Parul® University

Unit -5

Parul® University

NAAC A++ ACCREDITED UNIVERSITY

Uncertainty

Syllabus

- Non-Monotonic Reasoning
- Logics for Non-Monotonic Reasoning
- Forward rules and Backward rules
- Justification based Truth Maintenance Systems
- Semantic Nets Statistical Reasoning
- Probability and Bayes' theorem
- Bayesian Network
- Markov Networks
- Hidden Markov Model
- Basis of Utility Theory
- Utility Functions.

Non-Monotonic Reasoning

Type of Reasoning:

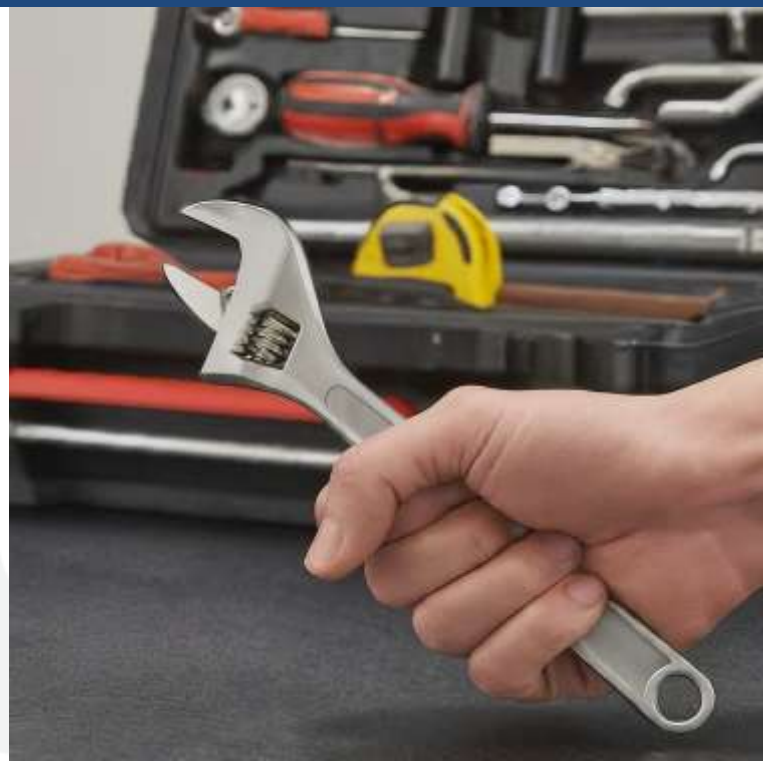
1. Monotonic
2. Non-Monotonics

Definition:

- Monotonic means something that does not vary or change
- Non-Monotonic means something which can vary according to the situation or condition.

Non-Monotonic Reasoning

- Non-monotonic reasoning allows revising conclusions with new information.
- Think of it as a toolbox where you might remove a tool if it becomes irrelevant.
- This makes reasoning more flexible and adaptable to changing situations.



Non-Monotonic Reasoning

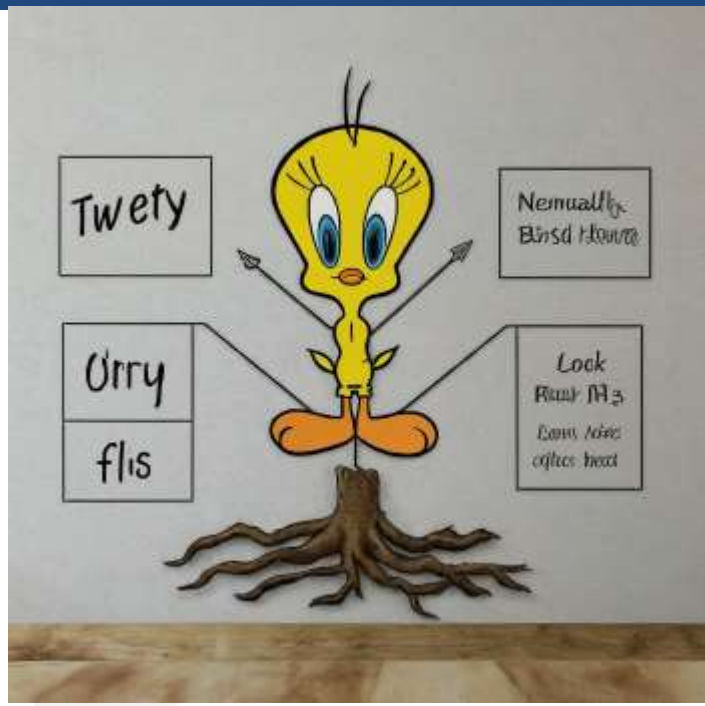
- Non-monotonic Reasoning is the process that changes its direction or values as the knowledge base increases.
- It is also known as NMR in Artificial Intelligence.
- Non-monotonic Reasoning will increase or decrease based on the condition.
- Since that Non-monotonic Reasoning depends on assumptions, It will change itself with improving knowledge or facts.
- Example:

Consider a bowl of water, If we put it on the stove and turn the flame on it will obviously boil hot and as we will turn off the flame it will cool down gradually

Logics for Non-Monotonic Reasoning

Default Logic:

- Introduced by Raymond Reiter in 1980.
- Handles default assumptions.
- Makes assumptions (defaults) in the absence of specific information.
- Example: "Normally, birds fly" (default rule).



Mechanism:

- Defaults: $\frac{\text{Prerequisite:Justification}}{\text{Conclusion}}$.
- Example: $\frac{\text{bird(X):flies(X)}}{\text{flies(X)}}$.

Use Cases:

- Common-sense reasoning.
- Inferences with typical properties.

Autoepistemic Logic

- Developed by Robert C. Moore in the 1980s.
- Reasoning about an agent's own beliefs.
- Reasons about its own knowledge.
- Example: "I don't know it's raining, therefore it probably isn't raining.“
- Revise the conclusion upon seeing new information : "It's cloudy outside."

Use Cases:

- Self-reflective reasoning.
- Modeling agent knowledge and beliefs.

Circumscription:

- Circumscription, introduced by John McCarthy, is a technique used in non-monotonic reasoning to capture the idea that things are typically as simple as possible unless there's evidence to the contrary. It essentially allows us to make inferences by minimizing the number of "special cases" that need to be considered.

- There are two main types of circumscription:

- **Predicate Circumscription:** This focuses on a specific predicate (a property or relation) in your knowledge base. It assumes that an entity satisfies that predicate only if there's a reason for it to do so based on the existing knowledge. In simpler terms, it avoids making assumptions about an entity having a property unless there's explicit evidence for it.
- **Domain Circumscription:** This type deals with the entire domain of discourse. It presumes that all the objects we're reasoning about are already accounted for in our knowledge base. There are no hidden entities waiting to be discovered.

Logic Programming with Negation as Failure

- Used in Prolog.
- Assumes unprovable statements are false.

Mechanism:

Example: **flies(X) :- bird(X), not penguin(X)**

Negation as Failure: $\neg p \neg p$ means "not provable pp".

Use Cases:

- Rule-based systems.
- Expert systems with default assumptions.

Comparison of Logics

- **Key Differences:**
  - **Default Logic:** Assumes typical properties.
  - **Circumscription:** Minimizes assumptions.
  - **Autoepistemic Logic:** Reflects on agent's beliefs.
  - **Negation as Failure:** Defaults to false if unprovable.
  - **ASP(Answer set Programming):** Uses stable models for problem-solving.
  - **Abductive Logic:** Generates best explanations.
- **Use Case Suitability:**
  - Varies based on problem domain and nature of information.

Traditional Reasoning: Forward and Backward rules

- Non-monotonic reasoning can be integrated with both forward and backward chaining.

- **Forward chaining:** Needs to revisit conclusions when encountering new information.
- Starts with known facts, applies rules to reach conclusions (like dominoes falling).

- **Backward chaining:** Might involve making assumptions (defaults) to reach the goal.
- Works backward from a desired goal, searching for rules to justify it.

Forward Chaining with Non-Monotonic Reasoning

- Starts with known facts and applies rules to reach conclusions (like dominoes falling).
- Needs to revisit conclusions when encountering new information that contradicts them.
- This is because new information might invalidate previously drawn conclusions.

Backward Chaining with Non-Monotonic Reasoning

- Works backward from a desired goal, searching for rules to justify it.
- Might involve making assumptions (defaults) to reach the goal.
- These assumptions can be revised if new information contradicts them.

Conclusion

- Forward chaining with revision: Incorporating non-monotonic reasoning, the system might need to revisit conclusions after applying a new rule. For instance, a forward-chaining system might conclude "Tweety flies" based on "Birds fly" and "Tweety is a bird." But upon encountering a rule "Penguins don't fly" and new information "Tweety is a penguin," it would need to revise its conclusion.

- Backward chaining with assumptions: Backward chaining with non-monotonic reasoning might involve making assumptions (defaults) to reach the goal. These assumptions can be revised if new information contradicts them. Imagine the system has a goal "Is the library open?" It might assume "Libraries are usually open on weekdays" unless it finds information about a specific closure.

Justification-Based Truth Maintenance Systems

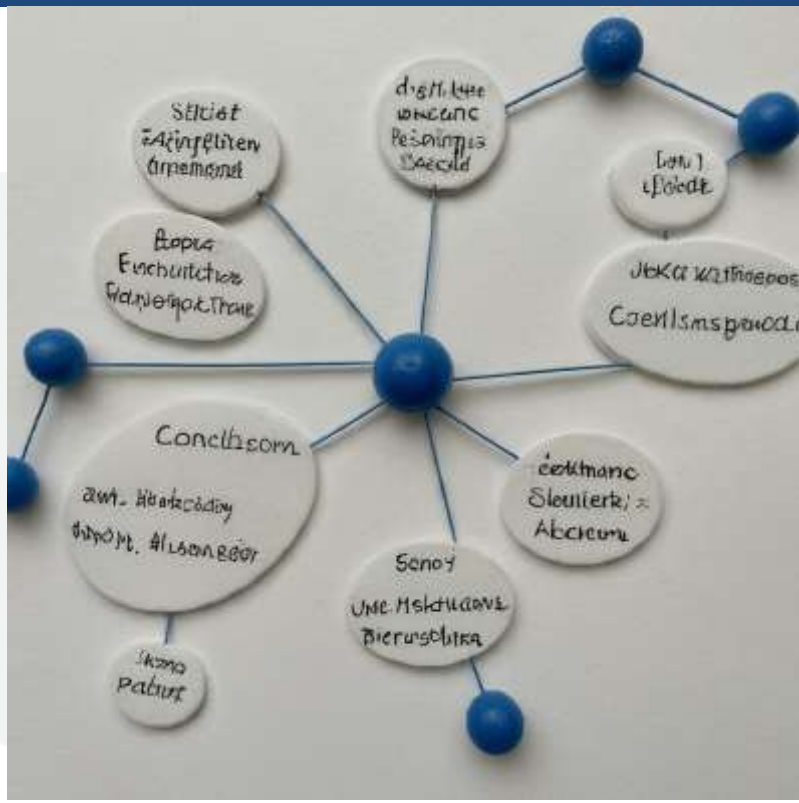
Justification-Based Truth Maintenance Systems (JTMS) are a type of system used in Artificial Intelligence (AI) to manage the consistency of beliefs within a knowledge base. Unlike a traditional database that simply stores true or false statements, a JTMS focuses on the reasons why a belief is held. This allows the system to handle situations where new information might contradict existing beliefs.

The Challenge of Belief Management in AI :

- Traditional databases store information as true or false statements.
- AI systems need to deal with uncertainty and potentially inconsistent information.
- New information might contradict existing beliefs.

How JTMS works

- Belief Representation: Each piece of information is a node (circle).
- Labels: Nodes are labelled "IN" (believed) or "OUT" (not believed).
- Justifications: Rules connect beliefs (lines with arrows). If conditions are met (beliefs "IN"), the conclusion is "IN".
- Maintaining Consistency: The system revises labels when new information or retractions arise

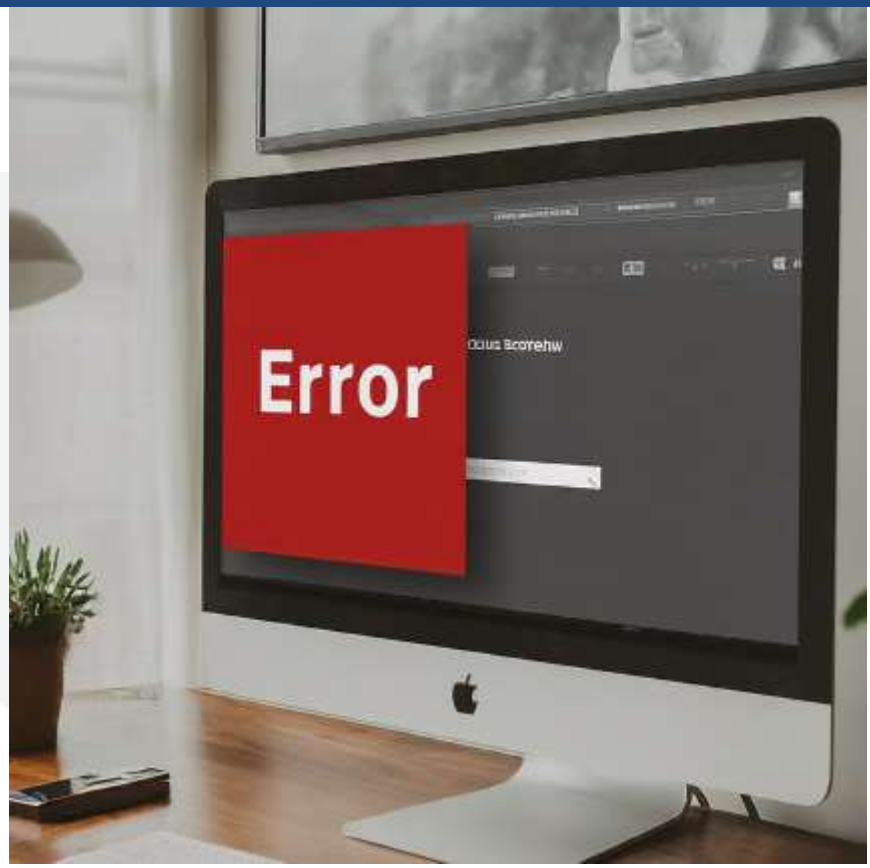


Advantages of JTMS

- Non-Monotonic Reasoning: Reasoning adapts based on new information. Conclusions aren't absolute and can be revised.
- Efficient Belief Revision: The system focuses on revising relevant beliefs based on justification dependencies.
- Reasoning about Assumptions: Analyze hypothetical scenarios by making assumptions and tracking consequences within the JTMS framework.

Limitations of JTMS

- Complexity: Managing consistency can become computationally expensive with a large number of beliefs and justifications.
- Limited Explanations: The primary focus is on maintaining consistency, explanations for specific beliefs might not be readily available.



Semantic Nets

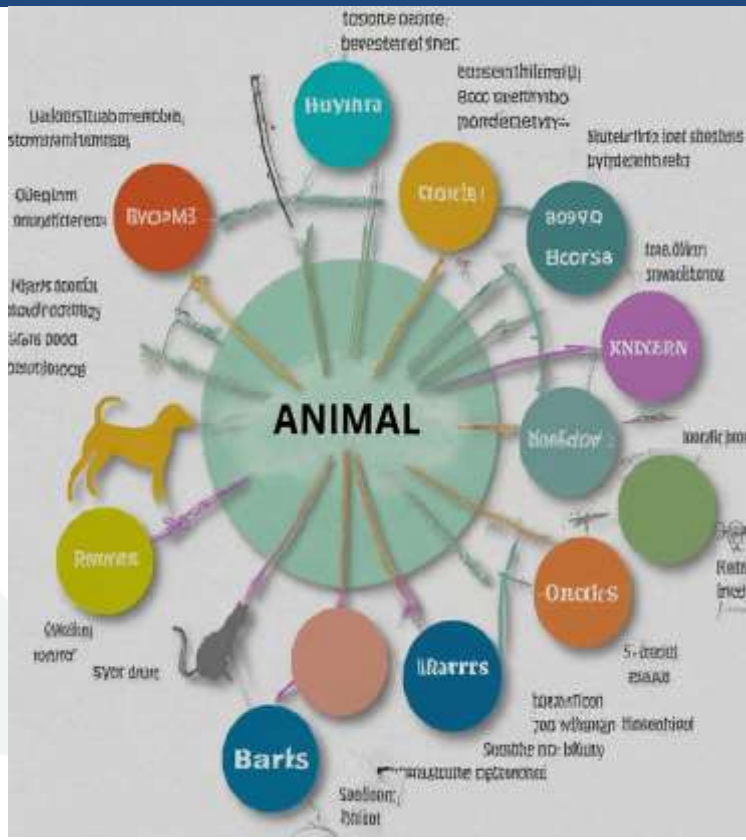
- Function: Capture and structure knowledge by portraying it as a network of interconnected nodes and edges.

Components:

- Nodes: Represent entities or concepts (like dog, cat, animal).
- Edges: Describe the relationships between those entities (like "is-a" - dog is-a animal).
- Benefits:

Intuitive representation: Easy to visualize knowledge structures.

Efficient reasoning: Enables reasoning tasks like inheritance (dogs inherit traits from animals)



Statistical Reasoning in AI

- Function: Reason about and draw conclusions from data by leveraging statistical methods.
- Methods:
- Probability Theory: Calculates the likelihood of events.
- Bayesian Networks: Represent relationships between variables and their probabilities.
- Benefits:

Handles uncertainty: Deals with situations where information is incomplete or imprecise.
Learning from data: Can learn and improve reasoning based on new data.

Combining Semantic Nets and Statistical Reasoning

- Merging the strengths of both approaches.
- Adding weights to edges in semantic nets: Represent the strength or likelihood of a relationship.
- Statistical reasoning to analyze data associated with concepts in a semantic net

Probability and Bayes' theorem

- Bayes' theorem is also known as **Bayes' rule, Bayes' law**, or **Bayesian reasoning**, which determines the probability of an event with uncertain knowledge.
- In probability theory, it relates the conditional probability and marginal probabilities of two random events.
- Bayes' theorem was named after the British mathematician **Thomas Bayes**. The **Bayesian inference** is an application of Bayes' theorem, which is fundamental to Bayesian statistics.

Understanding Probability:

- Probability is the likelihood of an event occurring.
- It's expressed as a value between 0 (impossible) and 1 (certain).
- In AI, probability helps quantify the belief an AI system has about the truth of a statement or the likelihood of a particular outcome.

Key Terms in Probability

- Random Variable: A variable whose value depends on chance (e.g., weather forecast: sunny, rainy)
- Event: A specific outcome (e.g., it rains today)
- Probability Distribution: Describes the probabilities of different events for a random variable

Bayes' Theorem: Updating Beliefs with New Evidence

- Bayes' theorem is a powerful tool for revising beliefs based on new evidence.
- It allows AI systems to continuously learn and improve their decision-making.

Applying Bayes' Theorem: A Medical Diagnosis

- Prior Probability: Doctor's initial belief about a patient's illness based on symptoms.
- Evidence: Test results (X-ray, blood work)
- Posterior Probability: Updated belief about the illness after considering test results.

Applications of Probability and Bayes' Theorem in AI

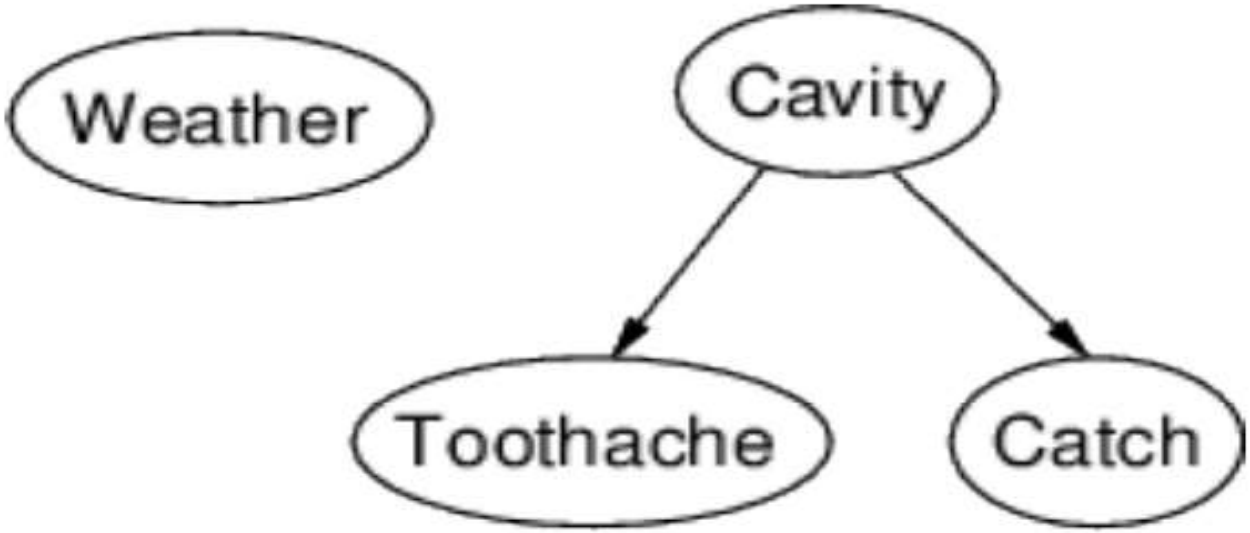
- Machine Learning- Machine learning algorithms learn from data by identifying patterns and relationships.
- Spam Filtering - Spam filters use Bayes' theorem to categorize emails.

They consider the presence of keywords and other indicators (evidence) associated with spam.

- Medical Diagnosis - AI systems can analyze patient data (symptoms, medical history) to estimate disease probabilities.
- Natural Language Processing (NLP) - NLP applications like chatbots or machine translation rely on probability.
- Robotics - Robots use probability to navigate their environment, avoid obstacles, and interact with objects.

Bayesian Network Example

- Topology of network encodes conditional independence assertions:



- Weather is independent of the other variables
- Toothache and Catch are conditionally independent given Cavity

Bayesian Network

- A simple, graphical notation for conditional independence assertions and hence for compact specification of full joint distributions
- Syntax – a set of nodes, one per variable – a directed, acyclic graph (link ≈ “directly influences”) – a conditional distribution for each node given its parents:

$$P(Xi \mid P\,arents(Xi))$$

- In the simplest case, conditional distribution represented as a conditional probability table (CPT) giving the distribution over Xi for each combination of parent values

Example

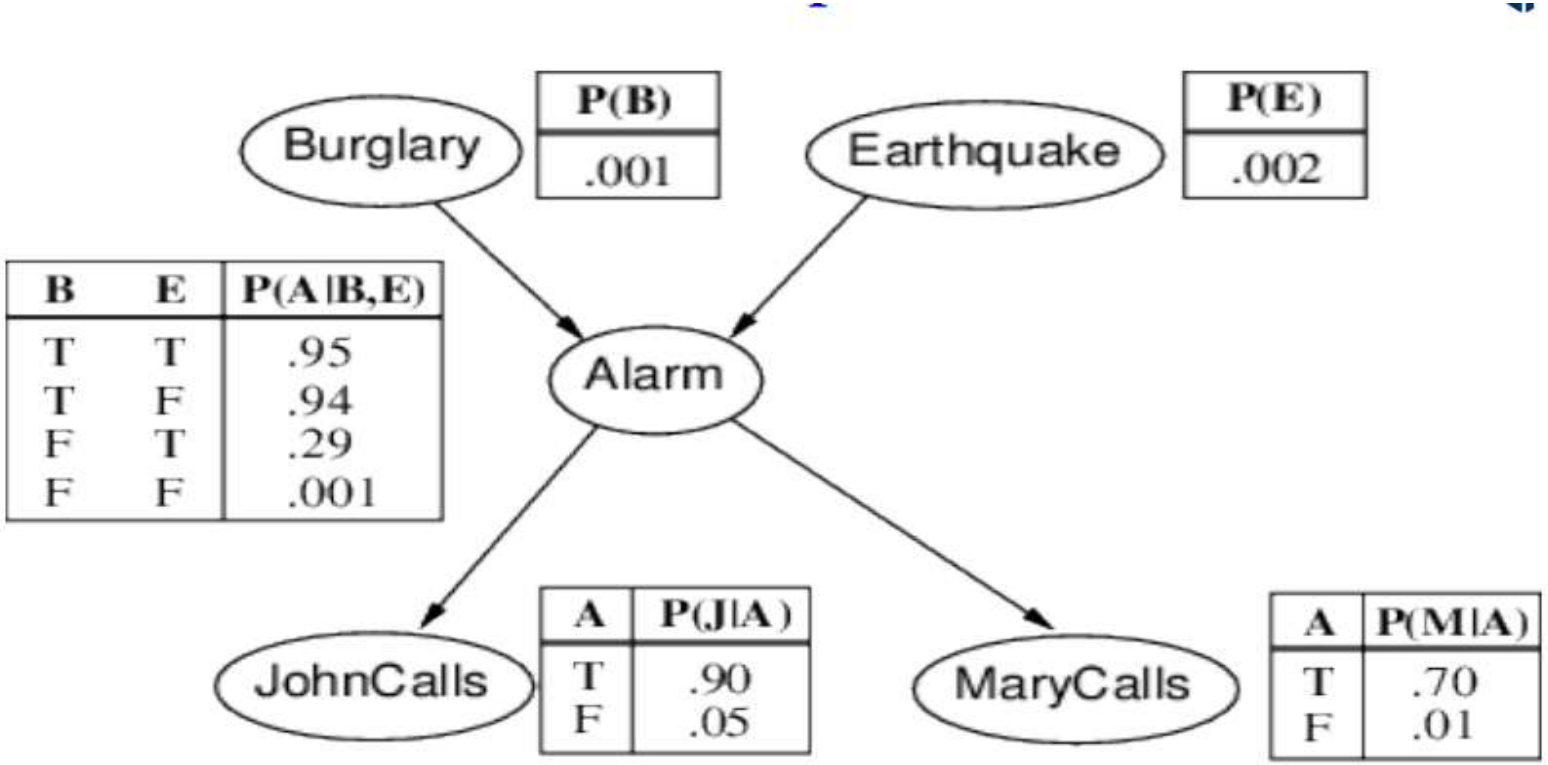
- I'm at work, neighbor John calls to say my alarm is ringing, but neighbor Mary doesn't call. Sometimes it's set off by minor earthquakes. Is there a burglar?

- **Variables:** Burglar, Earthquake, Alarm, John Calls, Mary Calls

- Network topology reflects "causal" knowledge
- A burglar can set the alarm off

- An earthquake can set the alarm off
- The alarm can cause Mary to call
- The alarm can cause John to call

Example


Burglary
P(B)
.001
Earthquake
P(E)
.002
Alarm
B E P(A|B,E)
T T .95
T F .94
F T .29
F F .001
JohnCalls
A P(J|A)
T .90
F .05
MaryCalls
A P(M|A)
T .70
F .01

Constructing Bayesian Networks

- Need a method such that a series of locally testable assertions of conditional independence guarantees the required global semantics

1. Choose an ordering of variables X1, . . . , Xn
2. For i = 1 to n

   add Xi to the network

   select parents from $X_1, \ldots, X_{i-1}$ such that

   $P(X_i \mid \text{Parents}(Xi)) = P(X_i \mid X_1, \ldots, X_{i-1})$

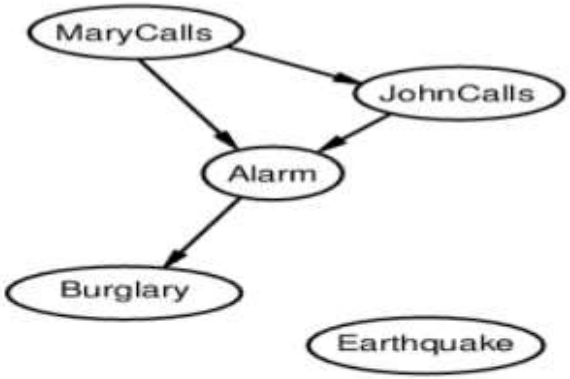
- This choice of parents guarantees the global semantics:

$$\mathbf{P}(X_1, \ldots, X_n) = \prod_{i=1}^{n} \mathbf{P}(X_i | X_1, \ldots, X_{i-1}) \quad \text{(chain rule)}$$

$$= \prod_{i=1}^{n} \mathbf{P}(X_i | Parents(X_i)) \quad \text{(by construction)}$$

Example

- Suppose we choose the ordering M, J, A, B, E



- $P(J|M) = P(J)$? No
- $P(A|J, M) = P(A|J)$? $P(A|J, M) = P(A)$? No
- $P(B|A, J, M) = P(B|A)$? Yes
- $P(B|A, J, M) = P(B)$? No
- $P(E|B, A, J, M) = P(E|A)$?
- $P(E|B, A, J, M) = P(E|A, B)$?

Parul® University NAAC A++ ACCREDITED UNIVERSITY

Markov Networks

- Markov networks are extensively used to model complex sequential, spatial, and relational interactions in fields as diverse as image processing, natural language analysis, and bioinformatics.

- The concept originates from the Sherrington–Kirkpatrick model. A Markov network or MRF is similar to a Bayesian network in its representation of dependencies; the differences being that Bayesian networks are directed and acyclic, whereas Markov networks are undirected and may be cyclic.

Definition

- ***A Markov network is a pair (G,F), where G is an undirected graph over variables V and $F(V)$ is a joint distribution for V such that***

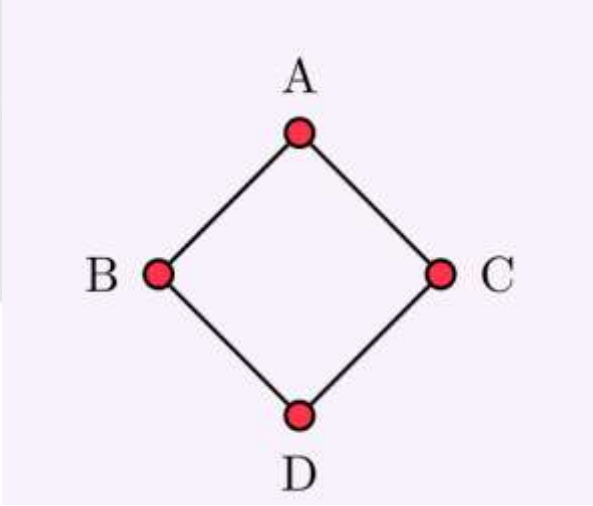
$$\mathcal{X} \perp_u \mathcal{Y} \mid \mathcal{Z} \text{ only if } \mathcal{X} \perp\!\!\!\perp \mathcal{Y} \mid \mathcal{Z}, \qquad \text{[global Markov property]}$$

  for any subsets $X,Y,Z \subseteq V$.

- Markov networks are also called **Markov Random Fields**, following their usage in Physics.

- The maximal cliques of the graph in Figure are formed by the edges. Assuming the distribution is positive, the Markov network in the previous example factorizes as

$$P(A, B, C, D) \propto \phi(A, B)\phi(A, C)\phi(B, D)\phi(C, D) .$$



- The clique potentials model the higher probability attributed to neighboring individuals having the same status (infected or not infected), plus any additional information about the status of an individual (e.g., the result of an imperfect test or a random chance of being infected by an exogenous phenomenon).

Parul® University NAAC A++ ACCREDITED UNIVERSITY

Hidden Markov Model

- A statistical model called a Hidden Markov Model (HMM) is used to describe systems with changing unobservable states over time.
- It is predicated on the idea that there is an underlying process with concealed states, each of which has a known result.

- Probabilities for switching between concealed states and emitting observable symbols are defined by the model.
- Because of their superior ability to capture uncertainty and temporal dependencies, HMMs are used in a wide range of industries, including finance, bioinformatics, and speech recognition.
- HMMs are useful for modelling dynamic systems and forecasting future states based on sequences that have been seen because of their flexibility.

- ***The hidden Markov Model (HMM) is a statistical model that is used to describe the probabilistic relationship between a sequence of observations and a sequence of hidden states.***
- It is often used in situations where the underlying system or process that generates the observations is unknown or hidden, hence it has the name “Hidden Markov Model.”
- It is used to predict future observations or classify sequences, based on the underlying hidden process that generates the data.

An HMM consists of two types of variables: hidden states and observations.

- The **hidden states** are the underlying variables that generate the observed data, but they are not directly observable.
- The **observations** are the variables that are measured and observed.

The relationship between the hidden states and the observations is modeled using a probability distribution. The Hidden Markov Model (HMM) is the relationship between the hidden states and the observations using two sets of probabilities:

the transition probabilities and
the emission probabilities.

- The **transition probabilities** describe the probability of transitioning from one hidden state to another.

- The **emission probabilities** describe the probability of observing an output given a hidden state.

The Hidden Markov Model (HMM) algorithm can be implemented using the following steps:

Step 1: Define the state space and observation space

- The state space is the set of all possible hidden states, and the observation space is the set of all possible observations.

Step 2: Define the initial state distribution

- This is the probability distribution over the initial state.

<u>Step 3:</u> Define the state transition probabilities

- These are the probabilities of transitioning from one state to another. This forms the transition matrix, which describes the probability of moving from one state to another.

<u>Step 4:</u> Define the observation likelihoods:

- These are the probabilities of generating each observation from each state. This forms the emission matrix, which describes the probability of generating each observation from each state.

Step 5: Train the model

- The parameters of the state transition probabilities and the observation likelihoods are estimated using the Baum-Welch algorithm, or the forward-backward algorithm. This is done by iteratively updating the parameters until convergence.

Step 6: Decode the most likely sequence of hidden states

- Given the observed data, the Viterbi algorithm is used to compute the most likely sequence of hidden states. This can be used to predict future observations, classify sequences, or detect patterns in sequential data.

Step 7: Evaluate the model

- The performance of the HMM can be evaluated using various metrics, such as accuracy, precision, recall, or F1 score.

Parul® University

NAAC A++ ACCREDITED UNIVERSITY

Question

What are the applications of Hidden Markov Models?

Parul® University

NAAC A++ ACCREDITED UNIVERSITY

Answer

Speech recognition, natural language processing, bioinformatics (gene prediction, for example), and many other fields where systems can be modeled as sequences of observable events with underlying hidden states are applications that heavily rely on HMMs.

Parul® University NAAC A++ ACCREDITED UNIVERSITY

Basis of Utility Theory

- Artificial intelligence (AI) is now widely used in our daily lives, from smartphone voice assistants to complex decision-making systems in industries like finance and healthcare. ***One fundamental concept that plays a crucial role in decision-making is the utility theory in artificial intelligence.***
- Utility theory in artificial intelligence provides a mathematical framework for understanding how AI systems make choices among different options based on their perceived value or utility.

- Decision-making is a critical aspect of human intelligence and a key component of AI systems. However, AI decision-making is often based on mathematical algorithms and models that are designed to optimize outcomes based on specific objectives.

- Utility theory in artificial intelligence provides a formal way of incorporating preferences and subjective values into the decision-making process of AI systems.
-
- Utility theory in artificial intelligence allows AI systems to choose different options based on their utility or perceived value, considering factors such as risk, uncertainty, and subjective preferences.

Parul® University NAAC A++ ACCREDITED UNIVERSITY

Utility Functions

- A utility function in AI is a mathematical framework that assigns numerical values to outcomes, representing the degree of satisfaction associated with each outcome. It is used to model decision-making based on perceived value or utility.

- Utility theory provides a mathematical framework for decision-making in AI systems. It allows AI systems to assign subjective values or preferences to different outcomes and make optimal choices based on these values.

- Utility functions play a crucial role in AI decision-making by allowing the system to compare and rank different options based on their utility values. AI systems aim to maximize the expected utility, considering the probability of different outcomes occurring.

- Designing an AI utility function is crucial for aligning the AI system's actions with the economic value of the stakeholders. It involves understanding the goals, preferences, and desired outcomes of the stakeholders and incorporating them into the utility function.

- Balancing conflicting objectives in AI utility functions can be achieved by considering the economic values of all stakeholders, understanding their diverse interests, and constantly seeking to balance the trade-offs between conflicting economic values.

- Pareto-optimal solutions represent the best trade-off between conflicting objectives. They cannot be improved in one objective without worsening another. Incorporating Pareto-optimal solutions into the AI utility function allows for finding the best compromise between conflicting economic values.

References

- **Artificial Intelligence (TextBook)**
  By Elaine Rich and Kevin Knight | TMH
- **Artificial Intelligence-Structures and Strategies For Complex Problem Solving**
  By George F. Luger | Pearson Education / PHI
- **Artificial Intelligence: A New Synthesis, Harcourt Publishers (TextBook)**
  By N. J. Nilsson | Harcourt Publishers

DIGITAL LEARNING CONTENT

Parul® University

www.paruluniversity.ac.in