

# First-Order Logic

(Based on [Gallier 1986], [Goubault-Larrecq and Mackie 1997], and [Huth and Ryan 2004])

Yih-Kuen Tsay

Department of Information Management  
National Taiwan University

# Introduction

- 🌐 Logic concerns two concepts:
  - ☀️ **truth** (in a specific or general context)
  - ☀️ **provability** (of truth from assumed truth)
- 🌐 **Formal (symbolic) logic** approaches logic by rules for manipulating symbols:
  - ☀️ **Syntax** rules: for writing statements (or formulae).  
(There are also **semantic** rules determining whether a statement is true or false in a context or mathematical structure.)
  - ☀️ **Inference** rules: for obtaining true statements from other true statements.
- 🌐 We shall introduce two main branches of formal logic:
  - ☀️ **propositional logic**
  - ☀️ **first-order logic** (predicate logic/calculus)
- 🌐 This lecture covers **first-order logic**.

# Why We Need Logic

- 🌐 Correctness of software hinges on a **precise** statement of its **requirements**.
- 🌐 Logical formulae give the most precise kind of statements about software requirements.
- 🌐 The fact that “a software program satisfies a requirement” is very much the same as “a mathematical structure satisfies a logical formula”:


$$\text{prog} \models \text{req} \text{ vs. } M \models \varphi$$



- 🌐 To **prove** (formally verify) that a software program is correct, one may utilize the kind of inferences seen in formal logic.
- 🌐 The verification may be done manually, semi-automatically, or fully automatically.

# Predicates


- 🌐 A *predicate* is a “parameterized” statement that, when supplied with actual arguments, is either *true* or *false* such as the following:
  - ☀️ Leslie is a teacher.
  - ☀️ Chris is a teacher.
  - ☀️ Leslie is a pop singer.
  - ☀️ Chris is a pop singer.
- 🌐 Like propositions, simplest (*atomic*) predicates may be combined to form *compound* predicates.

# Inferences

 We are given the following assumptions:

-  *For any* person, *either* the person is not a teacher *or* the person is not rich.
-  *For any* person, *if* the person is a pop singer, *then* the person is rich.

 We wish to conclude the following:

-  *For any* person, *if* the person is a teacher, *then* the person is not a pop singer.

# Symbolic Predicates

- Like propositions, predicates are represented by *symbols*.
  - $p(x)$ :  $x$  is a teacher.
  - $q(x)$ :  $x$  is rich.
  - $r(y)$ :  $y$  is a pop singer.
- Compound predicates can be expressed:
  - For all  $x$ ,  $r(x) \rightarrow q(x)$ : For any person, if the person is a pop singer, then the person is rich.
  - For all  $y$ ,  $p(y) \rightarrow \neg r(y)$ : For any person, if the person is a teacher, then the person is not a pop singer.

# Symbolic Inferences

🌐 We are given the following assumptions:

☀ For all  $x$ ,  $\neg p(x) \vee \neg q(x)$ .

☀ For all  $x$ ,  $r(x) \rightarrow q(x)$ .

🌐 We wish to conclude the following:

☀ For all  $x$ ,  $p(x) \rightarrow \neg r(x)$ .

🌐 To check the correctness of the inference above, we ask:

Is  $((\text{for all } x, \neg p(x) \vee \neg q(x)) \wedge (\text{for all } x, r(x) \rightarrow q(x))) \rightarrow$   
 $(\text{for all } x, p(x) \rightarrow \neg r(x))$  valid?

## 🌐 Logical symbols:

- ☀️ A countable set  $V$  of *variables*:  $x, y, z, \dots$ ;
- ☀️ *Logical connectives* (operators):  $\neg, \wedge, \vee, \rightarrow, \leftrightarrow, \perp, \forall$  (for all),  $\exists$  (there exists);
- ☀️ Auxiliary symbols: “(”, “)”.

## 🌐 Non-logical symbols:

- ☀️ A countable set of *function symbols* with associated ranks (arities);
- ☀️ A countable set of *constants* (which may be seen as functions with rank 0);
- ☀️ A countable set of *predicate symbols* with associated ranks (arities);

🌐 We refer to a first-order language as *Language L*, where  $L$  is the set of non-logical symbols (e.g.,  $\{+, 0, 1, <\}$ ). The set  $L$  is usually referred to as the *signature* of the first-order language.



# Syntax (cont.)

## 🌐 Terms:

- ☀ Every *constant* and every *variable* is a term.
- ☀ If  $t_1, t_2, \dots, t_k$  are terms and  $f$  is a  $k$ -ary function symbol ( $k > 0$ ), then  $f(t_1, t_2, \dots, t_k)$  is a term.

## 🌐 Atomic formulae:

- ☀ Every *predicate symbol* of 0-arity is an atomic formula and so is  $\perp$ .
- ☀ If  $t_1, t_2, \dots, t_k$  are terms and  $p$  is a  $k$ -ary predicate symbol ( $k > 0$ ), then  $p(t_1, t_2, \dots, t_k)$  is an atomic formula.

## 🌐 For example, consider Language $\{+, 0, 1, <\}$ .

- ☀  $0, x, x + 1, x + (x + 1)$ , etc. are terms.
- ☀  $0 < 1, x < (x + 1)$ , etc. are atomic formulae.

# Syntax (cont.)

- 🌐 Formulae:
  - ☀️ Every **atomic formula** is a formula.
  - ☀️ If  $A$  and  $B$  are formulae, then so are  $\neg A$ ,  $(A \wedge B)$ ,  $(A \vee B)$ ,  $(A \rightarrow B)$ , and  $(A \leftrightarrow B)$ .
  - ☀️ If  $x$  is a variable and  $A$  is a formula, then so are  $\forall xA$  and  $\exists xA$ .
- 🌐 First-order logic *with equality* includes equality ( $=$ ) as an additional logical symbol, which behaves like a predicate symbol.
- 🌐 Example formulae in Language  $\{+, 0, 1, <\}$ :
  - ☀️  $(0 < x) \vee (x < 1)$
  - ☀️  $\forall x(\exists y(x + y = 0))$

## Syntax (cont.)

- 🌐 We may give the logical connectives different binding powers, or **precedences**, to avoid excessive parentheses, usually in this order:

$$\neg, \{\forall, \exists\}, \{\wedge, \vee\}, \rightarrow, \leftrightarrow .$$

For example,  $(A \wedge B) \rightarrow C$  becomes  $A \wedge B \rightarrow C$ .

- 🌐 Common abbreviations:




- ☀  $x = y = z$  means  $x = y \wedge y = z$ .
- ☀  $p \rightarrow q \rightarrow r$  means  $p \rightarrow (q \rightarrow r)$ . Implication associates to the right, so do other logical symbols.
- ☀  $\forall x, y, z A$  means  $\forall x(\forall y(\forall z A))$ .

# Free and Bound Variables

- In a formula  $\forall xA$  (or  $\exists xA$ ), the variable  $x$  is *bound* by the quantifier  $\forall$  (or  $\exists$ ).
- A *free* variable is one that is not bound.
- The same variable may have both a free and a bound occurrence.
- For example, consider  $(\forall x(R(x, \underline{y}) \rightarrow P(x)) \wedge \forall y(\neg R(\underline{x}, y) \wedge \forall xP(x)))$ .  
The underlined occurrences of  $x$  and  $y$  are free, while others are bound.
- A formula is *closed*, also called a *sentence*, if it does not contain a free variable.

# Free Variables Formally Defined

For a term  $t$ , the set  $FV(t)$  of free variables of  $t$  is defined inductively as follows:

-   $FV(x) = \{x\}$ , for a variable  $x$ ;
-   $FV(c) = \emptyset$ , for a constant  $c$ ;
-   $FV(f(t_1, t_2, \dots, t_n)) = FV(t_1) \cup FV(t_2) \cup \dots \cup FV(t_n)$ , for an  $n$ -ary function  $f$  applied to  $n$  terms  $t_1, t_2, \dots, t_n$ .

# Free Variables Formally Defined (cont.)

For a formula  $A$ , the set  $FV(A)$  of free variables of  $A$  is defined inductively as follows:

- $FV(P(t_1, t_2, \dots, t_n)) = FV(t_1) \cup FV(t_2) \cup \dots \cup FV(t_n)$ , for an  $n$ -ary predicate  $P$  applied to  $n$  terms  $t_1, t_2, \dots, t_n$ ;
- $FV(t_1 = t_2) = FV(t_1) \cup FV(t_2)$ ;
- $FV(\neg B) = FV(B)$ ;
- $FV(B * C) = FV(B) \cup FV(C)$ , where  $*$   $\in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ ;
- $FV(\perp) = \emptyset$ ;
- $FV(\forall x B) = FV(B) - \{x\}$ ;
- $FV(\exists x B) = FV(B) - \{x\}$ .

# Bound Variables Formally Defined

For a formula  $A$ , the set  $BV(A)$  of bound variables in  $A$  is defined inductively as follows:

- $BV(P(t_1, t_2, \dots, t_n)) = \emptyset$ , for an  $n$ -ary predicate  $P$  applied to  $n$  terms  $t_1, t_2, \dots, t_n$ ;
- $BV(t_1 = t_2) = \emptyset$ ;
- $BV(\neg B) = BV(B)$ ;
- $BV(B * C) = BV(B) \cup BV(C)$ , where  $* \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ ;
- $BV(\perp) = \emptyset$ ;
- $BV(\forall x B) = BV(B) \cup \{x\}$ ;
- $BV(\exists x B) = BV(B) \cup \{x\}$ .




# First-Order Structures

- 🌐 A first-order structure  $\mathcal{M}$  is a pair  $(M, I)$ , where
  - ☀️  $M$  (a non-empty set) is the *domain* of the structure, and
  - ☀️  $I$  is the *interpretation function*, that assigns functions and predicates over  $M$  to the function and predicate symbols.
- 🌐 An interpretation may be represented by simply listing the functions and predicates.
- 🌐 For instance,  $(Z, \{+_Z, 0_Z\})$  is a structure for the language  $\{+, 0\}$ . The subscripts are omitted, as  $(Z, \{+, 0\})$ , when no confusion may arise.



- Since a formula may contain free variables, its truth value depends on the specific values that are assigned to these variables.
- Given a first-order language and a structure  $\mathcal{M} = (M, I)$ , an *assignment* is a function from the set of variables to  $M$ .
- The structure  $\mathcal{M}$  along with an assignment  $s$  determines the truth value of a formula  $A$ , denoted as  $A_{\mathcal{M}}[s]$ .
- For example,  $(x + 0 = x)_{(Z, \{+, 0\})}[x := 1]$  evaluates to  $T$ .

# Semantics (cont.)

-  We say  $\mathcal{M}, s \models A$  when  $A_{\mathcal{M}}[s]$  is  $T$  (true) and  $\mathcal{M}, s \not\models A$  otherwise.
-  Alternatively,  $\models$  may be defined as follows (propositional part is as in propositional logic):
  - $\mathcal{M}, s \models \forall x A \iff \mathcal{M}, s[x := m] \models A$  for all  $m \in M$ .
  - $\mathcal{M}, s \models \exists x A \iff \mathcal{M}, s[x := m] \models A$  for some  $m \in M$ .
 where  $s[x := m]$  denotes an updated assignment  $s'$  from  $s$  such that  $s'(y) = s(y)$  for  $y \neq x$  and  $s'(x) = m$ .
-  For example,  $(Z, \{+, 0\}), s \models \forall x(x + 0 = x)$  holds, since  $(Z, \{+, 0\}), s[x := m] \models x + 0 = x$  for all  $m \in Z$ .

# What about Types

- 🌐 Ordinary first-order formulae are interpreted over a single domain of discourse (the universe).
- 🌐 A variant of first-order logic, called **many-sorted** (or typed) first-order logic, allows variables of different **sorts** (which correspond to partitions of the universe).
- 🌐 When the number of sorts is finite, one can emulate sorts by introducing additional **unary predicates** in the ordinary first-order logic.
  - ☀ Suppose there are two sorts.
  - ☀ We introduce two new unary predicates  $P_1$  and  $P_2$ .
  - ☀ We then stipulate that
$$\forall x(P_1(x) \vee P_2(x)) \wedge \neg(\exists x(P_1(x) \wedge P_2(x))).$$
  - ☀ For example,  $\exists x(P_1(x) \wedge \varphi(x))$  means that there is an element of the first sort satisfying  $\varphi$ ;  $\forall x(P_1(x) \rightarrow \psi(x))$  means that every element of the first sort satisfies  $\psi$ .

# Satisfiability and Validity

- 🌐 A formula  $A$  is *satisfiable in  $\mathcal{M}$*  if there is an assignment  $s$  such that  $\mathcal{M}, s \models A$ .
- 🌐 A formula  $A$  is *valid in  $\mathcal{M}$* , denoted  $\mathcal{M} \models A$ , if  $\mathcal{M}, s \models A$  for every assignment  $s$ .
- 🌐 For instance,  $\forall x(x + 0 = x)$  is valid in  $(\mathbb{Z}, \{+, 0\})$ .
- 🌐  $\mathcal{M}$  is called a *model* of  $A$  if  $A$  is valid in  $\mathcal{M}$ .
- 🌐 A formula  $A$  is *valid* if it is valid in every structure, denoted  $\models A$ .

# Relating the Quantifiers

## Lemma

$$\models \neg \forall x A \leftrightarrow \exists x \neg A$$

$$\models \neg \exists x A \leftrightarrow \forall x \neg A$$

$$\models \forall x A \leftrightarrow \neg \exists x \neg A$$

$$\models \exists x A \leftrightarrow \neg \forall x \neg A$$

Note: These equivalences show that, with the help of negation, either quantifier can be expressed by the other.

# Substitutions

- 🌐 Let  $t$  be a term and  $A$  a formula.
- 🌐 The result of substituting  $t$  for a free variable  $x$  in  $A$  is denoted by  $A[t/x]$ .
- 🌐 Consider  $A = \forall x(P(x) \rightarrow Q(x, f(y)))$ .
  - ☀ When  $t = g(y)$ ,  $A[t/y] = \forall x(P(x) \rightarrow Q(x, f(g(y))))$ .
  - ☀ For any  $t$ ,  $A[t/x] = \forall x(P(x) \rightarrow Q(x, f(y))) = A$ , since there is no free occurrence of  $x$  in  $A$ .
- 🌐 A substitution is *admissible* if no free variable of  $t$  would become bound (be captured by a quantifier) after the substitution.
- 🌐 For example, when  $t = g(x, y)$ ,  $A[t/y]$  is not admissible, as the free variable  $x$  of  $t$  would become bound.

## Substitutions (cont.)

- Suppose we change the bound variable  $x$  in  $A$  to  $z$  and obtain another formula  $A' = \forall z(P(z) \rightarrow Q(z, f(y)))$ .
- Intuitively,  $A'$  and  $A$  should be equivalent (under any reasonable semantics). (Technically, the two formulae  $A$  and  $A'$  are said to be  *$\alpha$ -equivalent*.)
- We can avoid the capture in  $A[g(x, y)/y]$  by renaming the bound variable  $x$  to  $z$  and the result of the substitution then becomes  $A'[g(x, y)/y] = \forall z(P(z) \rightarrow Q(z, f(g(x, y))))$ .
- So, in principle, we **can make every substitution admissible** while preserving the semantics.

# Substitutions Formally Defined

Let  $s$  and  $t$  be terms. The result of substituting  $t$  in  $s$  for a variable  $x$ , denoted  $s[t/x]$ , is defined inductively as follows:

- 1.  $x[t/x] = t$ ;
- 2.  $y[t/x] = y$ , for a variable  $y$  that is not  $x$ ;
- 3.  $c[t/x] = c$ , for a constant  $c$ ;
- 4.  $f(t_1, t_2, \dots, t_n)[t/x] = f(t_1[t/x], t_2[t/x], \dots, t_n[t/x])$ , for an  $n$ -ary function  $f$  applied to  $n$  terms  $t_1, t_2, \dots, t_n$ .



# Substitutions Formally Defined (cont.)

For a formula  $A$ ,  $A[t/x]$  is defined inductively as follows:

- 🌐  $P(t_1, t_2, \dots, t_n)[t/x] = P(t_1[t/x], t_2[t/x], \dots, t_n[t/x])$ , for an  $n$ -ary predicate  $P$  applied to  $n$  terms  $t_1, t_2, \dots, t_n$ ;
- 🌐  $(t_1 = t_2)[t/x] = (t_1[t/x] = t_2[t/x])$ ;
- 🌐  $(\neg B)[t/x] = \neg B[t/x]$ ;
- 🌐  $(B * C)[t/x] = (B[t/x] * C[t/x])$ , where  $*$   $\in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ ;
- 🌐  $\perp[t/x] = \perp$ ;
- 🌐  $(\forall x B)[t/x] = (\forall x B)$ ;
- 🌐  $(\forall y B)[t/x] = (\forall y B[t/x])$ , if variable  $y$  is not  $x$ ;
- 🌐  $(\exists x B)[t/x] = (\exists x B)$ ;
- 🌐  $(\exists y B)[t/x] = (\exists y B[t/x])$ , if variable  $y$  is not  $x$ ;

# Quantifier Rules of Natural Deduction

$$\frac{\Gamma \vdash A[y/x]}{\Gamma \vdash \forall x A} (\forall I)$$

$$\frac{\Gamma \vdash \forall x A}{\Gamma \vdash A[t/x]} (\forall E)$$

$$\frac{\Gamma \vdash A[t/x]}{\Gamma \vdash \exists x A} (\exists I)$$

$$\frac{\Gamma \vdash \exists x A \quad \Gamma, A[y/x] \vdash B}{\Gamma \vdash B} (\exists E)$$

In the rules above, we assume that all substitutions are admissible and  $y$  does not occur free in  $\Gamma$  or  $A$ .

# A Proof in First-Order ND

Below is a partial proof of the validity of  $\forall x(\neg p(x) \vee \neg q(x)) \wedge \forall x(r(x) \rightarrow q(x)) \rightarrow \forall x(p(x) \rightarrow \neg r(x))$  in ND, where  $\gamma$  denotes  $\forall x(\neg p(x) \vee \neg q(x)) \wedge \forall x(r(x) \rightarrow q(x))$ .

$$\begin{array}{c}
 \vdots \\
 \hline
 \gamma, p(y), r(y) \vdash r(y) \rightarrow q(y) \quad \gamma, p(y), r(y) \vdash r(y) \quad (Ax) \\
 \hline
 \gamma, p(y), r(y) \vdash q(y) \quad (\rightarrow E) \\
 \vdots \\
 \hline
 \forall x(\neg p(x) \vee \neg q(x)) \wedge \forall x(r(x) \rightarrow q(x)), p(y), r(y) \vdash q(y) \wedge \neg q(y) \quad (\wedge I) \\
 \hline
 \forall x(\neg p(x) \vee \neg q(x)) \wedge \forall x(r(x) \rightarrow q(x)), p(y) \vdash \neg r(y) \quad (\neg I) \\
 \hline
 \forall x(\neg p(x) \vee \neg q(x)) \wedge \forall x(r(x) \rightarrow q(x)), p(y) \vdash \neg r(y) \quad (\rightarrow I) \\
 \hline
 \forall x(\neg p(x) \vee \neg q(x)) \wedge \forall x(r(x) \rightarrow q(x)) \vdash p(y) \rightarrow \neg r(y) \quad (\forall I) \\
 \hline
 \forall x(\neg p(x) \vee \neg q(x)) \wedge \forall x(r(x) \rightarrow q(x)) \vdash \forall x(p(x) \rightarrow \neg r(x)) \quad (\rightarrow I) \\
 \hline
 \vdash \forall x(\neg p(x) \vee \neg q(x)) \wedge \forall x(r(x) \rightarrow q(x)) \rightarrow \forall x(p(x) \rightarrow \neg r(x)) \quad (\rightarrow I)
 \end{array}$$

# Equality Rules of Natural Deduction

Let  $t, t_1, t_2$  be arbitrary terms; again, assume all substitutions are admissible.

$$\frac{}{\Gamma \vdash t = t} (= I) \qquad \frac{\Gamma \vdash t_1 = t_2 \quad \Gamma \vdash A[t_1/x]}{\Gamma \vdash A[t_2/x]} (= E)$$

Note: The  $=$  sign is part of the object language, not a meta symbol.

# Soundness and Completeness

Let System  $ND$  also include the quantifier rules.

## Theorem

System  $ND$  is *sound*, i.e., if a sequent  $\Gamma \vdash C$  is *provable* in  $ND$ , then  $\Gamma \vdash C$  is *valid*.

## Theorem

System  $ND$  is *complete*, i.e., if a sequent  $\Gamma \vdash C$  is *valid*, then  $\Gamma \vdash C$  is *provable* in  $ND$ .

Note: assume *no equality* in the logic language.

## Theorem

*For any (possibly infinite) set  $\Gamma$  of formulae, if every finite non-empty subset of  $\Gamma$  is satisfiable then  $\Gamma$  is satisfiable.*

# Consistency

Recall that a set  $\Gamma$  of formulae is *consistent* if there exists some formula  $B$  such that the sequent  $\Gamma \vdash B$  is not provable. Otherwise,  $\Gamma$  is *inconsistent*.

## Lemma

*For System ND, a set  $\Gamma$  of formulae is **inconsistent** if and only if there is some formula  $A$  such that both  $\Gamma \vdash A$  and  $\Gamma \vdash \neg A$  are provable.*

## Theorem

*For System ND, a set  $\Gamma$  of formulae is **satisfiable** if and only if  $\Gamma$  is **consistent**.*









- Assume a fixed first-order language.
- A set  $S$  of sentences is closed under provability if

$$S = \{A \mid A \text{ is a sentence and } S \vdash A \text{ is provable}\}.$$

- A set of sentences is called a *theory* if it is closed under provability.
- A theory is typically represented by a smaller set of sentences, called its *axioms*.






# Group as a First-Order Theory

-  The set of non-logical symbols is  $\{\cdot, e\}$ , where  $\cdot$  is a binary function (operation) and  $e$  is a constant (the identity).
-  Axioms:
  -   $\forall a, b, c(a \cdot (b \cdot c) = (a \cdot b) \cdot c)$  (Associativity)
  -   $\forall a(a \cdot e = e \cdot a = a)$  (Identity)
  -   $\forall a(\exists b(a \cdot b = b \cdot a = e))$  (Inverse)
-   $(\mathbb{Z}, \{+, 0\})$  and  $(\mathbb{Q} \setminus \{0\}, \{\times, 1\})$  are models of the theory.
-  Additional axiom for Abelian groups:
  -   $\forall a, b(a \cdot b = b \cdot a)$  (Commutativity)

- 🌐 A *theorem* is just a statement (sentence) in a theory (a set of sentences).
- 🌐 For example, the following are theorems in Group theory:
  - ☀  $\forall a \forall b \forall c ((a \cdot b = a \cdot c) \rightarrow b = c)$ .
  - ☀  $\forall a \forall b \forall c (((a \cdot b = e) \wedge (b \cdot a = e) \wedge (a \cdot c = e) \wedge (c \cdot a = e)) \rightarrow b = c)$ ,  
which says that every element has a unique inverse.

# References

-  J.H. Gallier. *Logic for Computer Science: Foundations of Automatic Theorem Proving*, Harper & Row Publishers, 1985.
-  J. Goubault-Larrecq and I. Mackie. *Proof Theory and Automated Deduction*, Kluwer Academic Publishers, 1997.
-  M. Huth and M. Ryan. *Logic in Computer Science: Modelling and Reasoning about Systems*, Cambridge University Press, 2004.