REFERENCES


GLOSSARY

**Algorithm:** A set of processing steps for solving a specific problem according to a precisely defined procedure.

**Antecedent:** In logic, the first part of an *if... then* statement is the antecedent; the second part is the consequence. In the statement, *if* the temperature falls below 70 degrees, *then* turn on the furnace. The part about the temperature is the antecedent; the part about the furnace is the consequence. This type of logic is used to develop rules for many AI systems.

**Architecture:** (1) The organizing framework imposed upon knowledge applications and problem solving activities. (2) The knowledge-engineering principles that govern selection of appropriate frameworks for specific expert systems.

**Arguments:** Parameters of a predicate.

**Artificial intelligence:** The sub-field of computer science concerned with developing intelligent computer programs. This includes programs that can solve problems, learn from experience, understand language, interpret visual scenes, and, in general, behave in a way that would be considered intelligent if observed in a human.

**Attributes:** The properties or qualities of an object.

**Backtracking:** A prolog process in which the program backs up through a Sequence of inferences, usually to try to find another path to a Solution.

**Backward chaining:** A type of reasoning process that begins at a specified goal and works backward in an attempt to prove that the goal is true. (See forward chaining.)

**Binding:** The process of assigning a value to a variable.

**Blackboard:** Work area in an expert system which is used by the inference mechanism to record solution paths and results that can be considered during the consultation in progress.

**Blackboard architecture:** A way of representing and controlling knowledge based on using independent groups of rules called *knowledge sources* that communicate through a central database called a *black board*.

**Break:** The connection between the head and body of a rule. Represented by the `:-` symbol.

**Built-in predicates:** See *standard predicates*. 

201
Certainty: A measure of confidence in a proposition, hypothesis, or rule in an expert system.

Clause: A fact or a rule, ending with a period.

Compiler: A language tool that converts a source program to a more efficient form for later execution.

Compound goal: A goal that contains at least two sub-goals.

Consultation paradigm: A type of user interface in which the user is moved through the problem space to the goal using a series of interactive questions.

Control: A procedure that determines the order of problem-solving activities done by an expert system.

Control Key: A key on a keyboard that is held down while another key is pressed to give the other key a different function.

Cut: A Prolog operator that, when invoked, succeeds and commits Prolog to use all choice and variable bindings made up to that point in the evaluation of the group of clauses (predicates). A cut prevents backtracking beyond the point where the cut is placed.

Database: A collection of known facts and rules about a particular domain. (see dynamic database: static database.)

Decision tree: A map of the reasoning process of an expert system showing questions as nodes and the possible answers as links.

Demon: A procedure activated by the changing or accessing of values in a database.

Diagnostic system: A type of expert system used to relate symptoms to causes.

Domain: A narrowly defined problem area.

Domain expert: A person who, through years of training and experience, has become extremely proficient at problem solving in a particular domain.

Domain knowledge: Knowledge about the problem domain, e.g., knowledge about geology in an expert system for finding mineral deposits.

Drivers: In software, drivers are special programs that interface external devices with the operating system of a computer. Drivers provide communications between the computer and keyboards, monitors, disk drivers, printers, plotters, and other external devices. Drivers are often written in machine language.

Dynamic database: That part of the database to which facts can be added or removed while the program is executing. Facts can be added using the asserta or
assertz predicates and Removed with the retract predicate. (see database: static database.)

End-user: The person who uses the finished expert system: the person for whom the system was developed.

Evaluation function: A procedure used to determine the value or worth of proposed intermediate steps during a hunt through a search space for a solution to a problem.

Exhaustive search: A problem-solving technique in which the problem solver systematically tries all possible solutions in some "brute force" manner until it finds an acceptable one.

Expertise: expert knowledge needed for solving specific problems.

Expert system: An intelligent computer program that uses knowledge and inference procedures to solve problems difficult enough to require significant human expertise for their solution.

Expert-system-building tool: The programming language and support package used to build the expert system.

Explanation: A description of the desirability of something, based on the presentation of goals, laws, and heuristic rules.

Explanation component: Feature of an expert system. The explanation component of an expert system must be able to trace the solution steps and supply reasons why a solution path was taken. In addition, it supplies explanations to system questions and deduced facts and can specify object attributes.

Explanation facility: That part of an expert system that explains how solutions were reached and justifies the steps used to reach them.

Explanatory interface: That part of an expert system that permits a user to query into the Module reasoning process used by the system in reaching a particular goal.

Fact: A statement considered to be valid; a fact exists in an expert system if it is included in the knowledge base.

Firing: A process in a production-system inference cycle in which a conclusion is accepted as true (all conditions or sub-goals have been proven true), and the conclusion is added to the dynamic database.

Formal reasoning: The process of solving a problem by inference or IF-THEN rules.
**Forward chaining:** A reasoning process that begins with the known facts and works forward, trying to find a successful goal from the known facts. (see *backward chaining*.)

**Functor:** The relationship part of a predicate or the parent objective of a compound Structure.

**Goal:** One or more relations, each involving objects or variables, that a prolog Program must prove as true.

**Goal-directed systems:** A system using backward chaining.

**Head:**
1. A conclusion of a rule or
2. the first item in a list.

**Heuristic:** A rule of thumb or any device that reduces the problem space for a specific problem solution. Heuristics do not guarantee a solution, nor (if a solution exists) do not they guarantee the most efficient path for a solution. (see *algorithm*.)

**Heuristic rule:** A procedural tip or incomplete method for performing some task. [Also *heuristic*.]

**Inference:** A process in which new facts are determined from known facts using rules and *modus ponens*.

**Inference chain:** The sequence of steps or rule applications used by a rule-based system to reach a conclusion.

**Inference engine:** That part of an expert system that infers new facts from existing facts by using rules in the database. The inference engine also controls the flow and order of the inference.

**Inference mechanism:** Part of an expert system that draws inferences from a knowledge base according to a fixed problem-solving method. The functions of the inference mechanism include controlling the actions between the individual parts of the expert system, determining the time for and the type of rule processing, controlling the dialogue with the users.

**Inference method:** The technique used by the inference engine to access and apply the domain knowledge, e.g., forward chaining and backward chaining.

**Inference strategy:** Method used by the inference mechanism in problem solving, e.g., backward chaining.

**Instantiate:** The process of binding or assigning a value to a variable.
Interface: In hardware, an interface is a card or a device that allows transfer of data between the computer and peripherals. In software, an interface is a program, often called the human interface, which employs prompts, menus typed entries, and commands to let a person use a total software environment. A human interface often incorporates an editor or can be used with an editor.

Interpreter: In an expert system, that part of the inference engine that decides how to apply the domain knowledge. In a programming system that part of the system that analyzes the code to decide what actions to take next.

Knowledge: Facts and relationships about a domain that can be used to solve problems within that domain.

Knowledge acquisition: The process of extracting, structuring, and organizing knowledge from some source, usually human experts, so it can be used in a program.

Knowledge base: The portion of a knowledge-based system or expert system that contains the domain knowledge.

Knowledge-based system: A program in which the domain knowledge is explicit and separate from the program's other knowledge.

Knowledge engineer: The person who designs and builds the expert system. This person is usually a computer scientist experienced in applied artificial intelligence methods.

Knowledge representation: The process of structuring knowledge about a problem in a way that makes the problem easier to solve.

Knowledge Source: The entire set of knowledge for a defined domain, especially a subject area. There are some gray areas in the definition that include: (1) the experts who supply the knowledge; (2) the codification that makes knowledge applicable to an expert system; and (3) the degree to which knowledge is relevant to specific problem.

Meta-knowledge: Knowledge about knowledge.

Modus ponens: A rule of formal logic that asserts that if A implies B and A is true, then B is true.

Multiple lines of reasoning: A problem-solving technique in which a limited number of possibly independent approaches to solving the problem are developed in parallel.

NIL: An empty list, designated by brackets [ ].

.205
**Predicate:** A function with a value of true or false that expresses a property of relationship.

**Problem reformulation:** Converting a problem stated in some arbitrary way to a form that lends itself to a fast, efficient solution.

**Procedure-oriented methods:** Programming methods using nested subroutines to organize and control program execution.

**Production rule:** The type of rule used in a production system, usually expressed as IF condition THEN action.

**Production system:** A type of rule-based system containing IF-THEN statements with conditions that may be satisfied in a data-base and actions that may change the data-base.

**PROLOG:** PROgramming in LOGic; an implementation of 1st order predicate calculus as a programming language. At present, prolog is widely used in Japan and European countries, Lisp is widely used in the United States. In particular, the fifth-generation computers being developed in Japan use prolog. A. Colmerauer and P. Proussel developed prolog in 1973 at the university of Marseille AI laboratory. Later, AI experts at the university of Edinburgh in Great Britain added several logic-programming features.

**Prototype:** A preliminary version of a system. With expert systems, the prototype represents a much smaller domain than the final system does.

**Pruning:** The process of making the problem space smaller by removing one or more branches of the decision tree.

**Recursion:** A method in which an entity is defined in terms of itself. In prolog, a recursive clause defined in terms of itself.

**REX:** Regression expert, a frame-based expert system that does statistical analysis, developed by Bell Laboratories.

**Rule:** A formal way of specifying a recommendation, directive, or strategy, expressed as IF premise THEN conclusion or IF condition THEN action.

**Rule base:** That part of an expert system used to store static information about a particular domain (knowledge that does not change during a particular consultation). In prolog, the static database is the rule base. (see static database).

**Rule based methods:** Programming methods using If-THEN rules to perform forward or backward chaining.
Rule-based system: A program organized as a set of rules. See rules.
Scheduler: The part of the inference engine that decides when and in what order to apply different pieces of domain knowledge.
Search: The process of looking through the set of possible solutions to a problem in order to find an acceptable solution.
Search space: The set of all possible solutions to a problem.
Semantic: Pertaining to the meaning, intention, or significance of a symbolic expression, as opposed to its form. [Contrast syntactic.]
Skill: The efficient and effective application of knowledge to produce solutions in some problem domain.
Standard predicates: Prolog predicates that are integral to a specific implementation.
Static database: That part of the database that does not change during program execution. (see database; dynamic database.)
Sub goal: A goal that is also part of another goal.
Suffix notation: Mathematical format in which the operator in a mathematical expression is placed after the operands or values on which it operates. (see infix notation; prefix notation.)
Support environment: Facilities associated with an expert-system-building tool that help the user interact with the expert system. These may include sophisticated debugging aids, friendly editing programs, and advanced graphic devices.
Symbol: A string of characters that sends for some real-world concept.
Symbol-manipulation language: A computer language designed expressly for representing and manipulating complex concepts, e.g., LISP and PROLOG.
Symbolic reasoning: Problem solving based on the application of strategies and heuristics to manipulate symbols standing for problem concepts.
Syntax: The rules governing the order of the symbols in an expression. Computers are good at resolving syntax, but are not good at interpreting semantics. (see semantics.)
Syntax: The form of structure of a symbolic expression (statement). See also Semantics and Lexicon.
Tail: The part of a list that remains if the head is removed. The tail of a one-item list is a NIL or empty list and is expressed by empty brackets [].
Tool: A shorthand notation for expert-system-building tool.
**Tool Kit:** In an environment, a set of programs used to create applications software. The exact nature of the programs varies from one system to another.

**Unification:** A process in which prolog tries to match a sub goal with the facts and heads of other rules to prove the sub goal.

**User:** A person who uses an expert system, such as an end-user, a domain expert, a knowledge engineer, a tool builder, or a clerical staff member.

**Value:** An attribute quantity or quality. Objects have attributes, and attributes have values.

**Variable:** Any name used to represent a value.

**Window:** In relation to the software that operates a monitor or terminal, a window is a contiguous area for display of information, prompts, menus and such. A window can be the same size as a screen. Thus, a screen might appear to have just one window. A window can be smaller than a screen; then several windows can be displayed. These windows often overlap and some procedure is used to pop a window to the surface. A window can be larger than a screen, thus displaying a portion of the window. One hears about pop up windows, scrollable windows, expandable windows, and such.

**Windowing:** The sectioning of the display screen into one or more viewing areas so that different types of data can be viewed at the same time.

**Working memory:** That part of the expert system that stores facts and conclusions pertaining to a specific consultation. In prolog, working memory corresponds to the dynamic database.