# ME 620: Fundamentals of Artificial Intelligence

## Lecture 14: Propositional Logic

**Shyamanta M Hazarika**
Biomimetic Robotics and Artificial Intelligence Lab
Mechanical Engineering and M F School of Data Sc. & AI
IIT Guwahati

# Logic as a KR & R Language

- A Logic is a formal language, with precisely defined syntax and semantics
  - supports sound inference.
  - Independent of domain of application.

- Different logics exist, which allow one to represent different kinds of things.
  - allow more or less efficient inference.
  - propositional logic, predicate logic, temporal logic, modal logic, description logic..

# Advantages of Logic for KR

- Similar to declarative languages:
  - compact
  - task-independent
  - modular representation
  - reusable, flexible, maintainable
- Logic has formal well defined semantics
- Logic is expressive
  - incomplete knowledge
  - temporal logics
  - second order logic

# Proposition Logic

Propositional logic is a **mathematical system** for **reasoning about propositions** and **how they relate to one another.**

- Every statement in propositional logic consists of propositional variables combined via propositional connectives.
- Each variable represents some proposition.

  It is hot.
  It is humid.
- Connectives encode how propositions are related.

  If it is humid, then it is hot

# Proposition

**<u>Definition</u>**: A **proposition** is a statement that is, by itself, either true or false.

Sample Propositions

Can be either true or false.

- All humans are mortal.
- Ram is married.
- I'll pay for the meal.

**Things that aren't propositions**

Cannot be true or false.

- Come here! — Command.
- Why are you crying? — Question.

# Propositional Variable

**Definition**: A **propositional variable** represents an arbitrary proposition. We represent propositional variables with uppercase letters.

P  It is hot.

Q  It is humid.

**Definition**: Each variable can take one of two values: true or false. If a proposition is true, then we say its **truth value** is true, and if a proposition is false, we say its truth value is false.

# Propositional Connectives

## Logical NOT: $\neg P$

- Read "not P"
- $\neg P$ is true if and only if P is false.
- Also called **logical negation**.

## Logical AND: $P \wedge Q$

- Read "P and Q."
- $P \wedge Q$ is true if both P and Q are true.
- Also called **logical conjunction**.

## Logical OR: $P \vee Q$

- Read "P or Q."
- $P \vee Q$ is true if at least one of P or Q are true.
- Also called **logical disjunction**.

The OR operator is an *inclusive* OR; It is true if at least one of the operands is true.

# Propositional Connectives

## Implication: $P \rightarrow Q$

- Read "If P then Q".
- False when P is true and Q is false; and is true otherwise.
- Also called **material conditional operator**.

## Biconditional: $P \leftrightarrow Q$

- Read "P if and only if Q".
- True if P and Q have the same truth values; and false otherwise.
- Also called **material biconditional operator**

## **true** and **false**:

- The symbol $\top$ is a value that is always true.
- The symbol $\bot$ is a value that is always false.

# Well-formed Formula

**Definition**: A **sentence** also called a **well-formed formula** is defined as follows:

- A symbol S is a sentence
- If S is a sentence, then $\neg$S is a sentence
- If S is a sentence, then (S) is a sentence
- If S and T are sentences, then
  i. $(S \vee T)$ ii. $(S \wedge T)$ iii. $(S \rightarrow T)$ and iv. $(S \leftrightarrow T)$
  are sentences
- A sentence results from a finite number of applications of the above rules.

# Truth Table

A **truth table** is a table showing the truth value of a propositional logic formula as a function of its inputs.

- Represent the relationship between the truth values of propositions and compound propositions formed from those propositions.

| P | Q | P $\vee$ Q |
|---|---|---|
| T | T | T |
| T | F | T |
| F | T | T |
| F | F | F |

# Truth Table

A **truth table** is a table showing the truth value of a propositional logic formula as a function of its inputs.

- Represent the relationship between the truth values of propositions and compound propositions formed from those propositions.
- Formally defining what a connective `means’.
- Deciphering complex propositional formula.

# Implication

For propositions **P** and **Q**, **P → Q**, the **implication** or **conditional statement** is **false** when **P is true** and **Q is false**, and is **true otherwise**.

- P is called the premise or hypothesis.
- Q is called the conclusion.

| P | Q | P → Q |
|---|---|---|
| F | F | T |
| F | T | T |
| T | F | F |
| T | T | T |

We want P → Q to mean
"whenever P is true, Q is true as well."

Only way this doesn't happen is if P is true and Q is false.

# Implication

In English, a sentence of the form `if A then B' can have different meanings.

1. Typically there is a causal relationship between A and B, which is not required in logic.
2. We are often implying more than simply that if A holds, then B holds as well.

- Example

  If I earn a bonus, then I will buy a car.

  P: I earn a bonus.

  Q: I will buy a car.

  $P \rightarrow Q$

  - The common-sense interpretation of this sentence is that the inverse statement is also true:

    If I *do not* earn a bonus, then I will *not* buy a car.

  This is not implied by $P \rightarrow Q$.

# Biconditional

The **biconditional** of statements **P** and **Q**, denoted **P ↔ Q** , is read "P if and only if Q", and is **true** if **P** and **Q** have the **same truth values**, and **false otherwise**.

The biconditional operator is used to represent a two-directional implication.

| P | Q | P ↔ Q |
|---|---|---|
| F | F | T |
| F | T | F |
| T | F | F |
| T | T | T |

Specifically, p ↔ q means that p implies q and q implies p.

Conversely if both P implies Q and Q implies P are true, then P if and only if Q is true.

# Operator Precedence

**Operator precedence** for propositional logic:

$\neg$ NOT binds to whatever immediately follows it.

$\wedge$

$\vee$ $\wedge$ and $\vee$ bind more tightly than $\rightarrow$

$\rightarrow$

$\leftrightarrow$

- All operators are right-associative.
- Parentheses can be used to disambiguate.

$$\neg x \rightarrow x \vee z \wedge y$$

$$(\neg x) \rightarrow (x \vee (z \wedge y))$$

# Translating English Into Logic

User defines a **set of propositional symbols**, like P and Q.

User defines the **semantics** of each **propositional symbol**:

| | |
|---|---|
| P | It is hot. |
| Q | It is humid. |
| R | It is raining. |

1. If it is humid, then it is hot
   $Q \rightarrow P$

2. If it is hot and humid, then it is raining.
   $(P \wedge Q) \rightarrow R$

# Translating English Into Logic

| | |
|---|---|
| W | I will work hard. |
| V | There are vacancies. |
| J | I will get the job. |

If I don’t work hard, then I won’t get the job.

3. I won’t get the job, if I don’t work hard.

$$\neg W \rightarrow \neg J$$

P if Q

translates to

$Q \rightarrow P$

# Translating English Into Logic

W I will work hard.
V There are vacancies.
J I will get the job.

4. If I work hard but there are no vacancies, I won't get the job.

Because the second part of the sentence is a surprise, "but" is used instead of "and".

$$( W \wedge \neg V ) \rightarrow \neg J$$

P, but Q

translates to

$$P \wedge Q$$

# Logical Equivalence

$\neg(P \land Q)$ and $(\neg P \lor \neg Q)$ have the same truth tables, we say that they're **equivalent** to one another.

$\neg(P \land Q) \equiv (\neg P \lor \neg Q)$

- The $\equiv$ symbol is not a connective. It's related to $\leftrightarrow$, but it's not the same:
- The statement $\neg(P \land Q) \equiv (\neg P \lor \neg Q)$ means `the two formulas are equivalent.'

The formula evaluates to true every time.

- The statement $\neg(P \land Q) \leftrightarrow (\neg P \lor \neg Q)$ is a propositional formula. If you plug in different values of P and Q, it will evaluate to a truth value.

# De Morgan's Laws

Using truth table, we conclude

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

| P | Q | $\neg(P \wedge Q)$ | $\neg P \vee \neg Q$ |
|---|---|---|---|
| T | T | F | F |
| T | F | T | T |
| F | T | T | T |
| F | F | T | T |

# De Morgan's Laws

Using truth table, we conclude

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

| P | Q | $\neg(P \vee Q)$ | $\neg P \wedge \neg Q$ |
|---|---|---|---|
| T | T | F | F |
| T | F | F | F |
| F | T | F | F |
| F | F | T | T |

# Logical Equivalence

| P | Q | $P \to Q$ | $\neg(P \land \neg Q)$ |
|---|---|---|---|
| T | T | T | T |
| T | F | F | F |
| F | T | T | T |
| F | F | T | T |

Here's a useful equivalence.

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

Start with $P \to Q \equiv \neg (P \wedge \neg Q)$

By De Morgan's laws:

- $P \to Q \equiv \neg P \vee \neg\neg Q$
- $P \to Q \equiv \neg P \vee Q$

Thus $P \to Q \equiv \neg P \vee Q$

# Rules of Inference

A **rule of inference** is sound if its conclusion is true whenever the premise is true.

- Here are some examples of sound rules of inference.
- Each can be shown to be sound using a truth table.

| RULE | PREMISE | CONCLUSION |
|---|---|---|
| Modus Ponens | $A, A \rightarrow B$ | $B$ |
| And Introduction | $A, B$ | $A \wedge B$ |
| And Elimination | $A \wedge B$ | $A$ |
| Double Negation | $\neg \neg A$ | $A$ |
| Unit Resolution | $A \vee B, \neg B$ | $A$ |
| Resolution | $A \vee B, \neg B \vee C$ | $A \vee C$ |

# Proving Theorems

A **proof** is a **sequence of sentences**, where each sentence is either a premise or a sentence **derived from earlier sentences** in the proof by one of the rules of inference.

- The last sentence is the **theorem** (also called goal or query) that we want to prove.

- Example

| | | |
|---|---|---|
| 1. $Q$ | Premise | It is humid |
| 2. $Q \rightarrow P$ | Premise | If it is humid, it is hot |
| 3. $P$ | Modus Ponens(1,2) | It is hot |
| 4. $(P \wedge Q) \rightarrow R$ | Premise | If it's hot and humid, it's raining |
| 5. $P \wedge Q$ | And Introduction(1,3) | It is hot and humid |
| 6. $R$ | Modus Ponens(4,5) | It is raining |