First-order Logic

Mădălina Erașcu

West University of Timișoara and Institute e-Austria Timișoara
bvd. V. Parvan 4, Timișsoara, Romania

madalina.erascu@e-uvt.ro

Based on the book: Chin-Liang Chang and Richard Char-Tung Lee. Symbolic Logic and Mechanical Theorem Proving, Chapter 3

Outline

Outline

Syntax

(Un)Satisfiability & (In)Validity

Equivalences of Formulas

Normal Forms

Formula Clausification

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 $f[t_1, \ldots, t_n]$ 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 $P[t_1, \ldots, t_n]$ 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 $\neg F$, $F \vee G$, $F \wedge G$, $F \Longrightarrow G$, and $F \iff G$ are formulas.
3. If F is a formula and x is a variable, then $\underset{x}{\forall} F$ and $\underset{x}{\exists} F$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

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

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

A closed formula is a formula which has no free occurrences of variables; or equivalently, in which all occurrences of variables are bound.

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 variable, then $\underset{x}{\forall} F$ and $\underset{x}{\exists} F$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

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

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

A closed formula is a formula which has no free occurrences of variables; or equivalently, in which all occurrences of variables are bound.

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 variable, then $\underset{x}{\forall} F$ and $\underset{x}{\exists} F$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

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

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

A closed formula is a formula which has no free occurrences of variables; or equivalently, in which all occurrences of variables are bound.

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 variable, then $\underset{x}{\forall} F$ and $\underset{x}{\exists} F$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

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

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

A closed formula is a formula which has no free occurrences of variables; or equivalently, in which all occurrences of variables are bound.

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 variable, then $\underset{x}{\forall} F$ and $\underset{x}{\exists} F$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

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

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

A closed formula is a formula which has no free occurrences of variables; or equivalently, in which all occurrences of variables are bound.

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 variable, then $\underset{x}{\forall} F$ and $\underset{x}{\exists} F$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

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

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

A closed formula is a formula which has no free occurrences of variables; or equivalently, in which all occurrences of variables are bound.

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 variable, then $\underset{x}{\forall} F$ and $\underset{x}{\exists} F$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

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

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

A closed formula is a formula which has no free occurrences of variables; or equivalently, in which all occurrences of variables are bound.

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 variable, then $\underset{x}{\forall} F$ and $\underset{x}{\exists} F$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

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

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

A closed formula is a formula which has no free occurrences of variables; or equivalently, in which all occurrences of variables are bound.

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 variable, then $\underset{x}{\forall} F$ and $\underset{x}{\exists} F$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

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

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

A closed formula is a formula which has no free occurrences of variables; or equivalently, in which all occurrences of variables are bound.

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 variable, then $\underset{x}{\forall} F$ and $\underset{x}{\exists} F$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

A variable x is bound in the formula F if there is an occurrence of x in the scope of a binding quantifier $\underset{x}{\forall}$ or $\underset{x}{\exists}$.
A variable x is free in the formula F if there is an occurrence of x that is not bound by any quantifier.
A closed formula is a formula which has no free occurrences of variables; or equivalently, in which all occurrences of variables are bound.

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 variable, then $\underset{x}{\forall} F$ and $\underset{x}{\exists} F$ are formulas.
4. Formulas are generated only by a finite number of applications of the above rules.

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

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

A closed formula is a formula which has no free occurrences of variables; or equivalently, in which all occurrences of variables are bound.

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. $\underset{x}{\forall}\, \underset{y}{\exists} \left(E[y, f[x]] \;\wedge\; \underset{z}{\forall} (E[z, f[x]] \Rightarrow E[y, z]) \right)$

Semantics

Excursion: Semantics of propositional logic.

The semantics of a formula F is a function f_F, $f_F : \mathcal{I} \to \{\mathbb{T}, \mathbb{F}\}$, where $\mathcal{I}$ is the set of all interpretations I defined as bellow.

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 $\{\mathbb{T}, \mathbb{F}\}$.

Semantics

Excursion: Semantics of propositional logic.

The semantics of a formula F is a function f_F, $f_F : \mathcal{I} \rightarrow \{\mathbb{T}, \mathbb{F}\}$, where $\mathcal{I}$ is the set of all interpretations I defined as bellow.

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 $\{\mathbb{T}, \mathbb{F}\}$.

Semantics

Excursion: Semantics of propositional logic.

The semantics of a formula F is a function f_F, $f_F : \mathcal{I} \to \{\mathbb{T}, \mathbb{F}\}$, where $\mathcal{I}$ is the set of all interpretations I defined as bellow.

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 $\{\mathbb{T}, \mathbb{F}\}$.

Semantics

Excursion: Semantics of propositional logic.

The semantics of a formula F is a function f_F, $f_F : \mathcal{I} \rightarrow \{\mathbb{T}, \mathbb{F}\}$, where $\mathcal{I}$ is the set of all interpretations I defined as bellow.

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 $\{\mathbb{T}, \mathbb{F}\}$.

Semantics

Excursion: Semantics of propositional logic.

The semantics of a formula F is a function f_F, $f_F : \mathcal{I} \rightarrow \{\mathbb{T}, \mathbb{F}\}$, where $\mathcal{I}$ is the set of all interpretations I defined as bellow.

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 $\{\mathbb{T}, \mathbb{F}\}$.

Semantics

Excursion: Semantics of propositional logic.

The semantics of a formula F is a function f_F, $f_F : \mathcal{I} \rightarrow \{\mathbb{T}, \mathbb{F}\}$, where $\mathcal{I}$ is the set of all interpretations I defined as bellow.

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 $\{\mathbb{T}, \mathbb{F}\}$.

Semantics (cont'd)

Truth evaluation of a formula

For every interpretation of a formula over a domain D, the formula can be evaluated to $\mathbb{T}$ or $\mathbb{F}$ according to the following rules:

- If the truth values of formulas G and H are evaluated, then the truth values of the formulas $\neg G$, $G \wedge H$, $G \vee H$, $G \Rightarrow H$, $G \leftrightarrow H$ are evaluated as for propositional logic case.
- $\underset{x}{\forall} G$ is evaluated to $\mathbb{T}$ if the truth value of G is evaluated to $\mathbb{T}$ for every $x \in D$; otherwise is evaluated to $\mathbb{F}$
- $\underset{x}{\exists} G$ is evaluated to $\mathbb{T}$ if the truth value of G is evaluated to $\mathbb{T}$ for at least one $x \in D$; otherwise is evaluated to $\mathbb{F}$

Observation: Note that a formula containing free variables can not be evaluated.

Semantics (cont'd)

Truth evaluation of a formula

For every interpretation of a formula over a domain D, the formula can be evaluated to $\mathbb{T}$ or $\mathbb{F}$ according to the following rules:

- If the truth values of formulas G and H are evaluated, then the truth values of the formulas $\neg G$, $G \wedge H$, $G \vee H$, $G \Rightarrow H$, $G \leftrightarrow H$ are evaluated as for propositional logic case.
- $\underset{x}{\forall} G$ is evaluated to $\mathbb{T}$ if the truth value of G is evaluated to $\mathbb{T}$ for every $x \in D$; otherwise is evaluated to $\mathbb{F}$
- $\underset{x}{\exists} G$ is evaluated to $\mathbb{T}$ if the truth value of G is evaluated to $\mathbb{T}$ for at least one $x \in D$; otherwise is evaluated to $\mathbb{F}$

Observation: Note that a formula containing free variables can not be evaluated.

Semantics (cont'd)

Truth evaluation of a formula

For every interpretation of a formula over a domain D, the formula can be evaluated to $\mathbb{T}$ or $\mathbb{F}$ according to the following rules:

- If the truth values of formulas G and H are evaluated, then the truth values of the formulas $\neg G$, $G \wedge H$, $G \vee H$, $G \Rightarrow H$, $G \leftrightarrow H$ are evaluated as for propositional logic case.
- $\underset{x}{\forall} G$ is evaluated to $\mathbb{T}$ if the truth value of G is evaluated to $\mathbb{T}$ for every $x \in D$; otherwise is evaluated to $\mathbb{F}$
- $\underset{x}{\exists} G$ is evaluated to $\mathbb{T}$ if the truth value of G is evaluated to $\mathbb{T}$ for at least one $x \in D$; otherwise is evaluated to $\mathbb{F}$

Observation: Note that a formula containing free variables can not be evaluated.

Semantics (cont'd)

Truth evaluation of a formula

For every interpretation of a formula over a domain D, the formula can be evaluated to $\mathbb{T}$ or $\mathbb{F}$ according to the following rules:

- If the truth values of formulas G and H are evaluated, then the truth values of the formulas $\neg G$, $G \wedge H$, $G \vee H$, $G \Rightarrow H$, $G \leftrightarrow H$ are evaluated as for propositional logic case.
- $\underset{x}{\forall} G$ is evaluated to $\mathbb{T}$ if the truth value of G is evaluated to $\mathbb{T}$ for every $x \in D$; otherwise is evaluated to $\mathbb{F}$
- $\underset{x}{\exists} G$ is evaluated to $\mathbb{T}$ if the truth value of G is evaluated to $\mathbb{T}$ for at least one $x \in D$; otherwise is evaluated to $\mathbb{F}$

Observation: Note that a formula containing free variables can not be evaluated.

Semantics (cont'd)

Truth evaluation of a formula

For every interpretation of a formula over a domain D, the formula can be evaluated to $\mathbb{T}$ or $\mathbb{F}$ according to the following rules:

- If the truth values of formulas G and H are evaluated, then the truth values of the formulas $\neg G$, $G \wedge H$, $G \vee H$, $G \Rightarrow H$, $G \leftrightarrow H$ are evaluated as for propositional logic case.
- $\underset{x}{\forall} G$ is evaluated to $\mathbb{T}$ if the truth value of G is evaluated to $\mathbb{T}$ for every $x \in D$; otherwise is evaluated to $\mathbb{F}$
- $\underset{x}{\exists} G$ is evaluated to $\mathbb{T}$ if the truth value of G is evaluated to $\mathbb{T}$ for at least one $x \in D$; otherwise is evaluated to $\mathbb{F}$

Observation: Note that a formula containing free variables can not be evaluated.

Semantics (cont'd)

Truth evaluation of a formula

For every interpretation of a formula over a domain D, the formula can be evaluated to $\mathbb{T}$ or $\mathbb{F}$ according to the following rules:

- If the truth values of formulas G and H are evaluated, then the truth values of the formulas $\neg G$, $G \wedge H$, $G \vee H$, $G \Rightarrow H$, $G \leftrightarrow H$ are evaluated as for propositional logic case.
- $\underset{x}{\forall} G$ is evaluated to $\mathbb{T}$ if the truth value of G is evaluated to $\mathbb{T}$ for every $x \in D$; otherwise is evaluated to $\mathbb{F}$
- $\underset{x}{\exists} G$ is evaluated to $\mathbb{T}$ if the truth value of G is evaluated to $\mathbb{T}$ for at least one $x \in D$; otherwise is evaluated to $\mathbb{F}$

Observation: Note that a formula containing free variables can not be evaluated.

Outline

(Un)Satisfiability & (In)Validity

Once interpretations are defined, these notions are defined analogously to those in propositional logic, see below.

A formula F is satisfiable (consistent) iff there exists an interpretation I such that F is evaluated to $\mathbb{T}$ in I. If a formula F is $\mathbb{T}$ in an interpretation I, we say that I is a model of F and I satisfies G.

A formula F is unsatisfiable (inconsistent) iff for all interpretations I, F is evaluated to $\mathbb{F}$ in I.

A formula F is valid iff for all interpretations I, F is evaluated to $\mathbb{T}$ in I.

A formula F is invalid iff there exists an interpretation I, such that F is evaluated to $\mathbb{F}$ in I.

A formula G is a logical consequence of formulas F_1, F_2, ..., F_n iff for every interpretation I, if $F_1 \wedge F_2 \wedge \ldots \wedge F_n$ is true in I, G is also true in I.

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.

(Un)Satisfiability & (In)Validity

Once interpretations are defined, these notions are defined analogously to those in propositional logic, see below.

A formula F is satisfiable (consistent) iff there exists an interpretation I such that F is evaluated to $\mathbb{T}$ in I. If a formula F is $\mathbb{T}$ in an interpretation I, we say that I is a model of F and I satisfies G.

A formula F is unsatisfiable (inconsistent) iff for all interpretations I, F is evaluated to $\mathbb{F}$ in I.

A formula F is valid iff for all interpretations I, F is evaluated to $\mathbb{T}$ in I.

A formula F is invalid iff there exists an interpretation I, such that F is evaluated to $\mathbb{F}$ in I.

A formula G is a logical consequence of formulas F_1, F_2, ..., F_n iff for every interpretation I, if $F_1 \land F_2 \land \dots \land F_n$ is true in I, G is also true in I.

Note that validity and satisfiability applies to closed formulas.

Examples: Prove that

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

(Un)Satisfiability & (In)Validity

Once interpretations are defined, these notions are defined analogously to those in propositional logic, see below.

A formula F is satisfiable (consistent) iff there exists an interpretation I such that F is evaluated to $\mathbb{T}$ in I. If a formula F is $\mathbb{T}$ in an interpretation I, we say that I is a model of F and I satisfies G.

A formula F is unsatisfiable (inconsistent) iff for all interpretations I, F is evaluated to $\mathbb{F}$ in I.

A formula F is valid iff for all interpretations I, F is evaluated to $\mathbb{T}$ in I.

A formula F is invalid iff there exists an interpretation I, such that F is evaluated to $\mathbb{F}$ in I.

A formula G is a logical consequence of formulas F_1, F_2, ..., F_n iff for every interpretation I, if $F_1 \wedge F_2 \wedge \ldots \wedge F_n$ is true in I, G is also true in I.

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.

(Un)Satisfiability & (In)Validity

Once interpretations are defined, these notions are defined analogously to those in propositional logic, see below.

A formula F is satisfiable (consistent) iff there exists an interpretation I such that F is evaluated to $\mathbb{T}$ in I. If a formula F is $\mathbb{T}$ in an interpretation I, we say that I is a model of F and I satisfies G.

A formula F is unsatisfiable (inconsistent) iff for all interpretations I, F is evaluated to $\mathbb{F}$ in I.

A formula F is valid iff for all interpretations I, F is evaluated to $\mathbb{T}$ in I.

A formula F is invalid iff there exists an interpretation I, such that F is evaluated to $\mathbb{F}$ in I.

A formula G is a logical consequence of formulas F_1, F_2, ..., F_n iff for every interpretation I, if $F_1 \wedge F_2 \wedge \ldots \wedge F_n$ is true in I, G is also true in I.

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.

(Un)Satisfiability & (In)Validity

Once interpretations are defined, these notions are defined analogously to those in propositional logic, see below.

A formula F is satisfiable (consistent) iff there exists an interpretation I such that F is evaluated to $\mathbb{T}$ in I. If a formula F is $\mathbb{T}$ in an interpretation I, we say that I is a model of F and I satisfies G.

A formula F is unsatisfiable (inconsistent) iff for all interpretations I, F is evaluated to $\mathbb{F}$ in I.

A formula F is valid iff for all interpretations I, F is evaluated to $\mathbb{T}$ in I.

A formula F is invalid iff there exists an interpretation I, such that F is evaluated to $\mathbb{F}$ in I.

A formula G is a logical consequence of formulas F_1, F_2, ..., F_n iff for every interpretation I, if $F_1 \wedge F_2 \wedge \ldots \wedge F_n$ is true in I, G is also true in I.

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.

(Un)Satisfiability & (In)Validity

Once interpretations are defined, these notions are defined analogously to those in propositional logic, see below.

A formula F is satisfiable (consistent) iff there exists an interpretation I such that F is evaluated to $\mathbb{T}$ in I. If a formula F is $\mathbb{T}$ in an interpretation I, we say that I is a model of F and I satisfies G.

A formula F is unsatisfiable (inconsistent) iff for all interpretations I, F is evaluated to $\mathbb{F}$ in I.

A formula F is valid iff for all interpretations I, F is evaluated to $\mathbb{T}$ in I.

A formula F is invalid iff there exists an interpretation I, such that F is evaluated to $\mathbb{F}$ in I.

A formula G is a logical consequence of formulas F_1, F_2, ..., F_n iff for every interpretation I, if $F_1 \wedge F_2 \wedge \ldots \wedge F_n$ is true in I, G is also true in I.

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.

(Un)Satisfiability & (In)Validity

Once interpretations are defined, these notions are defined analogously to those in propositional logic, see below.

A formula F is satisfiable (consistent) iff there exists an interpretation I such that F is evaluated to $\mathbb{T}$ in I. If a formula F is $\mathbb{T}$ in an interpretation I, we say that I is a model of F and I satisfies G.

A formula F is unsatisfiable (inconsistent) iff for all interpretations I, F is evaluated to $\mathbb{F}$ in I.

A formula F is valid iff for all interpretations I, F is evaluated to $\mathbb{T}$ in I.

A formula F is invalid iff there exists an interpretation I, such that F is evaluated to $\mathbb{F}$ in I.

A formula G is a logical consequence of formulas F_1, F_2, ..., F_n iff for every interpretation I, if $F_1 \wedge F_2 \wedge \ldots \wedge F_n$ is true in I, G is also true in I.

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.

Outline

Equivalences of Formulas

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

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

Which implications do not hold in the $\not\equiv$ above?

$\Longrightarrow$	$\not\Longrightarrow$
$\not\Longleftarrow$	$\Longleftarrow$

Equivalences of Formulas

$$
\begin{aligned}
F \iff G &\equiv (F \Rightarrow G) \wedge (G \Rightarrow F) \\
F \Rightarrow G &\equiv \neg F \vee G \\
F \vee G &\equiv G \vee F \\
(F \vee G) \vee H &\equiv F \vee (G \vee H) \\
F \vee (G \wedge H) &\equiv (F \vee G) \wedge (F \vee H) \\
F \vee \mathbb{T} &\equiv \mathbb{T} \\
F \vee \mathbb{F} &\equiv F \\
F \vee \neg F &\equiv \mathbb{T} \\
\neg(\neg F) &\equiv F \\
\neg(F \vee G) &\equiv \neg F \wedge \neg G \\
(Qx)F[x] \vee G &\equiv (Qx)\,(F[x] \vee G) \\
\neg \underset{x}{\forall} F[x] &\equiv \underset{x}{\exists} \neg F[x] \\
\underset{x}{\forall} F[x] \vee \underset{x}{\forall} G[x] &\not\equiv \underset{x}{\forall}(F[x] \vee G[x]) \\
\underset{x}{\exists} F[x] \vee \underset{x}{\exists} G[x] &\equiv \underset{x}{\exists}(F[x] \vee G[x])
\end{aligned}
$$

$$
\begin{aligned}
F \wedge G &\equiv G \wedge F \\
(F \wedge G) \wedge H &\equiv F \wedge (G \wedge H) \\
F \wedge (G \vee H) &\equiv (F \wedge G) \vee (F \wedge H) \\
F \wedge \mathbb{T} &\equiv F \\
F \wedge \mathbb{F} &\equiv \mathbb{F} \\
F \wedge \neg F &\equiv \mathbb{F} \\
\neg(F \wedge G) &\equiv \neg F \vee \neg G \\
(Qx)F[x] \wedge G &\equiv (Qx)\,(F[x] \wedge G) \\
\neg(\underset{x}{\exists} x)F[x] &\equiv \underset{x}{\forall} \neg F[x] \\
\underset{x}{\forall} F[x] \wedge \underset{x}{\forall} G[x] &\equiv \underset{x}{\forall}(F[x] \wedge G[x]) \\
\underset{x}{\exists} F[x] \wedge \underset{x}{\exists} G[x] &\not\equiv \underset{x}{\exists}(F[x] \wedge G[x])
\end{aligned}
$$

Which implications do not hold in the $\not\equiv$ above?

$\Longrightarrow$	$\not\Longrightarrow$
$\not\Longleftarrow$	$\Longleftarrow$

Equivalences of Formulas

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

Which implications do not hold in the $\not\equiv$ above?

$\Longrightarrow$	$\not\Longrightarrow$
$\not\Longleftarrow$	$\Longleftarrow$

Equivalences of Formulas

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

Which implications do not hold in the $\not\equiv$ above?

$\Longrightarrow$	$\not\Longrightarrow$
$\not\Longleftarrow$	$\Longleftarrow$

Equivalences of Formulas (cont'd)

Note that

$$\forall_x F[x] \lor \forall_x G[x] \;\equiv\; \forall_x F[x] \lor \forall_y G[y] \;\equiv\; \forall_{x,y} F[x] \lor G[y]$$

$$\exists_x F[x] \;\land\; \exists_x G[x] \;\equiv\; \exists_x F[x] \land \exists_y G[y] \;\equiv\; \exists_{x,y} F[x] \land G[y]$$

Outline

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.

Observation: Transforming a formula into PNF is done by applying the transformations from the slide *Equivalences of formulae*.

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.

Examples: Obtain the standard form of the formulae

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

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.

Observation: Transforming a formula into PNF is done by applying the transformations from the slide *Equivalences of formulae*.

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.

Examples: Obtain the standard form of the formulae

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

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.

Observation: Transforming a formula into PNF is done by applying the transformations from the slide *Equivalences of formulae*.

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.

Examples: Obtain the standard form of the formulae

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

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 $(Q_1 x_1)\dots(Q_n x_n)\ M$, where $Q_i \in \{\forall, \exists\}$ and M is quantifier-free.

Observation: Transforming a formula into PNF is done by applying the transformations from the slide *Equivalences of formulae*.

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

Examples: Obtain the standard form of the formulae

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

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.

Observation: Transforming a formula into PNF is done by applying the transformations from the slide *Equivalences of formulae*.

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.

Examples: Obtain the standard form of the formulae

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

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.

Observation: Transforming a formula into PNF is done by applying the transformations from the slide *Equivalences of formulae*.

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.

Examples: Obtain the standard form of the formulae

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

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.

Observation: Transforming a formula into PNF is done by applying the transformations from the slide *Equivalences of formulae*.

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.

Examples: Obtain the standard form of the formulae

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

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.

Observation: Transforming a formula into PNF is done by applying the transformations from the slide *Equivalences of formulae*.

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.

Examples: Obtain the standard form of the formulae

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

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.

Observation: Transforming a formula into PNF is done by applying the transformations from the slide *Equivalences of formulae*.

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.

Examples: Obtain the standard form of the formulae

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

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.

Observation: Transforming a formula into PNF is done by applying the transformations from the slide *Equivalences of formulae*.

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.

Examples: Obtain the standard form of the formulae

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

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.

Observation: Transforming a formula into PNF is done by applying the transformations from the slide *Equivalences of formulae*.

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.

Examples: Obtain the standard form of the formulae

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

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 $(Q_1 x_1)\dots(Q_n x_n)\ M$, where $Q_i \in \{\forall, \exists\}$ and M is quantifier-free.

Observation: Transforming a formula into PNF is done by applying the transformations from the slide *Equivalences of formulae*.

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

Examples: Obtain the standard form of the formulae

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

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.

Observation: Transforming a formula into PNF is done by applying the transformations from the slide *Equivalences of formulae*.

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.

Examples: Obtain the standard form of the formulae

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

Normal Forms (cont'd)

Transforming Formulas into Prenex Normal Form

- Step 1. Use the laws
  - $F \iff G \;=\; (F \Rightarrow G) \wedge (G \Rightarrow F)$
  - $F \Rightarrow G \;=\; \neg F \vee G$

  to eliminate $\iff$ and $\Rightarrow$.
- Step 2. Repeatedly use the laws
  - $\neg(\neg F) \;=\; F$

  and de Morgan's laws
  - $\neg(F \vee G) \;=\; \neg F \wedge \neg G$
  - $\neg(F \wedge G) \;=\; \neg F \vee \neg G$

  to bring the negation signs immediately before atoms.
- Step 3. Rename bound variables if necessary.
- Step 4. Use the laws
  - $(Qx)F[x] \vee G \;=\; (Qx)(F[x] \vee G)$
  - $(Qx)F[x] \wedge G \;=\; (Qx)(F[x] \wedge G)$
  - $\underset{x}{\forall} F[x] \wedge \underset{x}{\forall} H[x] \;=\; \underset{x}{\forall}(F[x] \wedge H[x])$
  - $\underset{x}{\exists} F[x] \vee \underset{x}{\exists} H[x] \;=\; \underset{x}{\forall}(F[x] \vee H[x])$
  - $(Q_1x)F[x] \vee (Q_2x)H[x] \;=\; (Q_1x)(Q_2x)(F[x] \vee H[x])$
  - $(Q_1x)F[x] \wedge (Q_2x)H[x] \;=\; (Q_1x)(Q_2x)(F[x] \wedge H[x])$

  to move the quantifiers to the left of the entire formula to obtain a prenex normal form.

Normal Forms (cont'd)

Transforming Formulas into Prenex Normal Form

- Step 1. Use the laws
  - $F \iff G \quad = \quad (F \Rightarrow G) \wedge (G \Rightarrow F)$
  - $F \Rightarrow G \quad = \quad \neg F \vee G$

  to eliminate $\iff$ and $\Rightarrow$.
- Step 2. Repeatedly use the laws
  - $\neg(\neg F) \quad = \quad F$

  and de Morgan's laws
  - $\neg(F \vee G) \quad = \quad \neg F \wedge \neg G$
  - $\neg(F \wedge G) \quad = \quad \neg F \vee \neg G$

  to bring the negation signs immediately before atoms.
- Step 3. Rename bound variables if necessary.
- Step 4. Use the laws
  - $(Qx)F[x] \vee G \quad = \quad (Qx)(F[x] \vee G)$
  - $(Qx)F[x] \wedge G \quad = \quad (Qx)(F[x] \wedge G)$
  - $\underset{x}{\forall} F[x] \wedge \underset{x}{\forall} H[x] \quad = \quad \underset{x}{\forall}(F[x] \wedge H[x])$
  - $\underset{x}{\exists} F[x] \vee \underset{x}{\exists} H[x] \quad = \quad \underset{x}{\forall}(F[x] \vee H[x])$
  - $(Q_1x)F[x] \vee (Q_2x)H[x] \quad = \quad (Q_1x)(Q_2x)(F[x] \vee H[x])$
  - $(Q_1x)F[x] \wedge (Q_2x)H[x] \quad = \quad (Q_1x)(Q_2x)(F[x] \wedge H[x])$

  to move the quantifiers to the left of the entire formula to obtain a prenex normal form.

Normal Forms (cont'd)

Transforming Formulas into Prenex Normal Form

- Step 1. Use the laws
  - $F \iff G \;=\; (F \Rightarrow G) \wedge (G \Rightarrow F)$
  - $F \Rightarrow G \;=\; \neg F \vee G$

  to eliminate $\iff$ and $\Rightarrow$.
- Step 2. Repeatedly use the laws
  - $\neg(\neg F) \;=\; F$

  and de Morgan's laws
  - $\neg(F \vee G) \;=\; \neg F \wedge \neg G$
  - $\neg(F \wedge G) \;=\; \neg F \vee \neg G$

  to bring the negation signs immediately before atoms.
- Step 3. Rename bound variables if necessary.
- Step 4. Use the laws
  - $(Qx)F[x] \vee G \;=\; (Qx)(F[x] \vee G)$
  - $(Qx)F[x] \wedge G \;=\; (Qx)(F[x] \wedge G)$
  - $\forall_x F[x] \wedge \forall_x H[x] \;=\; \forall_x (F[x] \wedge H[x])$
  - $\exists_x F[x] \vee \exists_x H[x] \;=\; \forall_x (F[x] \vee H[x])$
  - $(Q_1 x)F[x] \vee (Q_2 x)H[x] \;=\; (Q_1 x)(Q_2 x)(F[x] \vee H[x])$
  - $(Q_1 x)F[x] \wedge (Q_2 x)H[x] \;=\; (Q_1 x)(Q_2 x)(F[x] \wedge H[x])$

  to move the quantifiers to the left of the entire formula to obtain a prenex normal form.

Normal Forms (cont'd)

Transforming Formulas into Prenex Normal Form

- Step 1. Use the laws
  - $F \iff G \quad = \quad (F \Rightarrow G) \land (G \Rightarrow F)$
  - $F \Rightarrow G \quad = \quad \neg F \lor G$

  to eliminate $\iff$ and $\Rightarrow$.
- Step 2. Repeatedly use the laws
  - $\neg(\neg F) \quad = \quad F$

  and de Morgan's laws
  - $\neg(F \lor G) \quad = \quad \neg F \land \neg G$
  - $\neg(F \land G) \quad = \quad \neg F \lor \neg G$

  to bring the negation signs immediately before atoms.
- Step 3. Rename bound variables if necessary.
- Step 4. Use the laws
  - $(Qx)F[x] \lor G \quad = \quad (Qx)(F[x] \lor G)$
  - $(Qx)F[x] \land G \quad = \quad (Qx)(F[x] \land G)$
  - $\forall_x F[x] \land \forall_x H[x] \quad = \quad \forall_x (F[x] \land H[x])$
  - $\exists_x F[x] \lor \exists_x H[x] \quad = \quad \forall_x (F[x] \lor H[x])$
  - $(Q_1 x)F[x] \lor (Q_2 x)H[x] \quad = \quad (Q_1 x)(Q_2 x)(F[x] \lor H[x])$
  - $(Q_1 x)F[x] \land (Q_2 x)H[x] \quad = \quad (Q_1 x)(Q_2 x)(F[x] \land H[x])$

  to move the quantifiers to the left of the entire formula to obtain a prenex normal form.

Normal Forms (cont'd)

Transforming Formulas into Prenex Normal Form

- Step 1. Use the laws
  - $F \iff G \;=\; (F \Rightarrow G) \wedge (G \Rightarrow F)$
  - $F \Rightarrow G \;=\; \neg F \vee G$

  to eliminate $\iff$ and $\Rightarrow$.
- Step 2. Repeatedly use the laws
  - $\neg(\neg F) \;=\; F$

  and de Morgan's laws
  - $\neg(F \vee G) \;=\; \neg F \wedge \neg G$
  - $\neg(F \wedge G) \;=\; \neg F \vee \neg G$

  to bring the negation signs immediately before atoms.
- Step 3. Rename bound variables if necessary.
- Step 4. Use the laws
  - $(Qx)F[x] \vee G \;=\; (Qx)(F[x] \vee G)$
  - $(Qx)F[x] \wedge G \;=\; (Qx)(F[x] \wedge G)$
  - $\underset{x}{\forall} F[x] \wedge \underset{x}{\forall} H[x] \;=\; \underset{x}{\forall}(F[x] \wedge H[x])$
  - $\underset{x}{\exists} F[x] \vee \underset{x}{\exists} H[x] \;=\; \underset{x}{\forall}(F[x] \vee H[x])$
  - $(Q_1 x)F[x] \vee (Q_2 x)H[x] \;=\; (Q_1 x)(Q_2 x)(F[x] \vee H[x])$
  - $(Q_1 x)F[x] \wedge (Q_2 x)H[x] \;=\; (Q_1 x)(Q_2 x)(F[x] \wedge H[x])$

  to move the quantifiers to the left of the entire formula to obtain a prenex normal form.

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

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

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

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

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

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

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

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

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

Outline

Syntax

(Un)Satisfiability & (In)Validity

Equivalences of Formulas

Normal Forms

Formula Clausification

Formula Clausification

A clause is a disjunction of literals.

Examples: $\neg P[x] \vee Q[y, f[x]]$, $P[x]$

A set of clauses S is regarded as a conjunction of all clauses in S, where every variable in S is considered governed by a universal quantifier.

Example: Let

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

The standard form of the formula above, that is

$$\underset{x}{\forall} ((\neg P[x, f[x]] \ \vee \ R[x, f[x], g[x]]) \ \wedge \ (Q(x, g[x]) \ \vee \ R[x, f[x], g[x]]))$$

can be represented by the following set of clauses

$$\{\neg P[x, f[x]] \ \vee \ R[x, f[x], g[x]], Q(x, g[x]) \ \vee \ R[x, f[x], g[x]]\}$$

Note that, if S is a set of clauses that represents a standard form of a formula F, then F is inconsistent iff S is inconsistent.

Formula Clausification

A clause is a disjunction of literals.

Examples: $\neg P[x] \vee Q[y, f[x]]$, $P[x]$

A set of clauses S is regarded as a conjunction of all clauses in S, where every variable in S is considered governed by a universal quantifier.

Example: Let

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

The standard form of the formula above, that is

$$\underset{x}{\forall} ((\neg P[x, f[x]] \ \vee \ R[x, f[x], g[x]]) \ \wedge \ (Q(x, g[x]) \ \vee \ R[x, f[x], g[x]]))$$

can be represented by the following set of clauses

$$\{\neg P[x, f[x]] \ \vee \ R[x, f[x], g[x]], Q(x, g[x]) \ \vee \ R[x, f[x], g[x]]\}$$

Note that, if S is a set of clauses that represents a standard form of a formula F, then F is inconsistent iff S is inconsistent.

Formula Clausification

A clause is a disjunction of literals.

Examples: $\neg P[x] \vee Q[y, f[x]]$, $P[x]$

A set of clauses S is regarded as a conjunction of all clauses in S, where every variable in S is considered governed by a universal quantifier.

Example: Let

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

The standard form of the formula above, that is

$$\underset{x}{\forall}((\neg P[x,f[x]]\ \vee\ R[x,f[x],g[x]])\ \wedge\ (Q(x,g[x])\ \vee\ R[x,f[x],g[x]]))$$

can be represented by the following set of clauses

$$\{\neg P[x,f[x]]\ \vee\ R[x,f[x],g[x]], Q(x,g[x])\ \vee\ R[x,f[x],g[x]]\}$$

Note that, if S is a set of clauses that represents a standard form of a formula F, then F is inconsistent iff S is inconsistent.

Formula Clausification

A clause is a disjunction of literals.

Examples: $\neg P[x] \vee Q[y, f[x]]$, $P[x]$

A set of clauses S is regarded as a conjunction of all clauses in S, where every variable in S is considered governed by a universal quantifier.

Example: Let

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

The standard form of the formula above, that is

$$\underset{x}{\forall}((\neg P[x, f[x]] \ \vee \ R[x, f[x], g[x]]) \ \wedge \ (Q(x, g[x]) \ \vee \ R[x, f[x], g[x]]))$$

can be represented by the following set of clauses

$$\{\neg P[x, f[x]] \ \vee \ R[x, f[x], g[x]], Q(x, g[x]) \ \vee \ R[x, f[x], g[x]]\}$$

Note that, if S is a set of clauses that represents a standard form of a formula F, then F is inconsistent iff S is inconsistent.

Formula Clausification

A clause is a disjunction of literals.

Examples: $\neg P[x] \vee Q[y, f[x]]$, $P[x]$

A set of clauses S is regarded as a conjunction of all clauses in S, where every variable in S is considered governed by a universal quantifier.

Example: Let

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

The standard form of the formula above, that is

$$\underset{x}{\forall}((\neg P[x, f[x]]\ \vee\ R[x, f[x], g[x]])\ \wedge\ (Q(x, g[x])\ \vee\ R[x, f[x], g[x]]))$$

can be represented by the following set of clauses

$$\{\neg P[x, f[x]]\ \vee\ R[x, f[x], g[x]], Q(x, g[x])\ \vee\ R[x, f[x], g[x]]\}$$

Note that, if S is a set of clauses that represents a standard form of a formula F, then F is inconsistent iff S is inconsistent.

Formulas Clausification (cont'd)

Example :

Transform the formulas F_1, F_2,F_3,F_4, and $\neg G$ into a set of clauses, where

$$F_1 : \quad \underset{x,y}{\forall}\ \underset{z}{\exists} P[x,y,z]$$

$$F_2 : \quad \begin{array}{l} \underset{x,y,z,u,v,w}{\forall} (P[x,y,u] \ \wedge\ P[y,z,v] \ \wedge\ P[u,z,w] \ \Rightarrow\ P[x,v,w]) \\ \wedge \\ \underset{x,y,z,u,v,w}{\forall} (P[x,y,u] \ \wedge\ (P[y,z,v] \ \wedge\ P[x,v,w]) \ \Rightarrow\ P[u,z,w]) \end{array}$$

$$F_3 : \quad \underset{x}{\forall} P[x,e,x] \wedge \underset{x}{\forall} P[e,x,x]$$

$$F_4 : \quad \underset{x}{\forall} P[x,i[x],e] \wedge \underset{x}{\forall} P[i[x],x,e]$$

$$G : \quad \left(\underset{x}{\forall} P[x,x,e]\right) \ \Rightarrow\ \underset{u,v,w}{\forall} (P[u,v,w] \ \Rightarrow\ P[v,u,w])$$