Knowledge Representation

Inference in Propositional Logic**, First order Predicate logic, Resolution**,

Logical Reasoning, Forward chaining, Backward chaining,

Knowledge representation techniques: Semantic networks and frames.

First-Order logic

First-order logic is another way of knowledge representation in artificial intelligence.

First-order logic is also known as **Predicate logic or First-order predicate logic**. First-order logic is a powerful language that develops information about the objects in a more easy way and can also express the relationship between those objects.

First-order logic (like natural language) does not only assume that the world contains facts like propositional logic but also assumes the following things in the world:

Objects: A, B, people, numbers, colors, wars, theories, squares, pits…

Relations: It can be unary relation such as: red, round, is adjacent, **or n-any relation such as:** the sister of, brother of, has color,

Function: relation that associates each element x of a set X, e.g Father of, best friend, third inning of, end of, ...

First-order logic also has two main parts:

- Syntax
- Semantics

Syntax of First-Order logic:

The syntax of FOL determines which collection of symbols is a logical expression in first-order logic. The basic syntactic elements of first-order logic are symbols.

Basic Elements of First-order logic:

Constant	1, 2, A, John, Mumbai, cat,....
Variables	x, y, z, a, b,....
Predicates	Brother, Father, >,....
Function	sqrt, LeftLegOf, ...
Connectives	∧, ∨, ¬, ⇒, ⇔
Equality	==
Quantifier	∀, ∃

Atomic sentences: Atomic sentences are the most basic sentences of first-order logic. These sentences are formed from a **predicate(the part of sentence or clause containing a verb and stating something about the subject**) symbol followed by a parenthesis with a sequence of terms. We can represent atomic sentences as **Predicate (term1, term2, ..., term n)**.

Example: Ravi and Ajay are brothers: => Brothers(Ravi, Ajay). Chinky is a cat: => cat (Chinky).

Complex Sentences:

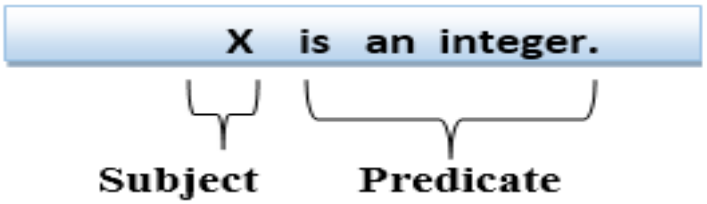
Complex sentences are made by combining atomic sentences using connectives.

First-order logic statements can be divided into two parts:

Subject: Subject is the main part of the statement.

Predicate: A predicate can be defined as a relation, which binds two atoms together in a statement.

Consider the statement: "x is an integer.", it consists of two parts, the first part x is the subject of the statement and second part "is an integer," is known as a predicate.



Quantifiers in First-order logic:

A quantifier is a language element which generates quantification, and quantification specifies the quantity of specimen in the universe of discourse.

These are the symbols that permit to determine or identify the range and scope of the variable in the logical expression.

There are two types of quantifier:

–Universal Quantifier, (for all, everyone, everything)

–Existential quantifier, (for some, at least one).

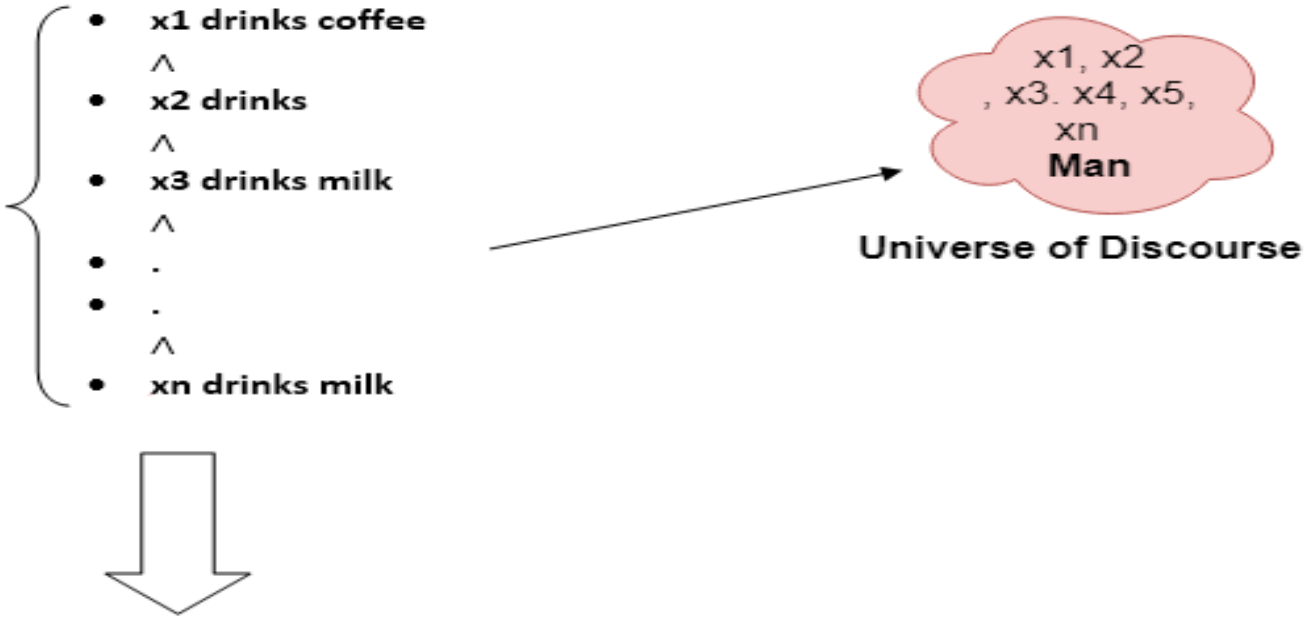
Universal Quantifier:

Universal quantifier is a symbol of logical representation, which specifies that the statement within its range is true for everything or every instance of a particular thing.

The Universal quantifier is represented by a symbol ∀.

If x is a variable, then ∀x is read as: **For all x, For each x, For every x.**

Example: All man drink coffee.



So in shorthand notation, we can write it as :

∀x man(x) → drink (x, coffee).

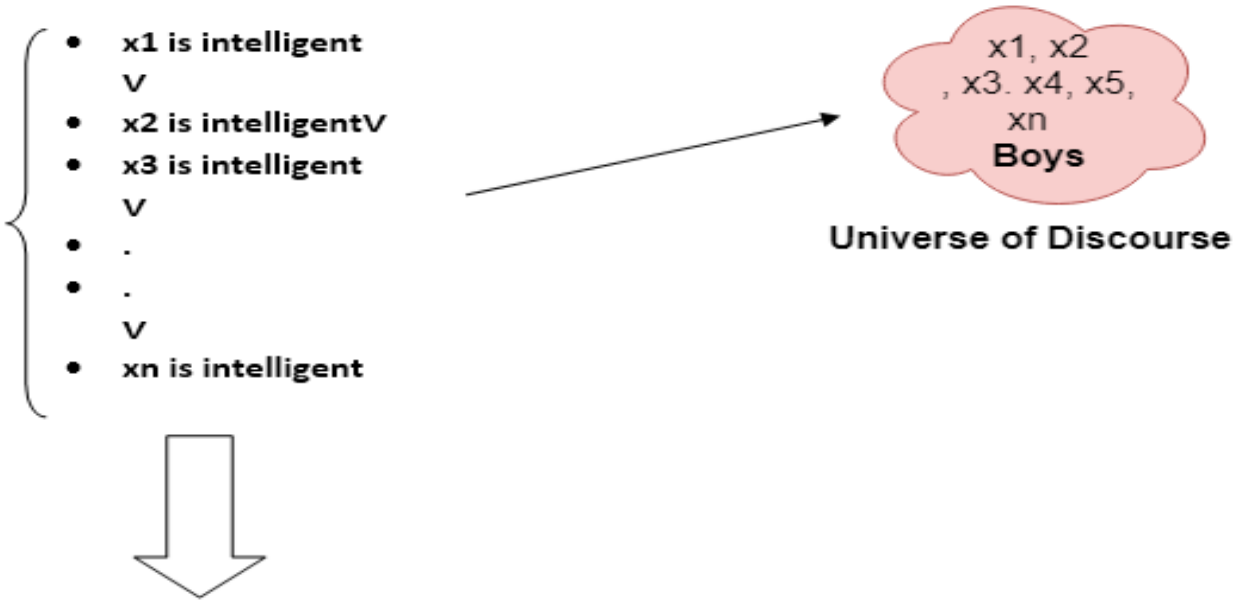
It will be read as: There are all x where x is a man who drink coffee.

Existential Quantifier:

Existential quantifiers are the type of quantifiers, which express that the statement within its scope is true for at least one instance of something. It is denoted by the logical operator ∃.

If x is a variable, then existential quantifier will be ∃x or ∃(x). And it will be read as: **There exists a 'x', For some 'x', For at least one 'x'**

Example: Some boys are intelligent.



Examples of FOL using quantifier

All birds fly.
In this question the predicate is "fly(bird)."
And since there are all birds who fly so it will be represented as follows.

$\forall x \text{ bird}(x) \rightarrow \text{fly}(x)$.

Every man respects his parent.
In this question, the predicate is "respect(x, y)," where x=man, and y= parent.
Since there is every man so will use ∀, and it will be represented as follows:

$\forall x \text{ man}(x) \rightarrow \text{respects}(x, \text{parent})$.

Not all students like both Mathematics and Science.
In this question, the predicate is "**like(x, y)," where x= student, and y= subject**.
Since there are not all students, so we will use **∀ with negation, so** following representation for this:

$\neg\forall (x) [\text{student}(x) \rightarrow \text{like}(x, \text{Mathematics}) \wedge \text{like}(x, \text{Science})]$.

1. Marcus was a Pompeian.
2. All Pompeian's were Romans.
3. Caesar was a ruler
4. All Romans were either loyal to Caesar or hated him.
5. Everyone is loyal to someone.

2. Marcus was a Pompeian.

   $Pompeian(Marcus)$

3. All Pompeians were Romans.

   $\forall x : Pompeian(x) \rightarrow Roman(x)$

4. Caesar was a ruler.

   $ruler(Caesar)$

   Here we ignore the fact that proper names are often not references to unique individuals, since many people share the same name. Sometimes deciding which of several people of the same name is being referred to in a particular statement may require a fair amount of knowledge and reasoning.

5. All Romans were either loyal to Caesar or hated him.

   $\forall x : Roman(x) \rightarrow loyalto(x, Caesar) \vee hate(x, Caesar)$

   In English, the word "or" sometimes means the logical *inclusive-or* and sometimes means the logical *exclusive-or* (XOR). Here we have used the inclusive interpretation. Some people will argue, however, that this English sentence is really stating an,exclusive-or. To express that, we would have to write:

   $\forall x : Romman(x) \rightarrow [(loyal\ to(x, Caesar) \vee hate(x, Caesar)) \wedge$
   $\neg(loyalto(x, Caesar) \wedge hate(x,Caesar))]$

6. Everyone is loyal to someone.

   $\forall x : \rightarrow y : loyalto(x,y)$

   A major problem that arises when trying to convert English sentences into logical statements is the scope of quantifiers. Does this sentence say, as we have assumed in writing the logical formula above, that for each person there exists someone to whom he or she is loyal, possibly a different someone for everyone? Or does it say that there exists someone to whom everyone is loyal (which would be written as $\exists y : \forall x : loyalto(x,y)$)? Often only one of the two interpretations seems likely, so people tend to favor it.

Inference in FOL: Truth table approach

- Is the Truth-table approach a viable approach for the FOL?
- **NO!**
- Why?
- It would require us to enumerate and list all possible interpretations I
- I = (assignments of symbols to objects, predicates to relations and functions to relational mappings)
- Simply there are too many interpretations

Inference in FOL: Inference rules

- Is the Inference rule approach a viable approach for the FOL?
- **Yes.**
- The inference rules represent sound inference patterns one can apply to sentences in the KB
- What is derived follows from the KB
- **Condition –**

  we need to add rules for handling quantifiers

Inference rules

- Inference rules from the propositional logic: –

Modus ponens – $\dfrac{A \Rightarrow B, \quad A}{B}$

Resolution – $\dfrac{A \vee B, \quad \neg B \vee C}{A \vee C}$

and others: Addition, Simplification

- Additional inference rules are needed for sentences with quantifiers and variables
  - Must involve variable substitutions.

Variable substitutions

- Variables in the sentences can be substituted with terms. (terms = constants, variables, functions)

- **Substitution:**

– Is a mapping from variables to terms

$$\{x_1 / t_1, x_2 / t_2, \ldots\}$$

– Application of the substitution to sentences

$$SUBST(\{x/Sam, y/Pam\}, Likes(x, y)) = Likes(Sam, Pam)$$

$$SUBST(\{x/z, y/fatherof(John)\}, Likes(x, y)) = Likes(z, fatherof(John))$$

Inference rules for quantifiers

1. Universal elimination:

$$\frac{\forall x\ \phi(x)}{\phi(a)}$$ a - is a constant symbol

– substitutes a variable with a **constant symbol**

Example:

$$\forall x\ Likes(x, IceCream)$$

$$\downarrow$$

$$Likes(Ben, IceCream)$$

2. Existential Elimination:

$$\frac{\exists x\ \phi(x)}{\phi(a)}$$

– Substitutes a variable with a constant symbol that does not appear elsewhere in the KB

Example:

$$\exists x\ Kill(x, Victim) \longrightarrow Kill(Murderer, Victim)$$

Special constant called a **Skolem** constant

Instantiation

Universal Instantiation –

- can be applied several times to add new sentences
- the new KB is logically equivalent to the old

Existential Instantiation –

- can be applied once to replace the existential sentence
- the new KB is not equivalent to the old – but is satisfiable iff the old KB was satisfiable.

Reduction to propositional inference

Suppose the KB contains just the following:

$\forall x$ King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)
King(John)
Greedy(John)
Brother(Richard,John)

- Instantiating the universal sentence in **all possible** ways, we have:
  King(John) $\wedge$ Greedy(John) $\Rightarrow$ Evil(John)
  King(Richard) $\wedge$ Greedy(Richard) $\Rightarrow$ Evil(Richard)
  King(John)
  Greedy(John)
  Brother(Richard,John)

- The new KB is **propositionalized**: proposition symbols are

  King(John), Greedy(John), Evil(John), King(Richard), etc.

Problems with propositionalization: Propositionalization seems to generate lots of irrelevant sentences

UNIFICATION

- **Problem in inference:** Universal elimination gives many opportunities for substituting variables with ground terms

$$\frac{\forall x\ \phi(x)}{\phi(a)}$$

a - is a constant symbol

- **Solution:** Try substitutions that help us to make progress
  - Use substitutions of "similar" sentences in KB
- **Example:**

  $\forall x\ \text{King}(x) \wedge \text{Greedy}(x) \Rightarrow \text{Evil}(x)$

  $\text{King}(\text{John})$

  $\forall y\ \text{Greedy}(y)$

  If we use a substitution $\sigma = \{x/\text{John}, y/\text{John}\}$

  we can use modus ponens to infer Evil(John) in one step

Unification: takes two similar sentences and computes the substitution that makes them look the same, if it exists.

Examples:

$$UNIFY(Knows(John, x), Knows(John, Jane)) = \{x / Jane\}$$

$$UNIFY(Knows(John, x), Knows(y, Ann)) = ?$$

$$UNIFY(Knows(John, x), Knows(y, MotherOf(y))) = ?$$

$$UNIFY(Knows(John, x), Knows(x, Elizabeth)) = ?$$

$$UNIFY(Knows(John, x), Knows(John, Jane)) = \{x / Jane\}$$

$$UNIFY(Knows(John, x), Knows(y, Ann)) = \{x / Ann, y / John\}$$

$$UNIFY(Knows(John, x), Knows(y, MotherOf(y))) = \{x / MotherOf(John), y / John\}$$

$$UNIFY(Knows(John, x), Knows(x, Elizabeth)) = fail$$

Generalized inference rules

- Use substitutions that let us make inferences

Example: Modus Ponens

- If there exists a substitution σ such that

$$SUBST\ (\sigma, A_i) = SUBST\ (\sigma, A_i')\quad \textbf{for all i=1,2, n}$$

$$\frac{A_1 \wedge A_2 \wedge \ldots A_n \Rightarrow B,\quad A_1', A_2', \ldots A_n'}{SUBST\ (\sigma, B)}$$

Substitution that satisfies the generalized inference rule can be build via unification process

Advantage of the generalized rules:

- they are focused
- only substitutions that allow the inferences to proceed

Generalized Modus Ponens (GMP)

$$\frac{A_1 \wedge A_2 \wedge \ldots A_n \Rightarrow B, \quad A_1', A_2', \ldots A_n'}{SUBST\ (\sigma, B)}$$

Example:
$\forall$x King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)
King(John)
$\forall$y Greedy(y)
Brother(Richard,John)

A_1' is *King*(*John*) A_1 is *King*(*x*)
A_2' is *Greedy*(*y*) A_2 is *Greedy*(*x*)
σ is {x/John,y/John} B is *Evil*(*x*)
SUBS(σ,B) = *Evil*(*John*)

- GMP is used with KB of **definite clauses**
- Definite clauses **exactly one positive literal**
- All variables assumed universally quantified