First-Order Logic

Yu-Fang Chen

Institute of Information Science
Academia Sinica

FLOLAC 2019

First-Order Logic

First-Order Logic (FOL)

- also called (first-order) predicate logic, predicate calculus, ...

First-Order Logic

First-Order Logic (FOL)

- also called (first-order) predicate logic, predicate calculus, ...
- can be seen as an extension of propositional logic.
- with additional concepts: quantifiers, functions, predicates.
- much more expressive than propositional logic!
- a well-known example from calculus
  - For all $\epsilon > 0$ there exists some n_0, such that for all $n \geq n_0$, $abs(f(n) - a) < \epsilon$.
- quantifiers: for all $\forall$ and exists $\exists$
- functions: abs, f, $-$
- predicates: $>$, $\geq$, $<$

First-Order Logic — Examples

What is expressible in FOL? (informal examples)

- "All men are mortal. Socrates is a man. Therefore Socrates is mortal."

$$\big((\forall x \,.\, man(x) \rightarrow mortal(x)) \wedge man(Socrates)\big) \rightarrow mortal(Socrates)$$

First-Order Logic — Examples

What is expressible in FOL? (informal examples)

- "All men are mortal. Socrates is a man. Therefore Socrates is mortal."

$$\big((\forall x\ .\ man(x) \to mortal(x)) \land man(Socrates)\big) \to mortal(Socrates)$$

- "All men are mortal. Elvis is immortal. Therefore Elvis is not a man."

$$\big((\forall x\ .\ man(x) \to mortal(x)) \land \neg mortal(Elvis)\big) \to \neg man(Elvis)$$

First-Order Logic — Examples

- “Luke is a Jedi.”:

$$isJedi(Luke)$$

First-Order Logic — Examples

- “Luke is a Jedi.”:

$$isJedi(Luke)$$

- “Anakin is the father of Luke.”:

$$isFatherOf(Anakin, Luke) \quad \text{or}$$
$$Anakin = fatherOf(Luke)$$

- also means “Luke is a son of Anakin.”

First-Order Logic — Examples

- "Luke is a Jedi.":

$$isJedi(Luke)$$

- "Anakin is the father of Luke.":

$$isFatherOf(Anakin, Luke) \quad \text{or}$$
$$Anakin = fatherOf(Luke)$$

- also means "Luke is a son of Anakin."

- "Gandalf is not the father of Luke.":

$$\neg isFatherOf(Gandalf, Luke) \quad \text{or}$$
$$\neg(Gandalf = fatherOf(Luke))$$
$$(\equiv\ \models Gandalf \neq fatherOf(Luke))$$

First-Order Logic — Examples

- "Anakin is the father of Luke and Leia.":

$$isFatherOf(Anakin, Luke) \wedge isFatherOf(Anakin, Leia)$$

First-Order Logic — Examples

- "Anakin is the father of Luke and Leia.":

$$isFatherOf(Anakin, Luke) \wedge isFatherOf(Anakin, Leia)$$

- "Luke has a father.":

$$\exists x \ . \ isFatherOf(x, Luke)$$

First-Order Logic — Examples

- “Anakin is the father of Luke and Leia.”:

$$isFatherOf(Anakin, Luke) \wedge isFatherOf(Anakin, Leia)$$

- “Luke has a father.”:

$$\exists x \,.\, isFatherOf(x, Luke)$$

- “Luke has a father and Leia also has a father.”:

$$(\exists x \,.\, isFatherOf(x, Luke)) \wedge (\exists y \,.\, isFatherOf(y, Leia))$$

First-Order Logic — Examples

- "Anakin is the father of Luke and Leia.":

$$isFatherOf(Anakin, Luke) \wedge isFatherOf(Anakin, Leia)$$

- "Luke has a father.":

$$\exists x \,.\, isFatherOf(x, Luke)$$

- "Luke has a father and Leia also has a father.":

$$(\exists x \,.\, isFatherOf(x, Luke)) \wedge (\exists y \,.\, isFatherOf(y, Leia))$$

- "Luke and Leia have the same father!":

$$\exists x \,.\, isFatherOf(x, Luke) \wedge isFatherOf(x, Leia)$$

First-Order Logic — Examples

- "There is a person who does not have a father.":

$$\exists x \, \neg \exists y \, . \, \mathit{isFatherOf}(y, x)$$
$$\big(\equiv \exists x \forall y \, . \, \neg \mathit{isFatherOf}(y, x) \big)$$

First-Order Logic — Examples

- "There is a person who does not have a father.":

$$\exists x\, \neg \exists y \,.\, \mathit{isFatherOf}(y, x)$$
$$\big(\equiv \exists x \forall y \,.\, \neg \mathit{isFatherOf}(y, x)\big)$$

- "All children of a Jedi are Jedis.":

$$\forall x, y \,.\, \big(\mathit{isJedi}(y) \wedge (\mathit{isFatherOf}(y, x) \vee \mathit{isMotherOf}(y, x))\big) \rightarrow \mathit{isJedi}(x)$$

First-Order Logic — Examples

- There are infinitely many primes [Euclid, c. 300 BC]

$$\forall x \exists y \,.\, y > x \wedge \big(\forall z \,.\, (1 < z \wedge z < y) \rightarrow y \,\mathsf{mod}\, z \neq 0\big)$$

First-Order Logic — Examples

- There are infinitely many primes [Euclid, c. 300 BC]

$$\forall x \exists y \,.\, y > x \wedge \big(\forall z \,.\, (1 < z \wedge z < y) \rightarrow y \,\mathsf{mod}\, z \neq 0\big)$$

- Last Fermat's Theorem [Fermat, 1637] (proven in [Wiles, 1994])

$$\forall n, x, y \in \mathbb{N} \,.\, n > 2 \quad \rightarrow \quad (\neg \exists z \in \mathbb{N} \,.\, x^n + y^n = z^n)$$

First-Order Logic — Examples

- There are infinitely many primes [Euclid, c. 300 BC]

$$\forall x \exists y \,.\, y > x \land \big(\forall z \,.\, (1 < z \land z < y) \rightarrow y \,\mathsf{mod}\, z \neq 0\big)$$

- Last Fermat's Theorem [Fermat, 1637] (proven in [Wiles, 1994])

$$\forall n, x, y \in \mathbb{N} \,.\, n > 2 \quad \rightarrow \quad (\neg \exists z \in \mathbb{N} \,.\, x^n + y^n = z^n)$$

- Goldbach Conjecture [Goldbach, 1742] (still open)

$$\forall x \,.\, (x > 2 \land even(x)) \rightarrow (\exists y, z \,.\, prime(y) \land prime(z) \land x = y + z)$$

First-Order Logic — Examples

- There are infinitely many primes [Euclid, c. 300 BC]

$$\forall x \exists y \,.\, y > x \wedge \big(\forall z \,.\, (1 < z \wedge z < y) \rightarrow y \,\mathsf{mod}\, z \neq 0\big)$$

- Last Fermat's Theorem [Fermat, 1637] (proven in [Wiles, 1994])

$$\forall n, x, y \in \mathbb{N} \,.\, n > 2 \quad \rightarrow \quad (\neg \exists z \in \mathbb{N} \,.\, x^n + y^n = z^n)$$

- Goldbach Conjecture [Goldbach, 1742] (still open)

$$\forall x \,.\, (x > 2 \wedge even(x)) \rightarrow (\exists y, z \,.\, prime(y) \wedge prime(z) \wedge x = y + z)$$

- Weak Goldbach Conjecture (proven in [Helfgott, 2013])

$$\forall x \,.\, (x > 5 \wedge odd(x)) \rightarrow$$
$$(\exists y, z, w \,.\, prime(y) \wedge prime(z) \wedge prime(w) \wedge x = y + z + w)$$

Syntax

Syntax:

- **Alphabet**:
  - variables: $x, y, \ldots, x_1, x_2, \ldots$ (hold elements of a universe)

Syntax

Syntax:

- **Alphabet**:
  - variables: $x, y, \ldots, x_1, x_2, \ldots$ (hold elements of a universe)
  - function symbols (with $/arity$): $f/2$, $(+)/2$, $\sin/1$, $\mathit{fatherOf}/1$, $\pi/0$, $42/0$, $(+1)/1$, ...
    - nullary functions (arity 0): constants
    - to be used as, e.g., $f(a,3)$, $+(40,2)$, $\sin(+1(x))$, $\mathit{fatherOf}(\mathit{Luke})$, $\pi()$
    - we often simplify the notation: $+(40,2) \mapsto 40+2$, $\pi() \mapsto \pi$, $+1(x) \mapsto x+1, \ldots$

Syntax

Syntax:

- **Alphabet**:
  - variables: $x, y, \ldots, x_1, x_2, \ldots$ (hold elements of a universe)
  - function symbols (with $/arity$): $f/2$, $(+)/2$, $\sin/1$, $\mathit{fatherOf}/1$, $\pi/0$, $42/0$, $(+1)/1$, ...
    - nullary functions (arity 0): constants
    - to be used as, e.g., $f(a, 3)$, $+(40, 2)$, $\sin(+1(x))$, $\mathit{fatherOf}(\mathit{Luke})$, $\pi()$
    - we often simplify the notation: $+(40, 2) \mapsto 40 + 2$, $\pi() \mapsto \pi$, $+1(x) \mapsto x + 1$, ...
  - predicate symbols (with $/arity$): $p/3$, $= /2$, $\mathit{isFatherOf}/2$, $(= 0)/1$, $\mathit{isJedi}/1$, $< /2$, ...
    - to be used as, e.g., $p(a, x, 9)$, $= (x, 42)$, $\mathit{isFatherOf}(\mathit{Anakin}, \mathit{Luke})$, $(= 0)(x)$, $\mathit{isJedi}(\mathit{Anakin})$, $< (x, \pi)$
    - we often simplify the notation: $= (x, 42) \mapsto x = 42$, $(= 0)(x) \mapsto x = 0$, $< (x, \pi) \mapsto x < \pi$, ...

Syntax

Syntax:

- **Alphabet**:
  - variables: $x, y, \ldots, x_1, x_2, \ldots$ (hold elements of a universe)
  - function symbols (with $/arity$): $f/2$, $(+)/2$, $\sin/1$, $\mathit{fatherOf}/1$, $\pi/0$, $42/0$, $(+1)/1$, $\ldots$
    - nullary functions (arity 0): constants
    - to be used as, e.g., $f(a,3)$, $+(40,2)$, $\sin(+1(x))$, $\mathit{fatherOf}(\mathit{Luke})$, $\pi()$
    - we often simplify the notation: $+(40,2) \mapsto 40+2$, $\pi() \mapsto \pi$, $+1(x) \mapsto x+1$, $\ldots$
  - predicate symbols (with $/arity$): $p/3$, $=/2$, $\mathit{isFatherOf}/2$, $(=0)/1$, $\mathit{isJedi}/1$, $</2$, $\ldots$
    - to be used as, e.g., $p(a,x,9)$, $=(x,42)$, $\mathit{isFatherOf}(\mathit{Anakin}, \mathit{Luke})$, $(=0)(x)$, $\mathit{isJedi}(\mathit{Anakin})$, $<(x,\pi)$
    - we often simplify the notation: $=(x,42) \mapsto x = 42$, $(=0)(x) \mapsto x = 0$, $<(x,\pi) \mapsto x < \pi$, $\ldots$
- Signature = function symbols + predicate symbols
  - can be seen as a parameter of an instance of FOL
  - sometimes called vocabulary or language of FOL

Syntax

Syntax:

- **Grammar**:
  - term: $t ::= x$ occurrence of a variable $x \in \mathbb{X}$
    $\mid f(t_1, \ldots, t_n)$ where f/n is a function symbol

Syntax

Syntax:

- **Grammar**:
  - term: $t ::= x$ occurrence of a variable $x \in \mathbb{X}$
    $\mid f(t_1, \ldots, t_n)$ where f/n is a function symbol
  - formula:

$F ::= p(t_1, \ldots, t_n)$ where p/n is a predicate symbol

$\mid \bot \mid \top \mid \neg F \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid F_1 \rightarrow F_2 \mid F_1 \leftrightarrow F_2$ PL

$\mid \exists x \mathrel{.} F$ exists, existential quantification

$\mid \forall x \mathrel{.} F$ for all, universal quantification

Syntax

Syntax:

- **Grammar**:
  - term: $t ::= x$ occurrence of a variable $x \in \mathbb{X}$
    $\mid f(t_1, \ldots, t_n)$ where f/n is a function symbol
  - formula:

$$
\begin{aligned}
F ::=\ & p(t_1, \ldots, t_n) && \text{where } p/n \text{ is a predicate symbol} \\
& \mid \bot \mid \top \mid \neg F \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid F_1 \rightarrow F_2 \mid F_1 \leftrightarrow F_2 && \text{PL} \\
& \mid \exists x \,.\, F && \text{exists, existential quantification} \\
& \mid \forall x \,.\, F && \text{for all, universal quantification}
\end{aligned}
$$

Example

$$\forall x \,.\, p(x, f(3)) \rightarrow \exists y \,.\, q(y, f(f(f(z))))$$

Syntax

Syntax:

- **Grammar**:
  - term: $t ::= x$ occurrence of a variable $x \in \mathbb{X}$
    $\mid f(t_1, \ldots, t_n)$ where f/n is a function symbol
  - formula:

$$F ::= p(t_1, \ldots, t_n) \quad \text{where } p/n \text{ is a predicate symbol}$$

$$\mid \bot \mid \top \mid \neg F \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid F_1 \rightarrow F_2 \mid F_1 \leftrightarrow F_2 \quad \text{PL}$$

$$\mid \exists x \,.\, F \quad \text{exists, existential quantification}$$

$$\mid \forall x \,.\, F \quad \text{for all, universal quantification}$$

Example

$$\forall x \,.\, p(x, f(3)) \rightarrow \exists y \,.\, q(y, f(f(f(z))))$$

- Precedence
  - PL connectives: as for PL
  - quantifiers: lowest—the scope of a quantifier extends to the right

Syntax

Definitions:

- atomic formulae: those built with the rule $p(t_1, \ldots, t_n)$
- F is a subformula of G: (1) F is a formula and (2) F is a part of G
- the matrix of F: obtained by removing all quantifiers in F

Example

$$F = \exists x_1 \,.\, P_1(x_1, f_1(x_2)) \vee \neg\forall x_2 \,.\, P_2(x_2, f_2(c, f_3(x_3)))$$

- what are the subformulae, terms, and matrix of F?

Syntax — Variables

Variables in formulae:

- **bound**: occur in the scope of a quantifier
  - e.g. $bound(\exists x \,.\, x = 4 \wedge \neg(y = 5)) = \{x\}$

Syntax — Variables

Variables in formulae:

- **bound**: occur in the scope of a quantifier
  - e.g. $bound(\exists x \,.\, x = 4 \land \neg(y = 5)) = \{x\}$
- **free**: there is an occurrence not bound by any quantifier
  - e.g. $free(x = 4 \land (\exists y \,.\, y = 5)) = \{x\}$

Syntax — Variables

Variables in formulae:

- **bound**: occur in the scope of a quantifier
  - e.g. $bound(\exists x \,.\, x = 4 \land \neg(y = 5)) = \{x\}$
- **free**: there is an occurrence not bound by any quantifier
  - e.g. $free(x = 4 \land (\exists y \,.\, y = 5)) = \{x\}$
- a variable can occur both bound and free in a formula

Syntax — Variables

Variables in formulae:

- **bound**: occur in the scope of a quantifier
  - e.g. $\mathit{bound}(\exists x \,.\, x = 4 \wedge \neg(y = 5)) = \{x\}$
- **free**: there is an occurrence not bound by any quantifier
  - e.g. $\mathit{free}(x = 4 \wedge (\exists y \,.\, y = 5)) = \{x\}$
- a variable can occur both bound and free in a formula

Example

$$\forall x \,.\, p(f(x), y) \rightarrow \forall y \,.\, p(f(x), y)$$

- x only occurs bound
- y occurs both free (antecedent) and bound (consequent)

Syntax — Variables

Variables in formulae:

- **bound**: occur in the scope of a quantifier
  - e.g. $\mathit{bound}(\exists x \,.\, x = 4 \wedge \neg(y = 5)) = \{x\}$
- **free**: there is an occurrence not bound by any quantifier
  - e.g. $\mathit{free}(x = 4 \wedge (\exists y \,.\, y = 5)) = \{x\}$
- a variable can occur both bound and free in a formula

Example

$$\forall x \,.\, p(f(x), y) \rightarrow \forall y \,.\, p(f(x), y)$$

- x only occurs bound
- y occurs both free (antecedent) and bound (consequent)

- we often write $F(x_1, \ldots, x_n)$ when $\mathit{free}(F) = \{x_1, \ldots, x_n\}$
  - $x_1, \ldots, x_n$ serve as the "interface" of F

Syntax — Variables

Variables in formulae:

- **bound**: occur in the scope of a quantifier
  - e.g. $bound(\exists x \,.\, x = 4 \land \neg(y = 5)) = \{x\}$
- **free**: there is an occurrence not bound by any quantifier
  - e.g. $\mathit{free}(x = 4 \land (\exists y \,.\, y = 5)) = \{x\}$
- a variable can occur both bound and free in a formula

Example

$$\forall x \,.\, p(f(x), y) \rightarrow \forall y \,.\, p(f(x), y)$$

- x only occurs bound
- y occurs both free (antecedent) and bound (consequent)

- we often write $F(x_1, \ldots, x_n)$ when $\mathit{free}(F) = \{x_1, \ldots, x_n\}$
  - $x_1, \ldots, x_n$ serve as the "interface" of F
- F is ground (or closed) if $\mathit{free}(F) = \emptyset$

Semantics

Semantics of FOL:

- so far, the symbols *did not have any meaning*!

Semantics

Semantics of FOL:

- so far, the symbols *did not have any meaning*!

Structure (or **Interpretation**) $\mathcal{A} = (U_{\mathcal{A}}, I_{\mathcal{A}})$: provides the *meaning* to the symbols

Semantics

Semantics of FOL:

- so far, the symbols *did not have any meaning*!

Structure (or **Interpretation**) $\mathcal{A} = (U_\mathcal{A}, I_\mathcal{A})$: provides the *meaning* to the symbols

- universe $U_\mathcal{A}$: a non-empty set of elements
  - e.g., $\mathbb{N}$, $\{0, 1, 2, 3, 4\}$, $\mathbb{R}^3$, People, $\mathit{List}[\mathbb{N}]$, Σ^*, $\ldots$

Semantics

Semantics of FOL:

- so far, the symbols *did not have any meaning*!

Structure (or **Interpretation**) $\mathcal{A} = (U_\mathcal{A}, I_\mathcal{A})$: provides the *meaning* to the symbols

- universe $U_\mathcal{A}$: a non-empty set of elements
  - e.g., $\mathbb{N}$, $\{0, 1, 2, 3, 4\}$, $\mathbb{R}^3$, People, $\mathit{List}[\mathbb{N}]$, Σ^*, ...
- assignment $I_\mathcal{A}$: a mapping that maps
  - each function symbol f/n to a function $f_I : \overbrace{U_\mathcal{A} \times \ldots \times U_\mathcal{A}}^{n} \to U_\mathcal{A}$
    - e.g., $(+) = \{(0,0) \mapsto 0, (0,1) \mapsto 1, (1,0) \mapsto 1, (1,1) \mapsto 2, \ldots\}$
    - e.g., $\mathit{fatherOf} = \{\mathit{Luke} \mapsto \mathit{Anakin}, \mathit{KyloRen} \mapsto \mathit{HanSolo}, \ldots\}$
    - for constants, this gives us one value, e.g., $\pi = \{() \mapsto 3.1415926\ldots\}$

Semantics

Semantics of FOL:

- so far, the symbols *did not have any meaning*!

Structure (or **Interpretation**) $\mathcal{A} = (U_\mathcal{A}, I_\mathcal{A})$: provides the *meaning* to the symbols

- universe $U_\mathcal{A}$: a non-empty set of elements
  - e.g., $\mathbb{N}$, $\{0, 1, 2, 3, 4\}$, $\mathbb{R}^3$, People, $\mathit{List}[\mathbb{N}]$, Σ^*, ...
- assignment $I_\mathcal{A}$: a mapping that maps
  - each function symbol f/n to a function $f_I : \overbrace{U_\mathcal{A} \times \ldots \times U_\mathcal{A}}^{n} \to U_\mathcal{A}$
    - e.g., $(+) = \{(0,0) \mapsto 0, (0,1) \mapsto 1, (1,0) \mapsto 1, (1,1) \mapsto 2, \ldots\}$
    - e.g., $\mathit{fatherOf} = \{\mathit{Luke} \mapsto \mathit{Anakin}, \mathit{KyloRen} \mapsto \mathit{HanSolo}, \ldots\}$
    - for constants, this gives us one value, e.g., $\pi = \{() \mapsto 3.1415926\ldots\}$
  - each predicate symbol p/n to a function $p_I : \overbrace{U_\mathcal{A} \times \ldots \times U_\mathcal{A}}^{n} \to \mathbb{B}$
    - e.g., $\mathit{isJedi} = \{\mathit{Luke} \mapsto T, \mathit{Anakin} \mapsto T, \mathit{Yoda} \mapsto T, \mathit{ObiWan} \mapsto T, \ldots\}$
    - e.g., $(<) = \{(0,1) \mapsto T, (0,2) \mapsto T, (1,2) \mapsto T, \ldots\}$
    - e.g., $(= 0) = \{0 \mapsto T, 1 \mapsto F, 2 \mapsto F, \ldots\}$
    - e.g., $\mathit{isFatherOf} = \{(\mathit{Anakin}, \mathit{Luke}) \mapsto T, \ldots\}$

Semantics

Semantics of FOL:

- so far, the symbols *did not have any meaning*!

Structure (or **Interpretation**) $\mathcal{A} = (U_\mathcal{A}, I_\mathcal{A})$: provides the *meaning* to the symbols

- universe $U_\mathcal{A}$: a non-empty set of elements
  - e.g., $\mathbb{N}$, $\{0, 1, 2, 3, 4\}$, $\mathbb{R}^3$, People, $\mathit{List}[\mathbb{N}]$, Σ^*, ...
- assignment $I_\mathcal{A}$: a mapping that maps
  - each function symbol f/n to a function $f_I : \overbrace{U_\mathcal{A} \times \ldots \times U_\mathcal{A}}^{n} \to U_\mathcal{A}$
    - e.g., $(+) = \{(0,0) \mapsto 0, (0,1) \mapsto 1, (1,0) \mapsto 1, (1,1) \mapsto 2, \ldots\}$
    - e.g., $\mathit{fatherOf} = \{\mathit{Luke} \mapsto \mathit{Anakin}, \mathit{KyloRen} \mapsto \mathit{HanSolo}, \ldots\}$
    - for constants, this gives us one value, e.g., $\pi = \{() \mapsto 3.1415926\ldots\}$
  - each predicate symbol p/n to a function $p_I : \overbrace{U_\mathcal{A} \times \ldots \times U_\mathcal{A}}^{n} \to \mathbb{B}$
    - e.g.,
      $\mathit{isJedi} = \{\mathit{Luke} \mapsto T, \mathit{Anakin} \mapsto T, \mathit{Yoda} \mapsto T, \mathit{ObiWan} \mapsto T, \ldots\}$
    - e.g., $(<) = \{(0,1) \mapsto T, (0,2) \mapsto T, (1,2) \mapsto T, \ldots\}$
    - e.g., $(= 0) = \{0 \mapsto T, 1 \mapsto F, 2 \mapsto F, \ldots\}$
    - e.g., $\mathit{isFatherOf} = \{(\mathit{Anakin}, \mathit{Luke}) \mapsto T, \ldots\}$
  - each variable $x \in \mathbb{X}$ to a value from $U_\mathcal{A}$, e.g., $\{x \mapsto 42, y \mapsto 0\}$

Semantics – Example of Structure

- A structure $\mathcal{A} = (U_\mathcal{A}, I_\mathcal{A})$ is suitable for F: $I_\mathcal{A}$ is defined over all predicate symbols, function symbols, and free variables of F.

Example

$\forall x \,.\, P(x, f(x)) \wedge Q(g(a, z))$ has a suitable structure $\mathcal{A} = (U_\mathcal{A}, I_\mathcal{A})$ defined as follows.

- $U_\mathcal{A} = \mathbb{N}$,
- $I_\mathcal{A}(P) = \{(m, n) \mapsto T \mid m < n\} \cup \{(m, n) \mapsto F \mid m \geq n\}$,
- $I_\mathcal{A}(Q) = \{n \mapsto T \mid n \text{ is prime}\} \cup \{n \mapsto F \mid n \text{ is not prime}\}$,
- $I_\mathcal{A}(f) = f^\mathcal{A} =$ the successor function, hence $f^\mathcal{A}(n) = n + 1$,
- $I_\mathcal{A}(g) = g^\mathcal{A} =$ the sum function, hence $g^\mathcal{A}(n, m) = n + m$,
- $I_\mathcal{A}(a) = 2, I_\mathcal{A}(z) = 3$.

- Observe that F is “true” under this structure.

Semantics – Example of Structure

- A structure $\mathcal{A} = (U_\mathcal{A}, I_\mathcal{A})$ is suitable for F: $I_\mathcal{A}$ is defined over all predicate symbols, function symbols, and free variables of F.

Example

$\forall x . P(x, f(x)) \wedge Q(g(a, z))$ has a suitable structure $\mathcal{A} = (U_\mathcal{A}, I_\mathcal{A})$ defined as follows.

- $U_\mathcal{A} = \mathbb{N}$,
- $I_\mathcal{A}(P) = \{(m, n) \mapsto T \mid m < n\} \cup \{(m, n) \mapsto F \mid m \geq n\}$,
- $I_\mathcal{A}(Q) = \{n \mapsto T \mid n$ is prime$\} \cup \{n \mapsto F \mid n$ is not prime$\}$,
- $I_\mathcal{A}(f) = f^\mathcal{A} =$ the successor function, hence $f^\mathcal{A}(n) = n + 1$,
- $I_\mathcal{A}(g) = g^\mathcal{A} =$ the sum function, hence $g^\mathcal{A}(n, m) = n + m$,
- $I_\mathcal{A}(a) = 2, I_\mathcal{A}(z) = 3$.

- Observe that F is “true” under this structure.
- Could you define a suitable structure in which F is “false”?

$U_{\mathcal{A}}$ needs not to be a set of numbers.

Example

Below we define a suitable structure $\mathcal{A} = (U_{\mathcal{A}}, I_{\mathcal{A}})$ for $F = \forall x \,.\, P(a, f(x))$.

- $U_{\mathcal{A}} =$ all varaible free terms from the symbols of $F = \{a, f(a)), f(f(a)), f(f(f(a))), \ldots)\}$,
- $I_{\mathcal{A}}(f) = f^{\mathcal{A}}$, where $f^{\mathcal{A}}(t) = t$,
- $I_{\mathcal{A}}(a) = a$,
- $I_{\mathcal{A}}(P) =?$.

Semantics

- The value of terms:
  $\mathcal{A}(x) \stackrel{\text{def}}{=} I_{\mathcal{A}}(x), \quad \mathcal{A}(f(t_1, \ldots, t_n)) \stackrel{\text{def}}{=} I_{\mathcal{A}}(f)\mathbf{(}\mathcal{A}(t_1), \ldots, \mathcal{A}(t_n)\mathbf{)}$

Semantics

- The value of terms:
  $\mathcal{A}(x) \stackrel{\text{def}}{=} I_{\mathcal{A}}(x), \quad \mathcal{A}(f(t_1, \ldots, t_n)) \stackrel{\text{def}}{=} I_{\mathcal{A}}(f)\mathbf{(}\mathcal{A}(t_1), \ldots, \mathcal{A}(t_n)\mathbf{)}$
- The truth value of a formula:
  - $\mathcal{A} \models P(t_1, \ldots, t_n)$ iff $I_{\mathcal{A}}(P)\mathbf{(}\mathcal{A}(t_1), \ldots, \mathcal{A}(t_n)\mathbf{)} = T$
    $\mathcal{A} \models \neg F$ iff $\mathcal{A} \not\models F$
    $\mathcal{A} \models G \wedge H$ iff $\mathcal{A} \models G$ and $\mathcal{A} \models H$
    $\mathcal{A} \models G \vee H$ iff $\mathcal{A} \models G$ or $\mathcal{A} \models H$
    $\mathcal{A} \models \forall x.G$ iff for all $u \in U_{\mathcal{A}}$, $\mathcal{A}_{[x \mapsto u]} \models G$
    $\mathcal{A} \models \exists x.G$ iff there exists $u \in U_{\mathcal{A}}$, $\mathcal{A}_{[x \mapsto u]} \models G$
  - $\mathcal{A} \models F$ means F is true in $\mathcal{A}$, or $\mathcal{A}$ is a model of F.
  - if F has a model, then F is satisfiable, otherwise unsatisfiable
  - if $\mathcal{A} \models F$ for all possible suitable structure $\mathcal{A}$, then F is valid
  - substitution : $\mathcal{A}_{[x \mapsto u]}$ is identical to $\mathcal{A}$ with the exception $I_{\mathcal{A}_{[x \mapsto u]}}(x) = u$.

Semantics

- The value of terms:
  $\mathcal{A}(x) \stackrel{\text{def}}{=} I_\mathcal{A}(x), \quad \mathcal{A}(f(t_1, \ldots, t_n)) \stackrel{\text{def}}{=} I_\mathcal{A}(f)\big(\mathcal{A}(t_1), \ldots, \mathcal{A}(t_n)\big)$

- The truth value of a formula:
  - $\mathcal{A} \models P(t_1, \ldots, t_n)$ iff $I_\mathcal{A}(P)\big(\mathcal{A}(t_1), \ldots, \mathcal{A}(t_n)\big) = T$
    $\mathcal{A} \models \neg F$ iff $\mathcal{A} \not\models F$
    $\mathcal{A} \models G \wedge H$ iff $\mathcal{A} \models G$ and $\mathcal{A} \models H$
    $\mathcal{A} \models G \vee H$ iff $\mathcal{A} \models G$ or $\mathcal{A} \models H$
    $\mathcal{A} \models \forall x.G$ iff for all $u \in U_\mathcal{A}$, $\mathcal{A}_{[x \mapsto u]} \models G$
    $\mathcal{A} \models \exists x.G$ iff there exists $u \in U_\mathcal{A}$, $\mathcal{A}_{[x \mapsto u]} \models G$
  - $\mathcal{A} \models F$ means F is true in $\mathcal{A}$, or $\mathcal{A}$ is a model of F.
  - if F has a model, then F is satisfiable, otherwise unsatisfiable
  - if $\mathcal{A} \models F$ for all possible suitable structure $\mathcal{A}$, then F is valid
  - substitution : $\mathcal{A}_{[x \mapsto u]}$ is identical to $\mathcal{A}$ with the exception $I_{\mathcal{A}_{[x \mapsto u]}}(x) = u$.

- Question: we no more have Boolean variables! Is that a problem?

Exercise

Consider the formula

$$F = \forall x. \exists y.\ P(x, y, f(z)).$$

Define a structure $\mathcal{A} = (\mathbb{N}, I_{\mathcal{A}})$ that is a model of F, and another structure $\mathcal{B} = (\mathbb{N}, I_{\mathcal{B}})$ that is not a model of F.

Semantics — Examples

Consider the signature $(\{(+)/2\}, \{(=)/2\})$

Semantics — Examples

Consider the signature $(\{(+)/2\}, \{(=)/2\})$

- Addition in $\mathbb{N}$: $\mathcal{A} = (\mathbb{N}, I_{\mathcal{A}})$ where
  - $I_{\mathcal{A}}(+) = (+_{\mathbb{N}})$
  - $I_{\mathcal{A}}(=)$ maps pairs in $\{(n, n) \mid n \in \mathbb{N}\}$ to T and others to F
  - $(=)$ is often considered an “inbuilt” predicate of FOL (regardless of the signature) with the standard meaning (identity)

Semantics — Examples

Consider the signature $(\{(+)/2\}, \{(=)/2\})$

- Addition in $\mathbb{N}$: $\mathcal{A} = (\mathbb{N}, I_{\mathcal{A}})$ where
  - $I_{\mathcal{A}}(+) = (+_{\mathbb{N}})$
  - $I_{\mathcal{A}}(=)$ maps pairs in $\{(n, n) \mid n \in \mathbb{N}\}$ to T and others to F
  - $(=)$ is often considered an "inbuilt" predicate of FOL (regardless of the signature) with the standard meaning (identity)
- Addition in $\mathbb{R}^3$: $\mathcal{A} = (\mathbb{R}^3, I_{\mathcal{A}})$ where
  - $I_{\mathcal{A}}(+) = \{((x_1, y_1, z_1), (x_2, y_2, z_2)) \mapsto (x_1 + x_2, y_1 + y_2, z_1 + z_2)\}$

Semantics — Examples

Consider the signature $(\{(+)/2\}, \{(=)/2\})$

- Addition in $\mathbb{N}$: $\mathcal{A} = (\mathbb{N}, I_{\mathcal{A}})$ where
  - $I_{\mathcal{A}}(+) = (+_{\mathbb{N}})$
  - $I_{\mathcal{A}}(=)$ maps pairs in $\{(n, n) \mid n \in \mathbb{N}\}$ to T and others to F
  - $(=)$ is often considered an "inbuilt" predicate of FOL (regardless of the signature) with the standard meaning (identity)
- Addition in $\mathbb{R}^3$: $\mathcal{A} = (\mathbb{R}^3, I_{\mathcal{A}})$ where
  - $I_{\mathcal{A}}(+) = \{((x_1, y_1, z_1), (x_2, y_2, z_2)) \mapsto (x_1 + x_2, y_1 + y_2, z_1 + z_2)\}$
- Disjunction in Boolean algebra: $\mathcal{A} = (\{0, 1\}, I_{\mathcal{A}})$
  - $I_{\mathcal{A}}(+) = \vee$

Semantics — Examples

Consider the signature $(\{(+)/2\}, \{(=)/2\})$

- Addition in $\mathbb{N}$: $\mathcal{A} = (\mathbb{N}, I_{\mathcal{A}})$ where
  - $I_{\mathcal{A}}(+) = (+_{\mathbb{N}})$
  - $I_{\mathcal{A}}(=)$ maps pairs in $\{(n, n) \mid n \in \mathbb{N}\}$ to T and others to F
  - $(=)$ is often considered an "inbuilt" predicate of FOL (regardless of the signature) with the standard meaning (identity)
- Addition in $\mathbb{R}^3$: $\mathcal{A} = (\mathbb{R}^3, I_{\mathcal{A}})$ where
  - $I_{\mathcal{A}}(+) = \{((x_1, y_1, z_1), (x_2, y_2, z_2)) \mapsto (x_1 + x_2, y_1 + y_2, z_1 + z_2)\}$
- Disjunction in Boolean algebra: $\mathcal{A} = (\{0, 1\}, I_{\mathcal{A}})$
  - $I_{\mathcal{A}}(+) = \vee$
- Modular addition in $\{0, 1, 2, 3\}$: $\mathcal{A} = (\{0, 1, 2, 3\}, I_{\mathcal{A}})$ where
  - $I_{\mathcal{A}}(+) = \{(x, y) \mapsto x + y \mod 4\}$

Exercise

The following formulas F_1, F_2, F_3 express that the predicate P is reflexive, symmetric, and transitive.

$$
\begin{array}{lcl}
F_1 & = & \forall x.P(x,x) \\
F_2 & = & \forall x.\forall y.(P(x,y) \to P(y,x)) \\
F_3 & = & \forall x.\forall y.\forall z.((P(x,y) \land P(y,z)) \to P(x,z))
\end{array}
$$

Show that none of them is a consequence of the other two by presenting structures that are models of two of the formulas, but not for the third one.

Normal Forms

Equivalences

Two formulas F and G are equivalent (written as $F \equiv G$) if $\mathcal{A}(F) = \mathcal{A}(F)$ for all suitable structures.

Example

Those we have seen in propositional logic

$$
\begin{aligned}
F &\equiv \neg\neg F && \text{(double negative elimination)} \\
\neg(F \land G) &\equiv \neg F \lor \neg G && \text{(De Morgan's law)} \\
F \leftrightarrow G &\equiv (F \to G) \land (G \to F) && \\
F \land (G \land H) &\equiv (F \land G) \land H && \text{(associativity)} \\
F \land (G \lor H) &\equiv (F \land G) \lor (F \land H) && \text{(distributivity)}
\end{aligned}
$$

Non-Propositional Equivalences

There are of course new equivalences

$$
\begin{array}{llcll}
1. & \forall x.\ \neg F & \equiv & \neg \exists x.\ F & \\
 & \exists x.\ \neg F & \equiv & \neg \forall x.\ F & \\
2. & (\forall x.\ F) \wedge (\forall x\ .\ G) & \equiv & \forall x.\ F \wedge G & \\
 & (\exists x.\ F) \vee (\exists x\ .\ G) & \equiv & \exists x.\ F \vee G & \\
3. & \forall x.\ F \circ G & \equiv & (\forall x.\ F) \circ G & \text{if } x \notin \mathit{free}(G), \circ \in \{\wedge, \vee\} \\
 & \exists x.\ F \circ G & \equiv & (\exists x.\ F) \circ G & \text{if } x \notin \mathit{free}(G), \circ \in \{\wedge, \vee\} \\
4. & \forall x. \forall y.\ F & \equiv & \forall y. \forall x.\ F & \\
 & \exists x. \exists y.\ F & \equiv & \exists y. \exists x.\ F &
\end{array}
$$

We sometimes write $\forall x, y$ and $\exists x, y$ as shorthand for $\forall x. \forall y$ and $\exists x. \exists y$.

Non-Propositional Equivalences (Correctness)

As an example, we prove the correctness of $\forall x.\ F \land G \equiv (\forall x.\ F) \land G$, if $x \notin \mathit{free}(G)$. Let $\mathcal{A} = (U_{\mathcal{A}}, I_{\mathcal{A}})$ be a structure suitable for both sides of the equivalence.

$$
\begin{aligned}
& \mathcal{A} \models (\forall x.\ F) \land G \\
\text{iff } & \text{for all } u \in U_{\mathcal{A}}, \mathcal{A}_{[x \mapsto u]} \models F \text{ and } \mathcal{A} \models G \\
\text{iff } & \text{for all } u \in U_{\mathcal{A}}, \mathcal{A}_{[x \mapsto u]} \models F \text{ and } \mathcal{A}_{[x \mapsto u]} \models G \qquad (x \notin \mathit{free}(G)) \\
\text{iff } & \text{for all } u \in U_{\mathcal{A}}, \mathcal{A}_{[x \mapsto u]} \models F \land G \\
\text{iff } & \mathcal{A} \models \forall x.\ F \land G
\end{aligned}
$$

Non-Propositional Equivalences

Some examples with very similar looking formulas, but are not equivalent

Equivalent

$$
\begin{aligned}
(\forall x.\ F) \wedge (\forall x\ .\ G) &\quad\equiv\quad \forall x.\ F \wedge G \\
(\exists x.\ F) \vee (\exists x\ .\ G) &\quad\equiv\quad \exists x.\ F \vee G
\end{aligned}
$$

Inequivalent

$$
\begin{aligned}
(\forall x.\ F) \vee (\forall x\ .\ G) &\quad\not\equiv\quad \forall x.\ F \vee G \\
(\exists x.\ F) \wedge (\exists x\ .\ G) &\quad\not\equiv\quad \exists x.\ F \wedge G
\end{aligned}
$$

Can you confirm this by exhibiting counterexamples?

Negation Normal Form (NNF)

A formula is in Negation Normal Form (NNF) if $\neg$ appears only in front of predicates

Negation Normal Form (NNF)

A formula is in Negation Normal Form (NNF) if $\neg$ appears only in front of predicates

Example

Let

$$F : \neg\exists n, x, y \, . \, n > 2 \quad \wedge \quad \exists z \, . \, x^n + y^n = z^n.$$

The formula

$$G : \forall n, x, y \, . \, \neg(n > 2) \quad \vee \quad \forall z \, . \, \neg(x^n + y^n = z^n)$$

is equivalent to F and is in NNF.

Negation Normal Form (NNF)

A formula is in Negation Normal Form (NNF) if $\neg$ appears only in front of predicates

Example

Let

$$F : \neg\exists n, x, y \,.\, n > 2 \quad \wedge \quad \exists z \,.\, x^n + y^n = z^n.$$

The formula

$$G : \forall n, x, y \,.\, \neg(n > 2) \quad \vee \quad \forall z \,.\, \neg(x^n + y^n = z^n)$$

is equivalent to F and is in NNF.

Question: how to get the NNF using the equivalence rules?

Prenex Normal Form (PNF)

- A formula is in Prenex Normal Form (PNF) is of the form

$$F = \underbrace{Q_1 x_1 \,.\, \ldots Q_n x_n}_{\text{prefix}} \,.\, \underbrace{G(x_1, \ldots, x_n, y_1, \ldots, y_m)}_{\text{matrix}}$$

  where $Q_i \in \{\forall, \exists\}$ and G is quantifier-free; $\{y_1, \ldots, y_m\}$ are the free variables of F

Prenex Normal Form (PNF)

- A formula is in Prenex Normal Form (PNF) is of the form

$$F = \underbrace{Q_1 x_1 \, . \, \ldots Q_n x_n}_{\text{prefix}} \, . \, \underbrace{G(x_1, \ldots, x_n, y_1, \ldots, y_m)}_{\text{matrix}}$$

  where $Q_i \in \{\forall, \exists\}$ and G is quantifier-free; $\{y_1, \ldots, y_m\}$ are the free variables of F

Example

Let

$$G : \forall n, x, y \, . \, \neg(n > 2) \quad \vee \quad \forall z \, . \, \neg(x^n + y^n = z^n).$$

The formula

$$H : \forall n, x, y, z \, . \, \neg(n > 2) \quad \vee \quad \neg(x^n + y^n = z^n)$$

is equivalent to G and is in PNF.

Equivalence so far are not enough

They are sometimes enough to produce to produce a formula in PNF:

Equivalence so far are not enough

They are sometimes enough to produce to produce a formula in PNF:

Example

$$\forall x \,.\, (x = z \vee \exists y \,.\, R(x,y)) \equiv \forall x . \exists y \,.\, (x = z \vee R(x,y))$$

The subformula $x = z$ does not have the variable y.

But in general, we need more rules, e.g.:

Example

$\exists x \,.\, P(x) \wedge \exists x \,.\, Q(x) \equiv ?$

To convert this to PNF, we need to apply "variable renaming"

Substitutions

We formalize variable renaming by substitutions

Definition

Given a variable x and a term t, we write $F[x \mapsto t]$ to denote the formula obtained by substituting all free occurrence of x in F to t.

Exercise

If $F = (\forall x \, . \, P(x)) \wedge Q(x)$, what is $F[x \mapsto f(x)]$?

Exercise

Give a recursive definition of $F[x \mapsto t]$.

Substitutable Terms

Unrestricted substitutions cause problems with "scoping"

Definition

A term t is substitutable for x in F if no variable in t occur bound in F

Example

$y + 5$ is not substitutable for x in the following formula

$$\forall y \,.\, (x + 3z = y)$$

Key: use only substitutable terms in substitution

Substitution Lemma

Lemma

Suppose t is a term that is substitutable for x in F. Then

$$\mathcal{A} \models F[x \mapsto t] \text{ iff } \mathcal{A}_{[x \mapsto \mathcal{A}(t)]} \models F$$

Can be proved by structural induction. Detailed proof can be found in the note of Eric Pacuit (`https://pdfs.semanticscholar.org/2b67/95e57bb5b2f63d46ced447952d9a00b0f33b.pdf`, pages 8-9)

Corollary

Let y be a variable not in $\forall x \,.\, F$. Then

$$\forall x \,.\, F = \forall y \,.\, (F[x \mapsto y])$$

The same hold for the $\exists$ counterpart.

Cleansing a Formula

By variable renaming in F, each variable can be made:

- to occur only free or bound in F, and
- to be quantified in F at most once.

Exercise

Cleansing the following formulas

- $\exists y\ .\ R(x) \wedge \exists x\ .\ P(x)$
- $\forall x\ .\ (x \neq x + 1) \wedge \exists y\ .\ (x = y)$

Conversion to PNF

- Cleansing the formula
- Convert to NNF
- Keeping applying the equivalences to bring the quantifiers out

Exercise

Turn the following into PNF

$$\forall x \,.\, (G(x,x) \wedge \neg(\exists y \,.\, \neg G(x,y) \wedge \forall y \,.\, G(y,y))) \wedge G(x,0)$$

Skolem Normal Form (SNF)

- A formula in Skolem Normal Form (SNF) if it is in PNF and there is no occurrence of an existential quantifier in it.
- In general, it is not always possible to find an equivalent formula in SNF for a given FOL formula.

Theorem

Every FOL formula can be converted into an equisatisfiable one in SNF (possibly over a different alphabet)

Skolemization

Methods to eliminate existential quantifiers

Lemma

Suppose $F = \forall x_1, \ldots, x_m . \exists y \, . \, G$ *and let* f/n *be a function symbol not in* G. *The following formula is equisatisfiable to* F

$$\forall x_1, \ldots, x_m . G[y \mapsto f(x1, \ldots, x_m)].$$

Conversion to SNF

- Turn the formula into a cleansed one in PNF
- Apply Skolemization from the outermost existential quantifier

Exercise

Turn the following formula into SNF

$$\forall x\ .\ \exists y\ .\ \forall x'\ .\ \exists y'\ .\ R(x, y, x', y')$$

Herbrand's Theorem

In a Nutshell

- The theorem enables a systematic approach to decide if a FOL formula is unsatisfiable or valid.
- We know how to do it in propositional logic (the possible models are finite).
- This is not easy in FOL. The possible suitable structures of a FOL can be an infinite set.

Herbrand Universe

- the Herbrand universe $D(F)$ of a closed formula F in Skolem form is the set of all ground (variable-free) terms that can be built from the components of F.
- when F does not contains any constant, we choose an arbitrary constant, say a, and use it to build up the variable-free terms.
- more precisely,
  - Every constant in F is also in $D(F)$. If F has no constant, then $a \in D(F)$.
  - For all function f/k in F and for all terms $t_1, \ldots, t_k$ already in $D(F)$, the term $f(t_1, \ldots, t_k) \in D(F)$.

Herbrand Universe

- Recall below the formal definition of Harbrand universe.
  - Every constant in F is also in $D(F)$. If F has no constant, then $a \in D(F)$.
  - For all function f/k in F and for all terms $t_1, \ldots, t_k$ already in $D(F)$, the term $f(t_1, \ldots, t_k) \in D(F)$.

Example

Consider the formulas

$$F = \forall x, y, z \,.\, P(x, f(y), g(z, x))$$

$$G = \forall x, y \,.\, Q(c, f(x), h(y, b))$$

.

The formula F does not contain a constant, Therefore

$$D(F) = \{a, f(a), g(a, a), f(f(a)), f(g(a, a)), g(a, f(a)), \ldots\}$$

$$D(G) = \{b, c, f(b), f(c), h(b, b), h(b, c), h(c, b), h(c, c), f(f(b)), \ldots\}$$

Herbrand Structure

Let F be a closed form formula in SNF. A structure $\mathcal{A} = (U_{\mathcal{A}}, I_{\mathcal{A}})$ is called a Herbrand structure for F if the following holds

- $U_{\mathcal{A}} = D(F)$,
- For all function symbol f/k in F and terms $t_1, t_2, \ldots, t_k \in D(F)$, $\mathcal{A}(f(t_1, \ldots, t_k)) = f(t_1, \ldots, t_k)$.
- One can freely choose the mapping (interpretation) for predicate symbols.

Herbrand Structure

Example

The Herbrand structure $\mathcal{A} = (U_\mathcal{A}, I_\mathcal{A})$ for $F = \forall x, y, z \,.\, P(x, f(y), g(z, x))$ has the following properties.

$$U_\mathcal{A} = D(F) = \{a, f(a), g(a, a), f(f(a)), f(g(a, a)), g(a, f(a)), \ldots\}$$

and

$$\mathcal{A}(f(t)) = f(t), \mathcal{A}(g(t_1, t_2)) = g(t_1, t_2), \text{ for all } t, t_1, t_2 \in D(F)$$

The choice of $I_\mathcal{A}(P)$ is still free. E.g., one can define $I_\mathcal{A}(P)(t_1, t_2, t_3) = T$ iff $g(t_1, t_2) = g(t_2, f(t_3))$. (is $\mathcal{A}$ a model of F?)

Facts about Herbrand Structure

Proposition

The value $\mathcal{A}(t)$ of a ground term t in a Herbrand structure $\mathcal{A}$ is t

Substitution lemma has a simplified form

Lemma

Suppose t is substitutable for x in F and $\mathcal{A}$ a Herbrand structure. Then

$$\mathcal{A} \models F[x \mapsto u] \text{ iff } \mathcal{A}_{[x \mapsto u]} \models F$$

We call a Herbrand structure $\mathcal{A}$ of a formula F a Herbrand model for F, if it is a model of F.

Herbrand's Theorem

Theorem

Let F be a closed formula in Skolem normal form. Then F is satisfiable iff F has a Herbrand model.

Example

Prove satisfiability of

$$\exists x, y, z \,.\, (P(x) \rightarrow P(y)) \land (P(y) \rightarrow P(z)) \land \neg P(z)$$

Skolemize:

$$(P(a) \rightarrow P(b)) \land (P(b) \rightarrow P(c)) \land \neg P(c)$$

Herbrand model has the universe $\{a, b, c\}$

- enumerate all such models.

Proof of Herbrand's Theorem

Assume a closed formula F that is in SNF.

- Prove: exists a model $\mathcal{A}$ implies exists a Herbrand model $\mathcal{H}$
- Idea: Define $\mathcal{H}$ that "mimics" $\mathcal{A}$

  $$I_{\mathcal{H}}(P)(t_1, \ldots, t_k) = T \text{ iff } \mathcal{A} \models P(t_1, \ldots, t_k)$$

  for all $t_1, \ldots, t_k \in D(F)$.
- Claim: For all n (# quantifiers), $\mathcal{A} \models G$ implies $\mathcal{H} \models G$.
  - G is any closed formula in PNF that is built from the same function symbols and predicate symbols of F
  - Proof by induction on n
  - $\underline{n = 0 :}$ G is a boolean combination of ground terms. Immediate.

Proof of Herbrand's Theorem

- $I_{\mathcal{H}}(P)(t_1, \ldots, t_k) = T$ iff $\mathcal{A} \models P(t_1, \ldots, t_k)$
- Claim: For all n (# quantifiers), $\mathcal{A} \models G$ implies $\mathcal{H} \models G$.
  - $\underline{n > 0:}$ We have $\mathcal{A} \models \forall x \,.\, G'$
  - Problem: G' is not ground, so (IH) cannot be applied.
  - Key: use substitution lemma

$$
\begin{array}{llll}
 & \mathcal{A} \models G & & \\
\Longrightarrow & \mathcal{A}_{[x \mapsto u]} \models G' & \text{for all } u \in U_{\mathcal{A}} & \text{(definition of } \forall) \\
\Longrightarrow & \mathcal{A}_{[x \mapsto \mathcal{A}(t)]} \models G' & \text{for all } t \in D(G) & (\mathcal{A}(t) \in U_{\mathcal{A}}) \\
\Longrightarrow & \mathcal{A} \models G'[x \mapsto t] & \text{for all } t \in D(G) & \text{(by sub. lem.)} \\
\Longrightarrow & \mathcal{H} \models G'[x \mapsto t] & \text{for all } t \in D(G) & \text{(by IH)} \\
\Longrightarrow & \mathcal{H}_{[x \mapsto t]} \models G' & \text{for all } t \in D(G) & \text{(by sub. lem.)} \\
\Longrightarrow & \mathcal{H} \models G & & \text{(definition of } \forall)
\end{array}
$$

Ground Resolution Theorem (a.k.a Gödel-Herbrand-Skolem Theorem)

Herbrand Expansion

Let $F = \forall x_1, \ldots, x_k \,.\, G$ be a closed formula in SNF, where G is quantifier free.

Definition

The Herbrand expansion of F, denoted $E(F)$, is defined as

$$E(F) = \{G[x_1 \mapsto t_1][x_2 \mapsto t_2] \cdots [x_k \mapsto t_k] \mid t_1, t_2 \ldots, t_k \in D(F)\}$$

Example

$$F = \forall x, y \,.\, P(x, f(y))$$

The elements in $E(F)$ includes

$$\begin{array}{rcl} P(a, f(a)) & \text{using} & [x \mapsto a][y \mapsto a] \\ P(f(a), f(a)) & \text{using} & [x \mapsto f(a)][y \mapsto a] \\ P(f(a), f(f(a))) & \text{using} & [x \mapsto f(a)][y \mapsto f(a)] \end{array}$$

$$\ldots$$

Ground Resolution Theorem

Let $F = \forall x_1, \ldots, x_k \,.\, G$ be a closed formula in SNF.

Theorem (Gödel-Herbrand-Skolem)

The formula F is satisfiable
iff
$E(F)$ is satisfiable in propositional logic

Observe that $E(F)$ can be treated as formulas in propositional logic because they do not contain variable.

Corollary

The formula F is unsatisfiable
iff
There is a finite unsatisfiable subset of $E(F)$

From compactness of propositional logic

Proof of GRT

Theorem (Gödel-Herbrand-Skolem)

The formula F is satisfiable
iff
$E(F)$ is satisfiable in propositional logic

Let F have the form $F = \forall x_1, x_2, \ldots, x_n \,.\, F^*$.

$$
\begin{array}{rl}
 & F \text{ is satisfiable} \\
\overset{\text{Herbrand Thm}}{\Longleftrightarrow} & \mathcal{H} \models F \text{ for a Herbrand model } \mathcal{H} \\
\overset{\text{def. of } \forall}{\Longleftrightarrow} & \mathcal{H}_{[x_1 \mapsto t_1]\ldots[x_n \mapsto t_n]} \models F^*, \text{ for all } t_1, t_2, \ldots, t_n \in D(F) \\
\overset{\text{Sub. Lemma}}{\Longleftrightarrow} & \mathcal{H} \models F^*[x_1 \mapsto t_1] \ldots [x_n \mapsto t_n], \text{ for all } t_1, t_2, \ldots, t_n \in D(F) \\
\overset{\text{def. of } E(F)}{\Longleftrightarrow} & \mathcal{H} \models G, \text{ for all } G \in E(F) \\
\Longleftrightarrow & \mathcal{H} \text{ is a model of } E(F)
\end{array}
$$

Glimore's Procedure

- Input: A closed formula F in SNF
- Task: Determine whether it is unsatisfiable
- Procedure:

  Let $E(F) = \{F_1, F_2, \ldots, F_n, \ldots\}$
  $n := 0$
  While $F_1 \wedge \ldots \wedge F_n$ is satisfiable:
      $n := n + 1$
  return "unsatisfiable"

Semi-decidability of FOL Validity

- Input: A formula F
- Task: Determine whether it is valid
- Procedure:
  1. Convert $\neg F$ to a formula F' in SNF
  2. Run Gilmore's procedure on F'
  3. If "unsatisfiable" was returned in (2), return "valid"

Exercise

Use Gilmore's procedure to show the formulas are valid

$$(\forall x \,.\, P(x) \rightarrow P(f(x))) \rightarrow (\forall x \,.\, P(x) \rightarrow P(f(f(x))))$$

$$\forall x . \exists y \,.\, (P(x) \rightarrow Q(y)) \rightarrow \exists y . \forall x \,.\, (P(x) \rightarrow Q(y))$$

Question

What happens if we run Gilmore's procedure on the formula below?

$$F = \forall x \, . \, (x < s(x))$$

Undecidability of FOL Validity

Theorem (Church-Turing)

FOL validity is undecidable.

Perhaps the most important in the theory of computation, and a negative answer to the famous challenge (Entscheidungsproblem, in English "Decision Problem") posed by Hilbert and Ackermann in 1928.

Resolution for FOL

The Setup

The formula is now in Skolem Clausal Form:
a cleansed formula in SNF,
where the quantifier-free part (the matrix) is in CNF.

Example (Clause Form)

We often represent a CNF formula

$$(l_1 \vee l_2 \vee l_3) \wedge (l_4 \vee l_5)$$

in clause form (as a set of clauses) as follows

$$\{\{l_1, l_2, l_3\}\{l_4, l_5\}\}$$

Ground Resolution Procedure

- Input: A closed formula F in SNF with $E(F) = \{F_1, F_2, \ldots, F_n\}$
- Task: Determine whether it is unsatisfiable
- Procedure:

  $i := 0, M = \emptyset$
  While $\perp \notin M$:
  - $n := n + 1$
  - $M := M \cup F_i$
  - $M := Res^*(M)$

  return "unsatisfiable"

Remarks: Resolution Proof in Propositional Logic

Definition

Resolution proof rule:

$$\frac{\{a_1, a_2, \ldots, a_n, c\} \quad \{b_1, b_2, \ldots, b_n, \neg c\}}{\{a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_n\}}$$

Lemma

A set of clauses that does not any inconsistent pair of propositions p, $\neg p$ is satisfiable.

Example

$\{\{p, q, \neg r\}\{q, s\}\{p, \neg r, s\}\}$ is satisfiable

Remarks: Resolution Proof in Propositional Logic

Example

$\{\{r\}\{p, \neg r\}\{q, \neg r\}\{\neg p, \neg q\}\}$ is unsatisfiable

Example

$$\dfrac{\dfrac{\{q, \neg r\} \quad \{r\}}{\{q\}} \qquad \dfrac{\dfrac{\{r\} \quad \{p, \neg r\}}{\{p\}} \quad \{\neg p, \neg q\}}{\{\neg q\}}}{\{\} = \bot}$$

Theorem

Resolution proof is sound and complete for propositional logic

Ground Resolution Example

Example

Prove the following formula is UNSAT using ground resolution

$$F = \forall x \,.\, P(x) \wedge \neg P(f(x))$$

The matrix of $F = \{\{P(x)\}\{\neg P(f(x))\}\}$

Already the first 2 ground substitutions $[x \mapsto a], [x \mapsto f(a)]$, that is, the first two elements in $E(F)$ lead to UNSAT.

$$\{P(a)\} \qquad \{\neg P(f(a))\} \qquad \{P(f(a))\} \qquad \{\neg P(f(f(a)))\}$$

$$\perp$$

Ground Resolution Diagrammatically

Substitution steps can be represented as an initial step of a resolution proof for propositional logic.

$$
\begin{tikzcd}
\{R(x)\} \arrow[d, dash, "{[x \mapsto f(c)]}"] & & \{\neg R(f(x))\} \arrow[d, dash, "{[x \mapsto c]}"] \\
R(f(c)) \arrow[dr, dash] & & \neg R(f(c)) \arrow[dl, dash] \\
& \bot &
\end{tikzcd}
$$

Example

Prove the following formula is UNSAT using ground resolution

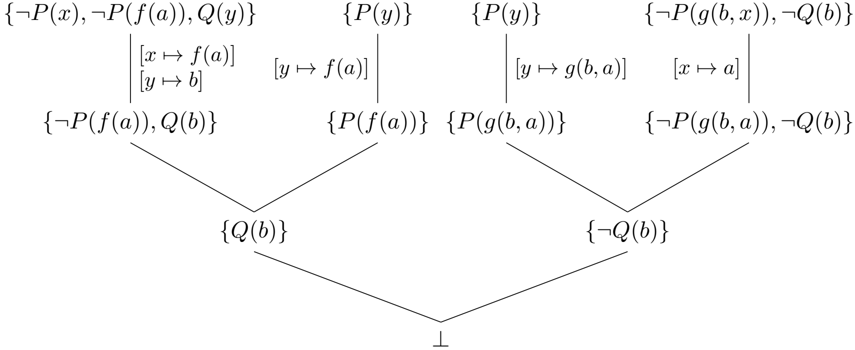
$$\forall x, y \,.\, \textbf{(}\neg P(x) \vee \neg P(f(a)) \vee Q(y)\textbf{)} \wedge \textbf{(}P(y)\textbf{)} \wedge \textbf{(}\neg P(g(b,x)) \vee \neg Q(b)\textbf{)}$$

Example

Prove the following formula is UNSAT using ground resolution

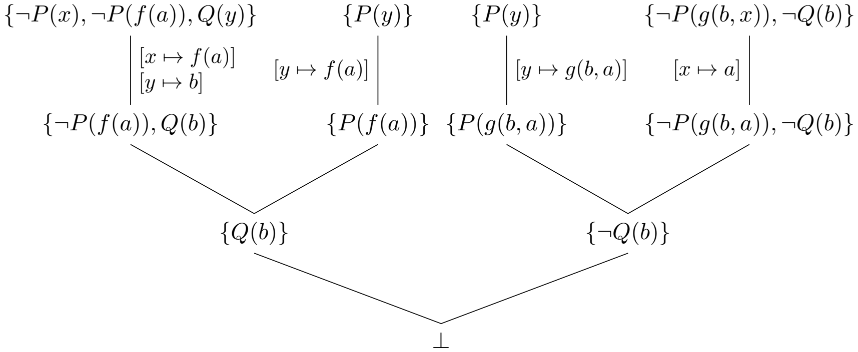
$$\forall x, y \,.\, \mathbf{(}\neg P(x) \vee \neg P(f(a)) \vee Q(y)\mathbf{)} \wedge \mathbf{(}P(y)\mathbf{)} \wedge \mathbf{(}\neg P(g(b,x)) \vee \neg Q(b)\mathbf{)}$$

Solution:



Unification and General Resolution

Problem with General Resolution



A blind (albeit systematic) enumeration of ground clauses will generate lots of irrelevant ground clauses
Solution: do pattern matching like human (a.k.a. unification).

Unification

Unification will allow us to do resolution with non-ground clauses.

Simultaneous Substitution

- A simultaneous substitution is a mapping from variables to terms (not necessarily ground terms).
- We write

$$\theta = [x_1 \ldots, x_n \mapsto t_1, \ldots, t_n]$$

to denote the simultaneous substitution θ that maps x_1 to t_1, x_2 to t_2,. . ., x_n to t_n, and y to itself for every variable $y \notin \{x_1, \ldots, x_n\}$.

Example

$$F = P(x) \wedge Q(f(y))$$

Then,

$$F[x, y \mapsto y, f(a)] = P(y) \wedge Q(f(f(a)))$$

.

Composing Substitution

Given two substitutions θ_1 and θ_2 their composition $\theta_1 \circ \theta_2$ is the substitution mapping x to $x\theta_1\theta_2$.

Example

Let

$$\theta_1 = [x, y \mapsto f(y), a]$$

and

$$\theta_2 = [y \mapsto g(a)]$$

. Then $\theta_1 \circ \theta_2 = [x, y \mapsto f(g(a)), a]$

Unifiers

- A literal is a predicate or a negation of a predicate
- A unifier for a set U of literals is a substitution that equates all literals in U. If U has a unifier, then it is unifiable.

Example

$U = \{P(f(x), g(y)), P(f(f(a)), g(z))\}$
Some unifiers for U:
$\theta_1 = [x, y, z \mapsto f(a), a, a]$
$\theta_2 = [x, y \mapsto f(a), z]$.

- A most general unifier (mgu) for U is a unifier θ such that each unifier θ' can factor through θ, i.e.,

$$\theta' = \theta \circ \gamma \text{ for some substitution } \gamma$$

Example

θ_2 is an mgu for U. Note that $\theta_1 = \theta_2 \circ [z \mapsto a]$.

Unifiers

Question: when is it impossible to unify two literals?

1. if they start with different predicate symbols or we need to match two different function symbols (obvious)
2. consider the following pair of literals

$$P(a, x), \qquad P(a, f(x))$$

   We can never make them identical using any substitution because the two terms x and $f(x)$ we are trying to unify contain the same variable.

Unifiers

Example

The problem of trying to unify the pair of literal

$$P(x, f(y)) \qquad P(f(f(y)), g(a))$$

can be viewed as solving the system of two term equations:

$$x = f(f(y))$$

$$f(y) = g(a)$$

Based on the previous remark, this cannot be unified because the second equation uses two different function symbols.

Unification Theorem

Theorem

Every unifiable set of literals has an mgu.

The proof is given constructively, i.e., by giving an algorithm and proving its correctness.

Unification Algorithm

- Input: A set U of literals
- Output: An mgu for U, or "fail"
- θ is identity substitution
- repeat the following until θ is a unifier for U:
  1. pick two distinct literals in $U\theta$ and find the first position they differ.
  2. if none of the corresponding symbols is a variable, "fail"
  3. now they are diff on a variable x and a subterm t. If t contains x, report "fail".
  4. $\theta := \theta \circ [x \mapsto t]$
- return θ

Example

$\{P(f(x), g(y)), P(f(f(a)), g(z))\}$

Exercise

Run unification algorithm on the following example:

Example

$U = \{P(g(y), f(x, h(x), y), P(x, f(g(z), w, z))\}$

Exercise

Show that unification algorithm (implemented in a straightforward way) can have exponential running time.

Hint: consider the example below

$$L = \{P(x_1, x_2, \ldots, x_n), P(f(x_0, x_0), f(x_1, x_1), \ldots, f(x_{n-1}, x_{n-1}))\}$$

General Resolution

Clashing Clauses

- C_1 and C_2 are clauses with no common variables
- They clash if there exists non-empty subset $D_i \subseteq C_i$ such that

$$unify(D_i \cup \bar{D}_{1-i}) \neq \text{“fail”}$$

.

- Here $\bar{D}_{1-i}$ negates every literal in the set

Example

Consider the two clauses

$$C_1 = \{P(f(x), g(y)), Q(x, y)\}$$

$$C_2 = \{\neg P(f(f(a)), g(z)), Q(f(a), z)\}$$

Q: How do they clash?
$[x, y \mapsto f(a), z]$ is an mgu of the first formulas.

Resolvent

C_1 and C_2 are two clashing clauses. A resolvent of C_1, C_2 is a clause of the form

$$(C_1\theta \setminus D_1\theta) \cup (C_2\theta \setminus D_2\theta)$$

if they clash on $D_i \subseteq C_i$ and $unify(D_i \cup \bar{D}_{1-i}) = \theta$.

Example

$$C_1 = \{P(f(x), g(y)), Q(x, y)\}$$

and

$$C_2 = \{\neg P(f(f(a)), g(z)), Q(f(a), z)\}$$

Q: Give at least one resolvent of C_1 and C_2

$\{Q(f(a), z)\}$

Remarks on Resolvent

- Most often, the two clauses we are trying to resolve will have common variables, so we cannot compute the resolvent as stated before.
- We need to rename variables so that the clauses no longer have ant common variables.

Example

Consider the two clauses

$$\{P(f(x), y)\} \qquad \{\neg P(x, a)\}$$

The two literals contain common variable x. We rename the x in the second literal to z

$$\{P(f(x), y)\} \qquad \{\neg P(z, a)\}$$

and then the empty resolvent can be computed.

General Resolution Rule

$$\begin{tikzcd}[column sep=small] C_1 \arrow[dr, dash] & & C_2 \arrow[dl, dash] \\ & R & \end{tikzcd}$$

denotes clashing clauses with a resolvent R

- Preprocessing: before any resolution rule, rename the variables so that $VAR(C_1) \cap VAR(C_2) = \emptyset$.
- Q: why this is okay?

Proof and the Goal

- The setting: we are given a set Σ of clauses
- The aim: prove Σ is unsatisfiable
- just like in resolutions for propositional logic, a proof is a sequence of clauses

$$C_1, C_2, \ldots, C_n$$

  such that
- for each $i > 0$, we have either
  1. $C_i \in \Sigma$, or
  2. C_i is a resolvent of two clauses C_a, C_b with $a, b < i$
  3. $C_n = \perp = \{\}$

General Resolution Rule

- Input: A set Σ of clauses
- Output: SAT or UNSAT
- $S := \Sigma$
- While $\perp \notin S$
  1. pick two clashing clauses $C, C' \in \Sigma$.
  2. after a suitable variable renaming, pick a resolvent R
  3. if $\forall C'' \in \Sigma \,.\, R \notin Rename(C'')$. $S := S \cup \{R\}$
  4. If no applications of the above three steps can increase S, return SAT
- return UNSAT

Examples

Example

Prove UNSAT for the following sets of clauses by resolution

$$\Sigma = \left\{ \begin{array}{c} \{\neg P(x), Q(x), R(x, f(x))\}, \\ \{\neg P(x), Q(x), S(f(x))\}, \\ \{T(a)\}, \\ \{P(a)\}, \\ \{\neg R(a, z), T(z)\}, \\ \{\neg T(x), \neg Q(x)\}, \\ \{\neg T(y), \neg S(y)\} \end{array} \right\}$$

Examples

Example

Prove UNSAT for the following sets of clauses by resolution

$$\Sigma = \left\{ \begin{array}{c} \{\neg P(x,y), P(y,x)\}, \\ \{\neg P(x,y), \neg P(y,x), P(x,z)\}, \\ \{\neg P(x,f(x))\}, \\ \{\neg P(x,x)\} \end{array} \right\}$$

Examples

Example

Prove UNSAT for the following formula

$$\exists y \forall x (shaves(y, x) \rightarrow \neg shaves(x, x))$$

Soundness and Completeness

Soundness

- a proof method is **sound** if it never proves a wrong formula:

$$\vdash F \qquad \Rightarrow \qquad \models F$$

$\vdash F$: F is provable

Theorem

The semantic argument is sound.

Soundness and Completeness

Soundness

- a proof method is **sound** if it never proves a wrong formula:

$$\vdash F \qquad \Rightarrow \qquad \models F$$

$\vdash F$: F is provable

Theorem

The semantic argument is sound.

Completeness

- a proof method is **complete** if it can prove every valid formula:

$$\models F \qquad \Rightarrow \qquad \vdash F$$

Theorem

The semantic argument is complete.

Soundness and Completeness

Soundness

- a proof method is **sound** if it never proves a wrong formula:

$$\vdash F \qquad \Rightarrow \qquad \models F$$

$\vdash F$: F is provable

Theorem

The semantic argument is sound.

Completeness

- a proof method is **complete** if it can prove every valid formula:

$$\models F \qquad \Rightarrow \qquad \vdash F$$

Theorem

The semantic argument is complete.

There are also other sound and complete methods for FOL (e.g. natural deduction, Hilbert system).

Soundness and Completeness

Theorem

If resolution generates $\perp$, then the input set of clauses is unsatisfiable.

Theorem

For a given unsatisfiable set of clauses, resolution can generate $\perp$.

First Order Theories

- A theory is a non-empty set T of formulas
- often restricted to some syntactical restriction (e.g., only has certain function symbols)
- it is closed under consequence, i.e., if $F_1, F_2, \ldots, F_n \in T$ and G is a consequence of $F_1, F_2, \ldots, F_n$, then $G \in T$.
- there are two different methods to define a particular theory: the model theoretic method and axiomatic method.

Model Theoretic Method

- define a structure $\mathcal{A}$ first, and then take theory of $\mathcal{A}$ as the set of formulas for which $\mathcal{A}$ is a model.

$$Th(\mathcal{A}) = \{F \mid \mathcal{A} \models F\}$$

- it is clear that $Th(\mathcal{A})$ is closed under consequence.

Example

Theories $Th(\mathbb{N}, +)$ and $Th(\mathbb{N}, +, *)$ are structures taking $\mathbb{N}$ as the universe, the interpretation of $+$ as the usual addition, and the interpretation of $*$ as the usual multiplication. The former is called Presburger arithmetic and the latter Peano arithmetic. The formulas are restricted to consists of the functions symbols $+$ and $*$ only. For example

$$\forall x, y \;.\; ((x + y) * (x + y) = (x * x) + (3 * y))$$

Axiomatic Method

- define a set of formulas M (the axioms) and take the set of all consequences of M as the theory associated with M.
- a theory is called (finitely) axiomatizable if there exists a (finite) axiom set that defines it. For example

$$Cons() = \{F \mid F \text{ is valid}\}$$

The theory of groups:

$$M = \left\{ \begin{array}{c} \forall x, y, z \,.\, (f(f(x,y),z) = f(x, f(y,z))), \\ \forall x \,.\, f(x,e) = x, \\ \forall x . \exists y \,.\, (f(x,y) = e) \end{array} \right\}$$

References and Acknowledgement

The main reference book: Uwe Schöning, Logic for Computer Scientists.

Most of the slides are taken from

- Anthony Lin (TU Kaiserslautern)
- Ondrej Lengal (Brno University of Tech.)