# First-order logic
# Second part:
# Interpretation of a formula

Frédéric Prost

Université Grenoble Alpes

March 2023

# A few examples

Formalize in first-order logic:

- Some people love each other.
- If two people are in love, then they are spouses.
- No one can love two distinct persons.

# A few examples

Formalize in first-order logic:

- Some people love each other.

  $\exists x \, \exists y \, (\ell(x,y) \wedge \ell(y,x))$

- If two people are in love, then they are spouses.

- No one can love two distinct persons.

# A few examples

Formalize in first-order logic:

- Some people love each other.

  $\exists x\ \exists y\ (\ell(x,y) \wedge \ell(y,x))$

- If two people are in love, then they are spouses.

  $\forall x\ \forall y\ (\ell(x,y) \wedge \ell(y,x) \Rightarrow s(x) = y \wedge s(y) = x)$

- No one can love two distinct persons.

# A few examples

Formalize in first-order logic:

- Some people love each other.

  $\exists x\, \exists y\ (\ell(x,y) \wedge \ell(y,x))$

- If two people are in love, then they are spouses.

  $\forall x\, \forall y\ (\ell(x,y) \wedge \ell(y,x) \Rightarrow s(x) = y \wedge s(y) = x)$

- No one can love two distinct persons.

  $\forall x\, \forall y\ (\ell(x,y) \Rightarrow \forall z\ (\ell(x,z) \Rightarrow y = z))$

# A few examples

Formalize in first-order logic:

- Some people love each other.

  $\exists x\ \exists y\ (\ell(x, y) \wedge \ell(y, x))$

- If two people are in love, then they are spouses.

  $\forall x\ \forall y\ (\ell(x, y) \wedge \ell(y, x) \Rightarrow s(x) = y \wedge s(y) = x)$

- No one can love two distinct persons.

  $\forall x\ \forall y\ (\ell(x, y) \Rightarrow \forall z\ (\ell(x, z) \Rightarrow y = z))$

  $\forall x\ \forall y\ \forall z\ (\ell(x, y) \wedge \ell(x, z) \Rightarrow y = z)$

# Overview

# Overview

# Reminder: Interpretation and state

## Definition 4.3.16

An interpretation $I$ over a signature $\Sigma$ is defined by:

- a non-empty domain $D$
- every symbol $s^{gn}$ is mapped to its value as follows:

| | |
|---|---|
| (constant) | $s_I^{f0}$ is an element of $D$ |
| (function) | $s_I^{fn}$ is a function from $D^n \rightarrow D$ |
| (propositional variable) | $s_I^{r0}$ is either 0 or 1 |
| (relation) | $s_I^{rn}$ is a set of $n$-uples in $D$ |

# Reminder: Interpretation and state

## Definition 4.3.16

An interpretation $I$ over a signature $\Sigma$ is defined by:

- a non-empty domain $D$
- every symbol $s^{gn}$ is mapped to its value as follows:

| | |
|---|---|
| (constant) | $s_I^{f0}$ is an element of $D$ |
| (function) | $s_I^{fn}$ is a function from $D^n \rightarrow D$ |
| (propositional variable) | $s_I^{r0}$ is either 0 or 1 |
| (relation) | $s_I^{rn}$ is a set of $n$-uples in $D$ |

## Definition 4.3.21

A state $e$ maps each variable to an element in the domain $D$.

# Remark 4.3.24

- For a formula with free variables, we need an assignment $(I, e)$ with a state $e$.
- For a formula with no free variables, simply give an interpretation $I$ of the symbols of the formula.

  Indeed, $(I, e)$ and $(I, e')$ will yield the same value for any formula: thus, we will identify $(I, e)$ and $I$.

# Terms

**Definition 4.3.25 Evaluation**

The evaluation of a term $t$ is inductively defined as:

1. if $t$ is a variable, then $[\![t]\!]_{(I,e)} = e(t)$,
2. if $t$ is a constant, then $[\![t]\!]_{(I,e)} = t_I^{f0}$,
3. if $t = s(t_1, \ldots, t_n)$ where $s$ is a function symbol, then $[\![t]\!]_{(I,e)} = s_I^{fn}([\![t_1]\!]_{(I,e)}, \ldots, [\![t_n]\!]_{(I,e)})$

# Example 4.3.26

Let the signature be $a^{f0}, f^{f2}, g^{f2}$.
Let $I$ be the interpretation of domain $\mathbb{N}$ which maps:

- $a$ to the integer 1;
- $f$ to the product;
- $g$ to the sum.

Let $e$ be the state such that $e(x) = 2$ and $e(y) = 3$.
Let us compute $[\![f(x, g(y, a))]\!]_{(I,e)}$.

# Example 4.3.26

Let the signature be $a^{f0}, f^{f2}, g^{f2}$.
Let $I$ be the interpretation of domain $\mathbb{N}$ which maps:

- $a$ to the integer 1;
- $f$ to the product;
- $g$ to the sum.

Let $e$ be the state such that $e(x) = 2$ and $e(y) = 3$.
Let us compute $[\![f(x, g(y, a))]\!]_{(I,e)}$.

$$\begin{array}{rcl} [\![f(x,g(y,a))]\!]_{(I,e)} & = & [\![x]\!]_{(I,e)} * [\![g(y,a)]\!]_{(I,e)} \\ & = & [\![x]\!]_{(I,e)} * ([\![y]\!]_{(I,e)} + [\![a]\!]_{(I,e)}) \\ & = & e(x) * (e(y) + 1) \\ & = & 2 * (3 + 1) = 8 \end{array}$$

# Formulae

**Definition 4.3.27** Truth value of an atomic formula

The truth value of an atomic formula is given by the following inductive rules:

1. $[\top]_{(I,e)} = 1$ and $[\bot]_{(I,e)} = 0$.
2. Let $s$ be a propositional variable, $[s]_{(I,e)} = s_I^{r0}$
3. Let $A = s(t_1, \ldots, t_n)$ where $s$ is a relation symbol:
   - if $(\llbracket t_1 \rrbracket_{(I,e)}, \ldots, \llbracket t_n \rrbracket_{(I,e)}) \in s_I^{rn}$ then $[A]_{(I,e)} = 1$
   - otherwise $[A]_{(I,e)} = 0$

# Example 4.3.19

Let us consider the following signature:

- $Anne^{f0}$, $Bernard^{f0}$ and $Claude^{f0}$: constants
- $\ell^{r2}$: a binary relation (we read $\ell(x, y)$ as "$x$ loves $y$")
- $s^{f1}$: a unary function (we read $s(x)$ as the spouse of $x$).

# Example 4.3.19

Let us consider the following signature:

- $Anne^{f0}$, $Bernard^{f0}$ and $Claude^{f0}$: constants
- $\ell^{r2}$: a binary relation (we read $\ell(x, y)$ as "$x$ loves $y$")
- $s^{f1}$: a unary function (we read $s(x)$ as the spouse of $x$).

A possible interpretation over this signature is the interpretation $I$ of domain $D = \{0, 1, 2\}$ where:

# Example 4.3.19

Let us consider the following signature:

- $Anne^{f0}$, $Bernard^{f0}$ and $Claude^{f0}$: constants
- $\ell^{r2}$: a binary relation (we read $\ell(x, y)$ as "$x$ loves $y$")
- $s^{f1}$: a unary function (we read $s(x)$ as the spouse of $x$).

A possible interpretation over this signature is the interpretation $I$ of domain $D = \{0, 1, 2\}$ where:

- $Anne_I^{f0} = 0$, $Bernard_I^{f0} = 1$, and $Claude_I^{f0} = 2$.

# Example 4.3.19

Let us consider the following signature:

- $Anne^{f0}$, $Bernard^{f0}$ and $Claude^{f0}$: constants
- $\ell^{r2}$: a binary relation (we read $\ell(x, y)$ as "$x$ loves $y$")
- $s^{f1}$: a unary function (we read $s(x)$ as the spouse of $x$).

A possible interpretation over this signature is the interpretation $I$ of domain $D = \{0, 1, 2\}$ where:

- $Anne_I^{f0} = 0$, $Bernard_I^{f0} = 1$, and $Claude_I^{f0} = 2$.
- $\ell_I^{r2} = \{(0, 1), (1, 0), (2, 0)\}$.

# Example 4.3.19

Let us consider the following signature:

- $Anne^{f0}$, $Bernard^{f0}$ and $Claude^{f0}$: constants
- $\ell^{r2}$: a binary relation (we read $\ell(x, y)$ as "$x$ loves $y$")
- $s^{f1}$: a unary function (we read $s(x)$ as the spouse of $x$).

A possible interpretation over this signature is the interpretation $I$ of domain $D = \{0, 1, 2\}$ where:

- $Anne_I^{f0} = 0$, $Bernard_I^{f0} = 1$, and $Claude_I^{f0} = 2$.
- $\ell_I^{r2} = \{(0, 1), (1, 0), (2, 0)\}$.
- $s_I^{f1}$ is a fonction from $D$ to $D$ defined as

| $x$ | 0 | 1 | 2 |
|---|---|---|---|
| $s_I^{f1}(x)$ | 1 | 0 | 2 |

# Example 4.3.19

Let us consider the following signature:

- $Anne^{f0}$, $Bernard^{f0}$ and $Claude^{f0}$: constants
- $\ell^{r2}$: a binary relation (we read $\ell(x, y)$ as "$x$ loves $y$")
- $s^{f1}$: a unary function (we read $s(x)$ as the spouse of $x$).

A possible interpretation over this signature is the interpretation $I$ of domain $D = \{0, 1, 2\}$ where:

- $Anne_I^{f0} = 0$, $Bernard_I^{f0} = 1$, and $Claude_I^{f0} = 2$.
- $\ell_I^{r2} = \{(0, 1), (1, 0), (2, 0)\}$.
- $s_I^{f1}$ is a fonction from $D$ to $D$ defined as

| $x$ | 0 | 1 | 2 |
|---|---|---|---|
| $s_I^{f1}(x)$ | 1 | 0 | 2 |

# Example 4.3.29

We obtain:

- $[\ell(Anne, Bernard)]_I =$

# Example 4.3.29

We obtain:

- $[\ell(Anne, Bernard)]_I =$

  $true$ since $([\![Anne]\!]_I, [\![Bernard]\!]_I) = (0, 1) \in \ell_I^{r2}$.

- $[\ell(Anne, Claude)]_I =$

# Example 4.3.29

We obtain:

- $[\ell(Anne, Bernard)]_I =$

  *true* since $([\![Anne]\!]_I, [\![Bernard]\!]_I) = (0, 1) \in \ell_I^{r2}$.

- $[\ell(Anne, Claude)]_I =$

  *false* since $([\![Anne]\!]_I, [\![Claude]\!]_I) = (0, 2) \notin \ell_I^{r2}$.

# Example 4.3.29

Let $e$ be the state $x = 0, y = 2$. We have:

- $[\ell(x, s(x))]_{(I,e)} =$

# Example 4.3.29

Let $e$ be the state $x = 0, y = 2$. We have:

- $[\ell(x, s(x))]_{(I,e)} =$

  *true* since $([\![x]\!]_{(I,e)}, [\![s(x)]\!]_{(I,e)}) = (0, s_I^{f1}(0)) = (0, 1) \in \ell_I^{r2}$.

- $[\ell(y, s(y))]_{(I,e)} =$

# Example 4.3.29

Let $e$ be the state $x = 0, y = 2$. We have:

- $[\ell(x, s(x))]_{(I,e)} =$

  *true* since $([\![x]\!]_{(I,e)}, [\![s(x)]\!]_{(I,e)}) = (0, s_I^{f1}(0)) = (0, 1) \in \ell_I^{r2}$.

- $[\ell(y, s(y))]_{(I,e)} =$

  *false* since $([\![y]\!]_{(I,e)}, [\![s(y)]\!]_{(I,e)}) = (2, s_I^{f1}(2)) = (2, 2) \notin \ell_I^{r2}$.

Here, we have used *true* and *false* instead of the truth values 0 and 1 in order to distinguish them from the elements 0 and 1 of the domain (beware of the ambiguity, use the context).

# Example 4.3.29

We have:

- $[(Anne = Bernard)]_I =$

# Example 4.3.29

We have:

- $[(Anne = Bernard)]_I =$

  *false*, since $(\llbracket Anne \rrbracket_I, \llbracket Bernard \rrbracket_I) = (0, 1)$ and $(0,1) \notin =_I^{r2}$.

- $[(s(Anne) = Anne)]_I =$

# Example 4.3.29

We have:

- $[(Anne = Bernard)]_I =$

  $\boxed{\textit{false}\text{, since } (\llbracket Anne \rrbracket_I, \llbracket Bernard \rrbracket_I) = (0,1) \text{ and } (0,1) \notin =_I^{r2}.}$

- $[(s(Anne) = Anne)]_I =$

  $\boxed{\textit{false}\text{, since } (\llbracket s(Anne) \rrbracket_I, \llbracket Anne \rrbracket_I) = (s_I^{f1}(0), 0) = (1,0).}$

- $[(s(s(Anne)) = Anne)]_I =$

# Example 4.3.29

We have:

- $[(Anne = Bernard)]_I =$

  $false$, since $(\llbracket Anne \rrbracket_I, \llbracket Bernard \rrbracket_I) = (0, 1)$ and $(0, 1) \notin =_I^{r2}$.

- $[(s(Anne) = Anne)]_I =$

  $false$, since $(\llbracket s(Anne) \rrbracket_I, \llbracket Anne \rrbracket_I) = (s_I^{f1}(0), 0) = (1, 0)$.

- $[(s(s(Anne)) = Anne)]_I =$

  $true$, since $(\llbracket s(s(Anne)) \rrbracket_I, \llbracket Anne \rrbracket_I) = (s_I^{f1}(s_I^{f1}(0)), 0) = (0, 0)$ and $(0, 0) \in =_I^{r2}$.

# Truth value of a formula 4.3.30

1. Propositional connectives have the same meaning as in propositional logic.

# Truth value of a formula 4.3.30

1. Propositional connectives have the same meaning as in propositional logic.
2. Let $e[x = d]$ be the state that is identical to $e$, except for $x$.

$$[\forall xB]_{(I,e)} = min_{d \in D}[B]_{(I,e[x=d])} = \prod_{d \in D}[B]_{(I,e[x=d])},$$

*i.e.* it is true if and only if $[B]_{(I,f)} = 1$ for every state $f$ identical to $e$, except for $x$.

# Truth value of a formula 4.3.30

1. Propositional connectives have the same meaning as in propositional logic.
2. Let $e[x = d]$ be the state that is identical to $e$, except for $x$.

$$[\forall xB]_{(I,e)} = min_{d \in D}[B]_{(I,e[x=d])} = \prod_{d \in D}[B]_{(I,e[x=d])},$$

   *i.e.* it is true if and only if $[B]_{(I,f)} = 1$ for every state $f$ identical to $e$, except for $x$.
3. 

$$[\exists xB]_{(I,e)} = max_{d \in D}[B]_{(I,e[x=d])} = \sum_{d \in D}[B]_{(I,e[x=d])},$$

   *i.e.* it is true if there is a state $f$ identical to $e$, except for $x$, such that $[B]_{(I,f)} = 1$.

# Example 4.3.32

Let us use the interpretation $I$ given in example 4.3.19.
(Reminder $D = \{0, 1, 2\}$)

- $[\exists x\ \ell(x, x)]_I =$

# Example 4.3.32

Let us use the interpretation $I$ given in example 4.3.19.
(Reminder $D = \{0, 1, 2\}$)

- $[\exists x\ \ell(x, x)]_I =$

$$= \max\{[\ell(0,0)]_I, [\ell(1,1)]_I, [\ell(2,2)]_I\} = \mathit{false}$$
$$= [\ell(0,0)]_I + [\ell(1,1)]_I + [\ell(2,2)]_I = \mathit{false} + \mathit{false} + \mathit{false} = \mathit{false}.$$

- $[\forall x \exists y\ \ell(x, y)]_I =$

# Example 4.3.32

Let us use the interpretation $I$ given in example 4.3.19.
(Reminder $D = \{0, 1, 2\}$)

- $[\exists x\ \ell(x, x)]_I =$

$$= \max\{[\ell(0,0)]_I, [\ell(1,1)]_I, [\ell(2,2)]_I\} = \mathit{false}$$
$$= [\ell(0,0)]_I + [\ell(1,1)]_I + [\ell(2,2)]_I = \mathit{false} + \mathit{false} + \mathit{false} = \mathit{false}.$$

- $[\forall x \exists y\ \ell(x, y)]_I =$

$$\begin{aligned} = \min\{&\max\{[\ell(0,0)]_I,\ [\ell(0,1)]_I,\ [\ell(0,2)]_I\},\\ &\max\{[\ell(1,0)]_I,\ [\ell(1,1)]_I,\ [\ell(1,2)]_I\},\\ &\max\{[\ell(2,0)]_I,\ [\ell(2,1)]_I,\ [\ell(2,2)]_I\}\}\end{aligned}$$

$$= \min\{\max\{\mathit{false}, \mathit{true}, \mathit{false}\}, \max\{\mathit{true}, \mathit{false}, \mathit{false}\}, \max\{\mathit{true}, \mathit{false}, \mathit{false}\}\}$$

$$= \min\{\mathit{true}, \mathit{true}, \mathit{true}\} = \mathit{true}.$$

# Example 4.3.32

- $[\exists y \forall x \, \ell(x, y)]_I =$

# Example 4.3.32

- $[\exists y \forall x\ \ell(x,y)]_I =$

$$= [\ell(0,0)]_I.[\ell(1,0)]_I.[\ell(2,0)]_I + [\ell(0,1)]_I.[\ell(1,1)]_I.[\ell(2,1)]_I$$
$$+ [\ell(0,2)]_I.\ [\ell(1,2)]_I.\ [\ell(2,2)]_I$$
$$= \mathit{false}.\mathit{true}.\mathit{true} + \mathit{true}.\mathit{false}.\mathit{false} + \mathit{false}.\mathit{false}.\mathit{false}$$
$$= \mathit{false} + \mathit{false} + \mathit{false} = \mathit{false}.$$

# Example 4.3.32

- $[\exists y \forall x\ \ell(x,y)]_I =$

$$= [\ell(0,0)]_I.[\ell(1,0)]_I.[\ell(2,0)]_I + [\ell(0,1)]_I.[\ell(1,1)]_I.[\ell(2,1)]_I$$
$$+[\ell(0,2)]_I.\ [\ell(1,2)]_I.\ [\ell(2,2)]_I$$
$$= \mathit{false}.\mathit{true}.\mathit{true} + \mathit{true}.\mathit{false}.\mathit{false} + \mathit{false}.\mathit{false}.\mathit{false}$$
$$= \mathit{false} + \mathit{false} + \mathit{false} = \mathit{false}.$$

## Remark 4.3.33

The formulae $\forall x \exists y\ \ell(x,y)$ and $\exists y \forall x\ \ell(x,y)$ do not have the same value. Exchanging a $\exists$ and a $\forall$ does not preserve the truth value of a formula.

# Model, validity, consequence, equivalence

Defined as in propositional logic but...

## What's needed to evaluate a formula

- **In propositional logic**: an assignment $V \to \{0, 1\}$
- **In first-order logic**: $(I, e)$ where
  - $I$ is a symbol interpretation
  - $e$ a variable state.

# Model, validity, consequence, equivalence

Defined as in propositional logic but...

## What's needed to evaluate a formula

- **In propositional logic**: an assignment $V \to \{0, 1\}$
- **In first-order logic**: $(I, e)$ where
  - $I$ is a symbol interpretation
  - $e$ a variable state.

... we use an interpretation instead of an assignment.

# Model, validity, consequence, equivalence

Defined as in propositional logic but...

### What's needed to evaluate a formula

- **In propositional logic**: an assignment $V \to \{0, 1\}$
- **In first-order logic**: $(I, e)$ where
  - $I$ is a symbol interpretation
  - $e$ a variable state.

... we use an interpretation instead of an assignment.
The truth value of a formula only depends on

- the state of its free variables
- and the interpretation of its symbols.

# Overview

# Reminders about finite expansions

We look for models with $n$ elements **by reduction to the propositional case**
**Base case:** a formula with no function symbol and no constant, except integers less than $n$.

### Building the $n$-elements model

1. eliminate the quantifiers by expansion over a domain of $n$ elements,
2. replace equalities with their value
3. search for a propositional assignment of atomic formulae which is a model of the formula.

# Property of the $n$-expansion

**Theorem 4.3.41**

Let $A$ be a formula containing only integers $< n$.
Let $B$ be the $n$-expansion of $A$.
Any interpretation over the domain $\{0, \ldots, n-1\}$ assigns the same value to $A$ and $B$.

# Property of the *n*-expansion

**Theorem 4.3.41**

Let $A$ be a formula containing only integers $< n$.
Let $B$ be the $n$-expansion of $A$.
Any interpretation over the domain $\{0, \ldots, n-1\}$ assigns the same value to $A$ and $B$.

Proof : by induction on the height of formulae.

# Assignment VS interpretation

Let $A$ be a formula:

- closed,
- with no quantifier,
- with no equality nor function symbol,
- with no constant except the integers less than $n$.

Let $P$ be the set of atomic formulae in $A$ (except $\top$ and $\bot$).

**Theorem 4.3.42**

For any propositional assignment $v : P \to \{false, true\}$
there exists an interpretation $I$ of $A$ such that $[A]_I = [A]_v$.

# Assignment VS interpretation

Let $A$ be a formula:

- closed,
- with no quantifier,
- with no equality nor function symbol,
- with no constant except the integers less than $n$.

Let $P$ be the set of atomic formulae in $A$ (except $\top$ and $\bot$).

**Theorem 4.3.42**

For any propositional assignment $v : P \rightarrow \{false, true\}$
there exists an interpretation $I$ of $A$ such that $[A]_I = [A]_v$.

**Theorem 4.3.44**

For any interpretation $I$
there exists an assignment $v : P \rightarrow \{false, true\}$ such that $[A]_I = [A]_v$.

# Example 4.3.43

Let $v$ be the assignment defined by $[p(0)]_v = \mathit{true}$ and $[p(1)]_v = \mathit{false}$.

$v$ gives the value *false* to the formula $(p(0) + p(1)) \Rightarrow (p(0).p(1))$.

# Example 4.3.43

Let $v$ be the assignment defined by $[p(0)]_v = \mathit{true}$ and $[p(1)]_v = \mathit{false}$.

$v$ gives the value *false* to the formula $(p(0) + p(1)) \Rightarrow (p(0).p(1))$.

The interpretation $I$ defined by $p_I = \{0\}$ gives the same value to the same formulae.

This example shows that $v$ and $I$ are two analogous ways of presenting an interpretation.

# Correctness of the method

| | n-expansion | | simplifications | | | |
|---|---|---|---|---|---|---|
| $A$ | $\longrightarrow$ | $B$ | $\longrightarrow$ | $C$ | $\longrightarrow$ | $C$ |
| (1st order) | | (without $\forall\exists$) | | (without const.) | | (propos.) |

# Correctness of the method

$$\begin{array}{ccccccc} & n\text{-expansion} & & \text{simplifications} & & & \\ A & \longrightarrow & B & \longrightarrow & C & \longrightarrow & C \\ \text{(1st order)} & \equiv_n & \text{(without } \forall\exists\text{)} & & \text{(without const.)} & & \text{(propos.)} \end{array}$$

- $[A]_I = [B]_I$ for any $I$ over a domain of $n$ elements

# Correctness of the method

$$\begin{array}{ccccccc} & n\text{-expansion} & & \text{simplifications} & & & \\ A & \longrightarrow & B & \longrightarrow & C & \longrightarrow & C \\ \text{(1st order)} & & \text{(without } \forall\exists\text{)} & \equiv & \text{(without const.)} & & \text{(propos.)} \end{array}$$

- $[A]_I = [B]_I$ for any $I$ over a domain of $n$ elements
- $B \equiv C$ by construction (hence $[B]_I = [C]_I$ for any $I$)

# Correctness of the method

$$\begin{array}{ccccccc} & n\text{-expansion} & & \text{simplifications} & & & \\ A & \longrightarrow & B & \longrightarrow & C & \longrightarrow & C \\ \text{(1st order)} & & \text{(without } \forall\exists\text{)} & & \text{(without const.)} & \approxeq & \text{(propos.)} \end{array}$$

- $[A]_I = [B]_I$ for any $I$ over a domain of $n$ elements
- $B \equiv C$ by construction (hence $[B]_I = [C]_I$ for any $I$)
- 
  - For any $v$ there is an $I$ such that $[C]_I = [C]_v$.
  - For any $I$ there is a $v$ such that $[C]_I = [C]_v$.

# Correctness of the method

| | $n$-expansion | | simplifications | | | |
|---|---|---|---|---|---|---|
| $A$ | $\longrightarrow$ | $B$ | $\longrightarrow$ | $C$ | $\longrightarrow$ | $C$ |
| (1st order) | | (without $\forall\exists$) | | (without const.) | | (propos.) |

- $[A]_I = [B]_I$ for any $I$ over a domain of $n$ elements
- $B \equiv C$ by construction (hence $[B]_I = [C]_I$ for any $I$)
- 
  - For any $v$ there is an $I$ such that $[C]_I = [C]_v$.
  - For any $I$ there is a $v$ such that $[C]_I = [C]_v$.

Thus $A$ has a model $I$ over a domain of $n$ elements
if and only if
$C$ has a model $v$ (and we can find $I$ from $v$ if need be).

# Finding a finite model of a closed formula **with** a function symbol

Let $A$ be a closed formula which can contain integers of value less than $n$.

**Procedure**

- Replace $A$ by its expansion
- Enumerate the choices of symbol values, by propagating as much as possible each of the realized choices.

Similar to `DPLL` *algorithm*.

# Example 4.3.46 : $A = \exists y P(y) \Rightarrow P(a)$

Look for a counter-model with 2 elements.

# Example 4.3.46 : $A = \exists y P(y) \Rightarrow P(a)$

Look for a counter-model with 2 elements.

**2-expansion of $A$**

# Example 4.3.46 : $A = \exists y P(y) \Rightarrow P(a)$

Look for a counter-model with 2 elements.

**2-expansion of $A$**

$$P(0) + P(1) \Rightarrow P(a)$$

Find the values of $P(0)$, $P(1)$, $a$.
We (arbitrarily) choose $a = 0$.

# Example 4.3.46 : $A = \exists y P(y) \Rightarrow P(a)$

Look for a counter-model with 2 elements.

**2-expansion of $A$**

$$P(0) + P(1) \Rightarrow P(a)$$

Find the values of $P(0)$, $P(1)$, $a$.
We (arbitrarily) choose $a = 0$.

$$P(0) + P(1) \Rightarrow P(0)$$

# Example 4.3.46 : $A = \exists y P(y) \Rightarrow P(a)$

Look for a counter-model with 2 elements.

**2-expansion of $A$**

$$P(0) + P(1) \Rightarrow P(a)$$

Find the values of $P(0)$, $P(1)$, $a$.
We (arbitrarily) choose $a = 0$.

$$P(0) + P(1) \Rightarrow P(0)$$

$P(0) \mapsto$ *false*, $P(1) \mapsto$ *true* is a propositional counter-model,
we deduce an interpretation $I$ such that $P_I = \{1\}$.

# Example 4.3.46 : $A = \exists y P(y) \Rightarrow P(a)$

Look for a counter-model with 2 elements.

**2-expansion of $A$**

$$P(0) + P(1) \Rightarrow P(a)$$

Find the values of $P(0)$, $P(1)$, $a$.
We (arbitrarily) choose $a = 0$.

$$P(0) + P(1) \Rightarrow P(0)$$

$P(0) \mapsto \mathit{false}, P(1) \mapsto \mathit{true}$ is a propositional counter-model,
we deduce an interpretation $I$ such that $P_I = \{1\}$.

A counter-model is $I$ over domain $\{0, 1\}$ such that $P_I = \{1\}$ and $a_I = 0$.

Example 4.3.47 : $P(a), \forall x(P(x) \Rightarrow P(f(x))), \neg P(f(b))$

1. 2-expansion:

Example 4.3.47 : $P(a), \forall x(P(x) \Rightarrow P(f(x))), \neg P(f(b))$

1. 2-expansion:
   $F = \{P(a), (P(0) \Rightarrow P(f(0))).(P(1) \Rightarrow P(f(1))), \neg P(f(b))\}$.
2. Find values for $P(0)$, $P(1)$, $a$, $b$, $f(0)$ and $f(1)$ which provide a model of $F$.

# Example 4.3.47 : $P(a), \forall x(P(x) \Rightarrow P(f(x))), \neg P(f(b))$

1. 2-expansion:
   $F = \{P(a), (P(0) \Rightarrow P(f(0))).(P(1) \Rightarrow P(f(1))), \neg P(f(b))\}$.
2. Find values for $P(0)$, $P(1)$, $a$, $b$, $f(0)$ and $f(1)$ which provide a model of $F$.
3. Let us choose $a = 0$

# Example 4.3.47 : $P(a), \forall x(P(x) \Rightarrow P(f(x))), \neg P(f(b))$

1. 2-expansion:
   $F = \{P(a), (P(0) \Rightarrow P(f(0))).(P(1) \Rightarrow P(f(1))), \neg P(f(b))\}$.
2. Find values for $P(0)$, $P(1)$, $a$, $b$, $f(0)$ and $f(1)$ which provide a model of $F$.
3. Let us choose $a = 0$
   - From $P(a) = \mathit{true}$ and $a = 0$, we deduce: $P(0) = \mathit{true}$

# Example 4.3.47 : $P(a), \forall x(P(x) \Rightarrow P(f(x))), \neg P(f(b))$

1. 2-expansion:
   $F = \{P(a), (P(0) \Rightarrow P(f(0))).(P(1) \Rightarrow P(f(1))), \neg P(f(b))\}$.
2. Find values for $P(0)$, $P(1)$, $a$, $b$, $f(0)$ and $f(1)$ which provide a model of $F$.
3. Let us choose $a = 0$
   - From $P(a) = \mathit{true}$ and $a = 0$, we deduce: $P(0) = \mathit{true}$
   - From $P(0) = \mathit{true}$ and $(P(0) \Rightarrow P(f(0))) = \mathit{true}$, we deduce: $P(f(0)) = \mathit{true}$

# Example 4.3.47 : $P(a), \forall x(P(x) \Rightarrow P(f(x))), \neg P(f(b))$

1. 2-expansion:
   $F = \{P(a), (P(0) \Rightarrow P(f(0))).(P(1) \Rightarrow P(f(1))), \neg P(f(b))\}$.
2. Find values for $P(0)$, $P(1)$, $a$, $b$, $f(0)$ and $f(1)$ which provide a model of $F$.
3. Let us choose $a = 0$
   - From $P(a) = true$ and $a = 0$, we deduce: $P(0) = true$
   - From $P(0) = true$ and $(P(0) \Rightarrow P(f(0))) = true$, we deduce: $P(f(0)) = true$
   - From $P(f(b)) = false$ and $P(f(0)) = true$, we deduce $f(0) \neq f(b)$ therefore $b \neq 0$, hence: $b = 1$ and $P(f(1)) = false$.

# Example 4.3.47 : $P(a), \forall x(P(x) \Rightarrow P(f(x))), \neg P(f(b))$

1. 2-expansion:
   $F = \{P(a), (P(0) \Rightarrow P(f(0))).(P(1) \Rightarrow P(f(1))), \neg P(f(b))\}$.
2. Find values for $P(0)$, $P(1)$, $a$, $b$, $f(0)$ and $f(1)$ which provide a model of $F$.
3. Let us choose $a = 0$
   - From $P(a) = \mathit{true}$ and $a = 0$, we deduce: $P(0) = \mathit{true}$
   - From $P(0) = \mathit{true}$ and $(P(0) \Rightarrow P(f(0))) = \mathit{true}$, we deduce: $P(f(0)) = \mathit{true}$
   - From $P(f(b)) = \mathit{false}$ and $P(f(0)) = \mathit{true}$, we deduce $f(0) \neq f(b)$ therefore $b \neq 0$, hence: $b = 1$ and $P(f(1)) = \mathit{false}$.
   - From $P(f(1)) = \mathit{false}$ and $P(0) = \mathit{true}$, we deduce $f(1) \neq 0$ hence: $f(1) = 1$ and $P(1) = \mathit{false}$

# Example 4.3.47 : $P(a), \forall x(P(x) \Rightarrow P(f(x))), \neg P(f(b))$

1. 2-expansion:
   $F = \{P(a), (P(0) \Rightarrow P(f(0))).(P(1) \Rightarrow P(f(1))), \neg P(f(b))\}$.
2. Find values for $P(0)$, $P(1)$, $a$, $b$, $f(0)$ and $f(1)$ which provide a model of $F$.
3. Let us choose $a = 0$
   - From $P(a) = true$ and $a = 0$, we deduce: $P(0) = true$
   - From $P(0) = true$ and $(P(0) \Rightarrow P(f(0))) = true$, we deduce: $P(f(0)) = true$
   - From $P(f(b)) = false$ and $P(f(0)) = true$, we deduce $f(0) \neq f(b)$ therefore $b \neq 0$, hence: $b = 1$ and $P(f(1)) = false$.
   - From $P(f(1)) = false$ and $P(0) = true$, we deduce $f(1) \neq 0$ hence: $f(1) = 1$ and $P(1) = false$
   - From $P(f(0)) = true$ and $P(1) = false$, we deduce: $f(0) = 0$

# Example 4.3.47 : $P(a), \forall x(P(x) \Rightarrow P(f(x))), \neg P(f(b))$

1. 2-expansion:
   $F = \{P(a), (P(0) \Rightarrow P(f(0))).(P(1) \Rightarrow P(f(1))), \neg P(f(b))\}$.
2. Find values for $P(0)$, $P(1)$, $a$, $b$, $f(0)$ and $f(1)$ which provide a model of $F$.
3. Let us choose $a = 0$
   - From $P(a) = true$ and $a = 0$, we deduce: $P(0) = true$
   - From $P(0) = true$ and $(P(0) \Rightarrow P(f(0))) = true$, we deduce: $P(f(0)) = true$
   - From $P(f(b)) = false$ and $P(f(0)) = true$, we deduce $f(0) \neq f(b)$ therefore $b \neq 0$, hence: $b = 1$ and $P(f(1)) = false$.
   - From $P(f(1)) = false$ and $P(0) = true$, we deduce $f(1) \neq 0$ hence: $f(1) = 1$ and $P(1) = false$
   - From $P(f(0)) = true$ and $P(1) = false$, we deduce: $f(0) = 0$
4. Model: $a = 0, b = 1, P = \{0\}, f(0) = 0, f(1) = 1$

# William McCune (1953-2011)

- Author of several automated reasoning systems: Otter, Prover9, Mace4

## MACE

- **expansion** of first-order formulas
- **efficient algorithms** such as DPLL

http://www.cs.unm.edu/~mccune/mace4/examples/2009-11A/mace4-misc/

# William McCune (1953-2011)

- Author of several automated reasoning systems: Otter, Prover9, Mace4

**MACE**

- **expansion** of first-order formulas
- **efficient algorithms** such as `DPLL`

`http://www.cs.unm.edu/~mccune/mace4/examples/2009-11A/mace4-misc/`

- 1996 : Proof of the Robbins conjecture using the automated theorem prover EQP
  - 8 days of computation on a 66 MHz processor, 30 Mo of memory
  - production of a proof witness by Otter, in turn checked by a third program

(Undecided conjecture since 1933)

# Overview

# Substitution at the propositional level

Recall that substituting a propositional variable in a valid formula yields another valid formula. This extends to first-order logic.

**Example:**

Let $\sigma(p) = \forall x\ q(x)$.
$p \vee \neg p$ is valid, the same holds for

$$\sigma(p \vee \neg p) = \forall x\ q(x) \vee \neg \forall x\ q(x)$$

# Substitution at the propositional level

Recall that substituting a propositional variable in a valid formula yields another valid formula. This extends to first-order logic.

**Example:**

Let $\sigma(p) = \forall x\, q(x)$.
$p \vee \neg p$ is valid, the same holds for

$$\sigma(p \vee \neg p) = \forall x\, q(x) \vee \neg \forall x\, q(x)$$

The replacement principle extends to first-order logic as well since:

For any formulae $A$ and $B$ and any variable $x$ :

- $(A \Leftrightarrow B) \models (\forall x A \Leftrightarrow \forall x B)$
- $(A \Leftrightarrow B) \models (\exists x A \Leftrightarrow \exists x B)$

# Instantiation of a variable in a term

**Definition 4.3.34**

$A < x := t >$ is the formula obtained by replacing in $A$ every free occurrence of $x$ with the term $t$.

# Instantiation of a variable in a term

## Definition 4.3.34

$A < x := t >$ is the formula obtained by replacing in $A$ every free occurrence of $x$ with the term $t$.

## Example 4.3.35

Let $A$ be the formula $(\forall x P(x) \vee Q(\mathbf{x}))$, the formula $A < x := b >$ is

$(\forall x P(x) \vee Q(b))$ since only the bold occurrence of $x$ is free.

# Instantiation of a variable in a term

### Definition 4.3.34

$A < x := t >$ is the formula obtained by replacing in $A$ every free occurrence of $x$ with the term $t$.

### Example 4.3.35

Let $A$ be the formula $(\forall x P(x) \vee Q(\mathbf{x}))$, the formula $A < x := b >$ is

$(\forall x P(x) \vee Q(b))$ since only the bold occurrence of $x$ is free.

But one cannot substitute any variable with anything:

### Example 4.3.37

Let $A$ be the formula $\exists y p(x, y)$.

# Instantiation of a variable in a term

**Definition 4.3.34**

$A < x := t >$ is the formula obtained by replacing in $A$ every free occurrence of $x$ with the term $t$.

**Example 4.3.35**

Let $A$ be the formula $(\forall x P(x) \vee Q(\mathbf{x}))$, the formula $A < x := b >$ is

$(\forall x P(x) \vee Q(b))$ since only the bold occurrence of $x$ is free.

But one cannot substitute any variable with anything:

**Example 4.3.37**

Let $A$ be the formula $\exists y p(x, y)$.

- $A < x := y > =$

# Instantiation of a variable in a term

## Definition 4.3.34

$A < x := t >$ is the formula obtained by replacing in $A$ every free occurrence of $x$ with the term $t$.

## Example 4.3.35

Let $A$ be the formula $(\forall x P(x) \vee Q(\mathbf{x}))$, the formula $A < x := b >$ is $(\forall x P(x) \vee Q(b))$ since only the bold occurrence of $x$ is free.

But one cannot substitute any variable with anything:

## Example 4.3.37

Let $A$ be the formula $\exists y p(x, y)$.

- $A < x := y >= \boxed{\exists y p(y, y)}$

# Instantiation of a variable in a term

**Definition 4.3.34**

$A < x := t >$ is the formula obtained by replacing in $A$ every free occurrence of $x$ with the term $t$.

**Example 4.3.35**

Let $A$ be the formula $(\forall x P(x) \vee Q(\mathbf{x}))$, the formula $A < x := b >$ is

$(\forall x P(x) \vee Q(b))$ since only the bold occurrence of $x$ is free.

But one cannot substitute any variable with anything:

**Example 4.3.37**

Let $A$ be the formula $\exists y p(x, y)$.

- $A < x := y >= \boxed{\exists y p(y, y)}$ (capture phenomenon)

# Capture changes the meaning of a formula

## Example 4.3.37

Let $p$ be a binary relation interpreted over $\{0, 1\}$ as $p_I = \{(0, 1)\}$
Let $e$ be a state where $y = 0$.

# Capture changes the meaning of a formula

## Example 4.3.37

Let $p$ be a binary relation interpreted over $\{0, 1\}$ as $p_I = \{(0, 1)\}$
Let $e$ be a state where $y = 0$.

- $[A < x := y >]_{(I,e)} =$

# Capture changes the meaning of a formula

## Example 4.3.37

Let $p$ be a binary relation interpreted over $\{0,1\}$ as $p_I = \{(0,1)\}$
Let $e$ be a state where $y = 0$.

- $[A < x := y >]_{(I,e)} =$

  $[\exists y p(y,y)]_{(I,e)} = [p(0,0)]_{(I,e)} + [p(1,1)]_{(I,e)} = \mathit{false} + \mathit{false} = \mathit{false}.$

# Capture changes the meaning of a formula

## Example 4.3.37

Let $p$ be a binary relation interpreted over $\{0, 1\}$ as $p_I = \{(0, 1)\}$
Let $e$ be a state where $y = 0$.

- $[A < x := y >]_{(I,e)} =$

  $[\exists y p(y, y)]_{(I,e)} = [p(0, 0)]_{(I,e)} + [p(1, 1)]_{(I,e)} = \mathit{false} + \mathit{false} = \mathit{false}.$
- Let $d = 0$.
  In the assignment $(I, e[x = d])$, we have $x = 0$.
  Hence $[A]_{(I,e[x=d])} =$

# Capture changes the meaning of a formula

## Example 4.3.37

Let $p$ be a binary relation interpreted over $\{0,1\}$ as $p_I = \{(0,1)\}$
Let $e$ be a state where $y = 0$.

- $[A < x := y >]_{(I,e)} =$

  $$[\exists y p(y,y)]_{(I,e)} = [p(0,0)]_{(I,e)} + [p(1,1)]_{(I,e)} = false + false = false.$$

- Let $d = 0$.
  In the assignment $(I, e[x = d])$, we have $x = 0$.
  Hence $[A]_{(I,e[x=d])} =$

  $$[\exists y p(x,y)]_{(I,e[x=d])} = [p(0,0)]_{(I,e)} + [p(0,1)]_{(I,e)} = false + true = true.$$

# Capture changes the meaning of a formula

## Example 4.3.37

Let $p$ be a binary relation interpreted over $\{0,1\}$ as $p_I = \{(0,1)\}$
Let $e$ be a state where $y = 0$.

- $[A < x := y >]_{(I,e)} =$

  $$[\exists y p(y,y)]_{(I,e)} = [p(0,0)]_{(I,e)} + [p(1,1)]_{(I,e)} = false + false = false.$$

- Let $d = 0$.
  In the assignment $(I, e[x = d])$, we have $x = 0$.
  Hence $[A]_{(I,e[x=d])} =$

  $$[\exists y p(x,y)]_{(I,e[x=d])} = [p(0,0)]_{(I,e)} + [p(0,1)]_{(I,e)} = false + true = true.$$

Thus, $[A < x := y >]_{(I,e)} \neq [A]_{(I,e[x=d])}$, for $d = [\![y]\!]_{(I,e)}$.

# Precautions for the instantiation of a variable in a term

Solution: notion of a term $t$ free for a variable

**Definition 4.3.34**

2. The term $t$ is free for $x$ in $A$ if the variables of $t$ are not bound in the free occurrences of $x$.

# Precautions for the instantiation of a variable in a term

Solution: notion of a term $t$ free for a variable

**Definition 4.3.34**

2. The term $t$ is free for $x$ in $A$ if the variables of $t$ are not bound in the free occurrences of $x$.

**Example 4.3.35**

- The term $f(z)$ is free for $x$ in formula $\exists y\, p(x, y)$.

# Precautions for the instantiation of a variable in a term

Solution: notion of a term $t$ free for a variable

**Definition 4.3.34**

2. The term $t$ is free for $x$ in $A$ if the variables of $t$ are not bound in the free occurrences of $x$.

**Example 4.3.35**

- The term $f(z)$ is free for $x$ in formula $\exists y\, p(x, y)$.
- On the opposite, the terms $y$ or $g(y)$ are not free for $x$ in this formula.

# Precautions for the instantiation of a variable in a term

Solution: notion of a term $t$ free for a variable

**Definition 4.3.34**

2. The term $t$ is free for $x$ in $A$ if the variables of $t$ are not bound in the free occurrences of $x$.

**Example 4.3.35**

- The term $f(z)$ is free for $x$ in formula $\exists y\, p(x, y)$.
- On the opposite, the terms $y$ or $g(y)$ are not free for $x$ in this formula.
- By definition, the term $x$ is free for $x$ in any formula.

# Properties

**Theorem 4.3.36**

Let $A$ be a formula and $t$ a free term for the variable $x$ in $A$.
For any assignment $(I, e)$ we have
$[A < x := t >]_{(I,e)} = [A]_{(I,e[x=d])}$ where $d = [\![t]\!]_{(I,e)}$.

# Properties

## Theorem 4.3.36

Let $A$ be a formula and $t$ a free term for the variable $x$ in $A$.
For any assignment $(I, e)$ we have
$[A < x := t >]_{(I,e)} = [A]_{(I,e[x=d])}$ where $d = [\![t]\!]_{(I,e)}$.

## Corollary 4.3.38

Let $A$ be a formula and $t$ a free term for $x$ in $A$.
The formulae $\forall xA \Rightarrow A < x := t >$ and $A < x := t > \Rightarrow \exists xA$ are valid.

# Overview

# Relation between $\forall$ and $\exists$

**Lemma 4.4.1**

Let $A$ be a formula and $x$ be a variable.

1. $\neg\forall xA \equiv \exists x\neg A$
2. $\forall xA \equiv \neg\exists x\neg A$
3. $\neg\exists xA \equiv \forall x\neg A$
4. $\exists xA \equiv \neg\forall x\neg A$

Let us prove the first two equivalences, the other are in exercise 78

# Proof of $\neg\forall x A \equiv \exists x \neg A$

Let us evaluate $[\neg\forall x A]_{(I,e)}$

# Proof of $\neg\forall x A \equiv \exists x \neg A$

Let us evaluate $[\neg\forall x A]_{(I,e)}$
$= 1 - [\forall x A]_{(I,e)}$

# Proof of $\neg\forall x A \equiv \exists x \neg A$

Let us evaluate $[\neg\forall x A]_{(I,e)}$
$= 1 - [\forall x A]_{(I,e)}$
$= 1 - \min_{d\in D}[A]_{(I,e[x=d])}$

# Proof of $\neg\forall xA \equiv \exists x\neg A$

Let us evaluate $[\neg\forall xA]_{(I,e)}$
$= 1 - [\forall xA]_{(I,e)}$
$= 1 - \min_{d\in D}[A]_{(I,e[x=d])}$
$= \max_{d\in D}(1 - [A]_{(I,e[x=d])})$

# Proof of $\neg\forall xA \equiv \exists x\neg A$

Let us evaluate $[\neg\forall xA]_{(I,e)}$
$= 1 - [\forall xA]_{(I,e)}$
$= 1 - \min_{d\in D}[A]_{(I,e[x=d])}$
$= \max_{d\in D}(1 - [A]_{(I,e[x=d])})$
$= \max_{d\in D}[\neg A]_{(I,e[x=d])}$

# Proof of $\neg\forall x A \equiv \exists x \neg A$

Let us evaluate $[\neg\forall x A]_{(I,e)}$

$= 1 - [\forall x A]_{(I,e)}$

$= 1 - \min_{d\in D}[A]_{(I,e[x=d])}$

$= \max_{d\in D}(1 - [A]_{(I,e[x=d])})$

$= \max_{d\in D}[\neg A]_{(I,e[x=d])}$

$= [\exists x \neg A]_{(I,e)}$

# Proof of $\neg\forall x A \equiv \exists x \neg A$

Let us evaluate $[\neg\forall x A]_{(I,e)}$
$= 1 - [\forall x A]_{(I,e)}$
$= 1 - \min_{d\in D}[A]_{(I,e[x=d])}$
$= \max_{d\in D}(1 - [A]_{(I,e[x=d])})$
$= \max_{d\in D}[\neg A]_{(I,e[x=d])}$
$= [\exists x \neg A]_{(I,e)}$

Proof of $\forall x A \equiv \neg\exists x \neg A$:

Let us evaluate $\forall x A$
$\equiv \neg\neg\forall x A$
$\equiv \neg\exists x \neg A$ (see above)

# Moving quantifiers

Let $x$, $y$ be two variables and $A$, $B$ be two formulae.

1. $\forall x \forall y A \equiv \forall y \forall x A$
2. $\exists x \exists y A \equiv \exists y \exists x A$

# Moving quantifiers

Let $x$, $y$ be two variables and $A$, $B$ be two formulae.

1. $\forall x \forall y A \equiv \forall y \forall x A$
2. $\exists x \exists y A \equiv \exists y \exists x A$
3. $\forall x (A \wedge B) \equiv (\forall x A \wedge \forall x B)$
4. $\exists x (A \vee B) \equiv (\exists x A \vee \exists x B)$

# Moving quantifiers

Let $x$, $y$ be two variables and $A$, $B$ be two formulae.

1. $\forall x \forall y A \equiv \forall y \forall x A$
2. $\exists x \exists y A \equiv \exists y \exists x A$
3. $\forall x (A \wedge B) \equiv (\forall x A \wedge \forall x B)$
4. $\exists x (A \vee B) \equiv (\exists x A \vee \exists x B)$
5. Let $Q$ be a quantifier and let $\circ$ be a connective among $\wedge, \vee$.
   If $x$ is not a free variable of $A$ then:
   - 5.1 $Qx\ A \equiv A$,
   - 5.2 $Qx(A \circ B) \equiv A \circ QxB$

# Example 4.4.2

Let us eliminate useless quantifiers from these two formulae:

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

  $\exists x P(x)$

- $\forall x (\exists x P(x) \vee Q(x)) \equiv$

  $\exists x P(x) \vee \forall x Q(x)$

# Renaming of bound variables (1/3)

**Theorem 4.4.3**

Let $Q$ be a quantifier.
If $y$ does not occur in $Qx\ A$ then: $Qx\ A \equiv Qy\ A < x := y >$.

# Renaming of bound variables (1/3)

**Theorem 4.4.3**

Let $Q$ be a quantifier.
If $y$ does not occur in $Qx\ A$ then: $Qx\ A \equiv Qy\ A < x := y >$.

**Example 4.4.4**

- $\forall x\ p(x,z) \equiv \forall y\ p(y,z)$
- $\forall x\ p(x,z) \not\equiv \forall z\ p(z,z)$

# Overview

# Today

- To evaluate a formula = to choose an interpretation for its **symbols** and a state for its **variables**
- Method for finding **(counter-)model** by finite interpretation and expansion
- Important equivalences about quantifiers
(beware, **no usable notion of normal form**)

# Next time

- Skolemisation
- Semi-algorithm to prove that a formula is unsatisfiable.

# Next time

- Skolemisation
- Semi-algorithm to prove that a formula is unsatisfiable.

**Homework**

*Every man is mortal.*
*Socrates is a man.*
*Hence Socrates is mortal.*

- Look for a counter-model using 1-expansion then 2-expansion.
- What is your conclusion ?