First-Order Logic

Outline

I. Syntax of FOL

II. Quantifiers

III. Model for FOL

* Figures are from the textbook site unless a source is specifically cited.

I. Propositional Logic: Strength and Weakness

- ♠ Programming languages lack a general mechanism for deriving facts from other facts.
- ♠ They lack the expressiveness required to handle partial information.

I. Propositional Logic: Strength and Weakness

- ♠ Programming languages lack a general mechanism for deriving facts from other facts.
- ♠ They lack the expressiveness required to handle partial information.
- ♦ Propositional logic addresses the above issues.

I. Propositional Logic: Strength and Weakness

♠ Programming languages lack a general mechanism for deriving facts from other facts.

♠ They lack the expressiveness required to handle partial information.

♦ Propositional logic addresses the above issues.

♦ It also has *compositionality* – the meaning of a sentence is a function of the meanings of its parts.

$\neg\, \mathtt{rain} \lor \neg\, \mathtt{outside} \lor \mathtt{wet}$

I. Propositional Logic: Strength and Weakness

♠ Programming languages lack a general mechanism for deriving facts from other facts.

♠ They lack the expressiveness required to handle partial information.

♦ Propositional logic addresses the above issues.

♦ It also has *compositionality* – the meaning of a sentence is a function of the meanings of its parts.

$$\neg \mathrm{rain} \lor \neg \mathrm{outside} \lor \mathrm{wet}$$

♠ It lacks the expressive power to describe an environment with many objects.

I. Propositional Logic: Strength and Weakness

♠ Programming languages lack a general mechanism for deriving facts from other facts.

♠ They lack the expressiveness required to handle partial information.

♦ Propositional logic addresses the above issues.

♦ It also has *compositionality* – the meaning of a sentence is a function of the meanings of its parts.

```
¬ rain ∨ ¬ outside ∨ wet
```

♠ It lacks the expressive power to describe an environment with many objects.

$$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

$$B_{1,2} \Leftrightarrow (P_{1,1} \vee P_{1,3} \vee P_{2,2})$$

$$\vdots$$

// Squares adjacent to pits are breezy.

I. Propositional Logic: Strength and Weakness

♠ Programming languages lack a general mechanism for deriving facts from other facts.

♠ They lack the expressiveness required to handle partial information.

♦ Propositional logic addresses the above issues.

♦ It also has *compositionality* – the meaning of a sentence is a function of the meanings of its parts.

$$\neg \text{rain} \lor \neg \text{outside} \lor \text{wet}$$

♠ It lacks the expressive power to describe an environment with many objects.

$$B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$$

$$B_{1,2} \Leftrightarrow (P_{1,1} \lor P_{1,3} \lor P_{2,2})$$

$$\vdots$$

// Squares adjacent to pits are breezy.

♠ Propositional logic assumes the world contains facts only.

Combining Formal and Natural Languages

First-order logic

- built around objects and relations
  - Objects: people, houses, cars, trees, colors, days, …
  - Relations:
    - unary properties such as `big`, `windy`, …
    - n-ary properties such as `bigger than`, `parent of`, `on`, `owns`, …
  - Functions: `square of`, `best friend`, `age`, …
- capable of expressing facts about some or all objects

Formal Languages

Language	Ontological Commitment (What exists in the world)	Epistemological Commitment (What an agent believes about facts)
Propositional logic	facts	true/false/unknown
First-order logic	facts, objects, relations	true/false/unknown
Temporal logic	facts, objects, relations, times	true/false/unknown
Probability theory	facts	degree of belief $\in [0, 1]$
Fuzzy logic	facts with degree of truth $\in [0, 1]$	known interval value

Alphabet of First-Order Logic

- Logical symbols
  - connectives: $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$, $\neg$
  - parenthesis: (,) and punctuation ,
  - equality: $=$
  - quantifiers: $\forall$ (universal quantification), $\exists$ (existential quantification)
  - variables: $x, y, z, \ldots;\ x_1, x_2, \ldots$

Alphabet of First-Order Logic

- Logical symbols
  - connectives: $\wedge, \vee, \Rightarrow, \Leftrightarrow, \neg$
  - parenthesis: (,) and punctuation ,
  - equality: $=$
  - quantifiers: $\forall$ (universal quantification), $\exists$ (existential quantification)
  - variables: $x, y, z, \ldots;\ x_1, x_2, \ldots$
- Non-logical symbols
  - constants: Socrates, Turing, 1, earth, …

Alphabet of First-Order Logic

- Logical symbols
  - connectives: $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$, $\neg$
  - parenthesis: (,) and punctuation ,
  - equality: $=$
  - quantifiers: $\forall$ (universal quantification), $\exists$ (existential quantification)
  - variables: $x, y, z, \ldots$; $x_1, x_2, \ldots$
- Non-logical symbols
  - constants: Socrates, Turing, 1, earth, …
  - predicate symbols: *true*, *false*

Alphabet of First-Order Logic

- Logical symbols
  - connectives: $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$, $\neg$
  - parenthesis: (,) and punctuation ,
  - equality: $=$
  - quantifiers: $\forall$ (universal quantification), $\exists$ (existential quantification)
  - variables: $x, y, z, \ldots$; $x_1, x_2, \ldots$
- Non-logical symbols
  - constants: Socrates, Turing, 1, earth, …
  - predicate symbols: *true*, *false*
    Father(x, y) // x is father of y
    Female(x) // x is female

Alphabet of First-Order Logic

- Logical symbols
  - connectives: $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$, $\neg$
  - parenthesis: (,) and punctuation ,
  - equality: =
  - quantifiers: ∀ (universal quantification), ∃ (existential quantification)
  - variables: $x, y, z, \ldots;\ x_1, x_2, \ldots$
- Non-logical symbols
  - constants: Socrates, Turing, 1, earth, …
  - predicate symbols: *true*, *false*
    - *Father*(x, y) // x is father of y
    - *Female*(x) // x is female
  - function symbols:
    - *gcd*(x, y) // greatest common divisor of x and y
    - *FatherOf*(x) // father of x

Terms and Atomic Sentences

- Terms
  - constants: Socrates, Turing, 1, earth, …

Terms and Atomic Sentences

- Terms
  - constants: Socrates, Turing, 1, earth, …
  - variables: $x, y, z, \ldots;\ x_1, x_2, \ldots$

Terms and Atomic Sentences

- Terms
  - constants: Socrates, Turing, 1, earth, …
  - variables: $x, y, z, \ldots;\ x_1, x_2, \ldots$
  - functions: *gcd*(x, y), *FatherOf*(x), …

Terms and Atomic Sentences

- Terms
  - constants: Socrates, Turing, 1, earth, …
  - variables: $x, y, z, \ldots;\ x_1, x_2, \ldots$
  - functions: *gcd*(x, y), *FatherOf*(x), …

$$\underbrace{f}_{\text{function symbol}}(\underbrace{x_1, x_2, \ldots, x_n}_{\text{terms}})$$

Terms and Atomic Sentences

- Terms
  - constants: Socrates, Turing, 1, earth, …
  - variables: $x, y, z, \ldots$; $x_1, x_2, \ldots$
  - functions: *gcd*(x, y), *FatherOf*(x), …

$$\underbrace{f}_{\text{function symbol}}(\underbrace{x_1, x_2, \ldots, x_n}_{\text{terms}})$$

- Atomic sentences
  - predicates: *true*, *false*

Terms and Atomic Sentences

- Terms
  - constants: Socrates, Turing, 1, earth, …
  - variables: $x, y, z, \ldots;\ x_1, x_2, \ldots$
  - functions: *gcd*(x, y), *FatherOf*(x), …

$$\underbrace{f}_{\text{function symbol}}\underbrace{(x_1, x_2, \ldots, x_n)}_{\text{terms}}$$

- Atomic sentences
  - predicates: *true*, *false*
    Mother(Aphrodite, Harmonia)
    Male(John)

- Greek mythology related examples are taken/extended from ones used in the book *Logic for Problem Solving* by Robert Kowalski, Elsevier Science Publishing Co., Inc., 1979.

Terms and Atomic Sentences

- Terms
  - constants: Socrates, Turing, 1, earth, …
  - variables: $x, y, z, \ldots;\ x_1, x_2, \ldots$
  - functions: *gcd*(x, y), *FatherOf*(x), …

$$\underbrace{f}_{\text{function symbol}}(\underbrace{x_1, x_2, \ldots, x_n}_{\text{terms}})$$

- Atomic sentences
  - predicates: *true*, *false*

    Mother(Aphrodite, Harmonia)

    Male(John)
  - term equalities

    FatherOf(Apollo) = Zeus

- Greek mythology related examples are taken/extended from ones used in the book *Logic for Problem Solving* by Robert Kowalski, Elsevier Science Publishing Co., Inc., 1979.

Complex Sentences

- made of atomic sentences using logical connectives

$$\mathit{Father}(x, y) \Rightarrow \mathit{Male}(x)$$
$$\mathit{Female}(x) \lor \neg \mathit{Mother}(x, y)$$
$$\mathit{Likes}(\text{Mary, John}) \Leftrightarrow \mathit{Likes}(\text{John, Mary})$$
$$(\mathit{Parent}(x, y) \land \mathit{Parent}(y, z)) \Rightarrow \mathit{GrandParent}(x, z)$$

Complex Sentences

- made of atomic sentences using logical connectives

$$Father(x, y) \Rightarrow Male(x)$$

$$Female(x) \vee \neg Mother(x, y)$$

$$Likes(\text{Mary, John}) \Leftrightarrow Likes(\text{John, Mary})$$

$$(Parent(x, y) \wedge Parent(y, z)) \Rightarrow GrandParent(x, z)$$

- universal quantification

$\forall x\ Circle(x) \Rightarrow Ellipse(x)$ // Every circle is an ellipse.

$\neg \forall x\ Likes(x, \text{sushi})$ // Not everyone likes sushi.

$\forall x\ Integer(x) \Rightarrow (Even(x) \vee Odd(x))$ // Every integer is either even or odd.

Complex Sentences

- made of atomic sentences using logical connectives

$$Father(x, y) \Rightarrow Male(x)$$

$$Female(x) \lor \neg Mother(x, y)$$

$$Likes(\text{Mary, John}) \Leftrightarrow Likes(\text{John, Mary})$$

$$(Parent(x, y) \land Parent(y, z)) \Rightarrow GrandParent(x, z)$$

- universal quantification

$\forall x\ Circle(x) \Rightarrow Ellipse(x)$ // Every circle is an ellipse.

$\neg \forall x\ Likes(x, \text{sushi})$ // Not everyone likes sushi.

$\forall x\ Integer(x) \Rightarrow (Even(x) \lor Odd(x))$ // Every integer is either even or odd.

- existential quantification

$\exists x\ Star(x) \land \neg (x = \text{Sun})$ // There are stars other than the sun.

$\exists x\ Whale(x) \land (Age(x) = 200)$ // Some whales live to 200 years.

Syntax of First-Order Logic

$$
\begin{array}{rcl}
\textit{Sentence} & \rightarrow & \textit{AtomicSentence} \mid \textit{ComplexSentence} \\
\textit{AtomicSentence} & \rightarrow & \textit{Predicate} \mid \textit{Predicate}(\textit{Term}, \ldots) \mid \textit{Term} = \textit{Term} \\
\textit{ComplexSentence} & \rightarrow & (\ \textit{Sentence}\ ) \\
& \mid & \neg\ \textit{Sentence} \\
& \mid & \textit{Sentence} \land \textit{Sentence} \\
& \mid & \textit{Sentence} \lor \textit{Sentence} \\
& \mid & \textit{Sentence} \ \Rightarrow \ \textit{Sentence} \\
& \mid & \textit{Sentence} \ \Leftrightarrow \ \textit{Sentence} \\
& \mid & \textit{Quantifier}\ \textit{Variable}, \ldots\ \textit{Sentence} \\
\\
\textit{Term} & \rightarrow & \textit{Function}(\textit{Term}, \ldots) \\
& \mid & \textit{Constant} \\
& \mid & \textit{Variable} \\
\\
\textit{Quantifier} & \rightarrow & \forall \mid \exists \\
\textit{Constant} & \rightarrow & A \mid X_1 \mid \textit{John} \mid \cdots \\
\textit{Variable} & \rightarrow & a \mid x \mid s \mid \cdots \\
\textit{Predicate} & \rightarrow & \textit{True} \mid \textit{False} \mid \textit{After} \mid \textit{Loves} \mid \textit{Raining} \mid \cdots \\
\textit{Function} & \rightarrow & \textit{Mother} \mid \textit{LeftLeg} \mid \cdots \\
\textsc{Operator Precedence} & : & \neg, =, \land, \lor, \Rightarrow, \Leftrightarrow
\end{array}
$$

II. Nested Quantifiers

$\forall x \exists y$ *Student*$(x) \wedge$ *Course*(y) $\wedge$ *Enrolled*(x, y)

$\forall x \exists y$ *Brother*$(x, y) \Rightarrow$ *Sibiling*(x, y)

Order matters for quantifiers of different types:

$\forall x \exists y$ *Loves*(x, y) // Everybody loves somebody

$\exists x \forall y$ *Loves*(y, x) // There is someone whom everyone loves.

$\exists x \exists y$ *Loves*$(x, y) \quad \equiv \quad \exists y \exists x$ *Loves*(x, y)

$\forall x \forall y$ (*Brother*$(x, y) \Rightarrow$ *Sibiling*(x, y))

$\equiv \quad \forall y \forall x$ (*Brother*$(x, y) \Rightarrow$ *Sibiling*(x, y))

Connections Between $\forall$ and $\exists$ Through $\neg$

$$\forall x \;\; \neg Likes(x, Parsnips) \qquad \equiv \qquad \neg \, \exists x \;\; Likes(x, Parsnips)$$

$$\forall x \;\; Likes(x, Icecream) \qquad \equiv \qquad \neg \, \exists x \, \neg \, Likes(x, Icecream)$$

Connections Between ∀ and ∃ Through ¬

$$\forall x \ \neg Likes(x, Parsnips) \equiv \neg \exists x \ Likes(x, Parsnips)$$

$$\forall x \ Likes(x, Icecream) \equiv \neg \exists x \neg Likes(x, Icecream)$$

De Morgan's rules still apply:

$$\neg \forall x \ P(x) \equiv \exists x \ \neg P(x)$$

$$\neg \exists x \ P(x) \equiv \forall x \ \neg P(x)$$

Connections Between ∀ and ∃ Through ¬

$\forall x \ \neg \mathit{Likes}(x, \mathit{Parsnips}) \quad \equiv \quad \neg \exists x \ \mathit{Likes}(x, \mathit{Parsnips})$

$\forall x \ \mathit{Likes}(x, \mathit{Icecream}) \quad \equiv \quad \neg \exists x \neg \mathit{Likes}(x, \mathit{Icecream})$

De Morgan's rules still apply:

$$\neg \forall x \, P(x) \quad \equiv \quad \exists x \, \neg P(x)$$

$$\neg \exists x \, P(x) \quad \equiv \quad \forall x \, \neg P(x)$$

Move negation inward, flipping the quantifiers:

$\neg \forall x \exists y \exists z \forall s \forall t \ \ P(x, y, z, s, t)$

Connections Between ∀ and ∃ Through ¬

$\forall x \ \neg \mathit{Likes}(x, \mathit{Parsnips}) \quad \equiv \quad \neg \exists x \ \mathit{Likes}(x, \mathit{Parsnips})$

$\forall x \ \mathit{Likes}(x, \mathit{Icecream}) \quad \equiv \quad \neg \exists x \neg \mathit{Likes}(x, \mathit{Icecream})$

De Morgan's rules still apply:

$$\neg\forall x\, P(x) \quad \equiv \quad \exists x\, \neg P(x)$$

$$\neg\exists x\, P(x) \quad \equiv \quad \forall x\, \neg P(x)$$

Move negation inward, flipping the quantifiers:

$$\neg\forall x \exists y \exists z \forall s \forall t \ \ P(x,y,z,s,t) \quad \equiv \quad \exists x \neg\exists y \exists z \forall s \forall t \ \ P(x,y,z,s,t)$$

Connections Between ∀ and ∃ Through ¬

$$\forall x \ \neg Likes(x, Parsnips) \equiv \neg \exists x \ Likes(x, Parsnips)$$

$$\forall x \ Likes(x, Icecream) \equiv \neg \exists x \neg Likes(x, Icecream)$$

De Morgan's rules still apply:

$$\neg \forall x \, P(x) \equiv \exists x \, \neg P(x)$$

$$\neg \exists x \, P(x) \equiv \forall x \, \neg P(x)$$

Move negation inward, flipping the quantifiers:

$$\begin{aligned} \neg \forall x \exists y \exists z \forall s \forall t \ P(x,y,z,s,t) &\equiv \exists x \neg \exists y \exists z \forall s \forall t \ P(x,y,z,s,t) \\ &\equiv \exists x \forall y \neg \exists z \forall s \forall t \ P(x,y,z,s,t) \end{aligned}$$

Connections Between $\forall$ and $\exists$ Through $\neg$

$$\forall x \ \neg Likes(x, Parsnips) \quad \equiv \quad \neg \exists x \ Likes(x, Parsnips)$$

$$\forall x \ Likes(x, Icecream) \quad \equiv \quad \neg \exists x \neg Likes(x, Icecream)$$

De Morgan's rules still apply:

$$\neg \forall x \, P(x) \quad \equiv \quad \exists x \, \neg P(x)$$

$$\neg \exists x \, P(x) \quad \equiv \quad \forall x \, \neg P(x)$$

Move negation inward, flipping the quantifiers:

$$\begin{aligned}
\neg \forall x \exists y \exists z \forall s \forall t \ \ P(x, y, z, s, t) \quad & \equiv \quad \exists x \neg \exists y \exists z \forall s \forall t \ \ P(x, y, z, s, t) \\
& \equiv \quad \exists x \forall y \neg \exists z \forall s \forall t \ \ P(x, y, z, s, t) \\
& \equiv \quad \exists x \forall y \forall z \neg \forall s \forall t \ \ P(x, y, z, s, t)
\end{aligned}$$

Connections Between $\forall$ and $\exists$ Through $\neg$

$$\forall x \ \neg Likes(x, Parsnips) \quad \equiv \quad \neg \exists x \ Likes(x, Parsnips)$$

$$\forall x \ Likes(x, Icecream) \quad \equiv \quad \neg \exists x \neg Likes(x, Icecream)$$

De Morgan's rules still apply:

$$\neg \forall x \, P(x) \quad \equiv \quad \exists x \, \neg P(x)$$

$$\neg \exists x \, P(x) \quad \equiv \quad \forall x \, \neg P(x)$$

Move negation inward, flipping the quantifiers:

$$\begin{aligned}
\neg \forall x \exists y \exists z \forall s \forall t \ P(x,y,z,s,t) \quad &\equiv \quad \exists x \neg \exists y \exists z \forall s \forall t \ P(x,y,z,s,t) \\
&\equiv \quad \exists x \forall y \neg \exists z \forall s \forall t \ P(x,y,z,s,t) \\
&\equiv \quad \exists x \forall y \forall z \neg \forall s \forall t \ P(x,y,z,s,t) \\
&\equiv \quad \exists x \forall y \forall z \exists s \neg \forall t \ P(x,y,z,s,t)
\end{aligned}$$

Connections Between ∀ and ∃ Through ¬

$\forall x \; \neg Likes(x, Parsnips) \quad \equiv \quad \neg \exists x \; Likes(x, Parsnips)$

$\forall x \; Likes(x, Icecream) \quad \equiv \quad \neg \exists x \neg Likes(x, Icecream)$

De Morgan's rules still apply:

$$\neg \forall x \, P(x) \equiv \exists x \, \neg P(x)$$

$$\neg \exists x \, P(x) \equiv \forall x \, \neg P(x)$$

Move negation inward, flipping the quantifiers:

$$\begin{aligned}
\neg \forall x \exists y \exists z \forall s \forall t \;\; P(x,y,z,s,t) &\equiv \exists x \neg \exists y \exists z \forall s \forall t \;\; P(x,y,z,s,t) \\
&\equiv \exists x \forall y \neg \exists z \forall s \forall t \;\; P(x,y,z,s,t) \\
&\equiv \exists x \forall y \forall z \neg \forall s \forall t \;\; P(x,y,z,s,t) \\
&\equiv \exists x \forall y \forall z \exists s \neg \forall t \;\; P(x,y,z,s,t) \\
&\equiv \exists x \forall y \forall z \exists s \exists t \;\; \neg P(x,y,z,s,t)
\end{aligned}$$

Free and Bound Variables

A variable occurrence is *bound* in a formula if it is quantified.

A variable occurrence is *free* in a formula if it is not quantified.

Free and Bound Variables

A variable occurrence is *bound* in a formula if it is quantified.

A variable occurrence is *free* in a formula if it is not quantified.

$\forall x\ \mathit{Father}(x, y) \Rightarrow \mathit{Male}(x)$

x is bound while y is free.

Free and Bound Variables

A variable occurrence is *bound* in a formula if it is quantified.

A variable occurrence is *free* in a formula if it is not quantified.

$\forall x \; \mathit{Father}(x, y) \Rightarrow \mathit{Male}(x)$ — x is bound while y is free.

$\neg\forall x \exists y \exists z \forall s \forall t \;\; P(x, y, z, s, t)$ — x, y, z, s, t are all bound

Free and Bound Variables

A variable occurrence is *bound* in a formula if it is quantified.

A variable occurrence is *free* in a formula if it is not quantified.

$\forall x\ Father(x, y) \Rightarrow Male(x)$	x is bound while y is free.
$\neg\forall x \exists y \exists z \forall s \forall t\ \ P(x, y, z, s, t)$	x, y, z, s, t are all bound
$\forall x\ \forall y\ (P(x)\ \Rightarrow\ Q(x, f(y), z))$	x, y are bound while z is free.

Free and Bound Variables

A variable occurrence is *bound* in a formula if it is quantified.

A variable occurrence is *free* in a formula if it is not quantified.

$\forall x\ Father(x, y) \Rightarrow Male(x)$	x is bound while y is free.
$\neg\forall x \exists y \exists z \forall s \forall t\ \ P(x, y, z, s, t)$	x, y, z, s, t are all bound
$\forall x\ \forall y\ (P(x)\ \Rightarrow\ Q(x, f(y), z))$	x, y are bound while z is free.

Free and bound variables can have the same name.

$$P(x)\ \Rightarrow \exists x\ Q(x)$$

Free and Bound Variables

A variable occurrence is *bound* in a formula if it is quantified.

A variable occurrence is *free* in a formula if it is not quantified.

$\forall x\ \mathit{Father}(x, y) \Rightarrow \mathit{Male}(x)$ — x is bound while y is free.

$\neg\forall x \exists y \exists z \forall s \forall t\ \ P(x, y, z, s, t)$ — x, y, z, s, t are all bound

$\forall x\ \forall y\ (P(x) \ \Rightarrow\ Q(x, f(y), z))$ — x, y are bound while z is free.

Free and bound variables can have the same name.

$$P(\underbrace{x}_{\text{free}})\ \Rightarrow \exists x\ Q(x)$$

Free and Bound Variables

A variable occurrence is *bound* in a formula if it is quantified.

A variable occurrence is *free* in a formula if it is not quantified.

$\forall x \; Father(x, y) \Rightarrow Male(x)$ — x is bound while y is free.

$\neg \forall x \exists y \exists z \forall s \forall t \;\; P(x, y, z, s, t)$ — x, y, z, s, t are all bound

$\forall x \; \forall y \; (P(x) \;\Rightarrow\; Q(x, f(y), z))$ — x, y are bound while z is free.

Free and bound variables can have the same name.

$$P(\underbrace{x}_{\text{free}}) \;\Rightarrow\; \exists x \; Q(\underbrace{x}_{\text{bound}})$$

Free and Bound Variables

A variable occurrence is *bound* in a formula if it is quantified.

A variable occurrence is *free* in a formula if it is not quantified.

$\forall x \; Father(x, y) \Rightarrow Male(x)$ — x is bound while y is free.

$\neg \forall x \exists y \exists z \forall s \forall t \;\; P(x, y, z, s, t)$ — x, y, z, s, t are all bound

$\forall x \; \forall y \; (P(x) \;\Rightarrow\; Q(x, f(y), z))$ — x, y are bound while z is free.

Free and bound variables can have the same name.

$P(\underbrace{x}_{\text{free}}) \;\Rightarrow \exists x \; Q(\underbrace{x}_{\text{bound}})$

$P(x) \;\Rightarrow (\exists x \; Q(x)) \wedge R(x)$

Free and Bound Variables

A variable occurrence is *bound* in a formula if it is quantified.

A variable occurrence is *free* in a formula if it is not quantified.

$\forall x\ \mathit{Father}(x, y) \Rightarrow \mathit{Male}(x)$	x is bound while y is free.
$\neg \forall x \exists y \exists z \forall s \forall t\ \ P(x, y, z, s, t)$	x, y, z, s, t are all bound
$\forall x\ \forall y\ (P(x) \ \Rightarrow\ Q(x, f(y), z))$	x, y are bound while z is free.

Free and bound variables can have the same name.

$P(x) \ \Rightarrow \exists x\ Q(x)$

free bound

$P(x) \ \Rightarrow (\exists x\ Q(x)) \wedge R(x)$

same free variable

Free and Bound Variables

A variable occurrence is *bound* in a formula if it is quantified.

A variable occurrence is *free* in a formula if it is not quantified.

$\forall x\ \mathit{Father}(x, y) \Rightarrow \mathit{Male}(x)$ — x is bound while y is free.

$\neg\forall x \exists y \exists z \forall s \forall t\ \ P(x, y, z, s, t)$ — x, y, z, s, t are all bound

$\forall x\ \forall y\ (P(x)\ \Rightarrow\ Q(x, f(y), z))$ — x, y are bound while z is free.

Free and bound variables can have the same name.

$$\underbrace{P(x)}_{\text{free}}\ \Rightarrow \exists x\ Q(\underbrace{x}_{\text{bound}})$$

different bound variable

$P(x)\ \Rightarrow (\exists x\ Q(x)) \wedge R(x)$

same free variable

Equality

- It states that two terms refer to the same object.

$$\mathit{Father}(\text{Zeus}) = \text{Cronus}$$

$$\mathit{Father}(\text{Cronus}) = \text{Uranus}$$

Equality

- It states that two terms refer to the same object.

$$Father(\text{Zeus}) = \text{Cronus}$$

$$Father(\text{Cronus}) = \text{Uranus}$$

- The symbol can also be used to state that two terms are not the same object.

Equality

- It states that two terms refer to the same object.

$$\mathit{Father}(\text{Zeus}) = \text{Cronus}$$

$$\mathit{Father}(\text{Cronus}) = \text{Uranus}$$

- The symbol can also be used to state that two terms are not the same object.

// Zeus has exactly two brothers

Equality

- It states that two terms refer to the same object.

$$Father(\text{Zeus}) = \text{Cronus}$$

$$Father(\text{Cronus}) = \text{Uranus}$$

- The symbol can also be used to state that two terms are not the same object.

// Zeus has exactly two brothers

$$\exists x, y \; Brother(x, \text{Zeus}) \land Brother(y, \text{Zeus}) \land \neg(x = y)$$
$$\land\ (\forall z \; Brother(z, \text{Zeus}) \Rightarrow (z = x) \lor (z = y))$$

Equality

- It states that two terms refer to the same object.

$$Father(\text{Zeus}) = \text{Cronus}$$

$$Father(\text{Cronus}) = \text{Uranus}$$

- The symbol can also be used to state that two terms are not the same object.

// Zeus has exactly two brothers
// ($x \equiv$ Poseidon and $y \equiv$ Hades, or $x \equiv$ Hades and $y \equiv$ Poseidon)

$$\exists x, y\ Brother(x, \text{Zeus}) \land Brother(y, \text{Zeus}) \land \neg(x = y)$$
$$\land\ (\forall z\ Brother(z, \text{Zeus}) \Rightarrow (z = x) \lor (z = y))$$

III. Model for First-Order Logic

- Sentences are true with respect to a model M.

III. Model for First-Order Logic

- Sentences are true with respect to a model M.
- The model M contains objects (called *domain elements*) and interpretations of symbols.

III. Model for First-Order Logic

- Sentences are true with respect to a model M.
- The model M contains objects (called *domain elements*) and interpretations of symbols.
  - constant symbols $\rightarrow$ objects in domain D

III. Model for First-Order Logic

- Sentences are true with respect to a model M.

- The model M contains objects (called *domain elements*) and interpretations of symbols.

  - constant symbols $\rightarrow$ objects in domain D
  - predicate symbols $\rightarrow$ relations

III. Model for First-Order Logic

- Sentences are true with respect to a model M.

- The model M contains objects (called *domain elements*) and interpretations of symbols.

  - constant symbols $\rightarrow$ objects in domain D
  - predicate symbols $\rightarrow$ relations
  - function symbols $\rightarrow$ functional relations

III. Model for First-Order Logic

- Sentences are true with respect to a model M.
- The model M contains objects (called *domain elements*) and interpretations of symbols.
    - constant symbols → objects in domain D
    - predicate symbols → relations
    - function symbols → functional relations
- Each predicate $P(x_1, \ldots, x_k)$ is mapped to a relation, which is a set of k-tuples over D.

III. Model for First-Order Logic

- Sentences are true with respect to a model M.
- The model M contains objects (called *domain elements*) and interpretations of symbols.
  - constant symbols → objects in domain D
  - predicate symbols → relations
  - function symbols → functional relations
- Each predicate $P(x_1, \ldots, x_k)$ is mapped to a relation, which is a set of k-tuples over D.
- Each function $f(x_1, \ldots, x_k)$ is mapped to a function from D^k to $D \cup \{o\}$, where o is some invisible object.

Model Example

Model for the family relationships of the Greek gods (incomplete).

Zeus, Ares, Hades, Aphrodite, Hera, Harmonia, Demeter, Dionysus, Apollo, Poseidon, Hermes, Persephone, Athena, Artemis, Hephaestos

domain D

Father(Zeus, Hermes)
Mother(Hera, Ares)
Mother(Aphrodite, Harmonia)
Father(Zeus, Athena)

⋮

predicates

Weapon(Zeus) // ≡ Thunderbolt
Weapon(Apollo) // ≡ BowAndArrows
Carry(Hermes) // ≡ Flute
Carry(Aphrodite) // ≡ Apple

⋮

functions

Truth in First-Order Logic

- A predicate $P(t_1, \ldots, t_k)$ is true if the objects referred to by the terms $t_1, \ldots, t_k$ are in the relation referred to by the predicate.

Truth in First-Order Logic

- A predicate $P(t_1, \ldots, t_k)$ is true if the objects referred to by the terms $t_1, \ldots, t_k$ are in the relation referred to by the predicate.

- $t_1 = t_2$ is true if the two terms t_1 and t_2 refer to the same object.

Truth in First-Order Logic

- A predicate $P(t_1, \dots, t_k)$ is true if the objects referred to by the terms $t_1, \dots, t_k$ are in the relation referred to by the predicate.

- $t_1 = t_2$ is true if the two terms t_1 and t_2 refer to the same object.

- The semantics of sentences formed with logical connectives are identical to those in propositional logic.

Truth in First-Order Logic

- A predicate $P(t_1, \dots, t_k)$ is true if the objects referred to by the terms $t_1, \dots, t_k$ are in the relation referred to by the predicate.

- $t_1 = t_2$ is true if the two terms t_1 and t_2 refer to the same object.

- The semantics of sentences formed with logical connectives are identical to those in propositional logic.

Quantifiers allow us to express properties of a collection of objects instead of enumerating them by name.

∀ (universal): "for all"

∃ (existential): "there exists"

Truths with Quantifications

- $\forall x\ P(x)$ is true in a model M iff $P(x)$ is true with x assuming every object in the model

$\forall x$ *Father*$(x, y) \Rightarrow$ *Male*(x) *true* (in every model)

$\forall x$ *Ellipse*$(x) \Rightarrow$ *Circle*(x) *true* (in every model)

Truths with Quantifications

- $\forall x\ P(x)$ is true in a model M iff $P(x)$ is true with x assuming every object in the model

$\forall x$ *Father*$(x, y) \Rightarrow$ *Male*(x) — *true* (in every model)

$\forall x$ *Ellipse*$(x) \Rightarrow$ *Circle*(x) — *true* (in every model)

- $\exists x\ P(x)$ is true in a model M iff $P(x)$ is true with x assuming some object in the model.

$\exists x \neg$ *Likes*(x, sushi) — *true* (in a model that includes all the people in the world)

Truths with Quantifications

- $\forall x\ P(x)$ is true in a model M iff $P(x)$ is true with x assuming every object in the model

$\forall x$ *Father*$(x, y) \Rightarrow$ *Male*(x) — *true* (in every model)

$\forall x$ *Ellipse*$(x) \Rightarrow$ *Circle*(x) — *true* (in every model)

- $\exists x\ P(x)$ is true in a model M iff $P(x)$ is true with x assuming some object in the model.

$\exists x \neg$ *Likes*$(x,$ sushi$)$ — *true* (in a model that includes all the people in the world)

$\exists x$ *Mother*$(x$, Ares$) \wedge$ *Mother*$(x$, Harmonia$)$ — *false* (in the model of Greek mythology)

Truths with Quantifications

- $\forall x\ P(x)$ is true in a model M iff $P(x)$ is true with x assuming every object in the model

$\forall x\ \textit{Father}(x, y) \Rightarrow \textit{Male}(x)$ — *true* (in every model)

$\forall x\ \textit{Ellipse}(x) \Rightarrow \textit{Circle}(x)$ — *true* (in every model)

- $\exists x\ P(x)$ is true in a model M iff $P(x)$ is true with x assuming some object in the model.

$\exists x \neg \textit{Likes}(x, \text{sushi})$ — *true* (in a model that includes all the people in the world)

$\exists x\ \textit{Mother}(x, \text{Ares}) \land \textit{Mother}(x, \text{Harmonia})$ — *false* (in the model of Greek mythology)