Automated Reasoning - WS 2013

Lecture 3: First-Order Logic
Syntax, Semantics, Normal Forms

Mădălina Erașcu   Tudor Jebelean

Research Institute for Symbolic Computation,
Johannes Kepler University, Linz, Austria

{merascu,tjebelea}@risc.jku.at

October 23, 2013

Outline

Syntax

Semantics

Equivalences of Formulas

Normal Forms

(Un)Satisfiability & (In)Validity

Outline

Syntax

Semantics

Equivalences of Formulas

Normal Forms

(Un)Satisfiability & (In)Validity

Syntax

The language of FOL consists in terms and formulas.

Terms are defined recursively as follows:

1. A constant is a term.
2. A variable is a term.
3. If f is an n-place function symbol, and t_1, ..., t_n are terms then $f[t_1, ..., t_n]$ is a term.
4. All terms are generated by applying the above rules.

If P is an n-place predicate symbol and t_1, ..., t_n are terms then $P[t_1, ..., t_n]$ is an atom.

An atom is $\mathbb{T}$, $\mathbb{F}$, or an n-ary predicate applied to n terms.

A literal is an atom or its negation.

Syntax

The language of FOL consists in terms and formulas.

Terms are defined recursively as follows:

1. A constant is a term.
2. A variable is a term.
3. If f is an n-place function symbol, and t_1, ..., t_n are terms then $f[t_1, ..., t_n]$ is a term.
4. All terms are generated by applying the above rules.

If P is an n-place predicate symbol and t_1, ..., t_n are terms then $P[t_1, ..., t_n]$ is an atom.

An atom is $\mathbb{T}$, $\mathbb{F}$, or an n-ary predicate applied to n terms.

A literal is an atom or its negation.

Syntax

The language of FOL consists in terms and formulas.

Terms are defined recursively as follows:

1. A constant is a term.
2. A variable is a term.
3. If f is an n-place function symbol, and t_1, ..., t_n are terms then $f[t_1, ..., t_n]$ is a term.
4. All terms are generated by applying the above rules.

If P is an n-place predicate symbol and t_1, ..., t_n are terms then $P[t_1, ..., t_n]$ is an atom.

An atom is $\mathbb{T}$, $\mathbb{F}$, or an n-ary predicate applied to n terms.

A literal is an atom or its negation.

Syntax

The language of FOL consists in terms and formulas.

Terms are defined recursively as follows:

1. A constant is a term.
2. A variable is a term.
3. If f is an n-place function symbol, and t_1, ..., t_n are terms then $f[t_1, ..., t_n]$ is a term.
4. All terms are generated by applying the above rules.

If P is an n-place predicate symbol and t_1, ..., t_n are terms then $P[t_1, ..., t_n]$ is an atom.

An atom is $\mathbb{T}$, $\mathbb{F}$, or an n-ary predicate applied to n terms.

A literal is an atom or its negation.

Syntax

The language of FOL consists in terms and formulas.

Terms are defined recursively as follows:

1. A constant is a term.
2. A variable is a term.
3. If f is an n-place function symbol, and t_1, ..., t_n are terms then $f[t_1, ..., t_n]$ is a term.
4. All terms are generated by applying the above rules.

If P is an n-place predicate symbol and t_1, ..., t_n are terms then $P[t_1, ..., t_n]$ is an atom.

An atom is $\mathbb{T}$, $\mathbb{F}$, or an n-ary predicate applied to n terms.

A literal is an atom or its negation.

Syntax

The language of FOL consists in terms and formulas.

Terms are defined recursively as follows:

1. A constant is a term.
2. A variable is a term.
3. If f is an n-place function symbol, and t_1, ..., t_n are terms then $f[t_1, ..., t_n]$ is a term.
4. All terms are generated by applying the above rules.

If P is an n-place predicate symbol and t_1, ..., t_n are terms then $P[t_1, ..., t_n]$ is an atom.

An atom is $\mathbb{T}$, $\mathbb{F}$, or an n-ary predicate applied to n terms.

A literal is an atom or its negation.

Syntax

The language of FOL consists in terms and formulas.

Terms are defined recursively as follows:

1. A constant is a term.
2. A variable is a term.
3. If f is an n-place function symbol, and t_1, ..., t_n are terms then $f[t_1, ..., t_n]$ is a term.
4. All terms are generated by applying the above rules.

If P is an n-place predicate symbol and t_1, ..., t_n are terms then $P[t_1, ..., t_n]$ is an atom.

An atom is $\mathbb{T}$, $\mathbb{F}$, or an n-ary predicate applied to n terms.

A literal is an atom or its negation.

Syntax

The language of FOL consists in terms and formulas.

Terms are defined recursively as follows:

1. A constant is a term.
2. A variable is a term.
3. If f is an n-place function symbol, and t_1, ..., t_n are terms then $f[t_1, ..., t_n]$ is a term.
4. All terms are generated by applying the above rules.

If P is an n-place predicate symbol and t_1, ..., t_n are terms then $P[t_1, ..., t_n]$ is an atom.

An atom is $\mathbb{T}$, $\mathbb{F}$, or an n-ary predicate applied to n terms.

A literal is an atom or its negation.

Syntax

The language of FOL consists in terms and formulas.

Terms are defined recursively as follows:

1. A constant is a term.
2. A variable is a term.
3. If f is an n-place function symbol, and t_1, ..., t_n are terms then $f[t_1, ..., t_n]$ is a term.
4. All terms are generated by applying the above rules.

If P is an n-place predicate symbol and t_1, ..., t_n are terms then $P[t_1, ..., t_n]$ is an atom.

An atom is $\mathbb{T}$, $\mathbb{F}$, or an n-ary predicate applied to n terms.

A literal is an atom or its negation.

Syntax (cont'd)

Formulas are defined as follows:

1. An atom is a formula.
2. If F and G are formulas then $\neg F$, $F \vee G$, $F \wedge G$, $F \implies G$, and $F \iff G$ are formulas.
3. If F is a formula and x is a free variable, then $\underset{x}{\forall} F[x]$ and $\underset{x}{\exists} F[x]$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

A variable is bound in formula $F[x]$ if there is an occurrence of x in the scope of a binding quantifier $\underset{x}{\forall}$ or $\underset{x}{\exists}$.

A variable is free in formula $F[x]$ if there is an occurrence of x that is not bound by any quantifier.

Examples: Identify constants, variables (free, bound), quantifiers, functions, predicates, atoms, terms, formulas from the bellow

1. $\underset{x}{\forall}\ x + 1 \geq x$
2. $\neg \left(\underset{x}{\exists}\ E[0, f[x]] \right)$
3. $\underset{x}{\forall} \underset{y}{\exists} \left(E[y, f[x]] \ \wedge\ \underset{z}{\forall} (E[z, f[x]] \Rightarrow E[y, z]) \right)$

Syntax (cont'd)

Formulas are defined as follows:

1. An atom is a formula.
2. If F and G are formulas then $\neg F$, $F \vee G$, $F \wedge G$, $F \Longrightarrow G$, and $F \iff G$ are formulas.
3. If F is a formula and x is a free variable, then $\underset{x}{\forall} F[x]$ and $\underset{x}{\exists} F[x]$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

A variable is bound in formula $F[x]$ if there is an occurrence of x in the scope of a binding quantifier $\underset{x}{\forall}$ or $\underset{x}{\exists}$.

A variable is free in formula $F[x]$ if there is an occurrence of x that is not bound by any quantifier.

Examples: Identify constants, variables (free, bound), quantifiers, functions, predicates, atoms, terms, formulas from the bellow

1. $\underset{x}{\forall}\ x + 1 \geq x$
2. $\neg \left(\underset{x}{\exists}\ E[0, f[x]] \right)$
3. $\underset{x}{\forall} \underset{y}{\exists} \left(E[y, f[x]] \ \wedge\ \underset{z}{\forall} \left(E[z, f[x]] \Rightarrow E[y, z] \right) \right)$

Syntax (cont'd)

Formulas are defined as follows:

1. An atom is a formula.
2. If F and G are formulas then $\neg F$, $F \vee G$, $F \wedge G$, $F \implies G$, and $F \iff G$ are formulas.
3. If F is a formula and x is a free variable, then $\underset{x}{\forall} F[x]$ and $\underset{x}{\exists} F[x]$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

A variable is bound in formula $F[x]$ if there is an occurrence of x in the scope of a binding quantifier $\underset{x}{\forall}$ or $\underset{x}{\exists}$.

A variable is free in formula $F[x]$ if there is an occurrence of x that is not bound by any quantifier.

Examples: Identify constants, variables (free, bound), quantifiers, functions, predicates, atoms, terms, formulas from the bellow

1. $\underset{x}{\forall}\ x + 1 \geq x$
2. $\neg \left(\underset{x}{\exists}\ E[0, f[x]] \right)$
3. $\underset{x}{\forall} \underset{y}{\exists} \left(E[y, f[x]] \ \wedge \ \underset{z}{\forall} (E[z, f[x]] \Rightarrow E[y, z]) \right)$

Syntax (cont'd)

Formulas are defined as follows:

1. An atom is a formula.
2. If F and G are formulas then $\neg F$, $F \vee G$, $F \wedge G$, $F \Longrightarrow G$, and $F \iff G$ are formulas.
3. If F is a formula and x is a free variable, then $\underset{x}{\forall} F[x]$ and $\underset{x}{\exists} F[x]$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

A variable is bound in formula $F[x]$ if there is an occurrence of x in the scope of a binding quantifier $\underset{x}{\forall}$ or $\underset{x}{\exists}$.

A variable is free in formula $F[x]$ if there is an occurrence of x that is not bound by any quantifier.

Examples: Identify constants, variables (free, bound), quantifiers, functions, predicates, atoms, terms, formulas from the bellow

1. $\underset{x}{\forall}\ x + 1 \geq x$
2. $\neg \left(\underset{x}{\exists}\ E[0, f[x]] \right)$
3. $\underset{x}{\forall} \underset{y}{\exists} \left(E[y, f[x]] \ \wedge\ \underset{z}{\forall} (E[z, f[x]] \Rightarrow E[y, z]) \right)$

Syntax (cont'd)

Formulas are defined as follows:

1. An atom is a formula.
2. If F and G are formulas then $\neg F$, $F \vee G$, $F \wedge G$, $F \implies G$, and $F \iff G$ are formulas.
3. If F is a formula and x is a free variable, then $\underset{x}{\forall} F[x]$ and $\underset{x}{\exists} F[x]$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

A variable is bound in formula $F[x]$ if there is an occurrence of x in the scope of a binding quantifier $\underset{x}{\forall}$ or $\underset{x}{\exists}$.

A variable is free in formula $F[x]$ if there is an occurrence of x that is not bound by any quantifier.

Examples: Identify constants, variables (free, bound), quantifiers, functions, predicates, atoms, terms, formulas from the bellow

1. $\underset{x}{\forall}\, x + 1 \geq x$
2. $\neg \left(\underset{x}{\exists}\, E[0, f[x]] \right)$
3. $\underset{x}{\forall} \underset{y}{\exists} \left(E[y, f[x]] \wedge \underset{z}{\forall} \left(E[z, f[x]] \Rightarrow E[y, z] \right) \right)$

Syntax (cont'd)

Formulas are defined as follows:

1. An atom is a formula.
2. If F and G are formulas then $\neg F$, $F \vee G$, $F \wedge G$, $F \implies G$, and $F \iff G$ are formulas.
3. If F is a formula and x is a free variable, then $\underset{x}{\forall} F[x]$ and $\underset{x}{\exists} F[x]$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

A variable is bound in formula $F[x]$ if there is an occurrence of x in the scope of a binding quantifier $\underset{x}{\forall}$ or $\underset{x}{\exists}$.

A variable is free in formula $F[x]$ if there is an occurrence of x that is not bound by any quantifier.

Examples: Identify constants, variables (free, bound), quantifiers, functions, predicates, atoms, terms, formulas from the bellow

1. $\underset{x}{\forall}\ x + 1 \geq x$
2. $\neg \left(\underset{x}{\exists}\ E[0, f[x]] \right)$
3. $\underset{x}{\forall} \underset{y}{\exists} \left(E[y, f[x]] \ \wedge \ \underset{z}{\forall} (E[z, f[x]] \Rightarrow E[y, z]) \right)$

Syntax (cont'd)

Formulas are defined as follows:

1. An atom is a formula.
2. If F and G are formulas then $\neg F$, $F \vee G$, $F \wedge G$, $F \implies G$, and $F \iff G$ are formulas.
3. If F is a formula and x is a free variable, then $\underset{x}{\forall} F[x]$ and $\underset{x}{\exists} F[x]$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

A variable is bound in formula $F[x]$ if there is an occurrence of x in the scope of a binding quantifier $\underset{x}{\forall}$ or $\underset{x}{\exists}$.

A variable is free in formula $F[x]$ if there is an occurrence of x that is not bound by any quantifier.

Examples: Identify constants, variables (free, bound), quantifiers, functions, predicates, atoms, terms, formulas from the bellow

1. $\underset{x}{\forall}\, x + 1 \geq x$
2. $\neg \left(\underset{x}{\exists}\, E[0, f[x]] \right)$
3. $\underset{x}{\forall} \underset{y}{\exists} \left(E[y, f[x]] \;\wedge\; \underset{z}{\forall} \left(E[z, f[x]] \Rightarrow E[y, z] \right) \right)$

Syntax (cont'd)

Formulas are defined as follows:

1. An atom is a formula.
2. If F and G are formulas then $\neg F$, $F \vee G$, $F \wedge G$, $F \implies G$, and $F \iff G$ are formulas.
3. If F is a formula and x is a free variable, then $\underset{x}{\forall} F[x]$ and $\underset{x}{\exists} F[x]$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

A variable is bound in formula $F[x]$ if there is an occurrence of x in the scope of a binding quantifier $\underset{x}{\forall}$ or $\underset{x}{\exists}$.

A variable is free in formula $F[x]$ if there is an occurrence of x that is not bound by any quantifier.

Examples: Identify constants, variables (free, bound), quantifiers, functions, predicates, atoms, terms, formulas from the bellow

1. $\underset{x}{\forall}\, x + 1 \geq x$
2. $\neg \left(\underset{x}{\exists}\, E[0, f[x]] \right)$
3. $\underset{x}{\forall}\, \underset{y}{\exists} \Big(E[y, f[x]] \,\wedge\, \underset{z}{\forall} (E[z, f[x]] \Rightarrow E[y, z]) \Big)$

Syntax (cont'd)

Formulas are defined as follows:

1. An atom is a formula.
2. If F and G are formulas then $\neg F$, $F \vee G$, $F \wedge G$, $F \implies G$, and $F \iff G$ are formulas.
3. If F is a formula and x is a free variable, then $\underset{x}{\forall} F[x]$ and $\underset{x}{\exists} F[x]$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

A variable is bound in formula $F[x]$ if there is an occurrence of x in the scope of a binding quantifier $\underset{x}{\forall}$ or $\underset{x}{\exists}$.

A variable is free in formula $F[x]$ if there is an occurrence of x that is not bound by any quantifier.

Examples: Identify constants, variables (free, bound), quantifiers, functions, predicates, atoms, terms, formulas from the bellow

1. $\underset{x}{\forall}\, x + 1 \geq x$
2. $\neg \left(\underset{x}{\exists}\, E[0, f[x]] \right)$
3. $\underset{x}{\forall}\, \underset{y}{\exists} \left(E[y, f[x]] \wedge \underset{z}{\forall} (E[z, f[x]] \Rightarrow E[y, z]) \right)$

Syntax (cont'd)

Formulas are defined as follows:

1. An atom is a formula.
2. If F and G are formulas then $\neg F$, $F \vee G$, $F \wedge G$, $F \implies G$, and $F \iff G$ are formulas.
3. If F is a formula and x is a free variable, then $\underset{x}{\forall} F[x]$ and $\underset{x}{\exists} F[x]$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

A variable is bound in formula $F[x]$ if there is an occurrence of x in the scope of a binding quantifier $\underset{x}{\forall}$ or $\underset{x}{\exists}$.

A variable is free in formula $F[x]$ if there is an occurrence of x that is not bound by any quantifier.

Examples: Identify constants, variables (free, bound), quantifiers, functions, predicates, atoms, terms, formulas from the bellow

1. $\underset{x}{\forall}\ x + 1 \geq x$
2. $\neg \left(\underset{x}{\exists}\ E[0, f[x]] \right)$
3. $\underset{x}{\forall}\ \underset{y}{\exists} \left(E[y, f[x]]\ \wedge\ \underset{z}{\forall} (E[z, f[x]] \Rightarrow E[y, z]) \right)$

Syntax (cont'd)

Formulas are defined as follows:

1. An atom is a formula.
2. If F and G are formulas then $\neg F$, $F \vee G$, $F \wedge G$, $F \Longrightarrow G$, and $F \iff G$ are formulas.
3. If F is a formula and x is a free variable, then $\underset{x}{\forall} F[x]$ and $\underset{x}{\exists} F[x]$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

A variable is bound in formula $F[x]$ if there is an occurrence of x in the scope of a binding quantifier $\underset{x}{\forall}$ or $\underset{x}{\exists}$.

A variable is free in formula $F[x]$ if there is an occurrence of x that is not bound by any quantifier.

Examples: Identify constants, variables (free, bound), quantifiers, functions, predicates, atoms, terms, formulas from the bellow

1. $\underset{x}{\forall}\ x + 1 \geq x$
2. $\neg \left(\underset{x}{\exists}\ E[0, f[x]] \right)$
3. $\underset{x}{\forall}\ \underset{y}{\exists} \Big(E[y, f[x]]\ \wedge\ \underset{z}{\forall} (E[z, f[x]] \Rightarrow E[y, z]) \Big)$

Outline

Syntax

Semantics

Equivalences of Formulas

Normal Forms

(Un)Satisfiability & (In)Validity

Semantics

An interpretation I of a formula F in FOL consists of a nonempty domain D and an assignment of values to each constant, function, symbol and predicate symbol occurring in F as follows:

- to each constant we assign an element in D
- to each function symbol we assign a mapping from D^n to D
- to each predicate symbol we assign a mapping from D^n to $\{\mathbb{T}, \mathbb{F}\}$.

Then the semantics of the formula F is a function $f : \mathcal{I} \to \{\mathbb{T}, \mathbb{F}\}$, where $I \in \mathcal{I}$ and $\mathcal{I}$ is the set of all interpretations of the formula F.

Semantics

An interpretation I of a formula F in FOL consists of a nonempty domain D and an assignment of values to each constant, function, symbol and predicate symbol occurring in F as follows:

- to each constant we assign an element in D
- to each function symbol we assign a mapping from D^n to D
- to each predicate symbol we assign a mapping from D^n to $\{\mathbb{T}, \mathbb{F}\}$.

Then the semantics of the formula F is a function $f : \mathcal{I} \to \{\mathbb{T}, \mathbb{F}\}$, where $I \in \mathcal{I}$ and $\mathcal{I}$ is the set of all interpretations of the formula F.

Semantics

An interpretation I of a formula F in FOL consists of a nonempty domain D and an assignment of values to each constant, function, symbol and predicate symbol occurring in F as follows:

- to each constant we assign an element in D
- to each function symbol we assign a mapping from D^n to D
- to each predicate symbol we assign a mapping from D^n to $\{\mathbb{T}, \mathbb{F}\}$.

Then the semantics of the formula F is a function $f : \mathcal{I} \rightarrow \{\mathbb{T}, \mathbb{F}\}$, where $I \in \mathcal{I}$ and $\mathcal{I}$ is the set of all interpretations of the formula F.

Semantics

An interpretation I of a formula F in FOL consists of a nonempty domain D and an assignment of values to each constant, function, symbol and predicate symbol occurring in F as follows:

- to each constant we assign an element in D
- to each function symbol we assign a mapping from D^n to D
- to each predicate symbol we assign a mapping from D^n to $\{\mathbb{T}, \mathbb{F}\}$.

Then the semantics of the formula F is a function $f : \mathcal{I} \to \{\mathbb{T}, \mathbb{F}\}$, where $I \in \mathcal{I}$ and $\mathcal{I}$ is the set of all interpretations of the formula F.

Semantics

An interpretation I of a formula F in FOL consists of a nonempty domain D and an assignment of values to each constant, function, symbol and predicate symbol occurring in F as follows:

- to each constant we assign an element in D
- to each function symbol we assign a mapping from D^n to D
- to each predicate symbol we assign a mapping from D^n to $\{\mathbb{T}, \mathbb{F}\}$.

Then the semantics of the formula F is a function $f : \mathcal{I} \to \{\mathbb{T}, \mathbb{F}\}$, where $I \in \mathcal{I}$ and $\mathcal{I}$ is the set of all interpretations of the formula F.

Semantics (cont'd)

Example: Find the truth value of the formula $F :\Longleftrightarrow \underset{x}{\forall}\, \underset{y}{\exists}\, x + y > c$, where

$$I : \begin{cases} D = \{0, 1\} \\ c_I = 0 \\ +_I \to +_{\mathbb{Z}} \\ >_I \to >_{\mathbb{Z}} \end{cases}$$

Outline

Syntax

Semantics

Equivalences of Formulas

Normal Forms

(Un)Satisfiability & (In)Validity

Equivalences of Formulas

Two formulas F and G are equivalent iff the truth values of F and G are the same under any interpretation.

$$F \iff G \equiv (F \Rightarrow G) \land (G \Rightarrow F)$$
$$F \Rightarrow G \equiv \neg F \lor G$$
$$F \lor G \equiv G \lor F$$
$$(F \lor G) \lor H \equiv F \lor (G \lor H)$$
$$F \lor (G \land H) \equiv (F \lor G) \land (F \lor H)$$
$$F \lor \mathbb{T} \equiv \mathbb{T}$$
$$F \lor \mathbb{F} \equiv F$$
$$F \lor \neg F \equiv \mathbb{T}$$
$$\neg(\neg F) \equiv F$$
$$\neg(F \lor G) \equiv \neg F \land \neg G$$
$$(Qx)F[x] \lor G \equiv (Qx)(F[x] \lor G)$$
$$\neg \underset{x}{\forall} F[x] \equiv \underset{x}{\exists} \neg F[x]$$
$$\underset{x}{\forall} F[x] \lor \underset{x}{\forall} G[x] \not\equiv \underset{x}{\forall}(F[x] \lor G[x])$$
$$\underset{x}{\exists} F[x] \lor \underset{x}{\exists} G[x] \equiv \underset{x}{\exists}(F[x] \lor G[x])$$

$$F \land G \equiv G \land F$$
$$(F \land G) \land H \equiv F \land (G \land H)$$
$$F \land (G \lor H) \equiv (F \land G) \lor (F \land H)$$
$$F \land \mathbb{T} \equiv F$$
$$F \land \mathbb{F} \equiv \mathbb{F}$$
$$F \land \neg F \equiv \mathbb{F}$$
$$\neg(F \land G) \equiv \neg F \lor \neg G$$
$$(Qx)F[x] \land G \equiv (Qx)(F[x] \land G)$$
$$\neg(\underset{x}{\exists} x)F[x] \equiv \underset{x}{\forall} \neg F[x]$$
$$\underset{x}{\forall} F[x] \land \underset{x}{\forall} G[x] \equiv \underset{x}{\forall}(F[x] \land G[x])$$
$$\underset{x}{\exists} F[x] \land \underset{x}{\exists} G[x] \not\equiv \underset{x}{\exists}(F[x] \land G[x])$$

Note that

$$\underset{x}{\forall} F[x] \lor \underset{x}{\forall} G[x] \equiv \underset{x}{\forall} F[x] \lor \underset{y}{\forall} G[y] \equiv \underset{x,y}{\forall} F[x] \lor G[y]$$
$$\exists F[x] \land \exists G[x] \equiv \exists F[x] \land \exists G[y] \equiv \exists F[x] \land G[y]$$

Equivalences of Formulas

$$F \iff G \equiv (F \Rightarrow G) \land (G \Rightarrow F)$$

$$F \Rightarrow G \equiv \neg F \lor G$$

$$F \lor G \equiv G \lor F$$

$$(F \lor G) \lor H \equiv F \lor (G \lor H)$$

$$F \lor (G \land H) \equiv (F \lor G) \land (F \lor H)$$

$$F \lor \mathbb{T} \equiv \mathbb{T}$$

$$F \lor \mathbb{F} \equiv F$$

$$F \lor \neg F \equiv \mathbb{T}$$

$$\neg(\neg F) \equiv F$$

$$\neg(F \lor G) \equiv \neg F \land \neg G$$

$$(Qx) F[x] \lor G \equiv (Qx)(F[x] \lor G)$$

$$\neg \underset{x}{\forall} F[x] \equiv \underset{x}{\exists} \neg F[x]$$

$$\underset{x}{\forall} F[x] \lor \underset{x}{\forall} G[x] \not\equiv \underset{x}{\forall}(F[x] \lor G[x])$$

$$\underset{x}{\exists} F[x] \lor \underset{x}{\exists} G[x] \equiv \underset{x}{\exists}(F[x] \lor G[x])$$

$$F \land G \equiv G \land F$$

$$(F \land G) \land H \equiv F \land (G \land H)$$

$$F \land (G \lor H) \equiv (F \land G) \lor (F \land H)$$

$$F \land \mathbb{T} \equiv F$$

$$F \land \mathbb{F} \equiv \mathbb{F}$$

$$F \land \neg F \equiv \mathbb{F}$$

$$\neg(F \land G) \equiv \neg F \lor \neg G$$

$$(Qx) F[x] \land G \equiv (Qx)(F[x] \land G)$$

$$\neg(\underset{x}{\exists} x) F[x] \equiv \underset{x}{\forall} \neg F[x]$$

$$\underset{x}{\forall} F[x] \land \underset{x}{\forall} G[x] \equiv \underset{x}{\forall}(F[x] \land G[x])$$

$$\underset{x}{\exists} F[x] \land \underset{x}{\exists} G[x] \not\equiv \underset{x}{\exists}(F[x] \land G[x])$$

Note that

$$\underset{x}{\forall} F[x] \lor \underset{x}{\forall} G[x] \equiv \underset{x}{\forall} F[x] \lor \underset{y}{\forall} G[y] \equiv \underset{x,y}{\forall} F[x] \lor G[y]$$

$$\underset{x}{\exists} F[x] \land \underset{x}{\exists} G[x] \equiv \underset{x}{\exists} F[x] \land \underset{y}{\exists} G[y] \equiv \underset{x,y}{\exists} F[x] \land G[y]$$

Equivalences of Formulas

$$F \iff G \equiv (F \Rightarrow G) \wedge (G \Rightarrow F)$$
$$F \Rightarrow G \equiv \neg F \vee G$$

$F \vee G \equiv G \vee F$	$F \wedge G \equiv G \wedge F$
$(F \vee G) \vee H \equiv F \vee (G \vee H)$	$(F \wedge G) \wedge H \equiv F \wedge (G \wedge H)$
$F \vee (G \wedge H) \equiv (F \vee G) \wedge (F \vee H)$	$F \wedge (G \vee H) \equiv (F \wedge G) \vee (F \wedge H)$
$F \vee \mathbb{T} \equiv \mathbb{T}$	$F \wedge \mathbb{T} \equiv F$
$F \vee \mathbb{F} \equiv F$	$F \wedge \mathbb{F} \equiv \mathbb{F}$
$F \vee \neg F \equiv \mathbb{T}$	$F \wedge \neg F \equiv \mathbb{F}$
$\neg(\neg F) \equiv F$	
$\neg(F \vee G) \equiv \neg F \wedge \neg G$	$\neg(F \wedge G) \equiv \neg F \vee \neg G$
$(Qx)F[x] \vee G \equiv (Qx)(F[x] \vee G)$	$(Qx)F[x] \wedge G \equiv (Qx)(F[x] \wedge G)$
$\neg \forall_x F[x] \equiv \exists_x \neg F[x]$	$\neg(\exists_x x)F[x] \equiv \forall_x \neg F[x]$
$\forall_x F[x] \vee \forall_x G[x] \not\equiv \forall_x (F[x] \vee G[x])$	$\forall_x F[x] \wedge \forall_x G[x] \equiv \forall_x (F[x] \wedge G[x])$
$\exists_x F[x] \vee \exists_x G[x] \equiv \exists_x (F[x] \vee G[x])$	$\exists_x F[x] \wedge \exists_x G[x] \not\equiv \exists_x (F[x] \wedge G[x])$

Note that

$$\forall_x F[x] \vee \forall_x G[x] \equiv \forall_x F[x] \vee \forall_y G[y] \equiv \forall_{x,y} F[x] \vee G[y]$$
$$\exists_x F[x] \wedge \exists_x G[x] \equiv \exists_x F[x] \wedge \exists_y G[y] \equiv \exists_{x,y} F[x] \wedge G[y]$$

Outline

Syntax

Semantics

Equivalences of Formulas

Normal Forms

(Un)Satisfiability & (In)Validity

Normal Forms

Normal forms:

1. CNF
2. DNF
3. negation normal form (NNF)
4. prenex normal form (PNF)
5. Skolem standard form

Negation normal form (NNF) requires that $\neg$, $\wedge$, and $\vee$ to be the only logical connectives and that negations appear only in literals.

A formula F in FOL is said to be in prenex normal form (PNF) iff the formula is in the form $(Q_1 x_1) \ldots (Q_n x_n)\ M$, where $Q_i \in \{\forall, \exists\}$ and M is quantifier-free.

A FOL formula is in Skolem standard form if it is of the form $\underset{x_1,\ldots,x_n}{\forall} M$, where M is a quantifier-free formula in CNF.

Normal Forms

Normal forms:

1. CNF
2. DNF
3. negation normal form (NNF)
4. prenex normal form (PNF)
5. Skolem standard form

Negation normal form (NNF) requires that $\neg$, $\wedge$, and $\vee$ to be the only logical connectives and that negations appear only in literals.

A formula F in FOL is said to be in prenex normal form (PNF) iff the formula is in the form $(Q_1 x_1)...(Q_n x_n)\ M$, where $Q_i \in \{\forall, \exists\}$ and M is quantifier-free.

A FOL formula is in Skolem standard form if it is of the form $\underset{x_1,\ldots,x_n}{\forall} M$, where M is a quantifier-free formula in CNF.

Normal Forms

Normal forms:

1. CNF
2. DNF
3. negation normal form (NNF)
4. prenex normal form (PNF)
5. Skolem standard form

Negation normal form (NNF) requires that $\neg$, $\wedge$, and $\vee$ to be the only logical connectives and that negations appear only in literals.

A formula F in FOL is said to be in prenex normal form (PNF) iff the formula is in the form $(Q_1 x_1)...(Q_n x_n)\ M$, where $Q_i \in \{\forall, \exists\}$ and M is quantifier-free.

A FOL formula is in Skolem standard form if it is of the form $\underset{x_1,\ldots,x_n}{\forall} M$, where M is a quantifier-free formula in CNF.

Normal Forms

Normal forms:

1. CNF
2. DNF
3. negation normal form (NNF)
4. prenex normal form (PNF)
5. Skolem standard form

Negation normal form (NNF) requires that $\neg$, $\wedge$, and $\vee$ to be the only logical connectives and that negations appear only in literals.

A formula F in FOL is said to be in prenex normal form (PNF) iff the formula is in the form $(Q_1x_1)\ldots(Q_nx_n)\ M$, where $Q_i \in \{\forall, \exists\}$ and M is quantifier-free.

A FOL formula is in Skolem standard form if it is of the form $\underset{x_1,\ldots,x_n}{\forall} M$, where M is a quantifier-free formula in CNF.

Normal Forms

Normal forms:

1. CNF
2. DNF
3. negation normal form (NNF)
4. prenex normal form (PNF)
5. Skolem standard form

Negation normal form (NNF) requires that $\neg$, $\wedge$, and $\vee$ to be the only logical connectives and that negations appear only in literals.

A formula F in FOL is said to be in prenex normal form (PNF) iff the formula is in the form $(Q_1 x_1)...(Q_n x_n)\ M$, where $Q_i \in \{\forall, \exists\}$ and M is quantifier-free.

A FOL formula is in Skolem standard form if it is of the form $\underset{x_1,\ldots,x_n}{\forall} M$, where M is a quantifier-free formula in CNF.

Normal Forms

Normal forms:

1. CNF
2. DNF
3. negation normal form (NNF)
4. prenex normal form (PNF)
5. Skolem standard form

Negation normal form (NNF) requires that $\neg$, $\wedge$, and $\vee$ to be the only logical connectives and that negations appear only in literals.

A formula F in FOL is said to be in prenex normal form (PNF) iff the formula is in the form $(Q_1 x_1) \ldots (Q_n x_n)\ M$, where $Q_i \in \{\forall, \exists\}$ and M is quantifier-free.

A FOL formula is in Skolem standard form if it is of the form $\underset{x_1,\ldots,x_n}{\forall} M$, where M is a quantifier-free formula in CNF.

Normal Forms

Normal forms:

1. CNF
2. DNF
3. negation normal form (NNF)
4. prenex normal form (PNF)
5. Skolem standard form

Negation normal form (NNF) requires that $\neg$, $\wedge$, and $\vee$ to be the only logical connectives and that negations appear only in literals.

A formula F in FOL is said to be in prenex normal form (PNF) iff the formula is in the form $(Q_1 x_1) \ldots (Q_n x_n)\ M$, where $Q_i \in \{\forall, \exists\}$ and M is quantifier-free.

A FOL formula is in Skolem standard form if it is of the form $\underset{x_1,\ldots,x_n}{\forall} M$, where M is a quantifier-free formula in CNF.

Normal Forms

Normal forms:

1. CNF
2. DNF
3. negation normal form (NNF)
4. prenex normal form (PNF)
5. Skolem standard form

Negation normal form (NNF) requires that $\neg$, $\wedge$, and $\vee$ to be the only logical connectives and that negations appear only in literals.

A formula F in FOL is said to be in prenex normal form (PNF) iff the formula is in the form $(Q_1 x_1)...(Q_n x_n)\ M$, where $Q_i \in \{\forall, \exists\}$ and M is quantifier-free.

A FOL formula is in Skolem standard form if it is of the form $\underset{x_1,\ldots,x_n}{\forall} M$, where M is a quantifier-free formula in CNF.

Normal Forms

Normal forms:

1. CNF
2. DNF
3. negation normal form (NNF)
4. prenex normal form (PNF)
5. Skolem standard form

Negation normal form (NNF) requires that $\neg$, $\wedge$, and $\vee$ to be the only logical connectives and that negations appear only in literals.

A formula F in FOL is said to be in prenex normal form (PNF) iff the formula is in the form $(Q_1 x_1) \ldots (Q_n x_n)\ M$, where $Q_i \in \{\forall, \exists\}$ and M is quantifier-free.

A FOL formula is in Skolem standard form if it is of the form $\underset{x_1,\ldots,x_n}{\forall} M$, where M is a quantifier-free formula in CNF.

Normal Forms (cont'd)

Examples:

1. Prove the following by bringing the formulas into conjunctive normal form

$$\left(\underset{x}{\forall} P[x]\right) \Rightarrow Q \quad \equiv \quad \underset{x}{\exists} \left(P[x] \Rightarrow Q\right).$$

2. Bring the following formulas into Skolem standard form

   - $$\underset{x}{\forall}\ \underset{y,z}{\exists} \left((\neg P[x,y] \ \wedge \ Q[x,z]) \ \vee \ R[x,y,z]\right)$$

   - $$\underset{x,y}{\forall} \left(\underset{z}{\exists} P[x,z] \wedge P[y,z]\right) \ \Rightarrow \ \underset{u}{\exists} Q[x,y,u]$$

Normal Forms (cont'd)

Examples:

1. Prove the following by bringing the formulas into conjunctive normal form
$$\left(\underset{x}{\forall} P[x]\right) \Rightarrow Q \;\equiv\; \underset{x}{\exists}\,(P[x] \Rightarrow Q)\,.$$

2. Bring the following formulas into Skolem standard form

   - $$\underset{x}{\forall}\ \underset{y,z}{\exists}\,((\neg P[x,y] \;\wedge\; Q[x,z]) \;\vee\; R[x,y,z])$$

   - $$\underset{x,y}{\forall}\left(\underset{z}{\exists} P[x,z] \wedge P[y,z]\right) \;\Rightarrow\; \underset{u}{\exists} Q[x,y,u]$$

Normal Forms (cont'd)

Examples:

1. Prove the following by bringing the formulas into conjunctive normal form
$$\left(\underset{x}{\forall} P[x]\right) \Rightarrow Q \;\equiv\; \underset{x}{\exists}\,(P[x] \Rightarrow Q)\,.$$
2. Bring the following formulas into Skolem standard form
    - $$\underset{x}{\forall}\;\underset{y,z}{\exists}\,((\neg P[x,y] \;\wedge\; Q[x,z]) \;\vee\; R[x,y,z])$$
    - $$\underset{x,y}{\forall}\left(\underset{z}{\exists} P[x,z] \wedge P[y,z]\right) \;\Rightarrow\; \underset{u}{\exists} Q[x,y,u]$$

Normal Forms (cont'd)

Examples:

1. Prove the following by bringing the formulas into conjunctive normal form
$$\left(\underset{x}{\forall} P[x]\right) \Rightarrow Q \quad \equiv \quad \underset{x}{\exists}\,(P[x] \Rightarrow Q)\,.$$
2. Bring the following formulas into Skolem standard form
   - $$\underset{x}{\forall}\ \underset{y,z}{\exists}((\neg P[x,y] \ \wedge\ Q[x,z]) \ \vee\ R[x,y,z])$$
   - $$\underset{x,y}{\forall}\left(\underset{z}{\exists} P[x,z] \wedge P[y,z]\right) \ \Rightarrow\ \underset{u}{\exists} Q[x,y,u]$$

Normal Forms (cont'd)

Examples:

1. Prove the following by bringing the formulas into conjunctive normal form
$$\left(\underset{x}{\forall} P[x]\right) \Rightarrow Q \quad \equiv \quad \underset{x}{\exists} (P[x] \Rightarrow Q)\,.$$
2. Bring the following formulas into Skolem standard form
   - $$\underset{x}{\forall}\ \underset{y,z}{\exists} ((\neg P[x,y]\ \wedge\ Q[x,z])\ \vee\ R[x,y,z])$$
   - $$\underset{x,y}{\forall} \left(\underset{z}{\exists} P[x,z] \wedge P[y,z]\right) \ \Rightarrow\ \underset{u}{\exists} Q[x,y,u]$$

Outline

Syntax

Semantics

Equivalences of Formulas

Normal Forms

(Un)Satisfiability & (In)Validity

(Un)Satisfiability & (In)Validity

A formula F is satisfiable iff there exists an interpretation I such that $I \models F$.

A formula F is valid iff for all interpretations I, $I \models F$.

Note that validity and satisfiability applies to closed formulas.

Examples: Prove that

- $\underset{x}{\forall} P[x] \;\wedge\; \underset{y}{\exists} \neg P[y]$ is inconsistent.
- $\underset{x}{\forall} P[x] \;\Rightarrow\; \underset{y}{\exists} P[y]$ is valid.

(Un)Satisfiability & (In)Validity

A formula F is satisfiable iff there exists an interpretation I such that $I \models F$.

A formula F is valid iff for all interpretations I, $I \models F$.

Note that validity and satisfiability applies to closed formulas.

Examples: Prove that

- $\underset{x}{\forall} P[x] \;\wedge\; \underset{y}{\exists} \neg P[y]$ is inconsistent.
- $\underset{x}{\forall} P[x] \;\Rightarrow\; \underset{y}{\exists} P[y]$ is valid.

(Un)Satisfiability & (In)Validity

A formula F is satisfiable iff there exists an interpretation I such that $I \models F$.

A formula F is valid iff for all interpretations I, $I \models F$.

Note that validity and satisfiability applies to closed formulas.

Examples: Prove that

- $\underset{x}{\forall} P[x] \ \wedge \ \underset{y}{\exists} \neg P[y]$ is inconsistent.
- $\underset{x}{\forall} P[x] \ \Rightarrow \ \underset{y}{\exists} P[y]$ is valid.

(Un)Satisfiability & (In)Validity

A formula F is satisfiable iff there exists an interpretation I such that $I \models F$.

A formula F is valid iff for all interpretations I, $I \models F$.

Note that validity and satisfiability applies to closed formulas.

Examples: Prove that

- $\underset{x}{\forall} P[x] \ \wedge \ \underset{y}{\exists} \neg P[y]$ is inconsistent.
- $\underset{x}{\forall} P[x] \ \Rightarrow \ \underset{y}{\exists} P[y]$ is valid.

(Un)Satisfiability & (In)Validity

A formula F is satisfiable iff there exists an interpretation I such that $I \models F$.

A formula F is valid iff for all interpretations I, $I \models F$.

Note that validity and satisfiability applies to closed formulas.

Examples: Prove that

- $\underset{x}{\forall} P[x] \;\wedge\; \underset{y}{\exists} \neg P[y]$ is inconsistent.
- $\underset{x}{\forall} P[x] \;\Rightarrow\; \underset{y}{\exists} P[y]$ is valid.

(Un)Satisfiability & (In)Validity

A formula F is satisfiable iff there exists an interpretation I such that $I \models F$.

A formula F is valid iff for all interpretations I, $I \models F$.

Note that validity and satisfiability applies to closed formulas.

Examples: Prove that

- $\underset{x}{\forall} P[x] \;\wedge\; \underset{y}{\exists} \neg P[y]$ is inconsistent.
- $\underset{x}{\forall} P[x] \;\Rightarrow\; \underset{y}{\exists} P[y]$ is valid.