First-order Logic

Simply, What is First Order Logic?

First-order logic is a way of writing and understanding statements about objects and their relationships. It's like a very precise language that helps us describe things clearly and reason about them.

Why Do We Need It?

We need first-order logic to:

1. **Make clear statements**: It helps us avoid confusion by being very specific about what we mean.
2. **Solve problems**: It allows us to make logical conclusions and solve problems systematically.
3. **Understand relationships**: It helps us describe how different things are connected or related.

Imagine you have a box of toys, and you want to say things about the toys. First-order logic gives you a special way to talk about the toys and their properties.

Basic Ideas:

- **Objects**: Think of objects as your toys (like a teddy bear, a toy car, and a doll).
- **Properties**: These are things you can say about your toys (like "is red," "has wheels," or "can talk").
- **Relationships**: These are ways your toys can be connected (like "is next to," "is bigger than," or "loves").

Using First-Order Logic:

1. **Names**: We use names to refer to specific toys (like calling the teddy bear "Teddy").
2. **Properties**: We use special words to talk about properties. For example:
   - "Teddy is brown" can be written as Brown(Teddy).
3. **Relationships**: We use special words for relationships. For example:
   - "Teddy loves Doll" can be written as Loves(Teddy,Doll).
4. **Quantifiers**:
   - **Everyone**: If you want to say something about all your toys, you use "everyone" (or "all"). For example, "All toys are fun" can be written as $\forall$x(Toy(x)$\to$Fun(x)).

- **Someone**: If you want to say something about at least one toy, you use "someone" (or "some"). For example, "Some toy is red" can be written as ∃x(Toy(x)∧Red(x)).

Relating to Real Life:

1. **School**: Think about your classmates. If you want to say "Every student likes recess," you can use first-order logic to write that clearly.
2. **Games**: When playing a game, you can use first-order logic to describe rules. For example, "If a player scores, they get a point."

By using first-order logic, you can clearly and precisely describe things in a way that makes it easy to understand and reason about the world around you.

First-order Language: The Language of Predicate Logic

Definition 1:

The language of first-order logic is a

$$L^{(1)} = \langle LC, Var, \text{Con, Term, Form} \rangle$$

ordered 5-tuple, where

1. $LC = \{\neg, \supset, \wedge, \vee, \equiv, =, \forall, \exists, ()\}$:, (the set of logical constants).
2. Var $(= \{x_n : n = 0, 1, 2, \ldots\})$: countable infinite set of variables
3. Con $= \bigcup_{n=0}^{\infty}(\mathcal{F}(n) \cup \mathcal{P}(n))$ the set of non-logical constants (at best countable infinite), where
   (a) $\mathcal{F}(0)$: the set of name parameters,
   (b) $\mathcal{F}(n)$: the set of n argument function parameters,
   (c) $\mathcal{P}(0)$: the set of proposition parameters,
   (d) $\mathcal{P}(n)$: the set of predicate parameters.
4. The sets $LC, \text{Var}, \mathcal{F}(n), \mathcal{P}(n)$ are pairwise disjoint $(n = 0, 1, 2, \ldots)$.
5. The set of terms, i.e. the set Term is given by the following inductive definition:
   (a) $\text{Var} \cup \mathcal{F}(0) \subseteq \text{Term}$
   (b) If $f \in \mathcal{F}(n), (n = 1, 2, \ldots)$, and $t_1, t_2, \ldots, t_n \in$ Term, then $f(t_1, t_2, \ldots, t_n) \in$ Term.
6. The set of formulas, i.e. the set Form is given by the following inductive definition:
   (a) $\mathcal{P}(0) \subseteq \text{Form}$
   (b) If $t_1, t_2 \in$ Term, then $(t_1 = t_2) \in$ Form
   (c) If $P \in \mathcal{P}(n), (n \geq 1)$, and $t_1, t_2, \ldots, t_n \in$ Term, then $P(t_1, t_2, \ldots, t_n) \in$ Form.
   (d) If $A \in$ Form, then $\neg A \in$ Form.

(e) If $A, B \in$ Form, then $(A \supset B), (A \wedge B), (A \vee B), (A \equiv B) \in$ Form.

(f) If $x \in$ Var, $A \in$ Form, then $\forall x A, \exists x A \in$ Form.

Definition 2:

Let $L^{(1)} = \langle LC, Var, \text{Con, Term, Form} \rangle$ be a language of first-order logic Then the set of atomic formulas of $L^{(1)}$ (in notation AtForm) is the following:

1. $\mathcal{P}(0) \subseteq$ AtForm
2. If $t_1, t_2 \in$ Term, then $(t_1 = t_2) \in$ AtForm
3. If $P \in \mathcal{P}(n), (n \geq 1)$, and $t_1, t_2, \ldots, t_n \in$ Term, then $P(t_1, t_2, \ldots, t_n) \in$ AtForm.

Syntactical Properties of Variables: Free and Bound Variables

Two different uses of variables in first-order formulae: 1. Free variables: used to denote unknown or unspecified objects, as in $(x > 5) \vee (x^2 + x - 2 = 0)$. 2. Bound variables: used to quantify, as in

$$\exists x \left(x^2 + x - \mathbf{2} = \mathbf{0} \right) \text{ and } \forall x \left(x > \mathbf{5} \rightarrow x^2 + x - \mathbf{2} > \mathbf{0} \right).$$

Scope of (an occurrence of) a quantifier in a formula A : the unique subformula $Q \times B$ beginning with that occurrence of the quantifier. An occurrence of a variable x in a formula A is bound if it is in the scope of some occurrence of a quantifier Qx in A. Otherwise, that occurrence of x is free. A variable is free (bound) in a formula, if it has a free (bound) occurrence in it. For instance, in the formula

$$A = (x > \mathbf{5}) \rightarrow \forall y (y < \mathbf{5} \rightarrow (y < x \wedge \exists x (x < 3))).$$

the first two occurrences of x are free, while all other occurrences of variables are bound. Thus, the only free variable in A is x, while both x and y are bound in A.

A simplified example:

Imagine you have some toys, and you label them with names. Let's think about how you might use those names when talking about the toys.

- **Free Occurrence**: When you mention a toy's name directly without any special rules.
  - Example: Saying "x is red" means you are directly talking about the toy named "x."
- **Bound Occurrence**: When you mention a toy's name but with a rule that applies to all toys or some toys.
  - Example: Saying "For every x, x is red" means you're talking about all toys being red, not just the toy named "x."

Definition:

Let $L^{(1)} = \langle LC, Var, \text{Con, Term, Form} \rangle$ be a first order language and $A \in$ Form be a formula. The set of free variables of the formula A (in notation: FreeVar(A)) is given by the following inductive definition:

1. If A is an atomic formula (i.e. $A \in \text{AtForm}$), then the members of the set $\text{FreeVar}(A)$ are the variables occuring in A.
2. If the formula A is $\neg B$, then $\text{FreeVar}(A) = \text{FreeVar}(B)$.
3. If the formula A is $(B \supset C), (B \wedge C), (B \vee C)$ or $(B \equiv C)$, then $\text{FreeVar}(A) = \text{FreeVar}(B) \bigcup \text{FreeVar}(C)$.
4. If the formula A is $\forall x B$ or $\exists x B$, then $\text{FreeVar}(A) = \text{FreeVar}(B) \backslash \{x\}$.

Definition:

Let $L^{(1)} = \langle LC, Var, \text{Con, Term, Form} \rangle$ be a first order language and $A \in \text{Form}$ be a formula. The set of bound variables of the formula A (in notation: $\text{BoundVar}(A)$) is given by the following inductive definition:

1. If A is an atomic formula (i.e. $A \in At$ Form), then $\text{BoundVar}(A) = \emptyset$.
2. If the formula A is $\neg B$, then $\text{BoundVar}(A) = \text{FreeVar}(B)$.
3. If the formula A is $(B \supset C), (B \wedge C), (B \vee C)$ or $(B \equiv C)$, then $\text{BoundVar}(A) = \text{BoundVar}(B) \cup \text{BoundVar}(C)$.
4. If the formula A is $\forall x B$ or $\exists x B$, then $\text{BoundVar}(A) = \text{BoundVar}(B) \cup \{x\}$.

Syntactical Properties of Variables: Free and Bound Occurrences

Definition:

Let $L^{(1)} = \langle LC, Var, \text{Con, Term, Form} \rangle$ be a first order language, $A \in \text{Form}$ be a formula, and $x \in Var$ be a variable.

1. A fixed occurrence of the variable x in the formula A is free if it is not in the subformulae $\forall x B$ or $\exists x B$ of the formula A.
2. A fixed occurrence of the variable x in the formula A is bound if it is not free.

Remarks:

1. If x is a free variable of the formula A (i.e. $x \in \text{FreeVar}(A)$), then it has at least one free occurrence in A.
2. If x is a bound variable of the formula A (i.e. $x \in \text{BoundVar}(A)$), then it has at least one bound occurrence in A.
3. A fixed occurrence of a variable x in the formula A is free if
   - it does not follow a universal or an existential quantifier, or
   - it is not in a scope of a $\forall x$ or a $\exists x$ quantification.
4. A variable x may be a free and a bound variable of the formula $A: \ (P(x) \wedge \exists x R(x))$

Definition:

Let $L^{(1)} = \langle LC, Var, \text{Con, Term, Form} \rangle$ be a first order language and $A \in \text{Form}$ be a formula.

1. If $\text{FreeVar}(A) \neq \emptyset$, then the formula A is an open formula.
2. If $\text{FreeVar}(A) = \emptyset$, then the formula A is a closed formula.

Interpretation and Assignment in First-order Logic

Interpretations

Interpretations provide the meaning for the symbols used in a first-order logic formula. An interpretation consists of a domain of discourse and an assignment of meanings to the non-logical symbols (constants, function symbols, and predicates) in the formula.

Components of an Interpretation:

1. **Domain of Discourse (D)**: A non-empty set of objects that the variables can refer to.
2. **Interpretation of Constants**: Each constant symbol is assigned a specific object in the domain.
   - Example: If a is a constant, it might be assigned to a particular object in the domain D.
3. **Interpretation of Function Symbols**: Each n-ary function symbol is assigned an n-ary function from the domain to the domain.
   - Example: If f is a unary function, it might be interpreted as a function $f : D \to D$.
4. **Interpretation of Predicate Symbols**: Each n-ary predicate symbol is assigned an n-ary relation over the domain.
   - Example: If P is a binary predicate, it might be interpreted as a relation $P \subseteq D \times D$.

Variable assignment

A **variable assignment** is a function that assigns values from the domain of discourse to the variables in a formula. This helps to evaluate the truth of formulas involving variables under a given interpretation.

How Variable Assignment Works:

- Suppose we have a variable assignment σ:
  - $\sigma(x)$ assigns a value from the domain D to the variable x.
  - For example, if the domain D is the set of natural numbers {1,2,3,...}, and $\sigma(x) = 3$, then x is assigned the value 3.

Example Consider a domain $D = \{1, 2, 3\}$.

Interpretation:

- Constants: $a \to 1$
- Functions: $f \to$ the function defined by $f(x) = x + 1$ (assuming we interpret addition in a natural way)
- Predicates: $P \to \{(1), (2)\}$ (meaning $P(x)$ is true if x is 1 or 2)

Variable Assignment:

- $\sigma(x) = 2$

Formula Evaluation:

- Atomic Formula: $P(x)$
- Check $P(\sigma(x))$: Since $\sigma(x) = 2$, and 2 is in the set $\{1, 2\}, P(x)$ is true.
- Quantified Formula: $\forall x P(x)$
- Check $P(x)$ for all x in D :
- $x = 1 : P(1)$ is true.
- $x = 2 : P(2)$ is true.
- $x = 3 : P(3)$ is false.
- Since $P(3)$ is false, $\forall x P(x)$ is false.

The concept of interpretation is a crucial component of the semantics of any logical system. It shows the possibilities how to gives 'meanings' (semantic values) to parameters (nonlogical constants). In first-order logic

- name parameters (members of $\mathcal{F}(0)$) represent proper names;
- function parameters (members of $\mathcal{F}(n)$) represent operations;
- propositional parameters (members of $\mathcal{P}(0)$) represent propositions;
- one-argument predicate parameters (members of $\mathcal{P}(1)$) represent properties;
- n-argument predicate parameters (members of $\mathcal{P}(n), n \geq 1$) represent n-argument relations.

Definition: (Interpretation of first-order logic)

The ordered pair $\langle U, \rho \rangle$ is an interpretation of the language $L^{(1)}$ if

1. $U \neq \emptyset$ (i.e. U is a nonempty set);
2. $\mathrm{Dom}(\rho) = \mathrm{Con}$;
   (a) If $a \in \mathcal{F}(0)$, then $\rho(a) \in U$;
   (b) If $f \in \mathcal{F}(n) \quad (n \neq 0)$, then $\rho(f)$ is a function from $U^{(n)}$ to U;
   (c) If $p \in \mathcal{P}(0)$, then $\rho(p) \in \{0, 1\}$;
   (d) If $P \in \mathcal{P}(n) \quad (n \neq 0)$, then $\rho(P) \subseteq U^{(n)}$.

Definition: (Assignment in a given interpretation)

The function v is an assignment relying on the interpretation $\langle U, \rho \rangle$ if the followings hold:

1. $\mathrm{Dom}(v) = \mathrm{Var}$;
2. If $x \in Var$, then $v(x) \in U$.

Definition: (Modified assignment)

Let v be an assignment relying on the interpretation $\langle U, \rho \rangle, x \in Var$ and $u \in U$.

$$v[x : u](y) = \begin{cases} u, & \text{if } y = x; \\ v(y), & \text{otherwise.} \end{cases}$$

for all $y \in Var$.

Central Logical Concepts of Classical First-order Logic

Definition: (Model of a set of formulae)

Let $L^{(1)} = \langle LC, Var, \text{Con}, \text{Term}, \text{Form}\rangle$ be a first order language and $\Gamma \subseteq \text{Form}$ be a set of formulae. An ordered triple $\langle U, \rho, v\rangle$ is a model of the set Γ, if

1. $\langle U, \rho\rangle$ is an interpretation of $L^{(1)}$;
2. v is an assignment relying on $\langle U, \rho\rangle$;
3. $|A|_v^{\langle U,\rho\rangle} = 1$ for all $A \in \Gamma$.

Definition: (Model of a formula)

Let $L^{(1)} = \langle LC, Var, \text{Con}, \text{Term}, \text{Form}\rangle$ be a first order language and $A \in \text{Form}$ be a formula. A model of a formula A is the model of the singleton $\{A\}$.

Definition: (Satisfiability of a set of formulae)

Let $L^{(1)} = \langle LC, Var, \text{Con}, \text{Term}, \text{Form}\rangle$ be a first order language and $\Gamma \subseteq \text{Form}$ be a set of formulae. $\Gamma \subseteq \text{Form}$ is satisfiable if it has a model. (If there is an interpretation and an assignment in which all members of the set Γ are true.)

Definition: (Satisfiability of a formula)

Let $L^{(1)} = \langle LC, Var, \text{Con}, \text{Term}, \text{Form}\rangle$ be a first order language and $A \in \text{Form}$ be a formula. The formula A is satisfiable, if the singleton $\{A\}$ is satisfiable. (If there is an interpretation and an assignment in which the formula A is true.)

Remarks:

- A satisfiable set of formulas does not involve a logical contradiction; its formulae may be true together.
- A satisfiable formula may be true.
- If a set of formulas is satisfiable, then its members are satisfiable.
- But: all members of the set $\{P(a), \neg P(a)\}$ are satisfiable, and the set is not satisfiable.

Definition: (Unsatisfiability of a set of formulae)

Let $L^{(1)} = \langle LC, Var, \text{Con}, \text{Term}, \text{Form}\rangle$ be a first order language and $\Gamma \subseteq \text{Form}$ be a set of formulae. The set $\Gamma \subseteq \text{Form}$ is unsatisfiable if it is not satisfiable.

Remark:

An unsatisfiable set of formulae involves a logical contradiction. (Its members cannot be true together.)

Definition: (Unsatisfiability of a formula)

Let $L^{(1)} = \langle LC, Var, \text{Con}, \text{Term}, \text{Form}\rangle$ be a first order language and $A \in \text{Form}$ be a formula. The formula A is unsatisfiable if the singleton $\{A\}$ is unsatisfiable.

Remark:

An unsatisfiable formula involves a logical contradiction. (It cannot be true, i.e. it is false with respect to all interpretations and assignment.)

Definition: (Logical consequence of a set of formulae)

Let $L^{(1)} = \langle LC, Var, \text{Con, Term, Form} \rangle$ be a first order language, $\Gamma, \subseteq$ Form be a set of formulae and $A \in$ Form be a formula. The formula A is the logical consequence of the set of formulae Γ if the set $\Gamma \cup \{\neg A\}$ is unsatifiable. (Notation: $\Gamma \models A$)

Definition: (Logical consequence of a formula)

Let $L^{(1)} = \langle LC, Var, \text{Con, Term, Form} \rangle$ be a first order language, and $A, B \in$ Form be formulae. The formula B is the logical consequence of the formula A if $\{A\} \vDash B$. (Notation: $A \models B$)

Definition: (Validity of a formula)

Let $L^{(1)} = \langle LC, Var, \text{Con, Term, Form} \rangle$ be a first order language, and $A \in$ Form be a formula. The formula A is valid if $\emptyset \models A$. (Notation: $\models A$)

Definition: (Logical equivalence)

Let $L^{(1)} = \langle LC, Var, \text{Con, Term, Form} \rangle$ be a first order language, and $A, B \in$ Form be formulae. The formulae A and B are logically equivalent if $A \vDash B$ and $B \vDash A$. (Notation: $A \Leftrightarrow B$)

Normal Forms

- A base is a set of truth functors whose members can express all truth functors.
  - For example: $\{\neg, \supset), \{\neg, \wedge\}, \{\neg, \vee\}$
    1. $(p \wedge q) \Leftrightarrow \neg(p \supset \neg q)$
    2. $(p \vee q) \Leftrightarrow (\neg p \supset q)$
  - Truth functor Sheffer: $(p \mid q) \Leftrightarrow_{\text{def}} \neg(p \wedge q)$
  - Truth functor neither-nor: $(p \| q) \Leftrightarrow_{\text{def}} (\neg p \wedge \neg q)$
  - Remark: Singleton bases: $(p \mid q), (p \| q)$

Definition: Let $L^{(0)} = \langle LC, \text{ Con, Form } \rangle$ be a language of propositional logic and $p \in$ Con a propositional parameter. Then the formulae $p, \neg p$ are literals (where p is the base of the literals).

Definition: If the formula A is a literal or a conjunction of literals, then A is an elementary conjunction.

Definition: If the formula A is a literal or a disjunction of literals, the A is an elementary disjunction.

Remark: If the literals of an elementary conjunction/disjunction have different bases, then the elementary conjunction/disjunction represents an interpretation (or a family of interpretations).

Definition: A disjunction of elementary conjunctions is a disjunctive normal form.

Definition: A conjunction of elementary disjunctions is a conjunctive normal form.

Definition:

Let $L^{(1)} = \langle LC, Var$, Con, Term,Form $\rangle$ be a first order language and $A \in$ Form be a formula.

The formula A is prenex if

1. there is no quantifier in A or
2. the formula A is in the form $Q_1x_1Q_2x_2 \dots Q_nx_nB(n = 1, 2, \dots)$, where
   (a) there is no quantifier in the formula $B \in$ Form;
   (b) $x_1, x_2 \dots x_n \in$ Var are diffrent variables;
   (c) $Q_1, Q_2, \dots, Q_n \in \{\forall, \exists\}$ are quantifiers.

Sequent calculus

Definition:

Logical calculus is a formal system used to derive logical conclusions from premises through a series of rules and logical operations. It consists of:

- **Syntax**: The formal structure of expressions, including variables, constants, functions, predicates, connectives (like $\wedge, \vee, \rightarrow, \leftrightarrow \neg$), and quantifiers ($\forall, \exists$).
- **Inference Rules**: The logical rules that dictate how new statements (conclusions) can be derived from existing statements (premises).

Components

1. **Axioms**: Basic assumptions or self-evident truths.
2. **Inference Rules**: Procedures for deriving new statements from existing ones (e.g., Modus Ponens, Universal Instantiation).

Sequent Calculus

Definition

Sequent calculus is a logical system for proving the validity of logical statements through sequents.

A **sequent** is an expression of the form: $\Gamma \vdash \Delta$

Where Γ (the antecedent) and Δ (the consequent) are sets (or multisets) of formulas). The sequent means "if all formulas in Γ are true, then at least on formula in Δ is true"