INTRODUCTION TO KNOWLEDGE ENGINEERING

COURSE REVIEW

Sistem Berbasis Pengetahuan

Agenda

- What is knowledge?
- Types of knowledge
- Knowledge engineering
- Knowledge engineers

Philosophical Basis

- Traditional questions that have been analyzed by philosophers, psychologists, and linguist:
  - What is knowledge?
  - What do people have inside their head when they know something?
  - Is knowledge expressed in words?
  - If so, how could one know things that are easier to do than to say, like tying a shoestring or hitting a baseball?
  - If knowledge is not expressed in words, how can it be transmitted in language?
  - How is knowledge related to the world?
  - What are the relationships between the external world, knowledge in the head, and the language used to express knowledge about the world?

Philosophical Basis

- With the advent of computers, the questions addressed by the field of artificial intelligence (AI) are:
  - Can knowledge be programmed in a digital computer?
  - Can computers encode and decode that knowledge in ordinary language?
  - Can they use it to interact with people and with other computer systems in a more flexible or helpful way?

Information Processing Views of Knowledge

- Hierarchical view: data → information → knowledge
  - Information is the input or raw material of new knowledge
  - Knowledge is authenticated/personalized information

- Reversed hierarchical view: knowledge → information → data
  - Knowledge must exist before information can be formulated and before data can be collected

- Non-hierarchical view: data → information
  ↑
  Knowledge
  - Knowledge is needed in converting data into information
  - Knowledge is the accumulation of experiences vs. knowledge is created through conjectures and refutations.

Alternative Perspectives on Knowledge

- Knowledge can be defined as a justified belief that increases an entity's capacity for effective action.
- It may be viewed from several perspectives:
  - (1) a state of mind – knowledge is the state of knowing and understanding
  - (2) an object – knowledge is an object to be stored and manipulated
  - (3) a process – knowledge is a process of applying expertise
  - (4) a condition – knowledge is organized access to and retrieval of content
  - (5) a capability – knowledge is the potential to influence action

Taxonomies of Knowledge

- Tacit vs. explicit
  - Explicit knowledge refers to knowledge that is transmittable in formal, systematic language
  - Tacit knowledge is deeply rooted in actions, experience, and involvement in a specific context. It consists of cognitive element (mental models) and technical element (know-how and skills applicable to specific work).
- Individual vs. social
  - Individual knowledge is created by and exists in the individual whereas social knowledge is created by and exists in the collective actions of a group.

Taxonomies of Knowledge

- Five Types of Knowledge
  - Declarative knowledge
    - Know-about
  - Procedural knowledge
    - Know-how
  - Causal knowledge
    - Know-why
  - Conditional knowledge
    - Know-when
  - Relational knowledge
    - Know-with
- Meta-knowledge
  - Knowledge about knowledge

Four Modes of Knowledge Conversion (Nonaka 1994)

From \ *To*	Tacit knowledge	Explicit knowledge
Tacit knowledge	Socialization	Externalization
Explicit knowledge	Internalization	Combination

Knowledge Engineering

- An engineering discipline that involves integrating knowledge into computer systems in order to solve complex problems normally requiring a high level of human expertise (Feigenbaum and Pamela, 1983)
- It normally involves five distinct steps in transferring human knowledge into some form of knowledge based systems (KBS)

Five Steps of Knowledge Engineering

- Knowledge acquisition
- Knowledge validation
- Knowledge representation
- Inferencing
- Explanation and justification

Two Main Views of Knowledge Engineering

- Transfer view – This is the traditional view. In this view, the key idea is to apply conventional knowledge engineering techniques to transfer human knowledge into the computerized system.
- Modeling view – In this view, the knowledge engineer attempts to model the knowledge and problem solving techniques of the domain expert into the computerized system.

Knowledge Engineering (KE) vs. Knowledge Management (KM)

- KE is primarily concerned with constructing a knowledge-bases system while KM is primarily concerned with identifying and leveraging knowledge to the organization's benefit.
- KE and KM activities are inherently interrelated.
- Knowledge engineers are interested in what technologies are needed to meet the enterprise's KM needs.

Knowledge Engineers

- A knowledge engineer is responsible for obtaining knowledge from human experts and then entering this knowledge into some form of KBS.
- In developing KBS, the knowledge engineer must apply methods, use tools, apply quality control and standards, plan and manage projects, and take into account human, financial, and environmental constraints.
- Required skills of a knowledge engineer
  - Knowledge representation
  - Fact finding (knowledge elicitation)
  - Human skills
  - Visualization skills
  - Analysis
  - Creativity
  - Managerial

KNOWLEDGE-BASED SYSTEMS

COURSE REVIEW

Agenda

- Expert systems
- Neural networks
- Case-based reasoning
- Genetic algorithms
- Intelligent agents

What are KBSs?

- A knowledge based system is a system that uses artificial intelligence techniques in problem-solving processes to support human decision-making, learning, and action.
- Two central components of KBSs are
  - Knowledge base
    - Consists of a set of facts and a set of rules, frames, or procedures
  - Inference engine
    - Responsible for the application of knowledge base to the problem on hand.
- There are pros and cons of using KBSs, compared to human expertise.

Types of KBSs

- Expert systems
- Neural networks
- Case-based reasoning
- Genetic algorithms
- Intelligent agents

Expert Systems

- An expert system is a computer program designed to emulate the problem-solving behavior of an expert in a specific domain of knowledge
- In order to qualify as an expert system, a system must have the capability of explaining or justifying its conclusions.
- A system which can explain its reasoning process is said to demonstrate meta-knowledge (knowledge about its own knowledge).

Features of Problem Solvers

- Human experts exhibit certain characteristics and techniques which help them perform at a high level in solving problems in their domain:
  - Solve the problem
  - Explain the result
  - Learn
  - Restructure knowledge
  - Break rules
  - Determine relevance
  - Degrade gracefully

Characteristics of Expert Systems

- The system performs at a level generally recognized as equivalent to that of a human expert or specialist in the field.
- The system is highly domain specific.
- The system can explain its reasoning.
- If the information with which it is working is probabilistic or fuzzy, the system can correctly propagate uncertainties and provide a range of alternative solutions with associated likelihoods.

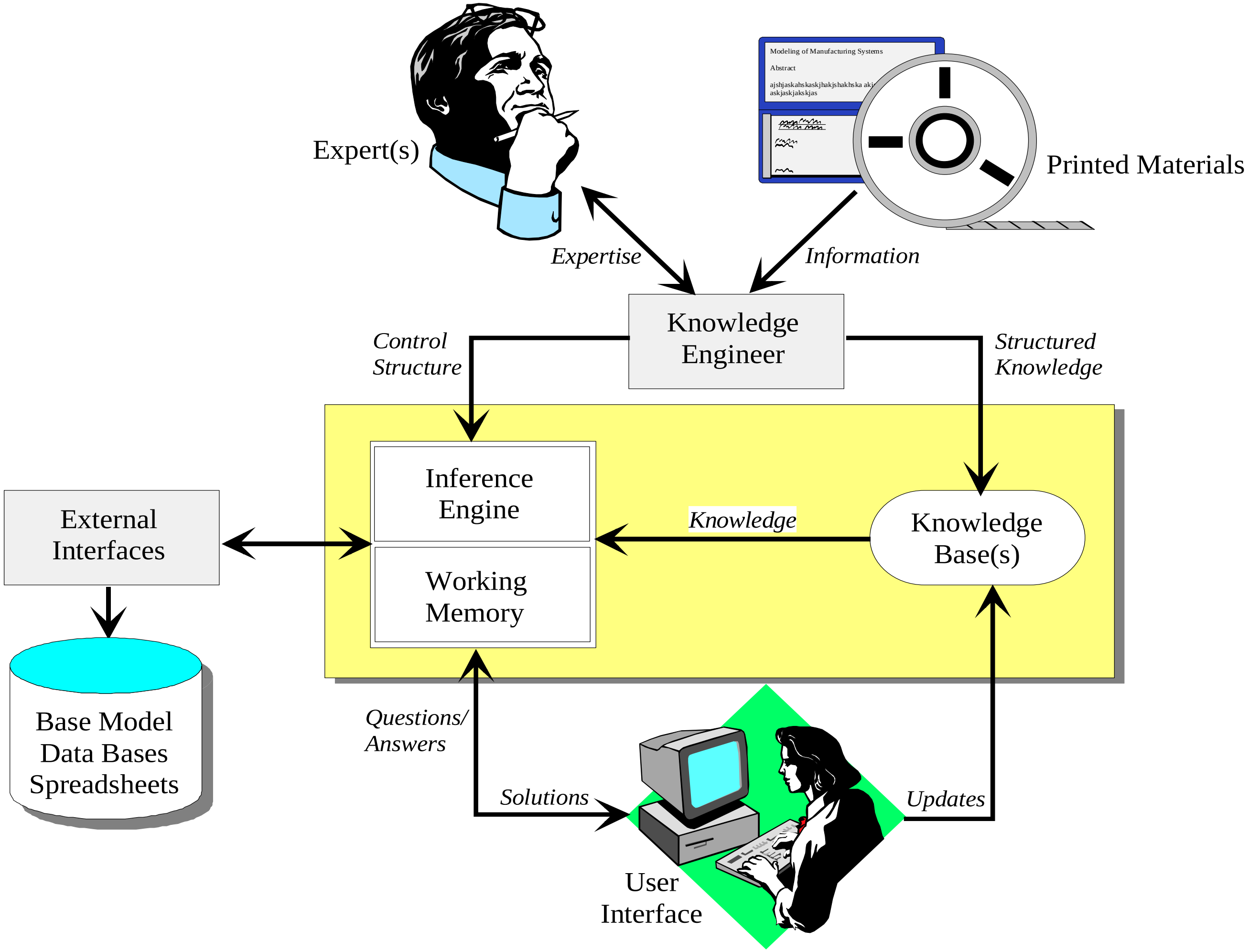
Applications of Expert Systems

- DENDRAL
  - Applied knowledge (i.e., rule-based reasoning)
  - Deduced likely molecular structure of compounds
- MYCIN
  - A rule-based expert system
  - Used for diagnosing and treating bacterial infections
- XCON
  - A rule-based expert system
  - Used to determine the optimal information systems configuration
- New applications: Credit analysis, Marketing, Finance, Manufacturing, Human resources, Science and Engineering, Education, …

Components of Expert Systems

- Knowledge base
  - Consists of facts and rules
  - Rules are commonly expressed in if-then structure (production rules)
    - If-premise then conclusion
    - If-condition then action
- Inference engine
  - Responsible for rule interpretation and scheduling
  - Forward chaining vs. backward chaining
- User interface
- Working memory
- Explanation facility

Conceptual Architecture of a Typical Expert Systems



Expert System Building Tools

- Programming language
  - An expert system can be implemented using a general purpose programming language. However, the programming language LISP and PROLOG are typically used in expert systems implementation, in particular Artificial intelligence applications.
- Shells
  - A shell consists mainly of an inference engine and an editor to assist developers in building their knowledge base.
  - Example: CLIPS is an expert system shell developed by NASA

Strengths and Limitations of Expert Systems

- Strengths
  - Human expertise can be expensive
  - Human advice can be inconsistent
  - Human knowledge may be lost
  - Human knowledge can only be accessed in one place at one time
- Limitations
  - Lack of common sense
  - Lack of inspiration or intuition
  - Lack of flexibility

Neural Networks

- Neural networks represent a brain metaphor for information processing. Neural computing refers to a pattern recognition methodology for machine learning. The resulting model from neural computing is often called an artificial neural network (ANN) or neural network (NN).
- Due to their ability to learn from the data, their nonparametric nature (i.e., no rigid assumptions), and their ability to generalize, neural networks have been shown to be promising in many forecasting and business classification applications.

Basic Concepts of Neural Networks

- The human brain is composed of special cells called nuerons.
- Neural network elements
  - **Nucleus**

    The central processing portion of a neuron
  - **Soma**

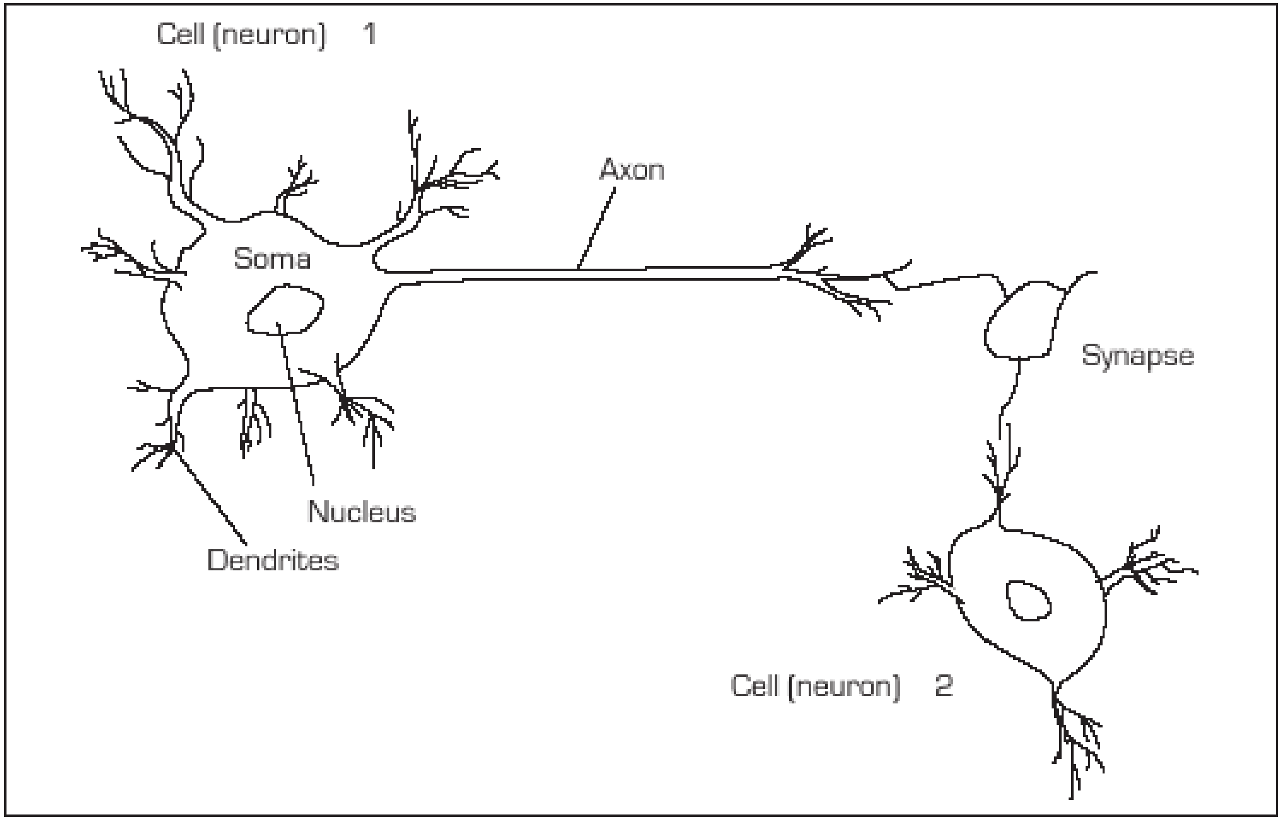
    The main body of the neuron in which the cell nucleus is contained
  - **Dendrite**

    The part of a biological neuron that provides inputs to the cell
  - **Axon**

    An outgoing connection (i.e., terminal) from a biological neuron
  - **Synapse**

    The connection (where the weights are) between processing elements in a neural network

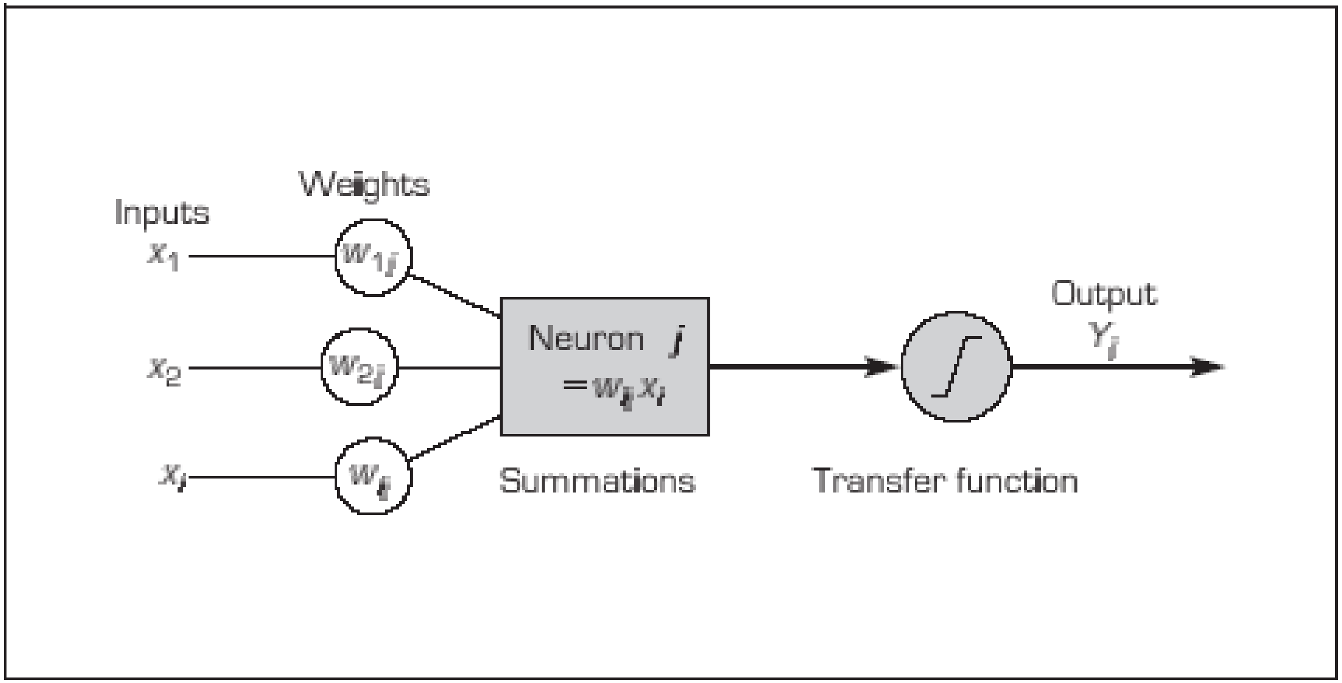
Structure of a Biological Neural Network



Artificial Neural Network

- An ANN model emulates a biological neural network.
- Neural concepts are usually implemented as software simulations of the massive parallel processes that involve processing elements (also called artificial neurons) interconnected in a network structure.
- Connections between neurons have an associated weight.
- Each neuron calculates a weighted sum of the incoming neuron values, transforms this input, and passes on its neural value as the input to subsequent neurons or external outputs.

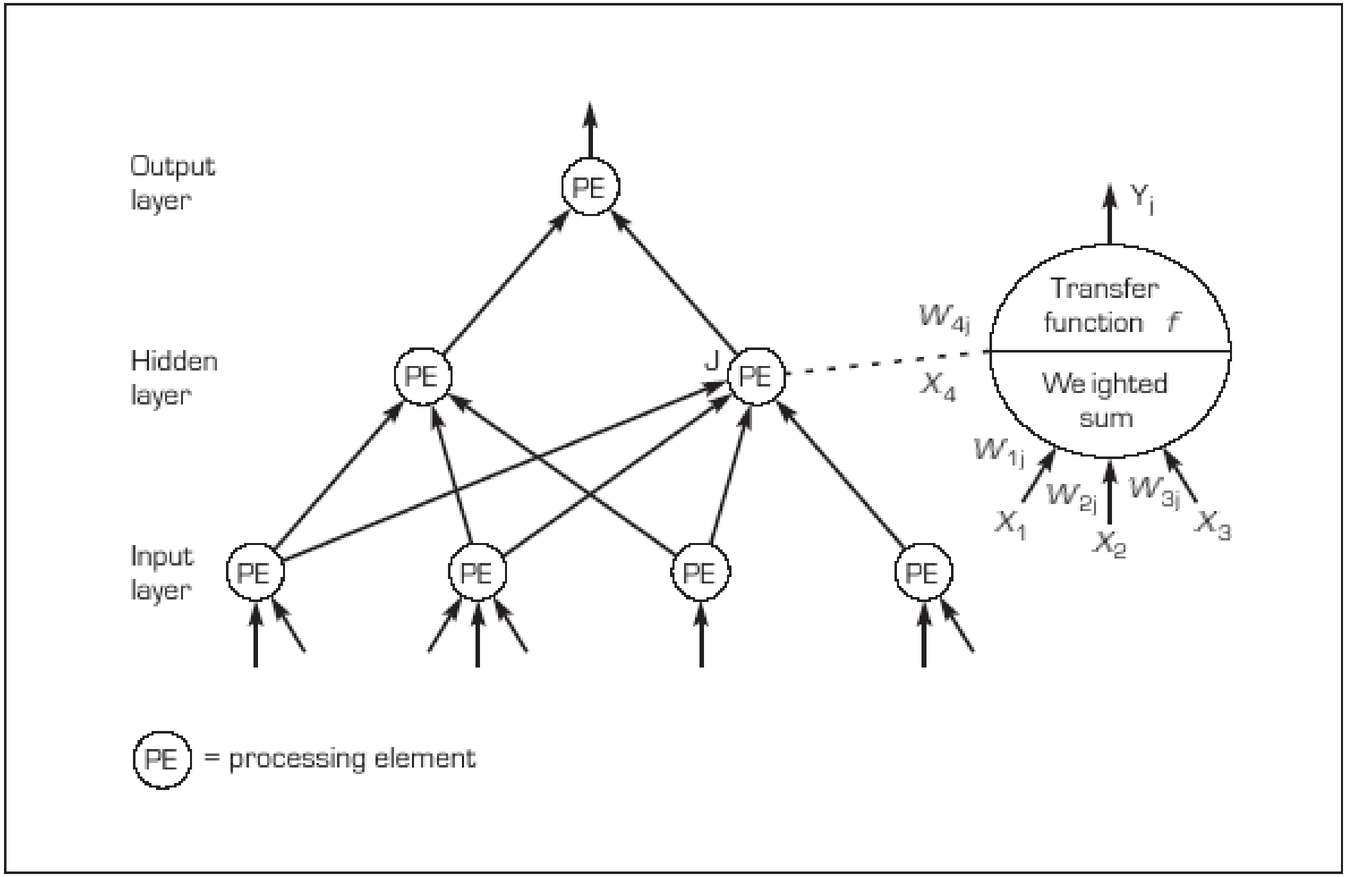
Processing Information in an Artificial Neuron



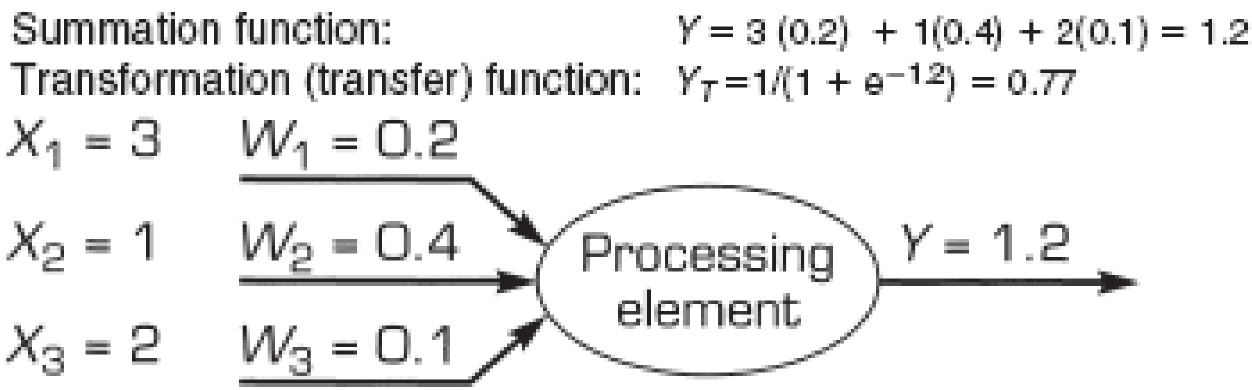
The Relationship Between Biological and Artificial Neural Networks

Biological	Artificial
Soma	Node
Dendrites	Input
Axon	Output
Synapse	Weight
Slow speed	Fast speed
Many neurons (10^9)	Few neurons (a dozen to hundreds of thousands)

Neural Network with One Hidden Layer



Example of ANN Functions



Learning in ANN

- Supervised learning
  - Uses a set of inputs for which the desired outputs are known
  - Example: Backpropagation algorithm
- Unsupervised learning
  - Uses a set of inputs for which no desired output are known.
  - The system is self-organizing; that is, it organizes itself internally. A human must examine the final categories to assign meaning and determine the usefulness of the results.
  - Example: Self-organizing map

Characteristics of ANNs

- Adaptive learning
- Self-organization
- Error tolerance
- Real-time operation
- Parallel information processing

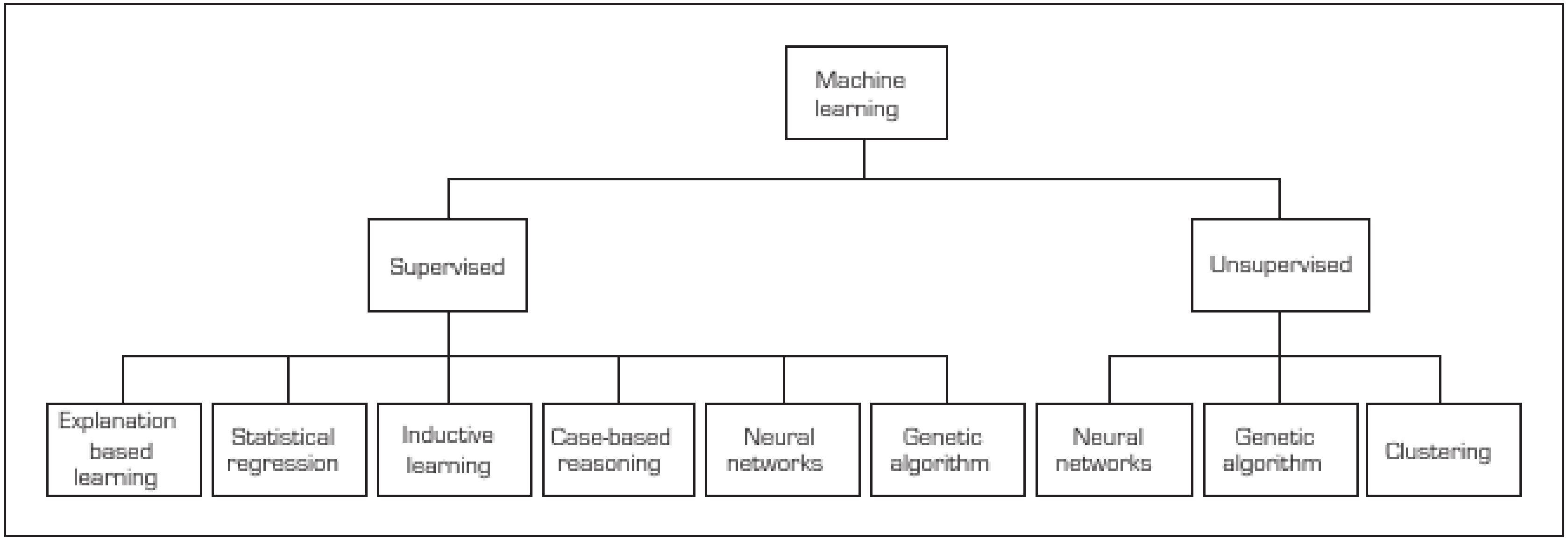
Benefits and Limitations of Neural Networks

- Benefits
  - Ability to tackle new kinds of problems
  - Robustness
- Limitations
  - Performs less well at tasks humans tend to find difficult
  - Lack of explanation facilities
  - Requires large amounts of test data

Machine Learning Methods

- **Machine learning**

  The process by which a computer learns from experience (e.g., using programs that can learn from historical cases)



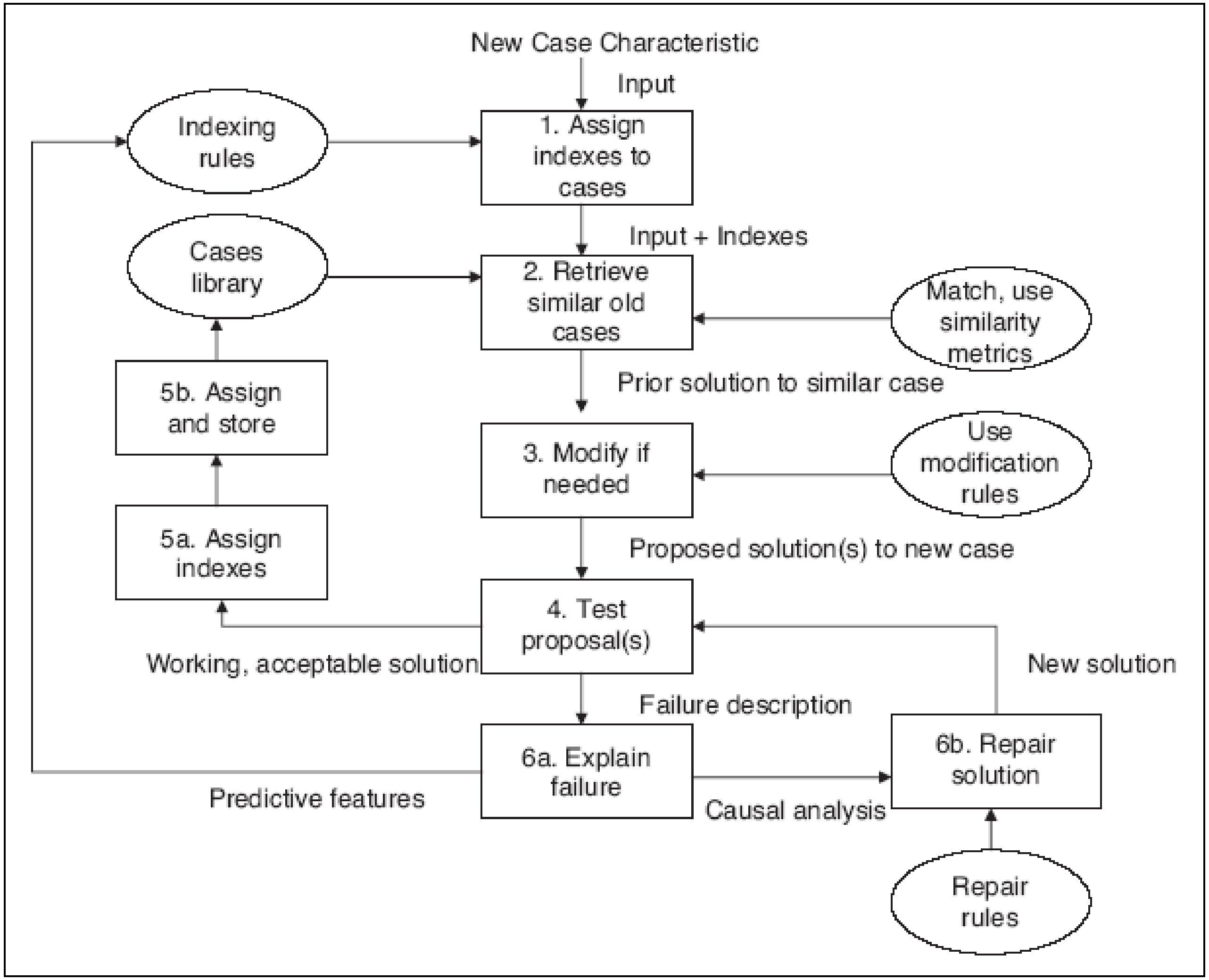
Case-Based Reasoning (CBR)

- A case has two parts: a problem and a solution
- Cases represent experience; that is, they record how a problem was solved in the past
- CBR is a methodology in which knowledge and/or inferences are derived from historical cases. It is based on the premise that new problems are often similar to previously encountered problems and that, past solutions may be of use in the current situations.
- CBR is particularly applicable to problems in which the domain is not understood well enough for a robust statistical model or system of equations to be formulated.

Process of CBR

1. Retrieve
   - Given a target problem, retrieve the most similar cases
2. Reuse
   - Map the solution and reuse the best old solution to solve the current case
3. Revise
   - Test the solution and, if necessary, revise the old case to come up with the solution
4. Retain
   - After the solution has been successfully adapted to the target problem, store the resulting experience as a new case

Step-by-Step Process of CBR



Similarity Computation

- Cases are ranked according to their similarity based on the similarity of each feature
- The degree of similarity can be expressed by a real number between 0 (not similar) and 1 (identical).
- The importance of different features may be different. In that case, similarity is computed by weighted average.

CBR Examples

- Intelligent customer support and sales support
- Retrieval of tour packages from travel catalogs
- Conflict resolution in air traffic control
- Conceptual building design aid
- Conceptual design aid for electronic devices
- Medical diagnosis
- Aircraft troubleshooting
- Heuristic retrieval of legal knowledge
- Computer supported conflict resolution through negotiation or mediation

Advantages and Disadvantages of Using CBR

- Advantages
  - Improved knowledge acquisition
  - Reduced development time
  - Easier explanation
  - Learning over time
- Disadvantages
  - Storing of cases in the KB.
  - Implicit link between problem and solution
  - Access and retrieval speed

Genetic Algorithms

- Programs that attempt to find optimal solutions to problems by conceptually following steps inspired by the biological processes of evolution
- The method learns by producing offspring that are better and better, as measured by a fitness-to-survive function, until an optimal or near-optimal solution is obtained.

Genetic Algorithm Fundamentals

- **Chromosome**

  A candidate solution for a genetic algorithm

- **Fitness function**

  A measure of the objective to be obtained.

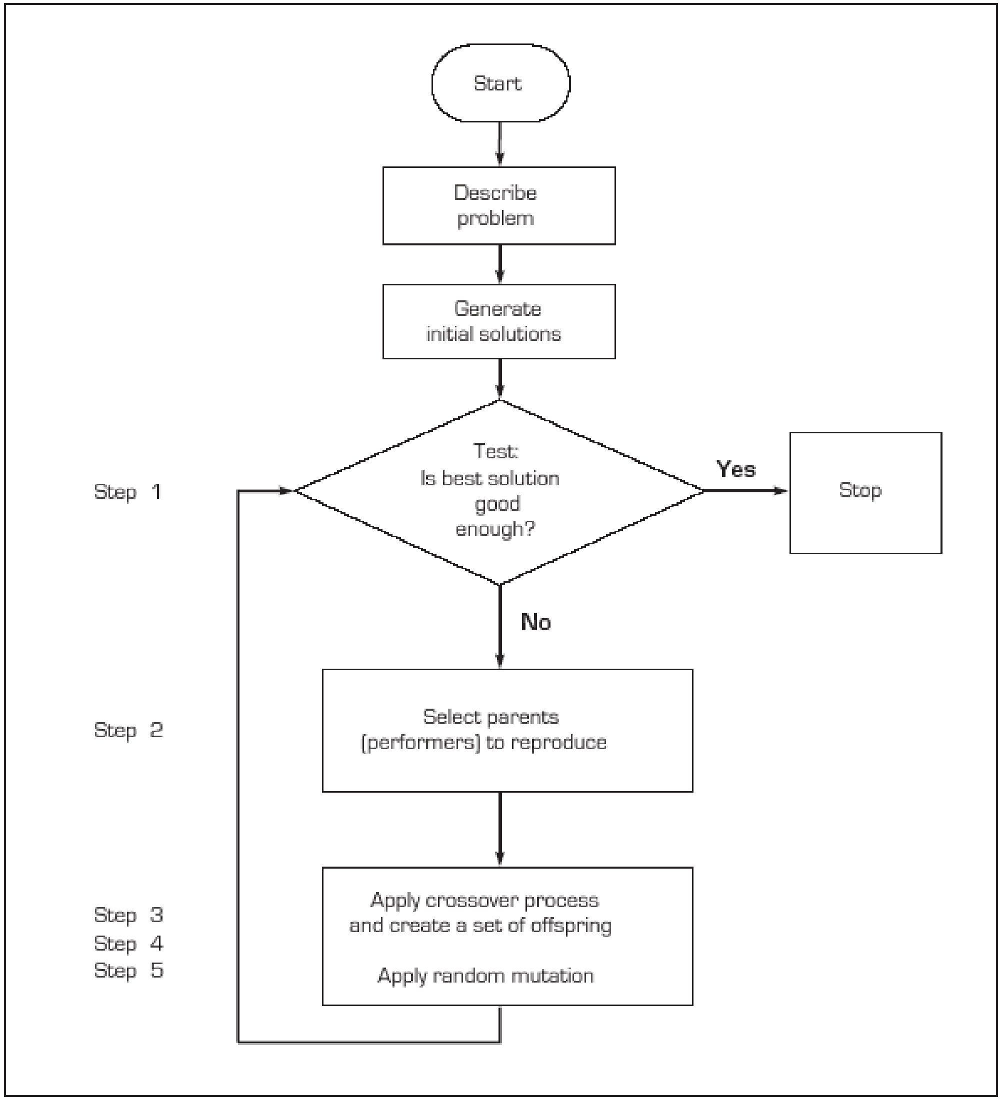
- **Generation**

  An iteration of the genetic algorithmic process in which candidate solutions are combined to produce offspring

Processes within Genetic Algorithm

- **Reproduction**
  - Through reproduction, genetic algorithms produce new generations of improved solutions by selecting parents with higher fitness ratings or by giving such parents a greater probability of being contributors and by using random selection.
- **Crossover**
  - The combining of parts of two superior solutions by a genetic algorithm in an attempt to produce an even better solution
- **Mutation**
  - A genetic operator that causes a random change in a potential solution

Genetic Algorithm Process



Genetic Algorithm Parameters

- Some parameters must be set for the genetic algorithm
  - Number of initial solutions to generate
  - Number of offspring to generate
  - Number of parents and offspring to keep for the next generation
  - Mutation probability
  - Probability distribution of crossover point occurrence
- Their values are dependent on the problem being solved and are usually determined through trial and error

Genetic Algorithm Benefits and Limitations

- Genetic algorithms are particularly useful for complex problems that require rapid development of set of good solutions
- Limitations
  - Not all problems can be framed in the mathematical manner that genetic algorithms demand
  - Development of a genetic algorithm is complex
  - In some situations, the "genes" from a few comparatively highly fit (but not optimal) individuals may come to dominate the population, causing it to converge on a local maximum
  - Most genetic algorithms rely on random number generators that produce different results each time the model runs

Genetic Algorithm Applications

- Genetic algorithms provide a set of efficient, domain-independent search heuristics for a broad spectrum of applications including
  - Dynamic process control
  - Complex design of engineering structures
  - Scheduling
  - Transportation and routing
  - Layout and circuit design
  - Telecommunications
  - Discovery of new connectivity typologies

Intelligent Agents

- A computer program that carries out a set of operations on behalf of a user or another program, with some degree of autonomy, and in doing so, employs some knowledge or representation of the user's goals or desires.
- Agents in various forms
  - Software agents, wizards, software daemons, e-mail agents (mailbots), web browsing assisting agents, intelligent search agents (Web robots, spiders), Internet softbots, network management and monitoring agents, e-commerce agents

Features of Intelligent Agents

- Reactivity
  - Agents perceive their environment and respond in a timely fashion to changes that occur in it
- Proactiveness
  - Agents are able to exhibit goal-directed behavior by taking initiative
- Social ability
  - Agents are capable of interacting with other agents in order to satisfy their design objectives
- Autonomy
  - Agents must have control over their own actions and be able to work and launch actions independently of the user or other actors

Why Use Intelligent Agents

- The Gartner Group findings on information overload:
  - The amount of data collected by large enterprises doubles every year.
  - Knowledge workers can analyze only about 5% of this data.
  - Most of the knowledge workers' efforts are spent in trying to discover important patterns in the data (60% or more), a much smaller percentage in determining what these patters mean (20% or less), and very little time (10% or less) is spend actually doing something about the patterns.
  - Information overload reduces our decision-making capabilities by 50 percent.
- A major value of intelligent agents is that they are able to assist in searching through all the data .
- Intelligent agents save time by making decisions about what is relevant to the user as well as by automating routine tasks.

Intelligent Agents: How Smart Are They?

- Intelligence levels
  - Level 0 - Agents retrieve documents for a user under straight orders
  - Level 1 - Agents provide a user-initiated searching facility for finding relevant Web pages
  - Level 2 - Agents maintain users' profiles
  - Level 3 - Agents have a learning and deductive component to help a user who cannot formalize a query or specify a target for a search

Intelligent Agents Vs. Expert Systems

- Agents and expert systems are similar in that they both intend to incorporate domain knowledge to automate decision making.
- They are different in the following aspects:
  - Classic ES are not coupled to any environment in which they act; they act through a user as a middle man. Agents can actively search information from the environment in which they reside.
  - ES are not generally capable of reactive and proactive behavior.
  - ES are not generally equipped with social ability in the sense of cooperation, coordination, and negotiation.

Internet-Based Software Agents

- Nine major application areas:
  - Assisting in workflow and administrative management
  - Collaborating with other agents and people
  - Supporting e-commerce
  - Supporting desktop applications
  - Assisting in information access and management, including searching and FAQs
  - Processing e-mail and messages
  - Controlling and managing network access
  - Managing systems and networks
  - Creating user interfaces, including navigation (browsing)

Issues to Consider for Intelligent Agents

- Learning
- Performance
- Multiagents
- Cost justification
- Security and privacy
- Ethical issues
- Acceptance

KNOWLEDGE ACQUISITION

COURSE REVIEW

Agenda

- Introduction to Knowledge Acquisition
- Knowledge Acquisition Issues and Difficulties
- Knowledge Elicitation Techniques
- Knowledge Modeling

Introduction to Knowledge Acquisition

- Knowledge acquisition is the process of acquiring knowledge from a human expert or a group of experts for the development of knowledge-based systems.
- It comprises a set of techniques and methods that attempt to elicit knowledge of a domain specialist through some form of direct interaction with the expert .

Knowledge Acquisition Issues and Difficulties

- Key issues
  - The end-product must be useful to the end-users
  - To be useful, the end-product must be full of high-quality knowledge that is correct, complete, and relevant, and stored in a structured manner
  - The project must be run in an efficient way making the most use of the available resources
  - The project should not unduly disrupt the normal running of the organization, hence should not involve too much time from experts

Knowledge Acquisition Issues and Difficulties

- Experts can find it difficult to:
  - Express their expertise in a manner that is fully comprehensible to the knowledge engineer
  - Ascertain what the knowledge engineer actually wants
  - Give the right level of detail
  - Present ideas in a clear and logical order
  - Explain all of the jargon and the domain-specific terminology
  - Recall everything that is relevant to the project
  - Avoid drifting off to talk about irrelevant things
- Knowledge engineers can find it difficult to
  - Understand everything the expert says
  - Note down everything the expert says
  - Keep the expert talking about relevant issues
  - Maintain the high level of concentration required to take in a mass of new knowledge

Knowledge Elicitation Techniques

- Interview
- Protocol analysis
- Laddering
- Concept sorting
- Repertory grids
- Structural assessment

Interviewing

- The interview is the most commonly used knowledge-elicitation technique
- Planning the interview
  - Read background material
  - Establish interviewing objectives
  - Decide whom to interview
  - Prepare the interviewee
  - Decide on structure and question types

Interview Structure

- Interview type
  - Unstructured interview
  - Semi-structured interview
  - Structured interview
- Question sequence
  - Pyramid, starting with specific questions and working toward general questions.
  - Funnel, starting with general questions and working toward specific questions.
  - Diamond, starting with specific, moving toward general, and ending with specific questions.

Question Types and Pitfalls

- Question types
  - Open-ended questions
  - Closed questions
  - Probing questions

- Question pitfalls
  - Leading questions
  - Double-barreled questions

Useful Probing Questions

- Why would you do that?
  - Converts an assertion into a rule
- How would you do that?
  - Generates lower-order rules
- When would you do that? IS <the rule> always the case?
  - Reveals the generality of the rule and may generate other rules
- What if it were not the case that <currently true condition>?
  - Generates rules for when current condition does not apply
- Can you tell me more about <any subject already mentioned>?
  - Used to generate further dialogue

Tips for Conducting the Interview

- One day before the interview, confirm times and places.
- Dress appropriately.
- Arrive a little early.
- Remind your interviewee that you will record important points.
- Pick up on vocabulary and jargon.
- Double check to ensure correct understanding.
- Be aware of time limit.
- End with a final checking question.
- Thank the interviewee. Send a thank-you card.
- Write the interview report.

Protocol Analysis

- Analysis of the expert actually solving problems in the domain
  - Online protocol analysis
    - Self-report (also called think-aloud)
    - Shadowing
  - Offline protocol analysis
    - Retrospective verbalization of the problem-solving
- Particularly useful in analyzing dynamic reasoning behavior
- Potential pitfalls
  - Unstructured transcripts
  - Limited scope of the knowledge
  - Inaccurate verbalization

Laddering

- The expert and the knowledge engineer construct a graphical representation of the domain in terms of the relations between domain and problem solving elements.
  - This method results in a qualitative, two-dimensional graph where nodes are connected by labeled arcs.
  - The graph takes the form of a hierarchy of trees.
- Laddering is most useful in the early phases of domain exploration.

Concept Sorting

- In its simplest version an expert is presented with a number of cards on each of which a concept word is printed. The cards are shuffled and the expert is asked to sort the cards into either a fixed number of piles or into any number of piles the expert finds appropriate. This process is repeated many times.
- It can uncover how an expert sees relationships between a fixed set of concepts. It is particularly helpful in constructing a domain schema in unfamiliar domains.
- It requires prestructuring of the data.

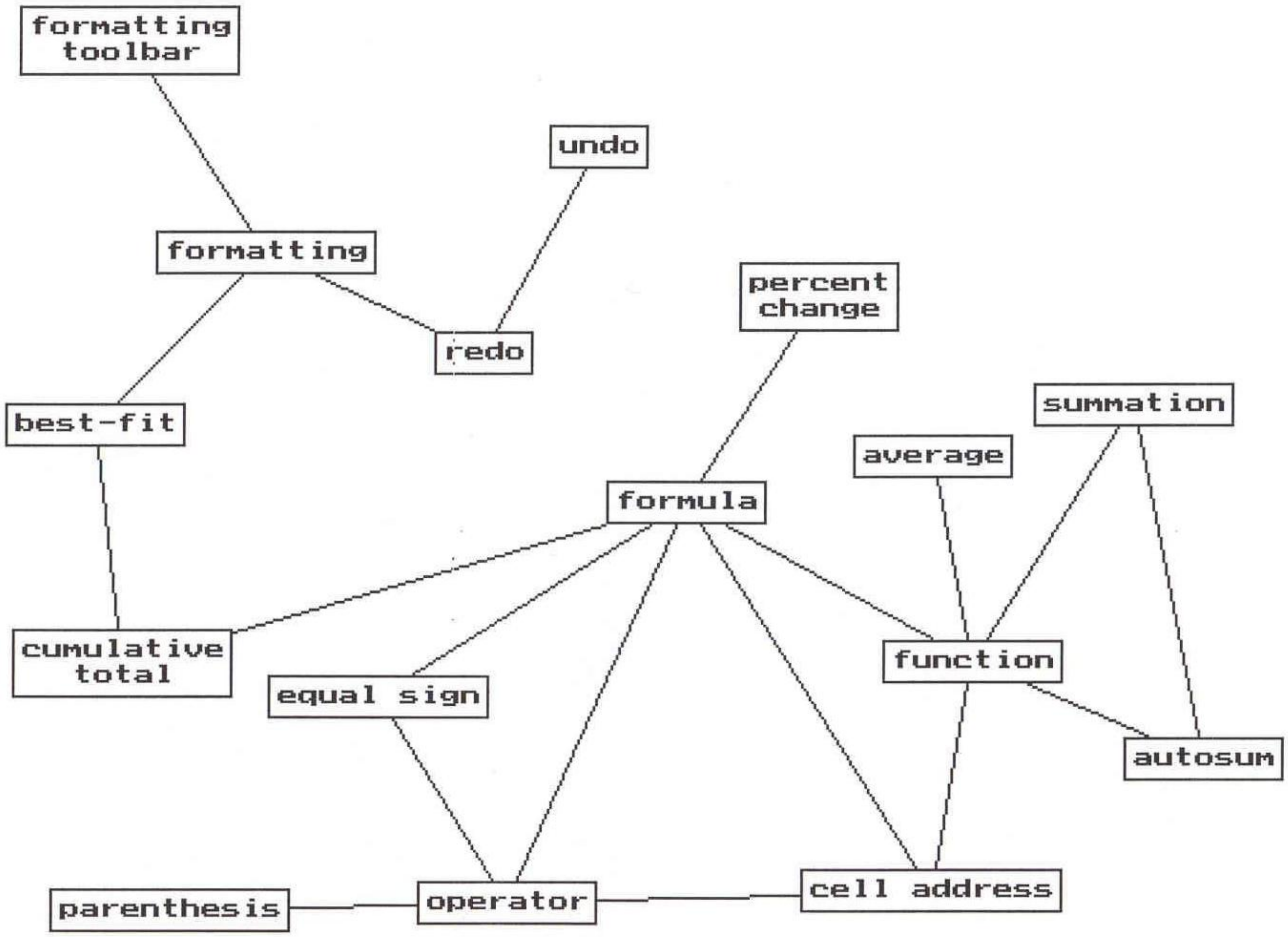
Repertory Grids

- The repertory grid technique has its roots in the psychology of personality and is designed to reveal a conceptual map of a domain.
- Grids are prepared in the following way
  1. Define the domain
  2. State the elements
  3. Select three elements and identify a construct for two similar elements
  4. Repeat Step 3 until no further discriminating constructs
  5. Rank the elements
  6. Analyze the elements
- This technique is useful when trying to uncover the structure of an unfamiliar domain.

Structural Assessment

- Formalized by Goldsmith and Johnson (1990)
- Structural assessment (SA) steps
  - Define a referent structure of knowledge structure
    - Identify a set of central concepts and obtain experts' judgments of relatedness between pairs of these concepts to define the referent structure
  - Elicit judgments of relatedness
    - Elicit an individual's judgments of the relationships among the selected concepts.
  - Derive representations of knowledge
    - Transform the relatedness ratings to a more meaningful, interpretable representation
    - Scaling methods: MDS, cluster analysis, Pathfinder
  - Evaluate the representations
    - Evaluate an individual's cognitive structure
    - Pathfinder's primary index: closeness, coherence

Referent Knowledge Structure Example (Davis & Yi, 2004)



Knowledge Modeling

- Concept tree
- Matrices
  - Attribute matrix
  - Relationship matrix
- Maps
  - Concept map
  - Process map
- Pathfinder network
- Timeline
- Frame

KNOWLEDGE REPRESENTATION AND REASONING

COURSE REVIEW

Agenda

- Introduction to Knowledge Representation and Reasoning
- Procedural vs. Declarative Programming
- Knowledge Representation Methods
- First-Order Logic
- Reasoning

Introduction to Knowledge Representation and reasoning

- Knowledge representation and reasoning is the field of study concerned with how to use a symbol system to represent a domain of knowledge with functions that allow inference (formalized reasoning) about the objects within the domain.
- We defined before knowledge as a justified belief that increases an entity's capacity for effective action.
  - Propositions
  - Formal symbols
  - Reasoning

Procedural vs. Declarative Programming

- Procedural programming
  - A program written in procedural language (e.g., C++ or Java) consists of a set of procedures that must be performed in a strict sequence to accomplish a purpose
    - Implies automatic response to stimuli – little or no thinking about the response involved
- Declarative programming
  - A program consists of a set of rules and facts that can be used by an inference engine to reach other true conclusions

Procedural vs. Declarative Programming

Procedural programming	Declarative programming

Procedural programming

```
If (x.equals("snow"))
    system.out.print("It is white.");
Else if (x.equals("grass"))
    system.out.print("It is green.");
Else if (x.equals("sky"))
    system.out.print("It is yellow.");
Else
    system.out.print("Beats me.");
```

Declarative programming

```
printColor(X) :- color(X,Y), !, write("It
    is "), write(Y), write(".").
printColor(X) :- write("Beats me.").
color(snow, white)
color(sky, yellow)
color(X,Y) :- made of(X,Z), color(Z,Y).
madeof(grass,vegetation).
color(vegetation, green).
```

Why Knowledge Representation and Reasoning

- Why knowledge representation?
  - We can add new tasks and easily make them depend on previous knowledge
  - We can extend the existing behavior by adding new beliefs.
  - We can debug faulty behavior by locating the erroneous beliefs of the system.
  - We can concisely explain and justify the behavior of the system.
- Why reasoning?
  - We would like action to depend on what the system believes about the world, as opposed to just the system has explicitly represented.

Requirements for Knowledge Representation Facility

- It should be able to represent the given knowledge to a sufficient depth.
- It should preserve the fundamental characteristics of knowledge, such as completeness, accessibility, transparency, naturalness, and so on.
- It should be able to infer new knowledge.
- It should be able to provide reasoning and explanation.
- It should be adaptive enough to store updates and support incremental development.

Common Knowledge Representation Methods

- Logic
  - First-order logic
- Rules
  - Production rules
- Frames
- Semantic networks

Factual Knowledge

- Constants
- Variables
- Functions
- Predicates
  - Special functions that return only Boolean values (true or false)
- (Well Formed) Formulas
  - String of symbols that is generated by a formal language

Introduction to First-Order Logic

- A formal logic generated by combining predicate logic and propositional logic.
  - Propositional logic is used to assert propositions, which are statements that are either true or false. It deals only with the truth value of complete statements and does not consider relationships or dependencies between objects.
  - Predicate logic is an extension and generalization of propositional logic. Its formulas contain variables which can be quantified. Two common quantifiers are the existential $\exists$ and universal $\forall$ quantifiers. The variables could be elements in the universe, or perhaps relations or functions over the universe.

First-Order Logic Syntax

- **Symbols**
  - Variable symbols: x, y, z, …
  - Function symbols: f, g, h, bestFriend, …
  - Predicate symbols: P, Q, R, OlderThan, …
  - Logic symbols: "$\neg$", "$\wedge$", "$\vee$", "$\exists$", "$\forall$", "$=$", "$\rightarrow$"
  - Punctuation symbols: "(", ")", and "."

First-Order Logic Syntax

- **Terms**
  - A term is used to refer to something in the world
  - Variables are terms and f(T) is a term, where f is a function and T is a sequence of n terms.
- **Formulas**
  - A formula is used to express a proposition
  - Atomic formula - P(T) is an atomic formula, where P is a predicate and T is a sequence of n terms.
  - Literals - atomic formulas and negated atomic formulas
  - Well-formed formulas (wffs) – literals are wffs and wffs connected or quantified are also wffs.
- **Sentence**
  - A sentence is any formula in which all variables are within the scope of corresponding quantifiers.
- **Clause**
  - A wff consisting of a literal or a disjunction of literals (literals connected by Ors).

Representing Procedural/Relational Knowledge

- Production Rules
  - If <premise>, then <conclusion>
  - If <condition>, then <action>
  - Rules permit the generation of new knowledge in the form of facts that are not initially available but that can be deduced from other knowledge parts. These facts are generated as the conclusions of the rules are applied.
- Semantic Networks
  - Graphical descriptions of knowledge composed of nodes and links that carry semantic information about the relationships between the nodes.
- Frames
  - Organizes knowledge typically according to cause-and-effect relationships. The slots of a frame contains items like rules, facts, references, and so on.

Reasoning: Types of Logic

- **Deduction**
  - The process of reasoning in which a conclusion follows necessarily from the stated premises; reasoning from the general to the specific.
  - If X is true and if X being true implies Y is true, then Y is true.
- **Induction**
  - The process of reasoning in which a conclusion about all members of a class from examination of only a few members of the class; reasoning from the particular to the general.
  - For a set of objects, X={a,b,c…}, if property P is true for a, b, and c, then P is true for all X.
- **Abduction**
  - A form of deductive logic which provides only a "plausible inference." Using statistics and probability theory, abduction may yield the most probable inference among many possible inferences.
  - If Y is true and X implies Y, then X is true.

Reasoning: Forward Chaining

- In order to prove X where X has the form $A \rightarrow C$, find an axiom or theorem of the form $A \rightarrow B$ and transform the problem to the problem of proving $B \rightarrow C$.
- Starts with some facts and applies rules to find all possible conclusions (data-driven)
- steps
  1. Consider the initial facts and store them in working memory of the knowledge base.
  2. Check the antecedent part of the rules.
  3. If all the conditions are matched, fire the rule.
  4. If there is only one rule, do the following:
     a. Perform necessary actions
     b. Modify working memory and update facts.
     c. Check for new conditions
  5. If more than one rule is selected, use the conflict resolution strategy to select the most appropriate rule and go to Step 4
  6. Continue until an appropriate rule is found and executed.

Reasoning: Backward Chaining

- In order to prove X where X has the form A → C, find an axiom or theorem of the form B → C and transform the problem to the problem of proving A → B.
- Starts with the desired conclusion(s) and works backward to find supporting facts (goal-driven)
- steps
  1. Start with a possible hypothesis, H.
  2. Store the hypothesis H in working memory, along with the available facts.
  3. If H is in the initial facts, the hypothesis is proven. Go to Step 7.
  4. If H is not in the initial facts, find a rule R that has a descendent (action) part mentioning the hypothesis.
  5. Store R in the working memory.
  6. Check conditions of R and match with the existing facts.
  7. If matched, then fire the rule R and stop. Otherwise, continue to Step 4.

Reasoning Example

- R1: if (nasal congestion and virosis), then diagnose (influenza) exit
- R2: if (runny nose), then assert (nasal congestion)
- R3: if (body aches), then assert (itchiness)
- R4: if (temp > 100), then assert (fever)
- R5: if (headache), then assert (itchiness)
- R6: if (fever and itchness and cough), then assert (virosis)

UNCERTAIN REASONING

TOPIC 7

Uncertainty in Knowledge Engineering

- Many situations and events cannot be predicted with absolute certainty (or confidence).
  - Exact reasoning vs. inexact reasoning
- One of the main strengths of a knowledge-based system is its ability to handle uncertainty just like a real person.
- In building a knowledge-based system, uncertainty can be handled via
  - Confidence factors
  - Probabilistic reasoning
  - Fuzzy logic

Confidence Factors

- Confidence factors can be used to manage uncertainty by acting as a measure of uncertainty.
  - Uncertainty in antecedents
    - Based on the data supplied by the user
    - Deduced from another rule in the rule base
  - Uncertainty in a rule
    - Based on the expert's confidence in the rule
- Uncertainty in the data and rules must be combined and propagated to the conclusions.

```
      0.8
A     =>    B
0.8         ?
```

Confidence Factors: Strengths and Limitations

- Strengths
  - Confidence factors allow us to express varying degrees of confidence, which in turn allow these values to be manipulated.
  - Confidence factors rank several possible solutions.
- Limitations
  - Confidence factors are generated from the opinions of one or more experts, and thus they can be unreliable.
  - As well as two people finding very different numbers, individuals will also be inconsistent on a day-to-day basis.

Probabilistic Reasoning

- Probability is a quantitative way of dealing with uncertainty.
- The conditional probability, P(A|B), states the probability of event A given that event B occurred. The inverse problem is to find the inverse probability, which states the probability of an earlier event given that a later one occurred. The solution to this problem is Bayes' Theorem.
- Most attempts to use probability theory to handle uncertainty in knowledge-based systems are based on Bayes' Theorem, which can be represented as:
  - $P(A|B) = (P(B|A)*P(A))/P(B)$

  Alternatively, it can be represented as

  - $P(A|B) = (P(B|A)*P(A))/(P(B|A)*P(A) + (P(B|\sim A)*P(\sim A))$

Probabilistic Reasoning (Cont)

- A Bayesian inference system can be established using the following steps:
  1. Define a set of hypotheses, which define the actual results expected.
  2. Assign a probability factor to each hypothesis to give an initial assessment of the likelihood of that outcome occurring.
  3. Check that the evidence produced meets one of these hypotheses.
  4. Amend the probability factors in the light of the evidence received from using the model.
- A Baysian network is a directed graph that represents a set of random variables and their dependence relations with quantitative probability information.

Probabilistic Reasoning: Strengths and Limitations

- Strengths
  - Bayes' Theorem is mathematically sound.
  - The results are based on mathematically proven reasoning and statistical data rather than people's opinions.
- Limitations
  - Needs statistical data to be collected. The data might not be available or accurate.

Fuzzy Logic

- Fuzzy logic is a form of multi-valued logic derived from fuzzy set theory to deal with reasoning that is approximate rather than precise.
- The central notion is that truth values (in fuzzy logic) or membership values (in fuzzy sets) are indicated by a value on the range between 0 and 1, with 0 representing absolute Falseness and 1representing absolute Truth.
- Fuzzy reasoning involves three steps:
  - Fuzzification of the terms in the conditions of the rules
  - Inference from fuzzy rules
  - Defuzzification of the fuzzy terms in the conclusions of the rules

Fuzzy Logic: An Example (Fan Control)

- Conventional model:

  if temperature > X, then run fan

  else, stop fan

- Fuzzy System:

  if temperature = hot, then run fan at full speed

  if temperature = warm, then run fan at moderate speed

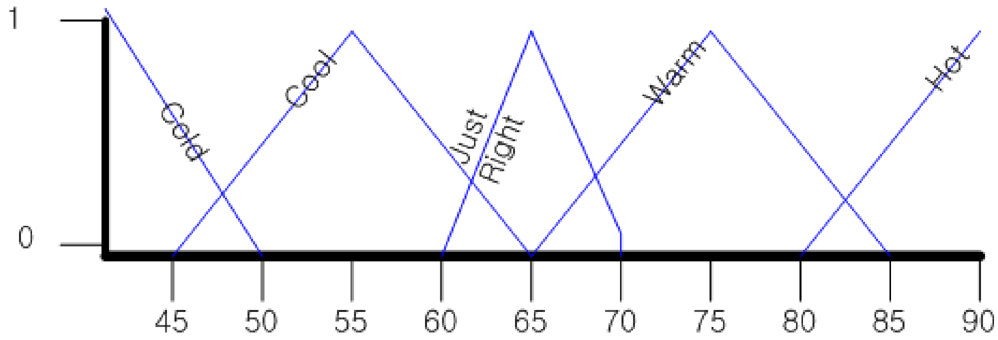
  if temperature = just right, maintain fan speed

  if temperature = cool, then slow fan

  if temperature = cold, then stop fan

Fuzzy Logic: An Example (Fan Control)

- Fuzzification
  - Scales and maps input variables to fuzzy sets



- Inference Mechanism
  - Examining the rules and deducing the control action
- Defuzzification
  - Convert fuzzy output values to control signals

Fuzzy Logic: Strengths and Limitations

- Strengths
  - Fewer rules are required within a knowledge base.
  - Inputs and outputs can be in terms of familiar to humans.
- Limitations
  - Fuzzy logic rules are difficult to write and check due to the imprecise nature of the logic.
  - Systems based on fuzzy logic are difficult to maintain and upgrade.