INTELLIGENT SYSTEMS
SEM-6TH
SECTION-B (NOTES)

Knowledge representation and reasoning:

Knowledge representation and reasoning (**KR**) is the field of artificial intelligence (AI) dedicated to representing information about the world in a form that a computer system can utilize to solve complex tasks such as diagnosing a medical condition or having a dialog in a natural language. Knowledge representation incorporates findings from psychology about how humans solve problems and represent knowledge in order to design formalisms that will make complex systems easier to design and build. Knowledge representation and reasoning also incorporates findings from logic to automate various kinds of *reasoning*, such as the application of rules or the relations of sets and subsets.

Knowledge is the body of facts and principles. Knowledge can be language, concepts, procedures, rules, ideas, abstractions, places, customs, and so on. Study of knowledge is called Epistemology.

Types of knowledge

The types of knowledge include procedural knowledge, declarative knowledge and heuristic knowledge.

1. Procedural knowledge

Procedural knowledge is compiled or processed form of information. Procedural knowledge is related to the performance of some task. For example, sequence of steps to solve a problem is procedural knowledge.

2. Declarative knowledge

Declarative knowledge is passive knowledge in the form of statements of facts about the world. For example, mark statement of a student is declarative knowledge.

3. Heuristic knowledge

Heuristics knowledge are rules of thumb or tricks. Heuristic knowledge is used to make judgments and also to simplify solution of problems. It is acquired through experience. An expert uses his knowledge that he has gathered due to his experience and learning.

Importance of knowledge

Intelligence requires knowledge. That is, to exhibit intelligence, knowledge is required. Knowledge plays a major role in building intelligent systems.

Knowledge representation techniques:

All of these, in different ways, involve hierarchical representation of data.

- Lists - linked lists are used to represent hierarchical knowledge
- Trees - graphs which represent hierarchical knowledge. LISP, the main programming language of AI, was developed to process lists and trees.
- Semantic networks - nodes and links - stored as propositions.
- Schemas - used to represent commonsense or stereotyped knowledge.
- Frames - Describe objects. Consist of a cluster of nodes and links manipulated as a whole. Knowledge is organized in slots. Frames are hierarchically organized.
- Scripts - Describe event rather than objects. Consist of stereotypically ordered causal or temporal chain of events.
- Rule-based representations - used in specific problem-solving contexts. Involve production rules containing *if-then* or *situation-action* pairs. Specific example: problem space representations. Contain:
    - Initial state
    - Goal state
    - Legal operators, i.e. things you are allowed to do
    - Operator restrictions, i.e. factors which constrain the application of operators

Frames:

Natural language understanding requires inference i.e., assumptions about what is typically true of the objects or situations under consideration. Such information can be coded in structures known as frames.

Need of frames

Frame is a type of schema used in many AI applications including vision and natural language processing. Frames provide a convenient structure for representing objects that are typical to a stereotypical situations. The situations to represent may be visual scenes, structure of complex physical objects, etc. Frames are also useful for representing commonsense knowledge. As frames allow nodes to have structures they can be regarded as three-dimensional representations of knowledge.

A frame is similar to a record structure and corresponding to the fields and values are slots and slot fillers. Basically it is a group of slots and fillers that defines a stereotypical object. A single frame is not much useful. Frame systems usually have collection of frames connected to each other. Value of an attribute of one frame may be another frame.

Frames can represent either generic or frame. Following is the example for generic frame.

The fillers may values such as computer in the name slot or a range of values as in types slot. The procedures attached to the slots are called procedural attachments. There are mainly three types of procedural attachments: if-needed, default and if-added. As the name implies if-needed types of procedures will be executed when a filler value is needed. Default value is taken if no other value exists. Defaults are used to represent *commonsense knowledge*. Commonsense is generally used when no more situation specific knowledge is available.

Frames can represent either generic or frame. The fillers may values such as computer in the name slot or a range of values as in types slot. The procedures attached to the slots are called procedural attachments. There are mainly three types of procedural attachments: if-needed, default and if-added. As the name implies if-needed types of procedures will be executed when a filler value is needed. Default value is taken if no other value exists. Defaults are used to represent *commonsense knowledge*. Commonsense is generally used when no more situation specific knowledge is available.

<u>Semantic Nets:</u>

A **semantic net** (or semantic network) is a knowledge representation technique used for propositional information. So it is also called a propositional net. Semantic nets convey meaning. They are two dimensional representations of knowledge. Mathematically a *semantic net* can be defined as a labelled directed graph.

Semantic nets consist of nodes, links (edges) and link labels. In the semantic network diagram, nodes appear as circles or ellipses or rectangles to represent objects such as physical objects, concepts or situations. Links appear as arrows to express the relationships between objects, and link labels specify particular relations. Relationships provide the basic structure for organizing knowledge. The objects and relations involved need not be so concrete. As nodes are associated with other nodes semantic nets are also referred to as associative nets.

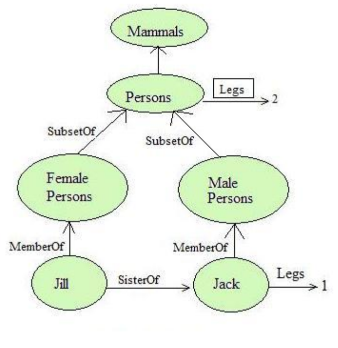

Figure: A Semantic Network

Disadvantage of Semantic Nets:

1. The links between the objects represent only binary relations.
2. There is no standard definition of link names.

Advantages of Semantic Nets

1. They convey some meaning in a transparent manner.

2. They nets are simple and easy to understand.

3. They are easy to translate into PROLOG.

Knowledge Representation Rules.

Some kind of knowledge are hard to represent in predicate logic. For example the degree of hotness in the statement " It is very hot today" can not be represented in predicate logic. A good deal of the reasoning people do involves manipulating beliefs. (eg: "I think it may rain today because it is cloudy "). The "belied system " is mostly incomplete and inconsistent (One may believe in something now and in something else later). consider the following situation.

A, B and C are suspects in a murder case. A has an alibi, in the register of a respectable hotel. B also has an alibi since his friend says that B was with him all the time. C pleads alibi too saying that he was watching a cricket match in the town. These make one believe that

1. A did not commit the crime

2. B did not

3. A or B or C did fortunately for c, he was caught on the TV while watching the match. Now we have a new belief that

4. C did not commit the crime

All the above four beliefs are inconsistent, so we must reject the weaker one or add new beliefs.

The above example illustrates some of the problems posed by uncertain and fuzzy knowledge. A variety of techniques for handling such problems with in computer programs have been proposed.

Non monotonic logic: This allows for addition and deletion of statements in the database. This also allows the belief in one statement to rest on the lack of belief in another.

Probabilistic Reasoning: This makes it possible to represent likely but uncertain inferences.

Fuzzy logic: This provides a way of representing fuzzy or continuous properties of objects.

Algorithm: Resolution In Predicate Logic:

If the choice of clauses to resolve together at each step is made in certain systematic ways, then the resolution procedure will find a contradiction if one exists. However, it may take a very long time. There exist strategies for making the choice that can speed up the process considerably as given below.

1. Resolve only pairs of clauses that contain complementary literals, since only such resolutions produce new clauses that are harder to satisfy than their parents. To facilitate this, index clauses by the predicates they contain, combined with an indication of whether the predicate is negated. Then given a particular clause, possible resolvents that contain a complementary occurrence of one of its predicates can be located directly.
2. Eliminate certain clauses as soon as they are generated so that they cannot participate in later resolutions. Two kinds of clauses should be eliminated: tautologies (which can never be satisfied) and clauses that are subsumed by other clauses (i.e., they are easier to satisfy) and clauses that are subsumed by other clauses (i.e., they are easier to satisfy. For example, P V Q is subsumed by P)

Rule-based system

In computer science, **rule-based systems** are used as a way to store and manipulate knowledge to interpret information in a useful way. They are often used in artificial intelligence applications and research. Rule-based systems can be used to perform lexical analysis to compile or interpret computer programs, or in natural language processing.

Rule-based programming attempts to derive execution instructions from a starting set of data and rules. This is a more indirect method than that employed by an imperative programming language, which lists execution steps sequentially.

A typical rule-based system has four basic components:

- A list of rules or **rule base**, which is a specific type of knowledge base.
- An inference engine or semantic reasoner, which infers information or takes action based on the interaction of input and the rule base. The interpreter executes a production system program by performing the following match-resolve-act cycle:

    - Match: In this first phase, the left-hand sides of all productions are matched against the contents of working memory. As a result a conflict set is obtained, which consists of instantiations of all satisfied productions. An instantiation of a production is an ordered list of working memory elements that satisfies the left-hand side of the production.
    - Conflict-Resolution: In this second phase, one of the production instantiations in the conflict set is chosen for execution. If no productions are satisfied, the interpreter halts.

- Act: In this third phase, the actions of the production selected in the conflict-resolution phase are executed. These actions may change the contents of working memory. At the end of this phase, execution returns to the first phase.

- Temporary working memory.
- A user interface or other connection to the outside world through which input and output signals are received and sent.

Reasoning with Uncertainty

Though there are various types of uncertainty in various aspects of a reasoning system, the "reasoning with uncertainty" (or "reasoning under uncertainty") research in AI has been focused on the uncertainty of *truth value*, that is, to allow and process truth values other than "true" and "false".

Generally speaking, to develop a system that reasons with uncertainty means to provide the following:

- a semantic explanation about the origin and nature of the uncertainty
- a way to represent uncertainty in a formal language
- a set of inference rules that derive uncertain (though well-justified) conclusions
- an efficient memory-control mechanism for uncertainty management

Nonmonotonic logics

A reasoning system is *monotonic* if the truthfulness of a conclusion does not change when new information is added to the system — the set of theorem can only monotonically grows when new axioms are added. In contrast, in a system doing *non-monotonic reasoning* the set of conclusions may either grow or shrink when new information is obtained.

Nonmonotonic logics are used to formalize plausible reasoning, such as the following inference step:

Birds typically fly.
Tweety is a bird.

Tweety (presumably) flies.

Such reasoning is characteristic of commonsense reasoning, where *default rules* are applied when case-specific information is not available.

The conclusion of nonmonotonic argument may turn out to be wrong. For example, if Tweety is a penguin, it is incorrect to conclude that Tweety flies. Nonmonotonic reasoning often requires jumping to a conclusion and subsequently retracting that conclusion as further information becomes available.

All systems of nonmonotonic reasoning are concerned with the issue of consistency. Inconsistency is resolved by removing the relevant conclusion(s) derived previously by default rules. Simply speaking, the truth value of propositions in a nonmonotonic logic can be classified into the following types:

1. *facts* that are definitely true, such as "Tweety is a bird"
2. *default rules* that are normally true, such as "Birds fly"
3. *tentative conclusions* that are presumably true, such as "Tweety flies"

When an inconsistency is recognized, only the truth value of the last type is changed.

A related issue is belief revision. Revising a knowledge base often follows the principle of minimal change: one conserves as much information as possible.

One approach towards this problem is truth maintenance system, in which a "justification" for each proposition is stored, so that when some propositions are rejected, some others may need to be removed, too.

Major problems in these approaches:

- conflicts in defaults, such as in the "Nixon Diamond"
- computational expense: to maintain the consistency in a huge knowledge base is hard, if not impossible

Probabilistic reasoning

Basic idea: to use probability theory to represent and process uncertainty. In probabilistic reasoning, the truth value of a proposition is extended from {0, 1} to [0, 1], with binary logic as its special case.

To extend the basic Boolean connectives to probabilty functions:

- *negation*: $P(\neg A) = 1 - P(A)$
- *conjunction*: $P(A \wedge B) = P(A) * P(B)$ if A and B are independent of each other
- *disjunction*: $P(A \vee B) = P(A) + P(B)$ if A and B never happen at the same time

Furthermore, the conditional probability of B given A is $P(B|A) = P(B \wedge A) / P(A)$, from which Bayes' Theorem is derived, and it is often used to update a system's belief according to new information: $P(H|E) = P(E|H) * P(H) / P(E)$.

Bayesian Networks are directed acyclic graphs in which the nodes represent variables of interest and the links represent informational or causal dependencies among the variables. The strength of dependency is represented by conditional probabilities. Compared to other approaches of probabilistic reasoning, Bayesian network is more efficient, though its actual computational cost is still high for complicated problems.

Challenges to probabilistic approaches:

- unknown probability values
- inconsistent probability assignments
- computational expense

Fuzzy logic

Fuzzy logic is a generalization of classical logic, and reflects the impression of human language and reasoning.

Examples of fuzzy concepts: "young", "furniture", "most", "cloudy", and so on.

According to fuzzy logic, whether an instance belongs to a concept is usually not a matter of "yes/no", but a matter of degree. Fuzzy logic uses a *degree of membership*, which is a real number in [0, 1].

A major difference between this number and probability is: the uncertainty in fuzzy concepts usually does not get reduced with the coming of new information. Compare the following two cases:

- I'm afraid that tomorrow will be cloudy, so let's take the picture today.
- I'm not sure whether the current weather should be classified as "cloudy" or not.

Basic fuzzy operators:

- *negation*: $M(\neg A) = 1 - M(A)$.
- *conjunction*: $M(A \wedge B) = \min\{M(A), M(B)\}$.
- *disjunction*: $M(A \vee B) = \max\{M(A), M(B)\}$.

Why Reason Probabilistically?

- In many problem domains it isn't possible to create complete, consistent models of the world. Therefore agents (and people) must act in uncertain worlds (which the real world is).
- Want an agent to make rational decisions even when there is not enough information to prove that an action will work.
- Some of the reasons for reasoning under uncertainty:
    - **True uncertainty**. E.g., flipping a coin.
    - **Theoretical ignorance**. There is no complete theory which is known about the problem domain. E.g., medical diagnosis.
    - **Laziness**. The space of relevant factors is very large, and would require too much work to list the complete set of antecedents and consequents. Furthermore, it would be too hard to use the enormous rules that resulted.
    - **Practical ignorance**. Uncertain about a particular individual in the domain because all of the information necessary for that individual has not been collected.

- Probability theory will serve as the formal language for representing and reasoning with uncertain knowledge.

Representing Belief about Propositions

- Rather than reasoning about the truth or falsity of a proposition, reason about the belief that a proposition or event is true or false
- For each primitive proposition or event, attach a **degree of belief** to the sentence
- Use **probability theory** as a formal means of manipulating degrees of belief
- Given a proposition, A, assign a probability, P(A), such that 0 <= P(A) <= 1, where if A is true, P(A)=1, and if A is false, P(A)=0. Proposition A must be either true or false, but P(A) summarizes our degree of belief in A being true/false.
- Examples
  - P(Weather=Sunny) = 0.7 means that we believe that the weather will be Sunny with 70% certainty. In this case Weather is a random variable that can take on values in a domain such as {Sunny, Rainy, Snowy, Cloudy}.
  - P(Cavity=True) = 0.05 means that we believe there is a 5% chance that a person has a cavity. Cavity is a Boolean random variable since it can take on possible values *True* and *False*.
  - Example: P(A=a ^ B=b) = P(A=a, B=b) = 0.2, where A=My_Mood, a=happy, B=Weather, and b=rainy, means that there is a 20% chance that when it's raining my mood is happy.

Axioms of Probability Theory

Probability Theory provides us with the formal mechanisms and rules for manipulating propositions represented probabilistically. The following are the three axioms of probability theory:

- 0 <= P(A=a) <= 1 for all *a* in sample space of A
- P(True)=1, P(False)=0
- P(A v B) = P(A) + P(B) - P(A ^ B)

From these axioms we can show the following properties also hold:

- P(~A) = 1 - P(A)
- P(A) = P(A ^ B) + P(A ^ ~B)
- Sum{P(A=a)} = 1, where the sum is over all possible values *a* in the sample space of A

Bayes' theorem:

In probability theory and statistics, **Bayes' theorem** (alternatively **Bayes' law** or Bayes' rule) describes the probability of an event, based on prior knowledge of conditions that might be related to the event. For example, if cancer is related to age, then, using Bayes' theorem, a person's age can be used to *more accurately* assess the probability that they have cancer, compared to the assessment of the probability of cancer made without knowledge of the person's age.

One of the many applications of Bayes' theorem is Bayesian inference, a particular approach to statistical inference. When applied, the probabilities involved in Bayes' theorem may have different probability interpretations. With the Bayesian probability interpretation the theorem expresses how a subjective degree of belief should rationally change to account for availability of prior related evidence. Bayesian inference is fundamental to Bayesian statistics.

- $P(A)$ and $P(B)$ are the probabilities of observing A and B without regard to each other.
- $P(A \mid B)$, a conditional probability, is the probability of observing event A given that B is true.
- $P(B \mid A)$ is the probability of observing event B given that A is true.

The entire output of a factory is produced on three machines. The three machines account for 20%, 30%, and 50% of the output, respectively. The fraction of defective items produced is this: for the first machine, 5%; for the second machine, 3%; for the third machine, 1%. If an item is chosen at random from the total output and is found to be defective, what is the probability that it was produced by the third machine?

A solution is as follows. Let A_i denote the event that a randomly chosen item was made by the ith machine (for $i = 1,2,3$). Let B denote the event that a randomly chosen item is defective. Then, we are given the following information:
$P(A_1) = 0.2, \quad P(A_2) = 0.3, \quad P(A_3) = 0.5.$

If the item was made by the first machine, then the probability that it is defective is 0.05; that is,
$P(B \mid A_1) = 0.05.$
Overall, we have
$P(B \mid A_1) = 0.05, \quad P(B \mid A_2) = 0.03, \quad P(B \mid A_3) = 0.01.$

To answer the original question, we first find $P(B)$. That can be done in the following way:
$P(B) = \Sigma_i P(B \mid A_i) P(A_i) = (0.05)(0.2) + (0.03)(0.3) + (0.01)(0.5) = 0.024.$

Hence 2.4% of the total output of the factory is defective.
We are given that B has occurred, and we want to calculate the conditional probability of A_3. By Bayes' theorem,
$P(A_3 \mid B) = P(B \mid A_3) P(A_3)/P(B) = (0.01)(0.50)/(0.024) = 5/24.$

Given that the item is defective, the probability that it was made by the third machine is only 5/24. Although machine 3 produces half of the total output, it produces a much smaller fraction of the defective items. Hence the knowledge that the item selected was defective enables us to replace the prior probability $P(A_3) = 1/2$ by the smaller posterior probability $P(A_3 \mid B) = 5/24$.

Once again, the answer can be reached without recourse to the formula by applying the conditions to any hypothetical number of cases. For example, in 100,000 items produced by the factory, 20,000 will be produced by Machine A, 30,000 by Machine B, and 50,000 by Machine C. Machine A will produce 1000 defective items, Machine B 900, and Machine C 500. Of the total 2400 defective items, only 500, or 5/24 were produced by Machine C.
The interpretation of Bayes' theorem depends on the interpretation of probability ascribed to the terms. The two main interpretations are described below.

Bayesian interpretation

In the Bayesian (or epistemological) interpretation, probability measures a "degree of belief." Bayes' theorem then links the degree of belief in a proposition before and after accounting for evidence. For example, suppose it is believed with 50% certainty that a coin is twice as likely to land heads than tails. If the coin is flipped a number of times and the outcomes observed, that degree of belief may rise, fall or remain the same depending on the results.

For proposition *A* and evidence *B*,

- $P(A)$, the *prior*, is the initial degree of belief in *A*.
- $P(A \mid B)$, the "posterior," is the degree of belief having accounted for *B*.
- the quotient $P(B \mid A)/\boxed{P(B)}$ represents the support *B* provides for *A*.

Frequentist interpretation

In the frequentist interpretation, probability measures a "proportion of outcomes." For example, suppose an experiment is performed many times. $P(A)$ is the proportion of outcomes with property *A*, and $P(B)$ that with property *B*. $P(B \mid A)$ is the proportion of outcomes with property *B out of* outcomes with property *A*, and $P(A \mid B)$ the proportion of those with *A out of* those with *B*.

The role of Bayes' theorem is best visualized with tree diagrams, as shown to the right. The two diagrams partition the same outcomes by *A* and *B* in opposite orders, to obtain the inverse probabilities. Bayes' theorem serves as the link between these different partitionings.

An entomologist spots what might be a rare subspecies of beetle, due to the pattern on its back. In the rare subspecies, 98% have the pattern, or $P(\text{Pattern} \mid \text{Rare}) = 98\%$. In the common subspecies, 5% have the pattern. The rare subspecies accounts for only 0.1% of the population. How likely is the beetle having the pattern to be rare, or what is $P(\text{Rare} \mid \text{Pattern})$?

Dempster–Shafer theory:

The theory of belief functions, also referred to as **evidence theory** or **Dempster–Shafer theory** (**DST**), is a general framework for reasoning with uncertainty, with understood connections to other frameworks such as probability, possibility and imprecise probability theories. First introduced by Arthur P. Dempster[1] in the context of statistical inference, the theory was later developed by Glenn Shafer into a general framework for modeling epistemic uncertainty—a mathematical theory of evidence.[2][3] The theory allows one to combine evidence from different sources and arrive at a degree of belief (represented by a mathematical object called *belief function*) that takes into account all the available evidence.

Dempster–Shafer theory is a generalization of the Bayesian theory of subjective probability. Belief functions base degrees of belief (or confidence, or trust) for one question on the probabilities for a related question. The degrees of belief itself may or may not have the mathematical properties of probabilities; how much they differ depends on how closely the two questions are related.[6] Put another way, it is a way of representing epistemic plausibilities but it can yield answers that contradict those arrived at using probability theory.

In this formalism a **degree of belief** (also referred to as a **mass**) is represented as a **belief function** rather than a Bayesian probability distribution. Probability values are assigned to *sets* of possibilities rather than single events: their appeal rests on the fact they naturally encode evidence in favor of propositions.

Dempster–Shafer theory assigns its masses to all of the non-empty subsets of the propositions that compose a system—in set-theoretic terms, the power set of the propositions. For instance, assume a situation where there are two related questions, or propositions, in a system. In this system, any belief function assigns mass to the first proposition, the second, both or neither.

For example, suppose we have a belief of 0.5 and a plausibility of 0.8 for a proposition, say “the cat in the box is dead.” This means that we have evidence that allows us to state strongly that the proposition is true with a confidence of 0.5. However, the evidence contrary to that hypothesis (i.e. “the cat is alive”) only has a confidence of 0.2. The remaining mass of 0.3 (the gap between the 0.5 supporting evidence on the one hand, and the 0.2 contrary evidence on the other) is “indeterminate,” meaning that the cat could either be dead or alive. This interval represents the level of uncertainty based on the evidence in your system.

Hypothesis	**Mass**	**Belief**	**Plausibility**
Null (neither alive nor dead)	0	0	0
Alive	0.2	0.2	0.5
Dead	0.5	0.5	0.8
Either (alive or dead)	0.3	1.0	1.0

The null hypothesis is set to zero by definition (it corresponds to “no solution”). The orthogonal hypotheses “Alive” and “Dead” have probabilities of 0.2 and 0.5, respectively. This could correspond to “Live/Dead Cat Detector” signals, which have respective reliabilities of 0.2 and 0.5. Finally, the all-encompassing “Either” hypothesis (which simply acknowledges there is a cat in the box) picks up the slack so that the sum of the masses is 1. The belief for the “Alive” and “Dead” hypotheses matches their corresponding masses because they have no subsets; belief for “Either” consists of the sum of all three masses (Either, Alive, and Dead) because “Alive” and “Dead” are each subsets of “Either”. The “Alive” plausibility is $1 - m$ (Dead) and the “Dead” plausibility is $1 - m$ (Alive). In other way, the “Alive” plausibility is m(Alive) + m (Either) and the “Dead” plausibility is m(Dead) + m(Either). Finally, the “Either” plausibility sums m(Alive) + m(Dead) + m(Either). The universal hypothesis (“Either”) will always have 100% belief and plausibility—it acts as a checksum of sorts.