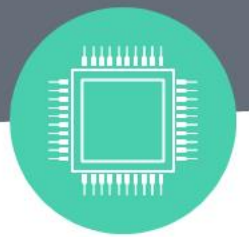
Artificial Intelligence

By

Dr. Manoj Kumar



University School of Automation and Robotics
GGSIP University, East Campus, Delhi, India

Ontology

- **An ontology, in the context of knowledge representation, refers to a formal, explicit specification of a conceptualization.**

- **It is a structured way of representing knowledge about a particular domain, which includes the concepts, entities, relationships, and rules that define that domain.**

- **Ontologies provide a shared and common vocabulary for a particular area of knowledge or subject matter, making it easier for humans and machines to understand and communicate about that domain**

Issues in Knowledge representation

The main objective of knowledge representation is **to draw the conclusions** from the knowledge, but there are many issues associated with the use of knowledge representation techniques.

- Are any attributes of objects so basic that they occur in almost every problem domain? If there are, we need to make sure that they are handled appropriately in each of the mechanisms we propose. If such attributes exist, what are they ?
- Are there any important relationships that exist among attributes of objects?
- At what level should knowledge be represented? Is there a good set of primitives into which all knowledge can be broken down? Is it helpful to use such primitives?
- How should set of objects be represented?
- Given a large amount of knowledge stored in a database, how can relevant parts be accessed when they are needed?

Issues in Knowledge representation

Some of the issues in knowledge representation are listed below:

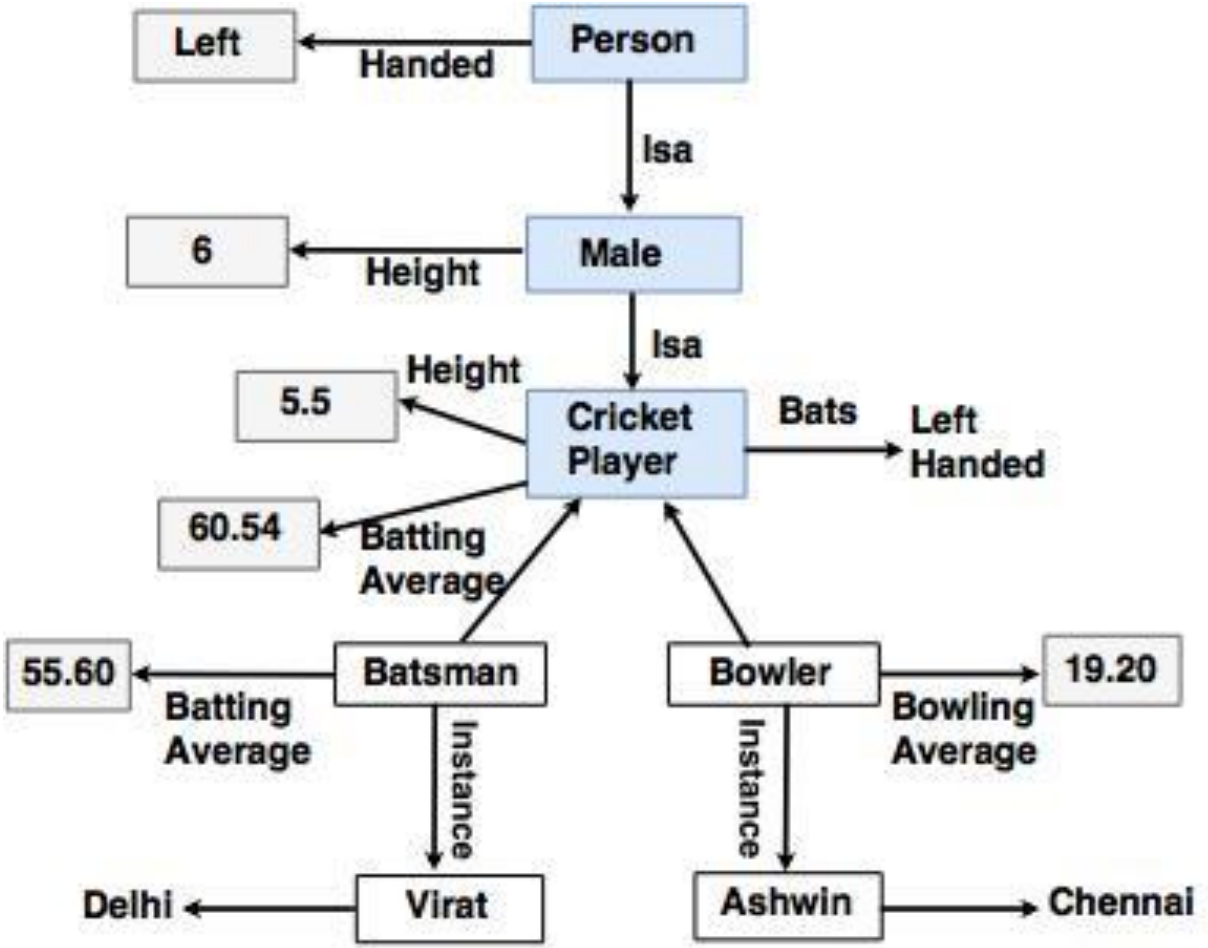


Fig: Inheratable Knowledge Representation

1. Important attributes

There are two attributes shown in the diagram, **instance** and **isa.** These attributes are of prime importance because they support property inheritance.

2. Relationships among attributes

Basically, the attributes used to describe objects are nothing but the entities. However, the attributes of an object do not depend on the encoded specific knowledge.

- *Inverses*
- *Existence in an isa hierarchy*
- *Techniques for reasoning about values*
- *Single-values attributes*

1.Inverses:

1. Inverses refer to a concept in ontologies and knowledge representation where a relationship between two attributes or classes is mirrored in reverse. In other words, if there's a relationship from A to B, the inverse relationship would exist from B to A. It helps capture bidirectional relationships.
2. Example: In a medical ontology, there may be an "has Symptom" relationship from a "Disease" class to a "Symptom" class. The inverse of this relationship could be "is Symptom of," connecting "Symptom" to "Disease.“

2.Existence in an isa Hierarchy:

1. In an "isa" (is-a) hierarchy, existence refers to whether a particular attribute or concept exists within the hierarchy. It pertains to whether an entity is a member of a specific class or category.
2. Example: In an ontology representing living organisms, the existence of "Dog" in the hierarchy would indicate whether "Dog" is considered a subclass of a broader class like "Mammal" or "Animal."

3. Techniques for Reasoning about Values:

1. Techniques for reasoning about values involve methods and algorithms used in knowledge representation and AI to infer or deduce information about attribute values based on the given knowledge and rules. These techniques are employed for making inferences or drawing conclusions.
2. Example: In a medical knowledge base, reasoning techniques might be used to deduce a patient's diagnosis based on their symptoms, medical history, and established medical rules.

4. Single-Values Attributes:

1. Single-values attributes refer to attributes or properties that have only one value associated with a given entity. In the context of a knowledge representation system, these attributes are unique for each entity.
2. Example: In a database for tracking employees, the "EmployeeID" attribute is typically a single-value attribute because each employee has a unique ID.

Issues in Knowledge representation

Some of the issues in knowledge representation are listed below:

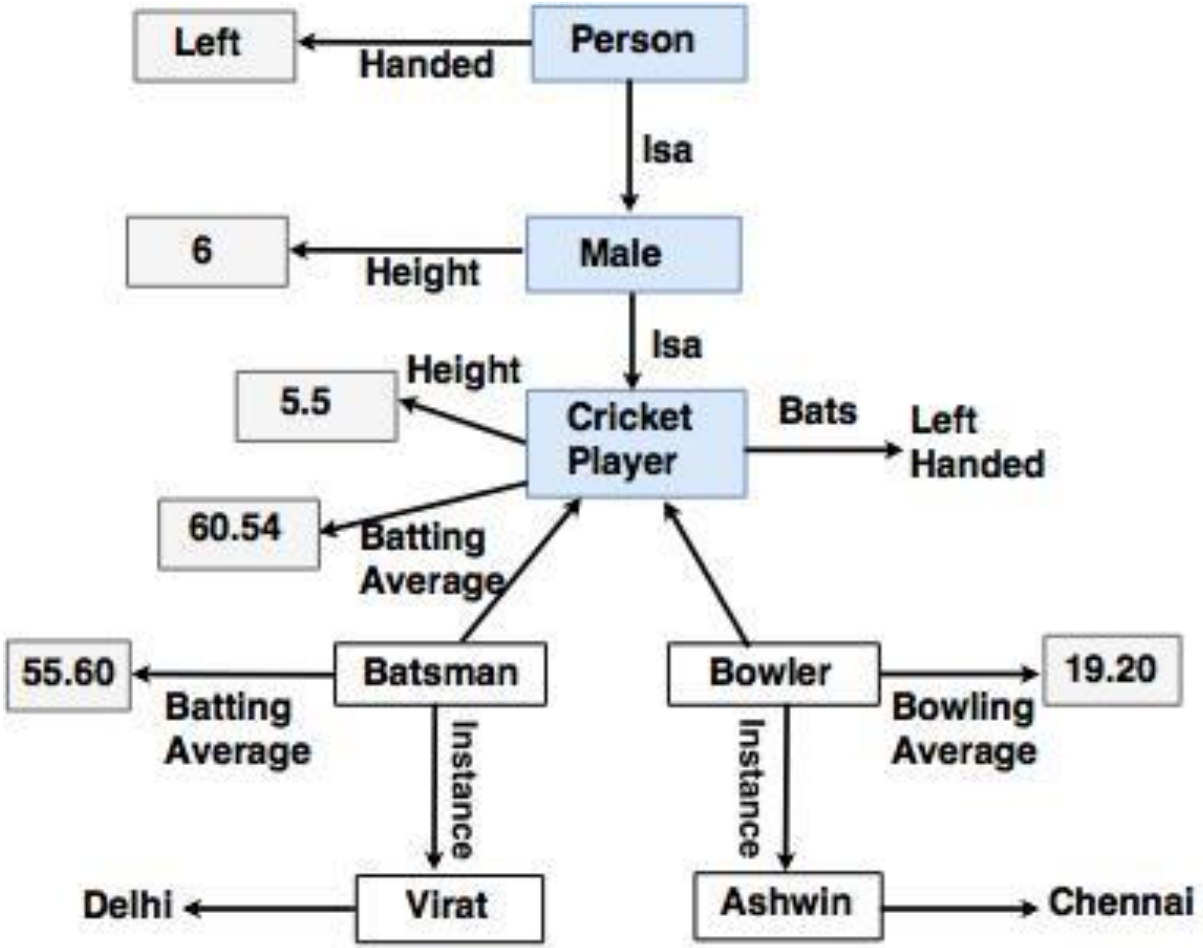


Fig: Inheratable Knowledge Representation

3. Choosing the granularity of representation

While deciding the granularity of representation (*the level of detail or refinement in the representation of data or information. How finely or coarsely you break down and describe elements within a dataset or a model*), it is necessary to know the following:

i. What are the primitives and at what level should the knowledge be represented?

ii. What should be the number (small or large) of low-level primitives or high-level facts?

High-level facts may be insufficient to draw the conclusion while Low-level primitives may require a lot of storage.

Issues in Knowledge representation

For example: Suppose that we are interested in following facts:
John spotted Alex.

Now, this could be represented as "Spotted (agent(John), object (Alex))"

Such a representation can make it easy to answer questions such as: Who spotted Alex?

Suppose we want to know : "Did John see Sue?"
Given only one fact, user cannot discover that answer.

Hence, the user can add other facts, such as "Spotted (x, y) → saw (x, y)"

Issues in Knowledge representation

4. Representing sets of objects.

There are some properties of objects which satisfy the condition of a set together but not as individual;

Example: Consider the assertion made in the sentences:

"There are more sheep than people in Australia", and "English speakers can be found all over the world. These facts can be described by including an assertion to the sets representing people, sheep, and English.

There are two ways to state a definition of a set and its element:

1.**Extensional** : list the members

2. **Intensional**: provide a rule that, when a particular object is evaluated, return true or false depending the object is in the set or not.

Ex:- extensional description of the set of our sun's planets on which people live is {earth} and intensional description is

$$\{x : sun\text{-}planet(x) \wedge human\text{-}inhabited(x)\}$$

extensional defined set {earth} has many intensional definitions

$$\{x : sun\text{-}planet\{x) \ ?\ nth\text{-}farthest\text{-}fmm\text{-}sun(x,\ 3)\}$$
$$\{x : sun\text{-}planet(x) \ ?\ nth\text{-}biggest(x,\ 5)\}$$

Issues in Knowledge representation

5. Finding the right structure as needed

To describe a particular situation, it is always important to find the access of right structure. This can be done by selecting an initial structure and then revising the choice.

However, in order to have access to the right structure for describing a particular situation. ***it is necessary to solve all of the following problems.***

1. How to perform an initial selection of the most appropriate structure.
2. How to fill in appropriate details from the current situation.
3. How to find a better structure if the one chosen initially turns out not to be appropriate.
4. What to do if none of the available structures is appropriate,
5. When to create and remember a new structure.

There is no good, general purpose method for solving all these problems, Some knowledge-representation techniques solve some of them. In this section we survey some solutions to two of these problems: how to select an initial structure to consider and how to find a better structure if that one tums out not to be a good match.

Issues in Knowledge representation

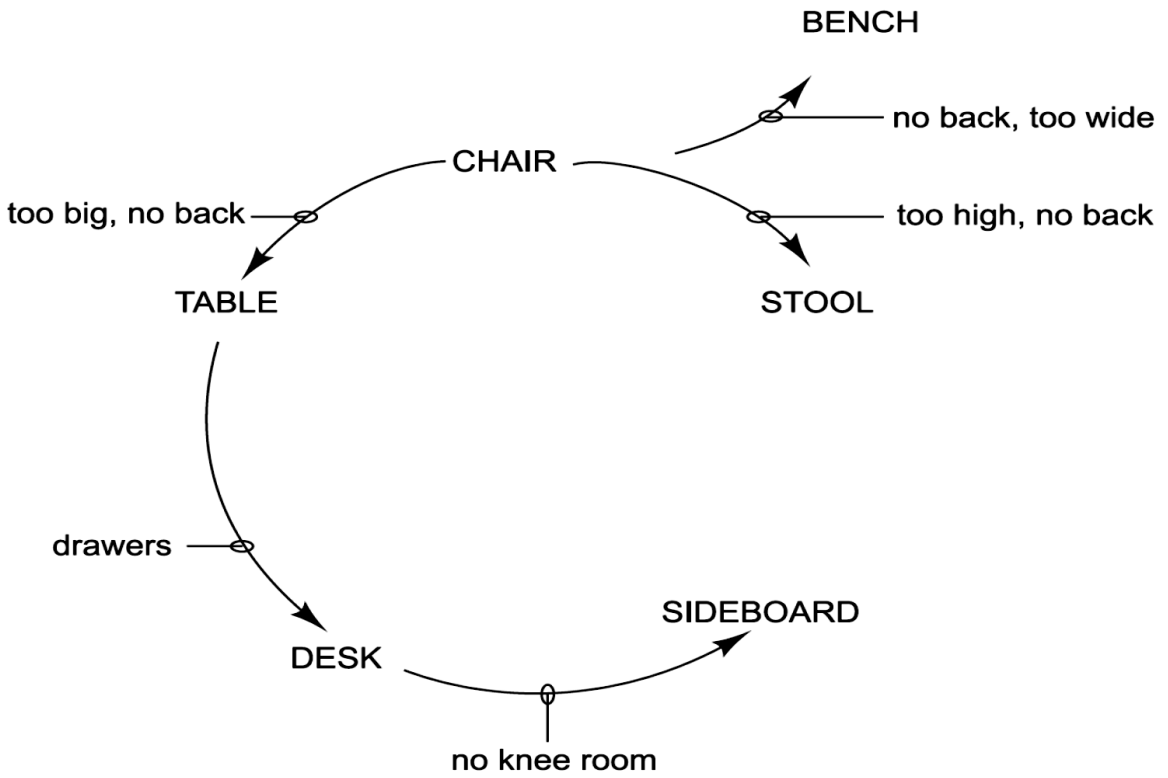
Selecting an Initial Structure

- Index structures by English words

  - John flew to New York.
    * He rode in a plane from one place to another.
  - John flew a kite.
    * He held a kite that was up in the air.
  - John flew down the street.
    * He moved very rapidly.
  - John flew into a rage.
    * An idiom.

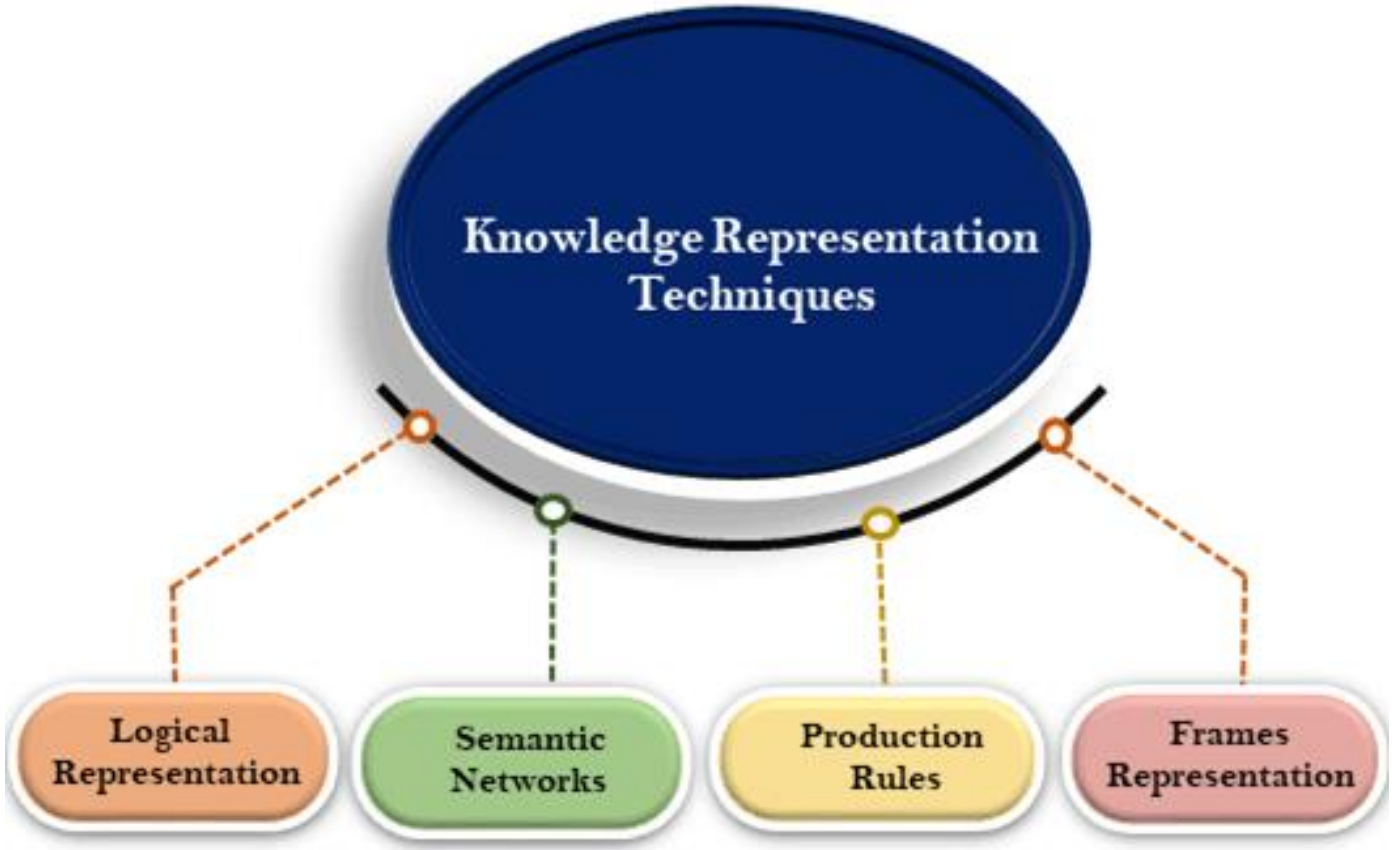
- Index structures by concepts
- Use one major clue.

Revising the choice when necessary: A Similarity Net



Types of knowledge representation

Basically 4 types of knowledge representation in AI



Logical representation

- Logical representation of knowledge is a way of representing information and knowledge using formal logic, typically symbolic or mathematical representations, to capture the relationships, facts, and rules that govern a particular domain of knowledge. This approach allows for precise, unambiguous, and systematic representation of knowledge, making it suitable for computer-based reasoning and inference.

- In order to give information to agent and get info without errors in communication.
- Logic is based on truth.
- There are 2 types of LR
- 1)propositional logic(PL)
- 2)first order predicate logic(FOL)

Facts are the general statements that may be either True or False. Thus, logic can be used to represent such simple facts.

Logical representation

- User has to define a set of primitive symbols along with the required semantics.

- The symbols are assigned together to define legal sentences in the language for representing TRUE facts.

- New logical statements are formed from the existing ones. The statements which can be either TRUE or false but not both , are called propositions. A declarative sentence expresses a statement with a proposition as content;
  Example: The declarative "Cotton is white" expresses that Cotton is white. So, the sentence "Cotton is white" is a true statement.

Propositional knowledge

•Propositional logic is a study of propositions.
•Each proposition has either a true or a false value but not both at a time.
•Propositions is represented by variables.

For example: Symbols 'p' and 'q' can be used to represent propositions.
There are two types of propositions:
1. Simple Preposition **(Atomic)**
2. compound Prepositions. **(complex)**

1. A simple preposition: It does not contain any other preposition.
For example: Rocky is a dog.

2. A compound preposition: It contains more than one prepositions.
For example: Surendra is a boy and he likes chocolate.

Propositional knowledge

Imagine you have a light switch, and it can be either ON or OFF. This is a bit like propositional logic. In this system:

•You have statements that are like switches. They can be either true (ON) or false (OFF).

•You use simple words like "AND," "OR" and "NOT" to combine these switches and make new statements.

Example:

Let's say we have two switches, A and B. They can be either ON (true) or OFF (false).

•A is ON, which means it's true.

•B is OFF, which means it's false.

Now, we can use propositional logic to combine these switches with words like "AND" and "OR":

1.A AND B: This is true only if both A and B are true. In our case, it's false because B is OFF.

2.A OR B: This is true if either A or B (or both) are true. In our case, it's true because A is ON.

3.NOT A: This is the opposite of A. It's false because A is true.

Propositional knowledge

Scenario: Weather Forecast
In this scenario, we'll use propositional logic to express statements about the weather forecast using simple true (T) or false (F) values.
1.Statement A: "It will rain tomorrow." (True if the weather forecast predicts rain, otherwise false)
2.Statement B: "It will be sunny tomorrow." (True if the weather forecast predicts sunshine, otherwise false)
Now, let's create some logical combinations:
3.A AND B: This statement is true only if both A and B are true. In other words, it will be true if the forecast predicts both rain and sunshine for the same day, which is logically inconsistent. Therefore, this combination is usually false.
4.A OR B: This statement is true if either A or B (or both) is true. It reflects the possibility that the forecast could predict either rain or sunshine or both.
5.NOT A: This is the opposite of A. It's true if the forecast does not predict rain, indicating a dry day.

Propositional knowledge

To make these atomic and complex sentences some symbols helps.

Connectives and the truth tables of compound prepositions are given below:
Consider 'p' and 'q' are two prepositions then,

1. Negation (¬p) indicates the opposite of p.

p	¬p
0	1
1	0

Today is Friday : p
Today is not Friday: ¬p

2. Conjunction (p ∧ q) indicates that p and q both and are enclosed in parenthesis. So, p and q are called conjuncts .

p	q	p ∧ q
0	0	0
0	1	0
1	0	0
1	1	1

You should attend the lecture and make the nodes.

Propositional knowledge

3. Disjunction (p ∨ q) indicates that either p or q or both are enclosed in parenthesis. Thus, p and q are called disjuncts.

p	q	p ∨ q
0	0	0
0	1	1
1	0	1
1	1	1

You should eat or watch tv at a time.

4. Implication (p ⇒ q) consists of a pair of sentences separated by the ⇒ operator and enclosed in parentheses. The sentence to the left of the operator is called as an antecedent, and the sentence to the right is called as the consequent.

p	q	p ⇒ q
0	0	1
0	1	1
1	0	0
1	1	1

If there is rain then the roads are wet.

Propositional knowledge

5. Equivalence (p ⇔ q) is a combination of an implication and a reduction. **If-And-Only-If (Iff) Statement ,** ≡ is a logical operator that represents a biconditional relationship between two propositions. A biconditional states that two statements are equivalent and have the same truth value, meaning they are either both true or both false. If one of the propositions is true, the other must also be true, and if one is false, the other must also be false.

p	q	p ⇔ q
0	0	1
0	1	0
1	0	0
1	1	1

Example 2: I will go to the mall if and only if I have to do shopping.

Example 1: Let P represent "It is raining," and Q represent "I will use an umbrella." You can express the statement "I will use an umbrella if and only if it is raining" using the equivalence operator as follows:

This means that you will use an umbrella if and only if it is raining, and vice versa. If it's not raining, you won't use an umbrella, and if it is raining, you will use an umbrella.

Logical representation: FOPL

First-order Predicate Logic

- First-order predicate calculus (FOPL) was developed by logicians to extend the expressiveness of Propositional Logic.
- It is generalization of propositional logic that permits reasoning about world entities (objects) as well as classes and subclasses of objects.
- Prolog is also based on FOPL.
- Predicate logic uses variables and quantifiers which is not present in propositional logic.

First-order predicate logic, often simply referred to as first-order logic or FOL, is a formal system of mathematical logic that extends propositional logic by allowing for the representation of complex statements involving **quantifiers**, **variables**, **predicates**, and **objects**. It is a powerful and expressive logic used in various fields, including mathematics, computer science, philosophy, and artificial intelligence.

Predicates: Predicates in FOL represent relationships between objects or properties. They are used to make statements about objects and are often denoted by symbols followed by variables. For example, "P(x)" could represent the predicate "x is a cat," where 'x' is a variable that can be instantiated with specific objects.

Logical representation: FOPL

the capability of reasoning and knowledge representation using predicate logic is higher than prepositional logic.
For instance, it includes two more quantifiers, namely, the essential quantifier $(\forall)$ and the existential quantifier$(\exists)$. To illustrate the use of the quantifiers, let us consider the following pieces of knowledge.

Knowledge 1: All boys like sweets.
Using predicate logic, we can write the above statement as

$$\forall X \left(Boy\left(X \right) \right) \rightarrow Likes\left(X ,sweets \right)$$

Knowledge 2: Some boys like flying kites.
Using predicate logic, the above statement can be represented as

$$\exists X \left(Boy\left(X \right) \rightarrow Likes\left(X , Flying - kites \right) \right)$$

Before describing predicate logic (PL) or first order logic (FOL) in a formal manner, we first present the alphabets of FOL.

Logical representation: FOPL

Alphabets of FOL

The alphabets of FOL are of the following types:

1. Constants: a, b, c
2. Variables: X, Y, Z
3. Functions: f, g, h
4. Operators: $\wedge, \vee, \neg, \rightarrow$
5. Quantifiers: $\forall, \exists$
6. Predicate: P, Q, R

FOPL: Example

1. Marcus was a man.
2. Marcus was a Pompeian.
3. All Pompeian were Romans.
4. Caesar was a ruler.
5. All Romans were either loyal to Caesar or hated him.
6. Everyone is loyal to someone.
7. People only try to assassinate rulers they are not loyal to.
8. Marcus tried to assassinate Caesar.

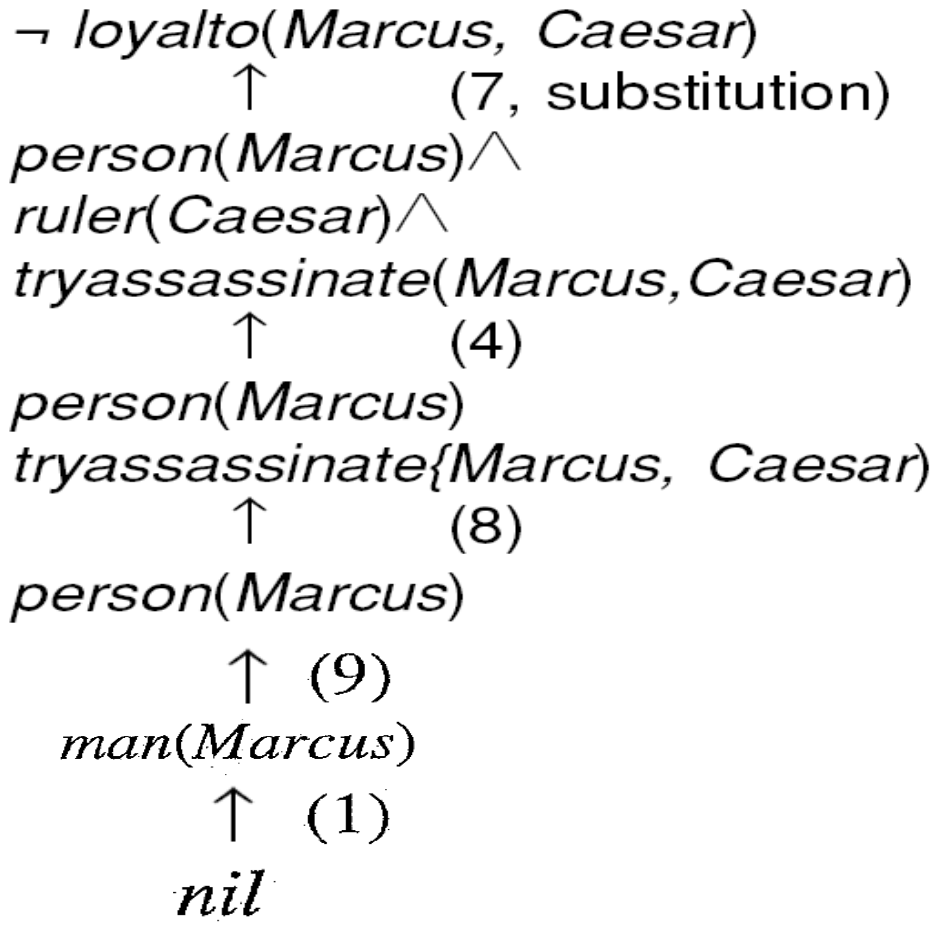
Prove Marcus not loyal to Caesar?

FOPL: Example

1. Marcus was a man.
   $man(Marcus)$
2. Marcus was a Pompeian.
   $Pompeian(Marcus)$
3. All Pompeians were Romans.
   $\forall x : Pompeian(x) \rightarrow Roman(x)$
4. Caesar was a ruler.
   $ruler(Caesar)$
5. All Romans were either loyal to Caesar or hated him.
   $\forall x : Roman(x) \rightarrow loyalto(x, Caesar) \vee hate(x, Caesar)$
6. Everyone is loyal to someone.
   $\forall x : \exists y : loyalto(x, y)$
7. People only try to assassinate rulers they aren't loyal to.
   $\forall x : \forall y : person(x) \wedge ruler(y) \wedge tryassassinate(x, y)$
   $\rightarrow \neg loyalto(x, y)$
8. Marcus tried to assassinate Caesar.
   $tryassassinate(Marcus, Caesar)$
9. All men are people.
   $\forall x : man(x) \rightarrow person(x)$

FOPL: Example

An Attempt to Prove $\neg\ loyalto(Marcus,\ Caesar)$



Production Rule

A production rules system works in a simple way. It's like having a set of rules that say, "If something is true, then do something (**condition, action**)."
There are three main parts to this system:
1.The Set of Rules: This is where all the rules are kept. Each rule has two parts - a condition (a statement that needs to be true) and an action (what to do if that condition is true).

2.Working Memory: Think of this like the agent's notepad. It keeps track of what's happening right now. Rules can also add things to this notepad.

3.Recognize-Act Cycle: This is the process where the agent looks at the rules and checks if the conditions are true. If they are, it follows the actions. This whole process is like a cycle.
Sometimes, many rules might be true at the same time. When this happens, it's called a "conflict set." The agent has to pick which rule to follow, and that's called "conflict resolution." It's like making a decision when you have many choices.

Example:
- **IF (at bus stop AND bus arrives) THEN action (get into the bus)**
- **IF (on the bus AND paid AND empty seat) THEN action (sit down).**
- **IF (on bus AND unpaid) THEN action (pay charges).**
- **IF (bus arrives at destination) THEN action (get down from the bus).**

Production Rule

Advantages of Production Rules:

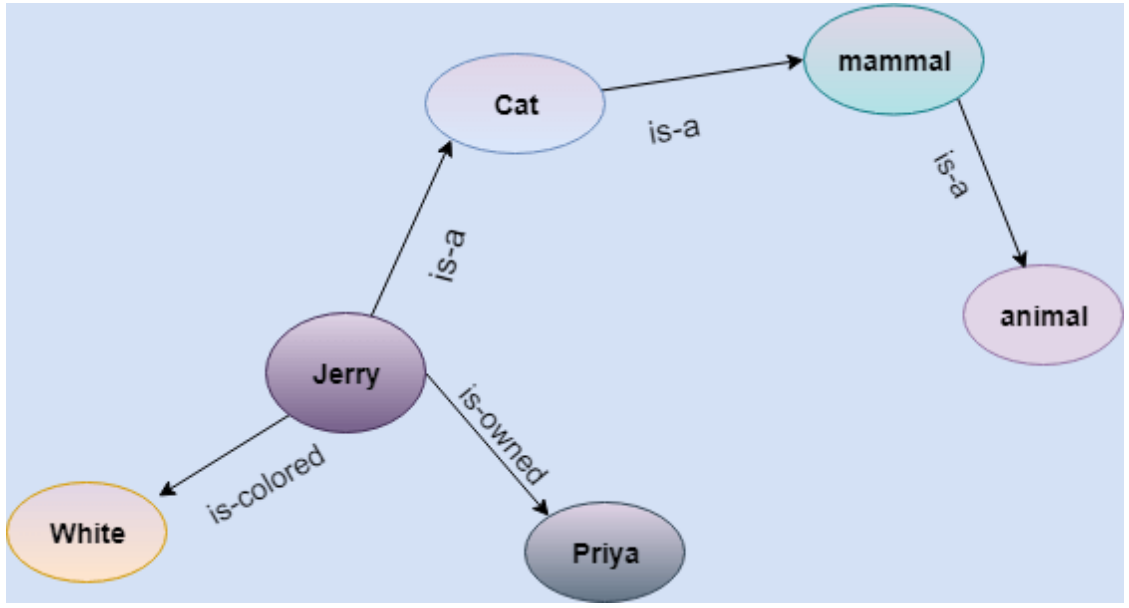
1.Easy to Understand: Production rules are written in everyday language, so they're easy to grasp.
2.Flexible and Changeable: You can change these rules easily. It's like adding, removing, or modifying individual instructions in a recipe.

Disadvantages of Production Rules:

1.No Learning: These rules can't learn from experience. They don't remember what they did before to solve a problem.
2.Sometimes Slow: When a computer uses lots of rules at once, it can be slow. It's like having too many chefs in the kitchen – things might get messy.

SEMENTIC NETWORK (Meaningful Graph)

- Semantic networks serve as an alternative to predicate logic for knowledge representation in the field of artificial intelligence
- These networks provide a graphical means to express knowledge.
- Within a semantic network, information is organized into a structure comprising nodes that represent objects and arcs connecting these nodes, signifying relationships between the objects.
- Semantic networks allow for the categorization of objects in various ways and facilitate the establishment of connections between them.
- One of the notable advantages of semantic networks is their inherent simplicity, making them readily comprehensible, and their capacity for straightforward expansion and modification.



1.Jerry is a cat.
2.Jerry is a mammal
3.Jerry is owned by Priya.
4.Jerry is brown colored.
5.All Mammals are animal.

Instance and Is-a

"Instance" and "Is-a" are fundamental concepts in knowledge representation, often used in the context of knowledge modeling, particularly in semantic networks and ontologies. They are used to describe relationships between entities or concepts.

Instance:

- An "instance" represents a specific individual or object that is a member of a broader category or class.
- It is used to denote a unique, concrete entity that can be identified or referred to.
- Instances are typically represented as specific examples or members of a class.

Example: In the context of animals, "Fido" might be an instance of the class "Dog." "Fido" is a specific, individual dog, and "Dog" is the class representing the broader category of dogs.

Is-a (Subsumption):

Instance:

- An "instance" represents a specific individual or object that is a member of a broader category or class.
- It is used to denote a unique, concrete entity that can be identified or referred to.
- Instances are typically represented as specific examples or members of a class.

Example: In the context of animals, "Fido" might be an instance of the class "Dog." "Fido" is a specific, individual dog, and "Dog" is the class representing the broader category of dogs.

SEMENTIC NETWORK

Drawbacks in Semantic Representation:

1.Slow Problem Solving: Semantic networks can be slow because we have to look through the entire network to answer questions. Sometimes, even after searching everything, we might not find the answer.

2.Too Much Data: Semantic networks try to work like the human brain, but in reality, it's impossible to create such a huge network. Our brain has trillions of neurons and connections.

3.Lack of Precise Words: These networks don't have clear words like "for all" or "for some." This makes it hard to express ideas precisely.

4.No Standard Names: There are no fixed names for the connections in semantic networks. It can be confusing because different systems might use different names for the same thing.

5.Not Smart on Their Own: Semantic networks depend on the people who create them. They don't think or learn by themselves; they can only do what they were made for.

SEMENTIC NETWORK

Drawbacks in Semantic Representation:

1.Natural Way to Show What We Know: Semantic networks are a natural way to show the things we know or understand.

2.Clear and Easy to Understand: These networks make it easy to see what things mean. It's like a clear window to the meaning.

3.Simplicity and Easy Comprehension: They are simple and not hard to understand. You don't need to be an expert to get what they're saying.

FRAME representation

- A frame is like a structured record that describes something in the world. It's used in AI to organize information into categories and has slots with names and values, which we call facets.
- Frame representation in AI is like filling out a form to describe something. Each form (frame) has fields (slots or objects) to fill out with specific information.

Facets: The various aspects of a slot is known as Facets. Facets are features of frames which enable us to put constraints on the frames. Example: IF-NEEDED facts are called when data of any particular slot is needed. A frame may consist of any number of slots, and a slot may include any number of facets and facets may have any number of values. A frame is also known as slot-filter knowledge representation in artificial intelligence.

Frames come from semantic networks and have evolved into the classes and objects we use in modern programming. A single frame by itself isn't very useful; we need a bunch of frames connected together. Frames let us store knowledge about objects or events in one place in our database. They're widely used in applications like natural language processing and machine vision.

FRAME representation: Example

Imagine we have a frame to describe a car. This car frame has slots like "Color," "Make," "Model," and "Year." Here's how it might look:

- Color: [Enter color here]
- Make: [Enter make here]
- Model: [Enter model here]
- Year: [Enter year here]

Now, you can fill in the values for each slot:

- Color: Red
- Make: Toyota
- Model: Camry
- Year: 2020

FRAME representation

- Let's suppose we are taking an entity, Peter. Peter is an engineer as a profession, and his age is 25, he lives in city London, and the country is England. So following is the frame representation for this:

Slots	Filters
Title	Artificial Intelligence
Genre	Computer Science
Author	Peter Norvig
Edition	Third Edition
Year	1996
Page	1152

FRAME representation

Advantages of frame representation:

1.Grouping related data makes programming easier.

2.Frame representation is flexible and widely used in AI applications.

3.Adding new attributes and relationships is simple.

4.Default data can be included, and missing values can be easily found.

5.Frame representation is straightforward and easy to grasp.

Disadvantages of frame representation:

1.Inference mechanisms in frame systems can be challenging to process.

2.Smooth processing of inference mechanisms may not be supported by frame representation.

3.Frame representation tends to have a more generalized approach.

FRAME representation

Example: Continuing with the animal domain, you could define a "Dog" frame with slots for "Breed," "Color," and "Average Lifespan." Each slot would contain specific values.

Frame: Dog
- Breed: Golden Retriever
- Color: Yellow
- Average Lifespan: 10-12 years

Inheritance

Inheritance is a fundamental concept in knowledge representation.
It allows for the propagation of properties or attributes from a parent object to its child objects in a hierarchical structure.
This means that if an attribute is defined at a higher level in the hierarchy, it is automatically inherited by the objects lower in the hierarchy.

Example: In the animal domain, if you define an attribute "Legs" at the level of "Animal," all specific types of animals like "Dog," "Cat," and "Bird" inherit this property.

Animal -has-> Legs
Dog
Cat
Bird

Constraint propagation

Constraint propagation is a technique used in knowledge representation to maintain and propagate constraints or rules within a knowledge base. The purpose of constraint propagation is to ensure that the knowledge base remains consistent and that any changes to one part of the knowledge base are reflected in related parts.

Constraint propagation involves using rules and constraints to enforce consistency in a knowledge base. When new information is added or existing information is modified, these constraints are checked to ensure that they are not violated. If a constraint is violated, the system takes appropriate actions to restore consistency. This can involve modifying other parts of the knowledge base or even preventing the new information from being added.

Constraint propagation

Example: Scheduling Problem

Imagine you are building a scheduling system for a university. The system needs to schedule classes in a way that avoids conflicts, such as two classes scheduled in the same room at the same time or two professors teaching two different classes at the same time. To represent this scheduling problem, you can use constraint propagation.

•**Constraints:**

- No two classes can be scheduled in the same room at the same time.
- No professor can teach two different classes at the same time.
- Each class must have a room and a professor assigned to it.

•**Knowledge Base:**

•Your knowledge base contains information about classes, professors, rooms, and their respective schedules. You have frames representing classes, professors, and rooms, and each frame has slots for its schedule.

•**Constraint Propagation:**

•When you add a new class to the schedule or change the schedule of an existing class, the system needs to check and propagate constraints to ensure consistency. For example, if you schedule a new class in Room A at 10:00 AM, the system must check if Room A is available at that time, if the professor is available, and if no other class is already scheduled in Room A at that time.

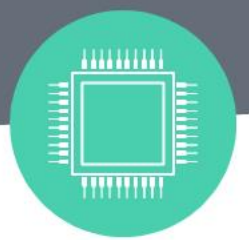
•If any of these constraints are violated, the system must take corrective actions, such as finding an available room, adjusting the professor's schedule, or rescheduling conflicting classes.

Constraint propagation ensures that the scheduling system remains consistent and that scheduling decisions do not lead to conflicts or violations of the defined constraints. It's a crucial part of building intelligent systems that can handle complex real-world scenarios while maintaining logical consistency.

Artificial Intelligence

By

Dr. Manoj Kumar



University School of Automation and Robotics
GGSIP University, East Campus, Delhi, India

Logic Programming

✓Logic programming is a programming language paradigm in which logical assertions are viewed as programs, e.g : PROLOG

✓A PROLOG program is described as a series of logical assertions, each of which is a Horn Clause.

✓A Horn Clause is a clause that has at most one positive literal.

✓Eg p, ¬ p V q etc are also Horn Clauses.

✓The fact that PROLOG programs are composed only of Horn Clauses and not of arbitrary logical expressions has two important consequences.

- ✓Because of uniform representation a simple & effective interpreter can be written.
- ✓The logic of Horn Clause systems is decidable.

Logic Programming: A Declarative and a Procedural Representation

$$\forall x : pet(x) \wedge small(x) \rightarrow apartmentpet(x)$$
$$\forall x : cat(x) \vee dog(x) \rightarrow pet(x)$$
$$\forall x : poodle(x) \rightarrow dog(x) \wedge small(x)$$
$$poodle(ftujfy)$$

A Representation in Logic

```
apartmentpet(X) :- pet(X), small(X).
pet(X) :- cat(X).
pet(X) :- dog(X).
dog(X) :- poodle (X) .
small(X) :- poodle(X).
poodle(fluffy).
```

A Representation in PROLOG

Logic Programming: Answering Questions in PROLOG

```
apartmentpet(X) :- pet(X), small(X).
pet(X) :- cat(X).
pet(X) :- dog(X).
dog(X) :- poodle (X) .
small(X) :- poodle(X).
poodle(fluffy).

?- apartmentpet(X).

?- cat(fluffy).
```

Logic Programming

✓**Logic Programming**

- ✓Even PROLOG works on backward reasoning.
- ✓The program is read top to bottom, left to right and search is performed depth-first with backtracking.
- ✓There are some syntactic difference between the logic and the PROLOG representations.
- ✓The key difference between the logic & PROLOG representation is that PROLOG interpreter has a fixed control strategy, so assertions in the PROLOG program define a particular search path to answer any question.

Where as Logical assertions define set of answers that they justify, there can be more than one answers, it can be forward or backward tracking .

Logic Programming

- **Logic Programming**
  - Control Strategy for PROLOG states that we begin with a problem statement, which is viewed as a goal to be proved.
  - Look for the assertions that can prove the goal.
  - To decide whether a fact or a rule can be applied to the current problem, invoke a standard unification procedure.
  - Reason backward from that goal until a path is found that terminates with assertions in the program.
  - Consider paths using a depth-first search strategy and use backtracking.
  - Propagate to the answer by satisfying the conditions.

Logic Programming

- **Forward v/s Backward Reasoning**
  - The objective of any search is to find a path through a problem space from the initial to the final one.
  - There are 2 directions to go and find the answer
    - Forward
    - Backward
  - 8-square problem
  - **Reason forward from the initial states** : Begin building a tree of move sequences that might be solution by starting with the initial configuration(s) at the root of the tree. Generate the next level of tree by finding all the rules whose left sides match the root node and use the right sides to create the new configurations. Generate each node by taking each node generated at the previous level and applying to it all of the rules whose left sides match it. Continue

Logic Programming

✓**Reason backward from the goal states** : Reason backward from the goal states : Begin building a tree of move sequences that might be solution by starting with the goal configuration(s) at the root of the tree. Generate the next level of tree by finding all the rules whose right sides match the root node and use the left sides to create the new configurations. Generate each node by taking each node generated at the previous level and applying to it all of the rules whose right sides match it. Continue. This is also called Goal-Directed Reasoning.

✓To summarize, to reason forward, the left sides(pre conditions) are matched against the current state and the right sides(the results) are used to generate new nodes until the goal is reached.

✓To reason backwards, the right sides are matched against the current node and the left sides are used to generate new nodes.

Constraint Propagation

- Constraint propagation is a technique used in knowledge representation to maintain and propagate constraints or rules within a knowledge base.
- The purpose of constraint propagation is to ensure that the knowledge base remains consistent and that any changes to one part of the knowledge base are reflected in related parts.

- Constraint propagation involves using rules and constraints to enforce consistency in a knowledge base.
- When new information is added or existing information is modified, these constraints are checked to ensure that they are not violated.
- If a constraint is violated, the system takes appropriate actions to restore consistency.
- This can involve modifying other parts of the knowledge base or even preventing the new information from being added.

Constraint Propagation Example

Imagine you are building a scheduling system for a university. The system needs to schedule classes in a way that avoids conflicts, such as two classes scheduled in the same room at the same time or two professors teaching two different classes at the same time. To represent this scheduling problem, you can use constraint propagation.

•**Constraints:**

- No two classes can be scheduled in the same room at the same time.
- No professor can teach two different classes at the same time.
- Each class must have a room and a professor assigned to it.

•**Knowledge Base:**

•Your knowledge base contains information about classes, professors, rooms, and their respective schedules. You have frames representing classes, professors, and rooms, and each frame has slots for its schedule.

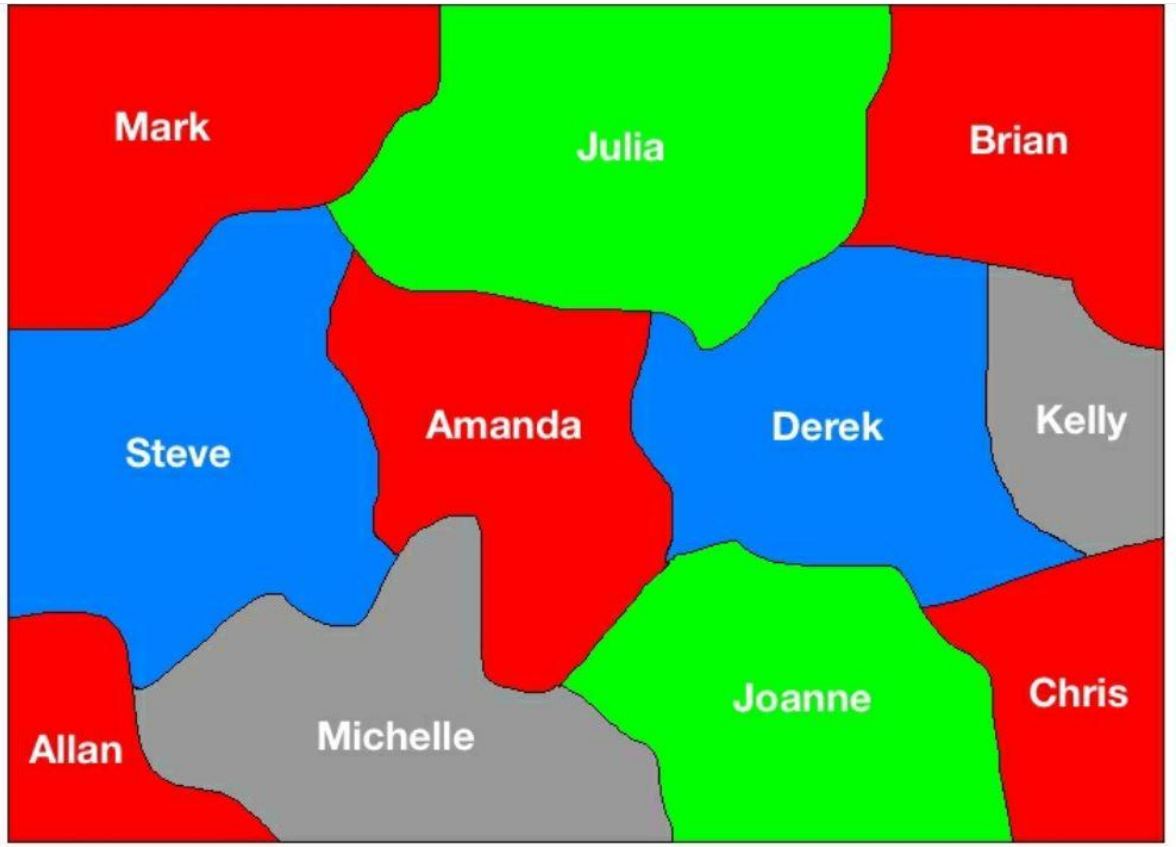
•**Constraint Propagation:**

•When you add a new class to the schedule or change the schedule of an existing class, the system needs to check and propagate constraints to ensure consistency. For example, if you schedule a new class in Room A at 10:00 AM, the system must check if Room A is available at that time, if the professor is available, and if no other class is already scheduled in Room A at that time.

•If any of these constraints are violated, the system must take corrective actions, such as finding an available room, adjusting the professor's schedule, or rescheduling conflicting classes.

Constraint Propagation Example

- Constraint propagation ensures that the scheduling system remains consistent and that scheduling decisions do not lead to conflicts or violations of the defined constraints.
- It's a crucial part of building intelligent systems that can handle complex real-world scenarios while maintaining logical consistency.



Constraint Propagation Example

In a Sudoku puzzle, the goal is to fill a 9x9 grid with numbers so that each row, column, and the nine 3x3 subgrids contain all of the digits from 1 to 9 without any repetitions. Constraint propagation plays a crucial role in solving Sudoku puzzles. The constraints in Sudoku are as follows:

1.Each row must contain the digits 1-9 with no repetitions.
2.Each column must contain the digits 1-9 with no repetitions.
3.Each 3x3 sub-grid (or box) must contain the digits 1-9 with no repetitions.

Semantic Network

- The idea behind a semantic network is that knowledge is often best understood as a set of concepts that are related to one another.
- *The meaning of a concept is defined by its relationship to other concepts.*
- A semantic network consists of a set of nodes that are connected by labeled arcs. The nodes represent concepts and the arcs represent relations between concepts.

•**Definition:** Semantic networks are graphical representations of knowledge that consist of nodes (representing concepts or objects) connected by labeled links (representing relationships or attributes). They are used to depict how different entities in a domain are related to each other.
•**Example:** Consider a semantic network representing information about animals. You might have nodes for "Dog," "Cat," and "Bird," and links such as "is a" to represent the fact that a "Dog" is a type of "Animal."

Animal -is a-> Dog
-is a-> Cat
-is a-> Bird

Common Semantic Relation

INSTANCE: X is an INSTANCE of Y if X is a specific example of the general concept Y.

Example: *Elvis* is an **INSTANCE** of *Human*

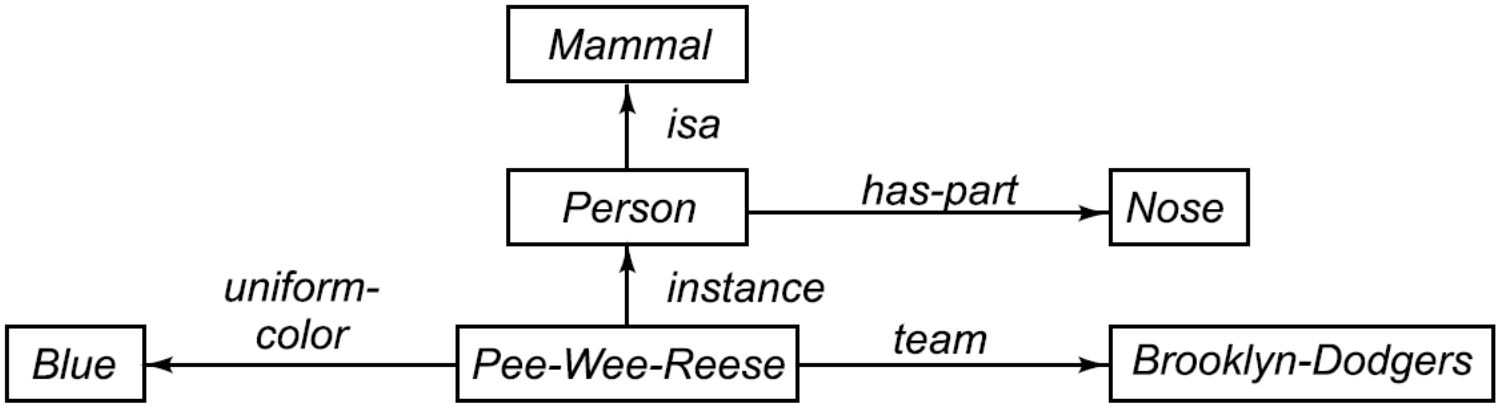
ISA: X ISA Y if X is a subset of the more general concept Y.

Example: *sparrow* **ISA** *bird*

HASPART: X HASPART Y if the concept Y is a part of the concept X. (Or this can be any other property)

Example: *sparrow* **HASPART** *tail*

Semantic Network



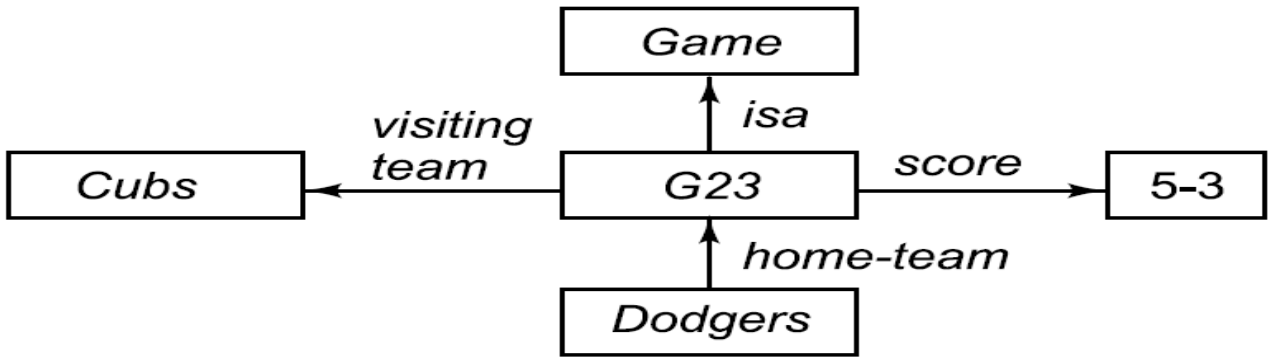
Semantic Network

- **Unary Predicates** can be rewritten as binary ones.

$$man(Marcus)$$

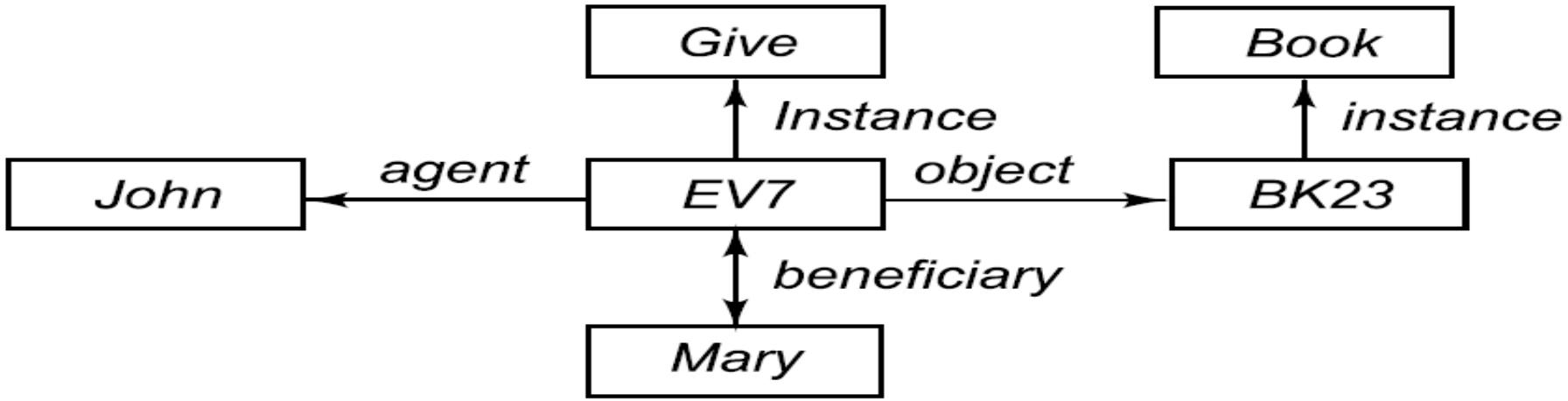
$$instance(Marcus,\ Man)$$

$$score(Cubs,\ Dodgers,\ 5\text{-}3)$$

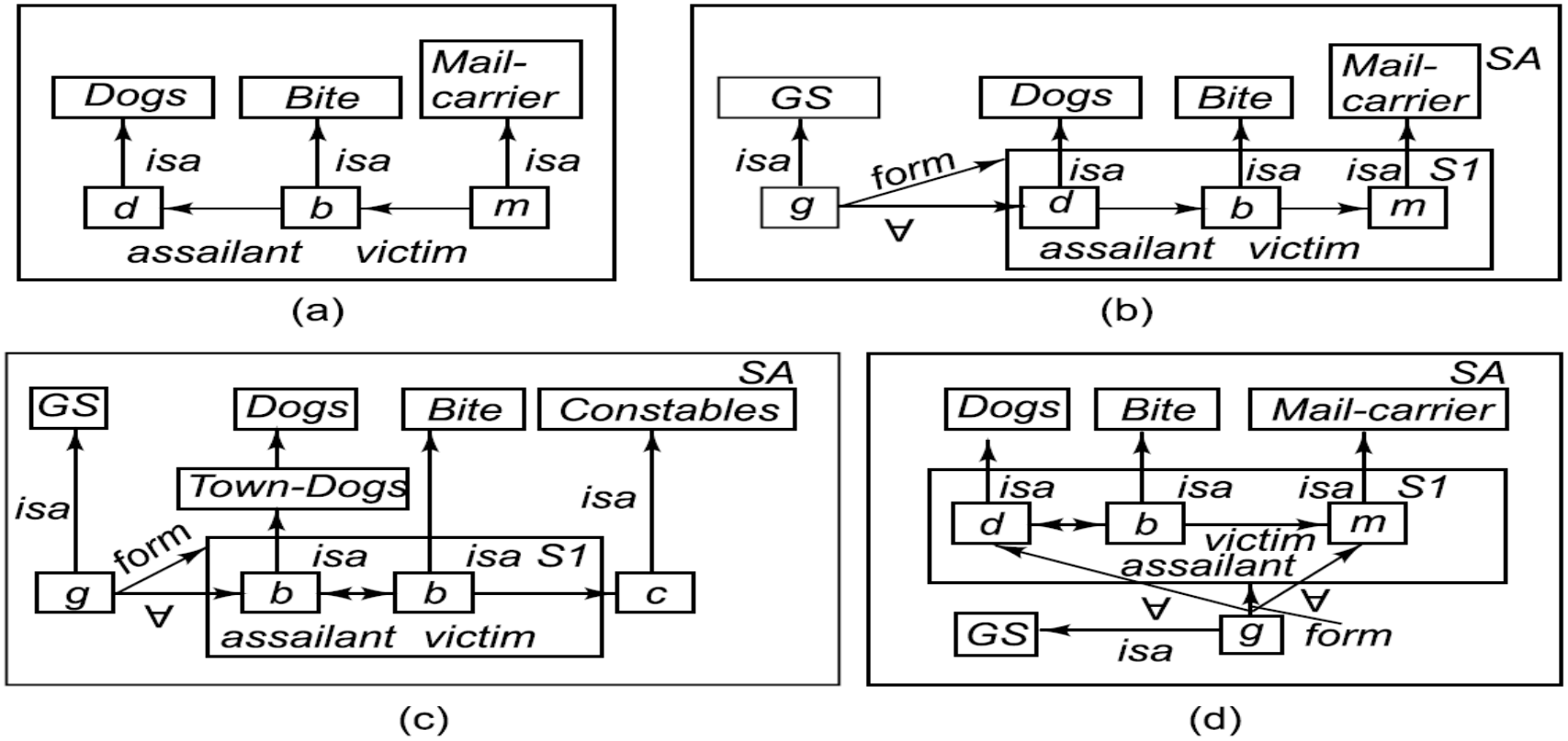


Semantic Network

- "John gave the book to Mary."



Partitioned Semantic Network



a) The dog bit the mail carrier.

b) Every dog has bitten a mail carrier.

c) Every dog in town has bitten the constable.

d) Every dog has bitten every mail carrier.

Semantic Network Advantages

- Semantic nets have the ability to represent default values for categories.
- Semantic nets convey some meaning in a transparent manner.
- Semantic nets are simple and easy to understand.
- Semantic nets are easy to translate into PROLOG.

Disadvantages

- Links between the objects represent only binary relations. For example, the sentence Run(ChennaiExpress, Chennai,Bangalore,Today) cannot be asserted directly.
- Some types of properties are not easily expressed using a semantic network. For example: negation, disjunction, and general non-taxonomic knowledge.
- There is no standard definition of link names.

Frames

- A frame represents an entity as a set of slots (attributes) and associated values.
- Each slot may have constraints that describe legal values that the slot can take.
- A frame can represent a specific entity, or a general concept.
- Frames are implicitly associated with one another because the value of a slot can be another frame.

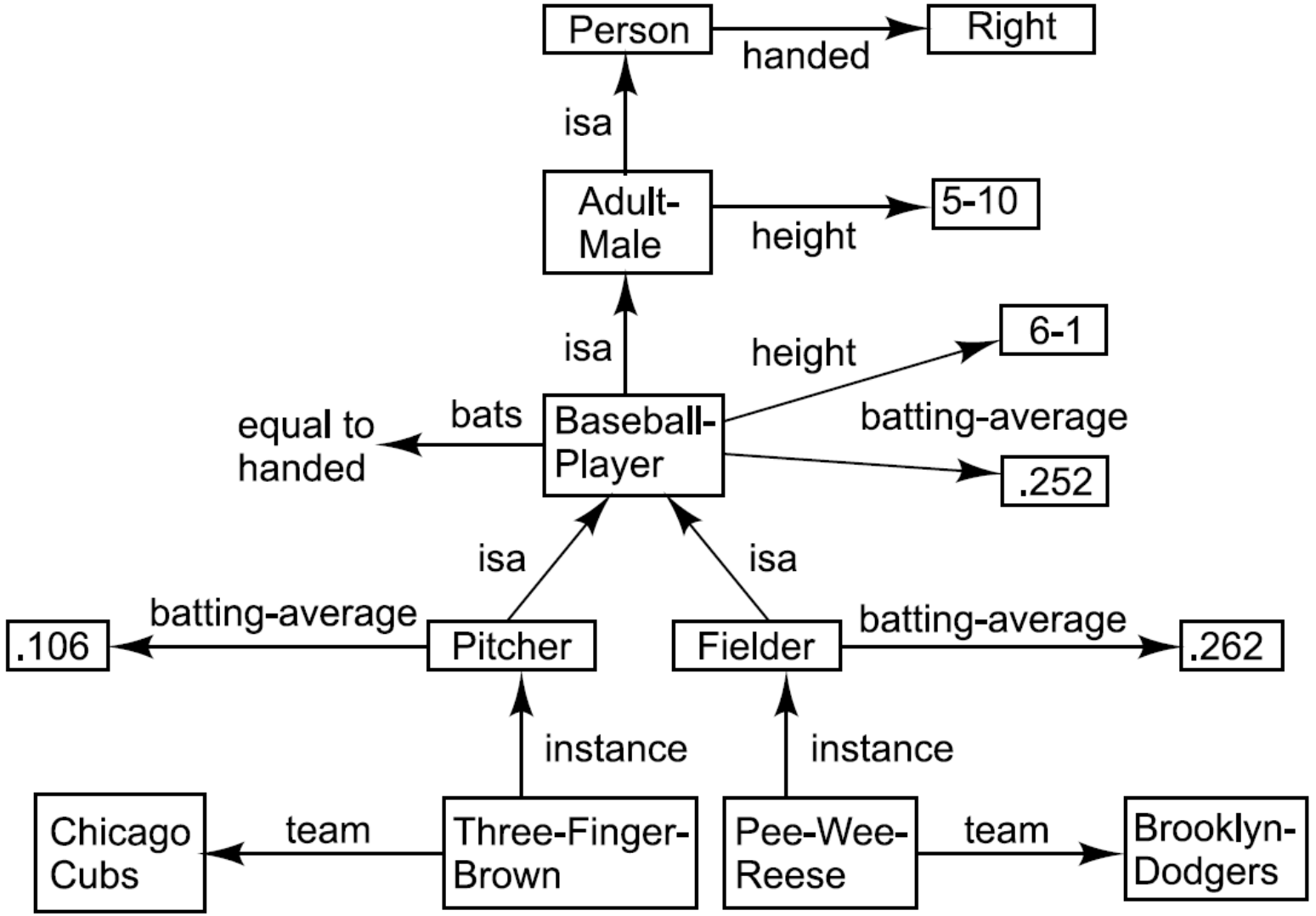
Frames:

•**Definition:** Frames are a knowledge representation technique that is similar to semantic networks but more structured. They consist of slots (attributes or properties) associated with a frame (an object or concept). These slots contain information or values related to the frame.

•**Example:** Continuing with the animal domain, you could define a "Dog" frame with slots for "Breed," "Color," and "Average Lifespan." Each slot would contain specific values.

Frame: Dog
- Breed: Golden Retriever
- Color: Yellow
- Average Lifespan: 10-12 years

Frames


Person
handed
Right
isa
Adult-
Male
height
5-10
isa
height
6-1
equal to
handed
bats
Baseball-
Player
batting-average
.252
isa
isa
batting-average
.106
Pitcher
Fielder
batting-average
.262
instance
instance
Chicago
Cubs
team
Three-Finger-
Brown
Pee-Wee-
Reese
team
Brooklyn-
Dodgers

Frames

Person

isa :	*Mammal*
cardinality :	6,000,000,000
* *handed* :	*Right*

Adult-Male

isa :	*Person*
cardinality :	2,000,000,000
* *height* :	5-10

ML-Baseball-Player

isa :	*Adult-Male*
cardinality :	624
**height* :	6-1
* *bats* :	equal to handed
* *batting-average* :	.252
* *team* :	
* *uniform-color* :	

Frames

Fielder

isa :	*ML-Baseball-Player*
cardinality :	376
**batting-average* :	.262

Pee-Wee-Reese

instance :	*Fielder*
height :	5-10
bats :	*Right*
batting-average :	.309
team :	*Brooklyn-Dodgers*
uniform-color :	*Blue*

ML-Baseball-Team

isa:	*Team*
cardinality :	26
* *team-size* :	24
* *manager* :	

Frames

Brooklyn-Dodgers

instance :	*ML-Baseball-Team*
team-size :	24
manager :	*Leo-Durocher*
players :	{*Pee-Wee-Reese*,...}

Inheritance

Inheritance:

•**Definition:** Inheritance is a fundamental concept in knowledge representation. It allows for the propagation of properties or attributes from a parent object to its child objects in a hierarchical structure. This means that if an attribute is defined at a higher level in the hierarchy, it is automatically inherited by the objects lower in the hierarchy.

•**Example:** In the animal domain, if you define an attribute "Legs" at the level of "Animal," all specific types of animals like "Dog," "Cat," and "Bird" inherit this property.

Animal -has-> Legs
Dog
Cat
Bird

In this example, all animals inherit the "Legs" attribute from the "Animal" frame.

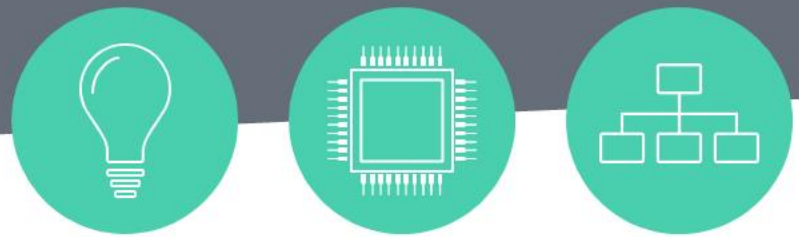
Artificial Intelligence

By

Dr. Manoj Kumar

University School of Automation and Robotics

GGSIP University, East Campus, Delhi, India



What is Probability

- **Probability is the branch of mathematics that studies the possible outcomes of the given events together with the outcome's relative likelihoods and distributions.**

- **In common usage, the word "probability" is used to mean the chance that a particular event (or set of events) will occur expressed on a linear scale from 0 (impossibility) to 1 (certainty), also expressed as a percentage between 0 and 100%.**

Uncertainty in AI

- **When an agent knows enough facts about its environment, the logical approach enables it to derive plans that are guaranteed to work.**
- **This is a good thing.**
- **Unfortunately, *agents almost never have access to the whole truth about their environment.***
- **Agents must, therefore, act under uncertainty.**
- **For a logical agent, it might be impossible to construct a complete and correct description of how its actions will work.**
- Suppose, for example, that the agent wants to drive someone to the airport to catch a flight and is considering a plan, A_{90}, that involves leaving home 90 minutes before the flight departs and driving at a reasonable speed. Even though the airport is only about 15 miles away, the agent will not be able to conclude with certainty that "Plan A_{90} will get us to the airport in time." Instead, it reaches the weaker conclusion "Plan A_{90} will get us to the airport in time, as long as my car doesn't break down or run out of gas, and I don't get into an accident, and there are no accidents on the bridge, and the plane doesn't leave early, and . . . .“ None of these conditions can be deduced, so the plan's success cannot be inferred.

Uncertainty in AI

- Artificial Intelligence is about developing predictive models from uncertain data where uncertainty means working with imperfect or incomplete information.
- There are three main sources of uncertainty in artificial intelligence, they are: noisy data, incomplete coverage of the problem domain and imperfect models.

- In this case the agent must have some preferences, which is achieved by using utility theory.
- Preferences, as expressed by utilities, are combined with probabilities in the general theory of rational decisions called decision theory.

Decision theory = probability theory + utility theory

Note: The word "utility" here refers to "the quality of being useful," not to the electric company or waterworks

- The utility function maps each state to a real number to check how efficiently each action achieves the goals.

Probability in AI

- So, to represent uncertain knowledge, where we are not sure about the predicates, we need uncertain reasoning or probabilistic reasoning.

ProbabilityOfRain(Today)

Causes of uncertainty in the real world:

1. Information occurred from unreliable sources.
2. Experimental Errors
3. Equipment fault
4. Temperature variation
5. Climate change.

- In Artificial Intelligence, probability is of two types:
  - **Prior or Unconditional Probability**
  - **Conditional Probability**

Probabilistic reasoning

- **Probabilistic reasoning is a way of knowledge representation where we apply the concept of probability to indicate the uncertainty in knowledge.**
- **In probabilistic reasoning, we combine probability theory with logic to handle the uncertainty.**
- **We use probability in probabilistic reasoning because it provides a way to handle the uncertainty that is the result of someone's laziness and ignorance.**
- **In the real world, there are lots of scenarios, where the certainty of something is not confirmed, such as "It will rain today," "behavior of someone for some situations," "A match between two teams or two players.“**
- **These are probable sentences for which we can assume that it will happen but not sure about it, so here we use probabilistic reasoning.**

Need of probabilistic reasoning in AI

- **When there are unpredictable outcomes.**
- **When specifications or possibilities of predicates becomes too large to handle.**
- **When an unknown error occurs during an experiment.**

Probability

- **Probability:** Probability can be defined as a chance that an uncertain event will occur. It is the numerical measure of the likelihood that an event will occur. The value of probability always remains between 0 and 1 that represent ideal uncertainties.

> $0 \le P(A) \le 1$, where P(A) is the probability of an event A.

> P(A) = 0, indicates total uncertainty in an event A.

> P(A) =1, indicates total certainty in an event A.

- We can find the probability of an uncertain event by using the below formula.

$$\text{Probabilty} = \frac{\text{Favorable outcomes}}{\text{Total outcomes}}$$

$$\textbf{Probability of occurrence} = \frac{\text{Number of desired outcomes}}{\text{Total number of outcomes}}$$

Example:

$P(red) = \frac{7}{12}$ ← Number of red marbles; ← Total number of marbles (sample space)

$P(blue) = \frac{5}{12}$ ← Number of blue marbles; ← Total number of marbles (sample space)

Probability

Event: Each possible outcome of a variable is called an event.

Sample space: The collection of all possible events is called sample space.

Random variables: Random variables are used to represent the events and objects in the real world.

Prior probability: The prior probability of an event is probability computed before observing new information.

Posterior Probability: The probability that is calculated after all evidence or information has taken into account. It is a combination of prior probability and new information.

Unconditional Probability Example 1

1) There are 6 pillows in a bed, 3 are red, 2 are yellow and 1 is blue. What is the probability of picking a yellow pillow?

Ans: The probability is equal to the number of yellow pillows in the bed divided by the total number of pillows, i.e. 2/6 = 1/3.

Unconditional Probability Example 1

1) There are 6 pillows in a bed, 3 are red, 2 are yellow and 1 is blue. What is the probability of picking a yellow pillow?

Ans: The probability is equal to the number of yellow pillows in the bed divided by the total number of pillows, i.e. 2/6 = 1/3.

Example: the chances of rolling a "4" with a die

Number of ways it can happen: 1 (there is only 1 face with a "4" on it)

Total number of outcomes: 6 (there are 6 faces altogether)

$$\text{So the probability} = \frac{1}{6}$$

Example: there are 5 marbles in a bag: 4 are blue, and 1 is red. What is the probability that a blue marble gets picked?

Number of ways it can happen: 4 (there are 4 blues)

Total number of outcomes: 5 (there are 5 marbles in total)

$$\text{So the probability} = \frac{4}{5} = 0.8$$

Unconditional Probability Example 3

2) There is a container full of colored bottles, red, blue, green and orange. Some of the bottles are picked out and displaced. Sumit did this 1000 times and got the following results:

- No. of blue bottles picked out: 300
- No. of red bottles: 200
- No. of green bottles: 450
- No. of orange bottles: 50

a) What is the probability that Sumit will pick a green bottle?

b) If there are 100 bottles in the container, how many of them are likely to be green?

Unconditional Probability Example 3

2) There is a container full of colored bottles, red, blue, green and orange. Some of the bottles are picked out and displaced. Sumit did this 1000 times and got the following results:

- No. of blue bottles picked out: 300
- No. of red bottles: 200
- No. of green bottles: 450
- No. of orange bottles: 50

a) What is the probability that Sumit will pick a green bottle?

Ans: For every 1000 bottles picked out, 450 are green.

Therefore, P(green) = 450/1000 = 0.45

b) If there are 100 bottles in the container, how many of them are likely to be green?

Ans: The experiment implies that 450 out of 1000 bottles are green.

Therefore, out of 100 bottles, 45 are green.

Unconditional Probability Example 4

Question 1: Find the probability of ‘getting 3 on rolling a die’.

Unconditional Probability Example 4

Question 1: Find the probability of 'getting 3 on rolling a die'.

Solution:

Sample Space = S = {1, 2, 3, 4, 5, 6}

Total number of outcomes = n(S) = 6

Let A be the event of getting 3.

Number of favorable outcomes = n(A) = 1

i.e., A = {3}

Probability, P(A) = n(A)/n(S) = 1/6

Hence, P(getting 3 on rolling a die) = 1/6

Unconditional Probability Example 5

Q. What is the probability of rolling an even number (2, 4, or 6) on a fair six-sided die?

Unconditional Probability Example 5

Q. What is the probability of rolling an even number (2, 4, or 6) on a fair six-sided die?

Solution: A fair six-sided die has six equally likely outcomes (numbers 1 through 6). The even numbers on the die are 2, 4, and 6. The probability of rolling an even number is:
Probability(Even Number) = (Number of favorable outcomes) / (Total number of possible outcomes)
Probability(Even Number) = 3 (favorable outcomes - 2, 4, and 6) / 6 (total possible outcomes on the die)
So, the probability of rolling an even number on a fair six-sided die is 3/6, which simplifies to 1/2

Unconditional Probability Example 5

What is the probability of drawing a red card (heart or diamond) from a standard deck of 52 playing cards?

Unconditional Probability Example 5

Solution: In a standard deck of 52 playing cards, there are two red suits: hearts and diamonds, each with 26 cards. The probability of drawing a red card is:
Probability(Red Card) = (Number of favorable outcomes) / (Total number of possible outcomes) Probability(Red Card) = 26 (favorable outcomes, as there are 26 red cards) / 52 (total number of possible outcomes in a deck of 52 cards)
So, the probability of drawing a red card from a standard deck of cards is 26/52, which simplifies to 1/2

Unconditional Probability Example 5

In a movie recommendation system, what is the unconditional probability that a user will watch a science fiction movie?

If the total movies 1,000 movie selections, 200 of them were science fiction movies, the unconditional probability of a user watching a science fiction movie is?

Types of Unconditional Probability

Theoretical Probability

It is based on the possible chances of something to happen. The theoretical probability is mainly based on the reasoning behind probability. For example, if a coin is tossed, the theoretical probability of getting a head will be ½.

Experimental Probability

It is based on the basis of the observations of an experiment. The experimental probability can be calculated based on the number of possible outcomes by the total number of trials. For example, if a coin is tossed 10 times and heads is recorded 6 times then, the experimental probability for heads is 6/10 or, 3/5.

Axiomatic Probability

In axiomatic probability, a set of rules or axioms are set which applies to all types. These axioms are set by Kolmogorov and are known as Kolmogorov's three axioms. With the axiomatic approach to probability, the chances of occurrence or non-occurrence of the events can be quantified. The axiomatic probability lesson covers this concept in detail with Kolmogorov's three rules (axioms) along with various examples.

Conditional Probability is the likelihood of an event or outcome occurring based on the occurrence of a previous event or outcome.

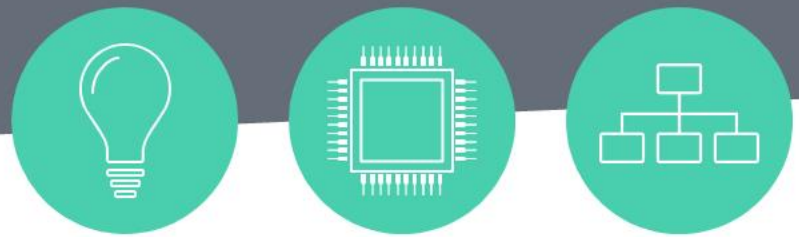
Artificial Intelligence

By

Dr. Manoj Kumar

University School of Automation and Robotics

GGSIP University, East Campus, Delhi, India



What is Conditional Probability

Probability is the branch of mathematics that studies the possible outcomes of the given events together with the outcome's relative likelihoods and distributions.

- **Conditional probability is a probability of occurring an event when another event has already happened.**

- Once the agent has obtained some evidence concerning the previously unknown random variables making up the domain, **prior probabilities are no longer applicable**.
- Instead, we use **conditional** or **posterior** probabilities.

What is Conditional Probability

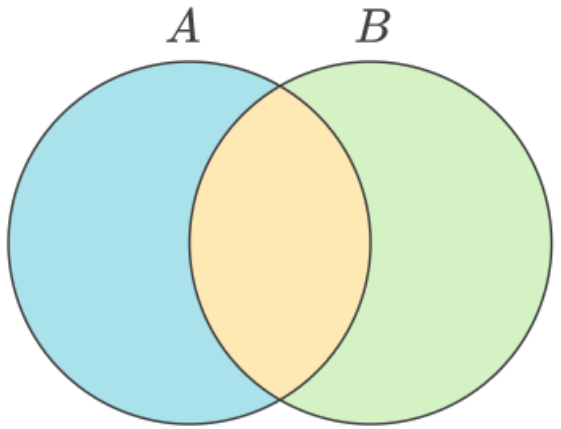
- Imagine you're predicting whether it will rain tomorrow, and your best guess (prior probability) is that there's a 50% chance of rain.
- But then, you check the weather forecast, and it says there's a storm coming, which increases the chance of rain.

- So, once you get new information (the weather forecast), your old guess (prior probability) might not be as accurate, and you might need to update it with the new information to get a better prediction.
- This is known as "updating the probabilities" based on the evidence or new knowledge you've gathered.

Conditional Probability

- Let's suppose, we want to calculate the event A when event B has already occurred, "the probability of A under the conditions of B", it can be written as:

- **So, what does this formula says?**



Conditional Probability Formula

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

Probability that A occurs given that B has already occured

Similarly: $$P(B|A) = \frac{P(A \text{ and } B)}{P(A)}$$

- It's NOT just both events happened, it's asking the probability of one event **AFTER** another event happened.
- It's based on a **happened** event, that's why you're to divide the probability of the happened event.

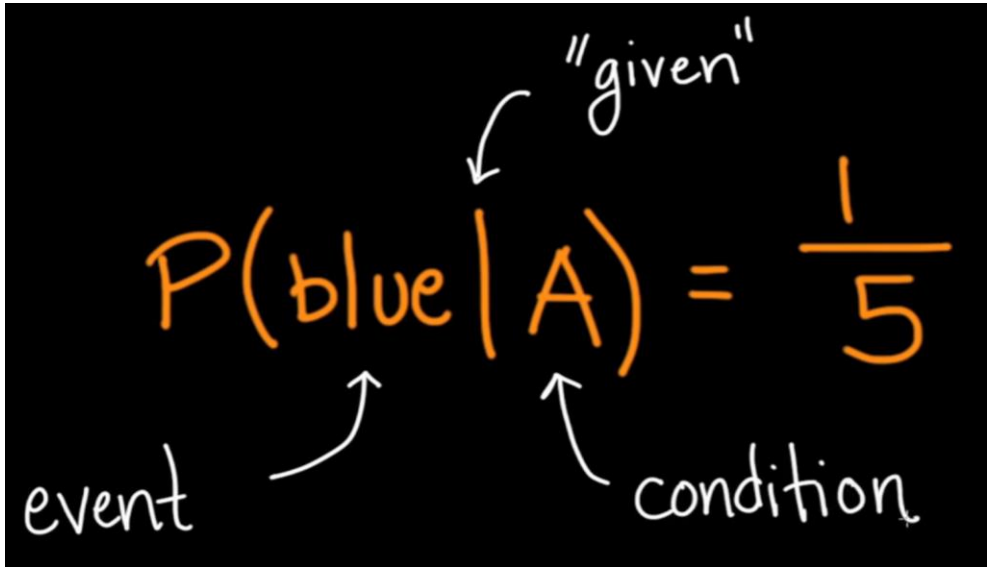
Conditional Probability

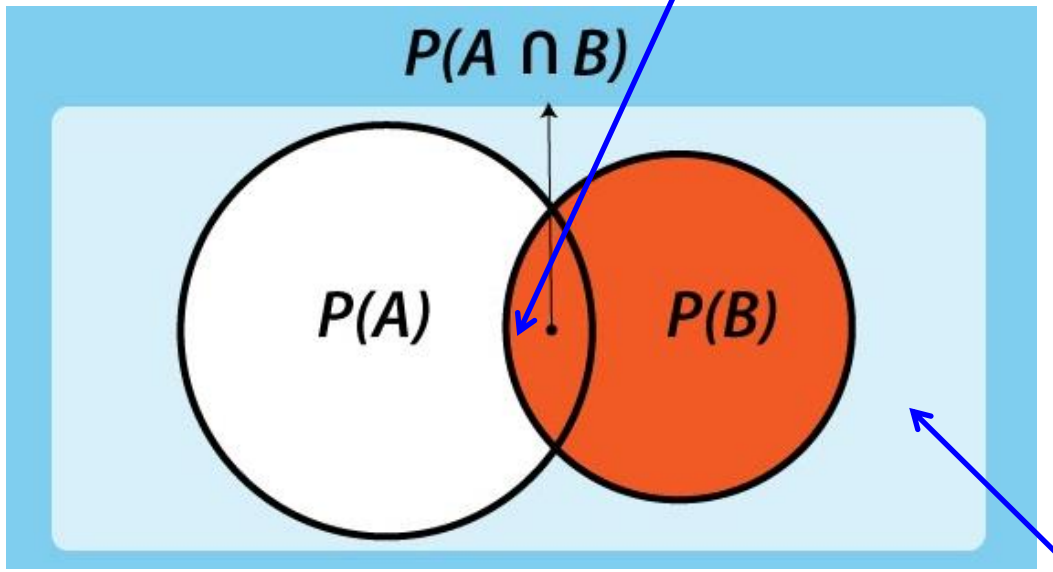
Probability of event A occured and event B occured

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

Probability of event A given B has occured

Probability of event B

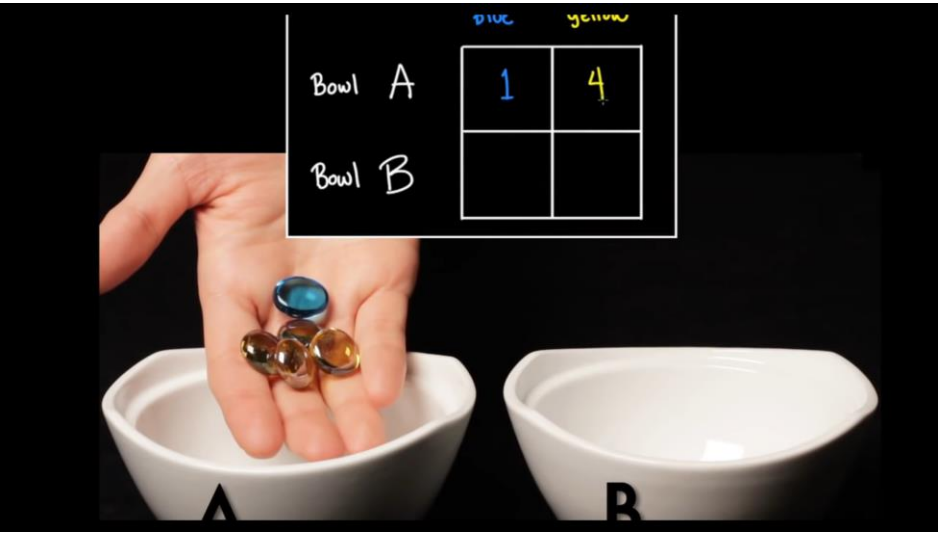


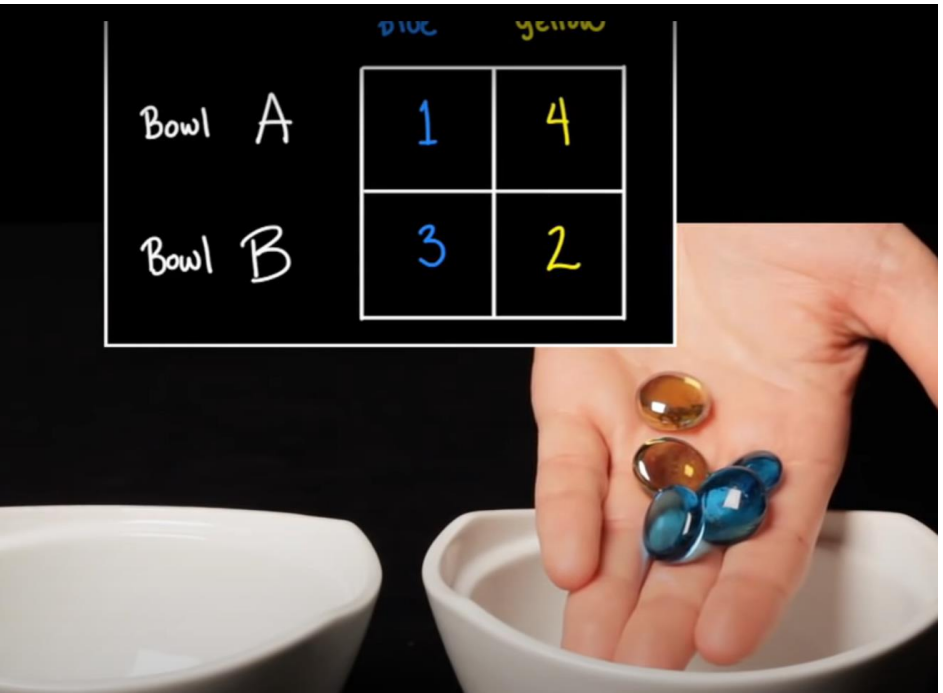


"and"

Sample space

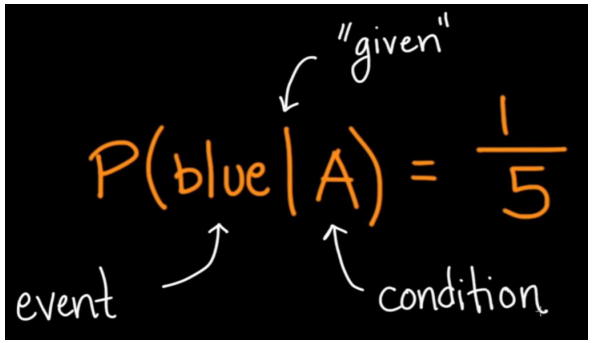
Conditional Probability



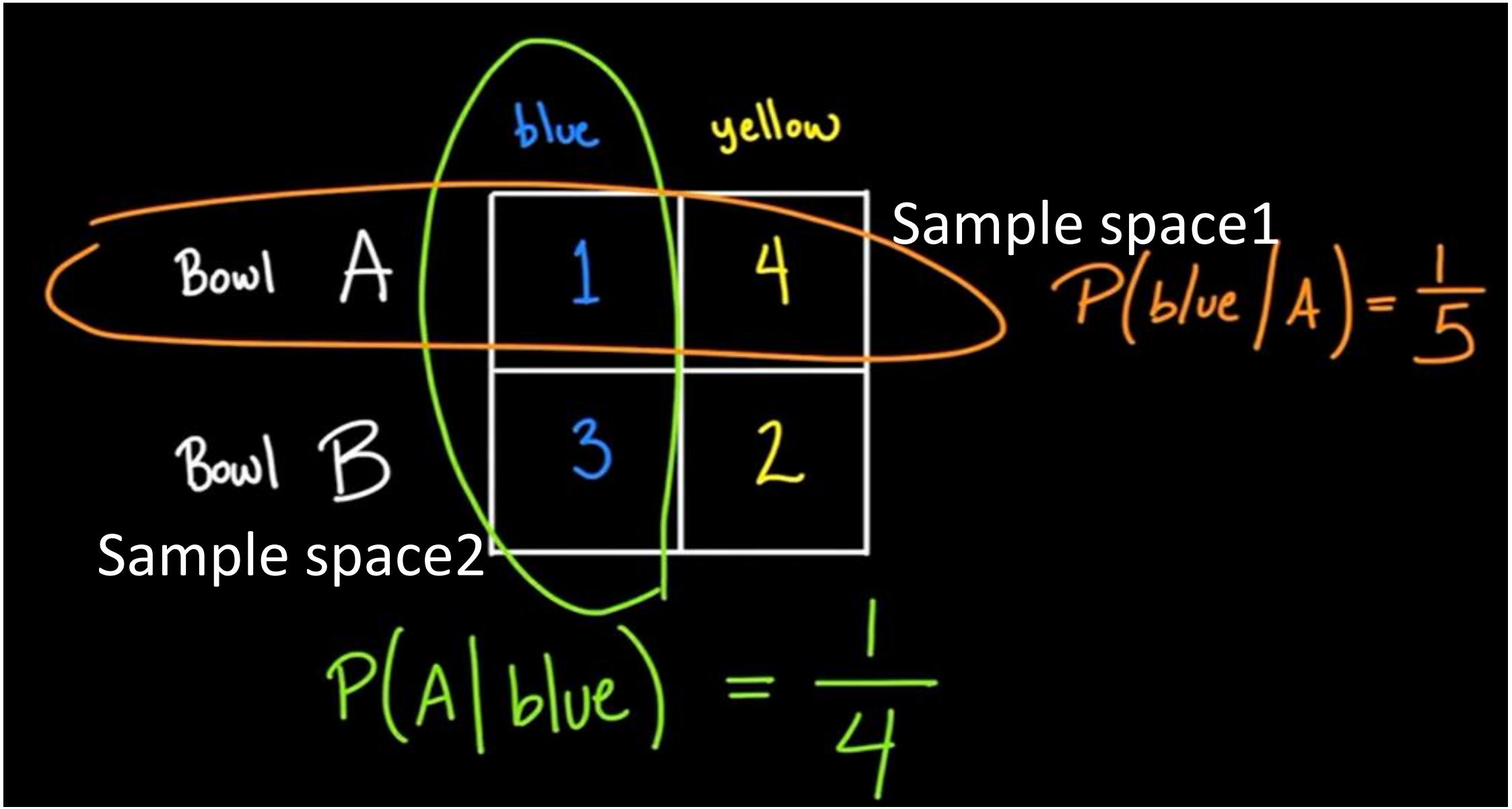


	blue	yellow	Total
Bowl A	1	4	5
Bowl B	3	2	5
	4	6	10

$$P(\text{blue}) = \frac{4}{10} \qquad P(\text{yellow}) = \frac{6}{10}$$



Conditional Probability

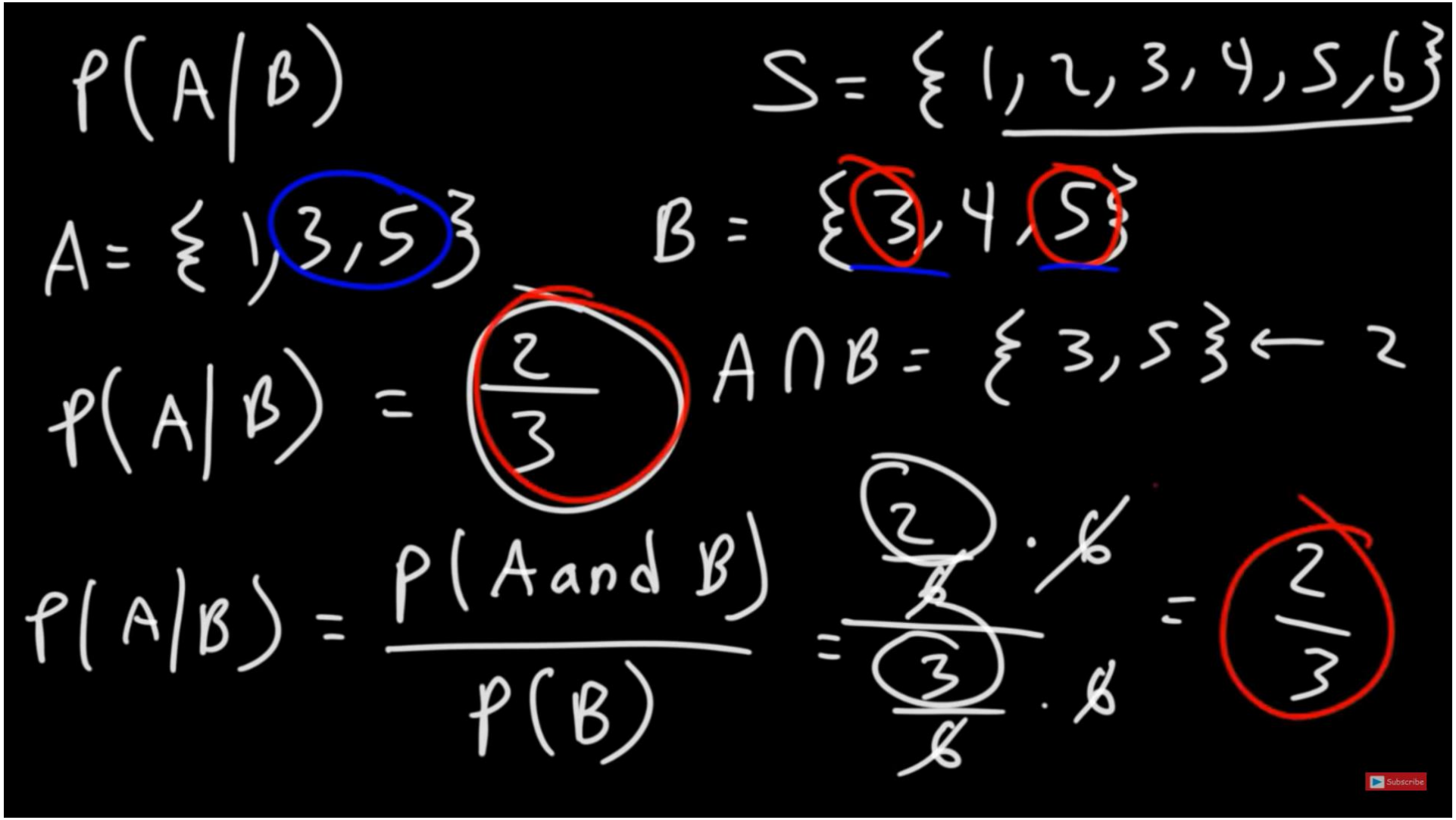


Conditional Probability

You toss a fair coin twice. What is the probability that the coin will land on heads both times?

B) If I already know that your coin toss resulted in heads, what would be the probability that both coins would land on heads?

Conditional Probability



Conditional Probability Example 1

Example:
In a class, there are 70% of the students who like English and 40% of the students who likes English and mathematics, and then what is the percent of students those who like English also like mathematics?

Solution:
Let, A is an event that a student likes Mathematics
B is an event that a student likes English.

$$P(A|B) = \frac{P(A \wedge B)}{P(B)} = \frac{0.4}{0.7} = 57\%$$

Hence, 57% are the students who like English also like Mathematics.

Conditional Probability Example 2

Example 2

A coin is tossed three times, sample space, S= {HHH, HHT, HTH, HTT, THH, THT, TTH, TTT}, i.e., 8 elements.

1.What is the probability of three heads?

Since from the sample space we can say that occurring 3 times head is once only, that is 1 element.
P(getting 3 heads)= 1/8.

2.If given that an event that shows the first toss was heads, then what is the probability of three heads.

Now, from sample space, let B is the event that shows the first toss is heads;
B= {HHH, HHT, HTH, HTT}, i.e, 4 elements,

A be the event of an occurrence of three heads
A={HHH}

Then (A∩B)= {HHH}, i.e, 1 element.
Then the P(getting 3 heads given that first toss is heads), or
P(A|B) = P(A∩B)/B
= 1/4.

Conditional Probability Example 3

A die is rolled twice, and two numbers are obtained, let X be the outcome of first role and Y be the outcome of the second roll. Given that X+Y=5, what is the probability of X=4 or Y=4?

Assume, A be the event the getting 4 as X or Y, and B be the event of X+Y=5, therefore
A={(4,1), (4,2), (4, 3), (4,4), (4,5), (4,6), (1,4), (2,4), (3,4), (4,4), (5,4), (6,4)}
B={ (1,4),(4,1), (2, 3), (3,2)}

We are interested in finding the probability of A given B
A∩ B= {(1,4), (4, 1)}

As die is rolled out two times, total sample space= 36
P(A∩ B)= 2/ 36
P(B)= 4/ 36
So, P(A|B)= P(A∩ b)/ P(B)
= 2/4, or
= 1/2.

	1	2	3	4	5	6
1	(1, 1)	(1, 2)	(1, 3)	(1, 4)	(1, 5)	(1, 6)
2	(2, 1)	(2, 2)	(2, 3)	(2, 4)	(2, 5)	(2, 6)
3	(3, 1)	(3, 2)	(3, 3)	(3, 4)	(3, 5)	(3, 6)
4	(4, 1)	(4, 2)	(4, 3)	(4, 4)	(4, 5)	(4, 6)
5	(5, 1)	(5, 2)	(5, 3)	(5, 4)	(5, 5)	(5, 6)
6	(6, 1)	(6, 2)	(6, 3)	(6, 4)	(6, 5)	(6, 6)

Bayes Rule

- Bayes' theorem is also known as **Bayes' rule, Bayes' law**, or **Bayesian reasoning**, which determines the probability of an event with uncertain knowledge.

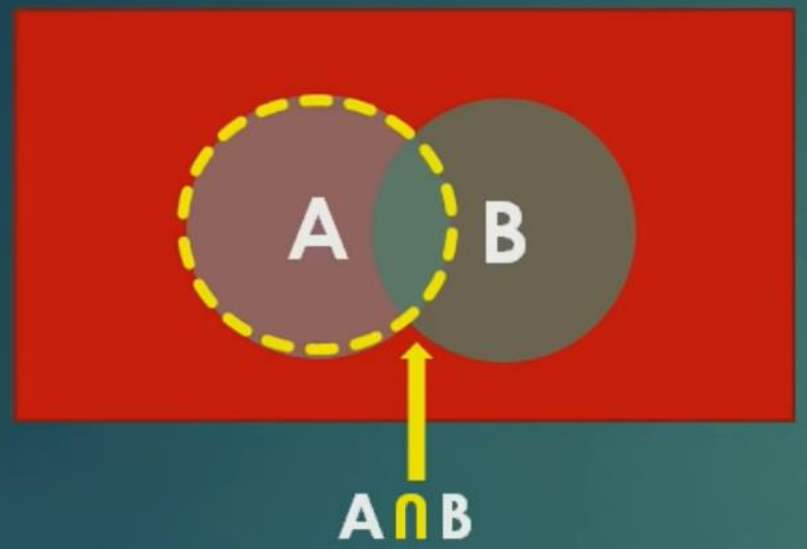
- In probability theory, it relates the conditional probability and marginal probabilities of two random events.

- 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.

- It is a way to calculate the value of P(B|A) with the knowledge of P(A|B).

- **Bayes' theorem allows updating the probability prediction of an event by observing new information of the real world.**

Bayes Rule



$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

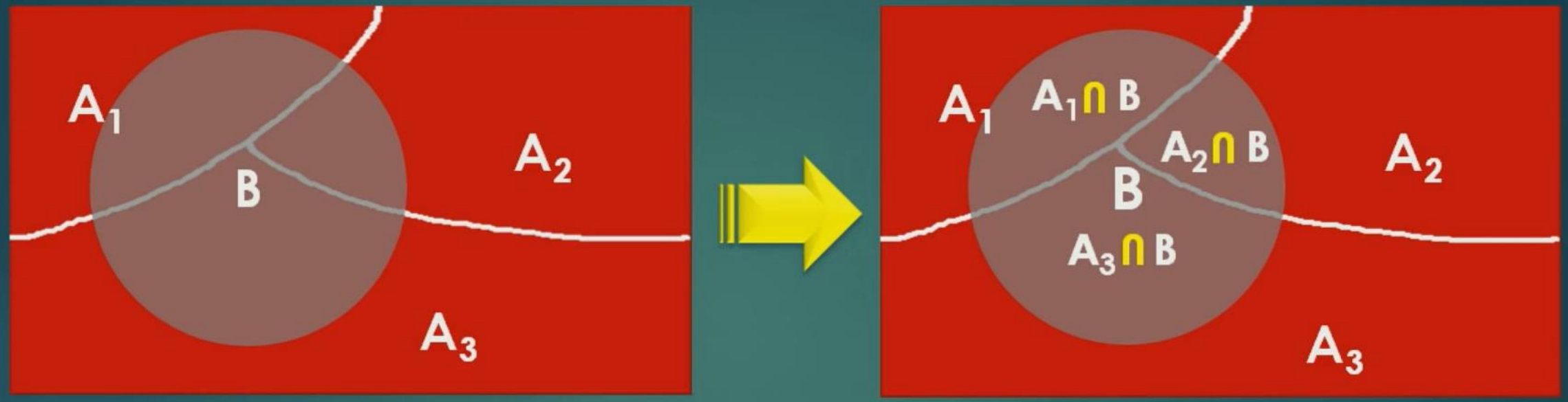
$$P(B \mid A) = \frac{P(A \cap B)}{P(A)}$$

$$P(A \cap B) = P(A \mid B) * P(B) = P(B \mid A) * P(A) \quad \text{---(i)}$$

$$P(A \mid B) * P(B) = P(B \mid A) * P(A)$$

$$P(A \mid B) = \frac{P(B \mid A) * P(A)}{P(B)} \quad \text{---(ii)}$$

Bayes Rule



$$P(B) = P(A_1 \cap B) + P(A_2 \cap B) + P(A_3 \cap B)$$

$$P(B) = P(B \mid A_1)*P(A_1) + P(B \mid A_2)*P(A_2) + P(B \mid A_3)*P(A_3)$$

$$P(B) = \sum_{i=1}^{n} P(B \mid A_i)*P(A_i) \quad \text{---(iii)}$$

Bayes Rule

$$P(A_i \mid B) = \frac{P(B \mid A_i) * P(A_i)}{P(B)} \quad \text{----(ii)}$$

$$P(B) = \sum_{i=1}^{n} P(B \mid A_i) * P(A_i) \quad \text{----(iii)}$$

$$P(A_i \mid B) = \frac{P(B \mid A_i) * P(A_i)}{\sum_{i=1}^{n} P(B \mid A_i) * P(A_i)}$$

Generalize form of Bayes Theorem

Bayes Rule

Example: If cancer corresponds to one's age then by using Bayes' theorem, we can determine the probability of cancer more accurately with the help of age.
Bayes' theorem can be derived using product rule and conditional probability of event A with known event B:
As from product rule we can write:

$$P(A \wedge B) = P(A|B)\, P(B)$$

Similarly, the probability of event B with known event A:

$$P(A \wedge B) = P(B|A)\, P(A)$$

Equating right hand side of both the equations, we will get:

$$P(A|B) = \frac{P(B|A)\, P(A)}{P(B)} \quad \text{....(a)}$$

The above equation (a) is called as **Bayes' rule** or **Bayes' theorem**. This equation is basic of most modern AI systems for **probabilistic inference**.

Applying Bayes Rule

- **Bayes' rule** allows us to compute the single term P(B|A) in terms of P(A|B), P(*B*), and P(A).
- This is very useful in cases where we have a good probability of these three terms and want to determine the fourth one.
- Suppose we want to perceive the effect of some unknown cause, and want to compute that cause, then the Bayes' rule becomes:

$$\mathbf{P(cause|effect)} = \frac{\mathrm{P(effect|cause)\,P(cause)}}{\mathrm{P(effect)}}$$

Bayes Rule: Example 1

Question: what is the probability that a patient has diseases meningitis with a stiff neck?
Given Data:
A doctor is aware that disease meningitis causes a patient to have a stiff neck, and it occurs 80% of the time. He is also aware of some more facts, which are given as follows:

- The Known probability that a patient has meningitis disease is 1/30,000.
- The Known probability that a patient has a stiff neck is 2%.

Let a be the proposition that patient has stiff neck and b be the proposition that patient has meningitis. , so we can calculate the following as:

P(a|b) = 0.8
P(b) = 1/30000
P(a)= .02

$$P(b|a) = \frac{P(a|b)P(b)}{P(a)} = \frac{0.8*(\frac{1}{30000})}{0.02} = 0.001333333.$$

Hence, we can assume that 1 patient out of 750 patients has meningitis disease with a stiff neck.

Bayes Rule: Example 2

Question: From a standard deck of playing cards, a single card is drawn. The probability that the card is king is 4/52, then calculate posterior probability P(King|Face), which means the drawn face card is a king card.

Solution:

$$\mathbf{P(king|face)} = \frac{P(Face|king)*P(King)}{P(Face)} \quad .....(i)$$

P(king): probability that the card is King= 4/52= 1/13
P(face): probability that a card is a face card= 3/13
P(Face|King): probability of face card when we assume it is a king = 1
Putting all values in equation (i) we will get:

$$P(king|face) = \frac{1*(\frac{1}{13})}{(\frac{3}{13})} = \mathbf{1/3,}\ \textbf{it is a probability that a face card is a king card.}$$

Bayes Theorem Applications

- **It is used to calculate the next step of the robot when the already executed step is given.**
- **Bayes' theorem is helpful in weather forecasting.**
- **It can solve the Monty Hall problem.**

Artificial Intelligence

By

Dr. Manoj Kumar

University School of Automation and Robotics

GGSIP University, East Campus, Delhi, India

Bayesian Belief Network

- Bayesian belief network is key computer technology for dealing with probabilistic events and to solve a problem that has uncertainty.
- **"A Bayesian network is a probabilistic graphical model which represents a set of variables and their conditional dependencies using a directed acyclic graph (DAG)."**
- It is also called a **Bayes network, belief network, decision network**, or **Bayesian model**.
- Bayesian networks are probabilistic, because these networks are built from a **probability distribution**, and also use probability theory for prediction and anomaly detection.

- Real world applications are probabilistic in nature, and to represent the relationship between multiple events, we need a Bayesian network.
- It can also be used in various tasks including **prediction, anomaly detection, diagnostics, automated insight, reasoning, time series prediction**, and **decision making under uncertainty**.

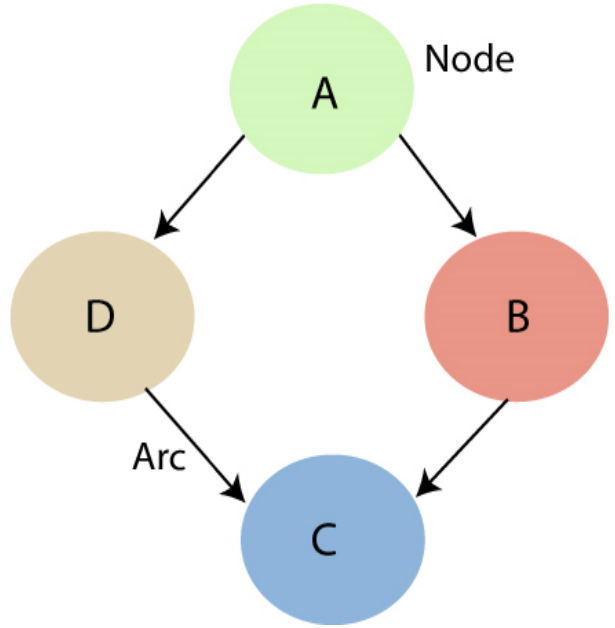
Bayesian Belief Network

- Bayesian Network can be used for building models from data and expert's opinion, and it consists of two parts:

•**Directed Acyclic Graph**

•**Table of conditional probabilities.**

The generalized form of Bayesian network that represents and solve decision problems under uncertain knowledge is known as an **Influence diagram**.



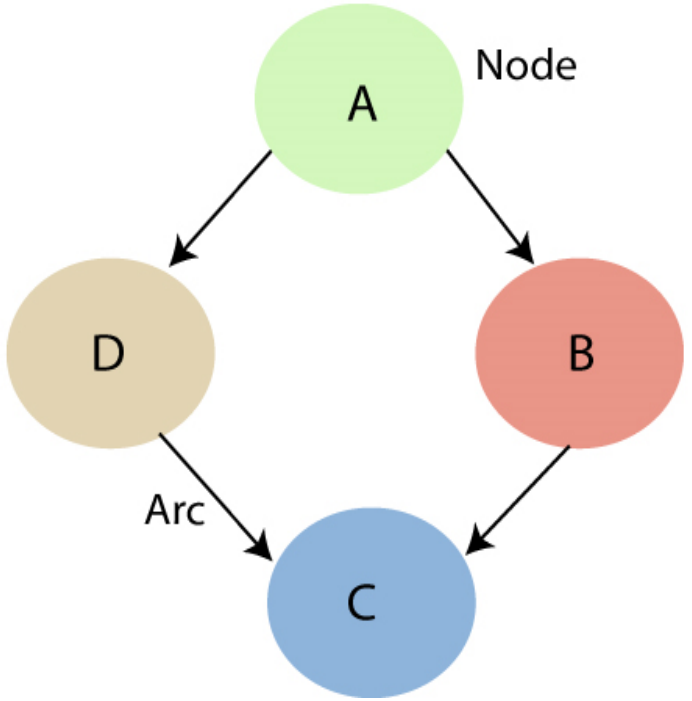
- **A Bayesian network graph is made up of nodes and Arcs (directed links).**

Bayesian Belief Network: DAG

•Each **node** corresponds to the random variables, and a variable can be **continuous** or **discrete**.

•**Arc or directed arrows** represent the causal relationship or conditional probabilities between random variables.

• These directed links or arrows connect the pair of nodes in the graph. These links represent that one node directly influence the other node, and if there is no directed link that means that nodes are independent with each other



- **In the above diagram, A, B, C, and D are random variables represented by the nodes of the network graph.**
- **If we are considering node B, which is connected with node A by a directed arrow, then node A is called the parent of Node B.**
- **Node C is independent of node A.**

Bayesian Belief Network

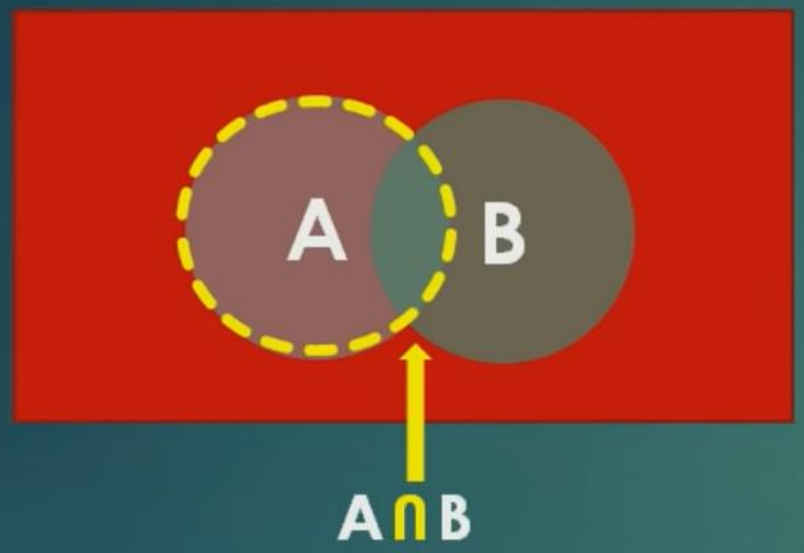
The Bayesian network has mainly two components:

- **Causal Component**
- **Actual numbers**

•Casual Component describes the structure of the domain in terms of dependencies between variables, and then the second part is the actual numbers, the quantitative part.

- Each node in the Bayesian network has condition probability distribution **$P(X_i \mid Parent(X_i))$**, which determines **the effect of the parent on that node.**
- Bayesian network is based on Joint probability distribution and conditional probability.

Conditional Probability: Bayes Rule



$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

$$P(B \mid A) = \frac{P(A \cap B)}{P(A)}$$

$$P(A \cap B) = P(A \mid B) * P(B) = P(B \mid A) * P(A) \quad \text{---(i)}$$

$$P(A \mid B) * P(B) = P(B \mid A) * P(A)$$

$$P(A \mid B) = \frac{P(B \mid A) * P(A)}{P(B)} \quad \text{---(ii)}$$

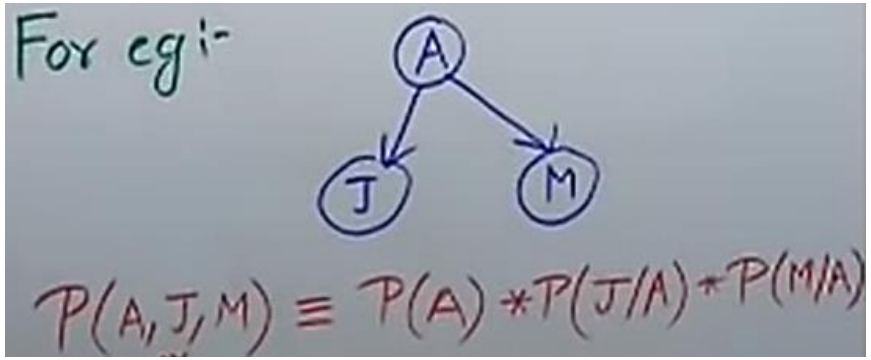
Joint Probability Distribution

So, let's first understand the joint probability distribution:

- **Joint probability is a measure of the likelihood that two or more events occur simultaneously.**

If we have variables x_1, x_2, x_3,....., x_n, then the probabilities of a different combination of x_1, x_2, x_3.. x_n, are known as Joint probability distribution.

$P[x_1, x_2, x_3,....., x_n]$, it can be written as the following way in terms of the joint probability distribution.



$= P[x_1| x_2, x_3,....., x_n]P[x_2, x_3,....., x_n]$

$= P[x_1| x_2, x_3,....., x_n]P[x_2|x_3,....., x_n]....P[x_{n-1}|x_n]P[x_n].$

In general, for each variable Xi, we can write the equation as:

$P(X_i|X_{i-1},....., X_1) = P(X_i |Parents(X_i))$

Joint Probability Distribution

- Joint probability should not be confused with conditional probability, which is the probability that one event will happen *given that* another action or event happens.

The conditional probability formula is as follows:

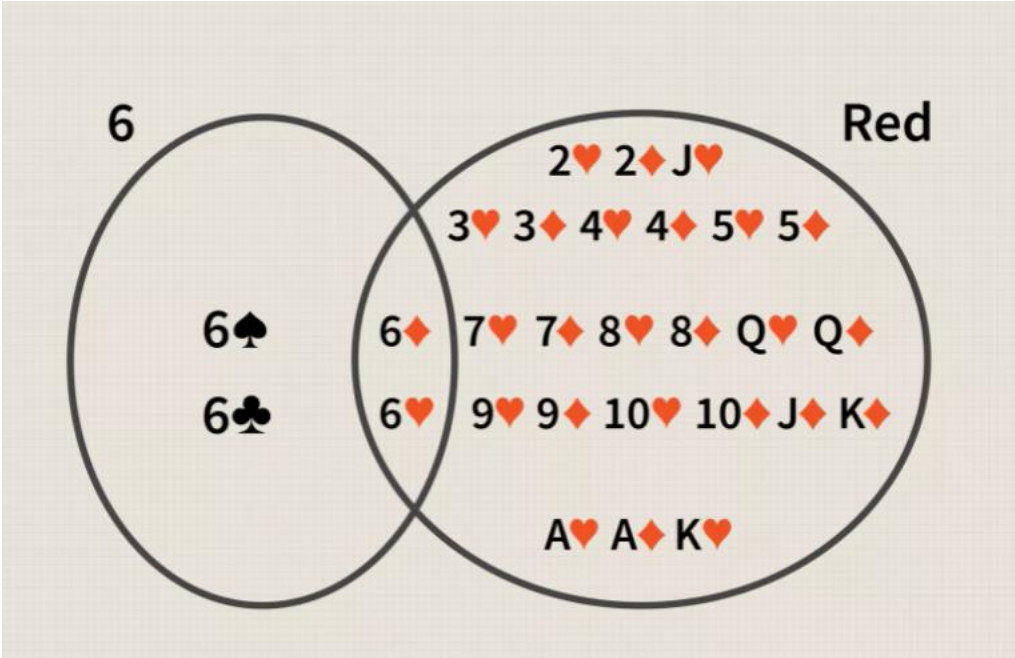
$$P(X, given\ Y) \text{ or } P(X|Y)$$

This is to say that the chance of one event happening is conditional on another event happening.
For example, from a deck of cards, the probability that you get a six, given that you drew a red card is P(6 | red) = 2/26 = 1/13, since there are two sixes out of 26 red cards.
Joint probability only factors the likelihood of both events occurring. Conditional probability can be used to calculate joint probability, as seen in this formula:

$$P(X \cap Y) = P(X|Y) \times P(Y)$$

The probability that A and B occurs is the probability of X occurring, given that Y occurs multiplied by the probability that Y occurs. Given this formula, the probability of drawing a 6 and a red at the same time will be as follows:

$$P(6 \cap red) = P(6|red) \times P(red) =$$
$$1/13 \times 26/52 = 1/13 \times 1/2 = 1/26$$

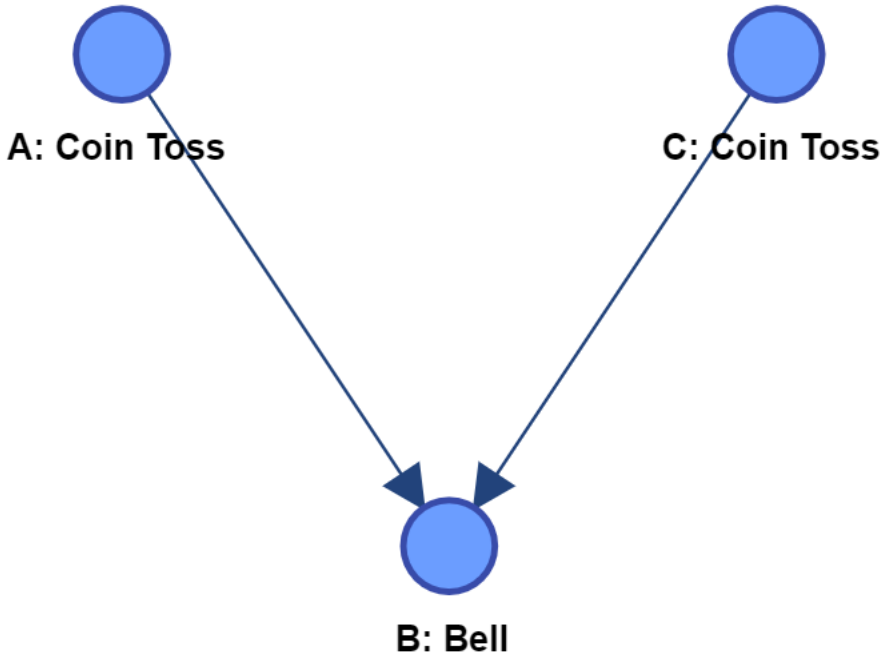


Bayesian Belief Network: Problem 1

Heads	Tails
50.000	50.000

Heads	Tails
50.000	50.000

- A and B are dependent, B and C are dependent, yet A and C are independent.
- If you asked a person to supply an example of three such events, the example would invariably portray A and C as two independent causes and B as their common effect, namely A → B ← C.
- For instance, A and C could be the outcomes of two fair coins, and B represents a bell that rings whenever either coin comes up heads.



A: Coin Toss	C: Coin Toss	Rings	Does Not Ring
Heads	Heads	100.000	0.000
	Tails	100.000	0.000
Tails	Heads	100.000	0.000
	Tails	0.000	100.000

Bayesian Belief Network: Problem 2

Calculate the probability that alarm has sounded, but there is neither a burglary, nor an earthquake occurred, and David and Sophia both called the Harry.

Solution:

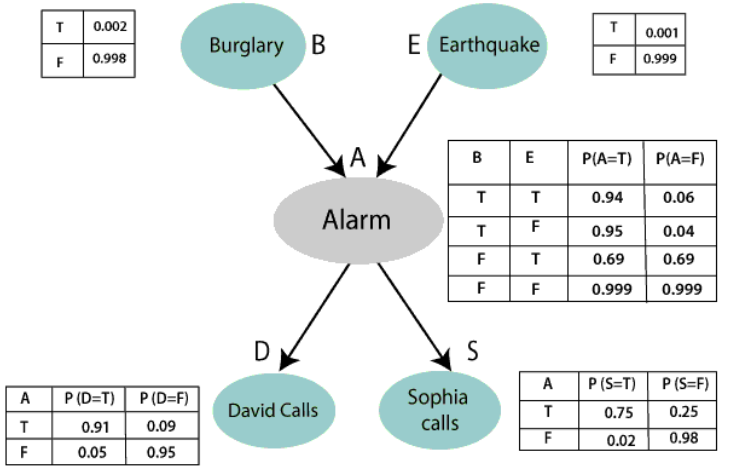
•The Bayesian network for the above problem is given below. The network structure is showing that burglary and earthquake is the parent node of the alarm and directly affecting the probability of alarm's going off, but David and Sophia's calls depend on alarm probability.

•The network is representing that our assumptions do not directly perceive the burglary and also do not notice the minor earthquake, and they also not confer before calling.

•The conditional distributions for each node are given as conditional probabilities table or CPT.

•Each row in the CPT must be sum to 1 because all the entries in the table represent an exhaustive set of cases for the variable.

•In CPT, a Boolean variable with k Boolean parents contains 2^K probabilities. Hence, if there are two parents, then CPT will contain 4 probability values



T	0.002
F	0.998

T	0.001
F	0.999

B	E	P(A=T)	P(A=F)
T	T	0.94	0.06
T	F	0.95	0.04
F	T	0.69	0.69
F	F	0.999	0.999

A	P (D=T)	P (D=F)
T	0.91	0.09
F	0.05	0.95

A	P (S=T)	P (S=F)
T	0.75	0.25
F	0.02	0.98

List of all events occurring in this network:

- **Burglary (B)**
- **Earthquake(E)**
- **Alarm(A)**
- **David Calls(D)**
- **Sophia calls(S)**

Bayesian Belief Network

We can write the events of problem statement in the form of probability: **P[D, S, A, B, E]**, can rewrite the above probability statement using joint probability distribution:

P[D, S, A, B, E]

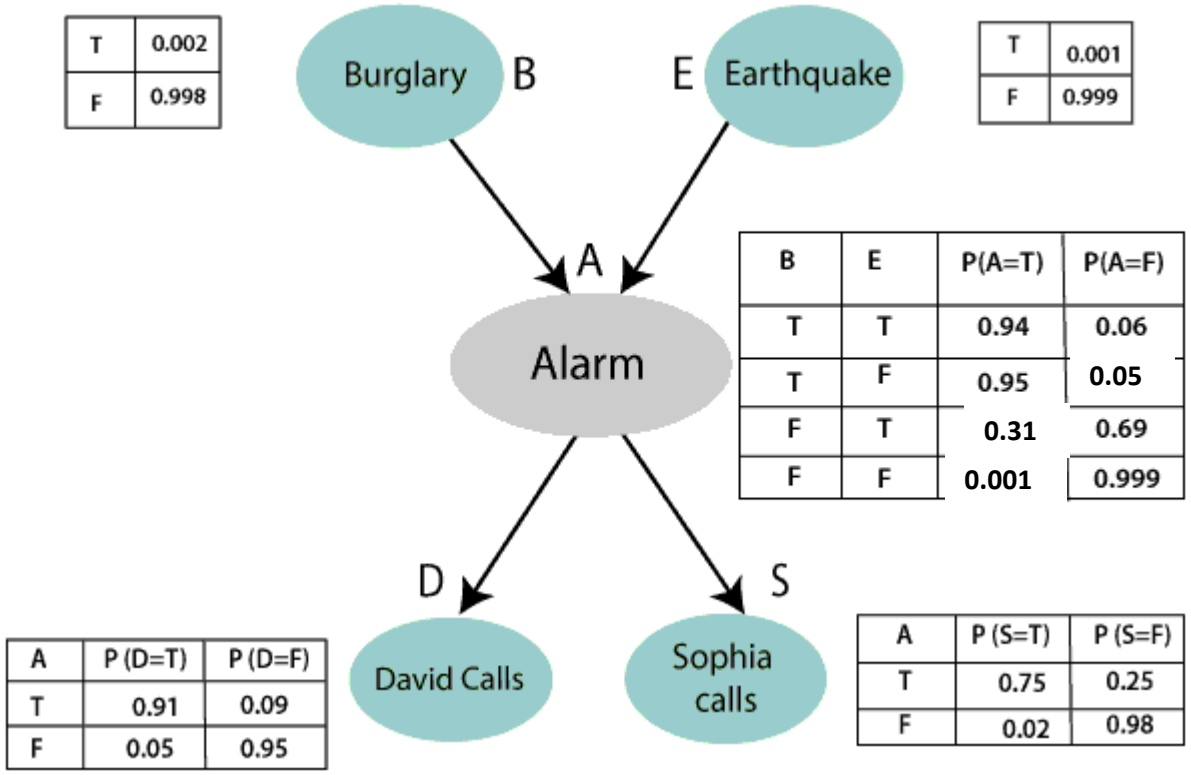
= P[D | S, A, B, E]. P[S, A, B, E]

= P[D | S, A, B, E]. P[S | A, B, E]. P[A, B, E]

= P[D| A]. P [S| A, B, E]. P[A, B, E]

= P[D | A]. P[S | A]. P[A| B, E]. P[B, E]

= P[D | A]. P[S | A]. P[A| B, E]. P[B |E]. P[E]



T	0.002
F	0.998

T	0.001
F	0.999

B	E	P(A=T)	P(A=F)
T	T	0.94	0.06
T	F	0.95	0.05
F	T	0.31	0.69
F	F	0.001	0.999

A	P (D=T)	P (D=F)
T	0.91	0.09
F	0.05	0.95

A	P (S=T)	P (S=F)
T	0.75	0.25
F	0.02	0.98

Bayesian Belief Network

Let's take the observed probability for the Burglary and earthquake component:

P(B= True) = 0.002, which is the probability of burglary.
P(B= False)= 0.998, which is the probability of no burglary.
P(E= True)= 0.001, which is the probability of a minor earthquake
P(E= False)= 0.999, Which is the probability that an earthquake not occurred.
We can provide the conditional probabilities as per the below tables:

Conditional probability table for Alarm A:
The Conditional probability of Alarm A depends on Burglar and earthquake:

B	E	P(A= True)	P(A= False)
True	True	0.94	0.06
True	False	0.95	0.04
False	True	0.31	0.69
False	False	0.001	0.999

Bayesian Belief Network

Conditional probability table for David Calls:

The Conditional probability of David that he will call depends on the probability of Alarm.

A	P(D= True)	P(D= False)
True	0.91	0.09
False	0.05	0.95

Conditional probability table for Sophia Calls:

The Conditional probability of Sophia that she calls is depending on its Parent Node "Alarm."

A	P(S= True)	P(S= False)
True	0.75	0.25
False	0.02	0.98

From the formula of joint distribution, we can write the problem statement in the form of probability distribution:

P(S, D, A, ¬B, ¬E) = P (S|A) *P (D|A)*P (A|¬B ^ ¬E) *P (¬B) *P (¬E).

= 0.75* 0.91* 0.001* 0.998*0.999

= 0.00068045.

Bayesian Belief Network

Hence, a Bayesian network can answer any query about the domain by using Joint distribution.

The semantics of Bayesian Network:

There are two ways to understand the semantics of the Bayesian network, which is given below:

1. To understand the network as the representation of the Joint probability distribution.

It is helpful to understand how to construct the network.

2. To understand the network as an encoding of a collection of conditional independence statements.

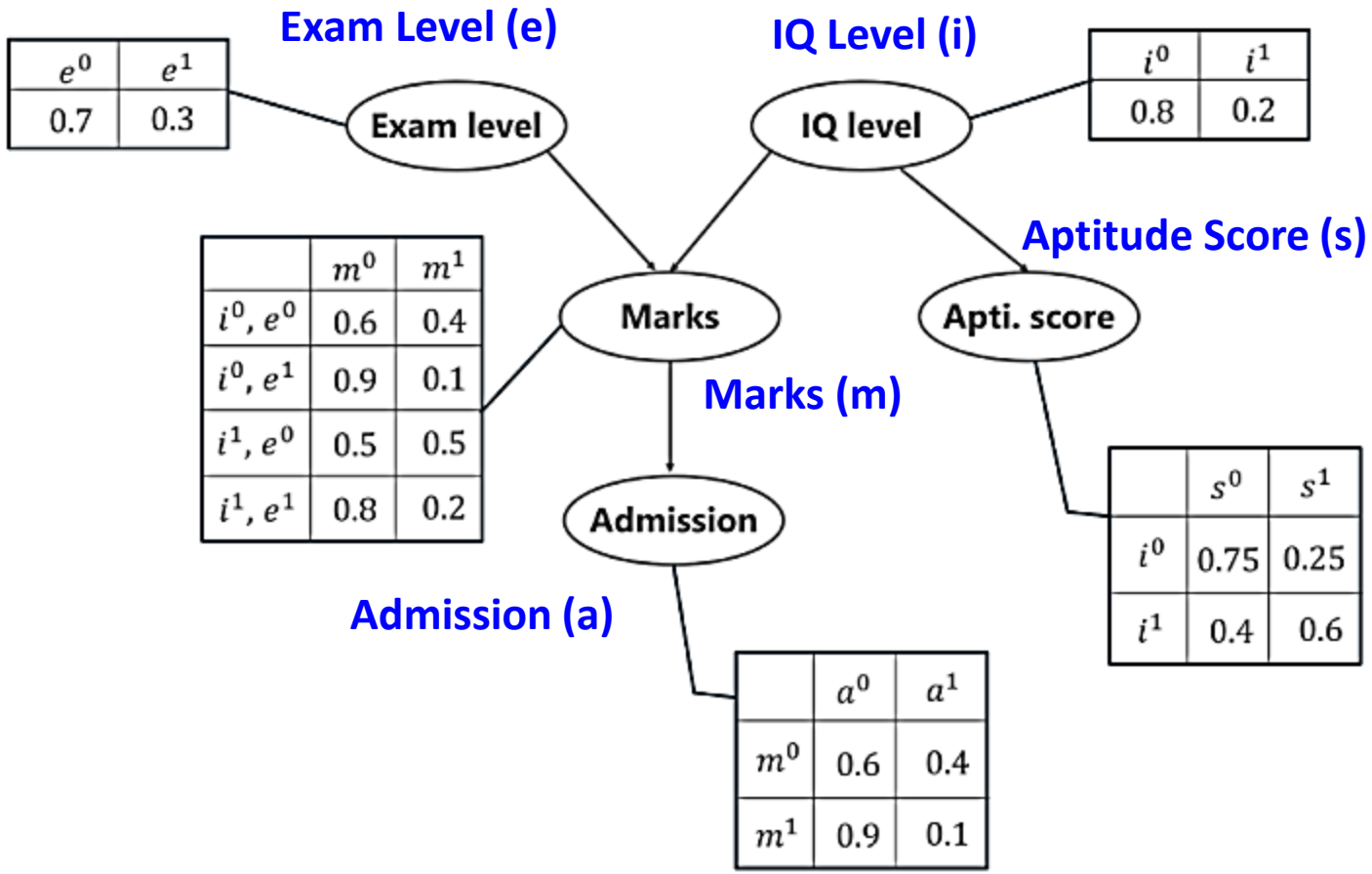
It is helpful in designing inference procedure.

Bayesian Belief Network : Problem 2

- Given the task of modeling a student's marks (***m***) for an exam he has just given.
- From the given Bayesian Network Graph below, we see that the marks depend upon two other variables.

•**Exam Level (*e*)**– This discrete variable denotes the difficulty of the exam and has two values (**0 for easy and 1 for difficult**)

•**IQ Level (*i*)** – This represents the Intelligence Quotient level of the student and is also discrete in nature having two values (**0 for low and 1 for high**)

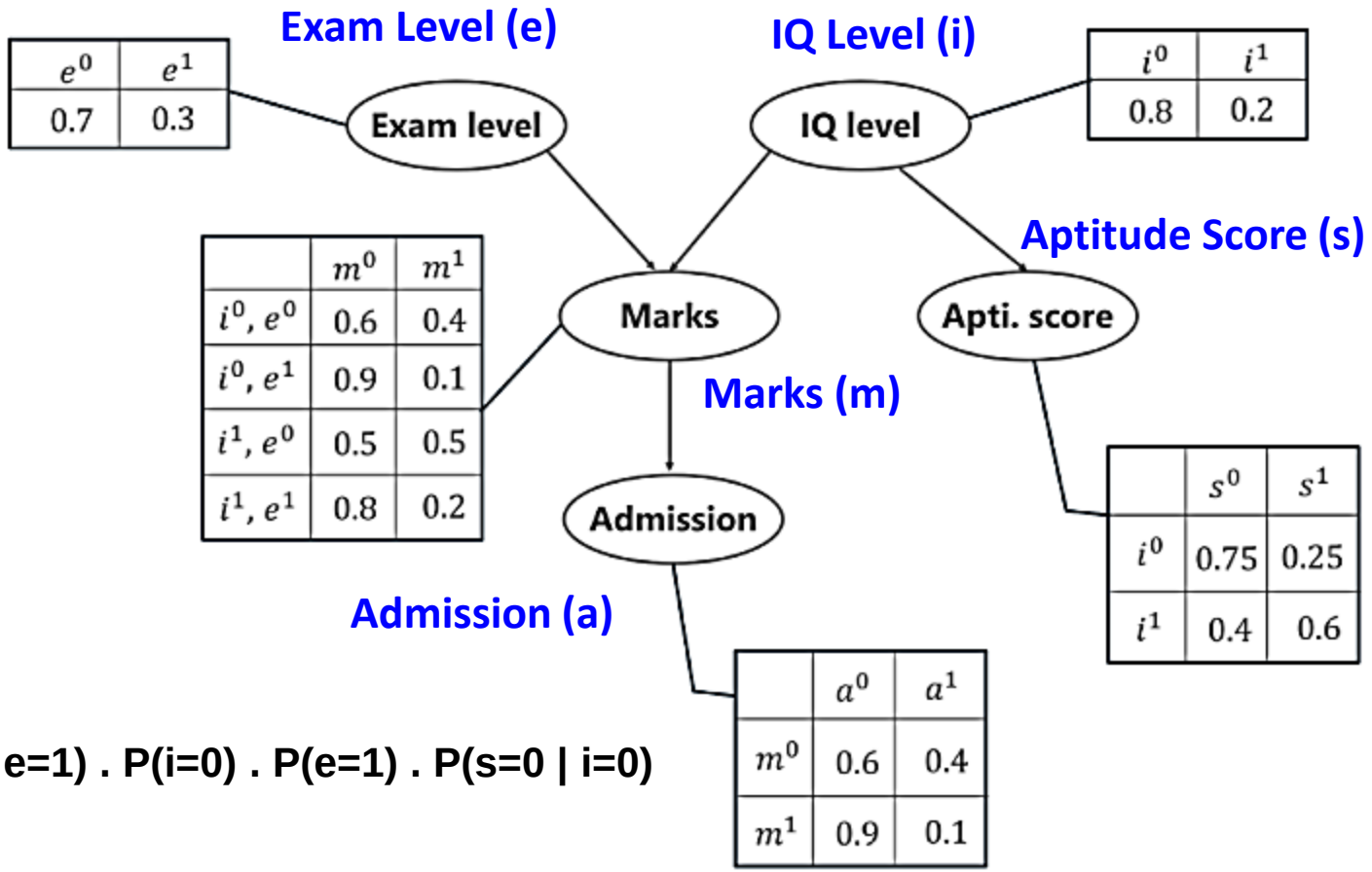


Bayesian Belief Network : Problem 2

Case 1: Calculate the probability that in spite of the exam level being difficult, the student having a low IQ level and a low Aptitude Score, manages to pass the exam and secure admission to the university.

- **Exam Level (*e*)**– This discrete variable denotes the difficulty of the exam and has two values (**0 for easy and 1 for difficult**)

- **IQ Level (*i*)** – This represents the Intelligence Quotient level of the student and is also discrete in nature having two values (**0 for low and 1 for high**)



P[a=1, m=1, i=0, e=1, s=0]

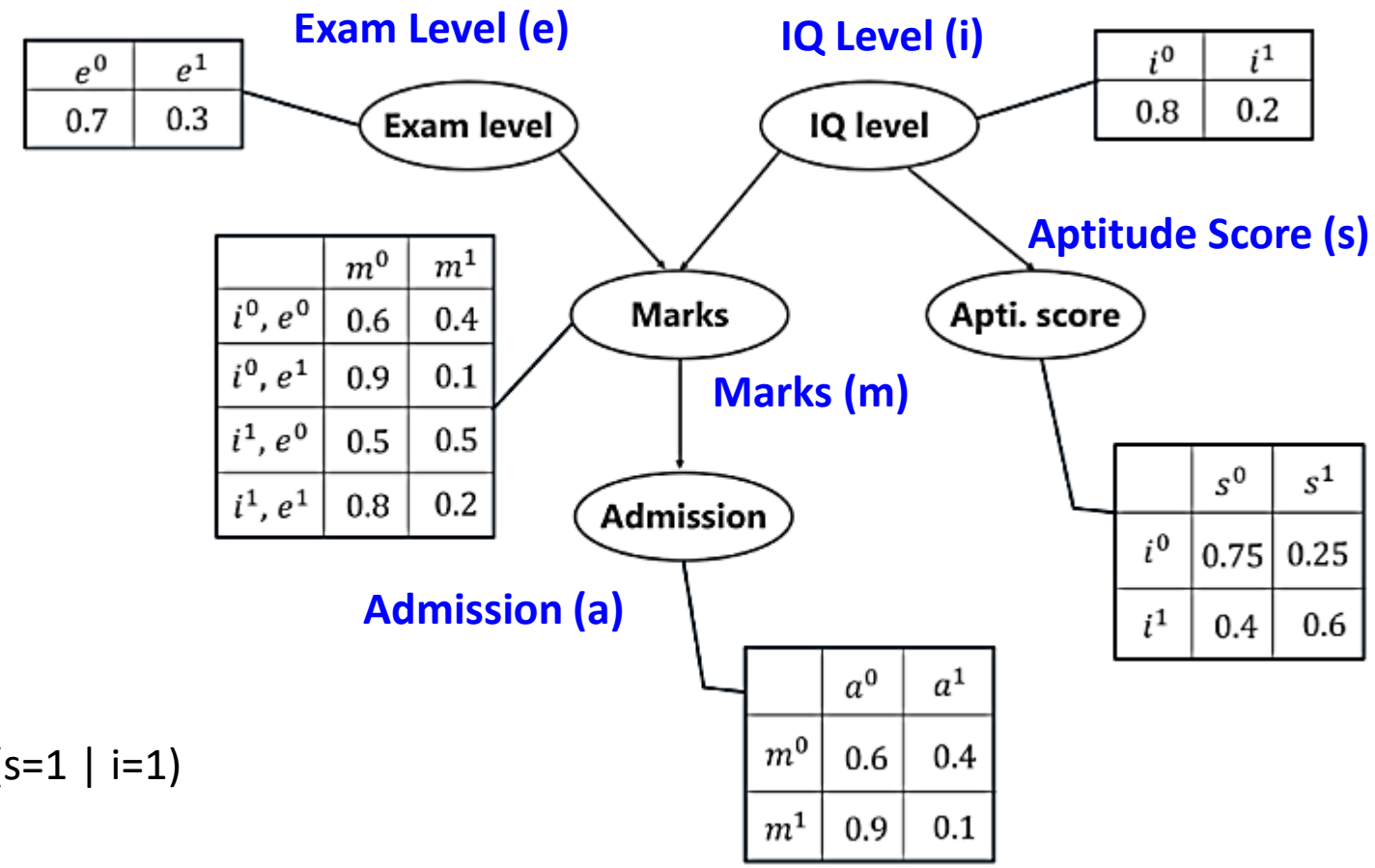
P[a=1, m=1, i=0, e=0, s=0] = P(a=1 | m=1) . P(m=1 | i=0, e=1) . P(i=0) . P(e=1) . P(s=0 | i=0)

Bayesian Belief Network : Problem 2

Case 2: In another case, calculate the probability that the student has a High IQ level and Aptitude Score, the exam being easy yet fails to pass and does not secure admission to the university.

- **Exam Level (*e*)**– This discrete variable denotes the difficulty of the exam and has two values (**0 for easy and 1 for difficult**)

- **IQ Level (*i*)** – This represents the Intelligence Quotient level of the student and is also discrete in nature having two values (**0 for low and 1 for high**)



P[a=0, m=0, i=1, e=0, s=1]

= P(a=0 | m=0) . P(m=0 | i=1, e=0) . P(i=1) . P(e=0) . P(s=1 | i=1)
= 0.6 * 0.5 * 0.2 * 0.7 * 0.6
= **0.0252**

Dempster Shafer Theory

Dempster-Shafer Theory (DST), also known as evidence theory, is a mathematical framework for reasoning under uncertainty and handling evidence in a principled manner. It was developed by Arthur P. Dempster and Glenn Shafer.

1. Basic Probability Assignment (BPA):
- **Symbol:** $m(A)$
- **Meaning:** A measure of belief assigned to a specific subset of a sample space A.
- **Example:** If we are uncertain about the color of a car and believe it could be red, blue, or green, we may assign belief masses $m(\text{Red})=0.4$, $m(\text{Blue})=0.3$, $m(\text{Green})=0.3$.

2. Mass Function:
- **Symbol:** m
- **Meaning:** A function that assigns belief masses to all possible subsets of the sample space.
- **Example:** For the car example, the mass function m would specify belief masses for all possible colors.

Dempster Shafer Theory

3. Dempster's Rule of Combination:

- **Symbol:** $\oplus$
- **Meaning:** Combines belief masses from different pieces of evidence.
- **Formula:**

$$m_C(X) = \frac{\sum_{A \cap B = X} m_A(A) \cdot m_B(B)}{1 - \sum_{A \cap B = \emptyset} m_A(A) \cdot m_B(B)}$$

- **Example:** Combining evidence about car colour from two sources, each providing belief masses m_1 and m_2, we use Dempster's rule to obtain a combined belief mass m_C.

Dempster Shafer Theory

Example 1: Medical Diagnosis

- **Scenario:** A patient's symptoms are being considered for a diagnosis.
- **Evidence Sources:** Symptoms A, B, C.
- **Belief Assignment:** $m_A(A)=0.3$, $m_B(B)=0.4$, $m_C(C)=0.2$.
- **Combining Evidence:** Use Dempster's rule to combine belief masses and obtain an overall belief assignment.

Given Information:

- **Symptoms:** A, B, C
- **Belief Assignment:**
  - $mA(A)=0.3$
  - $mB(B)=0.4$
  - $mC(C)=0.2$

$$m_C(X) = \frac{\sum_{A \cap B = X} m_A(A) \cdot m_B(B)}{1 - \sum_{A \cap B = \emptyset} m_A(A) \cdot m_B(B)}$$

Dempster Shafer Theory

1. **Combining m_A and m_B for Symptom A and B:**
   - $m_{A\cap B}(A \cap B) = m_A(A) \cdot m_B(B) = 0.3 \cdot 0.4 = 0.12$
2. **Combining $m_{A\cap C}$ for Symptom A and C:**
   - $m_{A\cap C}(A \cap C) = m_A(A) \cdot m_C(C) = 0.3 \cdot 0.2 = 0.06$
3. **Combining $m_{B\cap C}$ for Symptom B and C:**
   - $m_{B\cap C}(B \cap C) = m_B(B) \cdot m_C(C) = 0.4 \cdot 0.2 = 0.08$
4. **Combining $m_{A\cap B\cap C}$ for Symptom A, B, and C:**
   - $m_{A\cap B\cap C}(A \cap B \cap C) = m_A(A) \cdot m_B(B) \cdot m_C(C) = 0.3 \cdot 0.4 \cdot 0.2 = 0.024$
5. **Combining $m_\emptyset$ for the empty set (no symptoms):**
   - $m_\emptyset(\emptyset) = 1 - \sum_{A\cap B=\emptyset} m_A(A) \cdot m_B(B) = 1 - (1 - 0.3 \cdot 0.4) = 0.88$
6. **Applying Dempster's Rule for each subset:**
   - $m_C(A) = \frac{0.12}{0.88} = 0.1364$
   - $m_C(B) = \frac{0.12}{0.88} = 0.1364$
   - $m_C(C) = \frac{0.06+0.08+0.024}{0.88} = 0.2386$
   - $m_C(\emptyset) = \frac{0.88-(0.12+0.06+0.08+0.024)}{0.88} = 0.5955$

Result:
The combined belief assignment for symptoms A, B, and C using Dempster-Shafer Theory is as follows:

$$m_C(A) = 0.1364$$
$$m_C(B) = 0.1364$$
$$m_C(C) = 0.2386$$
$$m_C(\emptyset) = 0.5955$$

These values represent the updated belief masses after combining evidence from multiple sources.

Dempster Shafer Theory

Example 2: Fault Detection in a System

- **Scenario:** Detecting faults in a complex system using multiple sensors.
- **Evidence Sources:** Sensors X, Y, Z.
- **Belief Assignment:** $mX(\text{Fault})=0.6$, $mY(\text{Fault})=0.3$, $mZ(\text{Fault})=0.5$.
- **Combining Evidence:** Apply Dempster's rule to combine beliefs and make a decision about the presence of a fault.