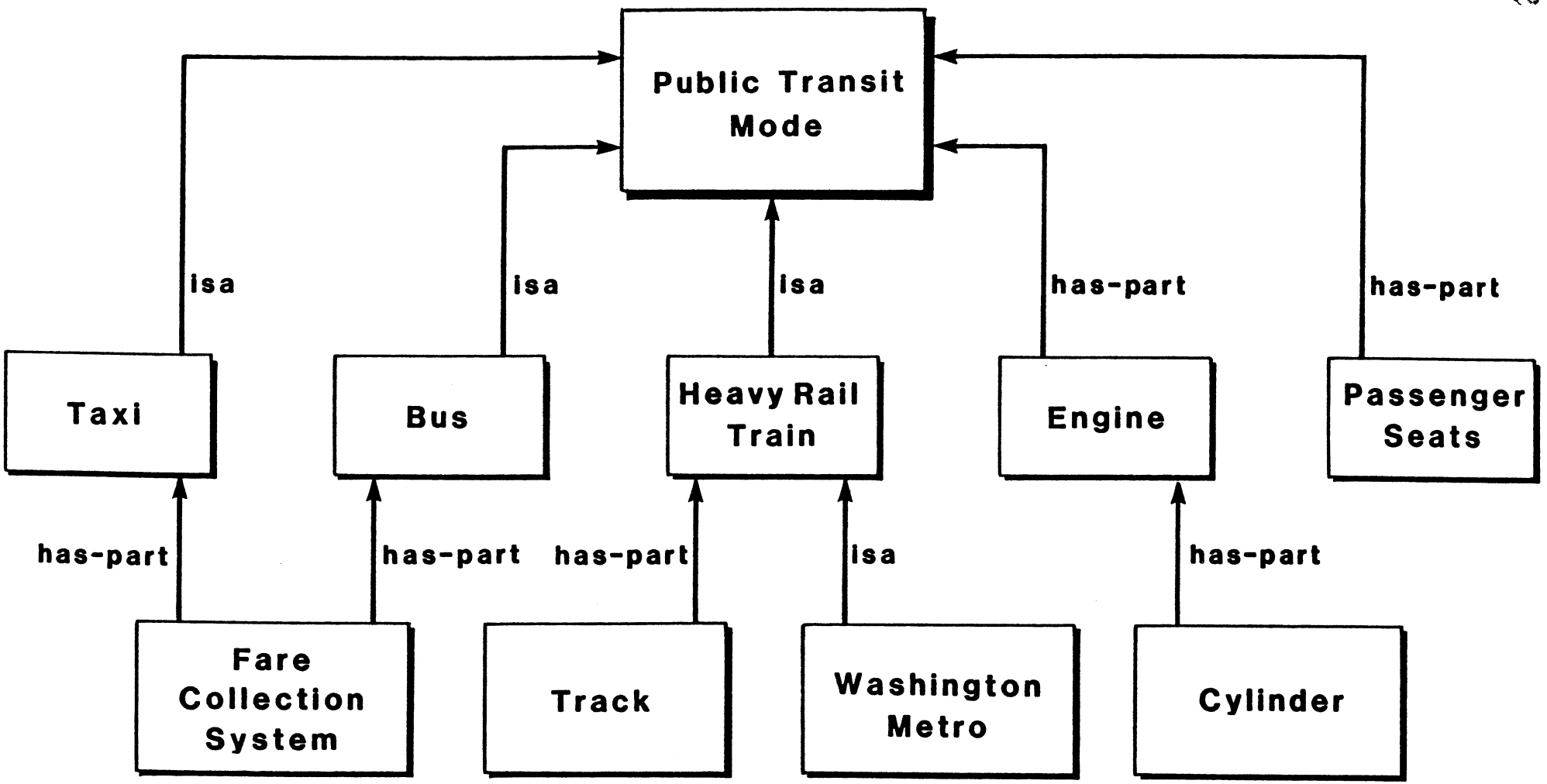


Figure 3. A simple semantic net for the concept of public transit mode.

These attached procedures can monitor the assignment of information to the node, thereby ensuring that appropriate action is taken when values change. Marvin Minsky, who originated the frame idea, describes it as follows:

> A frame is a data-structure for representing a stereotyped situation, like being in a certain kind of livingroom, or going to a child's birthday party. Attached to each frame are several kinds of information. Some of this information is about how to use the frame. Some is about what one can expect to happen next. Some is about what to do if these expectations are not confirmed (3).

A frame is organized much like a semantic net. It is a network of nodes and relations organized in a hierarchy, where the highest nodes represent general concepts and the lower nodes represent properties of those concepts.

Frame systems are useful for problem domains where expectations about the form and content of the data play an important role in problem solving, such as interpreting visual scenes or understanding speeches (3).

Object-Attribute-Value Triplets

Another way to represent factual information is with object-attribute-value (OAV) triplets. In this scheme, objects may be physical entities such as a door, a coat, or a transistor; or they may be conceptual entities such as a logic gate, a bank loan, or a sales episode. Attributes are general characteristics or properties of objects. Size, shape, and color are typical attributes of physical objects. Interest rate is an attribute for a bank loan, and setting might be an attribute for a sales episode. The final member of the triplet is the value of an attribute. The value specifies the specific nature of an attribute in a particular situation. An apple's color may be red or the interest rate for a bank loan may be 12 percent (2). Figure 4 shows an example of OAV representation.

Representing knowledge with OAV triplets is a specialized case of a semantic network. Exotic links are banished in favor of two simple relationships. The object-attribute link is a has-a link, and the attribute-value link is an isa link. For example, a bank loan has a rate of interest, and 12 percent is a rate of interest. Nodes are classified as either objects, attributes, or values (2).

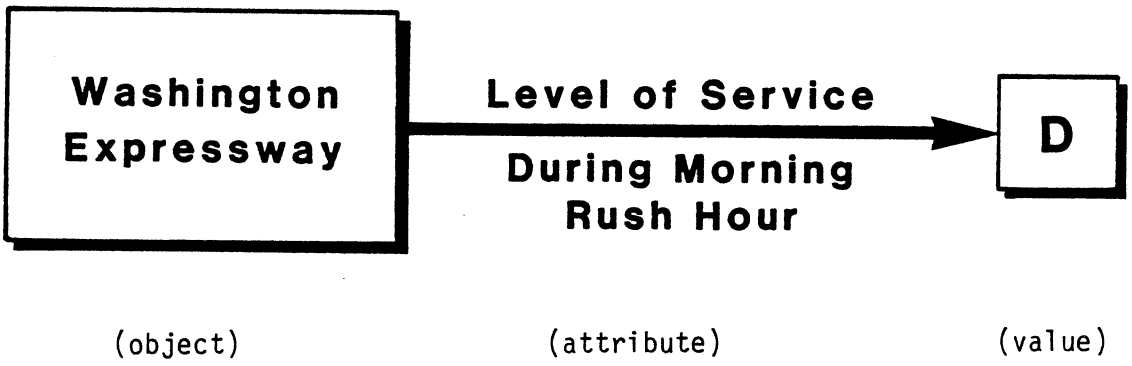


Figure 4. An example of OAV representation.

Logic

Logic provides another way of representing knowledge. The two most common logical notations are the propositional and the predicate calculi. The propositional and predicate calculi are formal systems designed to perspiciously represent the logical structure of propositions and the logical relations between them.

These logical systems are of interest because they are a formalization of a procedure for the derivation of new factual knowledge from old by means of deductive inference.

The propositional calculus is particularly concerned with the logical relations that exist between compound propositions connected by the logical connectives--"and," "or," "not," "if-then," and "if-and-only-if." Propositions are statements that can be true or false. If the statement "grass is green" is known to be true, then we may infer that the disjunction "grass is green or p" (where p is any statement) is true.

The predicate calculus is an extension of the propositional calculus. Whereas, the propositional calculus is an adequate depiction of logical connections in which propositions appear as unarticulated disjointed wholes, the predicate calculus is concerned to adequately depict the logical connections between propositions that depend on their inner logical structure.

A brief example will suffice to demonstrate this difference.

A. Plato is a philosopher.
B. Aristotle is a philosopher.

If for the moment we ignore the normal symbolism of the propositional calculus, we might say that "Platophilosopher" and "Aristotlephilosopher" are symbolic of A and B; whereas the predicate calculus depicts the internal similarity of these statements by symbolizing them as predicates (concepts) with an empty space to be filled by any object:

_____ is a philosopher.

Using the symbolism just adopted for the propositional calculus we would have:

_____ philosopher
or
( ) philosopher
or
(X) philospher

Two values of the variable X are "Plato" and "Aristotle." Thus, the predicate calculus shows the logical similarity in the internal structure of these statements. (Two-place predicates can be symbolized in the same way: (x,y) is the brother of.) This depiction of the

internal logical structure of statements makes it possible to represent a far wider range of inferences in the predicate calculus than in the propositional calculus.

INFERENCE AND CONTROL

Inference and control strategies guide a knowledge system as it uses and interprets the facts and rules stored in its knowledge base, and the information it acquires dynamically from the user. Figure 5 provides an overview of the architecture of an expert system. The inference engine is shown standing between the user and the knowledge base and performs two major tasks: First, it examines existing facts and rules, and it adds new facts when possible. Second, it decides the order in which inferences are made. The inference engine consults with the user (2).

Inference

Modus Ponens

The most common inference used in knowledge systems is modus ponens (2): if A is true, then B is true; A is true; therefore, B is true.

Figure 6 shows two rules and some facts from a hypothetical knowledge base. Lee Highway, for example, has an attribute called peak-period volume, which has the value "2500 veh/hr." Rule 1 states that if peak period volume is more than 2000 veh/hr/lane, then congestion occurs on the highway. Because the antecedent of Rule 1 is true, modus ponens allows us to conclude that congestion will occur on Lee Highway. If the second rule is tested, it will not succeed. Since the highway's shoulders are not less than five feet, the rule does not support the conclusion.

Reasoning About Uncertainty

In knowledge programming, an inference engine must be able to handle incomplete information. Incomplete information is handled by allowing rules to fail if the information necessary to evaluate the premises of these rules is unavailable (2). The result depends on the exact nature of the premise. If clauses in a premise are connected to each other by "and", then all the clauses must be true before the rule can succeed. If the user answers "unknown" to any part of the premise, the rule fails. If, however, the clauses are connected by "or," incomplete information need not preclude the rule from succeeding; a rule may succeed even though the truth of one clause in the premise is not known.

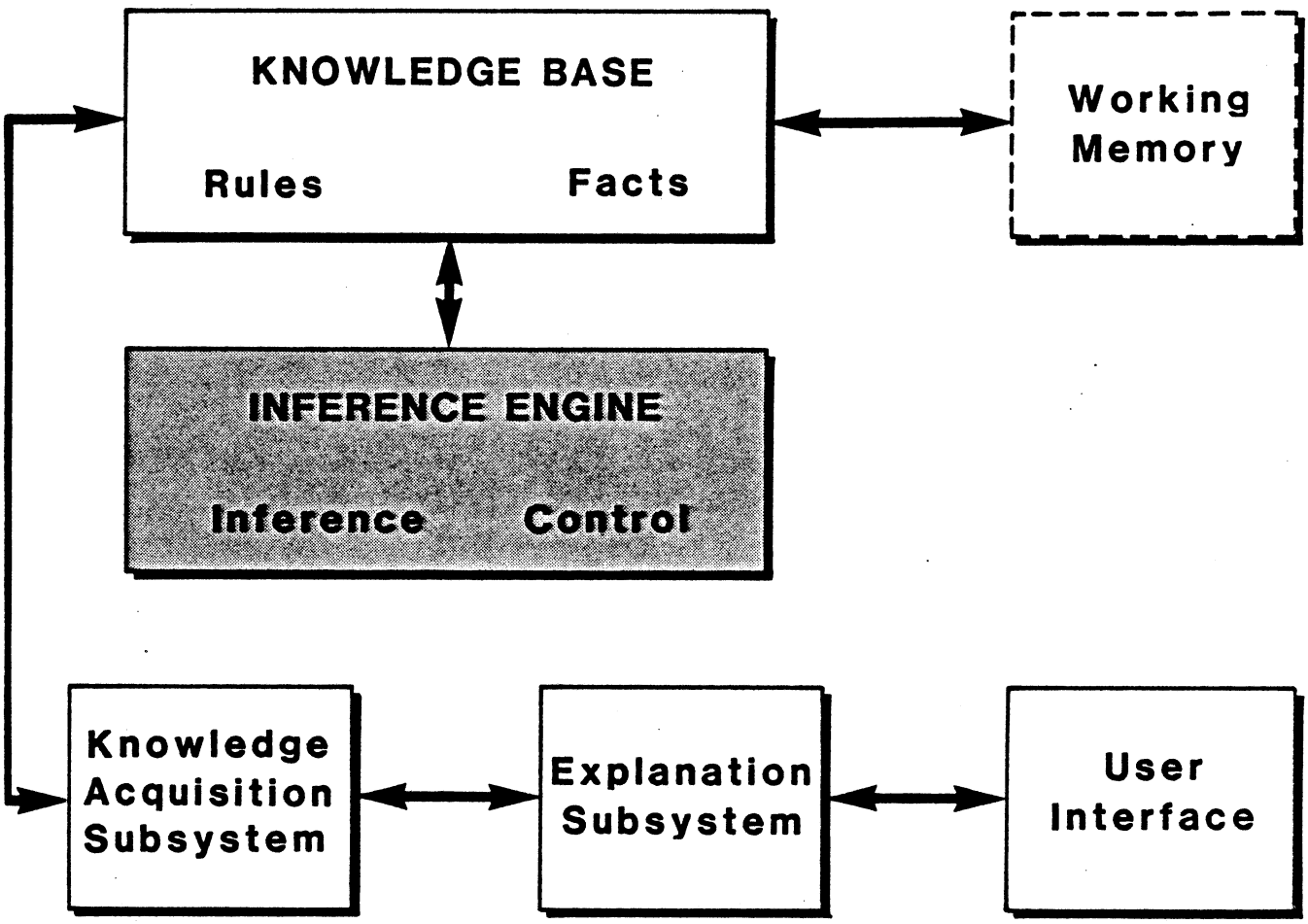
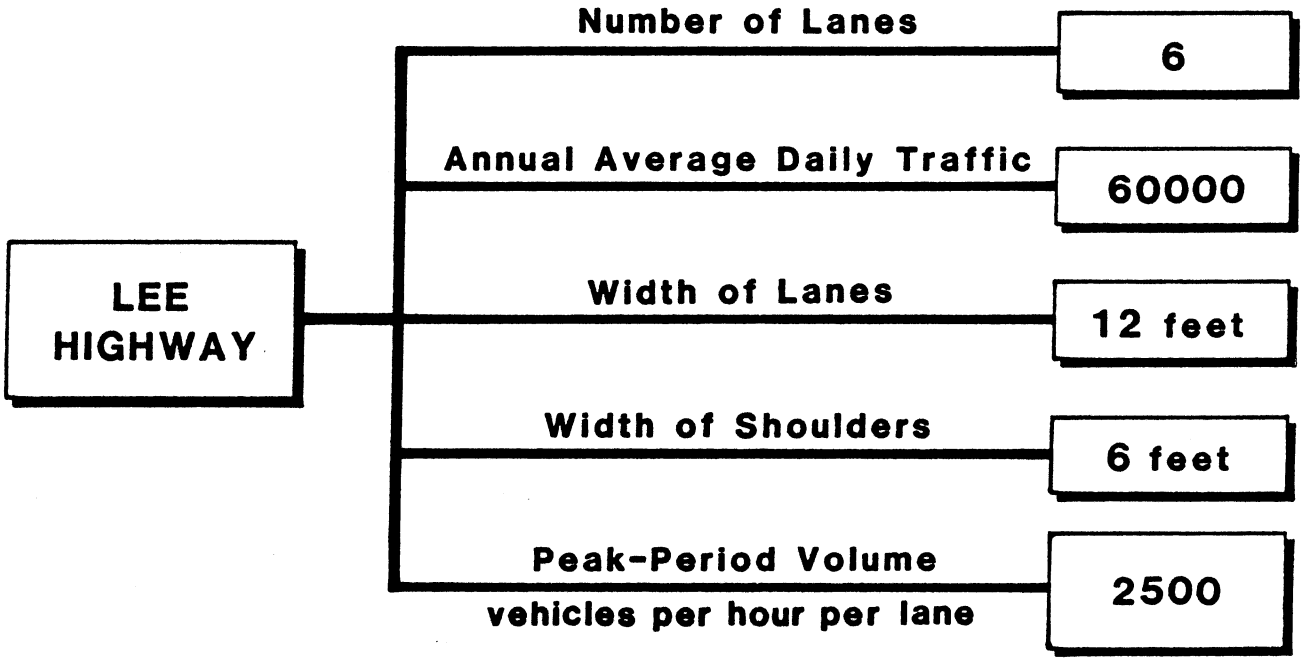


Figure 5. An overview of the architecture of an expert system.



RULE 1: **If the peak-period volume exceeds 2000 vehicles per hour, per lane,**

then congestion occurs.

RULE 2: **If the shoulders are less than 5 feet wide,**

then emergency vehicles should not park on the shoulders.

Figure 6. Some rules and facts from a hypothetetical knowledge base.

The user of an expert system may be uncertain that an assertion is true. Most knowledge systems provide for uncertain information. The degree of certainty is represented as a number attached to a fact. Certainty factors, for example, may range from -1 to +1. Rules also have a certainty factor associated with them. The inference engine handles indefinite or uncertain information by propagating certainty factors (2).

Resolution

Resolution is one way to discover whether a new assertation is true (2). In order to show an example of resolution, two other logical operations must be established. First, "If A, then B" and "Not (A) or B" are equivalent. For example, "If you have an apple, then you have a fruit" is truth-functionally equivalent to "Either you do not have an apple or you have a fruit."

In logic, in order to prove truth-functional equivalence, a truth table is needed. If two expressions are equivalent, they will have identical truth tables. The truth table for the expressions in question is shown below:

A	B	Not(A)	If A Then B	Not(A) or B
T	T	F	T	T
T	F	F	F	F
F	T	T	T	T
F	F	T	T	T

The second logical operation needed for resolution is the cancellation of A and not A. These two operations allow the resolution of "if A, then B" and "A or C" into "B or C."

Control

There are two primary problems addressed by the control portion of the inference engine:

1. A knowledge system must have a way to decide where to start.

2. The inference engine must resolve conflicts that occur when alternative lines of reasoning emerge. The system may reach a point at which two or more rules can be applied. The inference engine must choose which rule to examine next.

Some of the control strategies used by inference engines are discussed below.

Backward and Forward Chaining

Two basic problem-solving strategies are used by knowledge-based-systems: data driven (or forward chaining) and goal driven (or backward chaining). The first approach starts with the initial set of data and examines the most likely alternatives. Where additional data is required to resolve an ambiguity, the user is requested to provide the necessary information. For example, a set of rules may be chained together as:

```
If A, then B (Rule 1)
If B, then C (Rule 2)
A            (data)
Therefore C (Conclusion)
```

This is a forward chaining inference: the data that are known (in this case A) drive the inferences from left to right in rules, with rules chaining together to deduce a conclusion (C).

The backward chaining approach starts by assuming a certain solution, and then attempting to determine whether all the data are consistent with this solution. If an item of data is detected that renders the initial hypothesis invalid, an alternative solution is investigated. In this case, a system starts with a statement of the goal to be achieved and works "backward" through inference rules, i.e. from right to left, to find the data that establish that goal.

```
Find C       (Goal)
If B, then C (Rule 1)
If A, then B (Rule 2)
Therefore if A, then C (Implicit rule)
```

Question: Is A true? (Data)

Depth-first Versus Breadth-first Search

In addition to distinguishing between backward-chaining and forward-chaining strategies, we also need to distinguish between a depth-first and a breadth-first search of a knowledge base. In a depth-first search, the inference engine takes every opportunity to produce a subgoal. Searching for detail first is the mode of back-chaining in a depth-first manner. A breadth-first search sweeps across all premises in a rule before searching for greater detail. A breadth-first search will be more efficient if one rule succeeds and an attribute's value is obtained. Most systems employ depth-first search. Digging deeper and deeper into details and following a chain of rules directs in a meaningful way the questions that the knowledge system asks.

Monotonic Versus Nonmonotonic Reasoning

Another distinction among inference engines is whether they support monotonic or nonmonotonic reasoning. In a monotonic reasoning system, all values concluded for an attribute remain true for the duration of the consultation session. Facts that become true remain true, and the amount of true information in the system grows steadily or monotonically. In a nonmonotonic reasoning system, facts that are true may be retracted. Planning is a good example of a problem that demands nonmonotonic reasoning. In the early stages of a planning problem, it may make sense to go a certain way. Later, as information continues to come in, it may turn out that an early decision was wrong. Decisions and their consequences may need to be retracted.

EXPERT SYSTEM TOOLS

Expert system tools are the programming languages and support packages used to build the expert system. The three major categories of tools available for expert system building are programming languages, knowledge-engineering languages, and system-building aids.

Programming Languages

The programming languages used for expert systems applications are generally either problem-oriented languages, such as FORTRAN and PASCAL, or symbol-manipulation languages, such as LISP and PROLOG. Currently, the most popular symbol-manipulation language for artificial intelligence applications is LISP (3). A feature of LISP that distinguishes it from most other languages is its mechanism for manipulating symbols. LISP can manipulate symbols readily because of its list structure characteristics. List structures are collections of items enclosed by parantheses in which each item can be either a symbol or another list. Complex concepts can be represented and built into an expert system using the list structures.

Problem-oriented languages are generally designed for solving particular classes of problems. FORTRAN, for example, is designed to perform algebraic calculations for scientific, mathematical, and statistical problems. Problem-oriented languages have been used in expert system development, but are not very popular for extensive applications.

Knowledge Engineering Languages (SHELLS)

Knowledge-engineering languages represent a subclass of programming languages which are designed specifically for expert system development. They fall into two major categories: skeletal systems and general-purpose systems. Removing the expert system domain-specific knowledge leaves the skeletal system, the inference engine, and the

support facilities. Support facilities represent the environment associated with an expert system building tool that helps the user interact with the expert system. Because skeletal systems apply only to a limited class of problems, they lack generality and flexibility as a building tool method. The structure and built-in facilities of a skeletal system, however, make expert system development easy and fast.

In contrast, general-purpose knowledge-engineering languages can handle a wide range of different problem areas and types. They provide more control over accessing information in the knowledge base than does a skeletal system. The general-purpose languages, however, may be more difficult to use (3).

System-building Aids

The system-building aids consist of commercially available software programs that can be classified as either design aids or knowledge acquisition aids. The design aids help the expert system developer design and build an expert system by establishing a framework for the representation of knowledge and its supporting facilities. The knowledge acquisition aids assist the expert system builder in transfering the knowledge rules and heuristics from the human expert to the knowledge base of an expert system.

Table 2 summarizes selected available commercial systems by the three categories of tools discussed in the previous sections. The selection of a tool for any given task depends on specific aspects of the problem such as the time for development, needs of the problem, needs of the application, desired level of reliability and maintainability, and the availability of money, personnel, and hardware.

Table 2

Selected Commercial Systems

Category	*Tool*	*Use*	*Description*	*Hardware*	*Developer*
Programming Languages	INTERLISP-D	General-purpose	Procedure-oriented	Xerox 1100	Xerox Corporation
	LISP	General-purpose	Procedure-oriented	Lambda machines	LISP Machine, Inc.
	PROLOG	General-purpose	Logic-based Procedure-oriented	DEC 10 system DEC 20 system	Quintus Computer Systems, Inc.
	SMALLTALK -80	General-purpose	Object-oriented	Tektronix 4404	Xerox Corporation
	ZETALISP	General-purpose	Procedure-oriented	Symbolics 3600	Symbolics, Inc.
System-Building Aids	EXPERT-EASE	Knowledge acquisition	Infers a decision tree from examples	IBM PC IBM-XT Victor 9000 DEC Rainbow	Export Software International
	PLUME	Natural language interface development	Software tool for building interfaces	Symbolics 3600 VAX-11 systems	Carnegie Group, Inc.
	RULE-MASTER	Knowledge acquisition	Infers a decision tree from examples	VAX-11 systems Apollo system Sun system	Radian Corporation
	TIMM	Knowledge acquisition	Infers rules from examples	VAX 11/780 Prime 400	General Research Corporation
Knowledge Engineering Languages	ART	General-purpose	Rule-based Frame-based Procedure-oriented	CADR machines Symbolics 3600	Inference Corporation
	DUCK	General-purpose	Logic-based Rule-based	Symbolics 3600 DEC VAX systems	Smart Systems Technology
	KEE	General-purpose	Rule-based Frame-based Procedure-oriented Object-oriented	Xerox 1100 Symbolics 3600	Intellicorp

Table 2, continued . . .

Tool	*Use*	*Description*	*Hardware*	*Developer*
KES	General-purpose	Rule-based Frame-based	DEC VAX systems operating under UNIX or VMS	Software Architecture & Engineering, Inc.
M.1	General-purpose	Rule-based English-like syntax	IBM PC	Teknowledge
OPS5	General-purpose	Rule-based	VAX-11 systems	Digital Equipment Corporation
OPS5e	General-purpose	Rule-based	Symbolics 3600	Verac Corporation
OPS83	General-purpose	Rule-based Procedure-oriented	VAX-11 systems	Production Systems Technologies, Inc.
PERSONAL CONSULTANT	Diagnosis	Rule-based	TI Professional Computer	Texas Instruments
S.1	General-purpose	Rule-based Frame-based Procedure-oriented	Xerox 1100 Xerox 1108	Teknowledge
SeRIS	Diagnosis	Rule-based	IBM PC	SRI-International
SRL+	General-purpose	Frame-based	Symbolics 3600 VAX-11 systems	Carnegie Group, Inc.

Source: A Guide to Expert Systems.

APPLICATIONS OF EXPERT SYSTEMS

General Systems

Expert systems have been built to solve many different types of problems but their basic activities can be grouped into the categories shown in Table 3 (10).

TABLE 3
Generic Categories of Knowledge
Engineering Applications

Category	Problem Addressed	Transportation Engineering Application
Interpretation	Inferring situation descriptions from sensor data	Concrete delamination for pavements
Prediction	Inferring likely consequences of given situations	Transportation demand forecasting
Diagnosis	Inferring system malfunctions from observations	Highway pavement decay diagnosis
Design	Configuring objects under constraints	Highway geometric design
Planning	Designing actions	Bus transit network planning
Monitoring	Comparing observations to plan vulnerabilities	Intersection signal monitoring
Debugging	Prescribing remedies for malfunctions	Network congestion
Repair	Executing a plan to administer a prescribed remedy	Transit system vehicle repair
Instruction	Diagnosing, debugging, and repairing student behavior	Transportation short courses
Control	Interpreting, predicting, repairing, and monitoring system behaviors	Air traffic control

Although the basic functions of expert systems shown in Table 3 are easy to describe, it is misleading to use them to categorize existing expert systems because many expert systems perform more than just one function. For example, diagnosis often occurs with debugging, monitoring with control, and planning with design. Consequently, AI researchers find it useful to categorize expert systems by the types of problems they solve. Table 4 shows some of the problem domains in which expert systems are now working. Of these areas, the medical domain seems the most popular. More expert systems have been developed for medicine than for any other single problem area, although chemistry is a close second and closing fast (9). To help illustrate the relationship between an application area and its corresponding basic activities, Figure 7 shows selected expert systems currently in use in the field of engineering. The most successful of these expert systems is DELTA, a fault diagnosis system developed by General Electric in the mid-1980s. General Electric plans to use DELTA on a commercial basis to help maintenance personnel find malfunctions in diesel electric locomotives (3).

Currently, in the field of transportation engineering, only prototypes of expert systems have been developed. Figure 8 shows a few selected prototype expert systems in transportation engineering and how they relate to the basic expert system activities.

Table 4

Application Areas for Expert Systems

Agriculture	Mathematics
Chemistry	Medicine
Computer Systems	Meteorology
Engineering	Military
Geology	Physics
Law	Space Technology

Applications in Transportation Engineering

CHINA

The prototype expert system CHINA (Computerized Highway Noise Analyst) assists the transportation engineer in designing highway noise barriers and demonstrates the potential use of expert systems in transportation engineering planning problems. Since the decisions of CHINA require many algebraic calculations, it interacts with an existing FORTRAN design model (11).

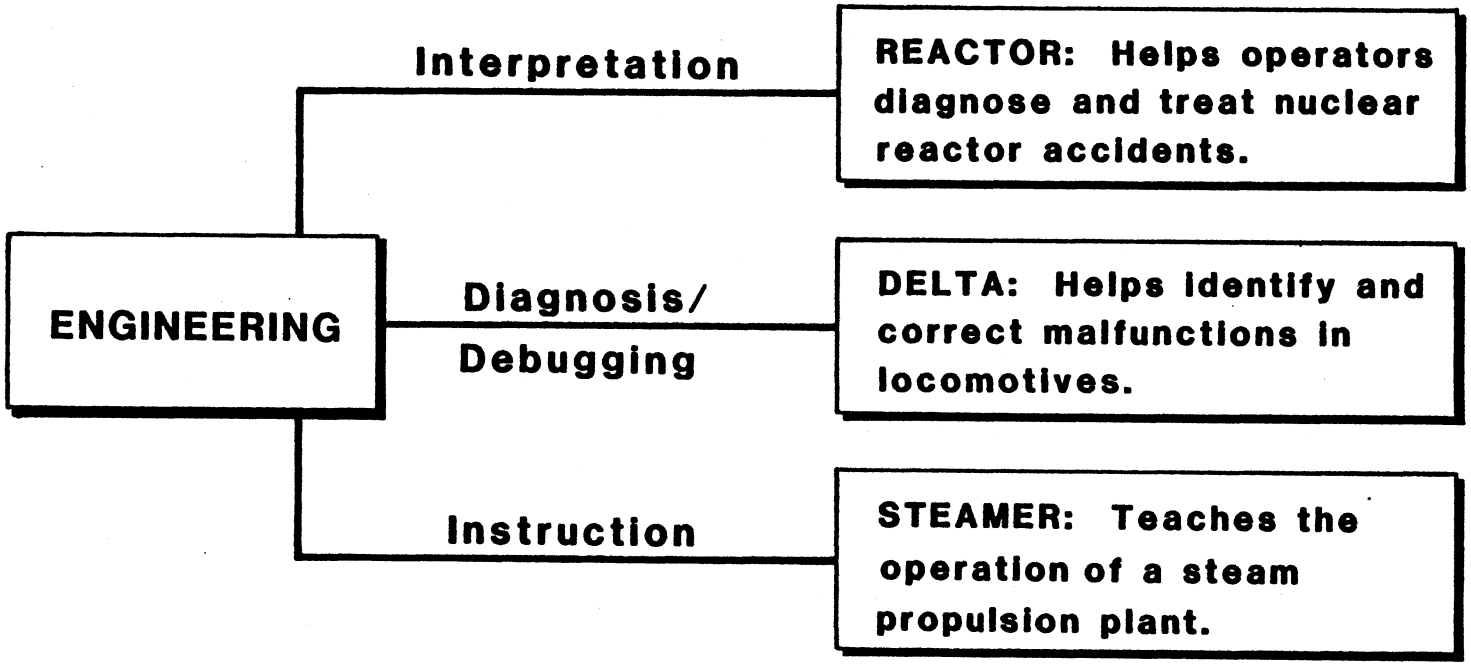


Figure 7. Selected expert systems currently in use in engineering.

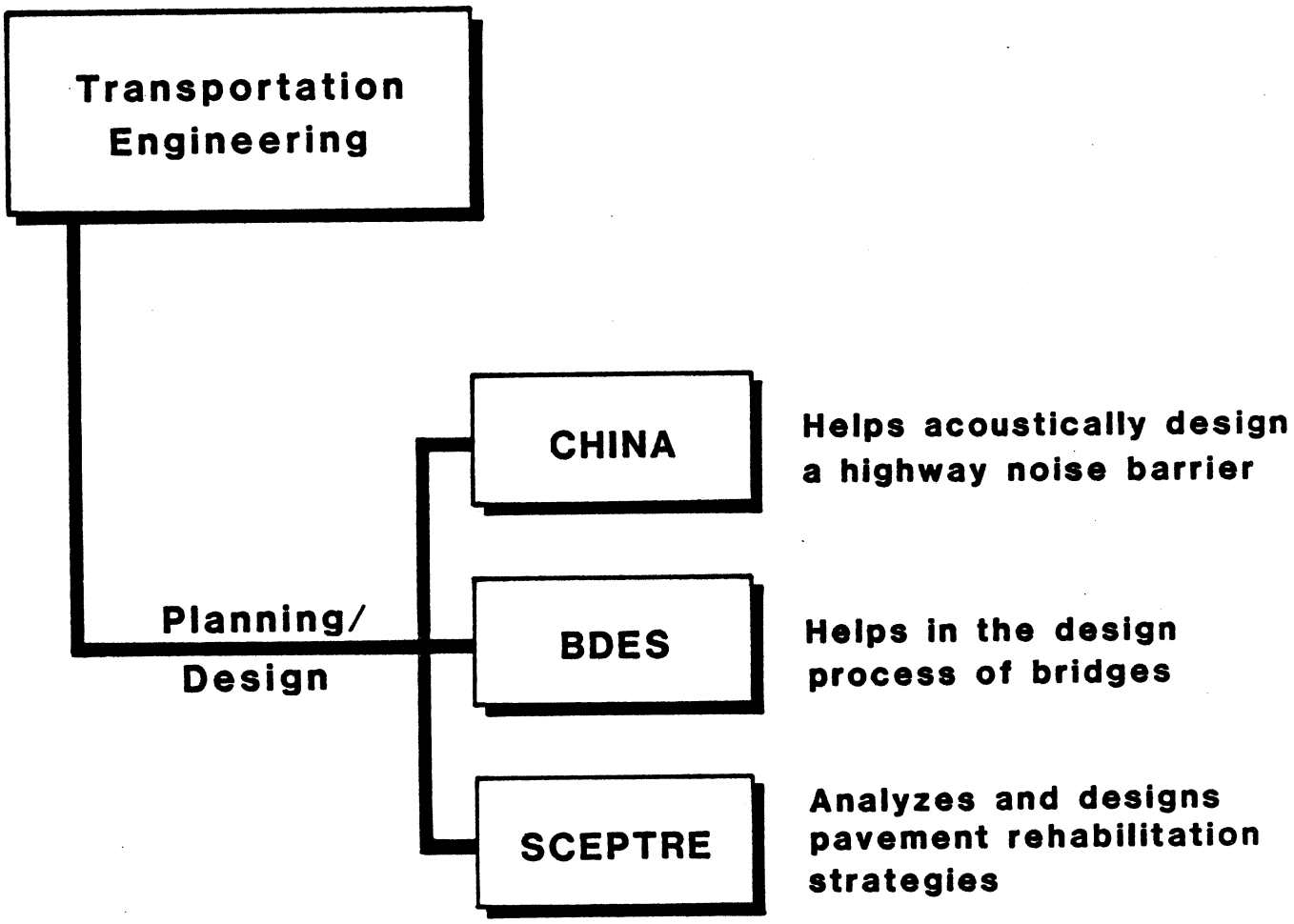


Figure 8. Selected prototype expert systems in transportation.

Figure 9 presents a test case scenario for a noise barrier design denoting barrier segments as numbers and receivers (recipients of the noise impact) as RN #. Using the given scenario, CHINA makes decisions regarding such design considerations as the effect of highway construction on receivers, the dropping of barrier segments, and the raising of ground line barriers. CHINA then recommends a barrier design and gives results on barrier performance for the proposed design. The results of testing CHINA on scenarios such as the one in Figure 9 show that CHINA and human experts design highway noise barriers that are at least equal in terms of performance and cost (11).

Several leading highway noise analysts allowed their expertise to be incorporated into CHINA's knowledge base. Since an expert system to aid with highway noise abatement problems has practical value as a tool to novice noise analysts or experts in the highway noise barrier design process, expert system development of CHINA could be justified.

BDES

The prototype expert system BDES (Bridge Design Expert System) aids the engineer in the bridge design process and demonstrates the potential use of expert systems in engineering design problems. Figure 10 outlines the steps of the bridge design process used to develop the knowledge base for BDES. The first, second, and third steps require only descriptive and factual knowledge related to the specific design problem. In the fourth step, however, design decisions, which are governed by heuristics and rules of thumb, are made (12). Since they include selecting feasible structure types, making appropriate approximations and assumptions, and sizing individual parts to meet the design criteria, they cannot be solved using the conventional algorithmic approaches. Finally, since genuine bridge designers are becoming increasingly scarce, expert system development is both possible and justified for the bridge design process (12).

SCEPTRE

The prototype expert system SCEPTRE (Surface Condition Expert for Pavement Rehabilitation) evaluates pavement surface distress and recommends feasible rehabilitation strategies for detailed analysis and design plans (13).

The pavement rehabilitation process consists of the following four tasks: (1) evaluation of pavement surface condition; (2) analysis and evaluation of structural adequacy; (3) design of alternative strategies, and (4) election of an "optimal" strategy.

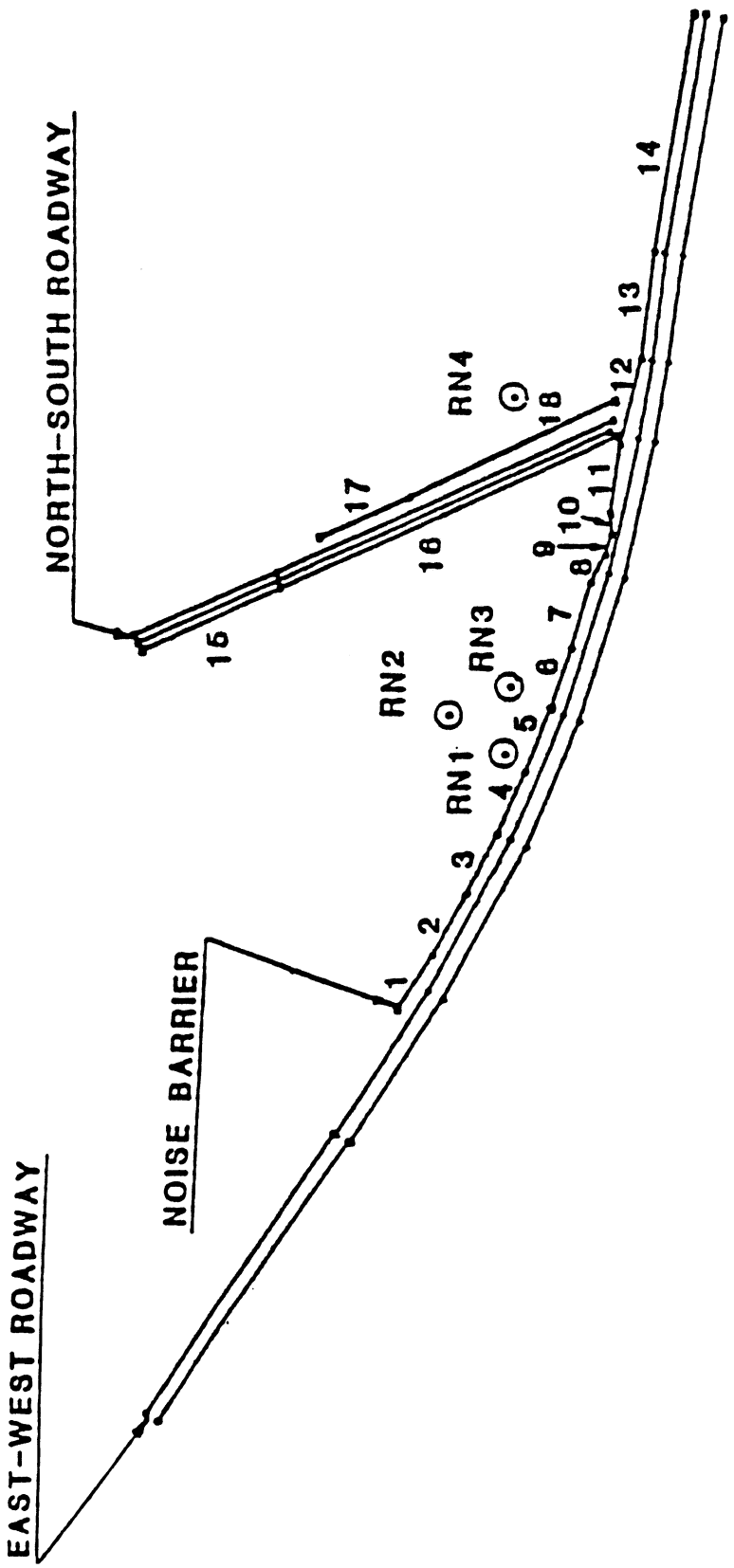


Figure 9. A test case scenario for a noise barrier design (from reference #11)

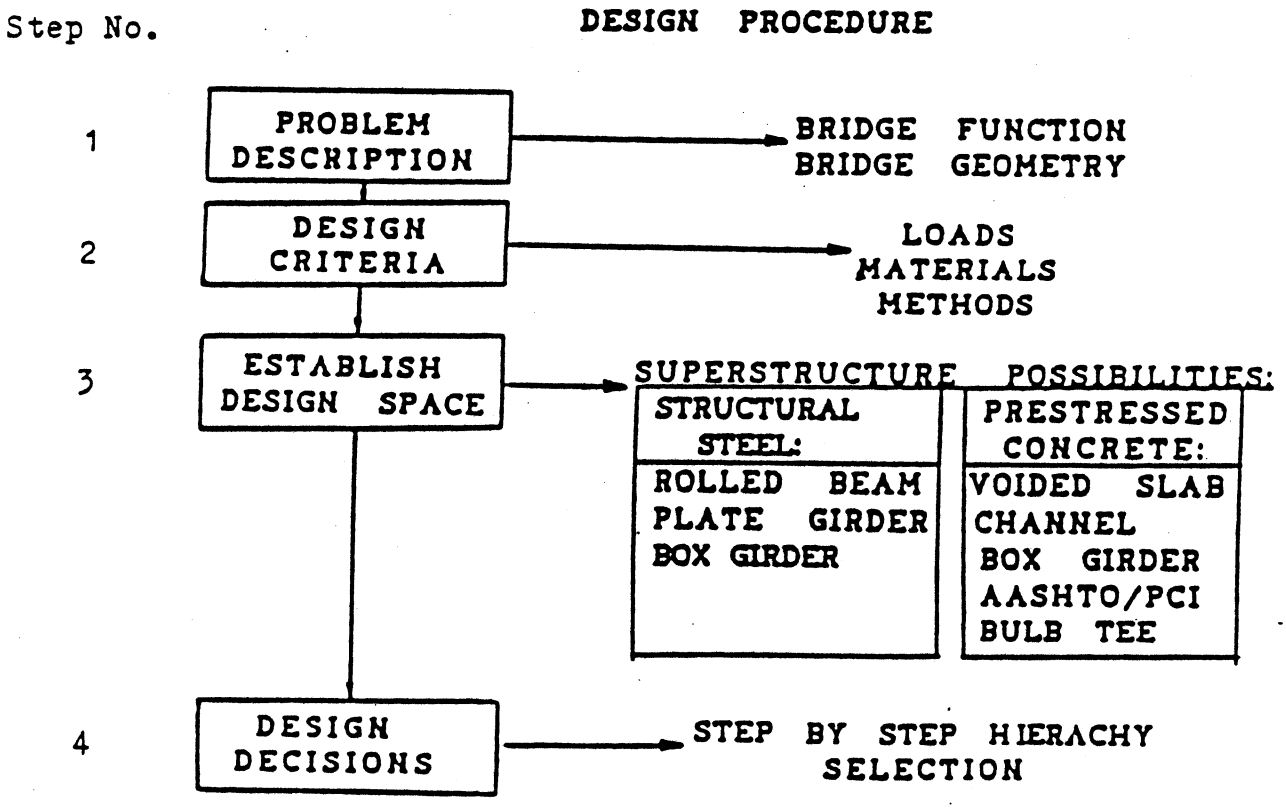


Figure 10. BDES design procedure (from reference #12)

Two pavement specialists combined their expertise to incorporate the four tasks into the knowledge base design of SCEPTRE. As in the previous example, the scarcity of human expertise with experience in pavement rehabilitation justified the development of this system. The knowledge base of SCEPTRE contains multiple estimates, both subjective and data-based, related to strategies performed by human experts.

PROPOSED PROTOTYPE & DEMONSTRATION EXPERT SYSTEM

Background

A review of several studies comparing accidents before and during highway construction demonstrates that there continues to be a traffic safety problem associated with construction zone traffic control practices (14).

In October 1978, the Federal Highway Administration adopted a policy (FHPM-6-4-2-12) requiring that a formal traffic control plan (TCP) be developed for all federal-aid highway construction projects. In response to FHPM-6-4-2-12, several states adopted a consistent policy on work zone TCPs. The policies for some states, however, go beyond the federal requirements. In the state of Texas, for example, a TCP on all construction projects, not just federal-aid projects, is required. The policy also states that a TCP should be utilized on all maintenance activities. A considerable amount of time and effort is being spent by the districts to prepare TCPs.

Traffic control is critical at highway reconstruction work zones. The complexity of reconstruction work zones requires that a variety of traffic handling approaches be utilized. The effectiveness of some of these traffic handling approaches has been evaluated and documented. These results need to be collected so that the best techniques available can be readily adopted.

Purpose

Although agencies responsible for traffic control and work area protection have attempted to develop some guidelines, a coordinated and comprehensive effort to develop greater uniformity is desirable (15). It is believed that an expert system approach for rationalizing the practice at work zones can produce fruitful results.

The prototype system is proposed for the following reasons:

1. All procedures and techniques used for controlling traffic around work zones are not straightforward.

2. Many tasks require heuristic solutions, the use of rules of thumb and expert judgment. (The guidelines only provide the available devices. The final decision making is done by the expert.)

3. Genuine experts exist to develop the system's knowledge base.

4. Many tasks require cognitive skills.

5. Human expertise in the field is not widely available.

6. An expert system can be used by many persons at many locations.

The knowledge base of this proposed prototype expert system will be based on a thorough survey of the literature on existing techniques and approaches for handling traffic around highway construction work zones and interviews with selected experts in charge of planning and design of traffic control strategies for these work zones. During this development stage we will consider: (1) the different types of construction associated with roadways, (2) the different techniques used to control traffic around different types of construction sites, and (3) the different types of roadways involved. The objectives of traffic control in maintenance work zones are: (1) the protection of the freeway user and the work force, (2) the movement of the maximum traffic volume, and (3) efficiency and economy in work procedures.

The users of the expert system will be the personnel in highway agencies, construction companies, utility agencies, and others involved in roadway construction involved in the planning and design of construction zones. The main objectives of the use of the expert system is to establish a uniform, rational practice for the execution of the previously mentioned objectives.

Methodology

The following categories and subcategories of construction types, road type, and techniques will be integrated into the system using tools that will be selected later.

Roadway types: urban freeway, rural freeway, and urban street.

Type 1 construction types: road construction, bridge construction, railway, tramway, water supply, electricity, and gas.

Type 2 construction types: short-term construction, intermittent moving, continuously moving, stop-and-go operations, and slow moving operations.

Types of traffic control: speed zoning (speed control), sequential flashing arrow boards, barricades, lane closures, and warning devices.

The following is an example of a rule in the knowledge base of this system: IF the construction is being conducted on a four-lane divided rural freeway, and the volume of traffic is less than 10,000 vehicles per day, and the construction involves pavement patching, and the length of the construction is 2 days; THEN barricades and flagging are required.

The system will then show how the flagging procedure should be done and where the barricades need to be placed.

CONCLUSIONS

This report introduces transportation professionals to knowledge-based expert systems. Expert systems are an area of artifical intelligence research and they have particular significance for the transportation engineering profession because they use heuristic judgments to supplement analytical computations to reach a solution to a wide range of problems. Only recently have efforts been initiated to bring expert systems into use in transportation engineering. The developers of these systems are not computer scientists or expert systems developers per se, but transportation professionals.

This development is possible owing to the availability of appropriate software. Prototypes are now ready and much activity is taking place, especially at universities in the United States.

This review encourages further investigations by the Research Council into the use of expert systems and alerts the Virginia Department of Transportation to be open minded toward adopting expert systems as they become available. The second phase of this project, which is the development of a prototype expert system, is being initiated. The problem addressed is traffic control through construction zones, and the resulting system will be designed to demonstrate the basic idea of a knowledge-based expert system and to solve this problem.

ACKNOWLEDGEMENTS

The research reported herein was conducted under the funding of Highway Planning and Research through the Virginia Department of Transportation. The authors would like to express their gratitude to all the staff members of the Virginia Transportation Research Council, especially Angela Andrews for providing the necessary references, Roger Howe for editing, Chris Quann for graphics, and Linda Runnett for typing this report.

REFERENCES

1. Charniak, Eugene, and Drew McDermott. 1985. Introduction to Artificial Intelligence. Reading: Addison-Wesley Publishing Company.

2. Harmon, Paul, and David King. 1985. Artificial Intelligence in Business. New York: John Wiley and Sons, Inc.

3. Waterman, Donald A. 1986. A Guide to Expert Systems. Reading: Addison-Wesley Publishing Company.

4. Takallou, Hossein. 1985. Knowledge-based Expert Systems: What's Happening in Transportation Engineering? ITE Journal (October).

5. Brachman, R., Amarel, S., Engleman, C., Engelmore, R., Feigenbaum, E., and Wilkins, D. "What are Expert Systems?" In Building Expert Systems, edited by F. Hayes-Roth, D. A. Waterman, and D. Lenat. Reading: Addison-Wesley, 1983.

6. Carter, Everett C., and Wolfgang S. Hamburger. 1978. Introduction to Transportation Engineering, Reston: Reston Publishing Company, Inc.

7. Barr, A., and E. A. Feigenbaum, eds. 1981. The Handbook of Artificial Intelligence, Vol. 1. Los Altos: William Kaufmann.

8. Watermann, D. A., and Hayes-Roth, F. "An Investigation of Tools for Building Expert Systems". In Building Expert Systems, edited by F. Hayes-Roth, D. A. Waterman, and D. Lenat. Reading: Addison-Wesley, 1983.

9. Brachman, R. J. "On the Epistemological Status of Semantic Networks". In Associative Networks: Representation and Use of Knowledge by Computers, edited by N. V. Findler. New York: Academic Press, 1979.

10. Hayes-Roth, F., Waterman, D. A., and Lenat, D. "An Overview of Expert Systems". In Building Expert Systems, edited by F. Hayes-Roth, D. A. Waterman, and D. Lenat. Reading: Addison-Wesley, 1983.

11. Harris, R. A., L. F. Cohn, and W. Bowlby. 1984. Application and Evaluation of an Expert System Interface to Highway Noise Barrier Design Model. Presented at the Transportation Research Board meeting, January, 1985, Washington, D. C.

12. Welch, James G., and Mrinmany Biswas. Application of Expert Systems in the Design of Bridges. Presented at the Transportation Research Board meeting, January, 1986, Washington, D. C.

13. Ritchie, Stephen G., Che-I Yeh, Joe P. Mahoney and Newt Jackson. Development of an Expert System for Pavement Rehabilitation Decision-Making. Presented at the Transportation Research Board annual meeting, January, 1986, Washington, D. C.

14. Texas Transportation Institute. 1985. Handling Traffic in Work Zones.

15. Federal Highway Department. 1978. Federal Highway Program Manual N-6.4.2.12.