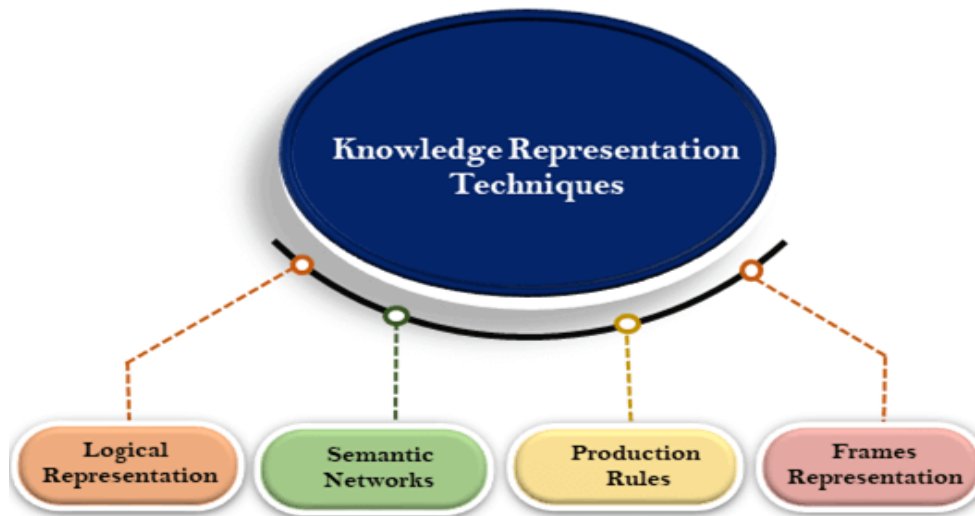


Knowledge Representation

There are mainly four ways of knowledge representation which are given as follows:

1. Logical Representation
2. Semantic Network Representation
3. Frame Representation
4. Production Rules



1. Logical Representation (Schema)

Logical representation is a language with some **definite rules** which deal with propositions and has no ambiguity in representation. It represents a conclusion based on various conditions and lays down some important **communication rules**. Also, it consists of precisely defined syntax and semantics which supports the sound inference. Each sentence can be translated into logics using syntax and semantics.

Syntax:

- Syntaxes are the rules which decide how can construct legal sentences in the logic.
- It determines which symbol we can use in knowledge representation.
- How to write those symbols.

Semantics:

- Semantics are the rules by which we can interpret the sentence in the logic.
- Semantic also involves assigning a meaning to each sentence.

Logical representation can be categorized into mainly two logics:

- a. Propositional Logics: is a collection of declarative statements that has either a truth value "true" or a false value "false". A propositional consists of propositional variables and

connectives. We denote the propositional variables by capital letters (A, B, etc). The connectives connect the propositional variables.

- Every propositional symbol and the true symbol is sentence.
- The conjunction or (and) sentence is a sentence Ex: $p \wedge q$.
- The Disjunction or and or two sentences is sentences example $P \vee Q$, $P \& Q$ call contact disconnect.
- The implication of one sentence from another is sent $P \rightarrow Q$ is the promise antecedent and the Q the calculation or consequence
- The equivalence of two sentences is a sentence
- The Bracts () or [] used to group symbols into sub expression.

b. Predicate logics: first-order logic or quantified logic is a formal language in which propositions are expressed in terms of predicates, variables and quantifiers. It is different from propositional logic which lacks quantifiers.

No one is happy.

a. (no one)(s/he is happy)

b. for all x, x is a person, x is not happy

c. = true iff 's/he is happy' is false for all possible values for 's/he'.

Advantages of logical representation:

1. Logical representation enables us to do logical reasoning.
2. Logical representation is the basis for the programming languages.

Disadvantages of logical Representation:

1. Logical representations have some restrictions and are challenging to work with.
2. Logical representation technique may not be very natural, and inference may not be so efficient.

2. Semantic Network Representation

Semantic networks are alternative of predicate logic for knowledge representation. In Semantic networks, can represent knowledge in the form of graphical networks. This network consists of nodes representing objects and arcs which describe the relationship between those objects. Semantic networks can categorize the object in different forms and can also link those objects.

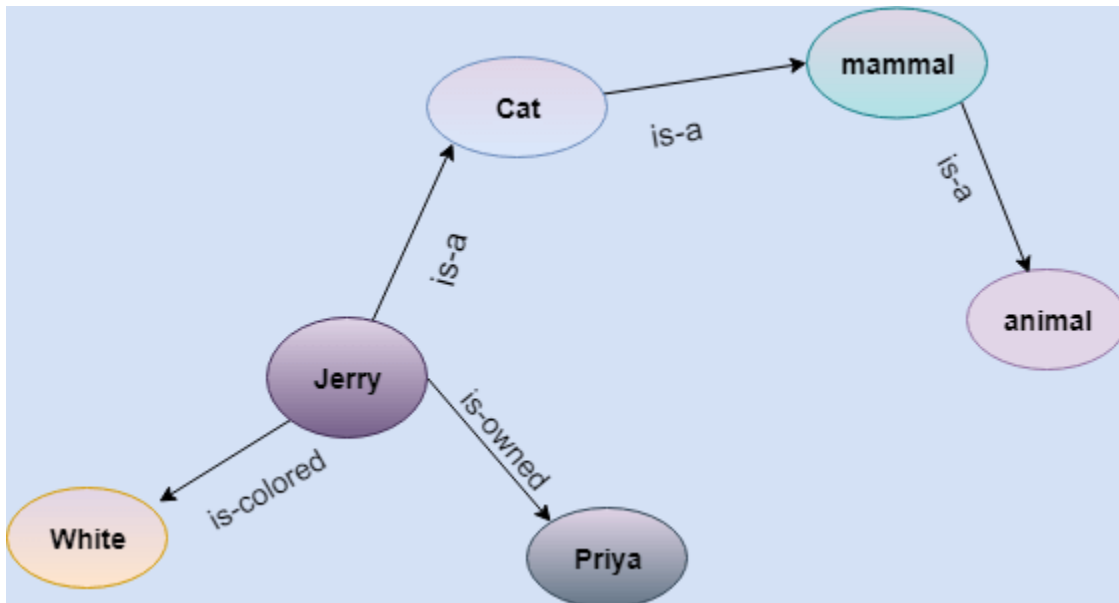
This representation consists of mainly two types of relations:

- a. IS-A relation (Inheritance).
- b. IS: to define the object or describe it
- c. Kind-of-relation

Example: Following are some statements which we need to represent in the form of nodes and arcs.

Statements:

- a. Jerry is a cat.
- b. Jerry is a mammal
- c. Jerry is owned by Priya.
- d. Jerry is white colored.
- e. All Mammals are animal.



In the above diagram, we have represented the different type of knowledge in the form of nodes and arcs. Each object is connected with another object by some relation.

Drawbacks in Semantic representation:

1. Semantic networks take more computational time at runtime as need to traverse the complete network tree to answer some questions. It might be possible in the worst case scenario that after traversing the entire tree, it finds that the solution does not exist in this network.
2. Semantic networks try to model human-like memory (Which has 1015 neurons and links) to store the information, but in practice, it is not possible to build such a vast semantic network.

3. These types of representations are inadequate as they do not have any equivalent quantifier, e.g., for all, for some, none, etc.
4. Semantic networks do not have any standard definition for the link names.
5. These networks are not intelligent and depend on the creator of the system.

Advantages of Semantic network:

1. Semantic networks are a natural representation of knowledge.
2. Semantic networks convey meaning in a transparent manner.
3. These networks are simple and easily understandable.

3. Frame Representation

A frame is a record like structure which consists of a collection of attributes and its values to describe an entity in the world. Frames are the AI data structure which divides knowledge into substructures by representing stereotypes situations. It 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.

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 are derived from semantic networks and later evolved into our modern-day classes and objects. A single frame is not much useful. Frames system consists of a collection of frames which are connected. In the frame, knowledge about an object or event can be stored together in the knowledge base. The frame is a type of technology which is widely used in various applications including Natural language processing and machine visions.

Example: 1

Let's take an example of a frame for a book

| Slots | Filters |
|----------------|-------------------------|
| Title | Artificial Intelligence |
| Genre | Computer Science |
| Author | Peter Norvig |
| Edition | Third Edition |

| | |
|-------------|------|
| Year | 1996 |
| Page | 1152 |

Example 2:

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 | Filter |
|-----------------------|---------------|
| Name | Peter |
| Profession | Doctor |
| Age | 25 |
| Marital status | Single |
| Weight | 78 |

Advantages of frame representation:

1. The frame knowledge representation makes the programming easier by grouping the related data.
2. The frame representation is comparably flexible and used by many applications in AI.
3. It is very easy to add slots for new attribute and relations.
4. It is easy to include default data and to search for missing values.
5. Frame representation is easy to understand and visualize.

Disadvantages of frame representation:

1. In frame system inference mechanism is not be easily processed.
2. Inference mechanism cannot be smoothly proceeded by frame representation.
3. Frame representation has a much generalized approach.

4. Production Rules

Production Rules consists primarily of a set of rules about behavior but it also includes the mechanism necessary to follow those rules as the system responds to states of the world.

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

- The set of production rules: production rules is conduction-action pair presented in the following form IF-condition –THEN action
- Working Memory: contain the description of the current state of the problem-solving.it used for matching rules and calculation. When the working finish this memory will be raised.
- The recognize-act-cycle: it implements the production system to move towards a goal within the set of rules.

In production rules agent checks for the condition and if the condition exists then production rule fires and corresponding action is carried out. The condition part of the rule determines which rule may be applied to a problem. And the action part carries out the associated problem-solving steps. This complete process is called a recognize-act cycle.

The working memory contains the description of the current state of problems-solving and rule can write knowledge to the working memory. This knowledge match and may fire other rules. If there is a new situation (state) generates, then multiple production rules will be fired together, this is called conflict set. In this situation, the agent needs to select a rule from these sets, and it is called a conflict resolution.

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).**

Advantages of Production rule:

1. The production rules are expressed in natural language.
2. The production rules are highly modular, so we can easily remove, add or modify an individual rule.

Disadvantages of Production rule:

1. Production rule system does not exhibit any learning capabilities, as it does not store the result of the problem for the future uses.
2. During the execution of the program, many rules may be active hence rule-based production systems are inefficient.

Propositional logic in Artificial intelligence

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 mathematical form.

Example:

- a) It is Sunday.
- b) The Sun rises from West (False proposition)
- c) $3+3=7$ (False proposition)
- d) 5 is a prime number.

Following are some 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.
- 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**.
- A proposition formula which has both true and false values is called
- Statements which are questions, commands, or opinions are not propositions such as "Where is Rohini", "How are you", "What is your name", are not propositions.
-

Syntax of propositional logic:

The syntax of propositional logic defines the allowable sentences for the knowledge representation. There are two types of Propositions:

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

Example:

- 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.

- b. 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, which are given as follows:

1. **Negation:** A sentence such as $\neg P$ is called negation of P. A literal can be either Positive literal or negative literal.

2. **Conjunction:** A sentence which has \wedge connective such as, $P \wedge Q$ is called a conjunction.

Example: Rohan is intelligent and hardworking. It can be written as,

P= Rohan is intelligent,

Q= Rohan is hardworking. $\rightarrow P \wedge Q$.

3. **Disjunction:** A sentence which has \vee connective, such as $P \vee Q$. is called disjunction, where P and Q are the propositions.

Example: "Ritika is a doctor or Engineer",

Here P= Ritika is Doctor. Q= Ritika is Doctor, so we can write it as $P \vee Q$.

4. **Implication:** A sentence such as $P \rightarrow Q$, 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 $P \rightarrow Q$

5. **Biconditional :** A sentence such as $P \Leftrightarrow Q$ 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 $P \Leftrightarrow Q$.

Following is the summarized table for Propositional Logic Connectives:

| 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:

In propositional logic, we need to know the truth values of propositions in all possible scenarios. We can combine all the possible combination with logical connectives, and the representation of these combinations in a tabular format is called **Truth table**. Following are the truth table for all logical connectives:

For Negation:

| P | $\neg P$ |
|-------|----------|
| True | False |
| False | True |

For disjunction:

| P | Q | $P \vee Q$ |
|-------|-------|------------|
| True | True | True |
| False | True | True |
| True | False | True |
| False | False | False |

For Conjunction:

| P | Q | $P \wedge Q$ |
|-------|-------|--------------|
| True | True | True |
| True | False | False |
| False | True | False |
| False | False | False |

For Implication:

| P | Q | $P \rightarrow Q$ |
|-------|-------|-------------------|
| True | True | True |
| True | False | False |
| False | True | True |
| False | False | True |

For Biconditional:

| P | Q | $P \leftrightarrow Q$ |
|-------|-------|-----------------------|
| True | True | True |
| True | False | False |
| False | True | False |
| False | False | True |

Truth table with three propositions:

We can build a proposition composing three propositions P, Q, and R. This truth table is made-up of 8n Tuples as we have taken three proposition symbols.

| P | Q | R | $\neg R$ | $P \vee Q$ | $P \vee Q \rightarrow \neg R$ |
|-------|-------|-------|----------|------------|-------------------------------|
| 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 |

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 $A \Leftrightarrow B$. In below truth table we can see that column for $\neg A \vee B$ and $A \rightarrow B$, are identical hence A is Equivalent to B

| A | B | $\neg A$ | $\neg A \vee B$ | $A \rightarrow B$ |
|---|---|----------|-----------------|-------------------|
| T | T | F | T | T |
| T | F | F | F | F |
| F | T | T | T | T |
| F | F | T | T | T |

Properties of Operators:

- **Commutativity:**
 - $P \wedge Q = Q \wedge P$, or
 - $P \vee Q = Q \vee P$.
- **Associativity:**
 - $(P \wedge Q) \wedge R = P \wedge (Q \wedge R)$,
 - $(P \vee Q) \vee R = P \vee (Q \vee R)$
- **Identity element:**
 - $P \wedge \text{True} = P$,
 - $P \vee \text{True} = \text{True}$.
- **Distributive:**
 - $P \wedge (Q \vee R) = (P \wedge Q) \vee (P \wedge R)$.
 - $P \vee (Q \wedge R) = (P \vee Q) \wedge (P \vee R)$.
- **DE Morgan's Law:**
 - $\neg (P \wedge Q) = (\neg P) \vee (\neg Q)$
 - $\neg (P \vee Q) = (\neg P) \wedge (\neg Q)$.
- **Double-negation elimination:**
 - $\neg (\neg P) = P$.

Limitations of Propositional logic:

- It represent relations like ALL, some, or none with propositional logic. Example:
 - a. **All the girls are intelligent.**
 - b. **Some apples are sweet.**
- Propositional logic has limited expressive power.

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

Rules of Inference in Artificial intelligence

Inference:

Generating the conclusions from evidence and facts is termed as Inference. Inference engine is a component of the system that applies logical rules to the knowledge base to deduce new information.

Inference rules:

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. Following are some terminologies related to 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 $\neg P \rightarrow \neg Q$.

From the above term some of the compound statements are equivalent to each other, which we can prove using truth table:

| P | Q | $P \rightarrow Q$ | $Q \rightarrow P$ | $\neg Q \rightarrow \neg P$ | $\neg P \rightarrow \neg Q$ |
|---|---|-------------------|-------------------|-----------------------------|-----------------------------|
| T | T | T | T | T | T |
| T | F | F | T | F | T |
| F | T | T | F | T | F |
| F | F | T | T | T | T |

Hence from the above truth table, we can prove that $P \rightarrow Q$ is equivalent to $\neg Q \rightarrow \neg P$, and $Q \rightarrow P$ is equivalent to $\neg P \rightarrow \neg Q$.

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 $P \rightarrow Q$ is true, then we can infer that Q will be true. It can be represented as:

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

Example:

Statement-1: "If I am sleepy then I go to bed" $\implies P \rightarrow Q$

Statement-2: "I am sleepy" $\implies P$

Conclusion: "I go to bed." $\implies Q$.

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 |

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:

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

Statement-1: "If I am sleepy then I go to bed" $\implies P \rightarrow Q$

Statement-2: "I do not go to the bed." $\implies \sim Q$

Statement-3: Which infers that "I am not sleepy" $\implies \sim P$

Proof by Truth table:

| 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 |

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. $P \rightarrow R$

Proof by truth table:

| P | Q | R | $P \rightarrow Q$ | $Q \rightarrow R$ | $P \rightarrow R$ |
|---|---|---|-------------------|-------------------|-------------------|
| 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 |

4. Disjunctive Syllogism:

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

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

Example:

Statement-1: Today is Sunday or Monday. $\Rightarrow P \vee Q$

Statement-2: Today is not Sunday. $\Rightarrow \neg P$

Conclusion: Today is Monday. $\Rightarrow Q$

Proof by truth-table:

| P | Q | $\neg P$ | $P \vee Q$ |
|---|---|----------|------------|
| 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 |

5. Addition:

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

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

Example:**Statement:** I have a vanilla ice-cream. $\implies P$ **Statement-2:** I have Chocolate ice-cream.**Conclusion:** I have vanilla or chocolate ice-cream. $\implies (P \vee Q)$ **Proof by Truth-Table:**

| P | Q | $P \vee Q$ |
|---|---|------------|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

6. Simplification:

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

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

Proof by Truth-Table:

| P | Q | $P \wedge Q$ |
|---|---|--------------|
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |

7. Resolution:

The Resolution rule state that if $P \vee Q$ and $\neg P \wedge R$ is true, then $Q \vee R$ will also be true. **It can be represented as**

$$\text{Notation of Resolution } \frac{P \vee Q, \neg P \wedge R}{Q \vee R}$$

Proof by Truth-Table:

| P | $\neg P$ | Q | R | $P \vee Q$ | $\neg P \wedge R$ | $Q \vee R$ |
|---|----------|---|---|------------|-------------------|------------|
| 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 |