Automated Reasoning

Luca Panariello

November 14, 2024

Contents

Chapter 1

Syntax and Semantics of First-Order Logic

Automated Reasoning is the study of algorithms and systems that allow computers to reason about logical statements.

In this chapter, we will introduce the syntax and semantics of First-Order Logic (FOL), which is the most widely used logic in the field of Automated Reasoning.

We will also introduce the concept of a *model* and the notion of *validity* of a logical statement.

Automated Reasoning is achieved by using symbol reasoning, which is the manipulation of symbols according to the rules of logic.

1.1 Syntax in First-Order Logic

Definition 1.1.1: Signature

A **signature** is a tuple $\Sigma = (\mathcal{C}, \mathcal{F}, \mathcal{P})$ where:

- $\mathcal{C}$ is a set of constant symbols.
- $\mathcal{F}$ is a set of function symbols.
- $\mathcal{P}$ is a set of predicate symbols.

The **constant** symbols denote individual elements (e.g. 0, 1, a, b, ...).

The **function** symbols denote functions that take a number of arguments and return a value (e.g. $+$, $\times$, f, g, ...).

The **predicate** symbols denote relations that take a number of arguments and return a truth value (e.g. $=$, $<$, P, Q, $\dots$).

The elements of $\mathcal{C}$, $\mathcal{F}$ and $\mathcal{P}$ are called *symbols*.

The arity of a function or predicate symbol is the number of arguments it takes.

The difference between a function and a predicate lies in their connection, infact a function returns a value while a predicate returns a truth value.

Example 1.1.2: Predicate vs Function

$f(x) = f(y)$ these functions are connected by the equality predicate $=$, while these predicates $P(x) \iff P(y)$ are connected by logical equivalence $\iff$

Remark 1.1.3: Functions of Predicate

A predicate P can be seen as a function f_P that returns a truth value. So introducing a constant symbol $\circ$ witch denotes "truth" than we can define f_P as:

$$f_P(x_1, \dots, x_n) = \circ \quad \text{if } P(x_1, \dots, x_n) \text{ is true}$$

where $\circ$ is added to the signature. $\Sigma' = \Sigma \cup \{\circ\}$

In FOL, there are logical connectives that are used to combine logical statements: $\neg$ (negation), $\wedge$ (conjunction), $\vee$ (disjunction), $\implies$ (implication), $\iff$ (equivalence).

Using a signature Σ and a set of variables $\mathcal{X}$, we can define the syntax of FOL.

Definition 1.1.4: Term

Let Σ be a signature and $\mathcal{X}$ a set of variables. A **term** is defined as follows:

- Every constant symbol $c \in \mathcal{C}$ is a term.
- Every variable $x \in \mathcal{X}$ is a term.
- Every n-ary function symbol $f \in \mathcal{F}$ with $t_1, \dots, t_n$ are terms, then $f(t_1, \dots, t_n)$ is a term.

Definition 1.1.5: Atom

Let Σ be a signature and $\mathcal{X}$ a set of variables. An **atom** is defined as follows:

- Every n-ary predicate symbol $P \in \mathcal{P}$ with $t_1, \dots, t_n$ are terms, then $P(t_1, \dots, t_n)$ is an atom.

Definition 1.1.6: Literal

A **literal** is an atom or the negation of an atom.

Definition 1.1.7: Formula

Let Σ be a signature and $\mathcal{X}$ a set of variables. A **formula** is defined as follows:

- Every atom is a formula.
- If F and G are formulas, then $\neg F$, $F \wedge G$, $F \vee G$, $F \implies G$, $F \iff G$ are formulas.
- If F is a formula and $x \in \mathcal{X}$ is a variable, then $\forall x.F$ and $\exists x.F$ are formulas.

From a formula we can destinguish the *free variables* and the *bound variables*. The free variables are the variables that are not bounded by a quantifier, while the bound variables are.

Example 1.1.8: Free and Bound Variables

Give the formula H:

$$f(x) = b \wedge \forall y.f(y) = b \implies f(f(y)) = b$$

We can define the free variables as $FV(H) = \{x\}$ and the bound variables as $BV(H) = \{y\}$. While b is a constant symbol.

Remark 1.1.9: Renaming Variables

Remark that a variable cannot be both free and bound at the same time. There is a issue with the renaming of variables: free variables cannot be renamed, while bound variables can, but the it cannot be renamed to a variable that is already in the formula.

[IMMAGINE VALIDITY PROBLEM IN FOL][NOTE IN SIGNATURE 3 OTTOBRE]

1.2 Semantics in First-Order Logic

Definition 1.2.1: Interpretation

Let Σ be a signature. An **interpretation** $\mathcal{I}$ of Σ is a tuple $\mathcal{I} = (\mathcal{D}, \Phi)$ where:

- $\mathcal{D}$ is a non-empty set called the *domain* of $\mathcal{I}$.
- Φ is a function that assigns to each symbol in Σ an element of $\mathcal{D}$.

The function Φ is defined as follows:

- $\forall c \in \mathcal{C}$, $\Phi(c) \in \mathcal{D}$.
- $\forall f \in \mathcal{F}$, $\Phi(f) : \mathcal{D}^n \to \mathcal{D}$.
- $\forall P \in \mathcal{P}$, $\Phi(P) : \mathcal{D}^n$.

Where n is the arity of the function or predicate symbol.

So an interpretation assigns a meaning to the symbols in the signature, there can be multiple interpretations for the same signature.

Example 1.2.2: Interpretation of Integers

Let $\Sigma = (\{a, b\}, \{f\}, \{R\})$ be a signature. An interpretation $\mathcal{I}$ of Σ can be defined as follows:

- $\mathcal{D} = \mathbb{Z}$.
- $\Phi(a) = -3$.
- $\Phi(b) = 3$.
- $\Phi(f) = + : \mathbb{Z}^2 \to \mathbb{Z}$ is the addition function.
- $\Phi(R) =\geq: \mathbb{Z}^2$ is the greater than or equal to predicate.

Example 1.2.3: Interpretation of Color

Let $\Sigma = (\{a, b, c\}, \emptyset, \{R\})$ be a signature. An interpretation $\mathcal{I}$ of Σ can be defined as follows:

- $\mathcal{D} = \{red, green, blue\}$.
- $\Phi(a) = red$.
- $\Phi(b) = blue$.
- $\Phi(c) = green$.
- $\Phi(R) = \{(red, blue), (red, red)\}$.

A term t is evaluated in an interpretation $\mathcal{I}$ as follows:

$$[t]_{\mathcal{I}} = \begin{cases} \Phi(c) & \text{if } t = c \in \mathcal{C} \text{ is a constant symbol} \\ \Phi(f)([t_1]_{\mathcal{I}}, \dots, [t_n]_{\mathcal{I}}) & \text{if } t = f(t_1, \dots, t_n) \in \mathcal{F} \text{ is a function symbol} \\ \beta(x) & \text{if } t = x \in \mathcal{X} \text{ is a variable} \end{cases}$$

Where n is the arity of the function symbol f and β is an assignment function that assigns a value to a variable.

1.3 Satifaction and Validity in First-Order Logic

A formula F is evaluated in an interpretation $\mathcal{I}$ and it is said to be *satisfied* in $\mathcal{I}$ if it evaluates to true, noted as $\mathcal{I} \models F$, otherwise it is said to be *unsatisfied* in $\mathcal{I}$, noted as $\mathcal{I} \not\models F$.

Definition 1.3.1: Satisfiable Formula

A formula F is **satisfiable** if $\exists \mathcal{I}$ interpretation such that $\mathcal{I} \models F$.

Definition 1.3.2: Valid Formula

A formula F is **valid** if $\forall \mathcal{I}$ interpretation such that $\mathcal{I} \models F$.
A valid formula F is indicated as $\models F$.

Remark 1.3.3: Unsatisfiable and Invalid Formulas

A formula F is **unsatisfiable** if $\forall \mathcal{I}$ interpretation such that $\mathcal{I} \not\models F$. A formula F is **invalid** if $\exists \mathcal{I}$ interpretation such that $\mathcal{I} \not\models F$.

Remark 1.3.4: Implication of Validity

Let F be a formula, than we can observe that:

F		$\neg F$
Satisfiable	$\implies$	Invalid
Valid	$\implies$	Unsatisfiable
Invalid	$\implies$	Satisfiable
Unsatisfiable	$\implies$	Valid

Also the satisfiable relation can be defined as follows: Let F, G, H be formulas and $\mathcal{I}$ be an interpretation, than:

- $\mathcal{I} \models \neg F$ if $\mathcal{I} \not\models F$.
- $\mathcal{I} \models F \land G$ if $\mathcal{I} \models F \land \mathcal{I} \models G$.
- $\mathcal{I} \models F \lor G$ if $\mathcal{I} \models F \lor \mathcal{I} \models G$.
- $\mathcal{I} \models F \implies G$ if $\mathcal{I} \not\models F \implies \mathcal{I} \models G$.
- $\mathcal{I} \models F \iff G$ if $\mathcal{I} \models F \iff \mathcal{I} \models G$.
- $\mathcal{I} \models \forall x.F$ if $\forall d \in \mathcal{D} : \mathcal{I} \models_{\beta[x \to d]} G$
- $\mathcal{I} \models \exists x.F$ if $\exists d \in \mathcal{D} : \mathcal{I} \models_{\beta[x \to d]} G$

Where $\mathcal{I} \models_{\beta[x \to d]} G$ means that the formula G is satisfied in $\mathcal{I}$ with the assignment function β that assigns the value d to the variable x.

$$\beta[x \to d](y) = \begin{cases} d & \text{if } y = x \\ \beta(y) & \text{otherwise} \end{cases} \quad \forall y \in \mathcal{X}$$

1.4 Theories in First-Order Logic

A **theory** formalize structures in a specific domain of interest, and help us reason about the properties of these structures. It really useful in verification.

Will be introduced some definitions that concerns theories in FOL.

Definition 1.4.1: Theory

A **theory** $\mathcal{T}$ is definted as a tuple $\mathcal{T} = (\Sigma, \mathcal{A})$ where:

- Σ is a signature.
- $\mathcal{A}$ is a set of formulas called *axioms*, with only elements of the signature.

Definition 1.4.2: Sigma-Forumla

A formula F is a **Σ-formula** if it contains symbols in the signature Σ, as well as the logical connectives, quantifiers and variables.

Definition 1.4.3: Theory-Interpretation

If $\mathcal{I}$ is an interpretation of Σ, then $\mathcal{I}$ is a **$\mathcal{T}$-interpretation** of a theory $\mathcal{T} = (\Sigma, \mathcal{A})$ if $\mathcal{I} \vDash \mathcal{A}$.

Definition 1.4.4: Theory-Satisfiable Formula

Let $\mathcal{T} = (\Sigma, \mathcal{A})$ be a theory. And F be a Σ-formula. If $\exists \mathcal{I}$ interpretation of Σ:

$$\mathcal{I} \vDash \mathcal{A} \land \mathcal{I} \vDash F$$

Which means that F is satisfied in $\mathcal{I}$ and $\mathcal{I}$ is a $\mathcal{T}$-interpretation.
So F is **$\mathcal{T}$-satisfiable** in the theory $\mathcal{T}$.

Definition 1.4.5: Theory-Vaild Formula

Let $\mathcal{T} = (\Sigma, \mathcal{A})$ be a theory. And F be a Σ-formula. If $\forall \mathcal{I}$ interpretation of Σ:

$$\mathcal{I} \vDash \mathcal{A} \implies \mathcal{I} \vDash F$$

Which means F is valid ($\vDash F$) in the theory $\mathcal{T}$ if every $\mathcal{T}$-interpretation satisfies F.
Then F is a **$\mathcal{T}$-valid formula**, also noted as $\mathcal{T} \vDash F$.

1.4.1 Theory of Equality

[NEW SECTION]

1.4.2 Logical Consequence

Logical Consequenceis the relation between a set of formulas (assumptions) and a formula (conjecture), where the conjecture is true if the assumptions are true.

Let H be a set of formulas, called "*assumption*", and φ be a formula, called "*conjecture*":

We have that $H \models \varphi$ or equivalently $\models H \implies \varphi$, which means that φ is **logica consequence** of H, then:

$$\forall \mathcal{I} \text{ interpretation } : \mathcal{I} \models H \iff \mathcal{I} \models \varphi \iff H \cup \{\neg\varphi\} \text{ is unsatisfiable}$$

The last coimplication derives from the fact that $\neg\varphi$ is the negation of the conjecture, so if we find an interpretation that satisfies all the formulas in H then it must satisfy the conjecture, making the satisfaction of $\neg\varphi$ impossible.

We need a way to determine if a formula is a logical consequence of a set of formulas: we can build a decision procedure that checks if the set of formulas is unsatisfiable, in particular the procedure checks if $H \cup \{\neg\varphi\} \models \bot$.