Artificial Intelligence ADC503

Prof. Smita S. Mane

Asst. Prof.

Dept. of AI & DS,

VESIT,Chembur

Chapter – 04

Knowledge and Reasoning

4		**Knowledge and Reasoning**	10
	4.1	Definition and importance of Knowledge, Issues in Knowledge Representation, Knowledge Representation Systems, Properties of Knowledge Representation Systems	
	4.2	Propositional Logic (PL): Syntax, Semantics, Formal logic-connectives, truth tables, tautology, validity, well-formed-formula, Introduction to logic programming (PROLOG)	
	4.3	Predicate Logic: FOPL, Syntax, Semantics, Quantification, Inference rules in FOPL,	
	4.4	Forward Chaining, Backward Chaining and Resolution in FOPL	

Content

- What is knowledge ? partially observable environment
- Importance of Knowledge representation in Intelligent agents
- Use of propositional logic, predicate logic, First order logic – syntax, semantics, validity , entailment etc
- Represent natural language statements in logic •

Deduct new sentences by applying inference rules.

What is Knowledge?

- Knowledge is **what I know** and Information is **what we know**
- Knowledge can be considered as the distillation of information that has been **collected, classified, organized, integrated, abstracted and value added.**
- Intelligent behavior is not dependent so much on the methods of reasoning as on the knowledge one has to reason with.

- **Meta Knowledge:** It is the information/knowledge about knowledge.
- **Heuristic Knowledge:** It is the knowledge regarding a specific topic.
- **Procedural Knowledge:** It gives information about achieving something.
- **Declarative Knowledge:** It is the information which describes a particular obj and its attributes.
- **Structural Knowledge:** It describes the knowledge between the objects.

Knowledge

- In artificial intelligence (AI), knowledge is how machines learn to act like humans by using their prior experiences and knowledge to perform tasks.

Knowledge: Knowledge is awareness or familiarity gained by experiences of facts, data, and situations. Following are the types of knowledge in artificial intelligence:

- Following are the kind of knowledge which needs to be represented in AI systems:
    - **Object:** All the facts about objects in our world domain. E.g., Guitars contains strings, trumpets are brass instruments.
    - **Events:** Events are the actions which occur in our world.
    - **Performance:** It describe behavior which involves knowledge about how to do things.
    - **Meta-knowledge:** It is knowledge about what we know.
    - **Facts:** Facts are the truths about the real world and what we represent.
    - Knowledge-Base: The central component of the knowledge-based agents is the **knowledge base.** It is represented as KB. The Knowledgebase is a group of the Sentences (Here, sentences are used as a technical term and not identical with the English language).

- Knowledge representation and reasoning (KR, KRR) is the part of Artificial intelligence which concerned with AI **agents thinking and how thinking contributes to intelligent behavior of agents.**
- It is responsible for representing information about the real world so that a computer can understand and can utilize this knowledge to solve the complex real world problems such as diagnosis a medical condition or communicating with humans in natural language.
- It is also a way which describes how we can represent knowledge in artificial intelligence. Knowledge representation is not just storing data into some database, but it also enables an intelligent machine to learn from that knowledge and experiences so that it can behave intelligently like a human.


Knowledge
Declare active Knowledge
Object facts
Structural Knowledge
Relationship between object, concept
Heuristic Knowledge
Rule of thumb
Meta Knowledge
Knowledge about knowledge
Procedural Knowledge
Rules Procedure

Declarative Knowledge:

- Declarative knowledge is to know about something.
- It includes concepts, facts, and objects.
- It is also called descriptive knowledge and expressed in declarative sentences.
- It is simpler than procedural language.

2. Procedural Knowledge

- It is also known as imperative knowledge.
- Procedural knowledge is a type of knowledge which is responsible for knowing how to do something.
- It can be directly applied to any task.
- It includes rules, strategies, procedures, agendas, etc.
- Procedural knowledge depends on the task on which it can be applied.

3. Meta-knowledge:

- Knowledge about the other types of knowledge is called Meta-knowledge.

4. Heuristic knowledge:

- Heuristic knowledge is representing knowledge of some experts in a filed or subject.
- Heuristic knowledge is rules of thumb based on previous experiences, awareness of approaches, and which are good to work but not guaranteed.

5. Structural knowledge:

- Structural knowledge is basic knowledge to problem-solving.
- It describes relationships between various concepts such as kind of, part of, and grouping of something.
- It describes the relationship that exists between concepts or objects.

Characteristics of Knowledge

- –Knowledge is **huge** (Large in number or quantity).
- –Knowledge is **hard to characterize** accurately.
- –Knowledge differs from data in that it is organized such that it corresponds to the ways it will be used.
- –Knowledge is **interpreted differently** by **different people.**

Knowledge

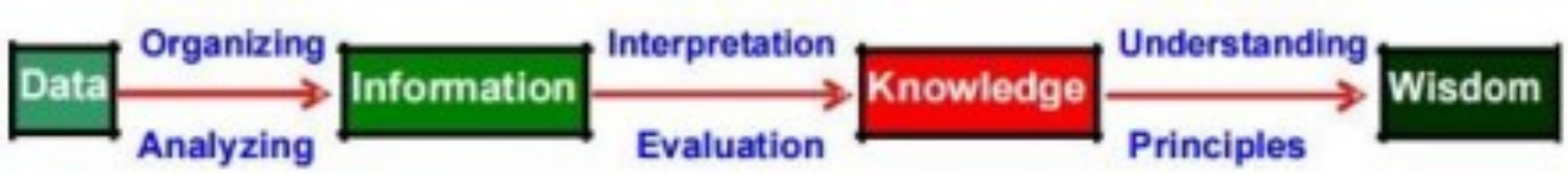


Fig 1 Knowledge Progression

- **Data** : Collection of **disconnected facts**
- **Information** :Emerges when **Relationships between facts** are established and understood. Provides answer to **Who,what, where and when.**

- **Knowledge** : emerges when **relationship between patterns** are identified and understood.
- Provides answer to **"How"**
- **Wisdom** : is the pinnacle of understanding, uncovers the **principles of relationships that describe patterns**.
- Provides answer as **" Why"**

Knowledge Model

- **Data and information** : Past data
- **Knowledge :** Present/ enable us to perform
- **Wisdom :** Future – acquire vision for what will be

Knowledge based agents / Intelligent agents

Importance of Knowledge representation in Intelligent Agents

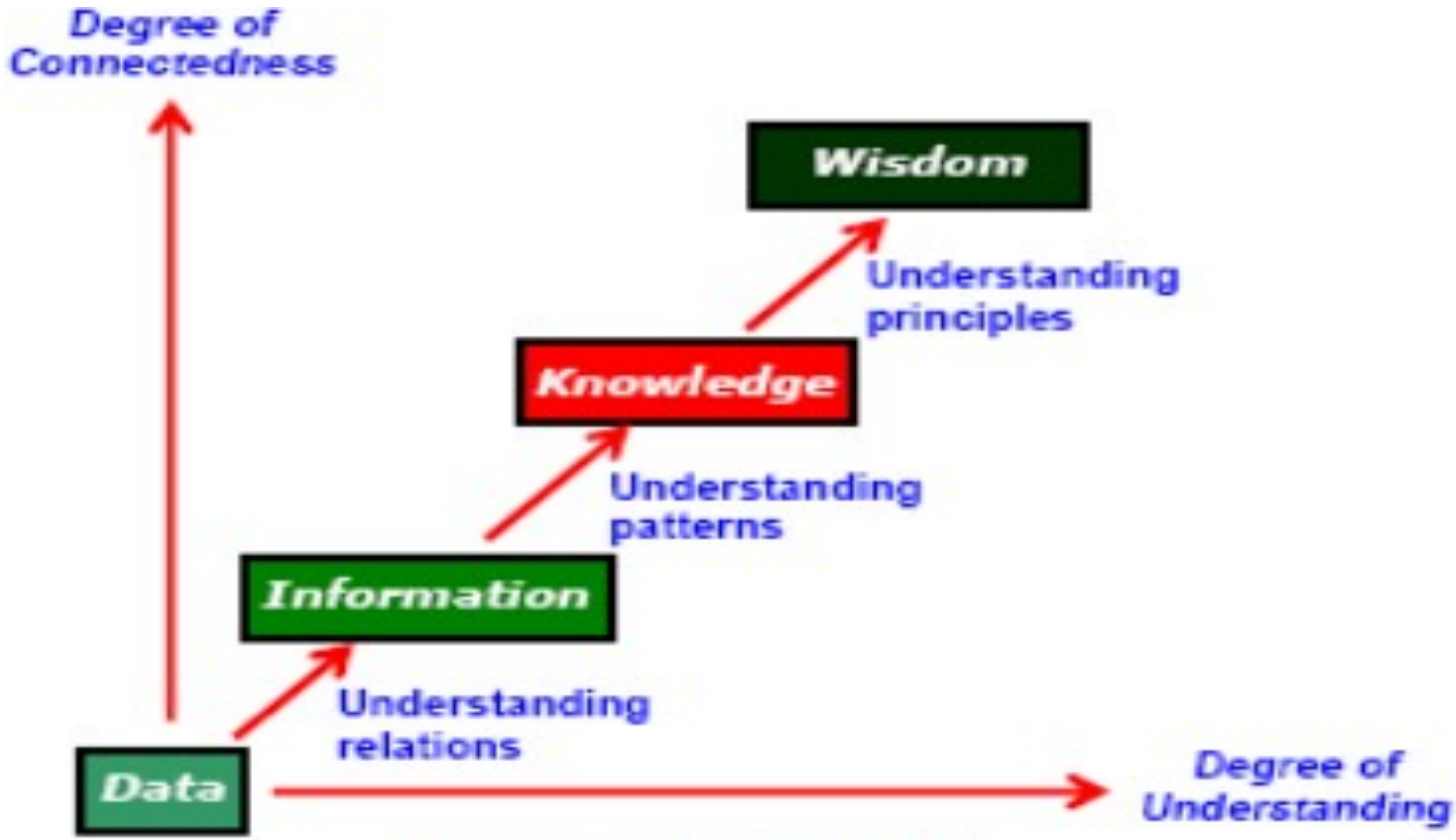


Fig. Knowledge Model

- **Intelligent agents** should have capacity for
  - **Perceiving** : acquiring information from environment
  - **Knowledge Representation** : representing its understanding of the world
  - **Reasoning :** inferring the implications of what it knows and of the choices it has
  - **Acting** : choosing what it want to do and carry it out.

Knowledge-based Agent

- Intelligent agents need knowledge about the world for making good decisions.
- A knowledge-based agent includes a knowledge base and an inference system.
- A knowledge base is a set of sentences.

What Agents do ?

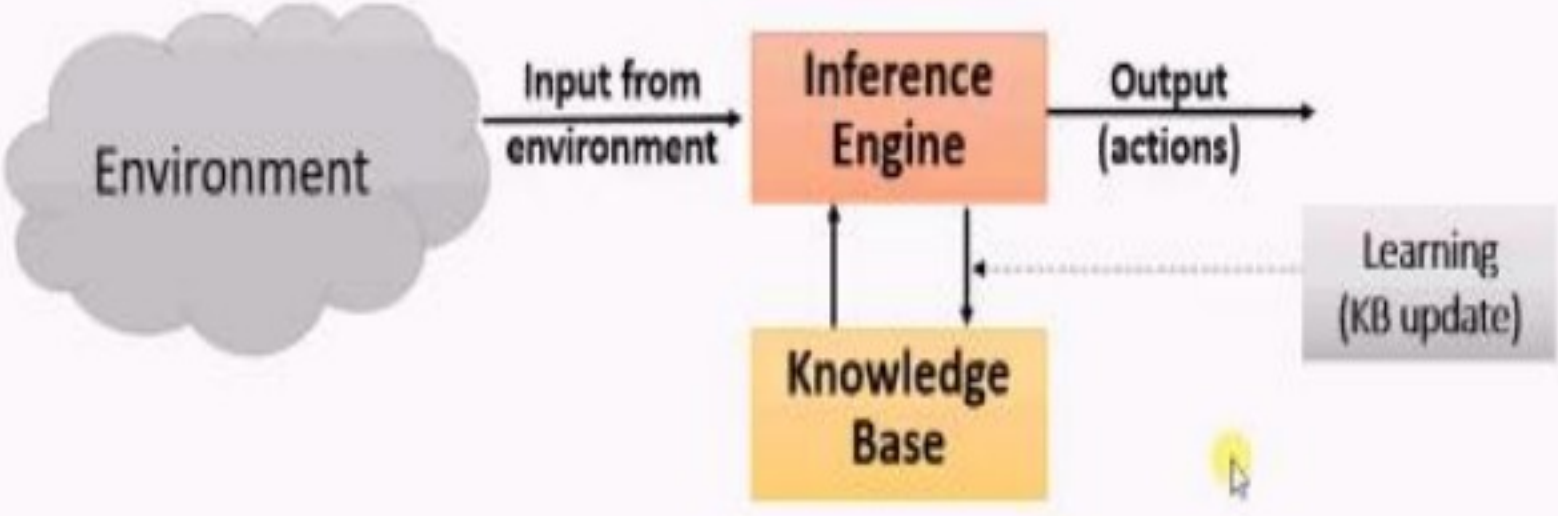
- It TELLs the knowledge base what it perceives.
- It ASKs the knowledge base what action it should perform.
- It performs the chosen action.

Types of Knowledge

Knowledge Base Agent

- **Implicit type : exist within human being – unconscious knowledge** – cannot be expressed in language . Exist with human beings, cannot copied, drawn from experiences, subjective insight Skills
- **Explicit Type : exist outside human being, can be shared ,copied, stored etc**
  - **Procedural knowledge** - knowing how to do something , e.g. Ram is older or John ? Soln – find their ages ? - algorithm
  - **Declarative knowledge**– Statement that can be True/false – Specification

Architecture of KBA



Logical Agents

4		**Knowledge and Reasoning**	10
	4.1	Definition and importance of Knowledge, Issues in Knowledge Representation, Knowledge Representation Systems, Properties of Knowledge Representation Systems	
	4.2	Propositional Logic (PL): Syntax, Semantics, Formal logic-connectives, truth tables, tautology, validity, well-formed-formula, Introduction to logic programming (PROLOG)	
	4.3	Predicate Logic: FOPL, Syntax, Semantics, Quantification, Inference rules in FOPL,	
	4.4	Forward Chaining, Backward Chaining and Resolution in FOPL	

"Knowledge Representation and Reasoning"

- Logic → Propositional logic
  → Predicate logic
- Rules → if then
- Semantic Net → GoogleGraph
  Meaning graph
- Frame → Slots and fillers
  Object Attribute
- Script

Syntax

Semantic

Propositional logic (PL)

- Propositional logic (PL) is the simplest form of logic where all the statements are made by propositions.
- A proposition is a declarative statement which is either true or false. It is a technique of knowledge representation in logical and mathematica) It is Sunday.
- E:g

b) The Sun rises from West (False proposition)

c) 3+3= 7(False proposition)

d) 5 is a prime number. al form.

- PL is also called Boolean logic as it works on 0 and 1.
- In PL, use symbolic variables to represent the logic, and we can use any symbol for a representing a proposition, such A, B, C, P, Q, R, etc.
- Propositions can be either true or false, but it cannot be both.
- PL consists of an object, relations or function, and logical connectives.
- These connectives are also called **logical operators.**
- The propositions and connectives are the basic elements of the propositional logic.
- Connectives can be said as a logical operator which connects two sentences.
- A proposition formula which is **always true** is called **tautology**, and it is also called a valid sentence.
- A proposition formula which is **always false** is called **Contradiction.**

Syntax of propositional logic:

- The syntax of propositional logic defines the **allowable sentences** for the knowledge representation.
- There are two types of Propositions:
  1. Atomic Propositions
  2. Compound propositions

- **Atomic Proposition:**

- simple propositions.
- It consists of a single proposition symbol.
- These are the sentences which must be either true or false.

a) 2+2 is 4, it is an atomic proposition as it is a true fact.

b) "The Sun is cold" is also a proposition as it is a false fact.

❑ **Compound proposition:**

- Compound propositions are constructed by combining simpler or atomic propositions, using parenthesis and logical connectives.
- Example:

  a) "It is raining today, and street is wet."

  b) "Ankit is a doctor, and his clinic is in Mumbai."

Logical Connectives:

- Logical connectives are used to connect two simpler propositions or representing a sentence logically.
- We can create compound propositions with the help of logical connectives.
- There are mainly five connectives,

Logical Connectives:

1) **Negation:** A sentence such as ¬ P is called negation of P. A literal can be either Positive literal or negative literal.
2) **Conjunction(and):** A sentence which has ∧ connective such as, P ∧ Q is called a conjunction.

- Example: Rohan is intelligent and hardworking. It can be written as,
- P= Rohan is intelligent,
- Q= Rohan is hardworking. → P∧ Q.

1) **Disjunction(Or):** A sentence which has ∨ connective, such as P ∨ Q. is called disjunction, where P and Q are the propositions.

- Example: "Ritika is a doctor or Engineer",
- Here P= Ritika is Doctor. Q= Engineer, so we can write it as P ∨ Q.

1) **Implication:** A sentence such as P → Q, is called an implication. Implications are also known as if-then rules. It can be represented as

- If it is raining, then the street is wet.
- Let P= It is raining, and Q= Street is wet, so it is represented as P → Q

1) **Biconditional:** A sentence such as P⇔ Q is a Biconditional sentence, example If I am breathing, then I am alive

- P= I am breathing, Q= I am alive, it can be represented as P ⇔ Q.

Propositional Logic (Either True or False, not Both)

Syntax Semantic

Atomic Complex

$1+1=2$ T

$2+1=4$ F

ND is C. T

Some st. are Int. T/F

T	T	T	T	T	T
T	F	T	F	F	F
F	T	T	F	T	F
F	F	F	F	T	T

$\underbrace{12+}=4$

Today is friday.

$-P$

Atomic: $1+1=2$ T

$-$ F

$P \rightarrow Q$

Complex:

- $\neg$ Negation (Today is Not Friday)
- $\vee$ Disjuction (You should Eat or Watch TV at a time) — $P \vee Q$
- $\wedge$ Conjuction (Please like my video And Subscribe my channel) — $P \wedge Q$
- $\rightarrow$ if then (if there is rain then the roads are wet)
- $\leftrightarrow$ iff (I will go to Mall iff I have to do shopping)

* You can access the internet from Campus only if you are CSE student or you are not freshman.

Connective symbols	Word	Technical term	Example
$\wedge$	AND	Conjunction	$A \wedge B$
$\vee$	OR	Disjunction	$A \vee B$
$\rightarrow$	Implies	Implication	$A \rightarrow B$
$\Leftrightarrow$	If and only if	Biconditional	$A \Leftrightarrow B$
$\neg$ or $\sim$	Not	Negation	$\neg A$ or $\neg B$

Truth table

For Negation:

P	¬ P
True	**False**
False	**True**

For Conjunction:

P	Q	P∧Q
True	True	**True**
True	False	**False**
False	True	**False**
False	False	**False**

For disjunction:

P	Q	P∨Q.
True	True	**True**
False	True	**True**
True	False	**True**
False	False	**False**

Truth table

For Implication:

P	Q	P→ Q
True	True	**True**
True	False	**False**
False	True	**True**
False	False	**True**

For Biconditional:

P	Q	P⇔ Q
True	True	**True**
True	False	**False**
False	True	**False**
False	False	**True**

Truth table three propositions:

P	Q	R	¬R	P∨Q	P∨Q→¬R
True	True	True	False	True	False
True	True	False	True	True	True
True	False	True	False	True	False
True	False	False	True	True	True
False	True	True	False	True	False
False	True	False	True	True	True
False	False	True	False	False	True
False	False	False	True	False	True

Precedence of connectives:

Precedence	**Operators**
First Precedence	Parenthesis
Second Precedence	Negation
Third Precedence	Conjunction(AND)
Fourth Precedence	Disjunction(OR)
Fifth Precedence	Implication
Six Precedence	Biconditional

Knowledge Base

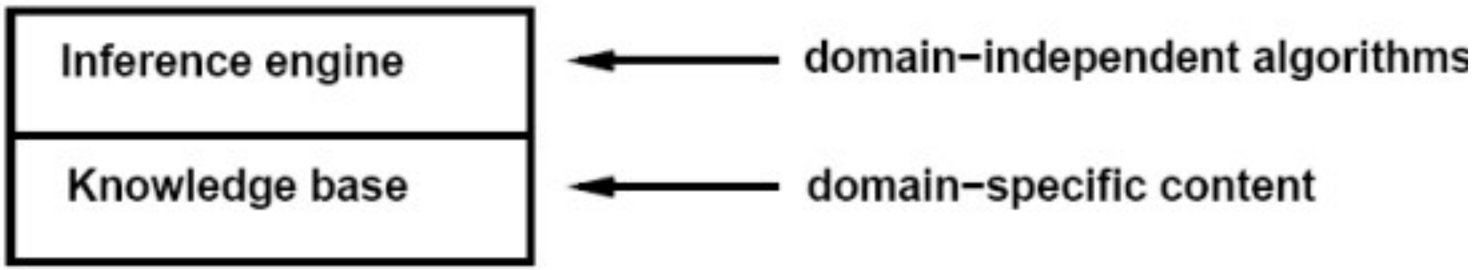
- Each time the agent program is called, it does three things.
- **First, it TELLs the knowledge base what it perceives.**
- **Second, it ASKs the knowledge base what action it should perform**
- **In the process of answering this query, extensive reasoning may be done about the current state of the world, about the outcomes of possible action sequences, and so on. Third, the agent program TELLs the knowledge base which action was chosen, and the agent executes the action.**

Generic Knowledge Base (KB) Based Agent

- **Knowledge Base:** set of sentences represented in a knowledge representation language and represents assertions about the world.

ask

tell



- **Inference rule:** when **one ASKs questions** of the KB, the answer should ***follow*** from what has been **TELLed** to the **KB previously**.

Reasoning

- Propositional Logic
- First order Logic
- Semantic Networks
- Frame representation

Logic

- Logic is concerned with the **truth of statements** about the world.
  - Generally each statement is either **TRUE or FALSE**.
  - Logic includes : **Syntax , Semantics and Inference Procedure**

.

- Different logics exist, which allow you to represent different kinds of things, and which allow more or less efficient inference.
- Propositional logic, Predicate logic, Temporal logic, Modal logic, Description logic..
-

SYNTAX and SEMANTICS

- Knowledge bases consist of sentences. (syntax and semantics)

- SYNTAX : **“x + y = 4” is a well-formed sentence, whereas “x4y+ =” is not**.

- SEMANTICS : A logic must also define the semantics or meaning of sentences. The semantics defines TRUTH the truth of each sentence with respect to each possible world.

- Example, **the semantics POSSIBLE WORLD for arithmetic specifies that the sentence “x + y = 4” is true in a world where x is 2 and y is 2, but false in a world where x is 1 and y is 1**.

- In standard logics, every sentence must be either true or false in each possible world—there is no “in between.

Logic

•Types of Logic

Facts, Objects, Relations, Time

"I am *always* hungry", "I will *eventually* be hungry"

- **Propositional Logic** T/F/Unknown
- **First Order Logic** T/F/Unknown
- **Temporal Logic** T/F/Unknown
- **Fuzzy Logic** Degree of Truth Degree of Belief (0…1)
- **What is probability that our new manager will be fat?**

Model of a Formula

- If the value of the formula X holds 1 for the assignment A, then the assignment A is called model for formula X.
- That means, all assignments for which the formula X is true are models of it.

assignment	a	b	$a \rightarrow b$	$a \wedge b$	$\neg(a \leftrightarrow b)$
A	0	0	1	0	1
B	0	1	1	0	0
C	1	0	0	0	0
D	1	1	1	1	1

a) The assignments A, B and D are models of the formula $a \rightarrow b$.

b) The assignment D is model of the formula $a \wedge b$.

c) The assignments A and D are models of the formula $\neg(a \leftrightarrow b)$.

Satisfiable Formulas

- If there exist at least one model of a formula then the formula is called satisfiable.
- The value of the formula is true for at least one assignment. It plays no rule how many models the formula has.

assignment	a	b	$a \rightarrow b$	$a \wedge b$	$\neg(a \leftrightarrow b)$	$\neg(a \vee b)$	$a \vee (\neg a)$
A	0	0	1	0	1	1	1
B	0	1	1	0	0	0	1
C	1	0	0	0	0	0	1
D	1	1	1	1	1	0	1

All these formulas are called satisfiable, because they have at least one model.

Valid Formulas

- A formula is called valid (or tautology) if all assignments are models of this formula.
- The value of the formula is true for all assignments. If a tautology is part of a more complex formula then you could replace it by the value 1.

valid Formulas example

- $1 \vee 1$
- $a \vee (\neg a)$
- $0 \vee a \leftrightarrow a$
- $1 \wedge a \leftrightarrow a$

Unsatisfiable Formulas

- A formula is unsatisfiable if none of its assignment is true in no models

- $0 \wedge 1$
- $a \wedge 0$
- $a(\neg a)$
- $((\neg a)b \vee a(\neg b)) \leftrightarrow (ab \vee (\neg a)(\neg b))$

- ***p*: You get an A on the final exam**
- ***q*: You do every exercise in the book.**
- ***r*: You get an A in this class.**
- Write the following formulas using *p*, *q*, and *r* and logical connectives.
- You get an A in this class, but you do not do every exercise
  in the book.?
- To get an A in this class, it is necessary for you to get an A on the final.?
- Getting an A on the final and doing every exercise in the book is sufficient for getting an A in this class.?

- $pAq \Rightarrow r$

Solution

4		**Knowledge and Reasoning**	10
	4.1	Definition and importance of Knowledge, Issues in Knowledge Representation, Knowledge Representation Systems, Properties of Knowledge Representation Systems	
	4.2	Propositional Logic (PL): Syntax, Semantics, Formal logic-connectives, truth tables, tautology, validity, well-formed-formula, Introduction to logic programming (PROLOG)	
	4.3	Predicate Logic: FOPL, Syntax, Semantics, Quantification, Inference rules in FOPL,	
	4.4	Forward Chaining, Backward Chaining and Resolution in FOPL	

Propositional logic is a weak language

- Hard to identify "individuals"
- Can't directly talk about properties of individuals or relations between individuals (e.g., "Bill is tall")
- Generalizations, patterns, regularities can't easily be represented (e.g., "all triangles have 3 sides")
- First-Order Logic (abbreviated FOL) is expressive enough to concisely represent this kind of information

FOL adds relations, variables, and quantifiers, e.g.,

- "Every elephant is gray": $\forall$ x (elephant(x) $\rightarrow$ gray(x))

PREDICATE LOGIC

- Also known as First order Logic(FOL)
- A predicate is a statement that contains variables and becomes a proposition when specific values are substituted for those variables.
- The key components of FOL include **constants, variables, predicates, functions, quantifiers, and logical connectives.**
- A predicate is an expression of one or more variables determined on some specific domain.
- A predicate with variables can be made a proposition by either authorizing a value to the variable or by quantifying the variable.
- Consider E(x, y) denote "x = y"
- Consider X(a, b, c) denote "a + b + c = 0"
- Consider M(x, y) denote "x is married to y."

PREDICATE LOGIC

- **Constants**: Constants represent specific objects within the domain of discourse. For example, in a given domain, Alice, 2, and NewYork could be constants.
- **Variables**: Variables stand for unspecified objects in the domain. Commonly used symbols for variables include x, y, and z.
- **Predicates**: Predicates are functions that return true or false, representing properties of objects or relationships between them. For example, Likes(Alice, Bob) indicates that Alice likes Bob, and GreaterThan(x, 2) means that x is greater than 2.
- **Functions:** Functions map objects to other objects. For instance, MotherOf(x) might denote the mother of x.

PREDICATE LOGIC

- **Quantifiers:** Quantifiers specify the scope of variables. The two main quantifiers are:

1) **Universal Quantifier (∀):** Indicates that a predicate applies to **all elements** in the domain. (Denotes that a statement is true for all values of a variable within a given domain.)

- For example, $\forall x\ (\text{Person}(x) \rightarrow \text{Mortal}(x))$ means "All persons are mortal."

1) **Existential Quantifier (∃):** Indicates that there is at least one element in the domain for which the predicate holds.

- For example, $\exists x\ (\text{Person}(x) \wedge \text{Likes}(x, \text{IceCream}))$ means "There exists a person who likes ice cream."

- **Logical Connectives:** Logical connectives include conjunction (∧), disjunction (∨), implication (→), biconditional (↔), and negation (¬). These connectives are used to form complex logical statements.

Syntax and Semantics

- **The syntax** of first-order logic specifies the rules for constructing valid expressions, including terms and formulas. Terms are expressions that refer to objects and include constants, variables, and functions. Formulas are logical statements formed by combining predicates, terms, quantifiers, and logical connectives.
- **The semantics** of FOL define the meaning of the expressions. An interpretation assigns a domain of discourse and provides meaning to the constants, functions, and predicates. A formula is considered true under an interpretation if it accurately describes the relationships and properties of the objects in the domain.

Predicates and Quantifiers

- The use of quantifiers and other methods to express quantity and predict class distributions in data
- Example 1:

Let $P(x)$ be the predicate "$x > 5$" where x is a real number.

$P(7)$ is true because $7 > 5$

$P(3)$ is false because 3 is not > 5

- Example 2:

Let $Q(x,y)$ be the predicate "$x + y = 10$" where x and y are integers.

$Q(3,7)$ is true because $3 + 7 = 10$

$Q(4,5)$ is false because $4 + 5 \neq 10$

Predicates and Quantifiers

Example 3:

Let R(x) be the predicate "$x^2 \geq 0$" where x is a real number.

The statement $\forall x\, R(x)$ is true because for all real numbers, their square is always non-negative.

Example 4:

Let S(x) be the predicate "$x^2 = 4$" where x is a real number.

The statement $\exists x\, S(x)$ is true because there exist real numbers (2 and -2) whose square is 4.

Predicates and Quantifiers

Let $Q(x)$ be the predicate "x is prime" where x is a positive integer.

- The statement $\forall x\, Q(x)$ is false because not all positive integers are prime.
- The statement $\exists x\, Q(x)$ is true because there exist prime numbers (e.g., 2, 3, 5, 7, etc.).

Example 7:

Let $R(x,y)$ be the predicate "$x < y$" where x and y are real numbers.

- The statement $\forall x\, \forall y\, R(x,y)$ is false because it's not true that every real number is less than every other real number.

- The statement $\exists x\, \exists y\, R(x,y)$ is true because we can find two real numbers where one is less than the other (e.g., $1 < 2$).

Inference rules

- In artificial intelligence, we need intelligent computers which can create new logic from old logic or by evidence, so generating the conclusions from evidence and facts is termed as Inference.
- Inference rules are the templates for generating valid arguments. Inference rules are applied to derive proofs in artificial intelligence, and the proof is a sequence of the conclusion that leads to the desired goal.
- In inference rules, the implication among all the connectives plays an important role.

Inference rules

- **Implication:** It is one of the logical connectives which can be represented as $P \rightarrow Q$. It is a Boolean expression.
- **Converse:** The converse of implication, which means the right-hand side proposition goes to the left-hand side and vice-versa. It can be written as $Q \rightarrow P$.
- **Contrapositive:** The negation of converse is termed as contrapositive, and it can be represented as $\neg Q \rightarrow \neg P$.
- **Inverse:** The negation of implication is called inverse. It can be represented as $\neg P \rightarrow \neg Q$.

Inference rules Truth table

-

P	Q	$P \to Q$	$Q \to P$	$\neg Q \to \neg P$	$\neg P \to \neg Q$.
T	T	T	T	T	T
T	F	F	T	F	T
F	T	T	F	T	F
F	F	T	T	T	T

Types of Inference rules:

1. Modus Ponens:

The Modus Ponens rule is one of the most important rules of inference, and it states that **if P and P → Q is true,** then we can infer that Q will be true. It can be represented as:

Notation for Modus ponens: $$\frac{P \to Q,\ P}{\therefore Q}$$

Example:

Statement-1: "If I am sleepy then I go to bed" ==> P→ Q
Statement-2: "I am sleepy" ==> P
Conclusion: "I go to bed." ==> Q.
Hence, we can say that, if P→ Q is true and P is true then Q will be true.

Proof by Truth table:

Types of Inference rules:

1. Modus Ponens:

Example:

Statement-1: "If I am sleepy then I go to bed" ==> $P \rightarrow Q$
Statement-2: "I am sleepy" ==> P
Conclusion: "I go to bed." ==> Q.
Hence, we can say that, if $P \rightarrow Q$ is true and P is true then Q will be true.

Proof by Truth table:

P	Q	$P \rightarrow Q$
0	0	0
0	1	1
1	0	0
1	1	1

Types of Inference rules:

2. **Modus Tollens:**

The Modus Tollens rule state that if P→ Q is true and ¬ Q is true, then ¬ P will also true. It can be represented as:

Notation for Modus Tollens: $$\frac{P \to Q, \ \sim Q}{\sim P}$$

Example:

Statement-1: "If I am sleepy then I go to bed" ==> P→ Q

Statement-2: "I do not go to the bed."==> ~Q

Statement-3: Which infers that "I am not sleepy" => ~P

Types of Inference rules:

2. **Modus Tollens:**

The Modus Tollens rule state that if $P \rightarrow Q$ is true and $\neg Q$ is true, then $\neg P$ will also true. It can be represented as:

Example:

Statement-1: "If I am sleepy then I go to bed" ==> $P \rightarrow Q$
Statement-2: "I do not go to the bed."==> $\sim Q$
Statement-3: Which infers that "I am not sleepy" => $\sim P$

P	Q	$\sim P$	$\sim Q$	$P \rightarrow Q$
0	0	1	1	1
0	1	1	0	1
1	0	0	1	0
1	1	0	0	1

Types of Inference rules:

3.**Hypothetical Syllogism**

The Hypothetical Syllogism rule state that if $P \rightarrow R$ is true whenever $P \rightarrow Q$ is true, and $Q \rightarrow R$ is true. It can be represented as the following notation:

Example:

- Statement-1: If you have my home key then you can unlock my home. $P \rightarrow Q$
- Statement-2: If you can unlock my home then you can take my money. $Q \rightarrow R$
- Conclusion: If you have my home key then you can take my money. $P \rightarrow R$

P	Q	R	$P \to Q$	$Q \to R$	$P \to R$
0	0	0	1	1	1
0	0	1	1	1	1
0	1	0	1	0	1
0	1	1	1	1	1
1	0	0	0	1	1
1	0	1	0	1	1
1	1	0	1	0	0
1	1	1	1	1	1

Problems

[Wampus Problem Grid]()

4		**Knowledge and Reasoning**	10
	4.1	Definition and importance of Knowledge, Issues in Knowledge Representation, Knowledge Representation Systems, Properties of Knowledge Representation Systems	
	4.2	Propositional Logic (PL): Syntax, Semantics, Formal logic-connectives, truth tables, tautology, validity, well-formed-formula, Introduction to logic programming (PROLOG)	
	4.3	Predicate Logic: FOPL, Syntax, Semantics, Quantification, Inference rules in FOPL,	
	4.4	Forward Chaining, Backward Chaining and Resolution in FOPL	

Inference engine

- The inference engine is the **component** of the intelligent system in artificial intelligence, which **applies logical rules** to the knowledge base to infer new information from known facts.
- The first inference engine was part of the expert system.
- Inference engine commonly proceeds in two modes

1) Forward chaining
2) Backward chaining

Inference engine

- **Horn Clause and Definite clause:**

  This are the forms of sentences, which enables knowledge base to use a more restricted and efficient inference algorithm.

- **Definite clause:**

  A clause which is a disjunction of literals with exactly one positive literal is known as a **definite clause** or **strict horn clause.**

- **Horn clause:**

  A clause which is a disjunction of literals with at most one positive literal is known as **horn clause.**

  Hence all the definite clauses are horn clauses.

⇒ Forward Chaining

- Starts with the known facts and asserts new facts.

⇒ Backward Chaining

- Starts with goals and works backward to determine what facts must be asserted so that the goals can be achieved.

"A" & "A→B" to B

⇒ It is raining (A)

"If it is raining, the road is wet"
(A) (B)

The road is wet (B)

"B" & "A→B" to A

⇒ The road is wet (B)

⇒ If it is raining, the road is wet
(A→B)

⇒ It is raining (A)

fac

the

Forward Chaining

- It is also known as a forward deduction or forward reasoning method when using an inference engine.
- It is a form of reasoning which start with atomic sentences in the knowledge base and applies inference rules (Modus Ponens) in the forward direction to extract more data until a goal is reached.
- The Forward-chaining algorithm starts from known facts, triggers all rules whose premises are satisfied, and add their conclusion to the known facts.
- This process repeats until the problem is solved.

Example:

A

A -> B

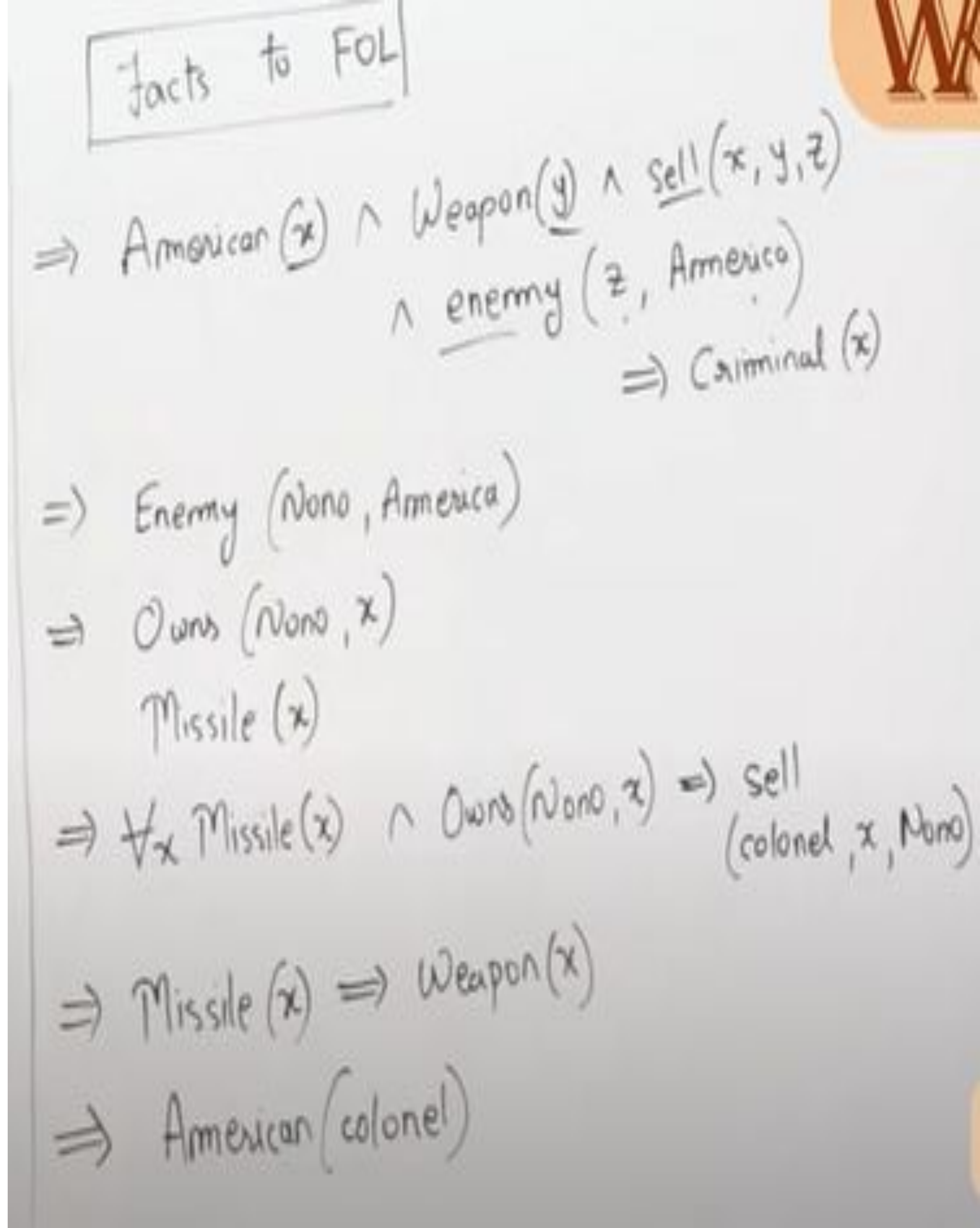
B

He is running.

If he is running, he sweats.

He is sweating.

- [Example Video]()
- [Example]()
- [Forward and backward Example]()



Properties of Forward-Chaining:

- It is a **down-up approach,** as it moves from bottom to top.
- It is a process of making a conclusion b**ased on known facts** or data, by starting from the initial state and reaches the goal state.
- Forward-chaining approach is also called as **data-driven** as we reach to the goal using available data.
- Forward -chaining approach is commonly used in the expert system, such as CLIPS, business, and production rule systems.

4		**Knowledge and Reasoning**	10
	4.1	Definition and importance of Knowledge, Issues in Knowledge Representation, Knowledge Representation Systems, Properties of Knowledge Representation Systems	
	4.2	Propositional Logic (PL): Syntax, Semantics, Formal logic-connectives, truth tables, tautology, validity, well-formed-formula, Introduction to logic programming (PROLOG)	
	4.3	Predicate Logic: FOPL, Syntax, Semantics, Quantification, Inference rules in FOPL,	
	4.4	Forward Chaining, Backward Chaining and Resolution in FOPL	

Backward Chaining

- A backward chaining algorithm is a form of reasoning, which **starts with the goal and works backward**, chaining through rules to find known facts that support the goal.

Example:

B

A -> B

A

He is sweating.

If he is running, he sweats.

He is running.

Example:

A

A -> B

B

He is running.

If he is running, he sweats.

He is sweating.

Properties of Backward-Chaining:

- It is known as a **top-down** approach.
- It is based on modus ponens inference rule.
- In this ,the **goal is broken into sub-goal** or sub-goals to prove the facts true.
- It is called a **goal-driven approach**, as a list of goals decides which rules are selected and used.
- It is used in game theory, automated theorem proving tools, inference engines, proof assistants, and various AI applications.
- The backward-chaining method mostly used a **depth-first search** strategy for proof.
-

Example

We have to prove that colonel is a criminal

- Given Facts

- It is crime for an American to sell weapons to the enemy of America.
- Country Nono is an enemy of America.
- Nono has some Missiles.
- All the missiles were sold to Nono by colonel.
- Missile is a weapon.
- colonel is American.

$\boxed{\text{Facts to FOL}}$

$\Rightarrow$ $American(x) \wedge Weapon(y) \wedge sell(x, y, z) \wedge enemy(z, America) \Rightarrow Criminal(x)$

$\Rightarrow$ $Enemy(Nono, America)$

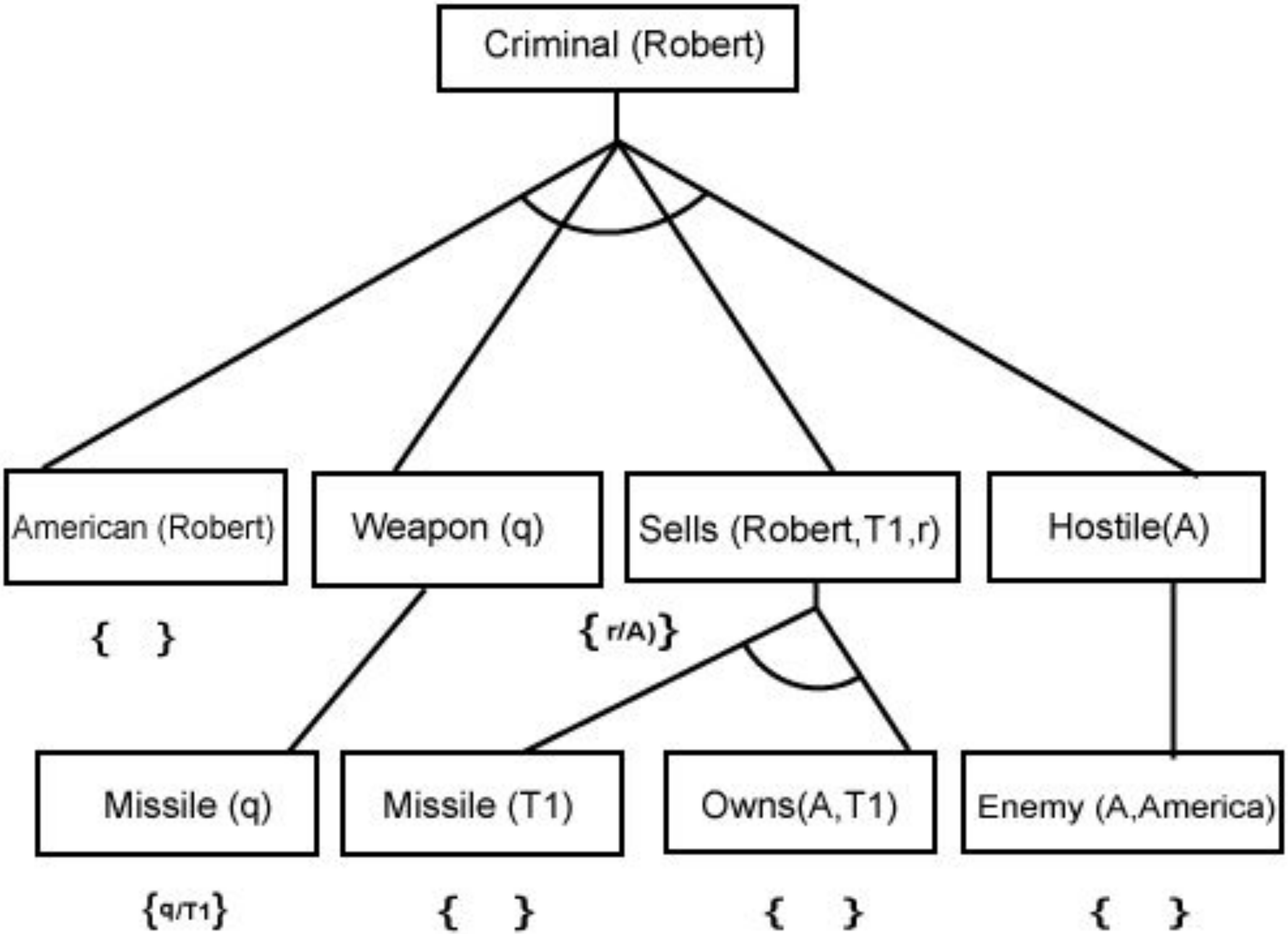
$\Rightarrow$ $Owns(Nono, x)$

$Missile(x)$

$\Rightarrow$ $\forall x\ Missile(x) \wedge Owns(Nono, x) \Rightarrow sell(colonel, x, Nono)$

$\Rightarrow$ $Missile(x) \Rightarrow Weapon(x)$

$\Rightarrow$ $American(colonel)$


Criminal (Robert)
American (Robert)
Weapon (q)
Sells (Robert,T1,r)
Hostile(A)
{ }
{ r/A)}
Missile (q)
Missile (T1)
Owns(A,T1)
Enemy (A,America)
{q/T1}
{ }
{ }
{ }

Backward chaning

Example Video

Forward and backward Example

When based on available data a decision is taken then the process is called as Forward chaining.	Backward chaining starts from the goal and works backward to determine what facts must be asserted so that the goal can be achieved.
Forward chaining is known as data-driven technique because we reaches to the goal using the available data.	Backward chaining is known as goal-driven technique because we start from the goal and reaches the initial state in order to extract the facts.
It is a bottom-up approach.	It is a top-down approach.
It applies the Breadth-First Strategy.	It applies the Depth-First Strategy.
Its goal is to get the conclusion.	Its goal is to get the possible facts or the required data.
Slow as it has to use all the rules.	Fast as it has to use only a few rules.
It operates in forward direction i.e it works from initial state to final decision.	It operates in backward direction i.e it works from goal to reach initial state.
Forward chaining is used for the planning, monitoring, control, and interpretation application.	It is used in automated inference engines, theorem proofs, proof assistants and other artificial intelligence applications.