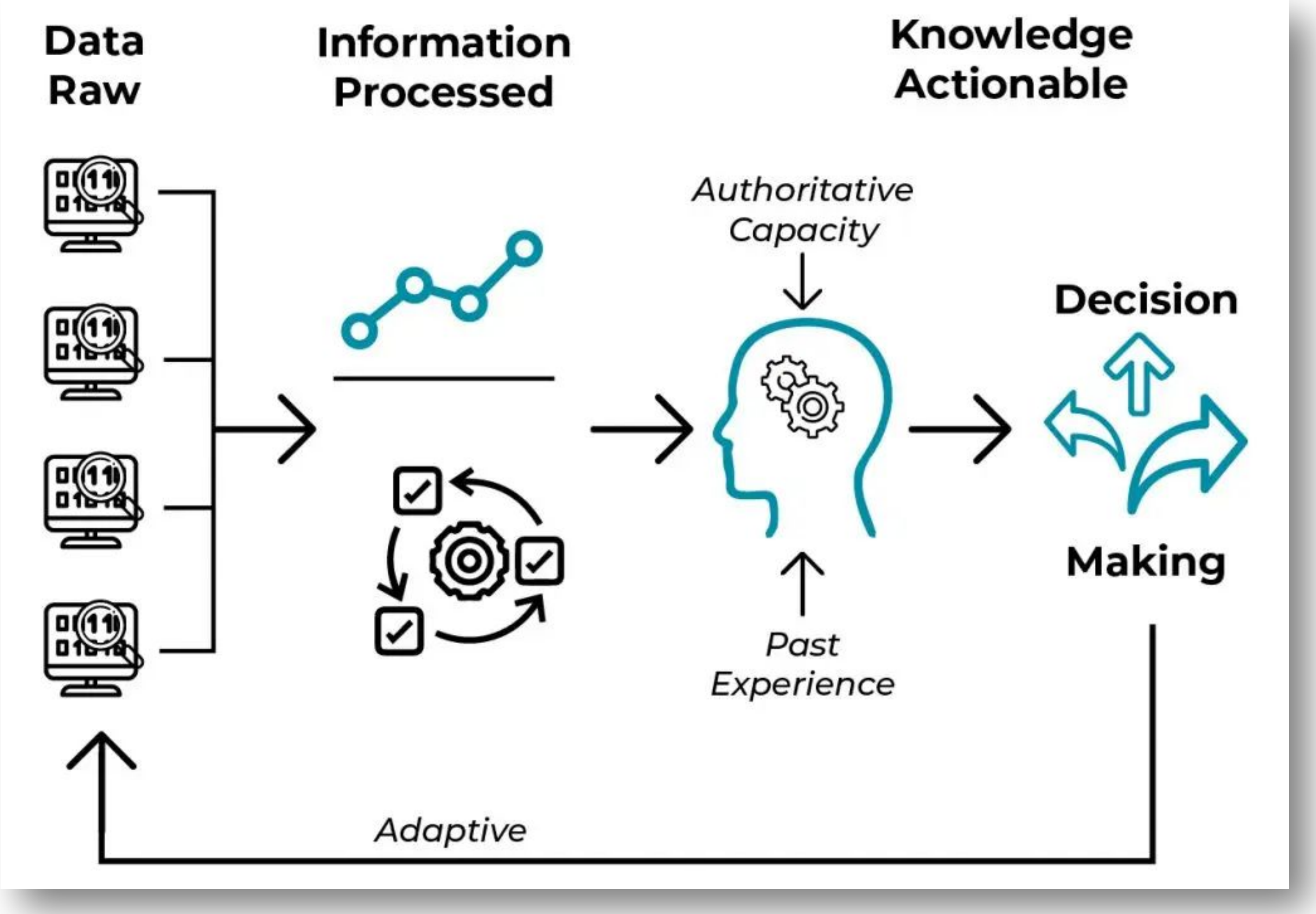
UNIT 5: Knowledge & Reasoning

Part - A

Knowledge Representation

What makes humans different from other animals or machines is our conscience

- **Conscience**: a person's moral sense of right and wrong, viewed as acting as a guide to one's behavior.
- Sum of our **memories**, i.e., all the knowledge we have gathered so far.
- This knowledge makes different personalities and makes humans **behave** differently and take different **actions.**
- To make AI more sophisticated, complex information about world requires, which leads to the concept of **Knowledge Representation** in Artificial Intelligence.



Data, Information and Knowledge

- **DATA** : Primitive verifiable facts

  E.g. name of novels available in a library

- **INFORMATION** : Analyzed Data

  E.g. the novel that is frequently asked by the member of library is “Harry Potter and the chamber of secrets”.

- **KNOWLEDGE**: Analyzed information that is often used for further information deduction.

  E.g. since the librarian knows the name of the novel that is frequently asked by members, s/he will ask for more copies of the novel the next time s/he place an order.

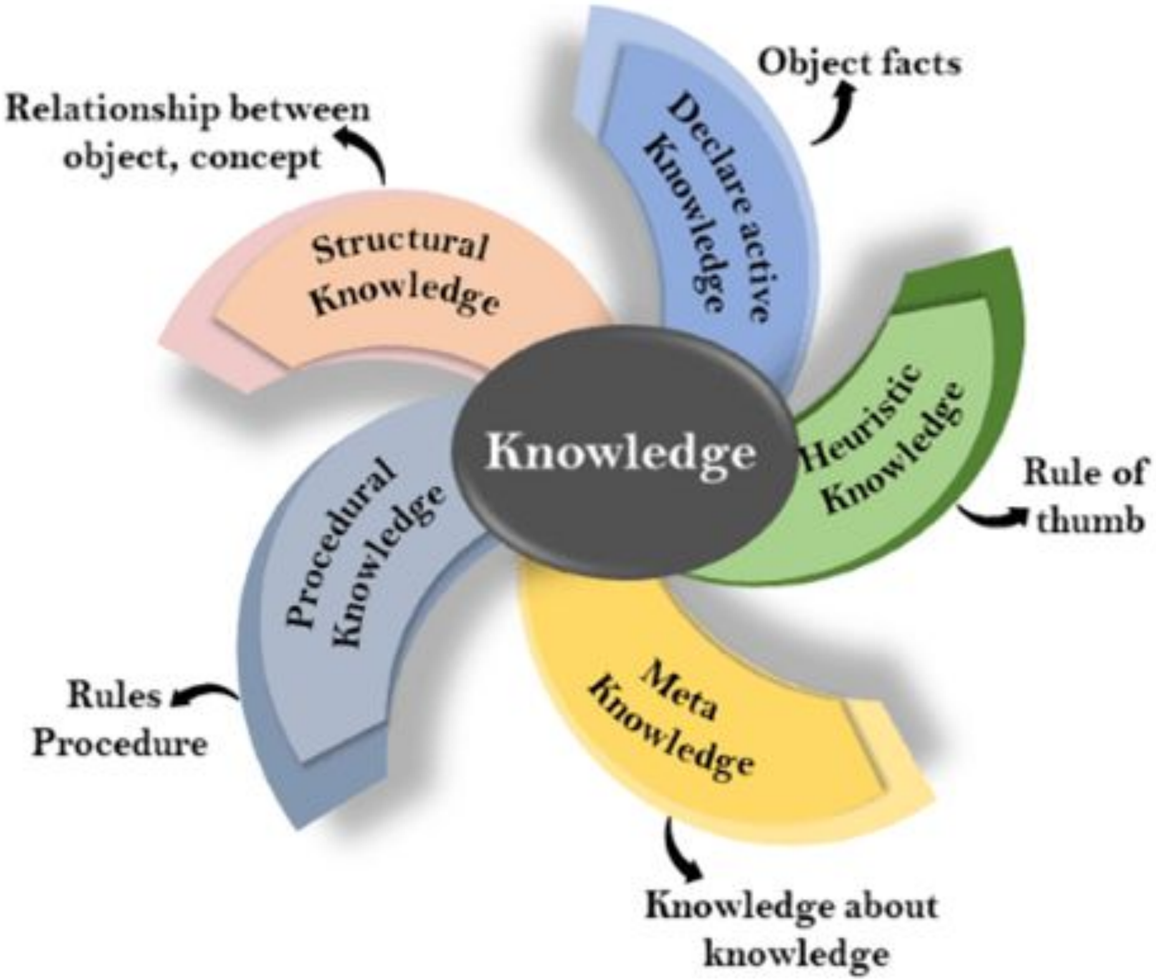
What is Knowledge Representation?

- Knowledge Representation in Artificial Intelligence refers to that concept where **ways are identified** to provide machines with the knowledge so that AI systems can act intelligently.
- We have concepts that are completely unknown to the machine such as intuition, intentions, prejudices, beliefs, judgments, common sense, etc.
- Some knowledge is straight forward such as 3+4=7 while others are more complex like shot a basketball into the hoop.
- 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.

What is to be presented?

- **Objects**: all facts about the object, e.g. cars have wheels, piano has keys
- **Events**: the actions which occur, e.g. war, achievements, advancement
- **Performance**: how to perform actions in different situations
- **Facts**: factual description of the world (truths about the real world)
- **Meta-Knowledge**: knowledge of what we know
- **Knowledge-base**: group of information regarding any discipline, field, e.g., knowledge-base for road construction

Types of Knowledge



Declarative Knowledge – It includes concepts, facts, and objects and expressed in a declarative sentence.

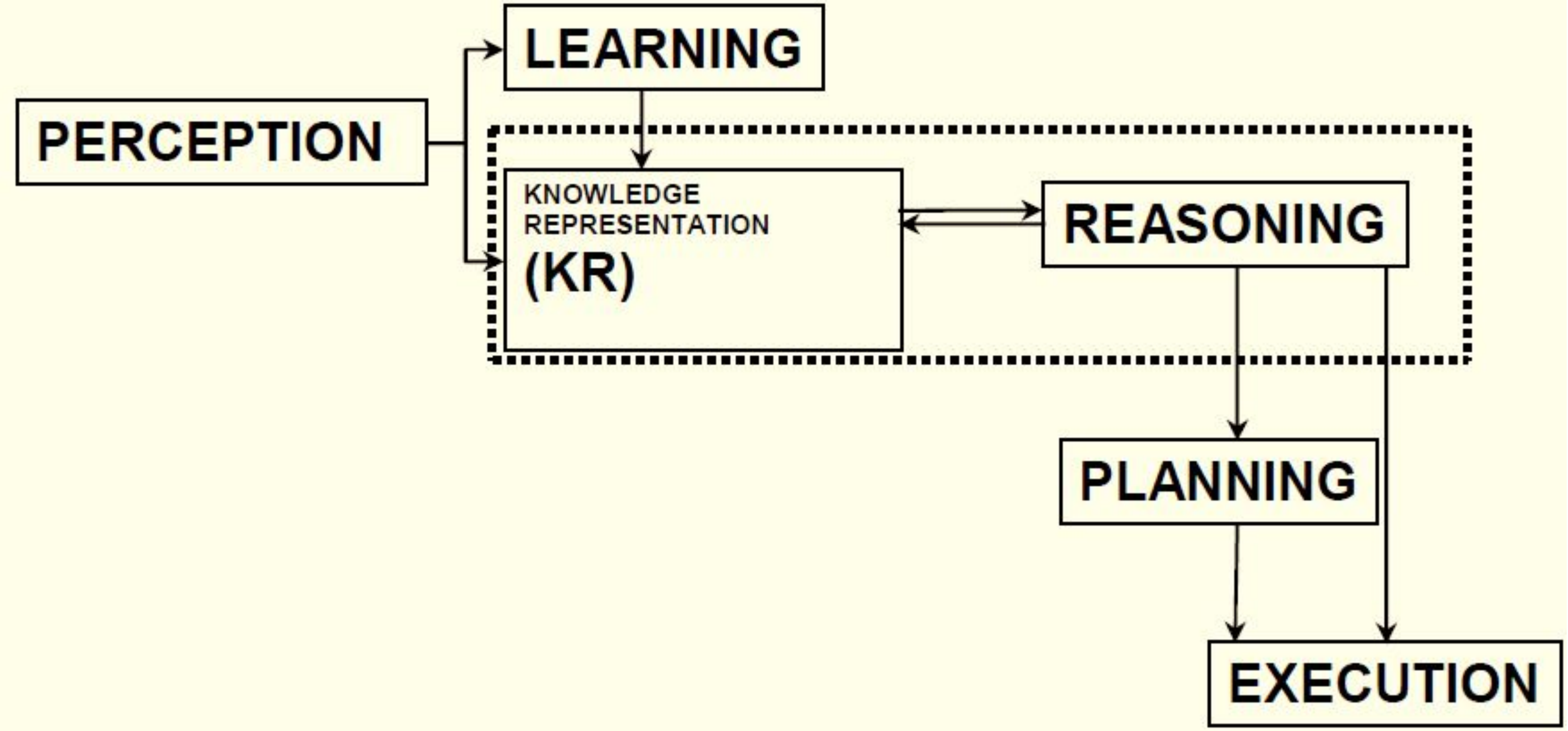
Structural Knowledge – It is a basic problem-solving knowledge that describes the relationship between concepts and objects.

Procedural Knowledge – This is responsible for knowing how to do something and includes rules, strategies, procedures, etc.

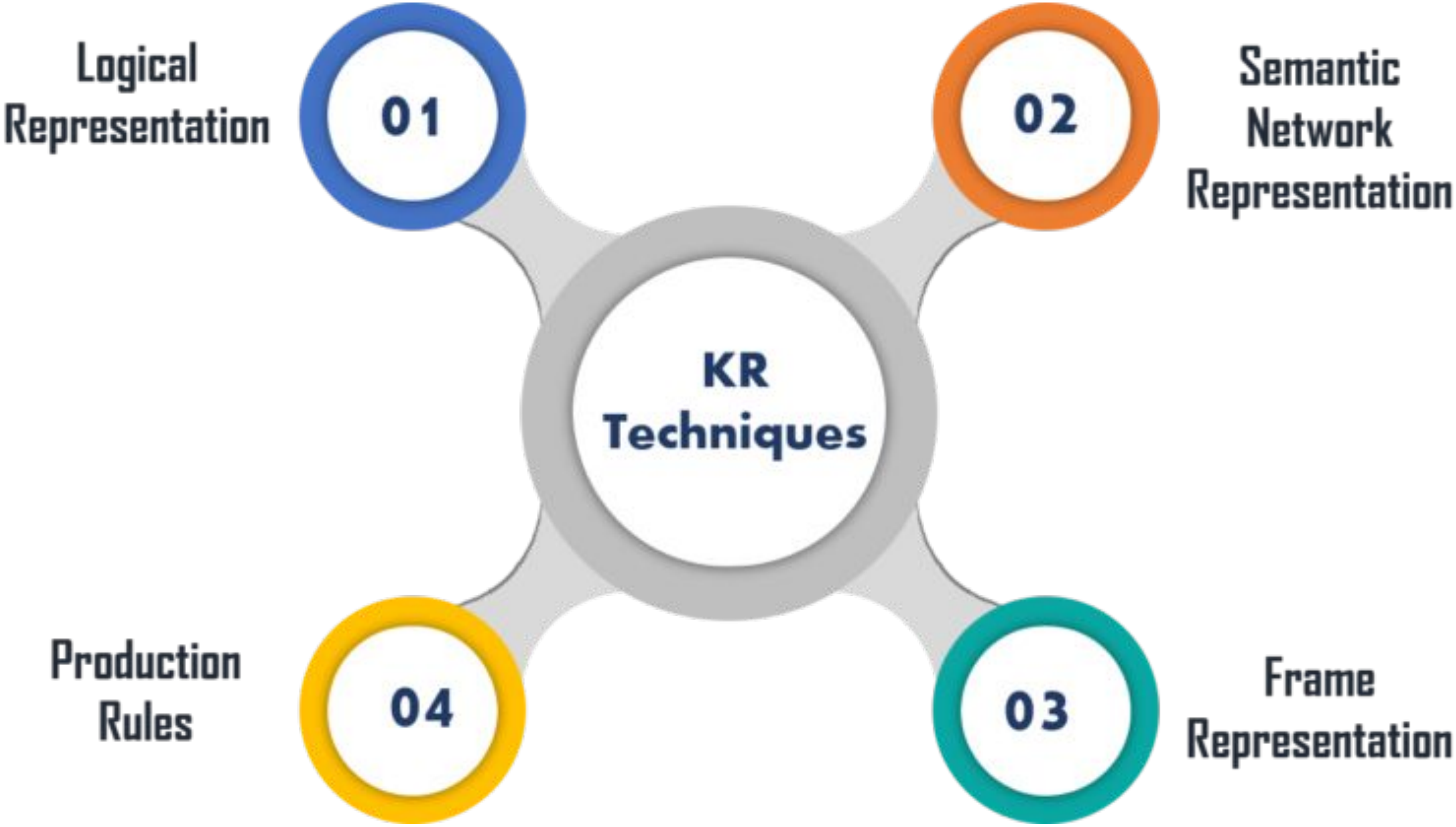
Meta Knowledge – Meta Knowledge defines knowledge about other types of Knowledge.

Heuristic Knowledge – This represents some expert knowledge in the field or subject.

Knowledge Representation in AI Cycle



Techniques of Knowledge Representation in AI

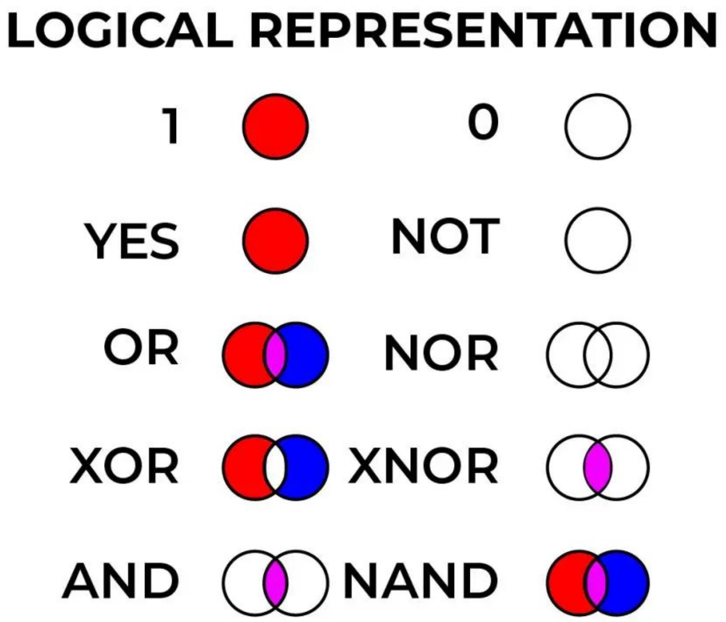


1. Logical Representation

- well-defined syntax with proper rules
- no ambiguity

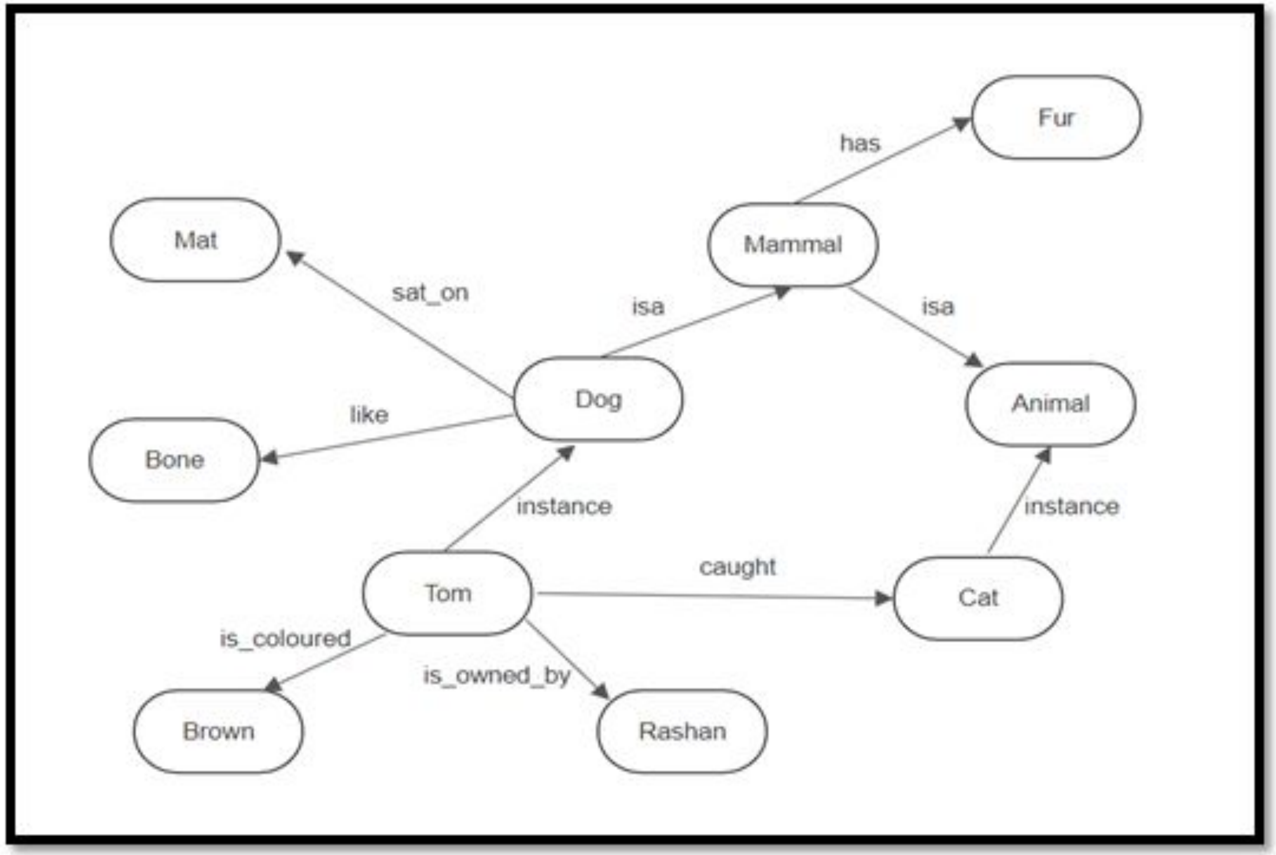
Logical Representation can be of two types-

- **Propositional Logic**
  - ✔ Works in a Boolean, i.e., True or False method
- **Predicate Logic** (First-order Logic)
  - ✔ represents the objects in quantifiers and predicates
  - ✔ advanced version of propositional logic



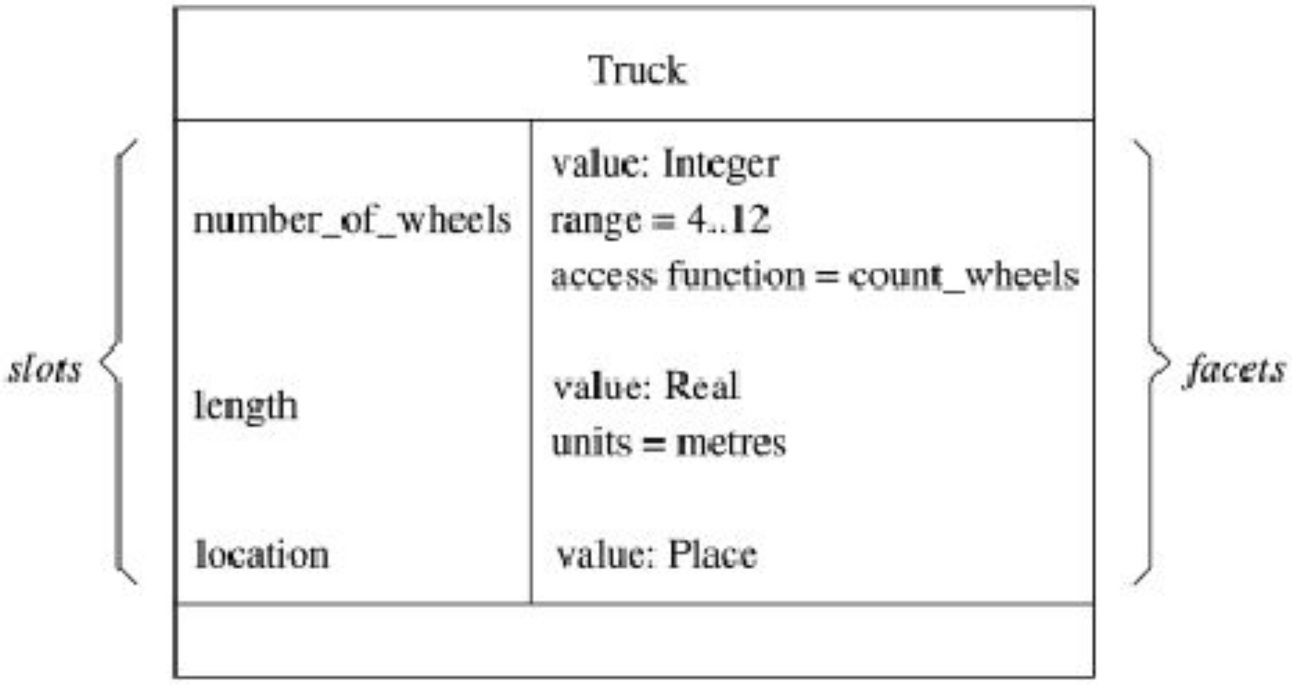
2. Semantic Networks

- Mathematically a semantic net can be defined as a **labeled directed graph**.
- How the objects are **connected**
- Semantic net allows us to perform inheritance reasoning as all members of a class will **inherit** all the properties of superclass
- **Is-A** relationship
- Ex. Google knowledge graph



3. Frame Representation

- A **record like structure** - consists of a collection of attributes and its values to describe an entity in the world.
- AI data structure - which divides knowledge into substructures by representing stereotypes situations.
- Consists of a collection of **slots** and slot values. These slots may be of any type and sizes.
- **Slots have names and values which are called facets**.

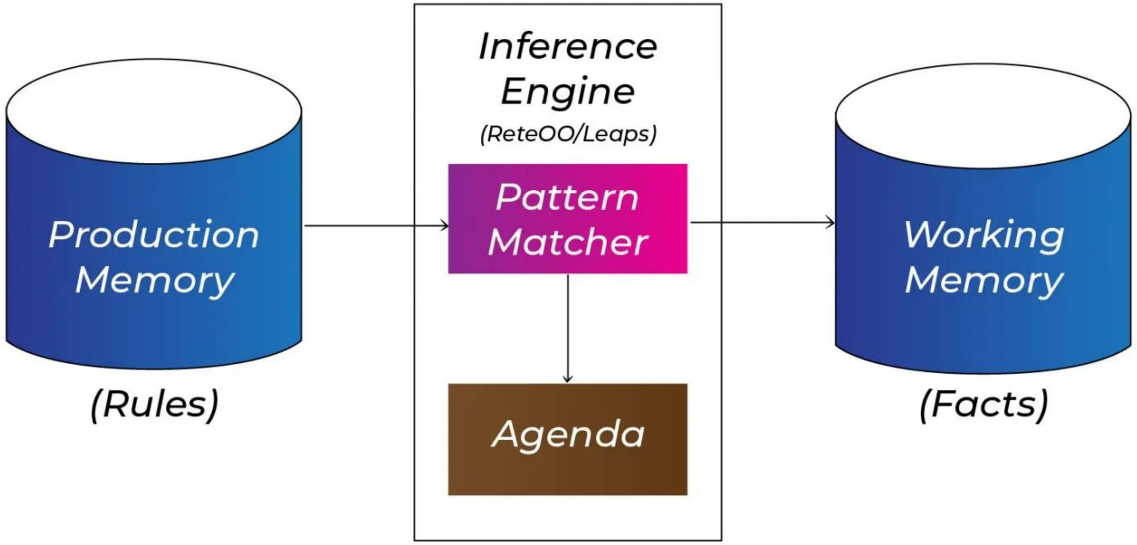


Truck	
number_of_wheels	value: Integer range = 4..12 access function = count_wheels
length	value: Real units = metres
location	value: Place

4. Production Rules

- Production rules system consist of (condition, action) pairs which mean, "If condition then action". It has mainly **three** parts:

1. The set of production rules
2. Working Memory
3. The recognize-act-cycle



E.g. water-jug problem

Summary: Knowledge Representation schemes

Logical schemes

– Predicate calculus

– Propositional calculus

Networked schemes

– Semantic nets

– Conceptual graphs

Procedural schemes

– IF..THEN..

Logical schemes

Logic, Propositional Logic, Predicate Logic

Logic

Logic is a formal language for expressing knowledge and ways of reasoning.

Logic is defined by:

- **A set of sentences**

– A sentence is constructed from a set of primitives according to **syntax** rules.

- **A set of interpretations**

– An interpretation gives a **semantic** to primitives. It associates primitives with values.

- **The valuation (meaning) function V**

– Assigns a value (typically the truth value) to a given sentence under some interpretation

$$V : \text{sentence} \times \text{interpretation} \rightarrow \{True, False\}$$

Language of numerical constraints:

- **A sentence:**

x + 3 <= z (where x, z are variable symbol)

- **An interpretation:**

I : x = 5 , z = 2

variables mapped to specific real numbers

- **Valuation (meaning) function V:**

v(x+3 <= z, I) is false for I: x=5 , z = 2

is True for I: x = 5 , z =10

Propositional Logic

True or False but not BOTH

Basics of Propositional Logic

- There are **four** different types of sentences.
- These sentences help us to define propositional logic.

  Declarative sentences assert or declare something, like “Richmond is the capital of Virginia.”

  Exclamatory sentences are emotional expressions, such as “watch out!”

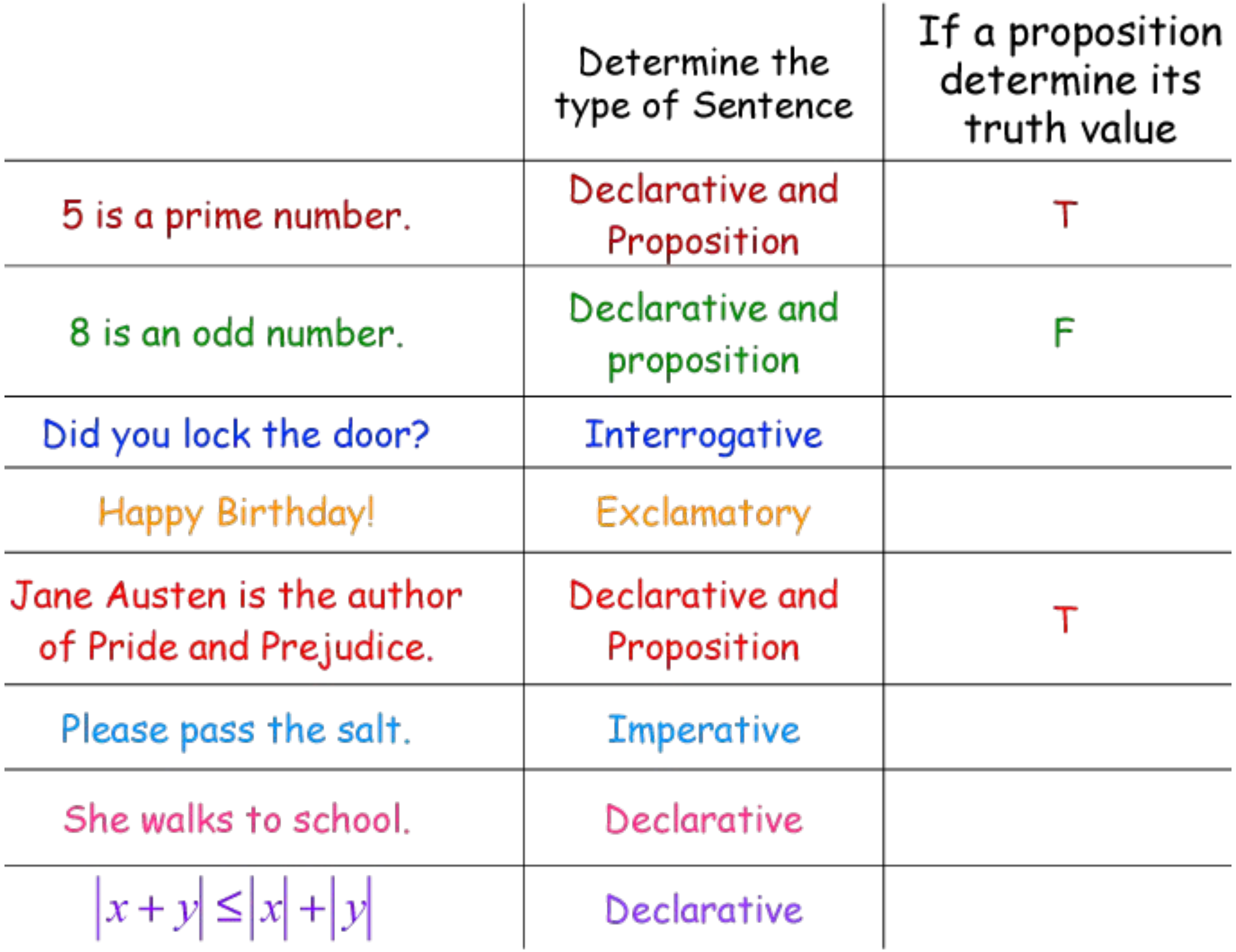
  Interrogative sentences ask questions, like “what time is it?”

  Imperative sentences give commands, such as “turn right at the traffic light.”

- And the sentences we are most interested in are **declarative!**
- Because propositions, also called statements, are declarative sentences that are either true or false, but not both.
- This means that every proposition is either true (T) or false (F).

Propositions Examples

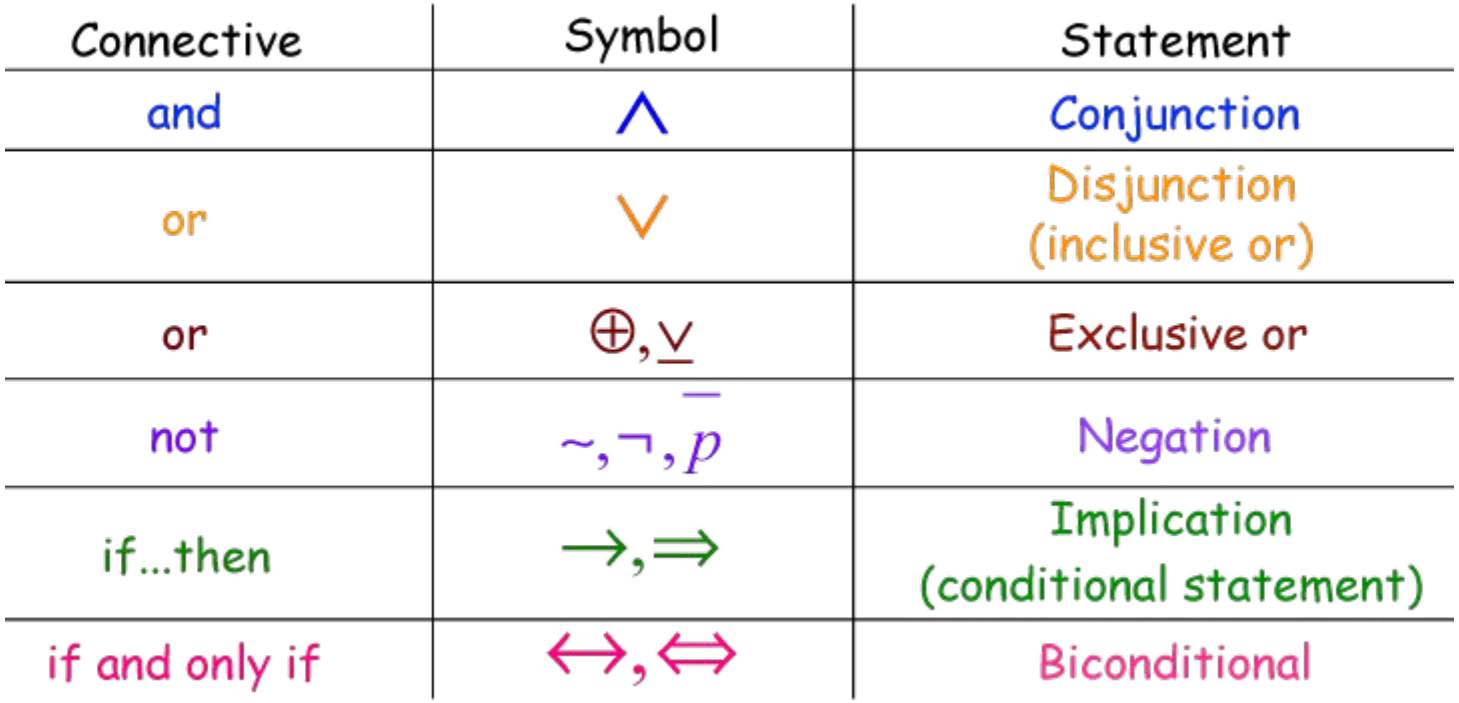
- Let’s look at a few examples of how we determine the type of sentence illustrated, and if it is a proposition, we will identify its truth value.
- Notice for our last two examples, that while the sentences are declarative, they are not a proposition because we don’t know the value of “she” or “x” or “y” — hence, we are unable to determine the truth value for the sentence.
- These types of scenarios are called paradoxes and open sentences, respectively.



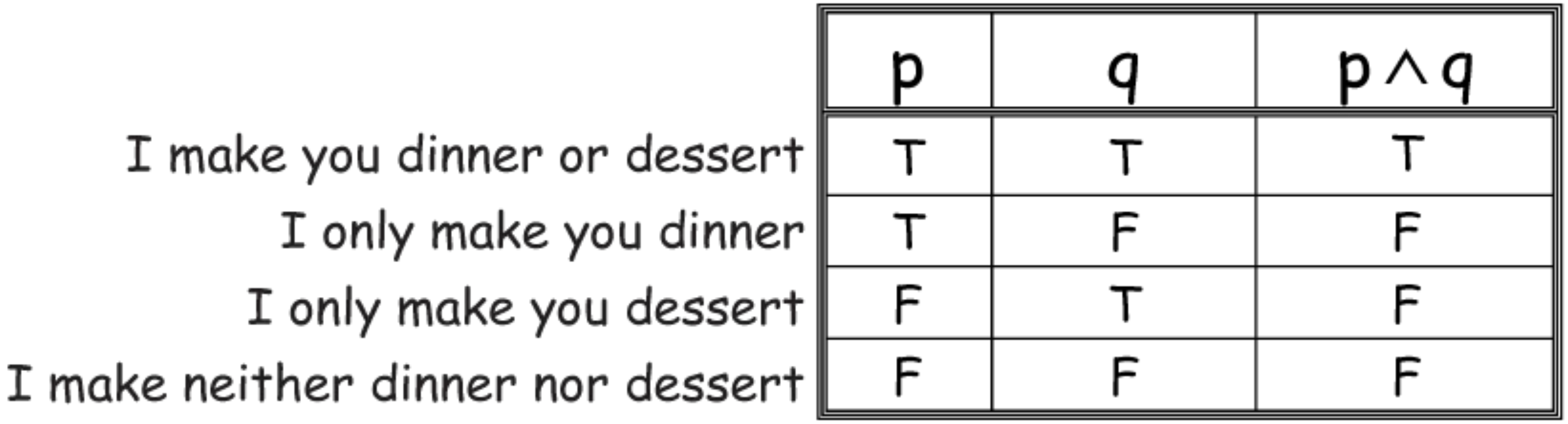
	Determine the type of Sentence	If a proposition determine its truth value
5 is a prime number.	Declarative and Proposition	T
8 is an odd number.	Declarative and proposition	F
Did you lock the door?	Interrogative	
Happy Birthday!	Exclamatory	
Jane Austen is the author of Pride and Prejudice.	Declarative and Proposition	T
Please pass the salt.	Imperative	
She walks to school.	Declarative	
$\|x+y\| \le \|x\| + \|y\|$	Declarative	

Compound Statement

- The combination of simple statements using logical connectives is called a **compound** statement
- The symbols we use to represent propositional variables and operations are called **symbolic logic**.

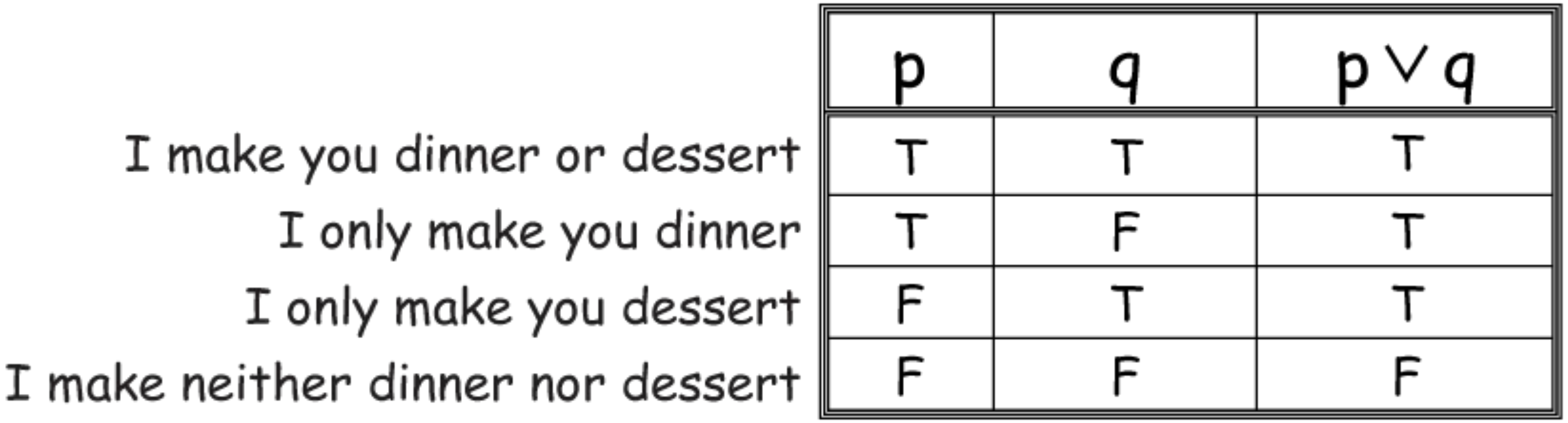


Connective	Symbol	Statement
and	$\wedge$	Conjunction
or	$\vee$	Disjunction (inclusive or)
or	$\oplus, \veebar$	Exclusive or
not	$\sim, \neg, \bar{p}$	Negation
if...then	$\rightarrow, \Rightarrow$	Implication (conditional statement)
if and only if	$\leftrightarrow, \Leftrightarrow$	Biconditional



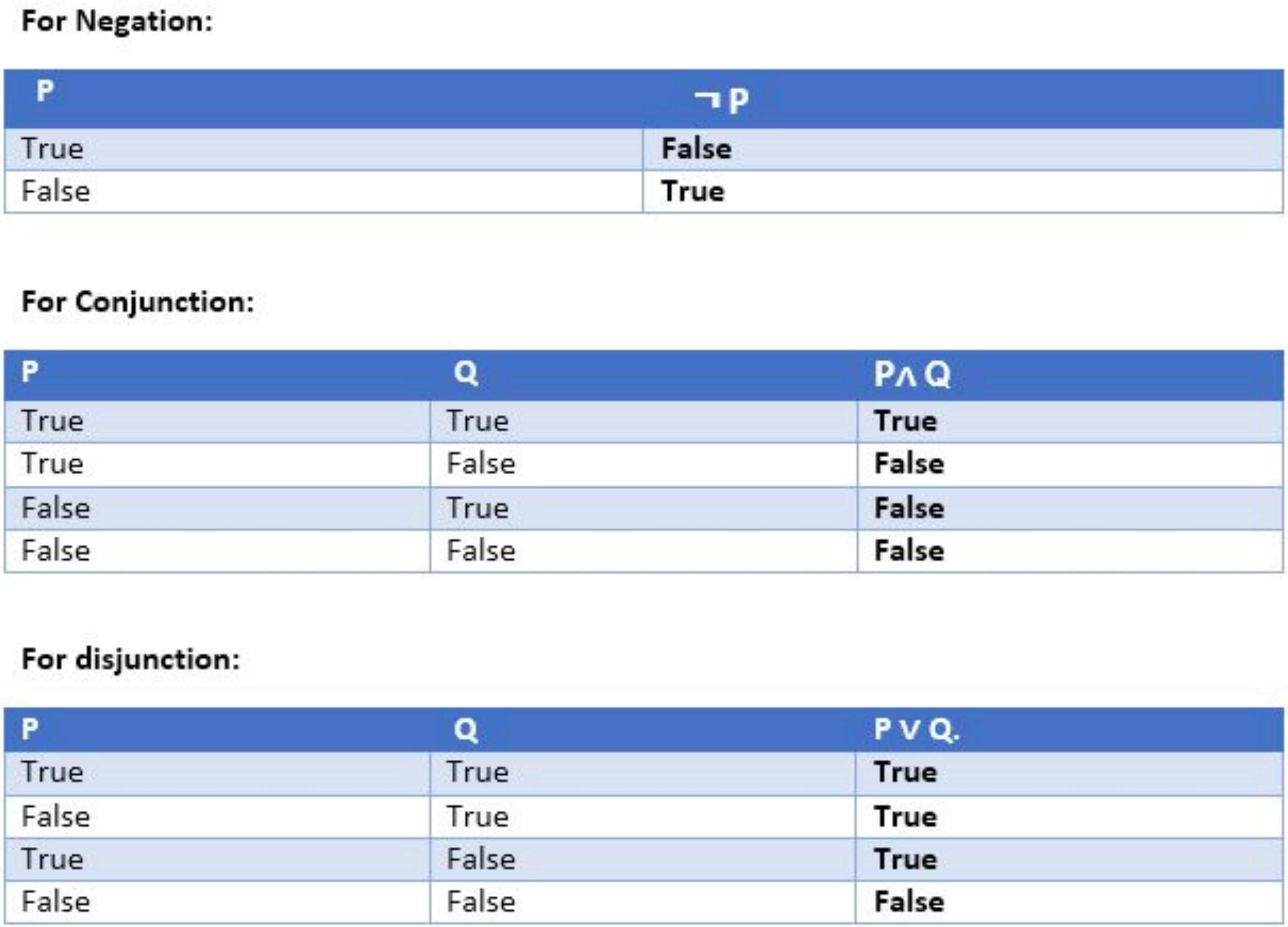
	p	q	$p \land q$
I make you dinner or dessert	T	T	T
I only make you dinner	T	F	F
I only make you dessert	F	T	F
I make neither dinner nor dessert	F	F	F

Conjunction



	p	q	$p \lor q$
I make you dinner or dessert	T	T	T
I only make you dinner	T	F	T
I only make you dessert	F	T	T
I make neither dinner nor dessert	F	F	F

Logical Disjunction



For Negation:

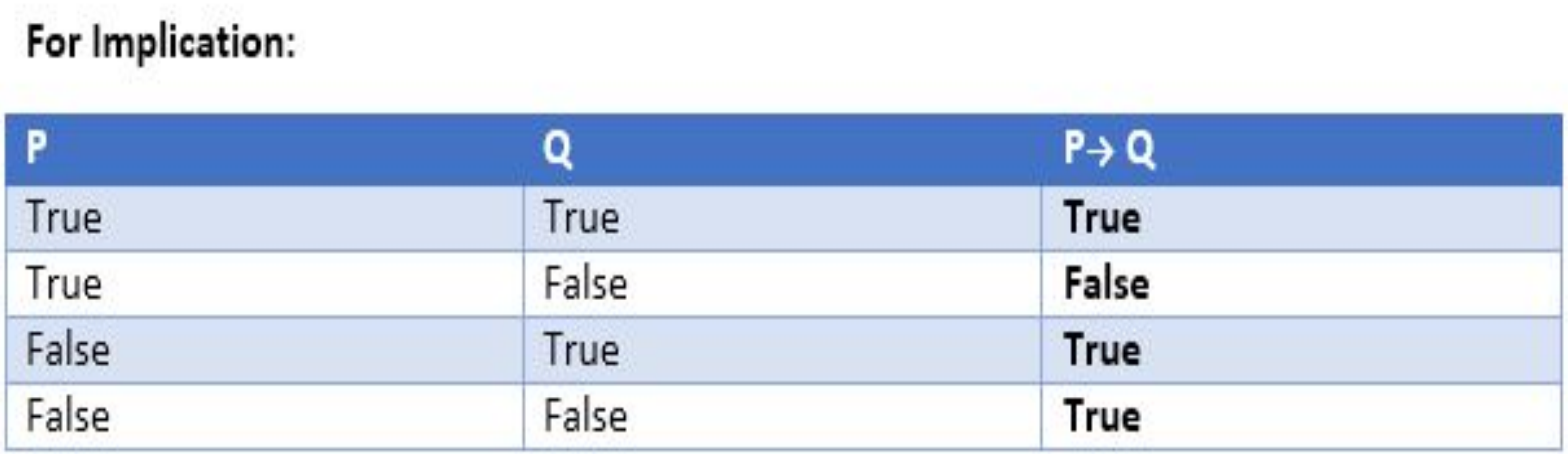
P	¬ P
True	**False**
False	**True**

For Conjunction:

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

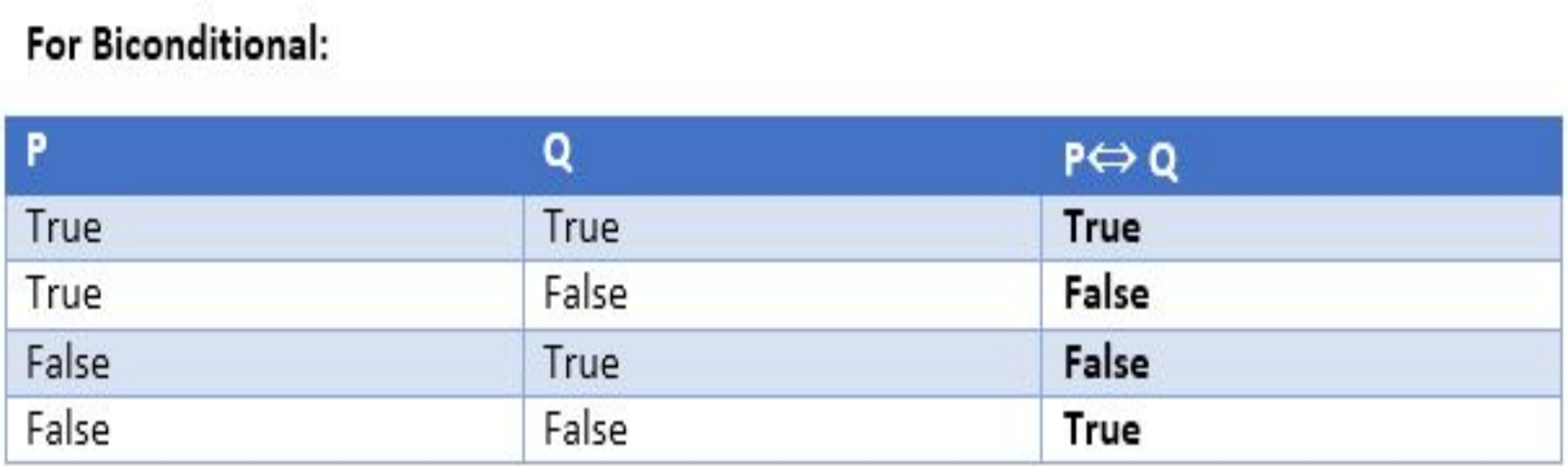
For disjunction:

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



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**

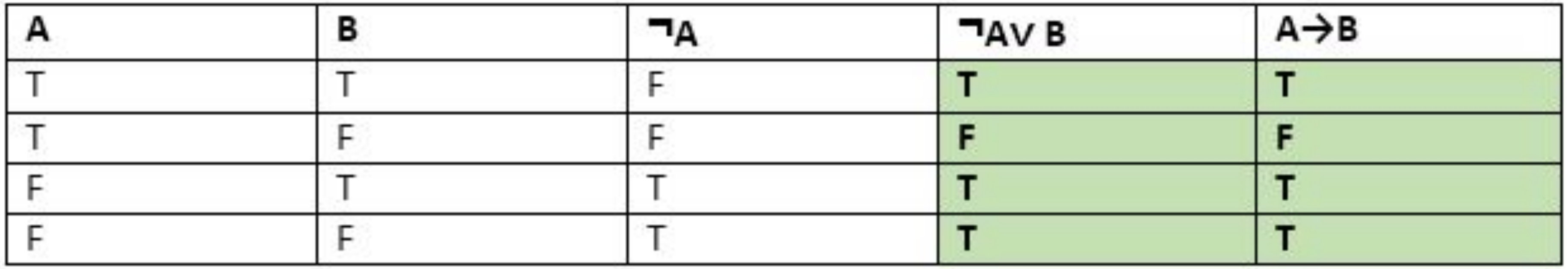
Precedence of connectives

- Just like arithmetic operators, there is a precedence order for propositional connectors or logical operators.
- This order should be followed while evaluating a propositional problem. Following is the list of the precedence order for operators:

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

Logical equivalence

- Logical equivalence is one of the features of propositional logic. Two propositions are said to be logically equivalent if and only if the columns in the truth table are identical to each other.
- Let's take two propositions A and B, so for logical equivalence, we can write it as A⇔B. In below truth table we can see that column for ¬A∨ B and A→B, are identical hence A is Equivalent to B



A	B	¬A	¬A∨ B	A→B
T	T	F	**T**	**T**
T	F	F	**F**	**F**
F	T	T	**T**	**T**
F	F	T	**T**	**T**

Properties of Operators

Commutativity:

P ∧ Q = Q ∧ P, or

P ∨ Q = Q ∨ P.

Associativity:

(P ∧ Q) ∧ R = P ∧ (Q ∧ R),

(P ∨ Q) ∨ R = P ∨ (Q ∨ R)

Identity element:

P ∧ True = P,

P ∨ True = True.

Distributive:

P ∧ (Q ∨ R) = (P ∧ Q) ∨ (P ∧ R).

Limitations of Propositional logic

1. We cannot represent relations like ALL, some, or none with propositional logic.
   Example:
   All the students are intelligent.
   Some apples are sweet.

2. Propositional logic has limited expressive power.

3. In propositional logic, we cannot describe statements in terms of their properties or logical relationships.

Basic facts about propositional logic

- Propositional logic is also called **Boolean** logic as it works on 0 and 1.
- In propositional logic, we 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.
- Propositional logic 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**.

Inference

- 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.

- Terminologies related to inference rules

  Implication: It is one of the logical connectives which can be represented as **P → Q**. It is a Boolean expression.

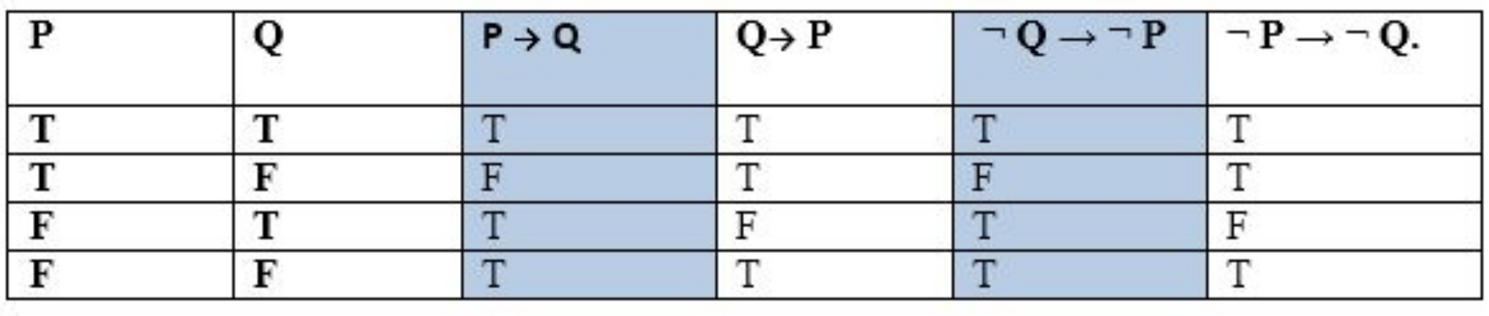
  Converse: The converse of implication, which means the right-hand side proposition goes to the left-hand side and vice-versa. It can be written as **Q → P**.

  Contrapositive: The negation of converse is termed as contrapositive, and it can be represented as **¬ Q → ¬ P**.

  Inverse: The negation of implication is called inverse. It can be represented as **¬ P → ¬ Q**.

Inference – Cont’d

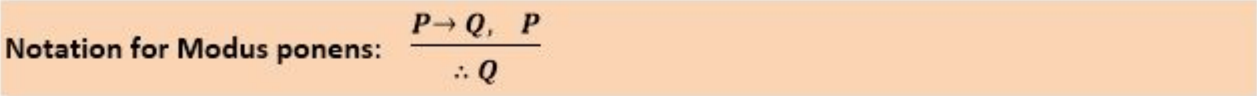
- Hence from the above truth table, we can prove that P → Q is equivalent to ¬ Q → ¬ P, and Q→ P is equivalent to ¬ P → ¬ Q.



P	Q	P → Q	Q→ P	¬ Q → ¬ P	¬ P → ¬ 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** (or Implication Elimination or Affirming the Antecedent), in Propositional Logic, is a rule of inference. It can be summarized as “P implies Q and P is asserted to be true, therefore Q must be true.”



Notation for Modus ponens: $\dfrac{P \rightarrow Q,\quad P}{\therefore Q}$

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.

An example of an argument that fits the form modus ponens:

- If today is Tuesday, then John will go to work.
- Today is Tuesday.
- Therefore, John will go to work.

2. Modus Tollens:

1. You can't log into the network
2. If you have a current password, then you can log into the network

Therefore, You don't have a current password.

This is an argument of the form:

- $\neg q$
- $p \rightarrow q$
- $\therefore \neg p$

Compare this with modes ponens

1. If you have a current password, then you can log on to the network
2. You have a current password

Therefore: You can log on to the network

This has the form:

- $p \rightarrow q$
- p
- $\therefore q$

Modes Pones vs. Modes Tollens

To help you understand good and bad way of logical constructions, here are some examples. The basic ideas are:

- There are two **consistent** logical argument constructions: modus ponens ("the way that affirms by affirming") and modus tollens ("the way that denies by denying").
  - **Modus Ponens:** "If A is true, then B is true. A is true. Therefore, B is true."
  - **Modus Tollens:** "If A is true, then B is true. B is not true. Therefore, A is not true."
- There are two related **incorrect** and **inconsist** constructions: affirming the consequent and denying the antecedent.
  - Affirming the Consequent: "If A is true, then B is true. B is true. Therefore, A is true."
  - Denying the Antecedent: "If A is true, then B is true. A is not true. Therefore, B is not true.“

- "If it is a car, then it has wheels. It is a car. Therefore, it has wheels." (Modus Ponens - CORRECT)
- "If it is a car, then it has wheels. It does not have wheels. Therefore, it is not a car." (Modus Tollens - CORRECT)
- "If it is a car, then it has wheels. It has wheels. Therefore, it is a car." (Affirming the Consequent - INCORRECT.)
- "If it is a car, then it has wheels. It is not a car. Therefore, it does not have wheels." (Denying the Antecedent - INCORRECT)

Examples:

- "If Xyrplex is 9, Guffaw is 1. Guffaw is 2. Therefore, Xyrplex is not 9.“

  (ANSWER: Modus Tollens)

- "If Nagini is a Snake, Snape is a goner. Nagini is a snake. Therefore, Snape is a goner."

  (ANSWER: Modus Ponens)

- "If Blurts are Flurts, Green is Grue. Green is Grue. Therefore, Blurts are Flurts."

  (ANSWER: INCORRECT – Affirming the Consequent)

- "If Sagan has hair, Tyson is awesome. Sagan has hair. Therefore, Tyson is awesome.“

  (ANSWER: Modus Ponens)

- "If Fordham brings a ram, Peruna will kick. Fordham did not bring a ram. Therefore, Peruna did not kick."

  (ANSWER: INCORRECT – Denying the Antecedent)

3. Hypothetical Syllogism:

- **Syllogism** : an instance of a form of reasoning in which a conclusion is drawn from two given or assumed propositions (premises); a common or middle term is present in the two premises but not in the conclusion, which may be invalid (e.g. *all dogs are animals; all animals have four legs; therefore all dogs have four legs*).

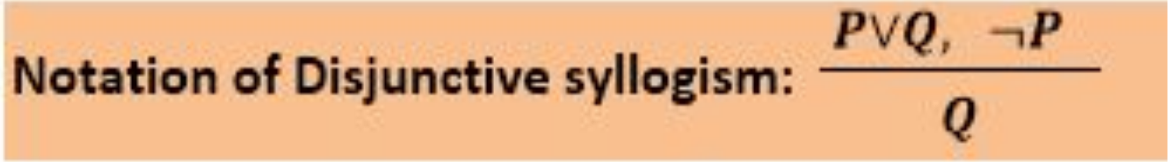
- The **Hypothetical Syllogism** rule state that if P→R is true whenever P→Q is true, and Q→R is true.

Example:

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

4. Disjunctive Syllogism:

- The Disjunctive syllogism rule state that if P$\vee$Q is true, and ¬P is true, then Q will be true. It can be represented as:



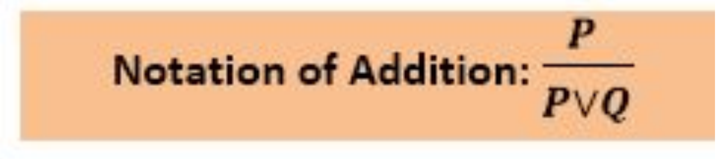
Notation of Disjunctive syllogism: $\dfrac{P\vee Q,\ \ \neg P}{Q}$

Example:

- Statement-1: Today is Sunday or Monday. $\Rightarrow$ P$\vee$Q
- Statement-2: Today is not Sunday. $\Rightarrow$ ¬P
- Conclusion: Today is Monday. $\Rightarrow$ Q

5. Addition:

- The Addition rule is one the common inference rule, and it states that If P is true, then P ∨ Q will be true.



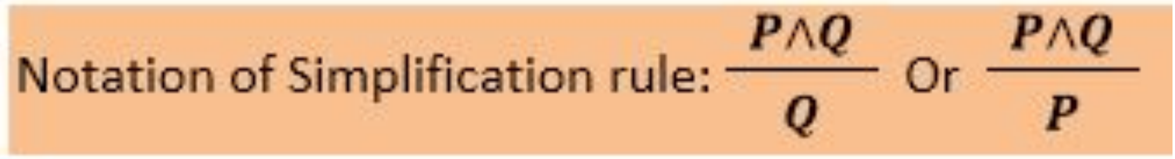
Notation of Addition: $\frac{P}{P \vee Q}$

Example:

- Statement-1: I have a vanilla ice-cream. ⇒ P
- Statement-2: I have Chocolate ice-cream. ⇒ Q
- Conclusion: I have vanilla or chocolate ice-cream. ⇒ (P ∨ Q)

6. Simplification:

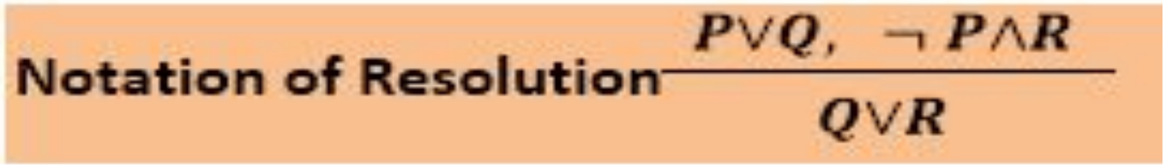
- The simplification rule state that if P ∧ Q is true, then Q or P will also be true. It can be represented as:



Notation of Simplification rule: $\frac{P \land Q}{Q}$ Or $\frac{P \land Q}{P}$

7. Resolution:

- The Resolution rule state that if P ∨ Q and ¬ P ∧ R is true, then Q ∨ R will also be true. It can be represented as



Notation of Resolution $\frac{P \lor Q, \ \neg P \land R}{Q \lor R}$

Translation

Assume the following sentences:

- It is not sunny this afternoon and it is colder than yesterday.
- We will go swimming only if it is sunny.
- If we do not go swimming then we will take a canoe trip.
- If we take a canoe trip, then we will be home by sunset.

Denote:

- p = It is sunny this afternoon
- q = it is colder than yesterday
- r = We will go swimming
- s= we will take a canoe trip
- t= We will be home by sunset

Translation

Assume the following sentences:

- It is not sunny this afternoon and it is colder than yesterday.

  $\neg p \land q$

- We will go swimming only if it is sunny.

  $r \rightarrow p$

- If we do not go swimming then we will take a canoe trip.

  $\neg r \rightarrow s$

- If we take a canoe trip, then we will be home by sunset.

  $s \rightarrow t$

- p = It is sunny this afternoon
- q = it is colder than yesterday
- r = We will go swimming
- s= we will take a canoe trip
- t= We will be home by sunset

Knowledge Representation and Reasoning

(putting all together)

Knowledge Representation & Reasoning

Knowledge representation is the study of how knowledge about the world can be represented and what kinds of reasoning can be done with that knowledge.

We will discuss two different systems that are commonly used to represent knowledge in machines and perform algorithmic reasoning:

- **Propositional calculus**
- **Predicate calculus**

Propositional Calculus

In propositional calculus,

- features of the world are represented by **propositions**,
- relationships between features (constraints) are represented by **connectives**.

Example:

LECTURE_BORING $\wedge$ TIME_LATE $\Rightarrow$ SLEEP

This expression in propositional calculus represents the fact that for some agent in our world, if the features LECTURE_BORING and TIME_LATE are both true, the feature SLEEP is also true.

Propositional Calculus

You see that the language of propositional calculus can be used to represent aspects of the world.

When there are

- a **language**, as defined by a syntax,
- **inference rules** for manipulating sentences in that language, and
- **semantics** for associating elements of the language with elements of the world,

then we have a system called **logic**.

The Language

Atoms (and atomic sentences):

The atoms T and F and all strings that begin with a capital letter, for instance, P, Q, LECTURE_BORING, and so on.

Literals: P and ¬ P

Connectives:

- $\vee$ “or”
- $\wedge$ “and”
- $\Rightarrow$ “implies” or “if-then”
- ¬ “not”

Sentences: can be formed using the connectives

The Language

Syntax of well-formed formulas (wffs):

- Any atom is a wff.
- If ω_1 and ω_2 are wffs, so are

  $\omega_1 \wedge \omega_2$ (conjunction)

  $\omega_1 \vee \omega_2$ (disjunction)

  $\omega_1 \Rightarrow \omega_2$ (implication)

  $\neg \omega_1$ (negation)
- There are no other wffs.

The Language

- Atoms and negated atoms are called **literals**.
- In $\omega_1 \Rightarrow \omega_2$, ω_1 is called the **antecedent**, and ω_2 is called the **consequent** of the implication.
- **Examples of wffs (sentences):**

  $(P \land Q) \Rightarrow \neg P$

  $P \Rightarrow \neg P$

  $P \lor P \Rightarrow P$

  $(P \Rightarrow Q) \Rightarrow (\neg Q \Rightarrow \neg P)$

  $\neg\neg P$

- The **precedence order** of the above operators is

  $\neg \quad \land \quad \lor \quad \Rightarrow$

  For example, $\neg P \lor Q \Rightarrow R$ means $((\neg P) \lor Q) \Rightarrow R$.

The Language

- **General Form:**

  $((q \wedge \neg r) \Rightarrow s) \quad \wedge \neg (s \wedge t)$

- **Conjunction Normal Form (CNF):**

  $(\neg q \vee r \vee s) \wedge (\neg s \vee \neg t)$

  Set Notation: $\{(\neg q, r, s), (\neg s, \neg t)\}$ □ set of clauses

  Empty clauses (): false

- **Binary Clauses: 1 or 2 literals per clause**

  $(\neg q \vee r) \qquad (\neg s \vee \neg t)$

- **Horn Clauses: 0 or 1 positive literals per clause**

  $(\neg q \vee \neg r \vee s) \qquad (\neg s \vee \neg t)$

Rules of Inference

We use **rules of inference** to generate new wffs from existing ones.

One important rule is called **modus ponens** or the **law of detachment**. It is based on the tautology $(P \wedge (P \Rightarrow Q)) \Rightarrow Q$. We write it in the following way:

$$\begin{array}{l} P \\ P \Rightarrow Q \\ \hline \therefore\ Q \end{array}$$

The two **hypotheses** P and P ⇒ Q are written in a column, and the **conclusion** below a bar, where ∴ means "therefore".

Rules of Inference

$$\begin{array}{l} P \\ \hline \therefore P \lor Q \end{array}$$

Addition

$$\begin{array}{l} P \land Q \\ \hline \therefore P \end{array}$$

Simplification

$$\begin{array}{l} P \\ Q \\ \hline \therefore P \land Q \end{array}$$

Conjunction

$$\begin{array}{l} \neg Q \\ P \Rightarrow Q \\ \hline \therefore \neg P \end{array}$$

Modus tollens

$$\begin{array}{l} P \Rightarrow Q \\ Q \Rightarrow R \\ \hline \therefore P \Rightarrow R \end{array}$$

Hypothetical syllogism

$$\begin{array}{l} P \lor Q \\ \neg P \\ \hline \therefore Q \end{array}$$

Disjunctive syllogism

Proofs

The sequence of wffs $\{\omega_1, \omega_2, \ldots, \omega_n\}$ is called a **proof** (or a **deduction**) of ω_n from a set of wffs Δ iff (if and only if) each ω_i in the sequence is either in Δ or can be inferred from one or more wffs earlier in the sequence by using one of the **rules of inference**.

If there is a proof of ω_n from Δ, we say that ω_n is a **theorem** of the set Δ. We use the following notation:

$\Delta \vdash \omega_n$

In this notation, we can also indicate the set of inference rules $\mathcal{R}$ that we use:

$\Delta \vdash_{\mathcal{R}} \omega_n$

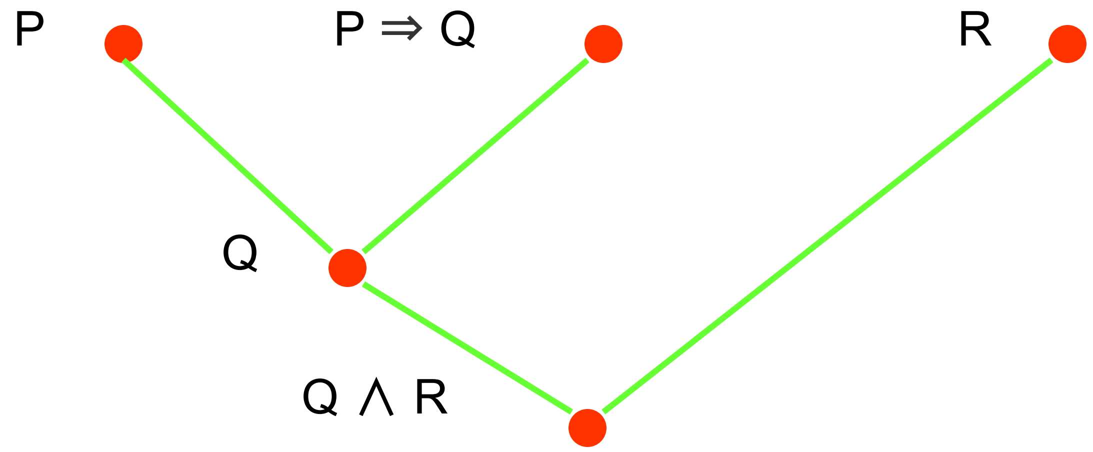
Proofs

Example:

Given a set of wffs $\Delta = \{P, R, P \Rightarrow Q\}$, the following sequence is a proof of $Q \wedge R$ given the inference rules that we discussed earlier:

$\{P, P \Rightarrow Q, Q, R, Q \wedge R\}$

Tree representation:



Semantics

- In propositional logic, we **associate** atoms with propositions about the world.
- We thereby specify the **semantics** of our logic, giving it a **“meaning”**.
- Such an association of atoms with propositions is called an **interpretation**.
- In a given interpretation, the proposition associated with an atom is called the **denotation** of that atom.
- Under a given interpretation, atoms have values – **True** or **False**. We are willing to accept this idealization (otherwise: **fuzzy logic**).

Semantics

Example:

"Sonali is either intelligent or a good actor.
If Sonali is intelligent, then she can count from 1 to 10.
Sonali can only count from 1 to 2.
Therefore, Sonali is a good actor."

Propositions:

I: "Sonali is intelligent."
A: "Sonali is a good actor."
C: "Sonali can count from 1 to 10."

Step 1: $\neg$ C Hypothesis
Step 2: I $\Rightarrow$ C Hypothesis
Step 3: $\neg$ I Modus Tollens Steps 1 & 2
Step 4: A $\vee$ I Hypothesis
Step 5: A Disjunctive Syllogism Steps 3 & 4

Conclusion: **A** ("Sonali is a good actor.")

Semantics

Let us consider an **agent** which cause action when it gets some input from the sensors

- The sensors inform the agent about a **set of features** of the outside world.
- We can **associate propositions** with these features such as "There is a wall in the cell on the right hand side."
- The sensors tell the agent whether each of these propositions is currently true or false.
- This propositional information that is available to the agent is called its **knowledge base**.

Semantics

- We say that an interpretation **satisfies** a wff if the wff is assigned the value **True** under that interpretation.
- An interpretation that satisfies a wff is called a **model** of that wff.
- An interpretation that satisfies each wff in a **set of wffs** is called a model of that set.
- The **more wffs** we have that describe the world, the **fewer models** there are.
- This means that the more we know about the world, the less uncertainty there is.

Semantics

- If no interpretation satisfies a wff (or a set of wffs), the wff is said to be **inconsistent** or **unsatisfiable**, for example, $P \wedge \neg P$.
- A wff is said to be **valid** if it has value **True** under all interpretations of its constituent atoms, for example, $P \vee \neg P$.
- Neither valid wffs nor inconsistent wffs tell us anything about the world.

Semantics

- Two wffs are said to be **equivalent** if and only if their truth values are identical under all interpretations. For two equivalent wffs ω_1 and ω_2 we write $\omega_1 \equiv \omega_2$.
- If a wff ω has value True under all of those interpretations for which each of the wffs in a set Δ has value True, then we say that
  - Δ **logically entails** ω
  - ω **logically follows** from Δ
  - ω is a **logical consequence** of Δ
  - $\Delta \models \omega$

Soundness and Completeness

- **Soundness:** <u>if something is provable, it is valid.</u>
  - If, for any set of wffs Δ and wff ω, $\Delta \vdash_{\mathcal{R}} \omega$ implies $\Delta \models \omega$, we say that the set of inference rules $\mathcal{R}$ is **sound**.
- **Completeness:** <u>if something is valid, it is provable.</u>
  - If, for any set of wffs Δ and wff ω, it is the case that whenever $\Delta \models \omega$, there exists a proof of ω from Δ using the set of inference rules $\mathcal{R}$, we say that $\mathcal{R}$ is **complete**.
- When $\mathcal{R}$ is **sound and complete**, we can determine whether one wff follows from a set of wffs by searching for a **proof** instead of using a **truth table** (increased efficiency).

Resolution

- Multiple rules of inference can be combined into one rule, called **resolution**.
- A **clause** is a set of literals, which is a short notation for the disjunction of all the literals in the set. For example, the wff $\{P, Q, \neg R\}$ is the same as the wff $P \vee Q \vee \neg R$.
- The **empty clause** { } (or NIL) is equivalent to F.

Resolution

Resolution rule for the propositional calculus:

From $\{\lambda\} \cup \Sigma_1$ and $\{\neg \lambda\} \cup \Sigma_2$ (where Σ_1 and Σ_2 are sets of literals and λ is an atom), we can infer $\Sigma_1 \cup \Sigma_2$, which is called the **resolvent** of the two clauses.

The atom λ is the **atom resolved upon**, and the process is called **resolution**.

Examples:

- Resolving $R \vee P$ and $\neg P \vee Q$ yields $R \vee Q$.
  In other words: $\{R, P\}$, $\{\neg P, Q\}$ yields $\{R, Q\}$.
- Resolving R and $\neg R \vee P$ yields P.

  In other words: {R}, {¬R, P} yields {P}.

Resolution

And yet Another Example:

Resolving $P \lor Q \lor \neg R$ with $P \lor W \lor \neg Q \lor R$

- on Q yields $P \lor \neg R \lor R \lor W$, which is **True**.
- on R yields $P \lor Q \lor \neg Q \lor W$, which is also **True**.

You cannot resolve on Q and R simultaneously!

Note:

- Any **set of wffs** containing ω and $\neg \omega$ is unsatisfiable, i.e. if one wff is the negation of another wff in that set, it is impossible that all wffs in that set are true.
- Any **clause** that contains λ and $\neg \lambda$ has value **True** regardless of the value of λ.

Converting wffs to Conjunctions of Clauses

Resolution is a powerful tool for algorithmic inference, but we can only apply it to **conjunctions of clauses** (conjunctive normal form, CNF).

So is there a way to **convert** any wff into such a conjunction of clauses?

Fortunately, there is such a way, allowing us to apply resolution to **any** wff.

Converting wffs to Conjunctions of Clauses

Example: $\neg (P \Rightarrow Q) \vee (R \Rightarrow P)$.

Step 1: Eliminate implication operators:

$\neg (\neg P \vee Q) \vee (\neg R \vee P)$

Step 2: Reduce the scopes of ¬ operators by using DeMorgan’s laws and eliminating double ¬ operators:

$(P \wedge \neg Q) \vee (\neg R \vee P)$

Step 3: Convert to CNF by using the associative and distributive laws:

$(P \vee \neg R \vee P) \wedge (\neg Q \vee \neg R \vee P)$, and then

$(P \vee \neg R) \wedge (\neg Q \vee \neg R \vee P)$

Convert into CNF (1^{st} step in Resolution)

It is a canonical formula, that is, any logic can be converted into canonical formula.

$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

1. Eliminate $\Leftrightarrow$, replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$:

   $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
2. Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg \alpha \vee \beta$:

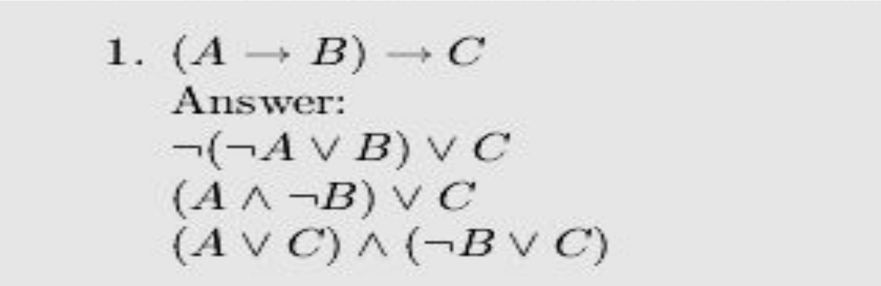
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg (P_{1,2} \vee P_{2,1}) \vee B_{1,1})$ i.e. part 1 is clause but part 2 is not
3. Move $\neg$ inwards using de Morgen's rules and double-negation:

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$
4. Apply distributive law ($\vee$ over $\wedge$):

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

Examples: Convert into CNF

1. $(A \rightarrow B) \rightarrow C$
2. $A \rightarrow (B \rightarrow C)$
3. $(A \rightarrow B) \vee (B \rightarrow A)$
4. $(\neg P \rightarrow (P \rightarrow Q))$



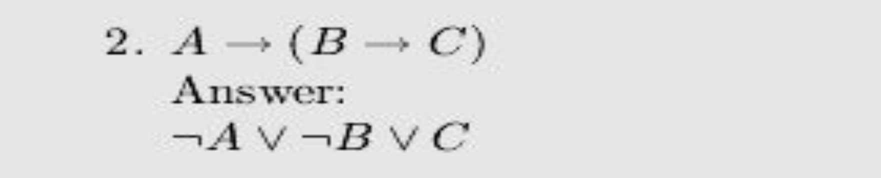
1. $(A \rightarrow B) \rightarrow C$

Answer:

$\neg(\neg A \vee B) \vee C$

$(A \wedge \neg B) \vee C$

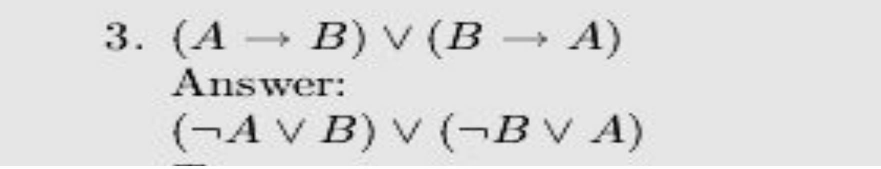
$(A \vee C) \wedge (\neg B \vee C)$



2. $A \rightarrow (B \rightarrow C)$

Answer:

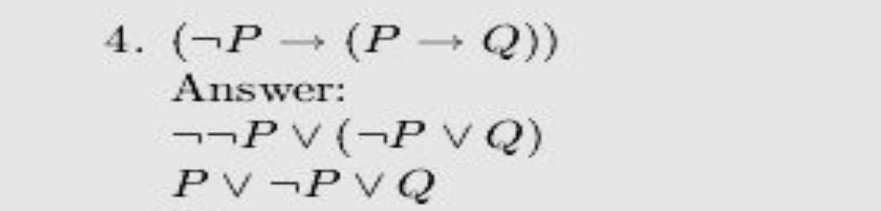
$\neg A \vee \neg B \vee C$



3. $(A \rightarrow B) \vee (B \rightarrow A)$

Answer:

$(\neg A \vee B) \vee (\neg B \vee A)$



4. $(\neg P \rightarrow (P \rightarrow Q))$

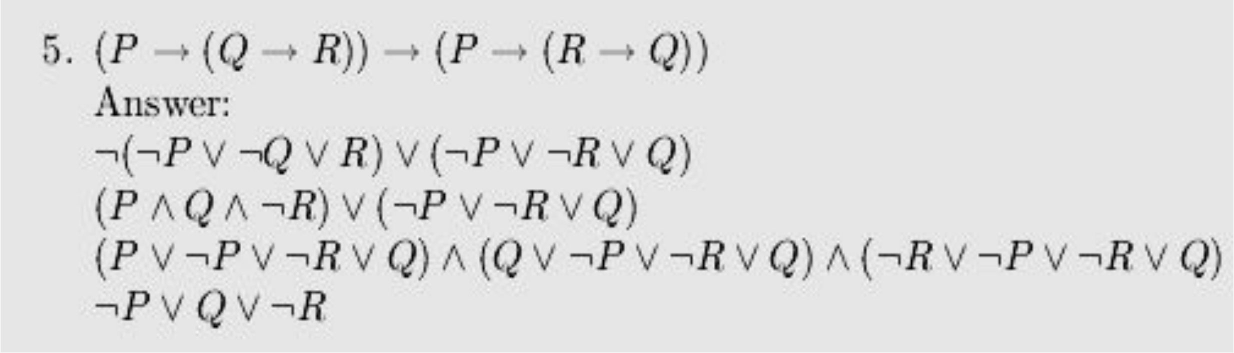
Answer:

$\neg\neg P \vee (\neg P \vee Q)$

$P \vee \neg P \vee Q$

More Examples: Convert into CNF

5. $(P \rightarrow (Q \rightarrow R)) \rightarrow (P \rightarrow (R \rightarrow Q))$
6. $(P \rightarrow Q) \rightarrow ((Q \rightarrow R) \rightarrow (P \rightarrow R))$



5. $(P \rightarrow (Q \rightarrow R)) \rightarrow (P \rightarrow (R \rightarrow Q))$

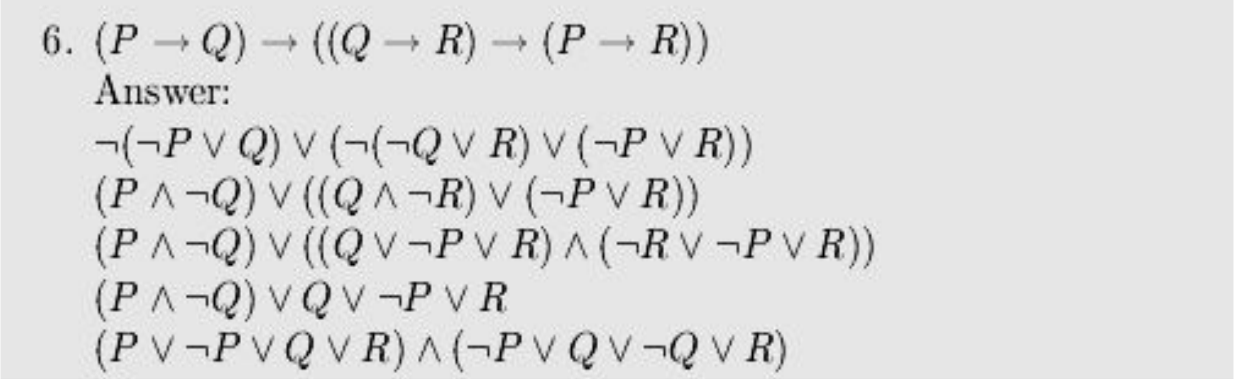
Answer:

$\neg(\neg P \vee \neg Q \vee R) \vee (\neg P \vee \neg R \vee Q)$

$(P \wedge Q \wedge \neg R) \vee (\neg P \vee \neg R \vee Q)$

$(P \vee \neg P \vee \neg R \vee Q) \wedge (Q \vee \neg P \vee \neg R \vee Q) \wedge (\neg R \vee \neg P \vee \neg R \vee Q)$

$\neg P \vee Q \vee \neg R$



6. $(P \rightarrow Q) \rightarrow ((Q \rightarrow R) \rightarrow (P \rightarrow R))$

Answer:

$\neg(\neg P \vee Q) \vee (\neg(\neg Q \vee R) \vee (\neg P \vee R))$

$(P \wedge \neg Q) \vee ((Q \wedge \neg R) \vee (\neg P \vee R))$

$(P \wedge \neg Q) \vee ((Q \vee \neg P \vee R) \wedge (\neg R \vee \neg P \vee R))$

$(P \wedge \neg Q) \vee Q \vee \neg P \vee R$

$(P \vee \neg P \vee Q \vee R) \wedge (\neg P \vee Q \vee \neg Q \vee R)$

Resolution

- If the unicorn is mythical, then it is immortal, but if the unicorn is not mythical, it is a mammal. If the unicorn is either immortal or mammal, then it is horned.
- Prove: the unicorn is horned.

- **Correctness:**
- If S1 $\vdash_R$ S2 then S1 $\models$ S2
- **Refutation Completeness:**
- If S is unsatisfiable then S $\vdash_R$ ( ), that is S is ***false*** (empty clause)

Gödel's theorem: you can not have a proof system which is sufficiently expressive and also complete, i.e. you can not build something which will prove everything

Resolution Example

- If the unicorn is mythical, then it is immortal, but if the unicorn is not mythical, it is a mammal. If the unicorn is either immortal or mammal, then it is horned.
- **Prove**: the unicorn is horned.

✔ M: mythical

✔ I: immortal

✔ A: mammal

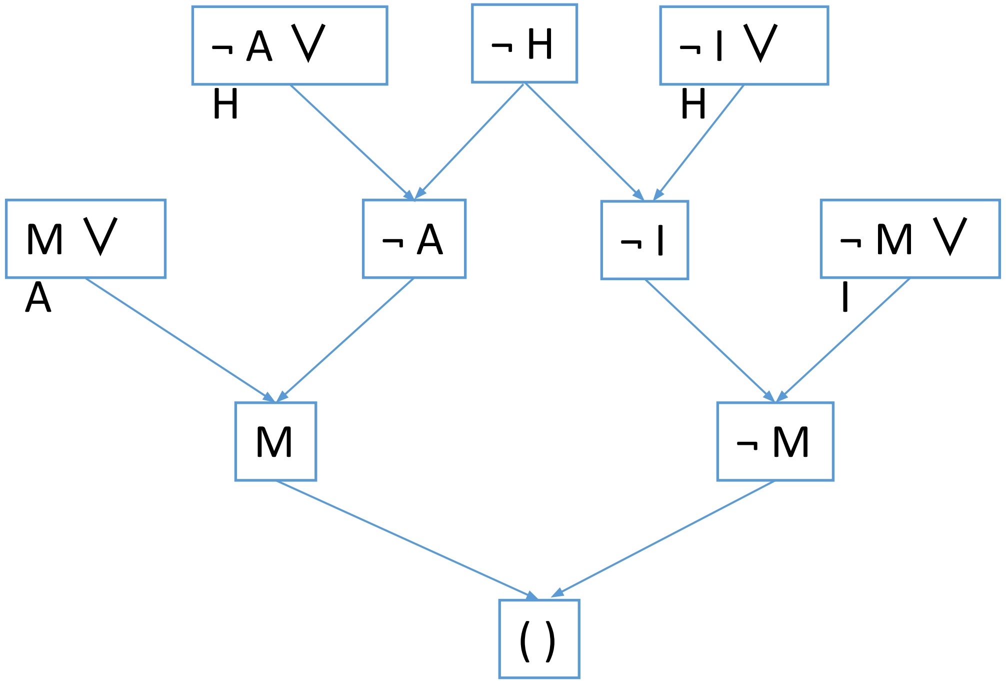
✔ H: horned

first statement, M ⇒ I clause is ¬ **M** ∨ **I**

second statement, **M** ∨ **A**

the last statement is,

A ⇒ H, i.e., ¬ **A** ∨ **H** and I ⇒ H, i.e., ¬ **I** ∨ **H**



¬ A ∨ H

¬ H

¬ I ∨ H

M ∨ A

¬ A

¬ I

¬ M ∨ I

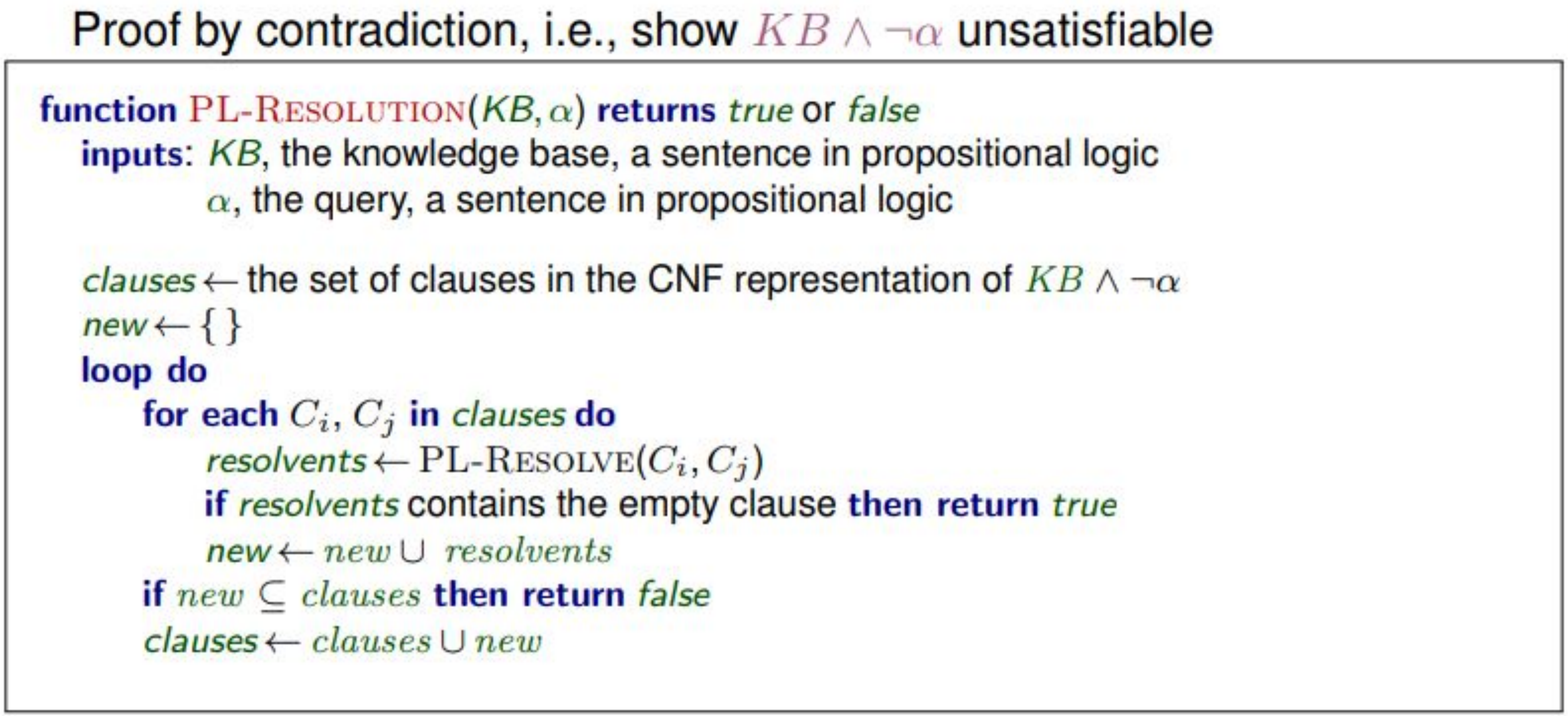
M

¬ M

( )

Proof by Contradiction

Resolution Algorithm



Proof by contradiction, i.e., show $KB \wedge \neg\alpha$ unsatisfiable

```
function PL-RESOLUTION($KB, \alpha$) returns true or false
   inputs: $KB$, the knowledge base, a sentence in propositional logic
           $\alpha$, the query, a sentence in propositional logic

   clauses ← the set of clauses in the CNF representation of $KB \wedge \neg\alpha$
   new ← { }
   loop do
      for each $C_i$, $C_j$ in clauses do
         resolvents ← PL-RESOLVE($C_i, C_j$)
         if resolvents contains the empty clause then return true
         new ← new ∪ resolvents
      if new ⊆ clauses then return false
      clauses ← clauses ∪ new
```